Operaciones, extración y otras funcionalidades entre tipos de estructuras de datos

Santiago Lozano

21 de febrero de 2020

Función stack()

Esta función concatena un simple vactor los valores de ciertas columnas de un data.frame. Esta función despliega un data.frame, con los vectores apilados en la primera columna y la segunda columna contiene un factor que indica la columna de origen. La función unstack() realiza la operación contraria

```
z <- data.frame(trt1=c(1,6,3,5),trt2=c(8,8,3,1))
z
```

```
##   trt1 trt2
## 1    1    8
## 2    6    8
## 3    3    3
## 4    5    1
```

Función stack()

```
stack(z)
```

```
##   values  ind
## 1      1 trt1
## 2      6 trt1
## 3      3 trt1
## 4      5 trt1
## 5      8 trt2
## 6      8 trt2
## 7      3 trt2
## 8      1 trt2
```

Función transform

Esta función realiza transformaciones sobre las columnas de una data.frame

```
Z <- data.frame(Peso=c(80,75,60,52),
                Altura=c(180,170,165,150),
                Colesterol=c(44,12,23,34),
                Genero=c("M","M","F","F"))
Z
```

```
##   Peso Altura Colesterol Genero
## 1   80    180          44      M
## 2   75    170          12      M
## 3   60    165          23      F
## 4   52    150          34      F
```

Función transform

```
Z <- transform(Z,Altura=Altura/100,BMI=Peso/(Peso/100)^2)
Z
```

```
##   Peso Altura Colesterol Genero      BMI
## 1   80   1.80          44      M 125.0000
## 2   75   1.70          12      M 133.3333
## 3   60   1.65          23      F 166.6667
## 4   52   1.50          34      F 192.3077
```

Función tapply

Calcula distintas operaciónes realizadas en masas, mediante combinaciones de variables cuantitativas con variables categóricas

```
data<-read.table("C:/Users/santiago/Documents/
        Progrmación en R/2020-I/PR04
        Manipulación de datos II/temperatures.txt"
         ,header=T)
attach(data)
names(data)
```

```
## [1] "temperature" "lower"  "rain"  "month" "yr"
```

Función tapply

```
tapply(temperature,month,mean)
```

```
##         1         2         3         4         5
##  7.930051  8.671136 11.200508 13.813708 17.880847
##         6         7         8         9        10
## 20.306151 22.673854 23.104924 19.344211 15.125976
##        11        12
## 10.720702  8.299830
```

Función tapply

```
tapply(temperature,month,min)
```

```
##    1    2   3   4   5    6    7    8   9   10  11  12
## -6.8 -3.5 1.5 2.8 8.8 11.5 14.3 15.0 7.5 8.3 0.5 -1.8
```

Función tapply

```
tapply(temperature,month,function(x)
  sqrt(var(x)/length(x)))
```

```
##         1         2         3         4         5
## 0.1401489 0.1414445 0.1358934 0.1476242 0.1673197
##         6         7         8         9        10
## 0.1596439 0.1539661 0.1516091 0.1309294 0.1155612
##        11        12
## 0.1291703 0.1398438
```

Función tapply

mediante esta función podemos producir tablas multidimensionales simplemente reemplzando una variable categórica por una lista de variables categóricas

```
tapply(temperature,list(yr,month),mean)[,1:6]
```

Función tapply

	1	2	3	4	5	6
1987	3.170	6.871	8.132	14.92	15.60	17.73
1988	8.048	8.248	9.959	12.74	17.31	18.71
1989	8.841	9.482	11.919	11.03	20.43	21.23
1990	9.445	11.021	12.487	13.80	20.19	18.57
1991	6.980	4.817	12.022	13.13	15.58	16.88
1992	6.964	8.686	11.477	13.30	20.45	22.21
1993	10.115	6.984	11.207	14.10	17.75	21.10
1994	8.825	7.217	11.806	12.67	16.23	20.86
1995	8.309	10.436	10.662	14.77	18.73	19.93
1996	7.019	6.065	8.4870	13.96	14.31	21.96
1997	4.932	10.171	13.378	15.07	18.19	19.90
1998	8.759	11.247	11.715	12.53	19.46	19.30
1999	9.523	8.485	11.790	14.60	18.94	20.00
2000	8.229	10.328	11.900	12.50	18.21	20.63
2001	7.067	9.121	9.012912.66		18.96	20.52

Función tapply

para corregir el inconveniente de trabajar con los missing data usamos el argumento na.rm=TRUE

```
tapply(temperature,yr,mean,na.rm=TRUE)
```

```
##      1987      1988      1989      1990      1991      1992
## 13.27014 13.79126 15.54986 15.62986 14.11945 14.61612
##      1993      1994      1995      1996      1997      1998
## 14.30984 15.12877 15.81260 13.98082 15.63918 15.02568
##      1999      2000      2001      2002      2003      2004
## 15.63736 14.94071 14.90849 15.47589 16.03260 15.25109
##      2005
## 15.06000
```

Función tapply

Usted puede querer eliminar ciertos valores extremos antes de calcular la media (pues la media aritmética es bastante sensible a datos atípicos), para ello, el argumento trim especifica la fracción de los datos (entre 0 y 0.5 que usted quiere omitir). Los valores extremos son omitidos en prioridad

Función tapply

```
tapply(temperature,yr,mean,trim=0.2)
```

```
##     1987     1988     1989     1990     1991     1992
## 13.45068 13.74500 14.99726 15.16301 13.92237 14.32091
##     1993     1994     1995     1996     1997     1998
## 14.28000 14.64658 15.25571 13.75845 15.54064 14.91500
##     1999     2000     2001     2002     2003     2004
## 15.44364 14.59318 14.63333 15.33927 15.70959 15.04136
##     2005
## 15.02009
```

Función aggregate()

La función aggregate() genera un data.frame en sub poblaciones de acuerdo a un factor (especificado por el argumento by) y aplica una función dada para cada subpoblación

```
Z <- data.frame(Peso=c(80,75,60,52),
                Altura=c(180,170,165,150),
                Colesterol=c(44,12,23,34),
                Genero=c("M","M","F","F"))
Z
```

```
##   Peso Altura Colesterol Genero
## 1   80    180          44      M
## 2   75    170          12      M
## 3   60    165          23      F
## 4   52    150          34      F
```

Función aggregate

```
aggregate(Z[,-4],by=list(Gender=Z[,4]),FUN=mean)
```

```
##   Gender Peso Altura Colesterol
## 1      F 56.0  157.5        28.5
## 2      M 77.5  175.0        28.0
```

Función aggregate()

Suponga que tenemos dos variables respuesta (y y z) y dos variables explicativas (x y w) que pueden ser usadas para realizar un resumen estadístico. la función aggregate() funciona de distintas maneras

- one to one: aggregate(y $\sim$ x, mean)
- one to many: aggregate(y $\sim$ x + w, mean)
- many to one: aggregate(cbind(y,z) $\sim$ x, mean)
- many to many: aggregate(cbind(y,z) $\sim$ x + w, mean)

Función aggregate()

```
data2<-read.table("C:/Users/santiago/Documents/
        "Progrmación en R/2020-I/PR04
        Manipulación de datos II/pHDaphnia.txt",header=T)
names(data2)
```

```
## [1] "Growth.rate" "Water" "Detergent" "Daphnia"  "pH"
```

```
aggregate(Growth.rate~Water,data2,mean)
```

```
##   Water Growth.rate
## 1  Tyne    3.685862
## 2  Wear    4.017948
```

Función aggregate()

```
aggregate(Growth.rate~Water+Detergent,data2,mean)
```

```
##   Water Detergent Growth.rate
## 1  Tyne    BrandA    3.661807
## 2  Wear    BrandA    4.107857
## 3  Tyne    BrandB    3.911116
## 4  Wear    BrandB    4.108972
## 5  Tyne    BrandC    3.814321
## 6  Wear    BrandC    4.094704
## 7  Tyne    BrandD    3.356203
## 8  Wear    BrandD    3.760259
```

Función aggregate()

```
aggregate(cbind(pH,Growth.rate)~Water+Detergent,
          data2,mean)
```

```
##   Water Detergent       pH Growth.rate
## 1  Tyne    BrandA 4.883908    3.661807
## 2  Wear    BrandA 5.054835    4.107857
## 3  Tyne    BrandB 5.043797    3.911116
## 4  Wear    BrandB 4.892346    4.108972
## 5  Tyne    BrandC 4.847069    3.814321
## 6  Wear    BrandC 4.912128    4.094704
## 7  Tyne    BrandD 4.809144    3.356203
## 8  Wear    BrandD 5.097039    3.760259
```