

Работа с наборами данных

- 1 Типы данных
- 2 Манипулирование прямоугольными данными в R
- 3 Задание

Существует два основных типа структурированных данных: *количественный (числовой)* и *качественный*.


В количественных данных выделяют две формы: *непрерывные числовые данные*, *дискретные числовые данные*.


В качественных данных выделяют три формы: *категориальные (атрибутивные) данные*, *двоичные (альтернативные) данные*, *порядковые данные*.


Ниже приведено их определение.

- Непрерывные данные (continuous): данные, которые могут принимать любое значение в интервале. Так, любые числа из интервала $(0, 1)$, например, 0.5674, являются непрерывными данными. Синонимы: интервал, число с плавающей точкой, числовое значение.
- Дискретные данные (discrete): данные, которые могут принимать только целочисленные значения. Например, числа $\{-3, -2, -1, 0, 1, 2, 3, 4, 5\}$ являются дискретным набором данных. Синонимы: целое число, количество.
- Категориальные (атрибутивные) данные (categorical): данные, которые могут принимать только определенный набор значений, в частности набор возможных категорий. Например, название городов, регионов - это категориальные данные. Синонимы: перечисления, факторы.
- Двоичные (альтернативные) данные (binary): особый случай категориальных данных всего с двумя категориями значений (0/1, истина/ложь). Например, мужчина/женщина; брак/небрак. Синонимы: дихотомические, логические, булевы данные.
- Порядковые данные (ordinal): категориальные данные, которые допускают естественное упорядочение. Например, набор воинских званий (лейтенант, старший лейтенант, капитан, майор, подполковник, полковник), (токарь 1 разряда, токарь 2 разряда, ... токарь 6 разряда). Синонимы: порядковый фактор.

R



Класс языка	мультипарадигмальный
Тип исполнения	интерпретируемый
Появился в	1993 ^[1]
Автор	Росс Айхэка Роберт Джентлмен
Выпуск	4.0.2 (22 июня 2020) ^[3]
Система типов	динамическая
Испытал влияние	S, Scheme
Лицензия	GNU GPL 2 ^[4]
Сайт	r-project.org  (англ.)
ОС	GNU/Linux ^{[d][5]} , Microsoft Windows ^[5] , macOS ^[5] и BSD ^{[d][5]}

 [Медиафайлы на Викискладе](#)

Для работы с R будем использовать [RStudio Cloud](#)

Учет типов данных в R позволяет определить, каким образом программная среда будет обращаться с вычислениями над этими данными.

- 1 В R порядковые данные могут быть представлены как порядковый тип данных: `ordered.factor`, сохраняя определенную пользователем упорядоченность на графиках, таблицах, моделях.
- 2 Учет типов данных может позволить оптимизировать хранение и индексацию данных.
- 3 Часто импорт данных, например, с помощью `read.csv`, ведет к автоматическому преобразованию столбцов текста в тип данных `factor`. Последующие операции на этом столбце будут исходить из предположения, что единственными допустимыми значениями для этого столбца являются значения, которые были импортированы первоначально, и присвоение нового значения выдаст предупреждение об отсутствии значения `NA` (not available).

Прямоугольные данные в R

Прямоугольные данные представляют собой двумерную матрицу, в которой строки обозначают описание *объекта (записи)* по столбцам - *признакам (синонимы: атрибут, вход, предиктор, переменная)*. Сам набор прямоугольных данных в R называется фреймом (`data.frame`). Чтение прямоугольных (табличных) данных во фрейм осуществляется рядом команд:

- При чтении данных из формата `csv` (текстовый формат с заданными разделителями):

```
D<-read.csv('File.csv',header=TRUE,sep=',')
```

Данным оператором переменная `D` становится фреймом данных из файла `File.csv`, параметр `header` указывает, что признаки (столбцы) имеют имена, в качестве разделителя используется запятая.

- При чтении данных из формата `Excel`:

```
library(readxl)
```

```
D<-read_excel("File.xlsx", sheet = "Name_sheet")
```

Здесь первая строка позволяет активировать библиотеку по работе с `Excel`. Вторым оператором переменная `D` становится фреймом данных из файла `File.xlsx`, параметр `sheet` указывает, с какого листа книги `Excel` загружать данные.

- Также можно читать данные, хранящиеся удаленно в сети. Для этого нужно узнать url сетевого ресурса. После этого загрузку можно осуществить командой:

```
D<-read.csv("https://raw.githubusercontent.com/junaart/DataForR/master/testcsv.csv", sep=";", header=T)
```

Данным оператором переменная D становится фреймом данных по ссылке из сетевого ресурса.

- Следующая команда позволяет отобразить фрейм данных:

```
View(D)
```

- Еще одна полезная команда:

```
attach(D)
```

позволяет разбить фрейм на части. После ее выполнения вводятся переменные, имена которых совпадают с именами столбцов фрейма. Сами эти переменные являются вектором из значений соответствующего столбца.

- Также можно заворачивать различные объекты одинаковой размерности (по количеству записей) во фрейм данных с его последующим сохранением. Ниже представлены соответствующие примеры.

Примеры работы с прямоугольными данными

```
1 #Вводим два вектора одинаковой размерности
2 t<-c(1,2,3,4,6)
3 q<-c(4,5,6,7,8)
4
5 #Заворачиваем эти векторы во фрейм
6 s<-data.frame(t,q)
7
8 #Отображаем фрейм данных
9 View(s)
10
11 #Получаем список названий столбцов фрейма
12 colnames(s)
13
14 #Меняем названия столбцов фрейма
15 colnames(s)<-c("first","second")
16
17 #Разбиваем фрейм на столбцы
18 attach(s)
19
20 #Заворачиваем данные в новый фрейм
21 d<-data.frame(first,second, first+second)
22
23 #Отображаем фрейм данных
24 View(d)
25
26 #Сохраняем новый фрейм
27 write.csv(d, 'MyData.csv', row.names = FALSE)
28 |
```

Еще пример работы с прямоугольными данными (продолжение)

```
1 #Заполняем дата-фрейм данными
2 q<-data.frame(c("a","b","c","d","e","f","g","h"),c(2,6,7,9,1,2,6,4),c(3, 6, 8, 9, 0, 4, 2, 2))
3 colnames(q)<-c("first","second","third")
4
5 #Получение столбца с именем "first"
6 q$first
7
8 #Отбираем только значения второго столбца, которые больше 4
9 q$second[q$second>4]
10
11 #Отбираем данные, которые в третьем столбце не равны нулю
12 q$third[q$third!=0]
13
14 #Суммируем и умножаем элементы второго столбца
15 sum(q$second)
16 prod(q$second)
17
18 #Находим произведение элементов второго и третьего столбца
19 q$second*q$third
20
21 #Берем элемент 1 строки, второго столбца:
22 q[1,2]
23 q$second[1]
24
25 #Строим график: по оси x располагаются элементы первого столбца фрейма, по оси y - элементы второго столбца
26 plot(q$first,q$second)
27
28 #Перебираем все данные первого столбца
29 for(i in q$first) print(i)
30
31 #Выводим данные первого столбца, только если элементы второго столбца в диапазоне [6;9]
32 for(i in seq(1:8)){if (q[i,2]>=6 & q[i,2]<=9) {print(q[i,1])}}
33
34 #Смотрим на тип данного
35 str(q$first)
36 >Factor w/ 8 levels "a","b","c","d",...: 1 2 3 4 5 6 7 8
37
38 #Скапливаем в result только те элементы первого столбца, для которых в третьем столбце значения меньше 3
39 result<-c(); for (i in seq(1:8)){if (q$third[i]<3) result<-c(result,as.vector(q$first[i]))}
40
```

Задание по манипулированию данными

- 1 Загрузить фрейм данных «demo26.xlsx» с сайта <https://github.com/junaart/DataForR>
- 2 Создать фрейм данных, транспонируя исходный фрейм данных, присвоив имена столбцов «FIRST»; «SECOND»; «THIRD»; «FOURTH»; «FIFTH».
- 3 Отобрать года, для которых значение во втором столбце больше среднего значения этого столбца.
- 4 Вывести на экран все строки фрейма за четные года.
- 5 Построить график по оси x - года, по оси y значения третьего столбца.
- 6 Добавить к фрейму новый столбец категориальных данных по принципу: если значение третьего столбца в строке больше медианного значения этого столбца и значение четвертого столбца этой строки больше медианного значения этого столбца, то в новый столбец пишем «good», в противном случае пишем «bad».
- 7 Сохранить полученный фрейм с именем «Test.csv», загрузить его на github