Основы работы с R. Обработка статистических данных

Варвара Кожухова

5 октября 2017 г.

Задание

1. Загрузить данные для своего варианта в переменную-вектор.
2. Получить справочную информацию по своим данным, просмотреть их содержимое.
3. Проверить, есть ли среди данных пропуски.
4. Создать новую переменную-вектор, в которой будут 1, если значение в исходном векторе больше среднего, и –1 если значение переменной меньше среднего и 0 если значение равно среднему.
5. Вывести описательную статистику.
6. Построить графики абсолютных частот и плотности распределения.

Варианты

№ варианта	1	2	3	4	5	6	7	8	9	10
Набор данных	CO2		ChickWeight		Orange		airquality		faithful	
Имя вектора	conc	uptake	weight	Time	age	circumference	Wind	Temp	eruptions	waiting

Первые шаги в R

Рассмотрим основные выражения в R: числа, строки и логические переменные. Можно использовать R как калькулятор, например:

```
1+1
```

```
## [1] 2
```

```
6*7
```

```
## [1] 42
```

```
sqrt(16)
```

```
## [1] 4
```

Строки печатаются в кавычках: двойных или одинарных.

```
"Hello world!"
```

```
## [1] "Hello world!"
```

```
'Hello world!'
```

```
## [1] "Hello world!"
```

Логические выражения возвращают TRUE или FALSE. Чтобы сравнить два выражения, используется двойной знак равенства. Как и в других языках программирования, можно сохранять значения в переменную. Сохраним 42 в переменную x, а 5 – в переменную X, но в обратную сторону. Можно так же повторно назначить любое значение переменной в любое время. R чувствителен к регистру: переменные x и X – это разные переменные.

```
3 < 4

## [1] TRUE

2 + 2 == 5

## [1] FALSE

x <- 42
5 -> X
```

Чтобы вызвать функцию, нужно обратиться к ней по имени, указав в скобках нужные аргументы.

```
sum(1, 3, 5)

## [1] 9
```

Получить помощь по функции можно командой help(functionname) или ?functionname.

Зададим вектор y с помощью функции c (сокр. от англ. Combine). NA – это пропущенное наблюдение (от англ. Not Available). Его не следует путать с NaN (Not a Number – «не число», неопределенность). Попробуем просуммировать элементы вектора y. Необязательным аргументом функции sum является na.rm (сокр. от англ. Remove NA), по умолчанию равный FALSE. Если указать для него значение «истина», то функция суммы будет складывать все элементы вектора, исключая пропущенные.

```
y <- c(-3, 2, NA, 5)
y

## [1] -3  2 NA  5

0/0

## [1] NaN

sum(y); sum(y, na.rm = TRUE)

## [1] NA
## [1] 4
```

Последовательность чисел можно задать двумя способами: start:end либо функцией seq(). Обращаться к элементам вектора можно, используя квадратные скобки, либо можно задать элементам вектора имена.

```
5:9 ; seq(5,9)

## [1] 5 6 7 8 9
## [1] 5 6 7 8 9

seq(10,50, by = 10)
```

```
## [1] 10 20 30 40 50

sentence <- c('mack', 'the', 'knife')
sentence[3]

## [1] "knife"

sentence[c(1,3)]

## [1] "mack"  "knife"

ranks <- 1:3
names(ranks) <- c("first", "second", "third")
ranks

##  first second  third
##      1      2      3

ranks["first"]

## first
##     1
```

В основном в R работают с наборами данных. Такая структура носит в R название data.frame и представляет собой таблицу, в которой каждый столбец – это некоторая переменная, а каждая строка – это одно наблюдение. Создадим в режиме скрипта data.frame. Пусть имеются наблюдения за ростом и весом некоторых людей. Зададим два вектора:

```
rost <- c(160, 175, 155, 190, NA)
ves <- c(NA, 70, 48, 85, 60)
```

И объединим их в набор данных, который поместим в переменную df, а затем выведем на экран. Обращаться к конкретным наблюдениям df можно, используя квадратные скобки.

```
df <- data.frame(rost, ves)
df

##   rost ves
## 1  160  NA
## 2  175  70
## 3  155  48
## 4  190  85
## 5   NA  60

df[3,1]

## [1] 155
```

Обращаться к переменным можно, используя знак $ или указывая столбец с пропуском номера строки.Обращаться к наблюдениям можно, указывая конкретную строку и пропуская номер столбца.

```
df$rost

## [1] 160 175 155 190  NA

df[ ,1] ; df[4,]
```

```
## [1] 160 175 155 190  NA
##   rost ves
## 4  190  85
```

Основные описательные статистики (среднее, стандартное отклонение и медиану) можно получить с помощью функций mean, sd и median.

```
mean(df$rost, na.rm = T)

## [1] 170

sd(df$rost, na.rm = T)

## [1] 15.81139

median(df$rost, na.rm = T)

## [1] 167.5
```

Подключим дополнительные пакеты для работы со статистикой.

```
library("psych")# описательные статистики
library("lmtest") # тестирование гипотез в линейных моделях
library("ggplot2")# графики
library("dplyr") # манипуляции с данными
library("MASS") # подгонка распределений
```

Поместим в переменную *d* встроенный в R набор данных по автомобилям. В этом наборе данных 50 наблюдений и две переменных (скорость, миль/час и длина тормозного пути в футах). Теперь d имеет тип данных data.frame (набор данных). Командой glimpse можно посмотреть на этот набор данных, в результате чего будут перечислены все переменные и типы данных. Переменные speed и dist имеют тип данных dbl (double) и содержат по 50 наблюдений. Для других типов данных используются следующие сокращения: chr (character/string), int (integer), fctr (factor), tims (time), lgl (logical).

```
d <- cars
glimpse(d)

## Observations: 50
## Variables: 2
## $ speed <dbl> 4, 4, 7, 7, 8, 9, 10, 10, 10, 11, 11, 12, 12, 12, 12, 13, 13, 13, 13, 1...
## $ dist  <dbl> 2, 10, 4, 22, 16, 10, 18, 26, 34, 17, 28, 14, 20, 24, 28, 26, 34, 34, 4...
```

Посмотрим на первые шесть и последние шесть наблюдений набора данных d ("голова"и "хвост"набора данных).

```
head(d)

##   speed dist
## 1     4    2
## 2     4   10
## 3     7    4
## 4     7   22
## 5     8   16
## 6     9   10
```

```
tail(d)

##    speed dist
## 45    23   54
## 46    24   70
## 47    24   92
## 48    24   93
## 49    24  120
## 50    25   85
```

Получим таблицу с описательными статистиками: среднее, мода, медиана, стандартное отклонение, минимум/максимум, асимметрия, эксцесс и т.д.

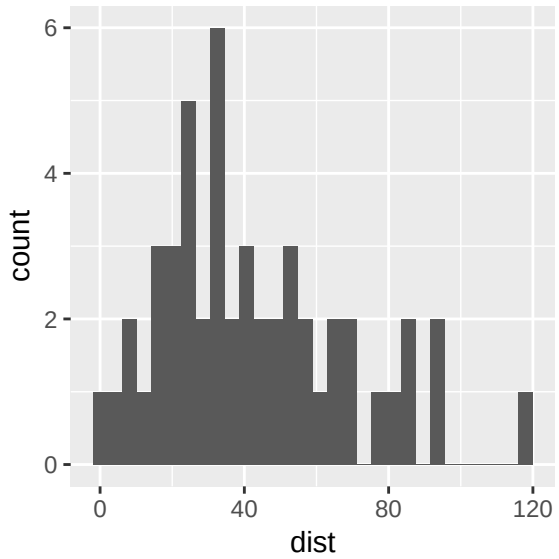
```
describe(d) # функция из пакета psych

##       vars  n  mean    sd median trimmed   mad min max range  skew kurtosis   se
## speed    1 50 15.40  5.29     15   15.47  5.93   4  25    21 -0.11    -0.67 0.75
## dist     2 50 42.98 25.77     36   40.88 23.72   2 120   118  0.76     0.12 3.64
```

Построим гистограмму абсолютных частот для переменной dist (длины тормозного пути). Воспользуемся функцией qplot, задав источник данных d (аргумент data), переменную для построения графика (dist), подпишем оси (параметры функции xlab и ylab) и название графика (параметр main).

```
qplot(data=d, dist) # функция из пакета ggplot2

## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Можно построить так же гистограмму плотности распределения.

```
# функция из базового пакета graphics
p <- hist(d$dist, probability = TRUE, col="grey")
```

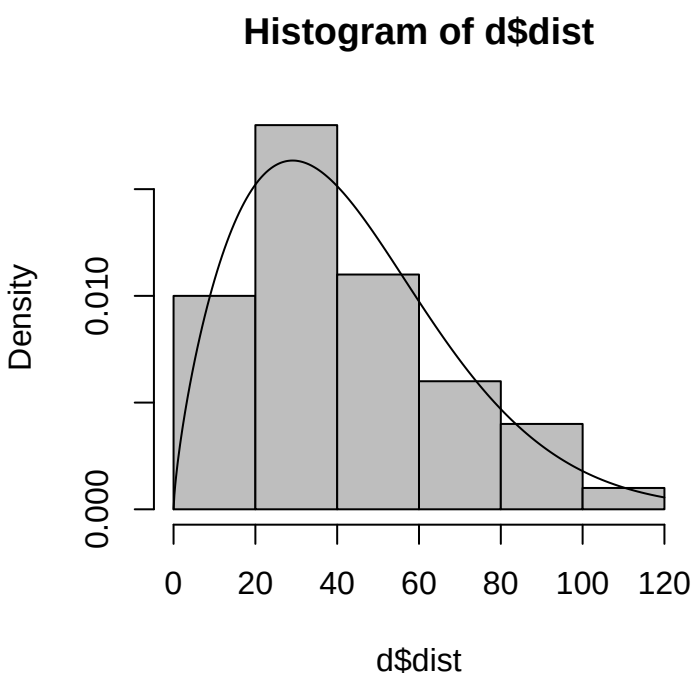
Подгоним плотность распределения Вейбулла, поместив результат (оценки параметров распределения) в переменную fit. Переменная fit теперь представляет собой список (List) из 5 элементов. Доступ к элементам списка можно получить через значок доллара $. Оценки двух параметров распределения Вейбулла были рассчитаны методом максимального правдоподобия. Просмотрим их, обратившись к элементу списка fit.

```
fit <- fitdistr(d$dist, "weibull") # функция из пакета MASS
fit$estimate

##    shape    scale
##  1.72371 48.15234
```

Покажем на том же графике теоретическую плотность распределения Вейбулла. Первый аргумент функции lines – это значения по оси абсцисс, на основе которых будет построен график. Далее указывается функция плотности dweibull. Для нее нужно указать значения аргумента для расчета и значения двух параметров распределения: коэффициент формы (shape) и масштаба (scale).

```
p <- hist(d$dist, probability = TRUE, col="grey")
xvals <- seq(0, 120, .20)  # значения по оси абсцисс от 0 до 120 с шагом 0,2
lines(xvals, dweibull(xvals, shape=fit$estimate[1], scale=fit$estimate[2]))
```


Histogram of d$dist
Density
0.000
0.010
0
20
40
60
80
100
120
d$dist