Лабораторная работа 1.

Введение в R. Методы первичного разведочного анализа данных в R

1. Цель лабораторной работы

- получить навыки применения методов первичного разведочного анализа данных и статистической проверки гипотез для решения задач АД;
- получить базовые навыки работы в среде R;
- изучить средства R для проведения первичного разведочного анализа данных (методы визуализации, описательной статистики, статистической проверки гипотез, корреляционного анализа данных) на примере решения конкретной задачи ИАД (интеллектуального анализа данных).

2. Задание к лабораторной работе

Прочитайте содержательную постановку задачи для вашего варианта. Выберите файл с данными (в формате .csv), соответствующий вашему варианту.

1. Загрузите файл с данными, соответствующий варианту.
2. Просмотрите загруженную таблицу с данными. Попробуйте использовать команды R для доступа к определенному столбцу/строке данных, редактирования данных, получения подвыборок из данных в соответствии с заданными условиями.
3. Посмотрите структуру данных. Рассчитайте основные статистические характеристики по количественным данным (минимальное, максимальное, среднее значение, стандартное отклонение, первый и третий квартили, медиана, мода, асимметрия, эксцесс) отдельно для первой и второй группы и для всей выборки. Сделайте выводы.
4. Проведите графический анализ данных, постройте:
   - диаграмму рассеяния по двум количественным признакам;
   - радиальную диаграмму по качественному признаку;
   - категориальную радиальную диаграмму по одному из качественных признаков в зависимости от пола и группы;
   - категориальную столбиковую диаграмму по одному из количественных признаков в зависимости от пола и группы;
   - диаграмму размаха для одного из количественных признаков в зависимости от значений пола или группы;
   - гистограммы для всех количественных признаков (отдельный график для каждого признака, графики построены в одном графическом окне);
   - матричный график по всем количественным признакам.

На основе проведенного анализа сделайте выводы о структуре данных, о характере распределения данных в терминах решаемой задачи.

5. Проверьте гипотезу о соответствии данных модели нормального закона распределения для одного из количественных признаков (отдельно для первой и второй группы) на основе критериев Шапиро – Уилка, Крамера – Мизеса, Андерсона – Дарлинга. Сделайте выводы в терминах прикладной задачи.
6. Проведите корреляционный анализ данных.

6.1. Оцените степень взаимосвязи между качественными переменными на основе критериев χ^2 (Chi-квадрат) и Фишера для первой и второй группы. Сделайте выводы о силе и направлении связи в терминах решаемой задачи.

6.2. Оцените степень взаимосвязи между одной из качественных переменных и количественными переменными на основе использования однофакторного дисперсионного анализа (ANOVA) и критерия Краскела-Уоллиса.

6.3. Оцените степень взаимосвязи между количественными переменными на основе расчета коэффициентов корреляции Пирсона, Спирмена, Кендалла для первой и второй группы.

6.4. Оцените степень взаимосвязи между двумя количественными переменными (для которых коэффициент корреляции Пирсона максимален по модулю) на основе расчета частного коэффициента корреляции для первой и второй группы.

6.5. Графически представьте матрицы коэффициентов корреляции с наложением тепловой карты. Оцените статистическую значимость связи. Сделайте выводы о силе и направлении связи в терминах решаемой задачи. Постройте матричный график с помощью функций пакета ggpairs.

7. Сделайте выводы по работе в произвольной форме.

3. Методические указания к лабораторной работе

R – это специализированный язык программирования и среда для решения задач статистической обработки данных. R работает во всех основных операционных системах и поддерживает тысячи специализированных модулей, библиотек и утилит. Все это делает R удобным средством для извлечения полезной информации из гор сырых данных.

R можно бесплатно скачать из сетевого архива R (Comprehensive R Archive Network, CRAN) по адресу http://cran.r-project.org. Предварительно скомпилированные загрузочные файлы доступны для Linux, macOS (ранее Mac OS X) и Windows.

Удобная интегрированная среда разработки для R – RStudio, которая наиболее широко используется в настоящее время. Чтобы установить RStudio, необходимо загрузить последнюю версию установщика для вашей платформы с сайта RStudio (https://rstudio.com/products/rstudio/download/). Версию RStudio Desktop с открытым исходным кодом можно скачать и использовать бесплатно. Новые версии R и RStudio выпускаются каждые несколько месяцев. Обратите внимание, что RStudio работает с любой версией R, которую вы установили, поэтому обновление до последней версии RStudio не обновляет вашу версию R. R необходимо обновлять отдельно.

R – это чувствительный к регистру клавиатуры интерпретируемый язык программирования. Вы можете либо вводить команды по одной в ответ на приглашение на ввод команды (>), либо запускать набор команд из исходного файла. Типы данных очень разнообразны: векторы, матрицы, таблицы данных и списки (совокупность нескольких объектов). Основной функционал программы реализуется при помощи встроенных и создаваемых пользователем функций, и все объекты (наборы данных) хранятся в памяти программы до завершения интерактивной сессии. Основные функции доступны по умолчанию. Остальные функции содержатся в пакетах, которые могут активироваться в ходе работы с программой по мере необходимости.

Командные строки состоят из функций и присвоений. R использует символ <- для присвоений вместо обычного знака равенства. Например, командная строка x <- rnorm(10) создает объект типа вектор, который называется x и содержит десять случайных элементов нормального распределения.

Также в R можно использовать знак равенства для присвоения, но этот синтаксис поддерживается только некоторыми функциями и не является стандартным.

Комментарии в R начинаются с символа #.

R в базовой установке уже обладает обширными возможностями. Однако некоторые наиболее впечатляющие опции программы реализованы в дополнительных модулях, которые можно скачать и установить. По состоянию на сентябрь 2025 года существует более 22600 созданных пользователями модулей, называемых *пакетами* (packages), которые вы можете скачать с http://cran.r-project.org/web/packages и пакеты добавляются ежедневно.

Пакеты – это собрания функций R, данных и скомпилированного программного кода в определенном формате. Директория, в которой пакеты хранятся на вашем компьютере, называется библиотекой. Функция .libPath() показывает, где расположена ваша библиотека, а функция library() выводит на экран названия всех имеющихся в библиотеке пакетов.

R поставляется уже со стандартным набором пакетов (включая base, datasets, utils, grDevices, graphics и methods). В них уже содержатся разнообразные функции и наборы данных, доступных по умолчанию. Другие пакеты нужно скачивать и устанавливать. После установки они загружаются во время сессии по мере необходимости. Команда search() выводит на экран названия загруженных и готовых к использованию пакетов.

В R существует множество функций, предназначенных для управления пакетами. Для установки пакета используйте команду
>install.packages()

Эта команда, введенная без аргументов, вызовет список зеркал сайта CRAN. Выберите ближайший к вам сайт или самый первый, указанный в верхней части списка (0-Cloud), который, как правило, хорошо работает для большинства местоположений. После выбора зеркала вы увидите список всех доступных пакетов. Выберите один из них, и он будет скачан и загружен.

Если вы знаете название пакета, который хотите установить, то сообщите функции это название в виде аргумента. Например, пакет gclus содержит функции для создания усовершенствованных диаграмм рассеяния. Этот пакет можно скачать и установить при помощи команды install.packages("gclus"). Пакет нужно установить только один раз. Однако, как и любые другие программы, пакеты часто обновляются их разработчиками. Используйте команду update.package() для обновления всех установленных пакетов. Для получения подробной информации об установленных пакетах можно использовать команду installed.packages(). Она выводит на экран список всех имеющихся пакетов с номерами их версий, названиями пакетов, от которых они зависят, и другой информацией.

Когда вы устанавливаете пакет, он скачивается с сайта CRAN и загружается в вашу библиотеку. Для использования этого пакета в текущей сессии программы вам нужно загрузить его при помощи команды library(). Например, для того чтобы использовать пакет gclus, введите команду:

```
> library(gclus).
```

Задав значение переменной .libPaths, можно сформировать локальную библиотеку пакетов вне системы директорий R. Это может пригодиться для создания резервных копий при обновлении пакетов.

Пример:

```
>.libPaths("C:\\Users\\AppData\\Local\\Temp\\RtmpqiArKq\\downloaded_packages")
```

R обладает обширными справочными материалами. Встроенная система помощи содержит подробные разъяснения, ссылки на литературу и примеры для каждой функции из установленных пакетов. Справку можно вызвать при помощи функций, перечисленных в табл. 1.

Таблица 1. – Функции справки

Функция	Действие
`help.start()`	Общая справка
`help ("нечто")` или `?нечто`	Справка по функции *нечто* (кавычки необязательны)
`help.search("нечто")` или `??нечто`	Поиск в справке записей, содержащих *нечто*
`example("нечто")`	Примеры использования функции *нечто* (кавычки необязательны)
`RSiteSearch("нечто")`	Поиск записей, содержащих *нечто* в онлайн-руководствах и заархивированных рассылках
`apropos("нечто", mode="function")`	Список всех доступных функций, в названии которых есть *нечто*
`data()`	Список всех демонстрационных данных, содержащихся в загруженных пакетах
`vignette()`	Список всех доступных руководств по загруженным пакетам
`vignette("нечто")`	Список руководств по теме *нечто*

1. Пояснения к пп. 1-3.

Рабочее пространство – это текущая рабочая среда R в памяти вашего компьютера, которая включает в себя любые созданные пользователем объекты (векторы, матрицы, функции, таблицы данных или списки). В конце каждой сессии вы можете сохранить рабочее пространство, и оно автоматически загрузится при следующем запуске программы. Команды интерактивно вводятся в ответ на приглашение к их вводу. Можно использовать стрелки вверх и вниз для перемещения между введенными ранее командами. Это позволяет вызвать предыдущую команду, отредактировать ее и вновь выполнить ее, нажав клавишу **Enter**. Текущая рабочая директория – это та директория, где находятся файлы данных и куда по умолчанию сохраняются результаты. Функция **getwd()** позволяет узнать, какая директория в данный момент является рабочей.

Вы можете назначить рабочую директорию при помощи функции:
setwd("C:\\project1")

Работа в R начинается, как правило, с установки рабочей директории.

Если появляется необходимость импортировать файл, который находится не в рабочей директории, нужно написать полный путь к нему. Всегда заключайте в кавычки названия файлов и директорий. Некоторые стандартные команды для управления рабочим пространством приведены в табл. 2.

Таблица 2. – Функции управления рабочим пространством

Функция	Действие
`getwd()`	Вывести на экран название текущей рабочей директории
`setwd("моя_директория")`	Назначить *моя_директория* текущей рабочей директорией
`ls()`	Вывести на экран список объектов в текущем рабочем пространстве
`rm("список_объектов")`	Удалить один или несколько объектов
`help(options)`	Справка о возможных опциях
`options()`	Посмотреть или установить текущие опции
`history(#)`	Вывести на экран последние # команд (по умолчанию 25)
`savehistory("мой_файл")`	Сохранить историю команд в файл *мой_файл* (по умолчанию `.Rhistory`)
`loadhistory("мой_файл")`	Загрузить историю команд (по умолчанию `.Rhistory`)
`save.image("мой_файл")`	Сохранить рабочее пространство в файл *мой_файл* (по умолчанию `.Rdata`)
`save("список_объектов", file="мой_файл")`	Сохранить определенные объекты в файл
`load("мой_файл")`	Загрузить сохраненное рабочее пространство в текущую сессию (по умолчанию `.Rdata`)
`q()`	Выйти из программы. Появится вопрос, нужно ли сохранить рабочее пространство

Таблицы данных – это основной класс объектов R, используемых для хранения данных. Обычно такие таблицы подготавливаются при помощи сторонних приложений (особенно популярна и удобна программа Microsoft Excel) и затем загружаются в среду R (https://r-analytics.blogspot.com/2011/07/r_22.html).

Для загрузки данных из CSV-файла с заголовками используется команда:

```
data <- read.table("C:\\Users\\Пользователь\\Desktop\\MAD\\вариант-1(л.р.1).csv", header=TRUE, sep=";")

# read.table - это функция (процедура).
# В круглых скобках - аргументы функции.
# ("C:\\Users\\Пользователь\\Desktop\\вариант-1(л.р.1).csv" - путь к файлу с данными
```

header= указываем, есть в первой строке названия переменных, или нет
sep= указываем, какой символ отделяет поля
row.names= столбец, в котором хранятся названия строк (параметр не обязателен).

Возможно, что в загруженном файле некорректно будет отображаться текст на русском языке. В этом случае необходимо использовать команду:

Sys.setlocale("LC_ALL", "Russian")

Файл можно предварительно открыть в текстовом редакторе и сохранить в другой кодировке, либо явно задать аргумент fileEncoding или encoding в функции read.table(), например: fileEncoding = "Windows-1251", encoding = "UTF-8", или encoding = "1251". Выбор параметра зависит от используемой операционной системы!

Для записи данных в CSV-файл используется команда:

```
write.table(data , file = "C:\\Users\\ Пользователь \\Desktop\\MAD\\вариант-1(л.р.1).csv", sep = ";", row.names = TRUE, col.names = NA)
```

Для импорта/экспорта данных из rds команды:

```
saveRDS(data, "data.rds")
data <- readRDS("data.rds")
```

Для импорта/экспорта данных из RData команды:

```
save(data,cdata,file="data.RData")
load("data.RData")
```

Для просмотра таблицы с данными используется команда:

```
View(data)
```

Для обращения к конкретным столбцам используется команда:

```
data$column_name
```

Можно делать различные выборки из данных.

Например, вывод данных из столбца column_name_1, для которых значение в столбце column_name_2 больше 70:

```
View(data$column_name_1[data$column_name_2 >70])
```

Также можно использовать команду subset для формирования подвыборки из БД:

```
View(subset(data, column_name_2 >70))
```

Можно выбирать одновременно строки и переменные и сформировать новую БД:

```
data1<-subset(data, column_name_2 >70, select=c("column_name_1", "column_name_3"))
```

Вывод первых трех строчек столбца column_name:

```
View(data$column_name[1:3])
```

Вывод первых трех столбцов данных:

```
View(data[1:3])
```

Вывод первого, третьего и пятого столбцов данных:

```
View(data[c(1,3,5)])
```

Вывод первых 100 строк данных:

```
View(data[1:100,])
```

Удаление записей, содержащих пропущенные значения:

```
data <- na.omit(data)
```

Вызов редактора данных:

```
data <- edit(data)
```

***Если в таблице с исходными данными длинные названия столбцов, то лучше ввести сокращенные названия для качественного визуального представления графиков и результатов анализа*!**

Для переименования нескольких столбцов:

```
names(data)<-c("возраст", "стаж", "часы", "доход")
```

Для переименования одного столбца, в квадратных скобках номер столбца:

```
colnames(data)[6] <- "процентр_успешных_задач"
colnames(data)[7] <- "количество_ошибок"
colnames(data)[8] <- "удовлетворенность_вбаллах"
colnames(data)[9] <- "удовлетворенность_качественная"
colnames(data)[10] <- "документирование"
```

Удаление колонки:

```
data$column_name <- NULL
```

или

```
data <- data[, -c(1:10)]
```

В результате будут удалены первые 10 столбцов с данными.

Удаление строк:

```
data <- data[-c(1:99), ]
```

В результате будут удалены первые 99 строк с данными.

В R реализован ряд функций, которые позволяют определить тип данных и преобразовать их в другой формат. Эти преобразования происходят в R сходным с другими языками программирования образом. Например, если добавить текстовые данные к числовому вектору, все значения вектора преобразуются в текстовый формат. Для проверки типа данных и преобразования их в другой формат можно использовать функции, перечисленные в табл. 3.

Таблица 3. – Функции, используемые для изменения формата данных

Проверка	Преобразование
is.numeric()	as.numeric()
is.character()	as.character()
is.vector()	as.vector()
is.matrix()	as.matrix()
is.data.frame()	as.data.frame()
is.factor()	as.factor()
is.logical()	as.logical()

Функции типа is._() возвращают TRUE или FALSE, тогда как функции типа as._() преобразуют данные в соответствующий формат. Пример представлен в приведенном ниже программном коде.

```
a <- c(1,2,3)
a [1] 1 2 3
is.numeric(a)
[1] TRUE
is.vector(a)
[1] TRUE
a <- as.character(a)
a
```

При работе с большими таблицами данных бывает сложно визуально исследовать всё их содержимое перед началом анализа. Однако визуального просмотра содержимого таблиц и не требуется – полную сводную информацию о них (равно как и о других объектах R) можно легко получить при помощи команды str() (structure – структура):

```
str(data)
```

Для расчета основных статистических характеристик по данным используется команда:

```
summary(data)
```

В результате выполнения команды summary(data) для факторов (качественных признаков) выводится информация об уровнях, которые может принимать фактор и количестве наблюдений на каждом уровне, для количественных переменных выводятся значения основных числовых характеристик: минимальное и максимальное значения, среднее, медиана, первый и третий квартили.

Также для вычисления статистических характеристик могут использоваться статистические функции, представленные в табл. 4.

Таблица 4. – Статистические функции

Функция	Описание
`mean(x)`	Среднее арифметическое `mean(c(1,2,3,4))` равно 2.5
`median(x)`	Медиана `median(c(1,2,3,4))` равно 2.5
`sd(x)`	Стандартное отклонение `sd(c(1,2,3,4))` равно 1.29
`var(x)`	Дисперсия `var(c(1,2,3,4))` равно 1.67
`mad(x)`	Абсолютное отклонение медианы `mad(c(1,2,3,4))` равно 1.48

`quantile(x, probs)`	Квантили, где *x* – числовой вектор, для которого нужно вычислить квантили, а *probs* – числовой вектор с указанием вероятностей в диапазоне [0; 1]
	`# 30-й и 84-й процентили x`
	`y <- quantile(x, c(.3,.84))`

`range(x)`	Размах значений `x <- c(1,2,3,4)` `range(x)` равно `c(1,4)`. `diff(range(x))` равно 3
`sum(x)`	Сумма `sum(c(1,2,3,4))` равно 10

`diff(x, lag=n)`	Разность значений в выборке, взятых с заданным интервалом (`lag`). По умолчанию интервал равен 1.
	`x <- c(1,5,23,29)`
	`diff(x)` равно `c(4, 18, 6)`

`min(x)`	Минимум `min(c(1,2,3,4))` равно 1
`max(x)`	Максимум `max(c(1,2,3,4))` равно 4
`scale(x, center=TRUE, scale=TRUE)`	Значения объекта *x*, центрованные (`center=TRUE`) или стандартизованные (`center=TRUE, scale=TRUE`) по столбцам. Пример дан в программном коде 5.6

В базовой версии R не вычисляются статистические характеристики асимметрии и эксцесса, но можно написать собственную функцию для вычисления этих характеристик.

В R пользователи могут создавать свои функции. Структура этих функций выглядит примерно так:

```
myfunction <- function(arg1, arg2, ... )
{  statements
return(object) }
```

Объекты внутри функции не доступны вне функции. Объект, который возвращается в результате действия функции, может быть любого типа – от скаляра до списка.

Функция для вычисления статистических характеристик выглядит следующим образом:

```
mystats <- function(x)
{
    if(is.numeric(x))
    {m <- mean(x)
    n <- length(x)
    s <- sd(x)
    skew <- sum((x-m)^3/s^3)/n
    kurt <- sum((x-m)^4/s^4)/n - 3

return(c(n=n, mean=m, stdev=s, skew=skew, kurtosis=kurt))      }}
```

Вызов функции для вычисления статистических характеристик столбца name_1:

```
mystats(data$name_1)
```

Многие R-пакеты имеют собственные функции, аналогичные стандартной summary(), для вывода компактных описательных сводок по таблицам данных. Например, пакет psych имеет функцию describe().

```
install.packages("psych")
library(psych)
sm<-describe(data)
View(sm)
```

В результате выполнения кода значения статистических характеристик будут выведены в таблицу.

Пояснения к п. 4.

В R реализованы обширные возможности для работы с графикой.

Например, для построения диаграммы рассеяния используется команда:

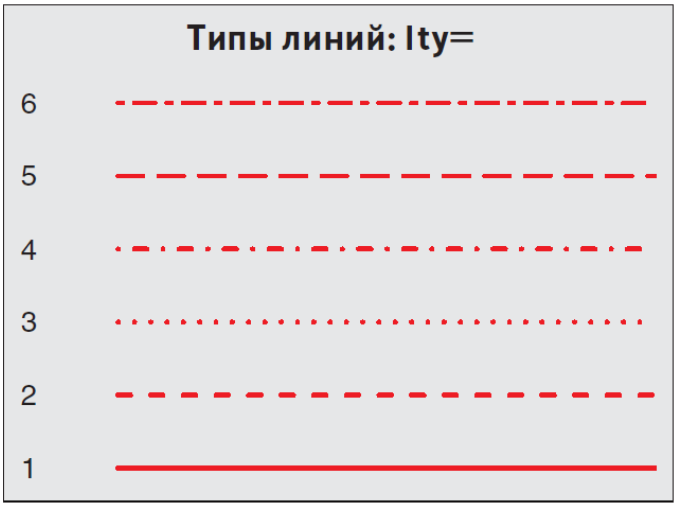
```
plot(data.frame(data$name_1,data$name_2))

#name_1 – имя признака, значения которого откладываются по оси X
#name_2 – имя признака, значения которого откладываются по оси Y.
```

У plot можно задать большое количество дополнительных параметров, список которых можно посмотреть, вызвав help(plot).

Например,

```
plot(data.frame(data$name_1,data$name_2), type="b", col="red", lty=2, pch=2,
lwd=2, main="название_графика", sub="подпись_графика",
    xlab="название_оси_X", ylab="название оси Y", xlim=c(0, 60), ylim=c(0,
70))
#type= - тип графика
#col= - цвет элементов
#lty= - тип линии (2 – dashed, пунктирный)
#pch= - тип точки
#lwd= - ширина линии
#xlim= - диапазон изменения оси X
#ylim= - диапазон изменения оси Y
```



В стандартной интерактивной сессии можно создавать график, вводя по одной команде и добавляя элементы графика, пока не получите то, что хотели.

Например:

```
attach(data)
plot(data$name_1, data$name_2)
abline(lm(name_2~name_1))
title("name")
detach(data)
```

Первая команда добавляет в траекторию поиска таблицу данных data. Вторая команда открывает окно графики и создает диаграмму рассеяния. Третья команда добавляет регрессионную прямую. Четвертая команда добавляет название. Последняя команда удаляет таблицу данных из пути поиска.

Диаграммы можно сохранять при помощи программного кода или меню графического пользовательского интерфейса. Для сохранения диаграммы при помощи кода разместите создающие диаграмму команды между командами, которые назначают место вывода и закрывают вывод. Например, следующий

программный код позволяет сохранить диаграмму в формате PNG под названием mygraph.png в текущей рабочей директории:

```
png("mygraph.png")
attach(data)
plot(data$name_1, data$name_2)
abline(lm(name_2~name_1))
title("name")
dev.off()
```

Можно сохранить диаграмму при помощи графического пользовательского интерфейса. При активированном графическом окне нужно выбрать в меню **Файл → Сохранить как**, а затем в появившемся диалоге выбрать нужный формат графического файла и директорию для сохранения.

Новая диаграмма, которая создается при помощи команды высокого уровня, такой как plot(), hist() или boxplot(), заменяет предыдущую диаграмму. Как же создать более одной диаграммы и иметь доступ к каждой из них? Есть несколько способов. Во-первых, можно открыть новое графическое устройство, *перед* тем как создавать новую диаграмму:

```
dev.new()
```

Каждая новая диаграмма будет появляться в последнем открытом окне.

Во-вторых, можно получить доступ к нескольким диаграммам сразу через пользовательский интерфейс. После того как открыто *первое* окно графики, выберите в меню **История команд → Запись**. Затем используйте пункты меню **Предыдущий** и **Следующий** для перемещения между созданными диаграммами.

В-третьих, можно использовать функции dev.new(), dev.next(), dev.prev(), dev.set() и dev.off() для одновременного открытия нескольких окон графики и выбора необходимой диаграммы. Эти команды работают под всеми операционными системами. Введите help(dev.cur), чтобы узнать больше об этом подходе.

Для построения радиальных диаграмм можно использовать команду pie():

```
x<-c(summary(data$name_1))
piepercent<- round(100*x/sum(x), 1)
pie(x, piepercent, radius=1, main="Радиальная диаграмма", xlab="название оси X", ylab="название оси Y", col=c("red", "blue"), clockwise=TRUE)
legend("topright", c("Yes","No"), cex = 0.8, fill =c("red", "blue"))
```

В данном примере признак name_1 является факторным (качественным) и принимает два значения (1 и 0). Создается объект x, который хранит количество 0 и 1 и объект piepercent – количество в процентах. Далее строится радиальная диаграмма, состоящая из красного и синего секторов с указанием

процентов. Последняя команда задает легенду, расположенную в правом верхнем углу графика, параметр cex= задает размер окна легенды.

Также можно размещать в одном графическом пространстве несколько диаграмм. Например, построим две столбиковые диаграммы по признаку name_2: первая диаграмма построена по значениям, для которых признак name_1="Yes"; вторая построена по значениям, для которых признак name_1="No":

```
par(mfrow = c(1, 2))
barplot(table(data$name_2[data$name_1=="Yes"]), col=c("red"), main="Yes" ,
xlab="название оси X", ylab="название оси Y")
barplot(table(data$name_2[data$name_1=="No"]), col=c("blue"), main="No",
xlab=" название оси X ", ylab=" название оси Y " )
```

Первая команда задает разбиение графического пространства (сколько графиков будет по строке и по столбцу). Вторая и третья команды строят столбиковую диаграмму в зависимости от условий.

Диаграммы размахов (box plot) иллюстрируют распределение значений количественной переменной, отображая пять параметров: минимум, нижний квартиль (75-й процентиль), медиану (50-й процентиль), верхний квартиль (25-й процентиль) и максимум. На этой диаграмме также могут быть отображены вероятные выбросы (значения, выходящие за диапазон в ±1.5 межквартильного размаха, разности верхнего и нижнего квартилей).

Например, команда:

```
boxplot(data$name_3, main="Ящик с усами", ylab="Название оси Y")
```

позволяет построить диаграмму, показанную на рис. 1. (подписи к элементам добавлены вручную).

Рисунок 1 – Диаграмма размаха

По умолчанию каждый «ус» продолжается до минимального или максимального значения, которое не выходит за пределы 1.5 межквартильного размаха. Выходящие за эти пределы значения отмечаются точками (не показано). Диаграммы размахов можно построить для отдельных переменных или для групп переменных. Общий вид команды таков:

boxplot(*formula*, data=*dataframe*)

где *formula* – это формула, а *dataframe* обозначает таблицу данных (или список), где содержатся данные. Примером формулы может служить выражение y~A, где для каждого значения качественной переменной A будет построена отдельная диаграмма размахов для количественной переменной y. Формула y~A*B позволит получить отдельные диаграммы размахов для всех комбинаций значений переменной y, заданных категориальными переменными A и B. Параметр varwidth=TRUE позволяет получить диаграмму, на которой ширина «ящиков» будет пропорциональна квадратному корню из размера выборки. Используйте параметр horizontal=TRUE, чтобы поменять оси местами. Если добавить параметр notch=TRUE, то получатся «ящики» с «насечками». Если «насечки» двух ящиков не перекрываются, высока вероятность того, что медианы соответствующих совокупностей различаются (Chambers et al., 1983, стр. 62).

Например, команда:

```
par(mfrow=c(1,1))
```

```
boxplot(data$name_3 ~ data$name_1,
+         xlab = "Название оси X",
+         ylab = "Название оси Y",
+         col = "coral", data = data)
```

Переменная name_3 – количественная, name_1 – качественная, принимающая два значения (1 и 0). В результате выполнения команды будет построено два ящика с усами на одном графике.

На рис. 2 приведен пример графика, построенного по четырем признакам и четырем классам (см. [1]).

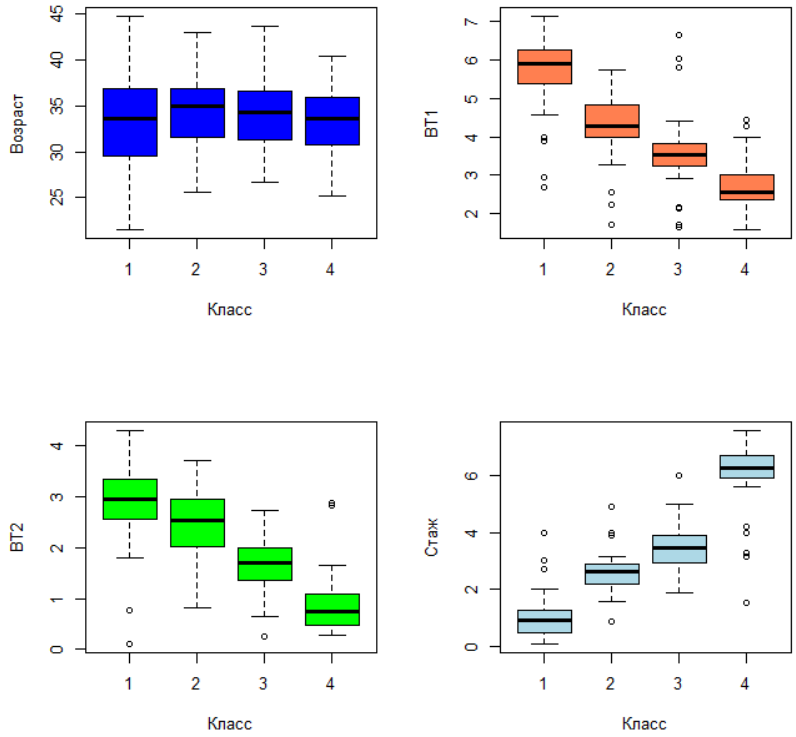


Рисунок 2 – Диаграмма размаха по четырем признакам

Гистограмма (оценка функции плотности распределения вероятностей) графически отображает распределение значений количественных переменных, разделяя диапазон значений на заданное число отрезков по оси *x* и отображая частоту значений внутри каждого отрезка на оси *y*. Гистограмму можно создать при помощи команды:

hist(x), где *x* – это числовой вектор.

Опция freq=FALSE позволяет построить гистограмму на основе плотности вероятности, а не частот значений. Параметр breaks определяет число отрезков. По умолчанию все отрезки имеют одинаковую длину.

Пример построения гистограммы с наложением кривой плотности распределения вероятностей:

hist(data$name_1, freq=FALSE, breaks=12, col=”red”, xlab=”название оси X”,
+ main=”Гистограмма”)
lines(density(data$name_1))

На рис. 3 приведен пример построения гистограмм с наложением функции нормального закона распределения (см. [1]).

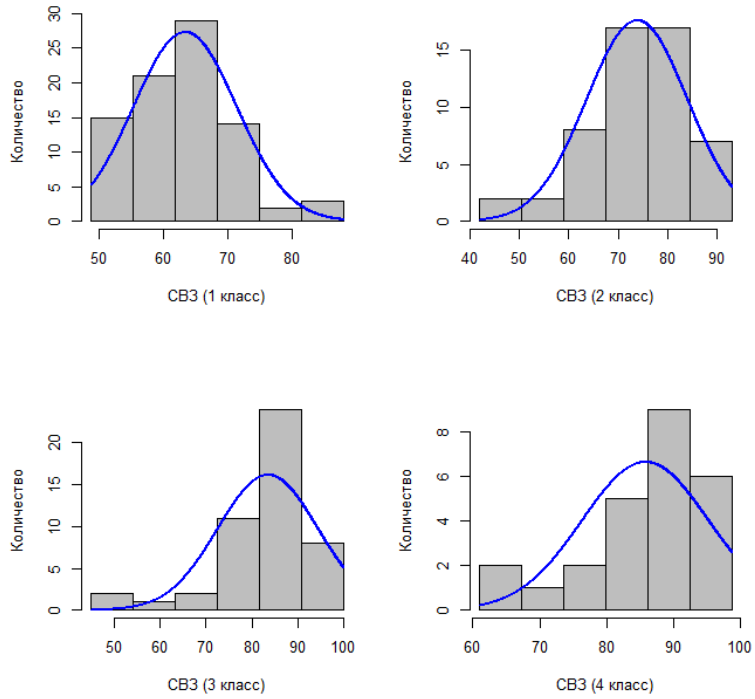


Рисунок 3 – Гистограмма с наложением нормального закона распределения

Матрицу диаграмм рассеяния можно построить с помощью функции:
pairs(~data$name_1+data$name_2+data$name_3, data=data, main="Матричный график")

Все переменные, перечисленные справа от знака ~, попадают на диаграмму. На этой диаграмме отражены взаимосвязи между всеми парами переменных. Например, диаграмма рассеяния для name_1 и name_2 расположена на пересечении строки и столбца с названиями этих переменных. Изменяя параметры команды, можно отображать только верхний или только нижний треугольник относительно главной диагонали. Например, опция upper.panel=NULL позволит оставить только нижний треугольник с диаграммами.

На рис. 4 приведен пример построения матричного графика (см. [1]).

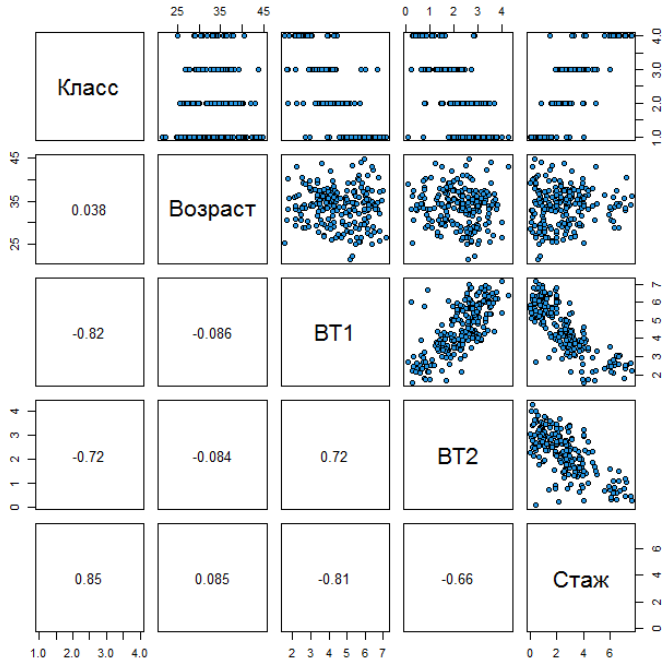


Рисунок 4 – Матричный график

Разные варианты матричных графиков также могут быть построены с помощью функции ggpairs() из библиотеки GGally. Код для построения матричных графиков между исследуемыми признаками следующий:

```
install.packages("GGally")

install.packages("ggplot2")

library(GGally)

library(ggplot2)

ggpairs(data, columns = 1:5, aes(color = группа,alpha = 0.5))
```

Преимущество использования функции ggpairs() по сравнению с базовой функцией pairs() заключается в том, что построенный матричный график содержит сравнительно больше информации о распределении и взаимосвязях между признаками. На матричном графике отображена следующая информация (см. рис 5, [1]):

– графики плотности распределения вероятностей для количественных признаков для каждого класса и столбиковые диаграммы для качественных признаков (главная диагональ);

– диаграммы рассеяния между парами количественных признаков с выделением цвета класса (под главной диагональю);

– диаграммы размаха для каждого класса для количественных признаков (первая строка);

– гистограммы распределения количественных признаков по классам (первый столбец);

– значения коэффициентов корреляции Пирсона с оценкой их статистической значимости для пар количественных признаков, рассчитанные по всем данным и с разделением по классам (выше главной диагонали).

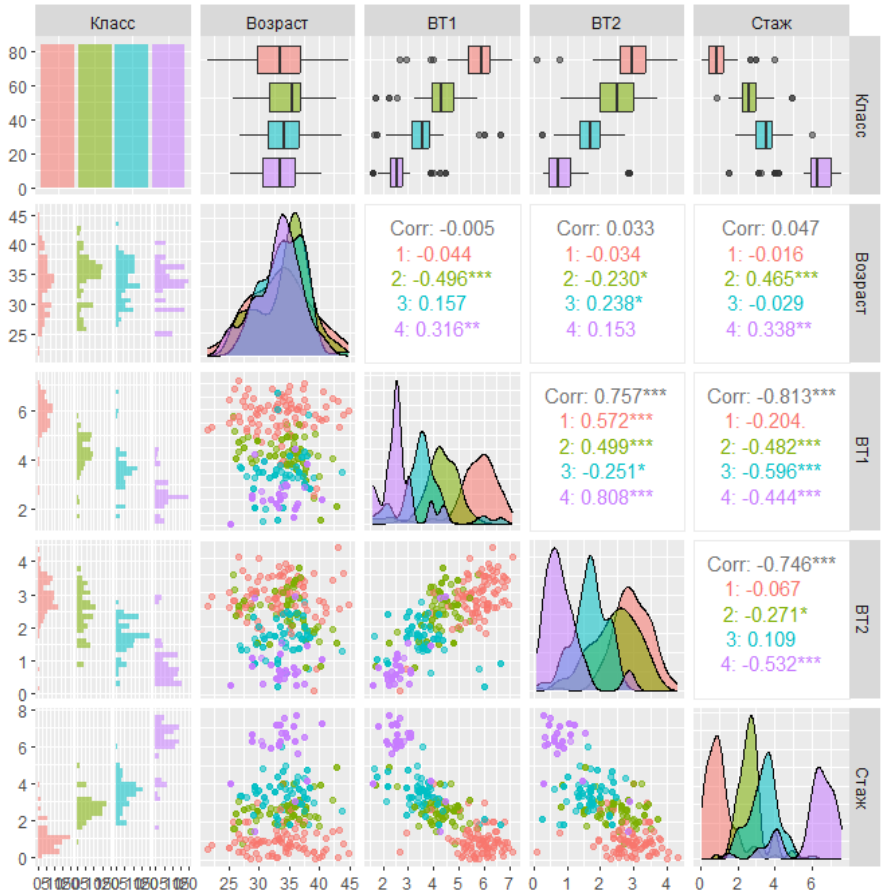


*Рисунок 5 – Матричный график (*ggpairs*)*

Пояснения к пп. 5-6.

Для аналитической проверки гипотезы о соответствии нормальному закону распределения могут быть применены критерии согласия: Шапиро – Уилка, Крамера – Мизеса, Андерсона – Дарлинга, соответственно функции, реализующие критерии [1]:

```
install.packages("nortest")
library(nortest)
shapiro.test(data1$CB3)
```

```
        Shapiro-Wilk normality test

data:  data1$CB3
W = 0.96328, p-value = 0.01703
```

```
cvm.test(data1$CB3)
```

```
        Cramer-von Mises normality test

data:  data1$CB3
W = 0.04193, p-value = 0.6415
```

```
ad.test(data1$CB3)
```

```
        Anderson-Darling normality test

data:  data1$СВЗ
A = 0.48432, p-value = 0.2223
```

Гипотеза о нормальном распределении признака СВЗ внутри классов не отвергается при уровне значимости 0,01 (p-value>0,01).

В R реализованы различные способы построения и анализа таблиц сопряженности для исследования взаимосвязей между качественными признаками. В табл. 5 приведены основные функции для построения таблиц сопряженности.

Таблица 5 – Основные функции для работы с таблицами сопряженности

Функция	Описание
`table(var1, var2, …, varN)`	Создает N-мерную таблицу сопряженности для N категориальных переменных (факторов)
`xtabs(formula, data)`	Создает N-мерную таблицу сопряженности на основе формулы и матрицы или таблицы данных
`prop.table(table, margins)`	Представляет значения таблицы в виде долей от значений крайнего поля таблицы, задаваемого параметром *margins*
`margin.table(table, margins)`	Суммирует значения таблицы по строкам или столбцам (определяется параметром *margins*)
`addmargins(table, margins)`	Вычисляет описательные статистики *margins* (суммы по умолчанию) для таблицы
`ftable(table)`	Создает компактную «плоскую» таблицу сопряженности

Для построения простой таблицы сопряженности между двумя качественными признаками используется команда:

```
table<-table(data$name_1,data$name_2)
View(table)
```

Для вычисления значений в процентах используется команда:

```
prop.table(table)
```

Для вычисления сумм по строкам и столбцам таблицы:

```
addmargins(table)
```

Для исследования взаимосвязей между качественными признаками реализованы различные тесты и коэффициенты корреляции. Один из самых часто используемых на практике – Chi-квадрат критерий (тест) Пирсона. Критерий

проверяет основную гипотезу об отсутствии взаимосвязи между двумя качественными переменными:
chisq.test(table(data$name_1,data$name_2))

В результате выполнения команды выдается расчетное значение критической статистики критерия (*X-squared*), числе степеней свободы (*df*) и p-значение:

Pearson's Chi-squared test with Yates' continuity correction

data: table(data$name_1, data$name_2)
X-squared = 0.016356, df = 1, p-value = 0.8982

Также реализован критерий Фишера для проверки основной гипотезы об отсутствии взаимосвязи между двумя качественными признаками:

fisher.test(table(data$name_1,data$name_2))

В результате выполнения команды выдается p-значение и отношение шансов (отношение шансов вычисляется только для бинарных переменных и таблиц сопряженности 2 на 2):

Fisher's Exact Test for Count Data

data: table(data$name_1, data$name_2)
p-value = 0.203
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
0.3690955 1.2131081
sample estimates:
odds ratio
0.6707924

Ниже приведен фрагмент кода, который выполняет расчет по критерию Chi-квадрат для всех качественных признаков, включенных в таблицу data и выводит p-значение:

```
#Получаем имена только качественных признаков
factor_names <- names(data[,unlist(lapply(data, is.factor))])

#Создаем таблицу для значений p.value
chi_results <- matrix(NA,nrow = length(factor_names),ncol=length(fac-
tor_names ))
colnames(chi_results) <- c(factor_names )
rownames(chi_results) <- c(factor_names )
```

```
#Проводим Chi.test и заполняем таблицу
for (col_name in factor_names){
      for (col_name2 in factor_names){
   chi_results[col_name,col_name2] <-    chisq.test(ta-
ble(data[,col_name],data[,col_name2]))$p.value
      }
}
chi_results
```

Для оценки взаимосвязи между качественной переменной и количественной переменной (в случае нормального распределения переменной) используется однофакторный одномерный дисперсионный анализ (ANOVA).

Функция aov() реализует однофакторный одномерный дисперсионный анализ

```
aov_model <- aov(количество.ошибок ~ пол, data=data)
summary(aov_model)
```

В примере оценивается взаимосвязь между количеством допущенных ошибок и полом.

В результате выполнения команды выдается расчетное F-значение (*F value*), число степеней свободы (*Df*) и p-значение (*Pr(>F))*

	Df	*Sum Sq*	*Mean Sq*	*F value*	*Pr(>F)*
пол	*1*	*6*	*5.604*	*0.285*	*0.594*
Residuals	*198*	*3887*	*19.629*		

Гипотеза о статистической незначимости отличий в распределении признака «количество.ошибок» от пола не отвергается при уровне значимости 0,05 (p-value=0,594>0,05).

Для оценки взаимосвязи между качественной переменной и количественной переменной (в случае, если распределение переменной не подчиняется нормальному закону распределения) используется критерий Краскела-Уоллиса:

```
kruskal.test(количество.ошибок ~ пол, data= data)
```

В результате выдается расчетное значение критической статистики критерия (chi-squared) и p-value:

```
        Kruskal-Wallis rank sum test

data:  Возраст by Класс
Kruskal-Wallis chi-squared = 6.4172, df = 3, p-value = 0.09299
```

Гипотеза о статистической незначимости отличий в распределении признака «количество.ошибок» от пола не отвергается при уровне значимости 0,05 (p-value=0,093>0,05).

В R можно рассчитывать разные коэффициенты корреляции для количественных показателей, включая коэффициенты Пирсона, Спирмена, Кэнделла, частные, полихорические и многорядные.

Линейный коэффициент корреляции Пирсона (Pearson product moment correlation) отражает степень линейной связи между двумя количественными переменными. Коэффициент ранговой корреляции Спирмана (Spearman's Rank Order correlation) – мера связи между двумя ранжированными переменными. Тау Кендалла (Kendall's Tau) – также непараметрический показатель ранговой корреляции.

Функция cor() позволяет вычислить все три коэффициента, а функция cov() рассчитывает ковариации. У этих функций есть много опций, но упрощенный формат вычисления корреляций таков:
cor(x, use= , method=).

Эти опции описаны в табл. 6. Значения опций по умолчанию: use="everything" и method="pearson".

Таблица 6 – Опции функций cor() и cov()

Опция	Описание
`x`	Матрица или таблица данных
`use=`	Упрощает работу с пропущенными данными. Может принимать следующие значения: `all.obs` (предполагается, что пропущенные значения отсутствуют; их наличие вызовет сообщение об ошибке), `everything` (любая корреляция, включающая строку с пропущенным значением, не будет вычисляться – обозначается как `missing`), `complete.obs` (учитываются только строки без пропущенных значений) и `pairwise.complete.obs` (учитываются все полные наблюдения для каждой пары переменных в отдельности)
`method=`	Определяет тип коэффициента корреляции. Возможные значения – `pearson`, `spearman` или `kendall`

Ниже приведен пример вычисления коэффициентов корреляции между количественными признаками, хранящимися в таблице data.

```
M <- data[,unlist(lapply(data, is.numeric))]
N1<-cor(M,use="pairwise.complete.obs")
N2<-cor(M,use="pairwise.complete.obs",method="spearman")
N3<-cor(M,use="pairwise.complete.obs",method="kendall")
```

Первая команда выбирает из таблицы data количественные признаки и сохраняет их в таблице M. Вторая команда вычисляет матрицу коэффициентов корреляции Пирсона между всеми количественными признаками, третья команда – матрицу коэффициентов корреляции Спирмена, четвертая команда – матрицу коэффициентов корреляции Кендалла.

Для вычисления коэффициентов частной корреляции можно использовать функцию pcor() из пакета ggm. Этот пакет не поставляется с базовой версией программы и требует установки.

Формат применения этой функции таков:

pcor(u, S),

где u – это числовой вектор, в котором первые два числа – это номера переменных, для которых нужно вычислить коэффициент, а остальные числа – номера «влияющих» переменных (воздействие которых должно быть исключено). S – это ковариационная матрица для всех этих переменных.

Пример:

```
install.packages("ggm")
library(ggm)
M <- data[,unlist(lapply(data, is.numeric))]
pcor(c(2,5,1,3,4), cov(M))
```

[1] 0.346

Вычислен частный коэффициент корреляции между второй и пятой (по номеру столбца) переменной таблицы M, при условии, что исключено влияние переменных 1,3,4.

Для графического представления степени взаимосвязей между признаками необходимо установить пакет corrplot:

```
install.packages("corrplot")
library(corrplot)

col <- colorRampPalette(c("#BB4444", "#EE9988", "#FFFFFF", "#77AADD",
"#4477AA"))
corrplot(N1, method="color", col=NULL,
         type="upper", order="hclust",
         addCoef.col = "black", tl.col="black", tl.srt=45,
         sig.level = 0.01, insig = "blank",
         diag=FALSE
)
```

В результате будет выведена корреляционная матрица Пирсона, в которой значения коэффициентов выделены цветами в соответствии с их величиной (пример на рис. 2).

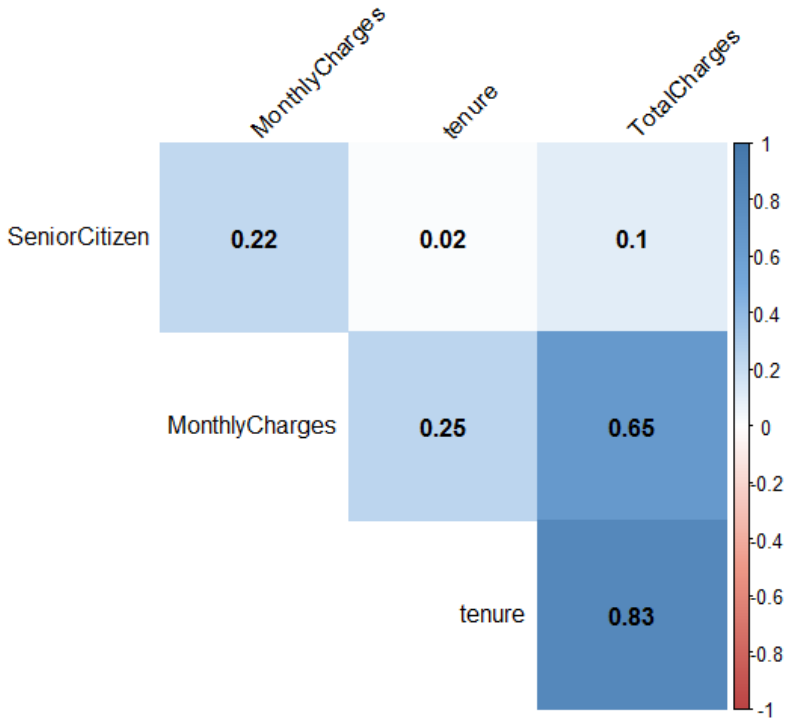


Рисунок 2 – Пример графического представления матрицы коэффициентов корреляции Пирсона

Более подробно о возможностях визуализации корреляций с помощью пакета corrplot можно посмотреть по ссылке: http://www.sthda.com/english/wiki/visualize-correlation-matrix-using-correlogram.

Для проверки значимости отдельных корреляционных коэффициентов Пирсона, Спирмена и Кендалла можно использовать функцию cor.test(). Упрощенный формат ее применения таков:

cor.test(x, y, alternative = , method =)

где *x* и *y* – это переменные, корреляция между которыми исследуется, опция alternative определяет тип теста (“two.side”, “less” или “greater”), опция method задает тип корреляции (“pearson”, “kendall” или “spearman”). Используйте опцию alternative=”less” для проверки гипотезы о том, что в генеральной совокупности коэффициент корреляции меньше нуля, а опцию alternative=”greater” – для проверки того, что он больше нуля. По умолчанию alternative=”two.side” (проверяется гипотеза о том, что коэффициент корреляции в генеральной совокупности не равен нулю). Пример приведен в следующем программном коде.

cor.test(data$name_1, data$name_2)

Pearson’s product-moment correlation
data: name_1 and name_2

t = 6.85, df = 48, p-value = 1.258e-08
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
0.528 0.821
sample estimates:
cor
0.703

В результате выполнения команды выдается расчетное значение критической статистики (t), число степеней свободы (df), p-значение (p-value) и 95% доверительный интервал, в который попадает истинное значение коэффициента корреляции Пирсона.

4. Отчет о работе

Содержание

1. Постановка задачи ИАД.

2. Расчет основных статистических характеристик для первой и второй группы и для всей выборки. Выводы.

3. Графический разведочный анализ данных (код R для построения графиков и построенные графики, выводы о структуре и характере распределения данных на основе анализа графиков, различие распределения данных по группам).

4. Проверка гипотезы о типе закона распределения. Выводы в терминах решаемой задачи.

5. Корреляционный анализ данных (код R, результаты Chi-критерия и критерия Фишера, результаты расчета дисперсионного анализа, критерия Краскела-Уоллиса, матрицы коэффициентов корреляции Пирсона, Спирмена, Кендалла, частные коэффициенты корреляции, оценка статистической значимости коэффициентов корреляции, выводы о силе и направлении связи между исследуемыми показателями в терминах решаемой задачи для первой и второй групп).

5. Выводы по работе в произвольной форме.

5. Вопросы к работе

1. Понятие интеллектуального анализа данных. Методы Data Mining.

2. Понятие разведочного анализа данных. В чем отличие процедуры Data Mining от методов классического статистического анализа данных?

3. Методы графического разведочного анализа данных средствами R.

4. Расчет основных статистических характеристик в среде R.

5. Какую информацию о природе данных можно получить при анализе диаграмм рассеяния, категорированных графиков, радиальных диаграмм, гистограмм, диаграмм размаха?

6. В чем достоинства и недостатки графических методов разведочного анализа данных?

7. Какие аналитические методы первичного разведочного анализа данных вы знаете?

8. Какие основные статистические характеристики количественных переменных вы знаете? Их описание и интерпретация в терминах решаемой задачи.

9. Какие критерии согласия используются для проверки гипотезы о типе закона распределения? Математическое описание. Расчет в R.

10. Какие измерители связи применяются для измерения степени тесноты связи между качественными переменными? Их расчет в R, оценка статистической значимости и интерпретация в терминах решаемой задачи.

11. Однофакторный дисперсионный анализ. Постановка задачи, алгоритм метода, расчет в R, интерпретация в терминах решаемой задачи.

12. Критерий Краскела-Уоллиса. Алгоритм критерия, расчет в R, интерпретация в терминах решаемой задачи.

13. Какие измерители связи применяются для измерения степени тесноты связи между количественными переменными? Их расчет в R, оценка статистической значимости и интерпретация в терминах решаемой задачи.

Литература

1. Альсова О. К. Методы и модели решения задачи классификации. Основные этапы : учеб. пособие / О. К. Альсова, Н. А. Зеленчук. - Новосибирск: НГТУ, 2024. - 88 с. - ISBN 978-5-7782-5280-6.
2. Анализ данных: учебник для вузов / под редакцией В. С. Мхитаряна. — Москва : Издательство Юрайт, 2025. — 448 с. — (Высшее образование). — ISBN 978-5-534-19964-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/560311 (дата обращения: 22.06.2025).
3. Миркин, Б. Г. Базовые методы анализа данных : учебник и практикум для вузов / Б. Г. Миркин. — 3-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2025. — 297 с. — (Высшее образование). — ISBN 978-5-534-19709-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/560414 (дата обращения: 22.06.2025).
4. Гмурман, В. Е. Теория вероятностей и математическая статистика : учебник для вузов / В. Е. Гмурман. — 12-е изд. — Москва : Издательство Юрайт, 2025. — 479 с. — (Высшее образование). — ISBN 978-5-534-00211-9. — Текст : электронный // ЭБС Юрайт [сайт]. — URL: https://urait.ru/bcode/449646
5. Кабаков Р. И. R в действии / И. Р. Кабаков ; пер. с англ. А. Н. Киселева. – Москва : ДМК Пресс, 2023. – 768 с.
6. Мастицкий С. Э. Статистический анализ и визуализация данных с помощью R / С. Э. Мастицкий, В. К. Шитиков. – Москва : ДМК Пресс, 2023. – 497 с.
7. Официальный сайт R [Электронный ресурс]. – Режим доступа: https://www.r-project.org (дата обращения: 15.03.2025).
8. Репозиторий пакетов R [Электронный ресурс]. – URL: https://cran.r-project.org/web/packages (дата обращения: 15.03.2025).

9. Дж. Д. Лонг и Пол Титор Книга рецептов: Проверенные рецепты для статистики, анализа и визуализации данных / пер. с анг. Д. А. Беликова. – М.: ДМК Пресс, 2020. – 510 с.
10. Шитиков В. К., Мастицкий С. Э. (2017) Классификация, регрессия, алгоритмы Data Mining с использованием R. - Электронная книга, адрес доступа: https://github.com/ranalytics/data-mining

Постановка задачи ИАД. Варианты 1-5. Изучаются показатели работы программистов крупной организации. Рассматриваются следующие показатели (признаки) для каждого программиста:

- пол (1-м, 2-ж);
- возраст;
- стаж работы;
- процент разработок, выполненных в срок в рамках бюджета с требуемым функционалом (за год);
- количество ошибок, выявленных пользователем (за год);
- стаж работы по специальности в данной организации;
- степень удовлетворенности заказчика;
- качество документирования (1 – низкое, 2 – среднее, 3 – выше среднего, 4 – высокое).

Необходимо провести предварительный разведочный анализ данных с целью описания характера распределения данных, выявления структуры взаимосвязей между показателями.

Программисты разбиты на две группы в зависимости от стажа работы.

Вариант 1
1 группа – стаж менее 3,5
2 группа – стаж более 3,5

Вариант 2
1 группа – стаж менее 4,5
2 группа – стаж более 4,5

Вариант 3
1 группа – стаж менее 2,5
2 группа – стаж более 2,5

Вариант 4
1 группа – стаж менее 5
2 группа – стаж более 5

Вариант 5
1 группа – стаж менее 4
2 группа – стаж более 4

Варианты 6-10. Исследуются покупатели Интернет-магазина. Было опрошено 200 клиентов, каждому анкетируемому предлагалось ответить на следующие вопросы:

- возраст;

- пол (1-м, 2-ж);
- количество покупок за год;
- средняя стоимость покупок за год;
- среднее число страниц, просмотренных за визит;
- количество обращений в службу поддержки за год;
- степень удовлетворенности услугами;
- степень активности (участие в Интернет-опросах, где 1 – никогда, 2 – редко, 3 – часто, 4 – постоянно)

Необходимо провести предварительный разведочный анализ данных с целью описания характера распределения данных, выявления структуры взаимосвязей между показателями.

Анкетируемые разбиты на две группы в зависимости от возраста.

Вариант 6

1 группа – возраст менее 25 лет
2 группа – возраст более 25 лет

Вариант 7

1 группа – возраст менее 25 лет
2 группа – возраст более 25 лет

Вариант 8

1 группа – возраст менее 30 лет
2 группа – возраст более 30 лет

Вариант 9

1 группа – возраст менее 30 лет
2 группа – возраст более 30 лет

Вариант 10

1 группа – возраст менее 30 лет
2 группа – возраст более 30 лет

Варианты 11-15. Исследуется рынок потребителей услуги «Подключение к сети Интернет». Было опрошено 200 потребителей услуг Интернет, каждому анкетируемому предлагалось ответить на следующие вопросы:

- возраст;
- пол (1-м, 2-ж);
- стаж работы в сети Интернет;
- средний доход в месяц, в тыс. руб.;
- профессиональная специализация (насколько часто используется сеть Интернет в профессиональной деятельности): 1 – не использую; 2 – кране редко; 3 – ежедневно, 4 – постоянно;
- среднее количество просматриваемых страниц в месяц;
- степень активности (участие в Интернет-опросах)

Необходимо провести предварительный разведочный анализ данных с целью описания характера распределения данных, выявления структуры взаимосвязей между показателями.

Анкетируемые разбиты на две группы в зависимости от возраста.

Вариант 11

возраст – менее 30 лет
возраст – более 30 лет

Вариант 12

возраст – менее 30 лет
возраст – более 30 лет

Вариант 13

возраст – менее 32 лет
возраст – более 32 лет

Вариант 14

возраст – менее 32 лет
возраст – более 32 лет

Вариант 15

возраст – менее 30 лет
возраст – более 30 лет