Министерство образования и науки Российской Федерации

ФГАОУ ВО «Севастопольский государственный университет»

Институт информационных технологий

и управления в технических системах

Лабораторная работа №1

«Исследование возможностей языка R для статистического анализа данных»

по дисциплине «Интеллектуальный анализ данных»

для студентов всех форм обучения направления подготовки

09.03.02 «Информационные системы и технологии»

Севастополь

2019

УДК 004.9:528(075.8)

Исследование возможностей языка R для статистического анализа данных. Методические указания к лабораторным занятиям по дисциплине «Интеллектуальный анализ данных» / Сост.: И.В. Дымченко, И.П. Шумейко, О.А. Сырых – Севастополь: Изд-во СевГУ, 2019 – 18 с.

Методические указания предназначены для проведения лабораторных работ по дисциплине «Интеллектуальный анализ данных». Целью методических указаний является помощь студентам в изучении возможностей системы RStudio. Излагаются практические сведения необходимые для выполнения лабораторной работы, требования к содержанию отчета.

Методические указания рассмотрены и утверждены на методическом семинаре и заседании кафедры «Информационные системы» (протокол № 1 от 29 августа 2018 г.)

Лабораторная работа №1
Исследование возможностей языка R для статистического анализа данных

Цель:

- изучить основные особенности языка R;
- исследовать возможности языка R для работы с графикой.

Время: 2 часа

Лабораторное оборудование: персональные компьютеры, выход в сеть Internet, RStudio.

Краткие теоретические сведения

1. Язык R, как инструмент статистического анализа данных

R – статистическая система анализа, созданная Россом Ихакой и Робертом Гентлеманом (1996, J.Comput. Граф. Stat., 5: 299-314). R является и языком и программным обеспечением; его наиболее замечательный особенности:

- эффективная обработка данных и простые средства для сохранения результатов;
- набор операторов для обработки массивов, матриц, и других сложных конструкций;
- большая, последовательная, интегрированная коллекция инструментальных средств для проведения статистического анализа,
- многочисленные графические средства;
- простой и эффективный язык программирования, который включает много возможностей.

Язык R – рассматривают как диалект языка S созданный AT&T Бэлл Лаборатории. S доступен как программное обеспечение S-PLUS коммерческой системы MathSoft (см.http://www.splus.mathsoft.com для получения дополнительной информации). Есть существенные различия в концепции R и S (те, кто хочет знать больше об этом может читать статью, написанную Gentleman и Ihaka (1996) или R-FAQ (часто задаваемые вопросы) (http://cran.r-project.org/doc/FAQ/R-FAQ.html).

R доступен в нескольких формах: исходный текст программ, написанный на C (и некоторые подпрограммы в Fortran77) и в откомпилированном виде.

R – язык со многими функциями для выполнения статистического анализа и графического отображения результатов, которые визуализируется сразу же в собственном окне и могут быть сохранены в различных форматах (например, jpg, png, bmp, eps, или wmf под Windows, ps, bmp, pictex под Unix). Результаты статистического анализа могут быть отображены на экране.

Некоторые промежуточные результаты (P-values, коэффициент регрессии и т.п.) могут быть сохранены в файле и использоваться для последующего анализа.

R – язык, позволяющий пользователю использовать операторы циклов, чтобы последовательно анализировать несколько наборов данных. Также возможно объединить в отдельную программу различные статистические функции, для проведения более сложного анализа.

В библиотеках среды статистического программирования R присутствуют процедуры, реализующие все необходимые методы предварительного анализа и обработки данных, вариограммного анализа данных, а также геостатистического оценивания на основе кригинга. Кроме этого, методы кригинга реализованы в целом ряде программных продуктов. В частности, это пакет пространственного анализа и моделирования Surfer; в котором представлены и средства картографической визуализации результатов моделирования.

R держит все свои вычисления в оперативной памяти, поэтому если в процессе работы выключится питание, то результаты сессии, не записанные явным образом в файл, пропадут. Эта

особенность, к сожалению, также не позволяет R работать с действительно большими объёмами (порядка сотен тысяч и более записей) данных.

2. Графические интерфейсы языка R

Для удобства работы с R разработан ряд графических интерфейсов, в том числе RStudio, JGR, RKWard, SciViews-R, Statistical Lab, R Commander, Rattle.

Кроме того, в ряде текстовых и кодовых редакторов предусмотренные специальные режимы для работы с R, в частности в ConTEXT, Emacs (Emacs Speaks Statistics), jEdit, Kate, Syn, TextMate, Tinn-R, Vim, Bluefish, WinEdt (с пакетом RWinEdt).

Для среды разработки Eclipse существует специализированный R-плагин; доступ к функциям и среде выполнения R возможен из Python с использованием пакета RPy; работать с R можно из эконометрического пакета Gretl.

После установки, для запуска интерпретатора R достаточно выполнить в терминале команду:

```
$ R
```

После чего в консоли появится:

```
R version 3.2.4 (2016-03-10) - "Very Secure Dishes"
Copyright (C) 2016 The R Foundation for Statistical Computing
Platform: x86_64-apple-darwin13.4.0 (64-bit)
```

R – это свободное ПО, и оно поставляется безо всяких гарантий. Его можно распространять при соблюдении некоторых условий. Команда 'license()' служит для получения более подробной информации.

R – это проект, в котором сотрудничает множество разработчиков. Команда `'contributors()'` служит для получения дополнительной информации и `'citation()'` для ознакомления с правилами упоминания R и его пакетов в публикациях.

Команда `'demo()'` для запуска демонстрационных программ, `'help()'` – для получения справки, `'help.start()'` - для доступа к справке через браузер.

Для выхода необходимо использовать команду

```
> q()
```

Как видно из выше приведенного листинга для получения подробной информации (справки) о любой функции, необходимо выполнить команду:

```
> help(<имя функции>)
```

либо

```
> ?<имя функции>
```

Например, выполнив одну из команд

```
> help(q)
> ?q
```

получится следующий результат:

```
Terminate an R Session

Description:

     The function ‘quit’ or its alias ‘q’ terminate the current R
     session.

Usage:

     quit(save = "default", status = 0, runLast = TRUE)
        q(save = "default", status = 0, runLast = TRUE)

Arguments:

    save: a character string indicating whether the environment
          (workspace) should be saved, one of ‘"no"’, ‘"yes"’,
‘"ask"’
          or ‘"default"’.

  status: the (numerical) error status to be returned to the
operating
          system, where relevant.  Conventionally ‘0’ indicates
          successful completion.
 runLast: should ‘.Last()’ be executed?

Details:

     ‘save’ must be one of ‘"no"’, ‘"yes"’, ‘"ask"’ or ‘"default"’.
In
     the first case the workspace is not saved, in the second it is
     saved and in the third the user is prompted and can also decide
     _not_ to quit.  The default is to ask in interactive use but
may
     be overridden by command-line arguments (which must be supplied
in
     non-interactive use)...
```

Кроме функции `help()`, полезной, если неизвестно точное названия функции, может оказаться команда:

```
> help.search("vector")
```

результатом которой, будет список команд, свойственных «векторам», с кратким описанием.

Команда `apropos()` выдаст просто список команд, содержащих строку, которая была в кавычках.

Одним из важных преимуществ R является наличие для него многочисленных расширений (пакетов) практически для решения любой задачи обработки данных, которые можно легко скачать с официального онлайн-репозитория – CRAN (http://cran.r-project.org/) и установить с помощью команды:

```
install.packages()
```

Под Windows есть соответствующий пункт «Установить пакет(ы)» в меню «Пакеты».

Как только пакет установлен, то он сразу готов к работе. Нужно только инициализировать его перед употреблением. Для этого служит команда `library()`.

Под Windows необходимо выполнить команду «Пакеты» → «Включить пакет»

При установке R автоматически устанавливаются так называемые базовые пакеты, без которых система просто не работает (например, это такие пакеты, как `base`, `grDevices`), и некоторые «рекомендованные» пакеты (например, `cluster` (для решения задач кластерного анализа), `nlme` (для анализа нелинейных моделей) и д.р.).

3. Основные особенности языка R

R – регистрозависимый язык, т. е., например, символы «A» и «a» могут обозначать разные объекты.

Для присваивания используется символ «<–» или «–>» (можно также использовать традиционное « =»).

```
2 + 3 -> x -> y          # x = 5; y = 5
z <- x + y               # z = 10
z = z * z           # z = 100
z <- x + y -> t          # z = x+y и t = x+y
```

Аргументы функций передаются в круглых скобках через запятую.

```
func(x, y)          # вызов функции с двумя аргументами x и y
f(g(x, y), y)  # Суперпозиция функций f и g
```

В самом простом случае R можно использовать как «продвинутый» калькулятор:

```
> # Использование R как калькулятора
> 1 + 2 + 3
[1] 6
> exp(1)^exp(1)# e в степени e
[1] 15.15426
> sin(pi/2)
[1] 1
```

Необходимо обратить внимание на единицу в квадратных скобках ([1]) – это индекс элемента вектора. Дело в том, что в R любой результат с числами трактуется как вектор единичной длины, так как скаляров в R, вообще говоря, нет. Кроме этого, нумерация элементов векторов начинается с 1, а не с 0, как во многих других языках программирования.

Порядок арифметических действий в R стандартный, знакомый со школьной математики. Скобки (раскрывающиеся изнутри наружу) позволяют этот порядок действий менять:

Вычислить:

```
> 3/7
> 3/7-0.4285714
> sqrt(2)*sqrt(2)
> (sqrt(2)*sqrt(2))-2
```

И ещё немного о работе с аргументами на примере команды `round()` (округлить). Она имеет два аргумента: число, которое нужно округлить, и значение `digits`, сообщающее, до какого знака округлять. Система аргументов работает разумно, так что все равно, что написать:

```
> round(pi)# Использовали значение по умолчанию для "digits"
```

```
[1] 3

> round(pi, 3) # Прямой порядок аргументов
[1] 3.142

> round(pi, digits = 10)# с использованием имени аргумента
[1] 3.141593

> round(pi, d = 5)# с использованием сокращенного имени
[1] 3.14159

> round(digits = 5, pi)# вызов с другим порядком аргументов
[1] 3.14159
```

Некоторые аргументы могут иметь имена, благодаря чему их можно перечислять не по порядку, а по имени. Имена можно сокращать вплоть до одной буквы, но только если нет других аргументов, которые от такого сокращения станут неразличимы. При перечислении аргументов по порядку имена можно опускать:

Для однострочных комментариев используется символ #:

```
# Комментарий
```

Команды разделяются точкой с запятой «;» или символом перевода на новую строку:

```
1:10 -> a; mean(a);
```
или
```
1:10 -> a
mean(a)
```

Сохранение числовых и строковых значений:

```
> number <- 10 # сохранение объект
> number # выводим объект
[1] 10
> (number <- 10) # Сохраняем и выводим объект
[1] 10
> string <- "Hello" # Сохраняем объект-строку
> string
[1] "Hello"
```

Кроме строк и чисел можно также создавать и сохранять векторы. Вектор создается с помощью функции `c()`, которая объединяет несколько однотипных элементов. Также, с помощью двоеточия «:» или функции `seq()` можно создать регулярную последовательность. Функция `rep()` позволяет повторять некоторый образец.

```
> v1 <- c(2, 3, 4, 6, 10)
> v1
[1]  2  3  4  6 10
> v1[3]  # Получить третий элемент вектора
[1] 4
> v1[3:5] # Получить третий, четвертый и пятый элементы вектора
[1]  4  6 10
```

```
> v2 <- c(1:5) # Определить вектор как последовательность от 1 до 5
> v2
[1] 1 2 3 4 5

> s <- rep("a", 4)
> s
[1] "a" "a" "a" "a"

> rep(1:4, 2)
[1] 1 2 3 4 1 2 3 4
> rep(1:4, each = 2)
[1] 1 1 2 2 3 3 4 4
> (v1 + v2) * v2 # Можно проводить простые операции над векторами
[1]  3 10 21 40 75
> crossprod(v1, v2) # Вычисление скалярного произведения
     [,1]
[1,]   94
> v1[v1 > 4]# Получить все координаты вектора большие 4
[1]  6 10
```

Можно получать различные свойства векторов:

```
> length(v1) # Длина вектора
[1] 5
> mean(v1) # Среднее значение элементов вектора
[1] 5
> var(v1) # Дисперсия элементов вектора
[1] 10
```

Матрицы создаются с помощью команды `matrix()`:

```
> args(matrix)
function (data = NA, nrow = 1, ncol = 1, byrow = FALSE, dimnames =
NULL)
NULL
> matrix(data = 1:5, nrow = 5, ncol = 5, byrow = FALSE) # матрица
заполняется столбцами
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    1    1    1    1
[2,]    2    2    2    2    2
[3,]    3    3    3    3    3
[4,]    4    4    4    4    4
[5,]    5    5    5    5    5
> matrix(data = 1:5, nrow = 5, ncol = 5, byrow = TRUE) # матрица
заполняется строками
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    2    3    4    5
[2,]    1    2    3    4    5
[3,]    1    2    3    4    5
[4,]    1    2    3    4    5
[5,]    1    2    3    4    5
```

Можно создать матрицу с неопределенными значениями (для этого используется специальное обозначение NA):

```
> matrix(data = NA, nrow = 3, ncol = 3)
     [,1] [,2] [,3]
[1,]   NA   NA   NA
[2,]   NA   NA   NA
[3,]   NA   NA   NA
```

Матрицу можно также получить с помощью функций-комбинаторов `cbind` (объединяет столбцы) и `rbind`(объединяет строки).

```
> m <- cbind(v1, v2) # Создаем матрицу
> m
     v1 v2
[1,]  2  1
[2,]  3  2
[3,]  4  3
[4,]  6  4
[5,] 10  5

> typeof(m) # Получаем тип элементов матрицы
[1] "double"
> class(m) # Получаем класс объекта
[1] "matrix"
> is.matrix(m) # Проверяем, является ли m матрицей
[1] TRUE
> is.vector(m) # m не вектор
[1] FALSE
> dim(m) # Получаем размерность m
[1] 5 2
```

4. Работа с графикой в R

Основной функцией для рисования объектов в R является функция plot(x, y, …):

x – координаты точек графика, либо некоторая графическая структура, функция или объект, содержащий методы рисования.

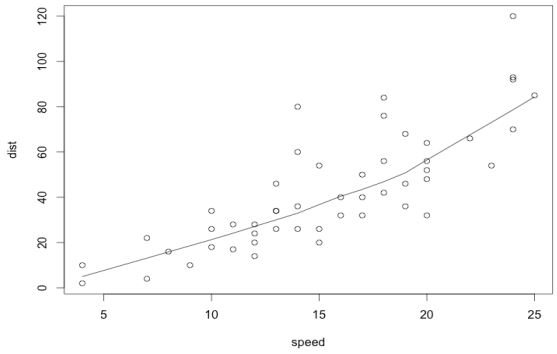
y – y-координаты точек графика, если x – соответствующего типа.

– остальные графические параметры. Перечислим некоторые из них:

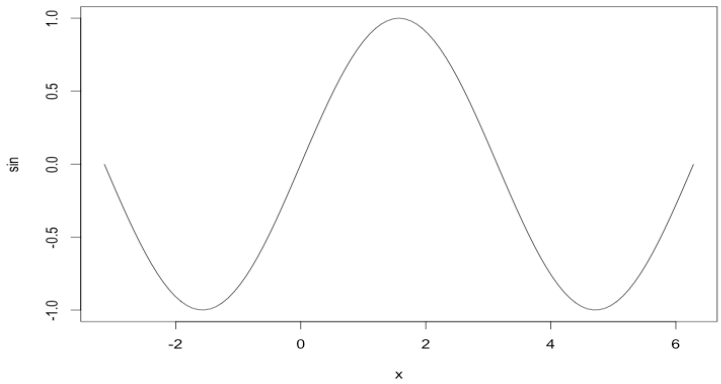
- параметр `type`позволяет изменять внешний вид точек на графике и может принимать одно из следующих значений:
  - `"p"` – точки (*points*; используется по умолчанию);
  - `"l"` – линии (*lines*);
  - `"b"` – изображаются и точки, и линии (*both points and lines*);
  - `"o"` – точки изображаются поверх линий (*points over lines*);
  - `"h"` – гистограмма (*histogram*);
  - `"s"` – ступенчатая кривая (*steps*);
  - `"n"` – данные не отображаются (*no points*).
- параметры `xlab`и `ylab`задают название осей абсцисс и ординат, соответственно;
- параметр `main`задаёт заголовок графика.

Примеры.

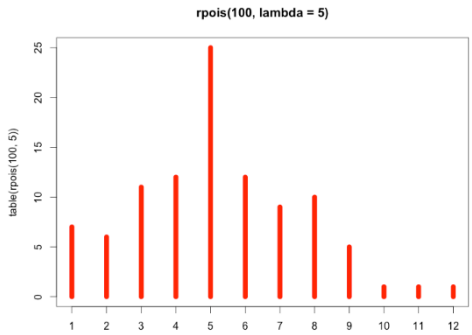
```
>require(stats) # for lowess, rpois, rnorm
plot(cars)
lines(lowess(cars))
```



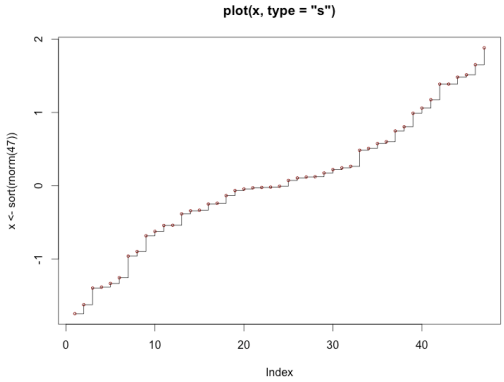

```
>plot(sin, -pi, 2*pi)
```




```
## Discrete Distribution Plot:
> plot(table(rpois(100, 5)), type = "h", col = "red", lwd = 10,
       main = "rpois(100, lambda = 5)")
```




```
## Simple quantiles/ECDF, see ecdf() {library(stats)} for a better
## one:
>plot(x <- sort(rnorm(47)), type = "s", main = "plot(x, type =
\"s\")")
points(x, cex = .5, col = "dark red")
```



Как видно из примеров, с помощью функции plot() можно создавать большое количество разнообразных графиков.

Задание и порядок выполнения лабораторной работы №1

1. Установить R на ПК – https://cran.rstudio.com
2. Установить RStudio – инсталлятор скачать с официального сайта проекта - https://www.rstudio.com/products/rstudio/download3/
3. Ознакомиться с кратким руководством пользователя RStudio – http://r-analytics.blogspot.ch/p/rstudio.html#.WAPrteuvUbf/
4. Исследовать команду `'demo()'`, полученные результаты вставить в отчет
5. Исследовать основные функции и команды языка R, представленные в данной лабораторной работе, полученные результаты вставить в отчет.
6. Ответить на контрольные вопросы

Контрольные вопросы

1. Особенности языка R.
2. Команда для получения подробной информации о функции в R.
3. Структура и особенности команды `round()`в R.
4. Команды для работы с векторами в R (изучить команды, не представленные в методических указаниях).
5. Команды для работы с матрицами в R (изучить команды, не представленные в методических указаниях).
6. Работа с графикой в R (изучить команды, не представленные в методических указаниях).

Библиография

1. Алексей Шипунов и др. Наглядная статистика. Используем R! – М.: ДМК Пресс, 2014. – 298 с. [Электронный ресурс]. Режим доступа: http://ashipunov.info/shipunov/school/books/rbook.pdf.
2. Зарядов И.С. Введение в статистический пакет R: типы переменных, структуры данных, чтение и запись информации, графика. М.: Издательство Российского университета дружбы народов, 2010. – 207 с.
3. Оффициальный сайт RStudio. Режим доступа: https://www.rstudio.com.
4. Профессиональный информационно-аналитический ресурс, посвященный машинному обучению, распознаванию образов и интеллектуальному анализу данных [Электронный ресурс]. Режим доступа: http://machinelearning.ru.