2021年数理统计上机课

–字符与字符串

黄启岳

北京师范大学统计学院

2021 年 3 月 10 日

目录

1. 字符型变量
1. 字符数统计
1. 字符串连接
1. 字符串截取
1. 字符串拆分
1. 字符匹配
1. 字符替换
1. 字符输出
1. 执行字符串

字符型变量

DEFINITION (字符)

字符型变量为类字形单位或符号，一组字符组成了字符串，输入时使用""。

举例：c("Hello the world","3278")

准备工作

以下操作需要使用R程序包stringr。这是一种专门用来处理字符串的程序包。

字符数统计

R语言自带函数nchar()，可以输入向量、矩阵，自动返回每一个位置的字符数。

stringr包中的函数str_length()于计算字符串长度的函数，输入的数据类型为向量。也可以输入矩阵，但返回的结果是向量，因为该包会将矩阵自动拉直。

向量中的数值型数据和字符串数据按照真实字符数(包括空格)计算，缺失值变量返回缺失值，不会计算。缺失值可以使用str_replace_na()替换为其他指定值。

如果只是计算部分指定元素的出现次数，可以使用函数str_count(字符串向量，指定字符)，此时仅返回指定字符在字符串各个位置出现的次数。

举例：

```
y <- c("Hello","WorldWorld","aroundWorld")
stringr::str_count(y,"World")
```

输出结果为c(0,2,1)。

计数需要满足两个条件：完全出现以及不重复。

思考：考虑输出结果：

```
y1 <- c("aba","ababa","baa")
stringr::str_count(y1,"aba")
```

字符串连接

R自带函数paste()与paste0()进行字符串的连接，使用格式如下：

paste(需要连接的元素, sep = " ", collapse = NULL)

paste0(需要连接的元素, collapse = NULL)

其中sep代表间隔使用的符号，collapse = NULL代表消除字符串之间的空间。

举例：

paste("abc","def",sep = ",")

输出值为"abc,def"

stringr包中则使用str_c()函数，使用格式为：

str_c(需要连接的元素, sep = "", collapse = NULL)，参数设定与paste一致。参数seq = ""，用来调节连接的字符，默认是直接连接，即输入若干个单个的字符或者数字，同时输入一个字符串向量，则返回一个字符串向量，每个元素是单个连接的字符串(批量连接)输入一个字符串向量。

指定参数collapse = ""，则以该间隔字符连接字符串向量，默认值为NULL。如果collapse = NULL(默认值)长度等于最长输入字符串的字符向量。如果collapse非NULL，则间隔为长度为1的字符向量。

str_c()还有一种功能，即结合条件语句使用。结构为：

str_c(字符1,if(条件1)字符2,...)

举例：str_c("abc",if(2<=1)"def","ghi",sep = ",",collapse = NULL)

输出结果为："abc,ghi"

字符串截取

R中自带函数substr()和substring()函数可以截取字符串指定部分，格式为：

(1)substr(字符串, start, stop):

(2)substring(字符串, first, last)：

举例：

```
x <- "123456789"
substr(x, c(2,4), c(4,5,8))
substring(x, c(2,4), c(4,5,8))
```

输出结果分别为"234"与c("234","45","2345678")。因为x的向量长度为1，所以substr获得的结果只有1个字串，即第2和第3个参数向量只用了第一个组合c(2,4)：起始位置2，终止位置4。

而substring的语句三个参数中最长的向量为c(4,5,8)，执行时按短向量循环使用的规则第一个参数事实上就是c(x,x,x)，第二个参数就成了c(2,4,2)，最终截取的字串起始位置组合为：2-4, 4-5和2-8。

在stringr中函数str_sub(string, start = 1L, end = -1L)具有与substring()类似的效果。

举例:

x3 <- "123456789"

str_sub(x3,start = c(1,2,3), end = c(4,5,6))

输出结果为"1234" "2345" "3456"。如果类似substring()输入str_sub(x3,start = c(2,3), end = c(4,5,6))，输出结果与substring()类似，但是会有一个warning。

字符串拆分

R中自带拆分用函数strsplit()，使用格式为：

strsplit(x, split, fixed = TRUE, perl = FALSE, useBytes = FALSE)

如果fixed设置为FALSE则需要在split处填写正则表达式进行匹配，这里不做要求。设置fixed=TRUE，表示使用普通文本匹配或正则表达式的精确匹配。普通文本的运算速度快。

perl=TRUE/FALSE的设置和perl语言版本有关，如果正则表达式很长，正确设置表达式并且使用perl=TRUE可以提高运算速度。一般按照默认设置即可。

useBytes设置是否逐个字节进行匹配，默认为FALSE，即按字符而不是字节进行匹配。

举例:

strsplit("HELLO THE WORLD !","")

运行结果为："H" "E" "L" "L" "O" " " "T" "H" "E" " " "W" "O" "R" "L" "D" " " "!"

strsplit("HELLO THE WOR\r LD !","")

运行结果为："H" "E" "L" "L" "O" " " "T" "H" "E" " " "W" "O" "R" "\ r" "L" "D" " " "!"

stringr 包的str_split()的函数也可以处理字符拆分的问题. 格式为:

str_split(字符串向量，间隔字符)

函数会按照间隔字符识别拆分的位置，如果间隔字符为空(即没有间隔)，则将字符串拆成单个字符。

sentence是自带的字符串数据，为向量格式，包含720个英文句子(字符串)。可以尝试拆分，程序如下：

str_split(sentence,"")

字符匹配

有时需要批量式查找字符，例如一键修改等功能。R中携带有grep()和grepl()两个函数可用于字符匹配。格式为：

grep(pattern, x, ignore.case = FALSE, perl = FALSE, value = FALSE, fixed = FALSE, useBytes = FALSE, invert = FALSE)

grepl(pattern, x, ignore.case = FALSE, perl = FALSE, fixed = FALSE, useBytes = FALSE)

其中grep输出的是出现关键字符的位置，返回的是数值；grepl输出的是文件是否满足关键字，返回逻辑型数值。

stringr中存在一个功能类似但输出更加“形象”的函数str_view()，格式为：

str_view(字符串向量，匹配字符)

输出为右下角viewer中的一张图片，用方框标识匹配字符，可以用浏览器打开。

str_view()只能匹配一次，反复出现的字符串可以用str_view_all()查找。

如果只需要显示哪些字符串被匹配，则可以用str_detect()，用法相同，返回逻辑值。

字符替换

在批量查找后，最常用功能为批量替换。R中函数为sub()和gsub()，但其不能改变原字符串。格式为：

sub(pattern = "", replacement = "",text)

纯粹的替换可以使用stringr中的函数str_replace()，格式为：

str_replace(text, pattern = "", replacement = "")

str_replace_all()同理。

字符输出

R中的输出字符命令主要是print()和cat()。它们除了可以输出字符外，合法的对象都可以展示(文件，表格，函数源代码等等)。

需要重复输出时，需要配合额外的制表符，换行符，换页符等等，保证输出的规范性。

执行字符串

所有的函数，命令等等，其实都是字符串，但是被""框住的字符在存储的时候是按照字符型数据存储的。

要把拼接好的字符编程实际的命令执行，这个过程叫做执行字符串，使用的核心函数是：eval(parse(text= 需执行的字符串))

总结

至此我们学习了R语言有关字符串的基本操作，可以对相对不复杂的文本信息做简单预处理。事实上，R语言只是兼容了文本信息处理的功能，并不是处理文本类信息的最好选择。

实践与工作中，常用perl进行专业的文本信息处理。近年来，python逐渐流行，也被广泛运用到文本信息处理中。