南開大學

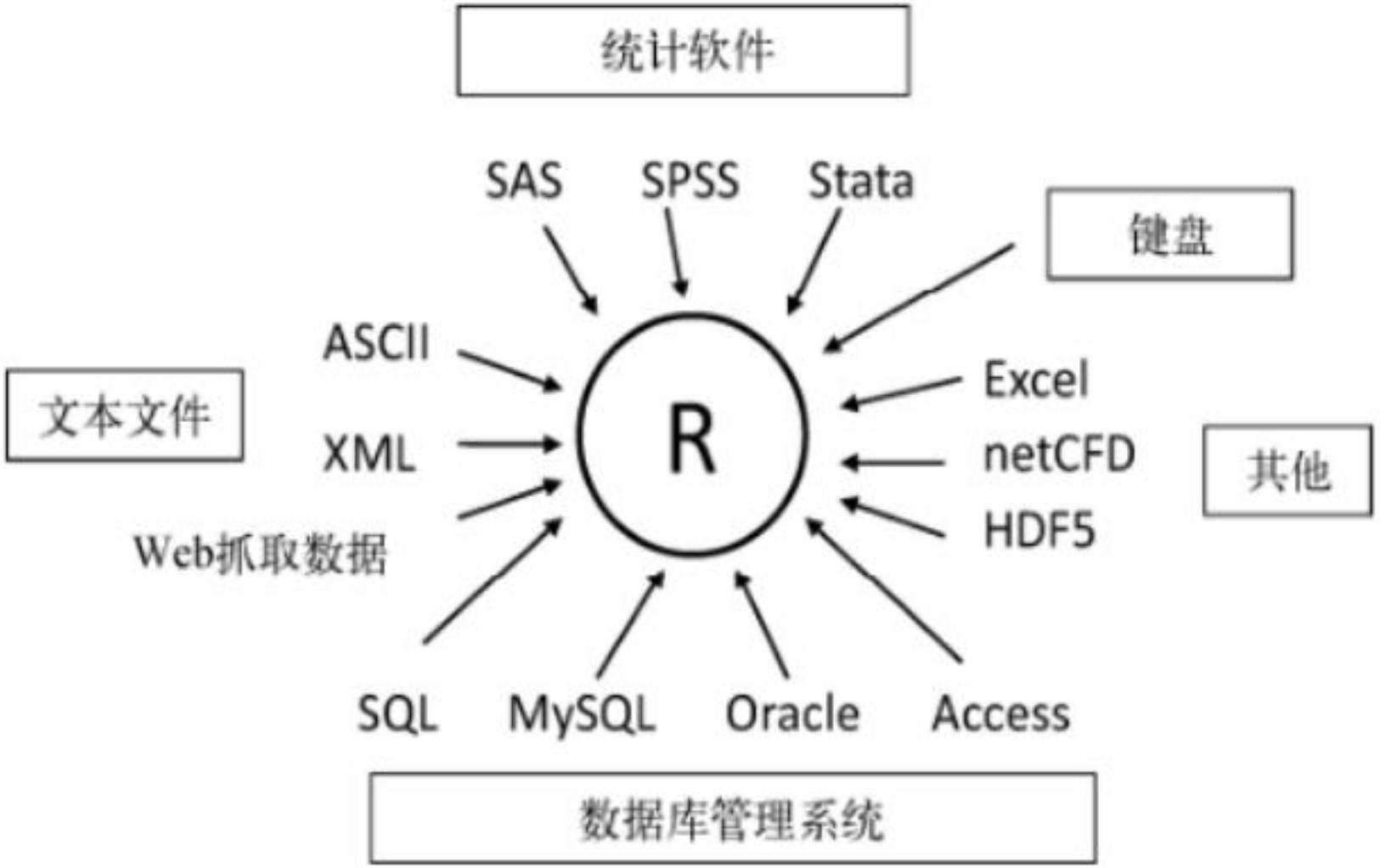
第二讲 数据的输入与基本处理

请进飞书课程群

提　纲

- 1. 数据的输入
- 2. 基本数据处理

1. 数据的输入


统计软件
SAS
SPSS
Stata
键盘
ASCII
文本文件
XML
Web抓取数据
R
Excel
netCFD
HDF5
其他
SQL
MySQL
Oracle
Access
数据库管理系统

1. 数据的输入

（a）键盘

(1) 创建一个空数据框（或矩阵），其中变量名和变量的类型需与理想中的最终数据集一致；

(2) 针对这个数据对象调用文本编辑器，输入你的数据，并将结果保存到此数据对象中。

【操作练习】使用键盘(5分钟)

```
mydata <- data.frame(age=numeric(0),
      gender=character(0),weight=numeric(0))
```

age=numeric(0) 的赋值语句将创建一个指定数据类型但不含实际数据的变量。

```
mydata <- edit(mydata)# 还可以用fix(mydata)
 View(mydata)                #查看编辑的数据
```

1. 数据的输入

（b）从带分隔符的文本文件导入数据

```
mydataframe <- read.table
    (file,header=logical_value, sep="delimiter",
     row.names="name")
```

其中，file是一个带分隔符的ASCII文本文件，header 是一个表明首行是否包含了变量名的逻辑值（ TRUE 或 FALSE ），sep用来指定分隔数据的分隔符(参数 sep 允许你导入那些使用逗号以外的符号来分隔行内数据的文件。可以使用sep="\t"读取以制表符分隔的文件。此参数的默认值为sep=""，即表示分隔符可为一个或多个空格、制表符、换行符或回车符)，row.names是一个可选参数，用以指定一个或多个表示行标识符（关键字）的变量。

1. 数据的输入

【操作练习】从带分隔符的文本文件导入数据（5分钟）

```
grades <- read.table("studentgrades.csv",
header=TRUE,row.names="StudentID", sep=",")
grades
str(grades)
gradesdata<-read.csv("studentgrades.csv")
gradesdata #与read.table有什么区别
```

1. 数据的输入

还可以：

- Excel数据
- 导入XML数据
- 从网页抓取数据
- 导入SPSS数据
- 导入SAS数据
- 导入Stata数据
- 导入netCDF数据
- 导入HDF5 数据
- 访问数据库管理系统
- 通过Stat/Transfer导入数据

2. 基本数据处理

【问题引入】思考一个典型的问题

- 处于管理岗位的男性和女性在听从上级的程度上是否有所不同？
- 这些由性别导致的不同是否在各国普遍存在？

解答这些问题的一种方法是让多个国家经理人的上司对其服从程度打分，如：

这名经理在做出人事决策之前会询问我的意见。

1	2	3	4	5
非常不同意	不同意	既不同意也不反对	同意	非常同意

2. 基本数据处理

问卷调查结果：

经理人	日 期	国 籍	性 别	年 龄	q1	q2	q3	q4	q5
1	10/24/08	US	M	32	5	4	5	5	5
2	10/28/08	US	F	45	3	5	2	5	5
3	10/01/08	UK	F	25	3	5	5	5	2
4	10/12/08	UK	M	39	3	3	4		
5	05/01/09	UK	F	99	2	2	1	2	1

2. 基本数据处理

为了解决感兴趣的问题，必须首先进行数据预处理，例如：

- 五个评分（q1到q5）需要组合起来，即为每位经理人生成一个平均服从程度得分。
- 需要一种处理不完整数据的方法。在问卷调查中，被调查者经常会跳过某些问题。例如，为4号经理人打分的上司跳过了问题4和问题5。
- 还需要处理不合理的数据，例如将99岁这样的年龄值设为缺失值。

2. 基本数据处理

- 一个数据集中也许会有数百个变量，但我们可能仅对其中的一些感兴趣。为了简化问题，我们往往希望创建一个只包含那些感兴趣变量的数据集。（特征选择）
- 既往研究表明，领导行为可能随经理人的年龄而改变，二者存在函数关系。要检验这种观点，我们希望将当前的年龄值重编码为类别型的年龄组（例如年轻、中年、年长）。
- 领导行为可能随时间推移而发生改变。我们可能想重点研究最近全球金融危机期间的服从行为。为了做到这一点，我们希望将研究范围限定在某一个特定时间段收集的数据上（比如，2009年1月1日到2009年12月31日）。

2. 基本数据处理

【操作练习】创建leadership数据框（3分钟）

```
manager <- c(1, 2, 3, 4, 5)
date <- c("10/24/08", "10/28/08", "10/1/08", "10/12/08",    "5/1/09")
country <- c("US", "US", "UK", "UK", "UK")
gender <- c("M", "F", "F", "M", "F")
age <- c(32, 45, 33, 80, 99)
q1 <- c(5, 3, 3, 3, 2)
q2 <- c(4, 5, 5, 3, 2)
q3 <- c(5, 2, 5, 4, 1)
q4 <- c(5, 5, 5, NA, 2)
q5 <- c(5, 5, 2, NA, 1)
leadership <- data.frame(manager, date, country, gender, age,
   q1, q2, q3, q4, q5, stringsAsFactors = FALSE)#不把字符串转换成因子
# 删除不在需要的变量
rm(manager, date, country, gender, age, q1, q2, q3, q4, q5)
```

2. 基本数据处理

- 创建新变量

  创建新变量或者对现有的变量变换格式：

  变量名←表达式

2. 基本数据处理

运算符	描述
x + y	加法
x - y	减法
x * y	乘法
x / y	除法
x ^ y	乘幂
x %% y	模运算
x %/% y	整数除法
x == y	判断是否相等
x <= y	判断是否小于等于
x >= y	判断是否大于等于
x && y	标量的逻辑“与”运算
x || y	标量的逻辑“或”运算
x & y	向量的逻辑“与”运算(x、y以及运算结果都是向量)
X | y	向量的逻辑“或”运算(x、y以及运算结果都是向量)
!x	逻辑非

2. 基本数据处理

【操作练习】创建变量（下面的两种方法都可以）（5分钟）

```
mydata <- data.frame(x1 = c(2, 2, 6, 4), x2 = c(3, 4, 2, 8))
#1
mydata$sumx <- mydata$x1 + mydata$x2
mydata$meanx <- (mydata$x1 + mydata$x2)/2
View(mydata)

#2
mydata <- data.frame(x1 = c(2, 2, 6, 4), x2 = c(3, 4, 2, 8))
mydata <- transform(mydata, sumx = x1 + x2, meanx = (x1+x2)/2)
View(mydata)
```

2. 基本数据处理

- 变量的重编码

重编码是根据同一个变量和/或其他变量的现有值创建新值的过程。例如：

- 将一个连续型变量修改为一组类别值；
- 将误编码的值替换为正确值；
- 基于一组分数线创建一个表示及格/不及格的变量；
- ……

提示：这些操作需要结果为逻辑值的表达式，即关系表达式或逻辑表达式

2. 基本数据处理

【操作练习】变量重编码：将leadership数据集中经理人的连续型年龄变量age重编码为类别型变量agecat（Young、 Middle Aged、Elder）（5分钟）。

```
leadership$agecat[leadership$age >= 99] <- 99
leadership$agecat[leadership$age >75 &
                  leadership$age < 99] <- "Elder"
leadership$agecat[leadership$age >45 &
                  leadership$age <= 75] <- "Middle Aged"
leadership$agecat[leadership$age <= 45] <- "Young"
```

2. 基本数据处理

- 变量的重命名
  - 可以交互地或者以编程的方式修改变量名。可以使用**fix(dataname)**调用一个交互式的编辑器，单击变量名，然后在弹出的对话框中将其重命名。
  - 若以编程方式，**reshape**包中有一个**rename()**函数，可用于修改变量名:

rename(dataframe,c(oldname1=”newname1”,oldname2=”newname2”,…))

reshape 包未被默认安装， 所以在首次使用它之前需要先使用**install.packages(“reshape”)**命令安装它。

  - 还可以通过**names()**函数来重命名变量。

2. 基本数据处理

【操作练习】变量重命名（5分钟）

```
fix(leadership)#看一下数据框的内容，可以修改，但此处不修改
install.packages("reshape")
library(reshape)
leadership
names(leadership)   #查看数据框所有变量名
leadership1<-leadership
names(leadership1)[1] <- "managerID "
leadership1<-rename(leadership1,c(date="testDate") )
names(leadership1)[6:10] <-
      c("item1","item2","item3","item4","item5")
leadership1
names(leadership1)
```

2. 基本数据处理

- 缺失值
  - 在任何规模的项目中，数据都可能由于未回答问题、设备故障或误编码数据的缘故而不完整。在R中，缺失值以符号NA（Not Available，不可用）表示。不可能出现的值（例如，被0除的结果）通过符号NaN（Not a Number，非数值）来表示。
  - 函数is.na() 检测缺失值是否存在。
  - na.omit()可以删除所有含有缺失数据的行。

2. 基本数据处理

【操作练习】缺失值处理（5分钟）

```
is.na(leadership[, 6:10])
leadership
#删除缺失值的行
newdata <- na.omit(leadership)
newdata
#缺失值重编码
newdata <- leadership
newdata
newdata$age[newdata$age == 99] <- NA
newdata
```

2. 基本数据处理

Break Time

5 minutes

2. 基本数据处理

- 日期值

日期值通常以字符串的形式输入到R中，然后转化为以数值形式存储的日期变量：

as.Date(x, "input_format")

其中*x*是字符型数据，input_format则给出了读入日期的格式:

符　号	含　义	示　例
%d	数字表示的日期（0~31）	01~31
%a	缩写的星期名	Mon
%A	非缩写星期名	Monday
%m	月份（00~12）	00~12
%b	缩写的月份	Jan
%B	非缩写月份	January
%y	两位数的年份	07
%Y	四位数的年份	2007

2. 基本数据处理

【操作练习】日期值处理（5分钟）

```
mydates <- as.Date(c("2007-06-22", "2004-02-13"))
strDates <- c("01/05/1965", "08/16/1975")
dates <- as.Date(strDates, "%m/%d/%Y")
myformat <- "%m/%d/%y"
leadership$testDate <- as.Date(leadership$date, myformat)
leadership
Sys.Date()#系统时间日期
date()     #系统日期时间函数
today <- Sys.Date()
today
```

2. 基本数据处理

- 类型转换

R中提供了一系列用来判断某个对象的数据类型和将其转换为另一种数据类型的函数。

判　断	转　换
is.numeric()	as.numeric()
is.character()	as.character()
is.vector()	as.vector()
is.matrix()	as.matrix()
is.data.frame()	as.data.frame()
is.factor()	as.factor()
is.logical()	as.logical()

2. 基本数据处理

【操作练习】转换数据类型（5分钟）

```
a <- c(1, 2, 3)
a
is.numeric(a)
is.vector(a)
is.logical(a)
a <- as.character(a)
a
is.numeric(a)
is.vector(a)
is.character(a)
```

2. 基本数据处理

- 数据排序

有些情况下，查看排序后的数据集可以获得相当多的信息。例如，哪些经理人最具服从意识？

在R中，可以使用**order()**函数对一个数据框进行排序。默认的排序顺序是升序。在排序变量的前边加一个减号即可得到降序的排序结果。

2. 基本数据处理

【操作练习】数据排序（5分钟）

```
attach(leadership)
newdata <- leadership[order(age), ]
newdata
newdata <- leadership[order(gender, -age), ]
newdata
detach(leadership)
```

2. 基本数据处理

- 数据集的合并
  - 横向合并：要横向合并两个数据框（数据集）。

    A.使用merge()函数。在多数情况下，两个数据框是通过一个或多个共有变量进行联结的（即一种内联结，inner join）。

    B.如果要直接横向合并两个矩阵或数据框，并且不需要指定一个公共索引，那么可以直接使用cbind()函数。适合情况：两个对象必须拥有相同的行数，且要以相同顺序排序。

2. 基本数据处理

【操作练习】merge和cbind的用法（5分钟）

```
ID<-c(1,2,3,4)
name<-c("Jim","Tony","Lisa","Tom")
score<-c(89,22,78,78)
student1<-data.frame(ID,name)
student2<-data.frame(ID,score)
student1
student2
ID<-c(1,2,3,5)
student3<-data.frame(ID,score)
student3
total_student1<-merge(student1,student2,by="ID")
total_student1
total_student2<-merge(student1,student3,by="ID")
total_student2
total<-cbind(student1,student2)
total
```

2. 基本数据处理

【操作练习】merge()的all用法

```
id=c("1","2","4")
R=c("9","7","9")
ink1=data.frame(id,R)
id=c("1","2","3")
M=c("7","2","3")
ink2=data.frame(id,M)
ink1
ink2
merge(ink1,ink2,by="id",all=T)
merge(ink1,ink2,by="id",all=F) #默认all=F
```

2. 基本数据处理

- 纵向合并：要纵向合并两个数据框（数据集），需使用**rbind()**函数。两个数据框必须拥有相同的变量，不过它们的顺序不必一定相同。

2. 基本数据处理

【操作练习】rbind的用法（3分钟）

```
ID<-c(1,2,3)
name<-c("Jame","Kevin","Sunny")
student1<-data.frame(ID,name)
ID<-c(4,5,6)
name<-c("Sun","Frame","Eric")
student2<-data.frame(ID,name)
student1
student2
total<-rbind(student1,student2)
total
```

2. 基本数据处理

- 数据集取子集
  - 选择变量

数据框中的元素是通过dataframe[row indices, column indices]这样的记号来访问的。

【操作练习】（2分钟）

```
newdata <- leadership[, c(6:10)]
newdata
myvars <- c("q1", "q2", "q3", "q4", "q5")
newdata <- leadership[myvars]
newdata
```

2. 基本数据处理

- 剔除（丢弃）变量（投影操作）

剔除变量的原因有很多。例如，如果某个变量中有若干缺失值，你可能就想在进一步分析之前将其丢弃。丢弃变量是保留变量的逆向操作。

【操作练习】剔除变量（2分钟）

```
leadership
newdata <- leadership[c(-7, -8)]#方法1
newdata
newdata$q4 <- NULL#方法2
newdata
```

2. 基本数据处理

- 选入观测（选择操作）

选入或剔除观测（行）通常是成功的数据准备和数据分析的一个关键。

【操作练习】选择观测（5分钟）

```
newdata <- leadership[1:3, ]
newdata
newdata <- newdata[which(newdata$gender == "M" & newdata$age>30), ]
newdata
newdata <- leadership[which(leadership$gender == "M" & leadership$age > 30), ]
newdata
```

2. 基本数据处理

```
#选择某一事件范围的行
startdate <- as.Date("2008-10-01")
enddate <- as.Date("2008-10-15")
newdata <- leadership[leadership$testDate >=
startdate & leadership$testDate <= enddate, ]
newdata
```

2. 基本数据处理

subset()函数

使用subset函数是选择变量和观测的一种简单方法。

【操作练习】subset函数（5分钟）

```
newdata <- subset(leadership, age >= 70 | age < 45,
                  select = c(q1, q2, q3, q4))
newdata
newdata <- subset(leadership, gender == "M" & age >
                    25, select = gender:q4)
newdata
```

2. 基本数据处理

随机抽样

假设需要选择两份随机样本，使用其中一份样本构建预测模型，使用另一份样本验证模型的有效性。**sample()**函数能够从数据集中（有放回或无放回地）抽取大小为n的一个随机样本。R中拥有齐全的抽样工具，包括抽取和校正调查样本（参见**sampling**包）以及分析复杂调查数据（参见**survey**包）的工具。

2. 基本数据处理

【操作练习】sample函数（5分钟）

```
mysample <-
leadership[sample(1:nrow(leadership),3,replace =
FALSE),] # replace = FALSE 无放回
mysample
mysample <-
leadership[sample(1:nrow(leadership),4,replace =
TRUE),] #  有放回
mysample
```

3. 课下作业

1、重复练习（但不限于）课上练习的内容，将操作界面截图到Word文档，发word文档到学堂云交作业。

2、你和你的小伙伴尝试去发现身边有什么问题值得你们去研究。发你研究的题目到学堂云，包括：

题目

背景及存在的问题

研究的意义