Aquisição de Arquivos e Datasets

Criando Novas Variáveis

Delermando Branquinho Filho

Por que criar novas variáveis?

- Muitas vezes, os dados brutos não terão um valor que você está procurando
- Você precisará transformar os dados para obter os valores que você gostaria
- Normalmente, você adicionará esses valores aos quadros de dados com os quais você está trabalhando
- Variáveis comuns para criar
- Indicadores de ausência
- "Cortar para cima" variáveis quantitativas
- Aplicando transformações

Exemplo data set

https://data.baltimorecity.gov/Community/Restaurants/k5ry-ef3g

---

Obtendo os dados da web

```
if(!file.exists("./data")){dir.create("./data")}
fileUrl <- "https://data.baltimorecity.gov/api/views/k5ry-ef3g/rows.csv?accessType=DOWNLOAD"
download.file(fileUrl,destfile="./data/restaurants.csv",method="curl")
```

```
## Warning: execução do comando 'curl "https://data.baltimorecity.gov/api/
## views/k5ry-ef3g/rows.csv?accessType=DOWNLOAD" -o "./data/restaurants.csv"'
## teve status 127
```

```
## Warning in download.file(fileUrl, destfile = "./data/restaurants.csv",
## method = "curl"): download had nonzero exit status
```

```
restData <- read.csv("./data/restaurants.csv")
```

Criando sequências

Às vezes você precisa de um índice para seu conjunto de dados

```
s1 <- seq(1,10,by=2) ; s1
```

```
## [1] 1 3 5 7 9
```

```
s2 <- seq(1,10,length=3); s2
```

```
## [1]  1.0  5.5 10.0
```

```r
x <- c(1,3,8,25,100); seq(along = x)
```

```
## [1] 1 2 3 4 5
```

Subsetting variáveis

```r
restData$nearMe = restData$neighborhood %in% c("Roland Park", "Homeland")
table(restData$nearMe)
```

```
##
## FALSE  TRUE
##  1314    13
```

Criando variáveis binárias

```r
restData$zipWrong = ifelse(restData$zipCode < 0, TRUE, FALSE)
table(restData$zipWrong,restData$zipCode < 0)
```

```
##
##         FALSE TRUE
##   FALSE  1326    0
##   TRUE      0    1
```

Criando variáveis categóricas

```r
restData$zipGroups = cut(restData$zipCode,breaks=quantile(restData$zipCode))
table(restData$zipGroups)
```

```
##
## (-2.123e+04,2.12e+04]  (2.12e+04,2.122e+04] (2.122e+04,2.123e+04]
##                   337                   375                   282
## (2.123e+04,2.129e+04]
##                   332
```

```r
table(restData$zipGroups,restData$zipCode)
```

```
##
##                         -21226 21201 21202 21205 21206 21207 21208 21209
##   (-2.123e+04,2.12e+04]      0   136   201     0     0     0     0     0
##   (2.12e+04,2.122e+04]       0     0     0    27    30     4     1     8
##   (2.122e+04,2.123e+04]      0     0     0     0     0     0     0     0
##   (2.123e+04,2.129e+04]      0     0     0     0     0     0     0     0
##
##                         21210 21211 21212 21213 21214 21215 21216 21217
##   (-2.123e+04,2.12e+04]     0     0     0     0     0     0     0     0
##   (2.12e+04,2.122e+04]     23    41    28    31    17    54    10    32
##   (2.122e+04,2.123e+04]     0     0     0     0     0     0     0     0
```

```
##   (2.123e+04,2.129e+04]     0     0     0     0     0     0     0     0
##
##                         21218 21220 21222 21223 21224 21225 21226 21227
##   (-2.123e+04,2.12e+04]     0     0     0     0     0     0     0     0
##   (2.12e+04,2.122e+04]     69     0     0     0     0     0     0     0
##   (2.122e+04,2.123e+04]     0     1     7    56   199    19     0     0
##   (2.123e+04,2.129e+04]     0     0     0     0     0     0    18     4
##
##                         21229 21230 21231 21234 21237 21239 21251 21287
##   (-2.123e+04,2.12e+04]     0     0     0     0     0     0     0     0
##   (2.12e+04,2.122e+04]      0     0     0     0     0     0     0     0
##   (2.122e+04,2.123e+04]     0     0     0     0     0     0     0     0
##   (2.123e+04,2.129e+04]    13   156   127     7     1     3     2     1
```

Corte mais fácil

```
library(Hmisc)
```

```
## Loading required package: lattice

## Loading required package: survival

## Loading required package: Formula

## Loading required package: ggplot2

##
## Attaching package: 'Hmisc'

## The following objects are masked from 'package:base':
##
##     format.pval, round.POSIXt, trunc.POSIXt, units
```

```
restData$zipGroups = cut2(restData$zipCode,g=4)
table(restData$zipGroups)
```

```
##
## [-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]
##            338            375            300            314
```

Criando variáveis de fator

```
restData$zcf <- factor(restData$zipCode)
restData$zcf[1:10]
```

```
##  [1] 21206 21231 21224 21211 21223 21218 21205 21211 21205 21231
## 32 Levels: -21226 21201 21202 21205 21206 21207 21208 21209 ... 21287
```

```
class(restData$zcf)
```

```
## [1] "factor"
```

Níveis de variáveis fatoriais

```
yesno <- sample(c("yes","no"),size=10,replace=TRUE)
yesnofac = factor(yesno,levels=c("yes","no"))
relevel(yesnofac,ref="no")
```

```
##  [1] yes yes yes yes yes no  no  no  no  yes
## Levels: no yes
```

```
as.numeric(yesnofac)
```

```
##  [1] 1 1 1 1 1 2 2 2 2 1
```

O corte produz variáveis fatoriais

```
library(Hmisc)
restData$zipGroups = cut2(restData$zipCode,g=4)
table(restData$zipGroups)
```

```
##
## [-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]
##            338            375            300            314
```

Usando a função mutate

```
library(Hmisc); library(plyr)
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:Hmisc':
##
##     is.discrete, summarize
```

```
restData2 = mutate(restData,zipGroups=cut2(zipCode,g=4))
table(restData2$zipGroups)
```

```
##
## [-21226,21205) [ 21205,21220) [ 21220,21227) [ 21227,21287]
##            338            375            300            314
```

Transformações comuns

- `abs(x)` valor absoluto
- `sqrt(x)` raiz quadrada
- `ceiling(x)` «Limite máximo (x)» (3,475) é de 4
- `floor(x)` floor(3.475) é 3
- `round(x,digits=n)` round(3.475,digits=2) é 3.48

- `signif(x,digits=n)` signif(3.475,digits=2) é 3.5
- `cos(x), sin(x)` etc.
- `log(x)` Logaritmo natural
- `log2(x)`, `log10(x)` Outros registros comuns
- `exp(x)` exponencial x