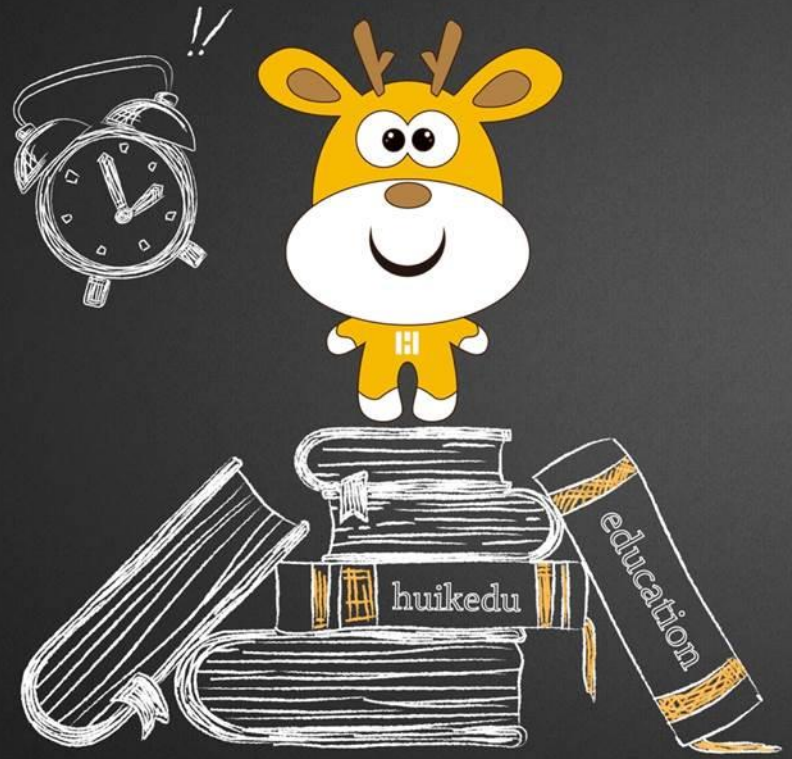
慧科集团
Huikedu Group



《统计分析与R语言》

高级数据管理

内容回顾

慧科集团 Huikedu Group

- 操作日期和缺失值
- 熟悉数据类型的转换
- 变量的创建和重编码
- 数据集的排序、合并和取子集
- 选入和丢弃变量

一个数据处理的难题

慧科集团 Huikedu Group

Table 5.1. Student exam data

Student	Math	Science	English
John Davis	502	95	25
Angela Williams	600	99	22
Bullwinkle Moose	412	80	18
David Jones	358	82	15
Janice Markhammer	495	75	20
Cheryl Cushing	512	85	28
Reuven Ytzrhak	410	80	15
Greg Knox	625	95	30
Joel England	573	89	27
Mary Rayburn	522	86	18

一个数据处理的难题

慧科集团 Huikedu Group

Table 5.1. Student exam data

Student	Math	Science	English
John Davis	502	95	25
Angela Williams	600	99	22
Bullwinkle Moose	412	80	18
David Jones	358	82	15
Janice Markhammer	495	75	20
Cheryl Cushing	512	85	28
Reuven Ytzrhak	410	80	15
Greg Knox	625	95	30
Joel England	573	89	27
Mary Rayburn	522	86	18

数值和字符处理函数

- 数学函数

函　数	描　述
abs(x)	绝对值 abs(-4)返回值为4
sqrt(x)	平方根sqrt(25)返回值为5 和25^(0.5)等价
ceiling(x)	不小于x的最小整数 ceiling(3.475)返回值为4
floor(x)	不大于x的最大整数 floor(3.475)返回值为3
trunc(x)	向0的方向截取的x中的整数部分 trunc(5.99)返回值为5

函　数	描　述
round(x, digits=n)	将x舍入为指定位的小数 round(3.475, digits=2)返回值为3.48
signif(x, digits=n)	将x舍入为指定的有效数字位数 signif(3.475, digits=2)返回值为3.5
cos(x)、sin(x)、tan(x)	余弦、正弦和正切 cos(2)返回值为-0.416
acos(x)、asin(x)、atan(x)	反余弦、反正弦和反正切 acos(-0.416)返回值为2
cosh(x)、sinh(x)、tanh(x)	双曲余弦、双曲正弦和双曲正切 sinh(2)返回值为3.627
acosh(x)、asinh(x)、atanh(x)	反双曲余弦、反双曲正弦和反双曲正切 asinh(3.627)返回值为2
log(x,base=n)	对x取以n为底的对数
log(x)	为了方便起见
log10(x)	log(x)为自然对数 log10(x)为常用对数 log(10)返回值为2.3026 log10(10)返回值为1
exp(x)	指数函数 exp(2.3026)返回值为10

数值和字

- 统计函数

函　数	描　述
mean(*x*)	平均数 mean(c(1,2,3,4))返回值为2.5
median(*x*)	中位数 median(c(1,2,3,4))返回值为2.5
sd(*x*)	标准差 sd(c(1,2,3,4))返回值为1.29
var(*x*)	方差 var(c(1,2,3,4))返回值为1.67
mad(*x*)	绝对中位差（median absolute deviation） mad(c(1,2,3,4))返回值为1.48

quantile(*x*,*probs*)	求分位数。其中x为待求分位数的数值型向量，*probs*为一个由[0,1]之间的概率值组成的数值向量
	# 求x的30%和84%分位点
	y <- quantile(x, c(.3,.84))

range(*x*)	求值域 x <- c(1,2,3,4) range(x)返回值为c(1,4) diff(range(x))返回值为3
sum(*x*)	求和 sum(c(1,2,3,4))返回值为10
diff(*x*, lag=*n*)	滞后差分，lag用以指定滞后几项。默认的lag值为1 x<- c(1, 5, 23, 29) diff(x)返回值为c(4, 18, 6)
min(*x*)	求最小值 min(c(1,2,3,4))返回值为1
max(*x*)	求最大值 max(c(1,2,3,4))返回值为4
scale(*x*,center=TRUE, scale=TRUE)	为数据对象x按列进行中心化（center=TRUE）或标准化（center=TRUE,scale=TRUE）； 代码清单5-6中给出了一个示例

数值和字符处理函数

慧科集团 Huikedu Group

- 统计函数

```
> x <- c(1,2,3,4,5,6,7,8)

> mean(x)                          <-- 简洁的方式
[1] 4.5
> sd(x)
[1] 2.449490

> n <- length(x)                   <-- 冗长的方式
> meanx <- sum(x)/n
> css <- sum((x - meanx)^2)
> sdx <- sqrt(css / (n-1))
> meanx
[1] 4.5
> sdx
[1] 2.449490
```

数值和字符处理函数

慧科集团 Huikedu Group

- 统计函数

例如：1、指定列进行均值为0、标准差为1的标准化：

newdata <- scale(mydata)

2、要对每一列进行任意均值和标准差的标准化

scale(mydata)*SD + M

scale(myvar)*10+50

数值和字符处理函数

慧科集团 Huikedu Group

- 概率函数

[dpqr]distribution_abbreviation()

d = 密度函数（density）

p = 分布函数（distribution function）

q = 分位数函数（quantile function）

r = 生成随机数（随机偏差）

数值和字符处理函数

慧科集团 Huikedu Group

- 概率函数

分布名称	缩　写	分布名称	缩　写
Beta分布	beta	Logistic分布	logis
二项分布	binom	多项分布	multinom
柯西分布	cauchy	负二项分布	nbinom
（非中心）卡方分布	chisq	正态分布	norm
指数分布	exp	泊松分布	pois
F分布	f	Wilcoxon符号秩分布	signrank
Gamma分布	gamma	t分布	t
几何分布	geom	均匀分布	unif
超几何分布	hyper	Weibull分布	weibull
对数正态分布	lnorm	Wilcoxon秩和分布	wilcox

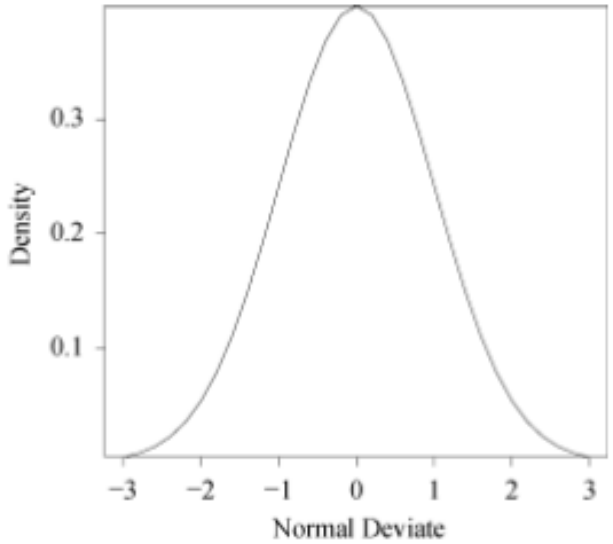
数值和字符处理函数

- 概率函数

问　题	解　法

在区间[−3, 3]上绘制标准正态曲线	`x <- pretty(c(-3,3), 30)`
	`y <-dnorm(x)`
	`plot(x, y,`
	`  type="l",`
	`  xlab="NormalDeviate",`
	`  ylab="Density",`
	`  yaxs="i"`
	`)`

位于 z=1.96 左侧的标准正态曲线下方面积是多少?	`pnorm(1.96)`等于0.975
均值为500，标准差为100的正态分布的0.9分位点值为多少?	`qnorm(.9, mean=500, sd=100)`等于628.16
生成50个均值为50，标准差为10的正态随机数	`rnorm(50, mean=50, sd=10)`



数值和字符处理函数

慧科集团 Huikedu Group

- 概率函数

  设定随机数种子

```
> runif(5)
[1] 0.8725344 0.3962501 0.6826534 0.3667821 0.9255909
> runif(5)
[1] 0.4273903 0.2641101 0.3550058 0.3233044 0.6584988
> set.seed(1234)
> runif(5)
[1] 0.1137034 0.6222994 0.6092747 0.6233794 0.8609154
> set.seed(1234)
> runif(5)
[1] 0.1137034 0.6222994 0.6092747 0.6233794 0.8609154
```

数值和字符处理函数

慧科集团 Huikedu Group

- 概率函数

  生成多元正态数据

  mvrnorm(n, mean, sigma)

均值向量	230.7	146.7	3.6
协方差阵	15360.8	6721.2	-47.1
	6721.2	4700.9	-16.5
	-47.1	-16.5	0.3

数值和字符处理函数

- 概率函数

  生成多元正态数据

  mvrnorm(n, me

```
> library(MASS)
> options(digits=3)
> set.seed(1234)                                   ← 1 Set random number seed

> mean <- c(230.7, 146.7, 3.6)                     ← 2 Specify mean vector, covariance matrix
> sigma <- matrix(c(15360.8, 6721.2, -47.1,
                    6721.2, 4700.9, -16.5,
                     -47.1,  -16.5,   0.3), nrow=3, ncol=3)

> mydata <- mvrnorm(500, mean, sigma)              ← 3 Generate data
> mydata <- as.data.frame(mydata)
> names(mydata) <- c("y","x1","x2")

> dim(mydata)                                      ← 4 View results
[1] 500 3
> head(mydata, n=10)
       y    x1   x2
1   98.8  41.3 4.35
2  244.5 205.2 3.57
3  375.7 186.7 3.69
4  -59.2  11.2 4.23
5  313.0 111.0 2.91
6  288.8 185.1 4.18
7  134.8 165.0 3.68
8  171.7  97.4 3.81
9  167.3 101.0 4.01
10 121.1  94.5 3.76
```

数值和字符处理函数

慧科集团 Huikedu Group

- 概率函数

  生成多元正态数据

```
install.packages("MASS")
library(MASS)
mean <- c(230.7, 146.7, 3.6)
sigma <- matrix( c(15360.8, 6721.2, -47.1,
                   6721.2, 4700.9, -16.5,
                   -47.1,  -16.5,   0.3), nrow=3, ncol=3)
set.seed(1234)
mydata <- mvrnorm(500, mean, sigma)
mydata <- as.data.frame(mydata)
names(mydata) <- c("y", "x1", "x2")
dim(mydata)
head(mydata, n=10)
```

数值和字符处理函数

慧科集团 Huikedu Group

- 字符处理函数

函　数	描　述
nchar(*x*)	计算x中的字符数量 x <- c("ab", "cde", "fghij") length(x)返回值为3（参见表5-7） nchar(x[3])返回值为5
substr(*x*, *start*, *stop*)	提取或替换一个字符向量中的子串 x <- "abcdef" substr(x, 2, 4)返回值为"bcd" substr(x, 2, 4) <- "22222"(*x*将变成"a222ef")
grep(*pattern*, *x*, ignore. case=FALSE, fixed=FALSE)	在*x*中搜索某种模式。若fixed=FALSE，则*pattern*为一个正则表达式。若fixed=TRUE，则*pattern*为一个文本字符串。返回值为匹配的下标 grep("A",c("b","A","c"),fixed=TRUE)返回值为2

数值和字符处理函数

慧科集团 Huikedu Group

- 字符处理函数

函　数	描　述
sub(*pattern*, *replacement*, *x*, ignore.case=FALSE, fixed=FALSE)	在*x*中搜索*pattern*，并以文本*replacement*将其替换。若fixed=FALSE，则*pattern*为一个正则表达式。若fixed=TRUE，则*pattern*为一个文本字符串 sub("\\s",".","Hello There")返回值为Hello.There。注意，"\s"是一个用来查找空白的正则表达式；使用"\\s"而不用"\"的原因是，后者是R中的转义字符（参见1.3.3节）
strsplit(*x*, *split*, fixed=FALSE)	在*split*处分割字符向量*x*中的元素。若fixed=FALSE，则*pattern*为一个正则表达式。若fixed=TRUE，则*pattern*为一个文本字符串 y <- strsplit("abc", "")将返回一个含有1个成分、3个元素的列表，包含的内容为"a" "b" "c" unlist(y)[2]和sapply(y, "[", 2)均会返回"b"
paste(…, sep="")	连接字符串，分隔符为sep paste("x", 1:3,sep="")返回值为c("x1", "x2", "x3") paste("x",1:3,sep="M")返回值为c("xM1","xM2" "xM3") paste("Today is", date())返回值为Today is Thu Jun 25 14:17:32 2011（我修改了日期以让它看起来更接近当前的时间）
toupper(*x*)	大写转换 toupper("abc")返回值为"ABC"
tolower(*x*)	小写转换 tolower("ABC")返回值为"abc"

数值和字符处理函数

慧科集团 Huikedu Group

- 其他实用函数

函　数	描　述
length(*x*)	对象x的长度 x <- c(2, 5, 6, 9) length(x)返回值为4

函　数	描　述
seq(*from*, *to*, *by*)	生成一个序列 indices <- seq(1,10,2) indices的值为c(1, 3, 5, 7, 9)
rep(*x*, *n*)	将*x*重复*n*次 y <- rep(1:3, 2) y的值为c(1, 2, 3, 1, 2, 3)
cut(*x*, *n*)	将连续型变量*x*分割为有着*n*个水平的因子 使用选项ordered_result = TRUE以创建一个有序型因子
pretty(*x*, *n*)	创建美观的分割点。通过选取*n*+1个等间距的取整值，将一个连续型变量*x*分割为*n*个区间。绘图中常用
cat(… , file ="myfile", append =FALSE)	连接…中的对象，并将其输出到屏幕上或文件中（如果声明了一个的话） firstname <- c("Jane") cat("Hello" ,firstname, "\n")

数值和字符处理函数

慧科集团 Huikedu Group

- 其他实用函数

```
name <- "Bob"
cat( "Hello", name, "\b.\n", "Isn\'t R", "\t", "GREAT?\n")
```

```
1  Hello Bob.
2   Isn't R		GREAT?
```

数值和字符处理函数

慧科集团 Huikedu Group

- 将函数应用于矩阵和数据框

```
> a <- 5

> sqrt(a)
[1] 2.236068

> b <- c(1.243, 5.654, 2.99)

> round(b)
[1] 1 6 3

> c <- matrix(runif(12), nrow=3)

> c
      [,1]  [,2]  [,3]  [,4]
[1,] 0.4205 0.355 0.699 0.323
[2,] 0.0270 0.601 0.181 0.926
[3,] 0.6682 0.319 0.599 0.215

> log(c)
       [,1]   [,2]   [,3]   [,4]
[1,] -0.866 -1.036 -0.358 -1.130
[2,] -3.614 -0.508 -1.711 -0.077
[3,] -0.403 -1.144 -0.513 -1.538

> mean(c)
[1] 0.444
```

数值和字符处理函数

- 将函数应用于矩阵和数据框

```
> mydata <- matrix(rnorm(30), nrow=6)
> mydata
         [,1]   [,2]    [,3]   [,4]   [,5]
[1,]  0.71298  1.368 -0.8320 -1.234 -0.790
[2,] -0.15096 -1.149 -1.0001 -0.725  0.506
[3,] -1.77770  0.519 -0.6675  0.721 -1.350
[4,] -0.00132 -0.308  0.9117 -1.391  1.558
[5,] -0.00543  0.378 -0.0906 -1.485 -0.350
[6,] -0.52178 -0.539 -1.7347  2.050  1.569
> apply(mydata, 1, mean)
[1] -0.155 -0.504 -0.511  0.154 -0.310  0.165
> apply(mydata, 2, mean)
[1] -0.2907  0.0449 -0.5688 -0.3442  0.1906
> apply(mydata, 2, mean, trim=0.2)
[1] -0.1699  0.0127 -0.6475 -0.6575  0.2312
```

1 生成数据

2 计算每行的均值

3 计算每列的均值

4 计算每列的截尾均值

数据处理难题的一套解决方案

慧科集团 Huikedu Group

- 示例解决方案

将学生的各科考试成绩组合为单一的成绩衡量指标、基于相对名次（前 20%，下20%，等等）给出从A到F的评分、根据学生姓氏和名字的首字母对花名册进行排序。代码 清单5-6给出了一种解决方案。

数值和字符处理函数

慧科集团 Huikedu Group

- 示例解决方案

scale函数是将一组数进行处理，默认情况下是将一组数的每个数都减去这组数的平均值后再除以这组数的均方根。有两个参数，center=TRUE，默认的，是将一组数中每个数减去平均值，若为false，则不减平均值

函数quantile()给出了学生综合得分的百分位数

控制流

慧科集团 Huikedu Group

- 语句（statement）是一条单独的R语句或一组复合语句（包含在花括号{ }中的一组R语句，使用分号分隔）；
- 条件（cond）是一条终被解析为真（TRUE）或假（FALSE）的表达式；
- 表达式（expr）是一条数值或字符串的求值语句；
- 序列（seq）是一个数值或字符串序列。

控制流

慧科集团 Huikedu Group

- 重复和循环

  1、for结构

  for (var in seq) statement

  for (i in 1:10) print("Hello")

控制流

慧科集团 Huikedu Group

- 重复和循环

  2、while结构

  while (cond) statement

  i <- 10

  while (i > 0) {print("Hello"); i <- i - 1}

控制流

慧科集团 Huikedu Group

- 条件执行

  1. if-else结构

  if (cond) statement

  if (cond) statement1 else statement2

```
x <- 2.0
if (x< 0.2){
  x <- x+1
  print("success!")
}else{
  x < x-1
  print("else success!")
}
```

控制流

慧科集团 Huikedu Group

- 条件执行

  2. ifelse结构

  ifelse(cond, statement1, statement2)

```
x <- c(1,1,1,0,0,1,1)
ifelse(x != 1, 1, 0)
```

控制流

慧科集团 Huikedu Group

- 条件执行

  3. switch结构

  switch(expr, ...)

```
result<-3
switch(result,"低异常点","偏低","正常","偏高","高异常点")
```

用户自编函数

慧科集团 Huikedu Group

- 函数的结构

```
myfunction <- function(arg1, arg2, ... ){
  statements
  return(object)
}
```

用户自编函数

慧科集团 Huikedu Group

- mystats()：一个由用户编写的描述性统计量计算函数

```
mystats <- function(x, parametric=TRUE, print=FALSE) {
  if (parametric) {
    center <- mean(x); spread <- sd(x)
  } else {
    center <- median(x); spread <- mad(x)
  }
  if (print & parametric) {
    cat("Mean=", center, "\n", "SD=", spread, "\n")
  } else if (print & !parametric) {
    cat("Median=", center, "\n", "MAD=", spread, "\n")
  }
  result <- list(center=center, spread=spread)
  return(result)
}
```

用户自编函数

慧科集团 Huikedu Group

- mystats()：一个由用户编写的描述性统计量计算函数

```
set.seed(1234)
x <- rnorm(500)
y <- mystats(x)
y <- mystats(x, parametric=FALSE, print=TRUE)
```

整合与重构

慧科集团 Huikedu Group

- 转置

  函数t()即可对 一个矩阵或数据框进行转置

```
> cars <- mtcars[1:5,1:4]
> cars
                   mpg cyl disp  hp
Mazda RX4         21.0   6  160 110
Mazda RX4 Wag     21.0   6  160 110
Datsun 710        22.8   4  108  93
Hornet 4 Drive    21.4   6  258 110
Hornet Sportabout 18.7   8  360 175
> t(cars)
     Mazda RX4 Mazda RX4 Wag Datsun 710 Hornet 4 Drive Hornet Sportabout
mpg         21             21       22.8           21.4              18.7
cyl          6              6        4.0            6.0               8.0
disp       160            160      108.0          258.0             360.0
hp         110            110       93.0          110.0             175.0
```

整合与重构

慧科集团 Huikedu Group

- 整合数据

aggregate(x, by, FUN)

x是待折叠的数据对象

by是一个变量名组成的列表

FUN则是用来计算描述性统计量的标量函数

整合与重构

慧科集团 Huikedu Group

- reshape包

Table 5.8. The original dataset `(mydata)`

ID	Time	X1	X2
1	1	5	6
1	2	3	5
2	1	6	1
2	2	2	4

整合与重构

慧科集团 Huikedu Group

- 融合

数据集的融合是将它重构为：每个测量变量独占一行，行中带有要唯一确定这个测量所需的标识符变量

Table 5.9. The melted dataset

ID	Time	Variable	Value
1	1	X1	5
1	2	X1	3
2	1	X1	6
2	2	X1	2
1	1	X2	6
1	2	X2	5
2	1	X2	1
2	2	X2	4

整合与重构

慧科集团 Huikedu Group

- 重铸

  newdata <- cast(md, formula, FUN)

  md为已融合的数据，formula描述想要的最后结果（公式），而FUN是（可选的）数据整合函数

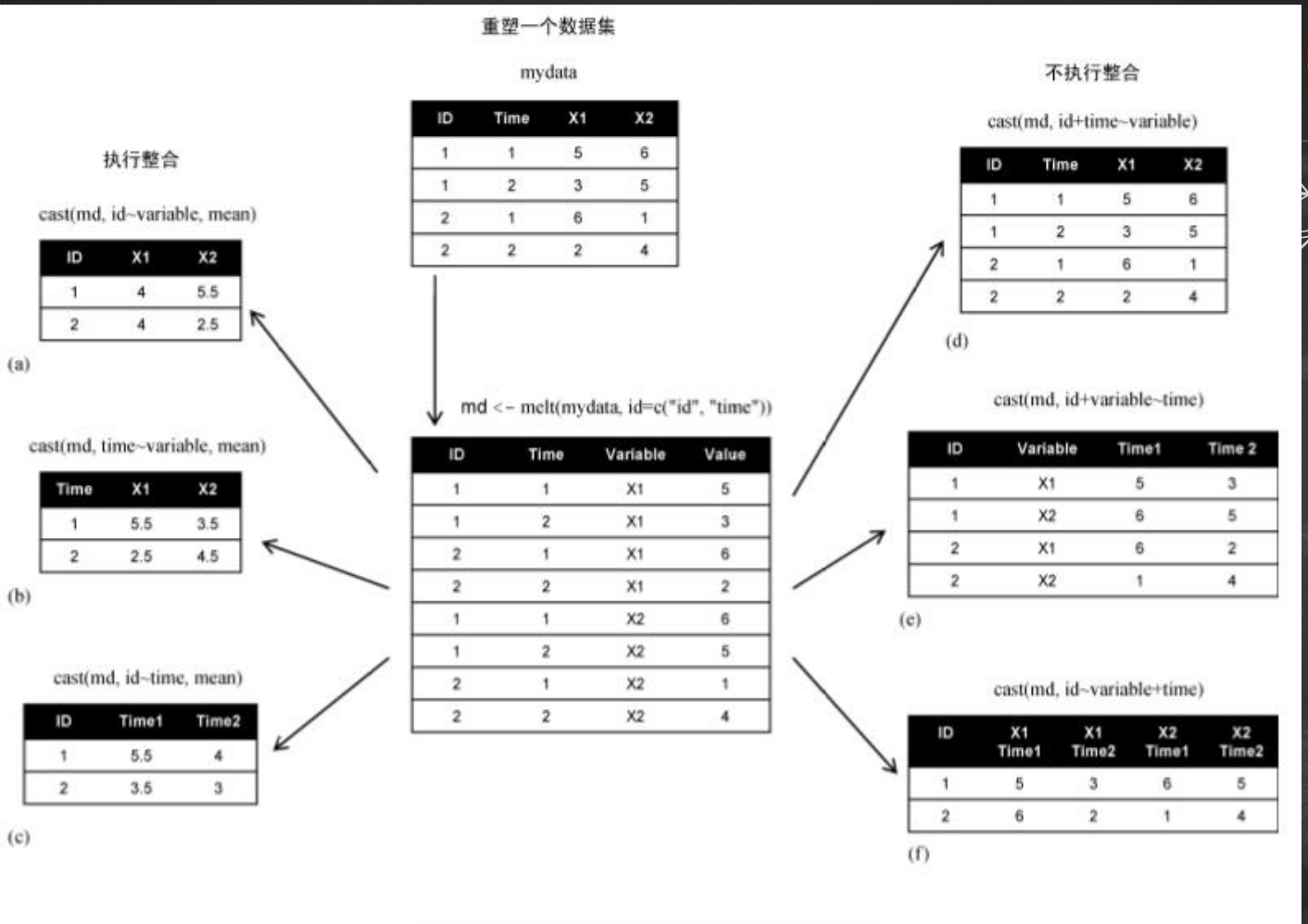


图5-1　使用函数melt()和cast()重塑数据

小结

慧科集团 Huikedu Group

- 数学和统计函数
- 字符和处理函数
- 循环和条件执行
- 自编函数
- 数据整合和重塑

慧科集团
Huikedu Group

Thankyou !