R for Data Science学习总结

Alex / 2020-08-21 / free_learner@163.com / AlexBrain.cn

更新于2023-09-02，主要是文字排版上的更新，内容基本保持不变。

总结*R for Data Science*一书中学到的一些函数用法。

一、ggplot2包的一些函数和功能

```
ggplot(data=<DATA>) +
   <GEOM_FUNCTION>(
      mapping = aes(<MAPPINGS>),
      stat = <STAT>,
      position = <POSITION>
   ) +
   <COORDINATE_FUNCTION> +
   <FACET_FUNCTION>
```

这是使用ggplot2画图时的一个基本结构模板，其中`DATA`表示数据，比如一个数据框；`GEOM_FUNCTION`表示用来表征数据的几何对象，比如`geom_point`表示用点来表征数据；`MAPPINGS`表示变量和视觉元素的映射关系，比如变量可以和坐标轴或者颜色形成映射；`STAT`表示根据计算出的统计量（而不是原始数据）来画图；`POSITION`用来调整数据之间的位置，比如用jitter来避免数据重叠太多；`COORDINATE_FUNCTION`用来设置坐标轴，比如`cood_flip`可以交换X和Y轴；`FACET_FUNCTION`可以将多个变量用子图的方式来呈现。

`labs()`用于给图添加或修改坐标轴/图例的文本，比如`labs(x="Engine displacement", y="Highway fuel economy", colour="Car type")`，表示修改X/Y轴的文本以及图例的文本。

`geom_text()`用于给特定的数据添加文本。

`geom_label()`类似于`geom_text()`，会给文本加一个矩形框。

`geom_hline()`/`geom_vline()`添加水平/垂直参考线。

`geom_rect()`画矩形框。

`geom_segment()`添加箭头。

`scale_x_continuous()`/`scale_y_continuous()`/`scale_color_discrete()`添加默认的尺度（scaling），比如`scale_y_continuous(breaks=seq(15,40,by=5), labels=NULL)`表示修改Y轴刻度以及去掉刻度文本。颜色的尺度表示颜色与数值的对应关系。

`scale_x_log10()` / `scale_y_log10()` 设置对数坐标。

`scale_color_brewer()` 使用ColorBrewer尺度。

`scale_color_manual()` 手动设置颜色尺度。

`scale_color_gradient()` / `scale_color_gradient2()` / `scale_fill_gradient()` 用于连续变量的颜色尺度。

`guides()` 的 `guide_legend()` 可以对图例做更多调整。

`theme()` 的legend.position参数可以控制图例出现的位置。

`coord_cartesian()` 的xlim/ylim参数设置坐标轴范围。

`theme_bw()` / `theme_classic()` / `theme_gray()` 等表示ggplot2自带有8个主题。

二、dplyr包的一些函数和功能

`fileter()` 用于筛选观测或样本，比如 `filter(flights, month==1, day==1)` ，表示筛选flights数据框中变量month和day都等于1的观测。

`arrange()` 用于调整观测顺序，比如 `arrange(flights, year, month, day)` ，表示分别根据year/month/day对观测进行排序。

`select()` 用于根据变量名提取变量，比如 `select(flights, year, month, day)` ，表示提取变量year、month和day。可以于 `starts_with()` / `ends_with()` / `contains()` / `matches()` / `num_range()` 等函数结合，根据一定的模式提取变量，比如 `select(flights, starts_with("abc"))` 提取以abc开头的变量。

`rename()` 用于更新变量名，比如 `rename(flights, tail_num=tailnum)` ， `tail_num` 是新的变量名。

`mutate()` 用于根据已有变量生成新的变量，比如 `mutate(flights, gain=arr_delay - dep_delay)` ，新生成的变量位于最后；如果是只想保留新生成的变量，可以使用 `transmute()` 。

`summarize()` 用于根据多个数值计算处一个数据，比如 `summary(flights, delay=mean(dep_delay, na.rm=TRUE))` 。

`group_by()` 用于将数据分组，与上述函数结合，可以实现对不同组进行相同操作，比如 `by_day <- group_by(flights, year, month, day); summarize(by_day, delay=mean(dep_delay, na.rm=TRUE))` ； `ungroup()` 可以去除数据的分组。

`inner_join()` 用于合并两个数据集并保留共有的观测； `left_join()` 保留左侧数据集的所有观测； `right_join()` 保留右侧数据集的所有观测； `full_join()` 保留所有的观测。这四个函数实现的功能同base包的 `merge()` 。由于这种合并会增加新的变量，书中称这种操作为mutating joins。

`semi_join()` 保留一个数据集中与另一个数据集匹配的观测，类似于 `%in%` ，但可用于多变量的情况； `anti_join()` 就是去掉匹配的观测。这种操作类似于filtering，书中称为filtering joins。

`intersect()` 保留两个数据集中共有的观测； `union()` 保留两个数据集独特（unique）的观测； `setdiff()` 保留在左侧不在右侧的观测。这种合并书中称为set operations，因为观测被当作集合的元素。

三、tidyr包的一些函数和功能

`gather()` 用于当列不是变量（variable）而是变量的值（value）的时候，比如 `table4a %>% gather(`1999`,`2000`,key="year",value="cases")` 。

`spread()` 用于当一个观测（observation）出现在多行的时候， `spread(table2, key=type, value=count)` 。

`separate()` 用于将一列分成多列，比如 `table3 %>% separate(rate, into=c("cases","population"),sep="/")` 。

`unite()` 用于将多列合成一列，比如 `table5 %>% unite(new, century, year, sep="")` 。

`complete()` 根据几列形成所有可能的组合，可用于将缺失值显示出来，比如 `stocks %>% complete(year, qtr)` 。

`nest()` 生成列表列（list-column），即一列的每个元素是一个列表。

四、stringr包的一些函数和功能

`str_length()` 计算字符串字符数量。

`str_c()` 连接多个字符串，比如 `str_c("x", "y", sep= ",")` 。使用collapse参数，将字符串向量转换成一个字符串，比如 `str(c("x", "y", "z"), collapse= ",")` 。

`str_replace_na()` 将NA替换为字符串 `"NA"` 。

`str_sub()` 提取部分字符串，比如 `str_sub("apple", 1, 3)` 返回 `"app"` ，如果是负数，则从后往前计数，比如 `str_sub("apple", -3, -1)` 返回 `"ple"` 。

`str_to_lower()` / `str_to_upper()` / `str_to_title()` / `str_to_sentence()` 进行大小写转换，比如 `str_to_upper("Dog", locale= "en")`，locale参数设置语言区域。

`str_sort()` / `str_order()` 对字符串向量进行排序，比如 `str_sort(c("apple", "banana"), locale="en")`。

`str_view()` / `str_view_all()` 显示字符串和正则表达式的匹配情况。

常用的正则表达式匹配规则：匹配 `.` 需要 `"\\."`，`\` 表示转义（escape），之所以需要两个\，是因为要表示成字符串；匹配 `\` 需要 `"\\\\"`；`^` 和 `$` 分别匹配字符串的开始和结束；`\d` 和 `\s` 分别匹配任意的数字和空白（whitespace），表示成字符串为 `"\\d"` 和 `"\\s"`；`[abc]` 和 `[^abc]` 分别匹配a/b/c和除了a/b/c以外的字符；`abc|xyz` 匹配abc或xyz；`?` / `+` / `*` 分别匹配0或1个、1或多个、0或多个字符；`{n}` / `{n,}` / `{,m}` / `{n,m}` 匹配数目分别为n、n或大于n、m或小于m、n到m之间；`(abc)\\1` 表示匹配abc重复出现1次，其中 `\1`，`\2` 称为backreferences。

`str_detect()` 判断一个字符串向量是否匹配一个字符模式（pattern），返回TRUE/FALSE。

`str_subset()` 提取与模式匹配的字符串。

`str_count()` 计算一个字符串中有多少次与模式匹配，比如 `str_count("banana", "a")` 返回3。

`str_extract()` / `str_extract_all()` 提取与模式匹配的字符串，不同于 `str_subset`，这两个函数只提取匹配的那部分字符。

`str_match()` / `str_match_all()` 可以给出单个匹配，比如 `str_match("abcxyz", "(abc)(xyz)")` 返回"abcxyz", "abc", "xyz"，其中括号用于形成一个组（group）。

`str_replace()` / `str_replace_all()` 将匹配的字符替换成新的字符。

`str_split()` 根据匹配模式将字符串进行分割，并返回一个列表。使用simplify参数返回一个矩阵。

`str_locate()` / `str_locate_all()` 返回匹配的起始位置。

五、forcats包的一些函数和功能

`fct_reorder()` 根据某个向量重新调整因子水平的次序。

`fct_reorder2()` 根据两个向量重新调整因子水平的次序。

`fct_relevel()` 将某些因子水平次序调整到最前面。

`fct_infreq()` 根据频率的升序调整因子水平次序。

`fct_rev()` 颠倒因子次序。

`fct_recode()` 重新编码因子水平。

`fct_collapse()` 将多个因子水平合并为一个。

`fct_lump()` 将样本较少的因子水平合并成一个。

六、lubridate包的一些函数和功能

`today()` / `now()` 返回当前的日期(date)和日期-时间(date-time)。

`ymd()` / `ymd_hms()` 等根据字符串解析成日期和日期-时间类型。

`make_date()` / `make_datetime()` 根据多个变量合成一个日期或日期-时间类型变量。

`as_date()` / `as_datetime()` 进行日期和日期-时间变量类型的转换。

`year()` / `month()` 等提取一个日期变量的年、月等成分，或者对相应的成分进行赋值。

`floor_date()` / `round_date()` / `ceiling_date()` 根据指定的单位对日期-时间进行取整等操作。

`update()` 可以同时改变一个日期-时间变量的多个成分，比如 `update(datetime, year=2020, month=2)` 。

`as.duration()` / `dseconds()` / `dminutes()` / `dhours()` 将时间差转换为秒为单位，比如 `dminutes(1)` 返回60。

`seconds()` / `minutes()` / `hours()` 表示一个固定的时间段，可用于平年或夏令时的情况。

`Sys.timezone()` / `OlsonNames()` 返回当前时区和所有时区的名字。

`with_tz()` / `force_tz()` 切换时区或修改时区。

七、purrr包的一些函数和功能

`map()` / `map_lgl()` / `map_int()` 等对一个向量（列表）的每个元素进行相同的函数操作，比如： `map_dbl(df, mean)` 计算每列的均值。

`safely` / `possibly()` / `quietly()` 用于调试程序。

`map2()` / `pmap()` 用于同时循环多个变量。

`invoke_map()` 循环多个函数。

`keep()` / `discard()` 根据一定条件保留和去除元素，比如 `keep(is.factor)`。

`reduce()` / `accumulate()` 对一个复杂的列表的两个元素反复应用一个函数。

八、modelr包的一些函数和功能

`seq_range()` 根据向量的范围生成一系列的数值。

`data_grid()` 生成一系列均匀分布网格点（grid of values）。

`add_predictions()` 根据数据和模型生成预测值。

`add_residuals()` 根据数据和模型生成残差。

`model_matrix()` 根据公式（formula，比如y~x）生成实际的数据矩阵。

`gather_predictions()` / `spread_predictions()` 同时生成多个模型的预测。