hw1

Korotkov Vitaliy Konstantinovich

2024-10-19

Работа с данными

Загрузка данных

```
data.df <-
read.table("http://people.math.umass.edu/~anna/Stat597AFall2016/rnf6080.dat",
header=FALSE)
cat("Количество строк: ", nrow(data.df), "\nКоличество столбцов: ",
ncol(data.df))
```

```
## Количество строк:  5070
## Количество столбцов:  27
```

Имена колонок

```
colnames(data.df)
```

```
##  [1] "V1"  "V2"  "V3"  "V4"  "V5"  "V6"  "V7"  "V8"  "V9"  "V10" "V11" "V12"
## [13] "V13" "V14" "V15" "V16" "V17" "V18" "V19" "V20" "V21" "V22" "V23" "V24"
## [25] "V25" "V26" "V27"
```

Значение 5 строки 7 столбца

```
data.df[5, 7]
```

```
## [1] 0
```

Вторая строка

```
data.df[2, ]
```

```
##   V1 V2 V3 V4 V5 V6 V7 V8 V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21
## 2 60  4  2  0  0  0  0  0  0   0   0   0   0   0   0   0   0   0   0   0   0
##   V22 V23 V24 V25 V26 V27
## 2   0   0   0   0   0   0
```

Замена заголовков столбцов

```
names(data.df) <- c("year", "month", "day", seq(0, 23))
```

Данная строка заменяет заголовки столбцов на “year”, “month”, “day”, и 0, 1, 2, ..., 23, которые соответствуют часу дня, в который были зафиксированы осадки.

Начало таблицы:

```
head(data.df)
```

```
##   year month day 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23
## 1   60     4   1 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
```

```
## 2   60     4   2 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 3   60     4   3 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 4   60     4   4 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 5   60     4   5 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 6   60     4   6 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
```

Конец таблицы:

```
tail(data.df)
```

```
##      year month day 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
22
## 5065   80    11  25 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 5066   80    11  26 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 5067   80    11  27 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 5068   80    11  28 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 5069   80    11  29 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 5070   80    11  30 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0
0
##      23
## 5065  0
## 5066  0
## 5067  0
## 5068  0
## 5069  0
## 5070  0
```

Последние 24 колонки представляют собой количество осадков по каждому часу дня.

Добавление колонки и построение гистограммы

Добавим новую колонку `daily`, которая будет содержать сумму осадков за день (по всем часам):

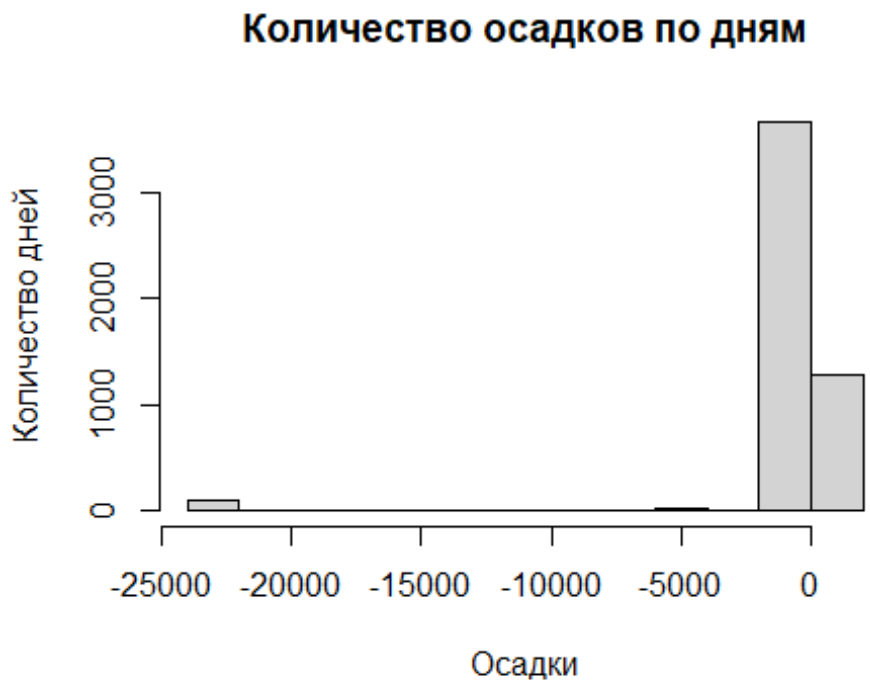
```
data.df$daily <- rowSums(data.df[, 4:27])
head(data.df)
```

```
##   year month day 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22
23
## 1   60     4   1 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 2   60     4   2 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 3   60     4   3 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
```

```
0
## 4   60     4   4 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 5   60     4   5 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
## 6   60     4   6 0 0 0 0 0 0 0 0 0 0  0  0  0  0  0  0  0  0  0  0  0  0  0
0
##   daily
## 1     0
## 2     0
## 3     0
## 4     0
## 5     0
## 6     0
```

Построим гистограмму для новой колонки `daily`:

```
hist(data.df$daily, main = "Количество осадков по дням", xlab = "Осадки", ylab =
"Количество дней")
```



Мы видим, что в данных есть некорректные значения (-999), что приводит к неправильной интерпретации гистограммы.

Исправление ошибок в данных и создание нового датафрейма

Удалим некорректные значения (-999):

```
fixed.df <- data.df[data.df$daily > 0, ]
```

Построим гистограмму для исправленного датафрейма:

```r
hist(fixed.df$daily, main = "Количество осадков по дням (исправленные данные)",
xlab = "Осадки", ylab = "Количество дней", breaks = 80)
```

Новая гистограмма более корректна, так как она построена на данных без ошибочных значений.

Синтаксис и типизирование

Пример 1

```r
v <- c("4", "8", "15", "16", "23", "42")
max(v)
```

```
## [1] "8"
```

Функция `max()` ищет максимальное значение вектора. Поскольку все элементы вектора являются строками, они сравниваются лексикографически, по первому символу.

```r
sort(v)
```

```
## [1] "15" "16" "23" "4"  "42" "8"
```

Функция `sort()` сортирует строки в алфавитном порядке, опять же лексикографически.

```r
# sum(v)
```

Функция `sum()` не работает для строковых векторов, выдаст ошибку.

Пример 2

```r
# v2 <- c("5", 7, 12)
# v2[2] + v2[3]
```

Хотя элементы вектора представлены как числа, вектор смешанного типа автоматически преобразуется в строковый. Следовательно, сложение вызовет ошибку.

```
df3 <- data.frame(z1="5", z2=7, z3=12)
df3[1, 2] + df3[1, 3]

## [1] 19
```

В данном примере элементы второго и третьего столбца могут быть сложены, так как они являются числовыми.

```
l4 <- list(z1="6", z2=42, z3="49", z4=126)
l4[[2]] + l4[[4]]

## [1] 168
```

Сложение значений из второго и четвертого элементов списка работает, так как оба являются числами.

```
# l4[2] + l4[4]
```

Это вызовет ошибку, так как обращение `l4[2]` и `l4[4]` возвращает список, а не числовые значения.

Работа с функциями и операторами

Последовательности чисел

Числа от 1 до 10000 с инкрементом 372:

```
seq(1, 10000, by=372)

##  [1]    1  373  745 1117 1489 1861 2233 2605 2977 3349 3721 4093 4465 4837
5209
## [16] 5581 5953 6325 6697 7069 7441 7813 8185 8557 8929 9301 9673
```

Числа от 1 до 10000 длиной 50:

```
seq(1, 10000, length.out=50)

##  [1]     1.0000   205.0612   409.1224   613.1837   817.2449  1021.3061
##  [7]  1225.3673  1429.4286  1633.4898  1837.5510  2041.6122  2245.6735
## [13]  2449.7347  2653.7959  2857.8571  3061.9184  3265.9796  3470.0408
## [19]  3674.1020  3878.1633  4082.2245  4286.2857  4490.3469  4694.4082
## [25]  4898.4694  5102.5306  5306.5918  5510.6531  5714.7143  5918.7755
## [31]  6122.8367  6326.8980  6530.9592  6735.0204  6939.0816  7143.1429
## [37]  7347.2041  7551.2653  7755.3265  7959.3878  8163.4490  8367.5102
## [43]  8571.5714  8775.6327  8979.6939  9183.7551  9387.8163  9591.8776
## [49]  9795.9388 10000.0000
```

Функция rep()

```
rep(1:5, times=3)

##  [1] 1 2 3 4 5 1 2 3 4 5 1 2 3 4 5
```

Этот код повторяет весь вектор `1:5` три раза.

```
rep(1:5, each=3)
```

```
##  [1] 1 1 1 2 2 2 3 3 3 4 4 4 5 5 5
```

Этот код повторяет каждый элемент вектора три раза.