Основы программирования в R

Работа с данными с `tidyverse`: часть 2

Алла Тамбовцева, НИУ ВШЭ

Содержание

Описание данных

Для работы загрузим файл `food_coded.csv`, который содержит результаты опроса студентов колледжа, посвященного их пищевым привычкам (любимая еда, еда для восстановления душевного спокойствия, любимая национальная кухня и прочее). Полное описание данных можно найти на сайте Kaggle по ссылке.

```
food <- read.csv("https://allatambov.github.io/rprog/data/food_coded.csv")
```

И, конечно, загрузим библиотеку `tidyverse`:

```
library(tidyverse)
```

При работе с данными мы часто сталкиваемся с тем, что нам нужно получить какую-то сводную информацию по переменным. Для этого существует функция `summarise()`. Попробуем пока получить общее число строк в датафрейме:

```
food %>% summarise(n())
```

```
##   n()
## 1 125
```

Функция `n()` используется для подчета числа строк (наблюдений). Вместо нее можно использовать функцию `tally()`, причем отдельно, без `summarise()`:

```
food %>% tally
```

```
##     n
## 1 125
```

Перейдем к примерам поинтереснее. Посчитаем среднее число фруктов в день, которое потребляют студенты.

```
food %>% summarise(mean(fruit_day))
```

```
##   mean(fruit_day)
## 1           4.224
```

А теперь посчитаем как среднее число фруктов в день, так и медианное, плюс, дадим столбцам с посчитанными показателями названия `mean` и `median`:

```
food %>% summarise(mean = mean(fruit_day),
                   median = median(fruit_day))
```

```
##    mean median
## 1 4.224      5
```

Названия столбцов могут быть написаны на кириллице и содержать пробелы, тогда вокруг названия необходимо поставить апострофы.

Группировка и агрегирование

Часто необходимо получить сводную информацию не по всем наблюдениям в датафрейме, а по определенной группе. Для этого сначала нужно сгруппировать данные, основываясь на значениях какой-нибудь переменной. Воспользуемся функцией `group_by()` и посмотрим на среднее число фруктов в день с группировкой по полу (`Gender`):

```
food %>% group_by(Gender) %>% summarise(mean(fruit_day))
```

```
## # A tibble: 2 x 2
##   Gender `mean(fruit_day)`
##    <int>             <dbl>
## 1      1              4.36
## 2      2              4.02
```

Обычно при описании данных указывают среднее значение и стандартное отклонение показателей. Давайте тоже так поступим:

```
food %>% group_by(Gender) %>% summarise(mean = mean(fruit_day),
                                        sd = sd(fruit_day))
```

```
## # A tibble: 2 x 3
##   Gender  mean    sd
##    <int> <dbl> <dbl>
## 1      1  4.36 0.844
## 2      2  4.02 1.01
```

А теперь попробуем назвать столбцы в `summarise()` на русском и с пробелами:

```
food %>% group_by(Gender) %>%
  summarise(`Среднее` = mean(fruit_day),
            `Стандартное отклонение` = sd(fruit_day))
```

```
## # A tibble: 2 x 3
##   Gender Среднее `Стандартное отклонение`
##    <int>   <dbl>                    <dbl>
## 1      1    4.36                    0.844
## 2      2    4.02                    1.01
```

Никто не мешает группировать данные по нескольким показателям сразу. Все необходимые показатели группировки указываются через запятую внутри `group_by()`. Сгруппируем данные по полу и курсу и выясним, сколько студентов мужского и женского пола с разных курсов было опрошено:

```
food %>% group_by(Gender, grade_level) %>%
  summarise(n = n())
```

```
## # A tibble: 8 x 3
## # Groups:   Gender [2]
##   Gender grade_level     n
##    <int>       <int> <int>
## 1      1           1    21
## 2      1           2    21
## 3      1           3    17
## 4      1           4    17
## 5      2           1    16
```

```
## 6      2           2    11
## 7      2           3    11
## 8      2           4    11
```

Посмотрим на результат группировки в отдельном окне (актуально для большого числа групп, когда в консоли смотреть на результаты неудобно):

```
food %>% group_by(Gender, grade_level) %>% summarise(n = n()) %>% View
```

При желании можем сохранить результат в переменную. Давайте для разнообразия добавим оператор присваивания в самом конце и напишем его в другую сторону `->` (да, так тоже можно):

```
# сохранили в dat_sum

food %>% group_by(Gender, grade_level) %>% summarise(n = n()) -> dat_sum
```

Напоследок проверим, что использовать несколько последовательных `summarise()` тоже можно. Посчитаем сначала численность каждой группы по полу и курсу, а потом выведем максимум из полученных значений:

```
food %>% group_by(Gender, grade_level) %>%
  summarise(n = n()) %>% summarise(max(n))
```

```
## # A tibble: 2 x 2
##   Gender `max(n)`
##    <int>    <dbl>
## 1      1       21
## 2      2       16
```