第5章

数据探索

5.1 数据描述

数据描述是通过分析数据的统计特征，加深对数据的理解，进而使用合适的统计分析或数据挖掘方法去探究数据潜在的信息，实现数据洞察。数据描述主要分析数据的类别、离散程度、数据的偏度和峰度等特征。

5.1.1 数据的类别

数据是数据对象及其属性的集合。一个数据对象是对一个事物或者物理对象的描述，可以是一条记录、一个实体、一个案例或一个样本，而数据对象的属性则是这个对象的性质、特征或维度。在数据集中，通常把数据对象称为样本，数据对象的属性称为变量。

在大数据时代，数据的来源多样化发展，数据的格式和形态也日渐丰富，如数字、文字、图片和视频、音频等。其中，能够使用统一的结构加以表示的数据称为结构化数据，如数字、字符等；其他无法使用统一的结构表示的数据，如音频、图像、视频等，称为非结构化数据。对于结构化数据，可按照对客观事物测度的程度或精确水平来划分，分为如下类别，详见表 5-1。

表 5-1 常见的数据类别

数据类别	数据特征	举例
分类数据	没有数量和顺序关系	状态，如“男”“女”“0”“1”
有序数据	有顺序关系	特征量，如“甲”“乙”“丙”“丁”，甲＞乙＞丙＞丁
区间数据	有数量关系，可比较大小，可排序，可计算差异	实数，如体重、身高
比例数据	有数量关系，可比较大小，可排序，可计算差异，具有绝对零点	实数，事物之间的比值

同类事物使用不同的尺度量化，会得到不同类别的数据。例如，学生成绩数据按实际数字填写就是区间数据；按 A、B、C 等分段进行区分就是有序数据；按是否及格区分则是分类数据；某同学的成绩是另一同学的两倍，便是比例数据。

5.1.2 数据的集中趋势

一般情况下，对一组数据的中心位置进行数量化的描述，能够代表这组数据的集中趋势，即反映大多数数据向某一点集中的情况。通常用来描述数据集中趋势的统计量主要包括平均数(Mean)、中位数(Median)、众数(Mode)等。平均数即样本的数据相加之和再除以样本个数；中位数是一组数据按顺序依次排列后处在中间位置的数；众数则是一组数据中出现次数最多的数。如果数据服从正态分布，则平均值就是数据的集中位置，它在一定程度上度量数据的平均水平。然而，数据的平均值易受数据分布的影响，有时使用中位数来衡量数据的集中位置会比使用平均值更有效。众数是最频繁出现的值，在数据中占比例最高。因此，判断一组数据的集中程度需要综合衡量上述几个统计量。

R 语言中使用 mean()函数和 median()函数来计算一组数据的平均数和中位数。R 中没有直接求众数的内置函数，但可以使用 table()函数来计算出现次数(频数)，再通过 max()、sort()、which.max()等函数查看其中频数最大的数值。

以下使用 apply()函数，对 iris 数据集中的样本进行平均值和中位数的计算。从结果可知，4 个属性的平均值和中位数差距并不大，表明 4 个属性的数据接近正态分布。然后在 iris 数据集中的一个变量 Sepal.Length 中，通过对其频数的排序找到众数，即出现了 10 次的取值为 5 的数据，是出现频率最大的数。取值为 5.1、6.3 的数各出现了 9 次，也是出现比较多的数。

```
> apply(iris[, c(1:4)], 2, mean)                #求平均值
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
    5.843333     3.057333     3.758000     1.199333
> apply(iris[, c(1:4)], 2, median)              #求中位数
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
        5.80         3.00         4.35         1.30
> sort(table(iris[, c(1)]),decreasing = T)    #查看频数
  5 5.1 6.3 5.7 6.7 5.5 5.8 6.4 4.9 5.4 5.6   6 6.1 4.8 6.5 4.6 5.2 6.2 6.9 7.7 4.4
 10   9   9   8   8   7   7   7   6   6   6   6   6   5   5   4   4   4   4   4   3
5.9 6.8 7.2 4.7 6.6 4.3 4.5 5.3   7 7.1 7.3 7.4 7.6 7.9
  3   3   3   2   2   1   1   1   1   1   1   1   1   1
```

5.1.3 数据的离散程度

描述数据离散程度的统计量主要有方差、标准差、中位数绝对偏差、变异系数、四分位数、极差等。方差(Variance)用来计算每一个样本数据与平均数之间的差异；标准差(Standard Deviation)也称为标准偏差，是方差的算术平方根，平均数相同的两组数据，其标准差未必相同；中位数绝对偏差(Median Absolute Deviation，MAD)是度量数据相对于中位数的离散情况；变异系数(Coefficient of Variation，CV)是数据标准差与数据平均数的比值，取值越大说明数据越分散，不受测量尺度和量纲的影响，比较客观；四分位数(Quartile)也称为四分位点，包括下四分位数、中位数和上四分位数，所有数值由小到大

排列并分成四等份，处于第一个分割点位置的数值是下四分位数，处于第二个分割点位置（中间位置）的数值是中位数，处于第三个分割点位置的数值是上四分位数；极差（Range）是指数据最大值和最小值之间的距离，极差越小说明数据越集中。

以下使用 apply()函数，对 iris 数据集调用 var()、sd()、mad()、quantile()和 range()函数计算每个变量的方差、标准差、中位数绝对偏差、四分位数和极差，变异系数使用公式计算。

```
> apply(iris[, c(1:4)], 2, var)                #方差
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
   0.6856935    0.1899794    3.1162779    0.5810063
> apply(iris[, c(1:4)], 2, sd)                 #标准差
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
   0.8280661    0.4358663    1.7652982    0.7622377
> apply(iris[, c(1:4)], 2, mad)                #中位数绝对偏差
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
    1.03782      0.44478      1.85325      1.03782
> #利用"标准差/平均值"计算出变异系数
> apply(iris[, c(1:4)], 2, sd)/apply(iris[, c(1:4)], 2, mean)
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
   0.1417113    0.1425642    0.4697441    0.6355511
> apply(iris[, c(1:4)], 2, quantile)           #四分位数
     Sepal.Length Sepal.Width Petal.Length Petal.Width
0%            4.3         2.0          1.00         0.1
25%           5.1         2.8          1.60         0.3
50%           5.8         3.0          4.35         1.3
75%           6.4         3.3          5.10         1.8
100%          7.9         4.4          6.90         2.5
> apply(iris[, c(1:4)], 2, range)             #极差
     Sepal.Length Sepal.Width Petal.Length Petal.Width
[1,]          4.3         2.0          1.0         0.1
[2,]          7.9         4.4          6.9         2.5
```

5.1.4　数据的分布特征

正态分布（Normal distribution）也称“常态分布”，又名高斯分布（Gaussian distribution），是一个在数学、物理及工程等领域都非常重要的概率分布。正态曲线呈钟型，两端低、中间高、左右对称，因其曲线呈钟形，因此又称为钟形曲线。偏度和峰度是描述数据分布特征的统计量：偏度（Skewness）是用于衡量数据分布的不对称程度或偏斜程度的指标；峰度（Kurtosis）又称峰态系数，直观反映了峰部的尖度。正态分布是一种无偏分布，其偏度等于 0。当偏度不为 0 时，表明数据分布是非对称的：偏度大于 0 时，数据分布是右偏或正偏；反之，偏度小于 0 表明数据分布是左偏或负偏。当数据为正态分布时，峰度近似等于 3。与正态分布相比较，当峰度大于 3 时，峰度越大，分布曲线越陡峭，表明

数据分布越集中；当峰度小于 3 时，峰度越小，表示分布曲线越平坦，数据分布越分散。

下例基于 iris 数据集计算 4 个变量的偏度和峰度。从输出结果可以发现，SepalLength 和 SepalWidth 两个变量的数据集为右偏，PetalLength 和 PetalWidth 为左偏。

在分析实际问题时，需要将计算出的偏度、峰度和图形结合起来进行判断。在计算出偏度和峰度后，利用 gather()函数对数据集进行变换，将每个变量转化为行，即宽型数据转换为长型数据，然后使用 ggplot2 绘制 4 个变量的密度曲线，如图 5-1 所示。

```
> install.packages("moments")
> library(moments)
> apply(iris[, 1:4], 2, skewness)              #计算偏度
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
   0.3117531    0.3157671   -0.2721277   -0.1019342
> apply(iris[, c(1:4)], 2, kurtosis)           #计算峰度
Sepal.Length  Sepal.Width Petal.Length  Petal.Width
    2.426432     3.180976     1.604464     1.663933
> install.packages("tidyr")
> install.packages("ggplot2")
> library(ggplot2)
> library(tidyr)
> irislong <- gather(iris[, 1:4], key = "varname", value = "value")
>##  可视化数据分布
> ggplot(irislong, aes(colours = varname, fill = varname, linetype = varname,
alpha = 0.5))
    +  theme_bw()
    + geom_density(aes(value), bw = 0.5, alpha = 0.4)
```

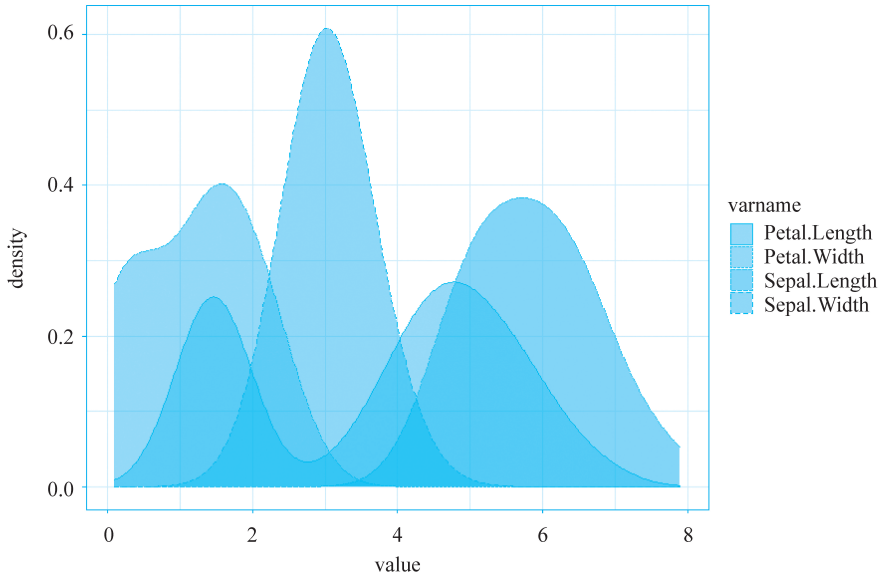

图 5-1　使用 ggplot()函数绘制 iris 数据集 4 个变量的密度曲线

5.1.5　数据的相似性

相似性度量(Similarity Measurement)是综合评定两个事物之间相似程度的度量,在聚类和分类中具有重要的地位。常用的相似性度量有相关系数(变量之间的接近程度)和相似系数(样本之间的接近程度)。

相关系数是度量数据变量之间线性相关性的指标。在二元变量的相关分析中,常用的有 Pearson 相关系数、Spearman 秩相关系数和判定系数等。Pearson 相关系数一般用于分析两个正态连续性变量之间的关系,取值范围是[−1, 1],如果小于 0,说明变量间负相关,越接近于−1 负相关性越强;大于 0 说明变量间正相关,越接近于 1 正相关性越强。Spearman 秩相关系数一般用于分析不服从正态分布的变量、分类变量或等级变量之间的关联性。而对于连续测量数据,更适合用 Pearson 相关系数进行分析。判定系数(Coefficient of Determination)也称为决定系数,是衡量自变量与因变量是否相关的重要指标,它的值越接近于 1,表明自变量与因变量之间的相关性越强。

在 R 语言中,使用 cor()函数计算相关系数,系统默认参数 method="pearson",也可设置为 method="spearman"等其他方法进行计算。下例对数据集 iris 使用 cor()函数计算 4 个数值变量之间的相关系数。

```
> cor(iris[, c(1:4)])
              Sepal.Length  Sepal.Width  Petal.Length  Petal.Width
Sepal.Length     1.0000000   -0.1175698     0.8717538    0.8179411
Sepal.Width     -0.1175698    1.0000000    -0.4284401   -0.3661259
Petal.Length     0.8717538   -0.4284401     1.0000000    0.9628654
Petal.Width      0.8179411   -0.3661259     0.9628654    1.0000000
```

对于不同样本之间的相似性度量,通常计算样本间的“距离”,主要有欧氏距离、曼哈顿距离、最大距离等方法。欧氏距离又称为欧几里得距离,是度量欧几里得空间中两点间的直线距离;曼哈顿距离用以表明两个点在欧几里得空间的固定直角坐标系上的绝对轴距的总和;最大距离为两个点之间的各个坐标分量差的最大值。

R 语言中使用 dist()函数计算距离,通过参数 method 来指定计算距离的方法,参数“euclidean”“manhattan”“maximum”分别代表欧氏距离、曼哈顿距离和最大距离三种距离方法。以下使用 iris 数据集计算每个样本间的欧氏距离,得到一个对角线为 0 的对称矩阵,可以看到距离为 0 表示其本身,距离越小表示样本越相似。

```
> dist(iris[, c(1:4)], method = "euclidean", upper = T, diag = T)
          1         2         3         4         5
1 0.0000000 0.5385165 0.5099020 0.6480741 0.1414214
2 0.5385165 0.0000000 0.3000000 0.3316625 0.6082763
3 0.5099020 0.3000000 0.0000000 0.2449490 0.5099020
4 0.6480741 0.3316625 0.2449490 0.0000000 0.6480741
5 0.1414214 0.6082763 0.5099020 0.6480741 0.0000000
6 0.6164414 1.0908712 1.0862780 1.1661904 0.6164414
(共有 150 个样本,产生 150 * 150 矩阵,后面数据略去)
```

5.2 数据清洗

在实际数据挖掘过程中，从外部获得的数据往往存在缺失值、重复值、异常值或者错误值，通常这类数据被称为“脏数据”，需要对其进行清洗。数据清洗是数据准备过程中最重要的一步，通过填补缺失数值、识别或删除离群点等方法解决不一致性、纠正错误数据，从而得到干净的数据。数据清洗的主要目的是提高数据质量，进而提高挖掘结果的可靠性和准确性，这是数据挖掘过程中非常必要的一个步骤。

5.2.1 处理缺失数据

数据存在缺失值非常普遍。数据缺失是指在数据采集、传输和处理过程中，由于某些原因导致数据不完整的情况。从数据缺失的分布来讲，缺失值可以分为完全随机缺失、随机缺失和完全非随机缺失。完全随机缺失是指数据的缺失是完全随机的，缺失情况相对于所有数据来说，在统计意义上是独立的，直接删除缺失数据对模型影响不大。完全非随机缺失指的是数据的缺失与缺失值本身存在某种关联，例如，在调查时所涉及的问题过于敏感，被调查者拒绝回答而造成的缺失。从统计角度来看，非随机缺失的数据会产生有偏估计，而这部分的缺失数据处理也是比较困难的。随机缺失处于两者之间。综上原因，需要对数据采集或数据来源中出现缺失值的原因进行了解后，再进行缺失值处理。

1. 缺失值的表示

在 R 语言中，缺失值用符号 NA 表示，代表数据集中该数据遗失或不存在。在对含 NA 的数据集进行函数操作时，该 NA 参与运算，因此需要进行预先处理来移除 NA 的影响。另外，R 语言中还有 NULL 和 NaN 等特殊类型的数据。NULL 表示未知的状态，它不会影响函数的计算；NaN 表示无意义的数，例如，除数为 0 的结果就是 NaN。三者的含义与处理方式不同。

2. 缺失值的判别

R 语言提供了一些函数用于判别缺失值，详见表 5-2。检查数据集中是否存在缺失值的最简单的方法是使用 summary()函数，该函数会输出数据中每个变量的基本信息，同时也会输出变量中含有缺失值的个数。确定缺失值的数量后，可以通过 is.na()函数查看缺失值的位置，若返回 TRUE 表示是缺失值。另外，还可以使用 vim 包中的 aggr()函数，通过可视化方法查看数据缺失值的图形描述。

表 5-2 缺失值处理的相关函数

函 数 名	含 义	返 回 值
summary()	显示数据的总体概况	
is.na()	检测缺失值是否存在	逻辑值：TRUE 或 FALSE

续表

函　数　名	含　　义	返　回　值
complete.cases()	检测行是否完整	逻辑值：TRUE 或 FALSE
na.omit()	移除所有含缺失数据的行	
aggr()	在 vim 包中，可视化描述缺失值	

3. 缺失值的处理

针对带有缺失值的数据集，如何使用合适的方法处理缺失值是数据预处理的关键。缺失值的主要处理方法有删除记录、数据插补和不处理 3 种。

（1）当缺失数据较少时，直接删除相应样本。

删除缺失数据的样本，其前提是缺失数据的比例较少，而且缺失数据是随机出现的，这样删除缺失数据后对分析结果的影响不大。使用 na.omit()函数移除所有含缺失数据的行，简单有效。

（2）对缺失数据进行插补。

有时直接删除缺失值会影响数据的客观性和分析结果的正确性，可采用插补法来完成缺失数据的处理，即在有缺失值的地方补上数据，不会减少样本信息。表 5-3 介绍了常用的插补方法。

表 5-3　常用插补方法

常用插补方法	描　　述
固定值插补	固定值
均值插补法	平均值/中位数/众数，近邻平均数
多重插补法	回归预测等模型方法

均值插补法是一种简便、快速的缺失数据处理方法。如果缺失数据是数值型的，则根据该变量的平均值来填充缺失值；如果缺失值是非数值型的，则根据该变量的众数填充缺失值。使用均值插补法处理简单，但缺点在于它建立在完全随机缺失的假设之上，当缺失数据不是随机出现时会产生偏误，当缺失比例较高时会错误估计该变量的方差。

多重插补法在面对复杂的缺失值问题时经常使用，它并不是用单一值来替换缺失值，而是通过不同数学模型反映的变量间关系来预测缺失数据，生成多组插补，形成多组完整数据集，再对这些数据集进行分析，得到最佳插补数据。这些操作使用 R 语言的 mice 包实现。

（3）使用对缺失数据不敏感的分析方法。

当缺失值数量不大，并且采用对缺失数据不敏感的数学模型进行分析时，缺失值可不必特别处理。

5.2.2　处理异常数据

异常值也称离群点，是指数据采集中出现的随机错误或偏差，包括错误值和偏离均值

的孤立点。在数据处理中，异常值会极大地影响回归或分类的效果。为了避免异常值造成损失，需要在数据预处理阶段进行异常值检测。在一些应用中，如质量检测，异常值检测也可能是数据处理的主要目标。

1. 异常值的判别

检测异常值的方法包括箱线图、散点图、聚类和回归分析等。

(1) 使用箱线图检测离群点。

箱线图又称盒式图或箱形图，是用来显示一组数据分布情况的统计图，应用广泛，在质量管理中尤为重要。箱线图的绘制方法是：首先，找出一组数据的上边缘、下边缘、中位数和两个四分位数；然后，连接两个四分位数画出箱体，再将上边缘和下边缘与箱体相连接，中位数在箱体中间。这样异常值就可以直观地显示出来，如图 5-2 所示。在 R 语言中，使用 boxplot()函数来绘制箱线图，使用 boxplot.stats()函数来检测异常数据。

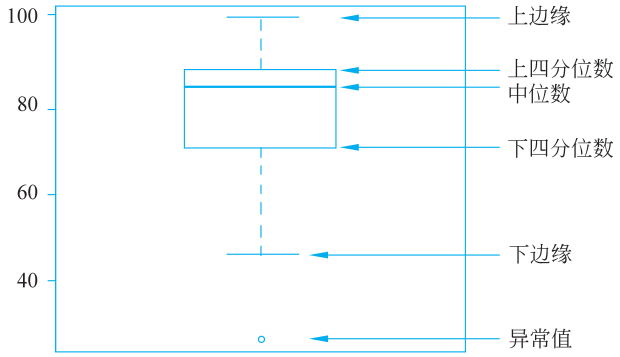

图 5-2　使用箱线图来查看异常值

(2) 使用散点图检测离群点。

散点图将数据值在图表中以点的位置表示，数据的类别可由图表中点的不同形状或颜色标记，通常用于跨类别的数据分布比较，或者衡量不同数据类别间的相似性和差异性。在散点图中可以通过离群点来检测异常值。

(3) 使用聚类方法检测异常值。

“物以类聚，人以群分”，聚类是很重要的一个概念。在自然科学和社会科学中，存在着大量的分类问题，聚类不等于分类，聚类所划分的类是未知的，聚类将不属于任何一类的数据作为异常值。在 R 语言中，通过聚类函数来找到聚类的个数和相应的聚类中心点，然后通过计算每个样本数据到聚类中心的最大距离来找到异常值(见第 7 章)。

(4) 回归分析检测异常值。

回归分析是一种数学模型，用来确定两种或两种以上变量间相互依赖的定量关系。在大数据分析中，回归分析是一种预测性的建模技术，用于预测分析、发现变量之间的因果关系等。在 R 语言中，可使用回归函数结合散点图来检测异常值(见第 6 章)。

2. 异常值的处理

对检测出来的异常值，可以按照表 5-4 的方法进行处理。

表 5-4　常用异常值处理方法

异常值处理方法	描　　述
删除含有异常值的记录	直接将含有异常值的样本删除
视为缺失值	将异常值视为缺失值，利用缺失值处理的方法进行处理
平均值修正	可用前后两个观测值的平均值修正该异常值
不处理	直接使用有异常值的数据集

5.2.3　处理重复数据

R 语言中的数据重复检测函数主要有 unique()和 duplicated()：unique()用于为向量数据去掉重复值；duplicated()用于向量或数据框，返回一个 TRUE 和 FALSE 的向量，标注该索引所对应的值是否是重复值。

5.3　数 据 集 成

从多种途径、多种方式得到的数据格式多种多样，需要对这些数据整理才能进行有效分析。数据集成包括分组汇总、透视表生成等工作，如果数据分散在多个地方，则需要进行数据集的合并，包括横向合并和纵向合并。数据集成可以改善数据的外观，是绘制图形、统计分析、数据挖掘前必要的预处理步骤。

5.3.1　数据集的合并

多个数据集按照应用需求进行横向和纵向的合并。横向合并指的是两个数据集(数据框)合并为一个具有更多变量的数据集，主要使用 merge()或 cbind()函数。合并之前，根据需要找到两个数据框的公共索引，也称为联结变量，联结变量通常是一个或多个共有变量。例如，当表 1 和表 2 都有 ID 这个列变量且含义相同时，可以根据 ID 进行合并，这样新的数据框就把相同 ID 的多列数据进行了横向合并。当没有或不需要公共索引时，可以使用 cbind()函数，合并前需要确认两个数据框对象是否拥有相同的行数以及相同顺序排序。

纵向合并指的是两个数据集(数据框)合并为更多行的数据集，可以使用 rbind()函数，合并前需要确认两个数据集具有相同的列变量，它们的顺序不一定相同。若两个数据集的列数量不同，在纵向合并时会做相应处理，例如删除多余的列，或追加列并将其值设为 NA。

```
>Totalframe1 <- merge(dataframe1, dataframe2, by="ID")     #横向合并
```

```
>Totalframe2 <- cbind(dataframe1, dataframe2)                    #横向合并
>Totalframe3 <- rbind(dataframe1, dataframe2)                    #纵向合并
```

5.3.2 数据子集的获取

在很多数据集成工作中，通过行和列（样本和变量）的增加、删除、修改等操作，可以使数据更清晰，更容易进行后续的统计分析。R 语言具有强大的索引特性，第 2 章介绍了各数据对象的索引方法，例如，使用[row, col]的方式选择行和列；通过“!”操作符、NULL 赋值等方式来剔除某行和列；通过逻辑比较“==”、“>”、以及“TRUE”“FALSE”等组成逻辑表达式来选取数据。这些索引方法可快速访问对象中的元素，对变量或样本进行选入和排除。

除此之外，还可以使用函数进行数据子集的获取。subset()函数是一个简单、灵活的通用函数，它通过逻辑表达式确定选取的样本，通过 select 参数选择变量。以 mtcars 数据集为例，示例如下。

```
> subset(mtcars, cyl == 4 & gear == 3)
               mpg cyl  disp hp drat    wt  qsec vs am gear carb
Toyota Corona 21.5   4 120.1 97  3.7 2.465 20.01  1  0    3    1
> subset(mtcars, cyl == 4 & gear == 3, select = c(1:8))
               mpg cyl  disp hp drat    wt  qsec vs
Toyota Corona 21.5   4 120.1 97  3.7 2.465 20.01  1
```

transform()和 within()函数主要对列变量进行操作。transform()函数可以在原数据框基础上增加或修改列变量生成一个新的数据框，或者通过 NULL 赋值的方式删除列变量，还可以将多个列变量替换为一些描述性统计值，便于进一步的处理。within()函数则不仅可以应用于数据框，还可以使用其他类型的数据，更为灵活。以 airquality 数据集为例，示例如下。

```
> head(airquality, 2)
  Ozone Solar.R Wind Temp Month Day
1    41     190  7.4   67     5   1
2    36     118  8.0   72     5   2
> newaq1 <- transform(airquality, logozone = log(Ozone))
> head(newaq1, 2)
  Ozone Solar.R Wind Temp Month Day logozone
1    41     190  7.4   67     5   1 3.713572
2    36     118  8.0   72     5   2 3.583519
> newaq2 <- transform(airquality, logozone = log(Ozone), Ozone = NULL,
                      WindWind = Wind * Wind, Wind = NULL)
> head(newaq2, 2)
  Solar.R Temp Month Day logozone WindWind
```