第5章

数据重构

数据重构是指将数据按照一定的规则或要求进行转换、整合，以适应特定的数据分析或数据挖掘任务。数据重构的目的是使数据更具有可分析性、可理解性和可利用性。经过数据重构，原始数据可以被转化为更易于分析和理解的形式，从而使得分析或挖掘的效率更高，结果更准确。本章涉及的数据重构的内容主要包括数据拆分、数据合并、数据长宽转化、数据转置以及数据的变列操作。

5.1 数据拆分

数据拆分主要利用 keep 命令和 drop 命令结合 if 与 in 实现对数据子集或者子样本的提取。数据拆分可以分为横向拆分与纵向拆分两种情形。横向拆分根据变量进行拆分，纵向拆分则根据数据的观测值进行拆分。

5.1.1 横向拆分数据

横向拆分在操作上把满足要求的变量和其对应的数据提取为一个单独的数据集即可。这里以本章数据文件夹下的“上市公司财务信息.dta”数据为例进行说明。上市公司信息相关的数据可以在国泰安数据库、Wind 数据库下载得到。本章使用的“上市公司财务信息.dta”经过了脱敏处理，并对相关变量的数据进行了微调。该数据集包括证券代码(id)、年份(year)等企业基本信息，金融资产持有比例(finratio)、总负债率(debt)、营业收入(income)等财务变量信息，以及所在的地区(area)、省份(prov_new)、行业(sic_men)及股权性质(EquityNatureID_p)等属性变量信息。要求将“上市公司财务信息.dta”数据集分成三个子数据文件，分别为数据信息文件 finance.dta、行业区域信息文件 indusregion.dta 和产权性质文件 property.dta。

例 5-1 横向数据拆分。

```
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use 上市公司财务信息.dta,clear      //打开数据文件
drop sic_men sic_da area prov_new city1 Lng Lat city_reg EquityNatureID_p Ownership_p govcon2_p
```

```
                                        //将 drop 后的变量数据删掉
save finance.dta, replace               //将删除后的数据集命名,并保存到当前路径

* 将"上市公司财务信息.dta" 拆分成行业区域信息文件 indusregion.dta
use 上市公司财务信息.dta,clear          //打开数据文件
keep id year sic_men sic_da area prov_new city1 Lng Lat city_reg   //保留 keep 后的变量数据
save indusregion.dta, replace           //将当前数据集重新命名,并保存到当前路径

* 将"上市公司财务信息.dta" 拆分成产权性质文件 property.dta
use 上市公司财务信息.dta,clear          //打开数据文件
keep id year EquityNatureID_p Ownership_p govcon2_p //保留 keep 后的变量数据
save property.dta, replace              //将当前数据集重新命名,并保存到当前路径
```

5.1.2 纵向拆分数据集

纵向拆分在操作上把满足要求的观测值单独保存为一个数据集即可。这里仍然以“上市公司财务信息.dta”的数据为例,要求根据变量 area 的信息(area 的取值为 1、2、3,分别代表了东部地区、中部地区和西部地区)将整个样本分成三个数据文件,分别代表东部地区数据集 dong.dta、中部地区数据集 zhong.dta 和西部地区数据集 xi.dta。

例 5-2 纵向数据拆分。

```
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use 上市公司财务信息.dta,clear                  //打开数据文件
keep id year debt tang roa area city_reg        //保留 keep 后的变量数据
preserve //preserve 和下面的 restore 连用,可以在操作结束后恢复 preserve 之前的数据
keep if area == 1                               //1 代表东部地区
save dong.dta, replace                          //将当前数据集重新命名,并保存到当前路径
restore
preserve
keep if area == 2                               //2 代表中部地区
save zhong.dta, replace                         //将当前数据集重新命名,并保存到当前路径
restore
keep if area == 3                               //3 代表西部地区
save xi.dta, replace                            //将当前数据集重新命名,并保存到当前路径
```

5.2 数据合并

数据合并分为纵向数据合并与横向数据合并,它们是数据拆分的反向操作。纵向数据合并是把变量相同、来源不同的数据纵向合并为一个整体。比如学校要统计期末考试成绩,不同的班级均涉及学生姓名、学生学号及期末成绩三个变量信息,因此将这三个变量所在班级的观测结果进行纵向合并就得到了全校的数据。横向数据合并是以特定的变量为关键词,对数据进行匹配。比如学校的期末考试成绩数据文件有学生的姓名、学号及期末成绩信息,现在学校想把学生的期末考试成绩数据和期中考试成绩数据进行合并(期中考

试成绩数据也包含学生姓名和学号的信息),直接合并是肯定不行的。我们可以根据学生的姓名将期末成绩和期中成绩进行匹配,为防止同名的情况发生,最为保险的方法是根据“学生姓名-学生学号”两个关键词,将学生的期末成绩和期中成绩进行匹配,达到横向合并的目的。

5.2.1　纵向合并数据

纵向合并数据是纵向拆分的反向操作,指将不同来源的数据在观测值的维度上进行合并,一般要求这些不同来源的数据完全具有相同的变量名。

1. 使用 append 命令合并 dta 文件

Stata 软件默认的数据格式是 dta 文件,在 Stata 中纵向合并 dta 文件使用的命令为 append,该命令的语法为

```
append using filename [filename ...] [, options]
```

其中,filename 表示文件名; options 为功能选项,主要包括 force 选项和 gen(newvar)选项。force 选项用于强制合并两个不同类型的变量,而不报错。gen(newvar)选项用于生成一个新的变量 newvar,并显示观测值的来源文件,0 为合并之前内存中的数据,1 为第一个合并的数据,2 为第二个合并的数据。

例 5-3　将东部、中部、西部数据进行合并。

```
* 将前面拆分的东部、中部、西部数据进行合并
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use dong, clear              //打开数据文件
append using zhong           //append using 加文件名,将数据进行纵向合并
append using xi              //append using 加文件名,将数据进行纵向合并
save area, replace           //将当前数据集重新命名,并保存到当前路径
* 也可以直接用 append
clear
append using dong   //当内存中没有数据时,直接使用 append 相当于读入数据(和一个空数据直接拼接)
append using zhong
append using xi
save area1, replace
* 进一步简写
clear
append using dong zhong xi //东部、中部和西部地区的数据会依次排序衔接下去
save area2, replace
* 选项 force、gen
use dong, clear
tostring debt, replace     //把数值转化为字符型
append using zhong,force gen(appvar)
save area4, replace
```

2. 用 openall 命令合并 dta 与 vcs 文件

openall 为外部命令,可以对 dta 与 csv 格式的多个数据集进行纵向合并,实现批量纵向

合并的功能，在面对海量的文件合并时，能很好地提高工作效率。该命令的语法结构为

```
openall [files], [directory(string)] [storefilename(string)] [insheet]
```

其中，files 表示文件名，可结合通配符 * 和？使用。需要注意的是，这里不能指定文件的扩展名，如“.dta”“.csv”等。其选项功能主要包括 directory 参数、storefilename 参数以及 insheet 参数等。directory 定义要合并文件所在的路径，若不指定则默认为当前工作目录。storefilename 用于生成一个新的变量，标识数据来源。insheet 用于指定合并文件的类型为 csv 格式。

另外需要注意的是，openall 命令可自动清除当前内存中的数据，然后纵向合并指定的文件。该命令的主要使用方法示意如下。

```
openall *    //* 表示默认路径下所有 dta 文件
openall      //等价于 openall *，合并路径下所有 dta 文件
count        //显示观测值
openall ,storefilename(source) //新生成一个变量 source,显示观测值的文件来源
tab source   //将所选数据表格化
openall ?, storefilename(source) //合并当前路径下文件名为一个字符的所有 dta 文件
tab v        //将所选数据表格化
openall * , directory(cd "D:\Stata 数据分析与建模\数据代码\第 5 章\openall")
             //定义要合并文件所在的路径
openall, insheet storefilename(source) directory(cd "D:\Stata 数据分析与建模\数据代码\第 5
章\openall") //合并当前路径下所有的 csv 文件，insheet 用于指定合并文件的类型为 csv 格式
```

类似的命令还有 csvconvert，该命令用于将多个 csv 格式文件合并为一个 dta 格式文件，csvconvert 命令比较适合处理具有时间周期性特点的变量。

3. 合并 xls 或 xlsx 文件

对于 xls 或 xlsx 类型的文件需先使用 import excel 命令转化为 dta 文件，再使用 append 命令进行合并，也可以使用 xls2dta 命令实现批量操作。

例 5-4 合并 xls 文件。

```
clear
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
shellout FS_Combas - 1.xls //shellout 命令可以直接从 Stata 内部打开一个文件
* 转化第一个 excel 文件，这里仅以 FS_Combas - 1.xls,FS_Combas - 2.xls 为例
import excel using FS_Combas - 1.xls, first case(lower) clear //打开 Excel 文件，first 表示将
                                                              //Excel 第一行的数据作为变量
                                                              //名，case(lower)表示将所有大
                                                              //写字母变小写
labone ,nrow(1)            //将第一行变为标签
drop in 1/2                //删除前两行
save f1.dta, replace       //保存
* 转化第二个 excel 文件
import excel using FS_Combas - 2.xls, first case(lower) clear
labone ,nrow(1)
drop in 1/2
```

```
save f2.dta, replace
* 文件合并
clear                          //合并之前,清空内存
append using f1.dta
append using f2.dta            //这两步可合并为 1 步:append using f1 f2
count                          //查看内存中观测值总数
save 资产负债表,replace
* 进一步处理
destring , replace             //转化为数值
split accper, p("-")           //根据 - 的字符将日期新建成字符型的年份、月份、日的数据
destring , replace
encode typrep, gen(type)       //encode(字符编码为数值)
erase f1.dta
erase f2.dta
```

5.2.2　横向合并与匹配


视频讲解

横向数据合并是横向拆分的反向操作,根据关键变量将不同来源的数据文件进行整合。如将来自不同数据库的上市公司信息,根据上市公司的代码和年份实现合并;将来自不同数据库的省份或城市统计指标依据省份或城市进行合并等。

1. 横向合并数据

merge 命令是 Stata 中一个用于横向合并数据的命令。它可以将两个数据集按照某个或某些变量进行合并,其中一个数据集称为“主数据集”,另一个数据集称为“副数据集”。如果某个观测值在主数据集中存在但在副数据集中不存在,那么这个观测值将被保留在主数据集中,但是在副数据集中不存在的变量值将被设为缺失值。merge 命令可以对数据进行一对一、一对多、多对一以及多对多等多种类型的合并。

一对一合并:

```
merge 1:1 varlist using filename [, options]
```

一对多合并:

```
merge 1:m varlist using filename [, options]
```

多对一合并:

```
merge m:1 varlist using filename [, options]
```

多对多合并:

```
merge m:m varlist using filename [, options]
```

其中,1∶1、1∶m、m∶1 等为合并类型,分别代表一对一、一对多、多对一以及多对多的匹配。varlist 为合并的依据,即所谓的“关键字”(一个或多个变量)。using 后面是副数据集的文件名,即 filename。options 是一些可选的参数,比如 keep、drop 等。在匹配后,Stata

会生成一个名为_merge的变量，用于标识匹配的结果。_merge＝1表示正在使用的数据，_merge＝2表示合并的数据，_merge＝3表示成功合并的数据。需要注意的是，在使用merge命令前，要确保两个数据集至少有一个共同变量，否则无法进行匹配。

(1) 一对一横向合并。

将上市公司的数据信息文件finance.dta、行业区域信息文件indusregion.dta和产权性质文件property.dta按照上市公司的公司代码和年份进行合并。

例 5-5 一对一合并数据。

```
cd "D:\Stata数据分析与建模\数据代码\第5章"
use finance.dta, clear
merge 1:1 id year using indusregion.dta //根据 id year 变量用 merge 1:1 进行一对一合并
keep if _merge == 3 //合并后会生成_merge变量,其中_merge==3代表合并成功的观测,keep if _m
                    // == 3 只保留合并成功的数据
drop _merge         //删除_merge变量
merge 1:1 id year using property.dta
keep if _merge == 3
drop _merge
save hebing1.dta, replace
```

将上市公司的资产负债表、利润表与现金流量表进行横向合并。

例 5-6 横向合并财务报表。

```
cd "D:\Stata数据分析与建模\数据代码\第5章"
use 资产负债表,clear
merge 1:1 stkcd year using 利润表
keep if _merge == 3
drop _merge
merge 1:1 stkcd year using 现金流量表
keep if _merge == 3
drop _merge
save hebing2.dta, clear
```

(2) 多对一(一对多)横向合并。

多对一合并中的“多”是指varlist中有重复取值的那一个文件，“一”指varlist中有唯一取值的那一个文件。

例 5-7 将企业财务信息与行业信息进行横向合并。

```
*1:m与m:1这两个是等价的,区别在于先导入哪个文件
cd "D:\Stata数据分析与建模\数据代码\第5章"
use 上市公司经营活动.dta,clear
merge m:1 stkcd using 行业分类 //m指varlist中有重复取值的那一个文件,1指varlist中有唯一
                               //取值的那一个文件,这里"上市公司经营活动"文件中的股票代码
                               //stkcd是重复的,而"行业分类"文件中的stkcd是唯一的
keep if _m == 3
use 行业分类,clear
merge 1:m stkcd using 上市公司经营活动   //m指varlist中有重复取值的那一个文件,1指
                                        //varlist中有唯一取值的那一个文件
keep if _m == 3
```

（3）多对多合并横向合并。

使用 merge 命令进行多对多合并容易出现意料之外的情形，因此不建议使用该命令进行多对多合并，示例如下。

例 5-8　多对多合并 m：m。

```
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
clear
input id str3 v1
1 "a"
1 "b"
1 "c"
2 "d"
2 "e"
end
save 1.dta, replace
clear
input id str3 v2
1 "f"
1 "g"
2 "h"
2 "i"
2 "j"
end
save 2.dta, replace
 * 直接用 merge m:m 横向合并数据
use 1.dta, clear
merge m:m id using 2.dta
drop _m //观察合并结果后发现这个命令合并的结果没有太大意义，一般不建议使用
```

当需要横向合并的数据集比较多的时候，可以使用 mergemany 命令，该命令可以一次性横向合并多个文件数据集，其语法格式为

```
mergemany 1:1 filename1 filename2..., match(varlist) [options]
```

其中 filename1、filename2...，表示文件名；options 表示选择项，包括 match(varlist)、saving(filename)、all、verbose、import(filetype)等。各参数说明如下。match(varlist)指定合并依据（关键字），此选项必须指定。

saving(filename)表示保存合并结果并指定文件名为 filename；

all 表示合并当前路径中的所有文件；

verbose 用于创建一个变量来标记合并结果，变量名为_merge_filename。

import(filetype)表示允许非.dta 文件直接导入和合并。

例 5-9　合并资产负债表、利润表、现金流量表。

```
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
 * 使用 merge 依次合并
clear
use 资产负债表,clear
```

```
merge 1:1 stkcd year using 利润表
keep if _m == 3
drop _m //不删除要出错,变量名冲突
merge 1:1 stkcd year using 现金流量表
keep if _m == 3
drop _m
save 财务报表1, replace
count
* 使用 mergemany 一次性横向合并多个文件
* ssc install mergemany,replace
clear
mergemany 1:1 资产负债表 利润表 现金流量表, match(stkcd year) //合并资产负债表、利润表、现金
                                                                  //流量表三个 dta 文件
save 财务报表2,replace
count   //发现与上面 merge 1:1 不一样,原因在于_merge == 1 与_merge == 2 在 mergemany 中保留
        //了,而 merger1:1 的操作只保留了_merge == 3 的观测值
clear
mergemany 1:1 资产负债表 利润表 现金流量表, match(stkcd year) saving(财务报表3) verbose
        //verbose:显示合并结果,一般要显示结果,然后只保留 == 3 的观测值
keep if _merge_利润表 == 3 & _merge_现金流量表 == 3
drop _m*
mergemany 1:1 , match(stkcd year) saving(财务报表4) all //合并当前目录下的所有 dta 文件
mergemany 1:1 , match(stkcd year) saving(财务报表5) all import(csv) //合并当前目录下的所有
                                                                    //csv 文件
```

2. 横向模糊匹配

merge 命令的横向合并实际上就是基于一个或多个关键变量,进行 1∶1、1∶m、m∶1 以及 m∶m 数据精确匹配操作,匹配样本关键变量要求是完全相同的。但在现实操作中,经常会遇到需要合并的变量在两个数据集中并没有完美的对应联系,这时候就需要用到模糊匹配。

比如想按照人名或者学校名或者公司名进行匹配,然而同一个人/学校/公司常常有相似却不完全相同的名称,这种情况尤其常见。例如,一个数据集中人名为“李白”,而另一个数据集中是“李太白”;一个是“江苏”,另一个是“江苏省”;或者按照公司名匹配时出现“平安保险(集团)股份有限公司”或“中国平安保险股份有限公司”,其实两者代表的是同一个公司,但是在 merge 的精确匹配下是无法匹配成功的,因此“模糊匹配”应运而生。下面对 Stata 自带的 matchit 以及 reclink 两个模糊匹配命令进行简要介绍。

(1) matchit 命令。

matchit 通过执行基于字符串的匹配技术,在两个不同的文本字符串之间提供相似性评分。该命令会返回一个新的匹配程度变量(similscore),其中包含从 0 到 1 的相似性评分。根据所选择的字符串匹配技术,similscore 为 1 意味着完全相似,当匹配不太相似时,相似度会降低。匹配程度是一个相对的度量,它可以根据所选择的技术而改变。该命令的语法结构如下:

同一数据集中两列数据的匹配

```
matchit varname1 varname2 [, options]
```

两个不同数据集中的数据匹配

```
matchit idmaster txtmaster using filename.dta, idusing (varname) txtusing(varname) [options]
```

其中，idmaster 表示来自当前文件（主文件）的数值型变量名（注：idmaster 应该唯一识别 txtmaster，即两者传达的是同一信息）；txtmaster 表示当前文件（主文件）中要进行匹配的字符型变量名；idusing(varname)与 txtusing(varname)同理，分别指来自 filename.dta 数据集中待匹配的数值型变量以及字符型变量。相关选项和参数说明如下。

similmethod(simfcn)表示字符串匹配方法。默认为二元组。其他内置 simfcn 有 ngram、ngram_circ、firstgram、token、cotoken、scotoken、soundex、soundex_nara、soundex_fk、soundex_ext、token_soundex 和 nysiis_fk。

score(scrfcn)指定相似度得分，默认为 Jaccard。其他内置选项有 simple 和 minsimple。

weights(wgtfcn)表示权重变换，默认为 noweights。其他内置选项有 simple、log 和 root。

generate(varname)指定相似度得分变量的名称，默认为 similscore。

两个数据集匹配的必需命令选项如下：

idmaster 表示当前文件（masterfile）中的数值型变量，无须唯一识别 masterfile 中样本。

txtmaster 表示当前文件（masterfile）中的字符串，将与 txtusing 匹配。

using(filename)指定要与 masterfile 匹配的 Stata 文件的名称（usingfile）。

idusing(varname)表示 usingfile 中的数值型变量，无须唯一识别 usingfile 中样本。

txtusing(varname)表示 usingfile 中的字符串，将与 txtmaster 匹配。

通用高级命令选项如下：

wgtfile(filename)表示允许从其他 Stata 文件加载权重，而不是基于 masterfile 和 usingfile 计算权重。默认不加载权重。

time 表示在执行期间输出时间戳。

flag(step)用于控制 matchit 返回屏幕的频率。只有通过尝试不同的 simfcn 来优化索引才真正有用，默认值为 step=20（百分比）。

仅对两个数据集匹配适用的高级命令选项如下：

threshold(num)表示最终结果中保留的最低相似性分数，默认值为 num = .5。

override 表示忽略未保存的数据警告。

diagnose 表示报告有关索引的初步分析，用于通过清理原始数据和尝试不同的 simfcn 来优化索引。

stopwordsauto 用于自动生成停用词列表。它提高了索引速度，但忽略了潜在的匹配可能性。

需要注意的是，matchit 区分大小写以及其他所有符号。虽然 matchit 使用前不需要数据清理，但一定程度的数据清理通常能提高相似度分数，从而提高匹配质量。但是过多的数据清理可能会导致数据信息缺失，降低匹配质量。另外需要注意的是，在软件自动匹配后仍需要人工检查和筛选匹配的结果，以确保不出现错误。

(2) reclink 命令。

reclink 本质上也是一种模糊匹配方法。在无法精确匹配关键字段的两个数据集之间，reclink 使用记录链接方法来匹配观察结果。该命令语法为

```
reclink varlist using filename , idmaster(varname) idusing(varname) gen(newvarname) [ wmatch
(match weight list) wnomatch(non - match weight list) orblock(varlist) required(varlist)
exactstr(varlist) exclude(filename) _merge(newvarname) uvarlist(varlist) uprefix(text)
minscore(#) minbigram(#)
```

相关参数和选项说明如下：

idmaster(varname)用于指定唯一识别 masterfile 样本值的变量。如果唯一标识符不存在，可以简单地创建为 gen idmaster=_n。

idusing(varname)用于指定唯一识别 usingfile 样本值的变量名称，与 idmaster(varname)类似。

gen(newvarname)用于指定由 reclink2 创建的存储匹配分数(分数范围：0～1)的新变量名称。

wmatch(numlist)用于指定参与数据匹配的每个变量的匹配权重。每个变量都需要匹配权重，若未指定，所有变量匹配权重默认为 1。权重通常是 1～20 的整数。权重反映了变量准确匹配后数据样本能够准确匹配的相对可能性。例如，和海关数据匹配时，可以赋予企业名称一个较大的权重(例如 10)，而邮编(多个企业使用一个邮编)的权重可能只有 2。

wnomatch(numlist)用于指定参与数据匹配的每个变量的不匹配权重，类似于 wmatch(numlist)，但反映了变量无法匹配后数据样本无法匹配的相对可能性。权重越小表明数据样本准确匹配情况下，变量之间的不匹配可能性越大。例如，电话号码的 wmatch 较大，wnomatch 较小(电话号码随时间变化或同一个人/实体拥有多个电话号码)。

orblock(varlist | none)用于通过提供一种从 usingfile 中选择样本子集进行数据匹配的方法来加速匹配。该过程仅保留 usingfile 中与 orblock 中至少一个变量匹配的样本。如果 orblock 指定了 4 个及以上变量，则默认指定了全部变量，等同于设置 orblock(none)。有时可以在 masterfile 和 usingfile 中创建新变量，例如名字和姓氏的首字母、从地址中提取的街道号码和电话区号，同时结合 orblock 选项来提高匹配速度。

required(varlist)允许用户指定一个或多个变量，这些变量必须完全匹配才能将样本视为匹配。

exclude(filename)允许用户指定包含之前匹配样本的文件名称，从而在当前的数据匹配中剔除之前匹配成功的样本。对不同数据样本使用不同匹配方法(例如通过 orblock 指定不同变量)，使得匹配过程更加灵活。

_merge(varname)指定标记数据样本来源的变量名称，默认为_merge(_merge)。

exactstr(varlist)允许用户指定一个或多个用于准确比较的字符串变量，比较结果为 0 或 1（二元变量）。

uvarlist(varlist)允许 usingfile 具有与要匹配 masterfile 变量不同的匹配变量名称。使用该选项时，两个数据集中变量数据和对应变量顺序必须相同。

uprefix(string)允许修改来源于 usingfile 中变量的前缀（默认前缀为 U）。例如，如果匹配变量是 name 和 address，则生成的匹配数据集将包含来源于 usingfile 的变量 Uname 和 Uaddress。

minscore(#)指定两个严格不能匹配的最小匹配得分值（范围为 0～1），默认为 0.6。usingfile 中只有最高匹配分数并且匹配分数 minscore 的样本才会合并到 masterfile 中。

manytoone 指定 reclink2 将 usingfile 中的样本与 masterfile 中的多条样本相匹配（多对一链接过程）。reclink 首先集中找到并删除两个数据完全匹配的样本对。因此，与 masterfile 中的样本完全匹配的 usingfile 中的样本随后无法链接到 masterfile 中的其他样本。

npairs(#)指定程序保留 usingfile 中与 masterfile 样本匹配成功前 # 个样本（高于最低分数阈值）。relink 仅保留匹配分数最高的样本。

常用的简要命令形式为

```
reclink varlist using filename , idmaster(varname) idusing(varname) gen(newvarname) [required
(varlist)]。
```

其中，reclink varlist 中的 varlist 表示待匹配的一系列变量名称，idmaster(varname)指定主数据集中唯一标识观测值的变量的名称，idusing(varname)与 idmaster 类似，表示 filename 中唯一标识观测值的变量的名称，gen(newvarname)则是创建的一个新变量的名称，该变量用于存储匹配值之间的匹配分数（范围为 0～1）。required(varlist)为可选择的命令，其允许用户指定一个或多个必须完全匹配的变量，这样观察才能被视为匹配。

5.2.3　交叉合并


视频讲解

交叉合并是数据多对多合并的一种情形。前文讲到使用 merge 的 m：m 可以实现多对多合并的功能但容易出错，这里介绍另外两个实现多对多合并功能的命令：cross 命令和 joinby 命令。

1. cross 命令

cross 命令可以创建两个数据集的交叉数据集，这个交叉数据集的每个观测值都是原来两个数据集的观测值的组合，其语法格式为

```
cross using filename
```

注意 cross 后面并没有 varlist，使用的时候需要注意。

例 5-10 cross 命令的理解。

```
* 为了理解 cross 命令的含义,首先运行并理解下面的代码案例
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
clear
* 创建 sex 数据集
input str6 sex
male
female
end
save sex.dta, replace
drop _all   //清空数据集
* 创建 agecat 数据集
 input agecat
20
30
40
end
* 交叉合并两个数据集
cross using sex
list
* 输出结果如下所示:/*
      | agecat    sex |
  1.  |   20      male |
  2.  |   30      male |
  3.  |   40      male |
  4.  |   20    female |
  5.  |   30    female |
  6.  |   40    female |
*/
```

cross 命令的交叉合并功能可以用来生成面板数据,这在模拟中常常用到。

例 5-11 使用 cross 命令生成面板数据。

```
clear
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
set obs 25                    //设定 25 个 obs 观测数据
gen code = _n + 10010         //生成新变量,代表股票代码
save code, replace            //保存

clear
set obs 10                    //设定 10 个观测值
gen year = _n + 2010          //生成年份
cross using code.dta
sort code year                //对变量进行排序
sort year code                //注意变量位置不同,排序结果不同
save "平衡面板数据", replace
```

在经济地理相关的研究中,通常涉及不同地点的匹配与比较,cross 命令的交叉合并功能为实现这一目的提供了便利。

例 5-12　使用 cross 命令生成地理相对位置。

```
clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use "university.dta",clear
rename (学校名称 startlng0 startlat0) (学校名称 经度 纬度)   //重命名
save 地址 1.dta, replace

use "university.dta",clear
rename (学校名称 startlng0 startlat0) (schoolname startlng1 startlat1)
cross using "地址 1.dta"
sort schoolname 学校名称    //对变量进行排序
compress
list                        //陈列数据
```

2. joinby 命令

joinby 不仅能够实现 cross 命令的功能，而且能够进行组内交叉合并，其语法格式为

```
joinby [varlist] using filename [, options]
```

下面通过一个例子来阐述该命令的用法。

例 5-13　joinby 命令的应用。

```
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
* 使用 joinby 进行交叉合并
clear all
set obs 3                       //obs 观测数据
gen privon = "江苏省"           //生成新变量
gen city1 = "常州市" in 1       //生成新变量 city1,并将第一个观测值赋值为"常州市"
replace city1 = "南京市" in 2
replace city1 = "苏州市" in 3
save joinby1,replace

clear all
input str15 privon str15 city2
"江苏省" "徐州市"
"江苏省" "连云港市"
"江苏省" "淮安市"
end
save joinby2,replace

use joinby2,clear
joinby privon using joinby1   //按 privon 进行合并
sort city2 city1
* 使用 cross
use joinby2,clear
cross using joinby1           //cross 后面不接 varlist
sort city2 city1              //可以看出 joinby 与 cross 在这里实现了同样的功能
* 使用 joinby 实现组内合并
```

```
clear all
set obs 6
gen privon = "江苏省" in 1/3
replace privon = "浙江省" in 4/6
gen city1 = "常州市" in 1
replace city1 = "南京市" in 2
replace city1 = "苏州市" in 3
replace city1 = "杭州市" in 4
replace city1 = "宁波市" in 5
replace city1 = "温州市" in 6
save joinby11,replace

clear all
input str15 privon str15 city2
"江苏省" "徐州市"
"江苏省" "连云港市"
"江苏省" "淮安市"
"浙江省" "绍兴市"
"浙江省" "舟山市"
"浙江省" "嘉兴市"
end
save joinby22,replace

use joinby22,clear
joinby privon using joinby11
sort privon city2 city1
* 使用 cross,注意对比结果
use joinby22,clear
cross using joinby11   //cross 后面不接 varlist
sort city2 city1
* 匹配变量不同的情形
use joinby22,clear
joinby privon using joinby1 //当匹配变量不同时,自动寻找相同的进行匹配,并且只保留匹配成功
                            //的数据
```

joinby 命令的这一功能可以在金融领域的研究中用于删除停牌期间的某个事件。

例 5-14 应用 joinby 命令交叉合并数据。

```
clear
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use 停复牌, clear                                          //查看数据
use 事件列表,clear
joinby stkcd using 停复牌                                  //进行代码组内的交叉合并
gen date1 = date(date,"YMD")
format date1 %dCY-N-D
gen num = 1 if date1 >= startdate & date1 <= enddate       //日期判断
keep if num == 1
merge 1:1 stkcd date using 事件列表
keep if _m == 2
keep stkcd date
```

5.3　数据长宽转化

数据的组织方式(如面板数据)有两种形式,分别是长数据和宽数据,其组织方式分别如表 5-1 和表 5-2 所示。

表 5-1　长数据类型

code	year	var1	var2
1	2018	18.4	0.01
1	2019	17.5	0.03
1	2020	19.2	0.04
2	2018	35.5	0.15
2	2019	38.2	0.19
2	2020	40.7	0.21
…	…	…	…

表 5-2　宽数据类型

code	var12018	var12019	var12020	var22018	var22019	var22020	…
1	18.4	17.5	19.2	0.01	0.03	0.04	…
2	35.5	38.2	40.7	0.15	0.19	0.21	…
…	…	…	…	…	…	…	…

在数据处理过程中,往往需要根据分析目的的不同将数据以长数据或者宽数据的方式进行组织,因此涉及数据的长宽转化,即将长数据转换为宽数据或者将宽数据转换为长数据。

5.3.1　reshape 命令


视频讲解

reshape 命令为 Stata 的官方命令,可以实现数据的长宽转换。使用 reshape 将长数据变成为宽数据的命令一般为

```
reshape wide stubnames, i(varlist) j(varname) string
```

其中,stubnames 是要进行转换的主变量;i(varlist)表示作为观测值标识的变量(如表 5-1 中的 code);j(varname)在长数据变宽数据时(如表 5-1 中的 year)表示在长数据中存在但是在宽数据中不存在的变量。

例 5-15　长型变宽型 reshape wide。

```
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
clear
input code        year    var1    var2
1                 2018    18.4    0.01
1                 2019    17.5    0.03
1                 2020    19.2    0.04
2                 2018    35.5    0.15
2                 2019    38.2    0.19
2                 2020    40.7    0.21
```

```
end
reshape wide var1 var2,i(code) j(year)   //长型变宽型

*将"上市公司财务信息.dta"数据转为宽型数据
clear
cd "D:\Stata数据分析与建模\数据代码\第5章"
use 上市公司财务信息.dta,clear
keep id year finratio debt tang cash tagr tobin roa   //为便于观察,保留部分变量
reshape wide finratio debt tang cash tagr tobin roa, i(id) j(year)
```

从宽数据变成长数据的命令一般为

```
reshape long stubnames, i(varlist) j(varname) string
```

其中,stubnames 表示要进行转换的主变量;i(varlist)表示要作为观测值标识的变量,需要注意这个变量必须是唯一可识别的,不可以有重复值,否则会出错;j(varname)表示在长数据变为宽数据时在长数据中存在但是在宽数据中不存在的变量;在宽数据变长数据时,varname 表示在宽数据中不存在而在长数据中新生成的变量;string 用于声明 j(varname)中的 varname 是字符型。

宽型变长型,分为三种情况。

1. 需要生成 j(varname)的数字位于变量名最后

例 5-16 宽型变长型 reshape long-1。

```
clear all
cd "D:\Stata数据分析与建模\数据代码\第5章"
use hsprice.dta, clear
/*观察发现,这个数据明显是宽型数据,变量名格式为相同的变量名前缀 + 不同的后缀(hsprice2010, ...,hsprice2015),宽型数据一般都可以通过这种方式来判断。如果想把它变成长型数据,只需要判断 i、j、stub 分别是什么,之后直接套用命令就可以了。本例中,i 是标识变量 province,stub 是特征变量的相同前缀 hsprice,j 是待生成的标识变量,用于存放特征变量的后缀(2010, ..., 2015),可以起名为 year。套用 reshape 宽型数据转长型数据的公式,很容易写出转化命令*/
reshape long hsprice, i(province) j(year)
*长变宽
reshape wide hsprice, i(province) j(year)
```

2. 需要生成 j(varname)的数字位于变量名中间

例 5-17 宽型变长型 reshape long-2。

```
clear all
cd "D:\Stata数据分析与建模\数据代码\第5章"
import excel using "公司所有权属性.xlsx", clear first
drop in -2/-1                          //删除最后两行
rename 证券代码 stkcd                   //用中文名字也行
replace stkcd = substr(stkcd, 1, 6)  //提取 stkcd 前6位,从第1个字节开始,长度为6
destring stkcd, replace                //提取股票代码并转换为数值型
reshape long 公司属性交易日期, i(stkcd) j(year)   //一般的宽变长
```

```
nsplit year,d(4 4)                    //year 为数值型,nsplit 对数值型变量分列
drop year year2
rename year1 year
order stkcd year

clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
import excel using "公司所有权属性.xlsx", clear first
drop in -2/-1                         //删除最后两行
rename 证券代码 stkcd                  //用中文名字也行
replace stkcd = substr(stkcd, 1, 6)   //提取 stkcd 前 6 位,从第 1 个字节开始,长度为 6
destring stkcd, replace               //提取股票代码并转换为数值型
reshape long 公司属性交易日期@1231, i(stkcd) j(year) //使用@代表中间的符号,年份出现在变
                                                     //量名的中间
rename 公司属性交易日期 1231 公司属性
```

3. 需要生成 j(varname)的不是数字,而是字符

例 5-18　宽型变长型 reshape long-3。

```
webuse reshape4, clear
list
/*输出结果
     | id  kids   incm   incf |
  1. |  1     0   5000   5500 |
  2. |  2     1   2000   2200 |
  3. |  3     2   3000   2000 |
*/
reshape long inc, i(id) j(sex) string     //j(varname)中的 varname 是字符型变量
list, sepby(id)
/*输出结果如下
     |  id   sex   kids   inc |
  1. |  1     f      0   5500 |
  2. |  1     m      0   5000 |
  3. |  2     f      1   2200 |
  4. |  2     m      1   2000 |
  5. |  3     f      2   2000 |
  6. |  3     m      2   3000 |
*/
```

5.3.2　spread 命令与 gather 命令

spread 命令与 gather 命令为外部命令,相比于 reshape 命令其用法更简洁。spread 有伸展、延伸的意思,能够将长型数据转为宽型数据,gather 有聚合、聚集的意思,能够将宽型数据转为长型数据。gather 与 spread 很好地解决了变量名无规律时不能使用 reshape 进行长宽数据转化的问题,但 gather 和 spread 每次只能转为一个类型的变量。spread 的语法为

```
spread variable value, [label(varname)]
```

其中 label(varname)表示使用字符型变量 varname 为新变量构造变量标签。

例 5-19 长型变宽型 spread。

```
clear
input code    year    var1
1             2018    18.4
1             2019    17.5
1             2020    19.2
2             2018    35.5
2             2019    38.2
2             2020    40.7
end
spread year var1          //长型变宽型
*将"上市公司财务信息.dta"数据转为宽型数据
clear
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use 上市公司财务信息.dta,clear
keep id year finratio     //为便于观察,保留部分变量
spread  year finratio     //spread 只能单次转化一个变量
```

gather 接受变量 varlist 的列表,并按照变量名-变量取值(variable-value)的形式进行转换。这是转换为长数据的一个简单版本。gather 的语法格式为

```
gather varlist, [variable(newvar) value(newvar) label(newvar)]
```

其中,varlist 定义当前宽型数据中用于转换的单个或多个变量;variable(newvar)表示变量名转化后的新变量名,默认是“variable”;value(newvar)为变量取值生成的新变量名称,默认为“value”;label(newvar)用于生成一个新变量来存储 varlist 的变量标签。

例 5-20 宽型变长型 gather。

```
clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use hsprice.dta, clear
gather hsprice*              //当然也可以指定变量名,help gather
rename variable year         //重命名
rename  value hsprice        //重命名
replace year = subinstr(year, "hsprice", "", .)
destring year, replace
*长变宽
spread year hsprice
```

下面通过一个实际的案例,对长宽变换中涉及的更为复杂的问题进行综合训练。

例 5-21 长宽变换综合应用。

```
clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
import excel  汽车制造业.xlsx, first   //可以看出是宽型数据
*用 reshape long 转为长型
split id, p(".")                      //对 id 变量提取代码
```

```
drop id id2
destring id1, replace
order id name
duplicates drop id, force
drop in 103                              //删除最后一行的缺失值
drop yingyeshouru2015 zonggushu2015      //只有两个变量有 2015 年的数据,考虑删除
* 用 reshape long 转为长型数据
reshape long zongzichan zongfuzhai jinglirun yingyeshouru liudongzichan liudongfuzhai
yanfatouru dagudongchigushuliang zonggushu, i(id) j(year)
* 用 reshape wide 转为宽型数据
reshape wide zongzichan zongfuzhai jinglirun yingyeshouru liudongzichan liudongfuzhai
yanfatouru dagudongchigushuliang zonggushu, i(id) j(year)

clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
import excel  汽车制造业.xlsx, first  //可以看出是宽型数据
* 用 reshape long 转为长型
split id, p(".")                         //对 id 变量提取代码
drop id id2
destring id1, replace
order id name
duplicates drop id, force
drop in 103                              //删除最后一行的缺失值
drop yingyeshouru2015 zonggushu2015      //只有两个变量有 2015 年的数据,考虑删除
* 用 gather 转为长型数据
keep id name  zongzichan *
gather zongzichan *                      //只能转换一个类型的变量
* 用 spread 转为宽型数据
spread variable value                    //只能转换一个类型的变量
```

长宽数据转化还有另外一个增强的外部命令 sreshape,读者可以自行了解。长宽转换的用处很多,一般而言 Stata 的操作模式为按列操作,在实际工作中,有时候需要进行按行操作,这时可以考虑利用 gather 先把行转为列,然后利用分组功能按列操作,最后再利用 spread 转回来。

例 5-22　行转为列的应用。

```
* 例如,计算每一行的最大值,并给出这个最大值所对应的变量
clear
set obs 100
gen id = _n
gen x1 = uniform()
gen x2 = uniform()
gen x3 = uniform()
gen x4 = uniform()
gather x *
bysort id: egen max = max(value)
gen x = var if max == value
gsort id var
```

```
bysort id: carryforward x, replace
gsort id -var
bysort id: carryforward x, replace
spread variable value
order id x1 x2 x3 x4
```

5.4 数据转置

数据转置是指交换数据的行与列,类似于矩阵转置。由于 Stata 中大部分的命令和函数是针对变量(列为单位)进行操作的,而针对观测值(行为单位)的操作并不多,通过对数据的转置可以利用功能丰富的 Stata 命令实现对行的操作。

5.4.1 数值型变量转置

对数值型变量进行转置的命令是 xpose,其语法格式为

```
xpose, clear [options]
```

选项 options 主要包括 clear(清空原始数据)、format(%fmt,设定数据格式)、varname(使用_varname 保留原始数据中的变量名)等。

例 5-23 xpose 命令的应用。

```
clear
webuse xposexmpl
list
xpose, clear  //转置后变为四行三列数据。clear 这个选项是必需的,是为了提醒转置之后原始数
              //据将不存在
list
osexmpl, clear
xpose, clear varname  //保留原始数据中的变量名称,可以加上 varname 选项
list
xpose, clear          //转置两次之后,又得到了原始数据(第一次转置需要加上 varname)
list
xpose, clear varname format( %6.2f)  //format( %fmt)选项设定数据的显示格式
list
```

5.4.2 字符型变量转置

对字符型变量转置使用的命令是 sxpose,该命令为外部命令,可以使用 ssc install sxpose 进行安装。该命令的语法格式为

```
sxpose, clear [force format(format) firstnames destring]
```

例 5-24 sxpose 命令的应用。

```
clear
sysuse auto
keep make price weight foreign  //保留变量
keep in 1/10          //保留 1～10 行的数据
list
sxpose, clear force  //将含有字符型变量的数据进行转置,由于存在数值型变量,需要加上选项 force
```

5.5　变列操作

变列操作是指将一列数据拆分和转换为多列数据或者将多列数据合并和转换为一列数据。变列操作需要根据变量类型和形式的不同选择使用不同的命令。

5.5.1　一列变多列


视频讲解

1. 利用 reshape 进行变换

长型数据转换为宽型数据,实际上就是一列变多列,因此在特定情形下前面介绍的 reshape 命令可以实现一列变多列的操作。

例 5-25　使用 reshape 变换。

```
clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use zazhixinxi.dta, clear
gen id = int((_n - 1)/3) + 1     //生成对应的 i(varname),int()代表取整
egen year = seq(), from(1) to(3)  //生成对应的 j(varname),seq()为顺序函数,重复的数字 1～3
reshape wide v, i(id) j(year)      //将长型数据转换为宽型数据,实际上就是一列变多列
compress                           //自动压缩至最小的、合适的长度
rename (v1 v2 v3) (name  host code)
```

2. 使用 split 与 nsplit 变量进行变换

在导入原始数据后,有很多变量都是用连接符连在一起的,如“时间”变量由“年”“月”“日”变量组成。如果想对其中的变量分别进行处理,或者只需要其中的某个变量,就要进行变量拆分工作,包括对字符变量的拆分和数值变量的拆分,其分别涉及 split 命令以及 nsplit 命令。

spilt 命令用于拆分字符串变量,其语法结构为

```
split strvar [if] [in] [, options]
```

其中 strvar 表示字符型变量,split 的选项较多,下面分别介绍。

generate(stub)表示新变量名开头为 stub,默认是原来的变量名 strvar,简写作 gen(stub)。

parse(parse_strings)指定分列的字符,默认是空格。

limit(#)表示最多创建的变量个数。

notrim 表示不删除原始变量最前面和最后面的空格。

destring 表示将 destring 应用于新的字符串变量，尽可能用数字变量替换初始字符串变量。使用 destring 这一选项，可以将生成的变量转化为数值型变量。在使用 destring 后，则可以使用剩下的 4 个 options：

ignore("chars")用于删除指定的非数字字符。

force 表示将非数字字符串转换为缺失值。

float 指定新生成的数值变量为浮点型(float)。

percent 表示将百分比变量转换为分数形式。

例 5-26 字符型变量拆分。

```
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use splitvar.dta,clear
split Date, p( - )                    //会将 Date 拆成三个变量 Date1、Date2、Date3
 * 也可以使用 destring Date, replace ignore(" - "),注意区别
destring Date1 Date2 Date3, replace//变量类型转化
list Date Date1 Date2 Date3 in 1/5
```

使用 split 命令可以按照一定的形式将字符型变量进行拆分，以实现换列的功能。

例 5-27 使用 split 命令变换。

```
clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use zazhixinxi.dta, clear
replace v = v + ";" + v[_n + 1] + ";" + v[_n + 2]  //让变量 v 加上下一行和下两行的内容,
                                                    //用分号隔开
keep if mod(_n, 3) == 1           //保留行号除以 3 余数为 1 的观测值
split v, p(";")                   //使用分号作为分隔符进行拆分
drop v                            //删除数据
rename * (name host code)

use 上市公司基本信息数据.dta, clear
drop in 1/2
keep ListedCoID Symbol  EndDate   //为便于观察,只保留部分变量
split EndDate, parse(" - ")

sysuse auto,clear
keep make price rep78 foreign     //保留 keep 后面的变量
split make, parse(" ")            //利用空格,将 make 变量下的内容分离
```

除了字符型变量，还会经常遇到需要拆分数值型变量的情况，数值型变量一般不会有分隔符进行分割，拆分思想是按照数字个数来拆分，使用的命令为 nsplit，其语法结构为

```
nsplit [varname] [if exp] [in range] , digits(digit pattern in existing variable) [generate
(newvarlist or stub) ]
```

其中，digits 选项是一个必选项，用于规定拆分的模式，如将七位数拆分为 2、2、3，则需要写成 digits(2 2 3)；如果需要拆分的位数相同，如需将六位数拆分为三个两位数，则只需

写成 digits(2)即可；generate()选项用于生成新变量的变量名。

例 5-28　使用 nsplit 命令变换。

```
clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use 上市公司基本信息数据.dta, clear
drop in 1/2
keep ListedCoID Symbol ShortName EndDate IndustryCode Zipcode
destring, replace                          //观察后发现,Zipcode 没有成功转化
compress
sort Zipcode                               //排序后观察,查找原因
drop in 1/25
split Zipcode, p(",")                      //观察数据,发现有些行存在以逗号分隔的两个编码
drop Zipcode2  Zipcode3                    //删除其中一个
destring Zipcode1, replace force
sort Zipcode1                              //排序后查看数据观测值
format Zipcode1 %06.0f
drop if Zipcode1 == .
drop  in -4/-1
nsplit  Zipcode1, d(2 2 2)
format Zipcode11 Zipcode12 Zipcode13 %02.0f//调整格式

clear all
input code str20
130102                                     //河北省石家庄市长安区
130103                                     //河北省石家庄市桥东区
130203                                     //河北省唐山市路北区
140106                                     //山西省太原市迎泽区
220203                                     //吉林省吉林市龙潭区
331082                                     //浙江省台州市临海市
371521                                     //山东省聊城市阳谷县
410105                                     //河南省郑州市金水区
420111                                     //湖北省武汉市洪山区
end
nsplit code, digits(2) gen(a)              //拆分变量
rename a* (provincecode citycode countycode)   //对拆分完的变量进行重命名
format provincecode citycode countycode %02.0f
list code provincecode citycode countycode in 1/9
```

3. 利用观测值所在行数除余数功能进行拆分

例 5-29　利用观测值所在行数除余数功能进行拆分。

```
clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use zazhixinxi.dta, clear
gen v1 = v[_n+1]
gen v2 = v[_n+2]
** 这两条命令可以用如下循环表示(关于循环,在后面章节介绍)
/*forvalues i = 1/2 {
```

```
        gen v'i' = v[_n + 'i']
} * /
gen num = mod(_n, 3)
keep if num == 1
compress
rename * (name host code)
drop num
```

4. varsplit

varsplit 为外部命令,该命令并没有上传到 ssc 上,无法通过 ssc install 进行安装。这个命令上传到了爬虫俱乐部的 github 主页上,可以通过 github install 进行安装。该命令的语法为

```
varsplit number  //表示直接将变量拆分为多少个(number 为具体的数字)
net install github, from("https://haghish.github.io/github/")  //安装 github 命令
github install Stata-Club/varsplit  //使用 github install 安装 varsplit 命令
```

例 5-30 使用 varsplit 命令。

```
clear
input str5 var
   "Sarah"
   "19"
   "89"
   "90"
   "Sam"
   "21"
   "92"
   "89"
end
varsplit 4
rename (var1 - var4) (name age course1 course2)
clear all
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use zazhixinxi.dta, clear
varsplit 3  //varsplit 命令没有任何选项,直接在命令后面跟上想要拆成的变量个数
compress
rename * (name host code)
```


视频讲解

5.5.2 多列变一列

1. 使用 reshape 命令

使用 reshaple 命令的长宽转换功能将数据多列变一列。

例 5-31 使用 reshape 命令。

```
clear
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use 公司名称.dta, clear
```

```
gen id = _n                          //生成对应的 i(varname)
reshape long v, i(i) j(year)         //转换为长型数据,实际上就是多列变一列
keep v                               //保留变量
drop if v == " "                     //删除数据
```

2. 使用 expand 命令

expand 命令是 Stata 中用于复制数据集中观测值的命令。它可以将数据集中的每个观测值复制多次,每次复制的份数由表达式结果决定。如果表达式结果小于 1 或者为缺失,则复制次数为 1,即不进行原数据的复制。expand 命令可以帮助用户在不改变数据集中变量个数的前提下,增加数据集中的观测值数量。对于一些需要重复观测或多次记录数据的情况,这个命令非常实用。需要注意的是,在使用 expand 命令时,用户需要指定一个表达式,用于确定每个观测值的复制次数。这个表达式可以是任何合法的数学表达式,包括算术运算、比较运算、逻辑运算等。

例 5-32　使用 expand 命令。

```
clear
set obs 5                            //设置 5 个观测值
gen company = _n                     //生成公司序号变量 company(1 - 5)
expand 10                            //将各个观测值扩大,每个观测值变为同样的 10 个
sort company
bys company: gen year = _n + 1950 //生成年份变量 year(1951 - 2050)

clear
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use 公司名称.dta, clear
expand 3  //将三个变量放在一个变量下面,在没有缺失值的情况下,观测值数量应当为之前的 3
          //倍。使用 expand 命令对观测值数量进行扩充
sort v1
gen num = mod(_n,3)
bysort v1: replace v3 = v1 if num == 1 //通过 v1 变量进行分组,将每组第 1 个观测值的 v3 替换为 v1
bysort v1: replace v3 = v2 if num == 2 //通过 v1 变量进行分组,将每组第 2 个观测值的 v3 替换为 v2
keep v3                              //v3 就是三个变量合并后的变量
drop if v3 == ""                     //删除数据
rename v3 univ                       //重命名
```

3. 使用 stack 命令

stack 命令是 Stata 中用于将不同数据集中的观测值堆叠在一起的命令。它可以将不同数据集中的观测值按照某个变量(或多个变量)进行匹配,并将匹配到的观测值纵向合并起来,形成一个更全面的数据集。

例 5-33　使用 stack 命令。

```
clear
cd "D:\Stata 数据分析与建模\数据代码\第 5 章"
use 公司名称.dta, clear
```

```
stack v1 - v3, into(univ) clear  //通过 stack 命令将变量 v1 - v3 堆在变量 univ 下
keep univ                        //保留变量
drop if univ == ""               //删除数据
```

习题

1. 数据重构的含义是什么？
2. 数据拆分包括哪些形式？
3. 请描述 merge 命令的含义。
4. 请描述 matchit 命令的含义。
5. 两个数据集匹配的必需命令选项有哪些？
6. reclink 命令的语法是什么？
7. 使用 reshape 将长数据变成为宽数据的命令是什么？
8. spread 命令与 gather 命令的含义什么？
9. 数据转置的含义是什么？
10. 字符型变量转置的命令是什么？