Основы программирования в R

Основы работы с датафреймами

Алла Тамбовцева, НИУ ВШЭ

Содержание

Загрузим данные из файла `tree` с учетом кодировки и вида ячеек с пропущенными значениями:

```
tree <- read.csv("/Users/allat/Desktop/firtree.csv",
                 encoding = "UTF-8", na.strings = "")
```

Выбор столбцов и строк датафрейма по названию и номеру

Как мы уже видели, столбец из датафрейма можно выбрать, воспользовавшись оператором `$` и указав после него название столбца:

```
head(tree$wish)  # первые несколько значений
```

```
## [1] да  нет да  да  нет да
## Levels: да нет
```

Мы можем «закрепить» датафрейм с помощью функции `attach()`, чтобы обращаться к переменным более простым способом:

```
attach(tree)
head(wish)  # имя переменной как есть, без $
```

```
## [1] да  нет да  да  нет да
## Levels: да нет
```

Однако это не всегда удобно, особенно если приходится работать с несколькими датафреймами одновременно (наложение переменных с одинаковыми именами, проблемы с редактированием и прочее). Вернем обратно:

```
detach(tree)
head(wish)  # теперь эта строчка выдает ошибку
```

Столбец можно выбрать и по номеру, главное помнить, что нумерация в R начинается с 1:

```
# третий столбец
tree[3]
```

Если в квадратных скобках после названия датафрейма указано одно число, R автоматически распознает его как номер столбца. В более общем случае указываются два индекса через запятую: номер строки и номер столбца. Если нас интересует только третий столбец, можем на месте строки ничего не писать:

```
tree[, 3]
```

Можем выбрать несколько столбцов подряд и сохранить их в новый датафрейм:

```
tree2 <- tree[, 3:5]
```

В R, в отличие от Python, правый конец среза тоже включается, поэтому в примере выше в `tree2` мы сохранили третий, четвертый и пятый столбец из `tree`.

Если выбираем столбцы не подряд, обязательно их номера нужно оформить в виде вектора:

```
tree[c(3, 5)]
```

В противном случае получится совсем не то:

```
tree[3, 5]
```

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

Это «совсем не то» связано с тем, что, когда мы указываем в квадратных скобках числа через запятую, R воспринимает первое число как номер строки, второе число — как номер столбца (как в матрицах — сначала строка, потом столбец). Можем посмотреть на датафрейм и убедиться в этом.

Если нас интересует определенная строка, то ее индекс стоит указать на первом месте, а место для индекса столбца оставить пустым:

```
tree[3, ]
```

```
##   X gender      ftype height score expenses wish
## 3 3 female сосна Крым    248     4      655   да
```

Можно совместить — выбрать сразу несколько строк и несколько столбцов:

```
tree[1:12, 2:4]
```

Названия столбцов и строк и их переименование

Чтобы получить вектор названий столбцов датафрейма, нам понадобится функция `colnames()`:

```
colnames(tree)
```

```
## [1] "X"        "gender"   "ftype"    "height"   "score"    "expenses" "wish"
```

Этот вектор можно использовать для переименования столбцов. Переименуем первый столбец — назовем его `id`:

```
colnames(tree)[1] <- "id"
colnames(tree)
```

```
## [1] "id"       "gender"   "ftype"    "height"   "score"    "expenses" "wish"
```

Для строк есть похожая функция — `rownames()`:

```
rownames(tree)
```

Фильтрация строк по условиям

Если хотим отобрать из датафрейма определенные наблюдения, это тоже можно сделать с помощью квадратных скобок. Например, мы хотим отобрать строки, соответствующие деревьям, оцененным выше, чем на 3 балла, и сохранить их в датафрейм `good`:

```
good <- tree[tree$score > 3,]
```

Тут важно не забыть поставить запятую, чтобы R понимал, что мы накладываем условие на строки, а столбцы берем все.

Теперь выберем строки, соответствующие деревьям, которые респонденты хотели бы себе в подарок, и выведем их на экран в отдельной вкладке через `View()`:

```
View(tree[tree$wish == "да",])
```

Для примера одновременно отберем только третий и четвертый столбцы:

```
View(tree[tree$wish == "да", 3:4])
```

Как обычно, для объединения условий используются операторы `&` (одновременные условия) и `|` (хотя бы одно верно). Выберем строки, соответствующие деревьям, за которые посетители готовы заплатить более 1500 рублей, и которые оценили выше, чем на 4:

```
tree[tree$expenses > 1500 & tree$score > 4,]
```

А теперь выберем только сосны, либо сосны Крым, либо датские сосны:

```
tree[tree$ftype == "сосна Крым" | tree$ftype == "сосна датская",]
```

Удаление строк с пропущенными значениями

Напоследок избавимся от строк с пропущенными значениями:

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