Programação 2017/2018
Mestrado em Informática Médica / Curso de Especialização em Informática na Saúde
Nuno Moniz DCC - FCUP

Folha 1: Conceitos Básicos do R - Soluções

Vetores e Funções

1. O R possui vários conjuntos de dados que vêm incluidos em diferentes *packages*. Um desses *packages* é o `datasets` que oferece o conjunto de dados `women`.

(a) Comece por carregar a *package* `datasets` com o comando: `library(datasets)`.

```
library(datasets)
```

(b) Veja o conteúdo do conjunto de dados `women`, escrevendo `women` na consola.

```
women
```

```
##    height weight
## 1      58    115
## 2      59    117
## 3      60    120
## 4      61    123
## 5      62    126
## 6      63    129
## 7      64    132
## 8      65    135
## 9      66    139
## 10     67    142
## 11     68    146
## 12     69    150
## 13     70    154
## 14     71    159
## 15     72    164
```

(c) Para saber a que se refere este conjunto dados, procure `women` na caixa de ajuda do RStudio. Em alternativa, pode executar o comando `help(women)` na consola.

(d) Crie um vetor com a informação dos pesos executando o comando `pesos <- women[,2]`.

```
pesos <- women[,2]
```

(e) Sobre o vetor `pesos` obtenha:

- a média;

```
mean(pesos)
```

```
## [1] 136.7333
```

- os pesos superiores $130lb$;

```
pesos[pesos > 130]

## [1] 132 135 139 142 146 150 154 159 164
```

- os pesos superiores à média;

```
pesos[pesos > mean(pesos)]

## [1] 139 142 146 150 154 159 164
```

- o número de mulheres com peso superior à média;

```
length(pesos[pesos > mean(pesos)])

## [1] 7
```

(f) Altere o vetor `pesos` de forma a que os pesos inferiores a $130lb$ seja reduzido em $5lb$.

```
pesos[pesos < 130] <- pesos[pesos < 130] - 5
pesos


##  [1] 110 112 115 118 121 124 132 135 139 142 146 150 154 159 164
```

(g) Qual o resultado do comando `range(pesos)` ?

```
range(pesos)


## [1] 110 164
```

(h) Sabendo que $1lb = 0.453592kg$, escreva uma função `lb2kg` que converta um peso em lb para kg.

```
lb2kg <- function(p) p*0.453592
```

(i) Teste a sua função com os seguintes comandos: `lb2kg(100)` e `lb2kg("100")`. O que acontece?

```
lb2kg(100)


## [1] 45.3592


lb2kg("100")


## Error in p * 0.453592:  non-numeric argument to binary operator
```

(j) Aplique a sua função ao vetor `pesos`.

```
lb2kg(pesos)


##  [1] 49.89512 50.80230 52.16308 53.52386 54.88463 56.24541 59.87414
##  [8] 61.23492 63.04929 64.41006 66.22443 68.03880 69.85317 72.12113
## [15] 74.38909
```

(k) Altere a sua função para que o peso em kg seja arredondado a um valor inteiro.

```
lb2kg <- function(p) round(p*0.453592)
lb2kg(pesos)


##  [1] 50 51 52 54 55 56 60 61 63 64 66 68 70 72 74
```

Data Frames

2. Aceda ao conteúdo do conjunto de dados `Pima.tr` disponível na *package* `MASS`

```
library(MASS)
```

Sobre estes dados responda às seguintes questões.

(a) Que informação está representada neste conjunto de dados?

```
help(Pima.tr)
```

(b) Quais são as mulheres identificadas como diabéticas?

```
subset(Pima.tr,type == "Yes")
```

(c) Qual é a percentagem de mulheres diabéticas? (Sugestão: use a função `nrow()`.)

```
nrow(subset(Pima.tr,type == "Yes"))/nrow(Pima.tr)


## [1] 0.34
```

(d) Quais são as mulheres diabéticas e com idade inferior a 40?

```
subset(Pima.tr,type == "Yes" & age < 40)
```

(e) Qual é a concentração de glucose e o IMC (`bmi`) das mulheres com um IMC normal, isto é no intervalo $]19.1, 25.8]$?

```
subset(Pima.tr,bmi > 19.1 & bmi <= 25.8,c(glu,bmi))
```

(f) Qual é a concentração de glucose e o IMC (`bmi`) das mulheres com um IMC abaixo ou acima do peso normal?

```
subset(Pima.tr,!(bmi > 19.1 & bmi <= 25.8),c(glu,bmi))
```

(g) Crie uma função que, dada uma idade, devolve o conjunto de mulheres cuja idade é superior à indicada. Por omissão, a idade dada deve ser 40.

```
procIdade <- function(i=40) {
  subset(Pima.tr,age > i)
}
procIdade(60)


##     npreg glu bp skin  bmi   ped age type
## 9       3 142 80   15 32.4 0.200  63   No
## 80     12 121 78   17 26.5 0.259  62   No
## 157     2 197 70   99 34.7 0.575  62  Yes
```

Funções e Manipulação Básica de Dados

3. Considerando o *dataset* `women` da pergunta 1, Crie um ficheiro em R com o nome `'IMC.R'` com a instrução `library(datasets)`. Adicione ao ficheiro o conteúdo solicitado abaixo. Sempre que precise de executar o ficheiro deve escrever na consola do R `source('IMC.R')` ou então carregar no botão `Source` e `Run`.

(a) Crie uma função `lb2kg` que dado um peso em *lb* devolve a peso em *kg* arredondado a um inteiro ($1lb = 0.453592kg$).

```r
lb2kg <- function(p) round(p*0.453592)
```

(b) Crie uma função `in2m` que dado uma altura em *inches* devolve a altura em *metros* arredondada a 2 casas decimais ($1in = 0.0254m$).

```r
in2m <- function(a) round(a*0.0254,2)
```

(c) A partir do *dataset* `women` e das duas funções acima, crie uma nova matriz `m` com a mesma informação contida em `women` mas com métricas diferentes.

```r
m <- matrix(c(in2m(women[,1]),
              lb2kg(women[,2])),ncol=2)
m
```

```
##       [,1] [,2]
##  [1,] 1.47   52
##  [2,] 1.50   53
##  [3,] 1.52   54
##  [4,] 1.55   56
##  [5,] 1.57   57
##  [6,] 1.60   59
##  [7,] 1.63   60
##  [8,] 1.65   61
##  [9,] 1.68   63
## [10,] 1.70   64
## [11,] 1.73   66
## [12,] 1.75   68
## [13,] 1.78   70
## [14,] 1.80   72
## [15,] 1.83   74
```

(d) Calcule o valor do IMC ($peso/altura^2$), arredondado a uma casa decimal, para cada uma das linhas da matriz.

```r
round(m[,2]/m[,1]^2,1)
```

```
##  [1] 24.1 23.6 23.4 23.3 23.1 23.0 22.6 22.4 22.3 22.1 22.1 22.2 22.1 22.2
## [15] 22.1
```

(e) Utilize a função `cbind()` para criar uma nova matriz `m1` que resulta de anexar à matriz `m` uma 3a coluna com o cálculo anterior.

```r
m1 <- cbind(m,round(m[,2]/m[,1]^2,1))
m1
```

```
##       [,1] [,2] [,3]
##  [1,] 1.47   52 24.1
##  [2,] 1.50   53 23.6
##  [3,] 1.52   54 23.4
##  [4,] 1.55   56 23.3
##  [5,] 1.57   57 23.1
##  [6,] 1.60   59 23.0
##  [7,] 1.63   60 22.6
##  [8,] 1.65   61 22.4
##  [9,] 1.68   63 22.3
## [10,] 1.70   64 22.1
## [11,] 1.73   66 22.1
## [12,] 1.75   68 22.2
## [13,] 1.78   70 22.1
## [14,] 1.80   72 22.2
## [15,] 1.83   74 22.1
```

(f) Crie uma função `calcIMC` que dado um peso em kg e uma altura em metros, calcula o valor de IMC e retorna uma resposta de acordo com o que é apresentado na tabela abaixo. Teste a sua função na linha de comandos.

IMC nas Mulheres	**Condição**
< 19.1	pesoAbaixo
$19.1 - 25.8$	pesoNormal
> 25.8	pesoAcima

```
calcIMC <- function(imc) {
  if(imc < 19.1)
    return("pesoAbaixo")
  else
    if(imc < 25.8) # & imc >= 19.1
      return("pesoNormal")
    else # imc >= 25.8
      return("pesoAcima")
}
calcIMC(22)


## [1] "pesoNormal"
```

(g) Usando as funções `cbind()`, `sapply()`, crie uma matriz `m2` que resulta de anexar a `m1` uma quarta coluna com a informação do peso definida pela função da alínea anterior.

```
m2 <- cbind(m1,sapply(m1[,3],calcIMC))
```

(h) O que acontece ao conteúdo da matriz `m2`?

```
m2


##       [,1]   [,2] [,3]   [,4]
##  [1,] "1.47" "52" "24.1" "pesoNormal"
##  [2,] "1.5"  "53" "23.6" "pesoNormal"
##  [3,] "1.52" "54" "23.4" "pesoNormal"
##  [4,] "1.55" "56" "23.3" "pesoNormal"
##  [5,] "1.57" "57" "23.1" "pesoNormal"
##  [6,] "1.6"  "59" "23"   "pesoNormal"
##  [7,] "1.63" "60" "22.6" "pesoNormal"
##  [8,] "1.65" "61" "22.4" "pesoNormal"
##  [9,] "1.68" "63" "22.3" "pesoNormal"
## [10,] "1.7"  "64" "22.1" "pesoNormal"
## [11,] "1.73" "66" "22.1" "pesoNormal"
## [12,] "1.75" "68" "22.2" "pesoNormal"
## [13,] "1.78" "70" "22.1" "pesoNormal"
## [14,] "1.8"  "72" "22.2" "pesoNormal"
## [15,] "1.83" "74" "22.1" "pesoNormal"
```

(i) A partir da matriz `m1`, crie um data frame `df` com as 4 colunas seguintes: `altura`, `peso`, `imc` e `info` (com o resultado da função `calcIMC`).

```
df <- data.frame(altura=m1[,1],peso=m1[,2],imc=m1[,3],info=sapply(m1[,3],calcIMC))
df


##    altura peso  imc       info
## 1    1.47   52 24.1 pesoNormal
## 2    1.50   53 23.6 pesoNormal
## 3    1.52   54 23.4 pesoNormal
```

```
## 4    1.55   56 23.3 pesoNormal
## 5    1.57   57 23.1 pesoNormal
## 6    1.60   59 23.0 pesoNormal
## 7    1.63   60 22.6 pesoNormal
## 8    1.65   61 22.4 pesoNormal
## 9    1.68   63 22.3 pesoNormal
## 10   1.70   64 22.1 pesoNormal
## 11   1.73   66 22.1 pesoNormal
## 12   1.75   68 22.2 pesoNormal
## 13   1.78   70 22.1 pesoNormal
## 14   1.80   72 22.2 pesoNormal
## 15   1.83   74 22.1 pesoNormal
```

(j) Execute a função `summary()` sobre a matriz `m2` e o data frame `df`. Qual é a diferença?

```
summary(m2)
```

```
##        V1          V2          V3             V4
##  1.47   :1   52     :1   22.1   :4   pesoNormal:15
##  1.5    :1   53     :1   22.2   :2
##  1.52   :1   54     :1   22.3   :1
##  1.55   :1   56     :1   22.4   :1
##  1.57   :1   57     :1   22.6   :1
##  1.6    :1   59     :1   23     :1
##  (Other):9   (Other):9   (Other):5
```

```
summary(df)
```

```
##      altura           peso            imc                info
##  Min.   :1.470   Min.   :52.00   Min.   :22.10   pesoNormal:15
##  1st Qu.:1.560   1st Qu.:56.50   1st Qu.:22.15
##  Median :1.650   Median :61.00   Median :22.40
##  Mean   :1.651   Mean   :61.93   Mean   :22.71
##  3rd Qu.:1.740   3rd Qu.:67.00   3rd Qu.:23.20
##  Max.   :1.830   Max.   :74.00   Max.   :24.10
```

(k) Execute o comando `df$info` e verifique que se trata de um *factor* com apenas uma categoria. Altere essa coluna do data frame de modo a que seja um factor com as 3 categorias possíveis. Atenção: o conteúdo da coluna não deve ser alterado, apenas o conjunto das categorias possíveis.

```
df$info
```

```
##  [1] pesoNormal pesoNormal pesoNormal pesoNormal pesoNormal pesoNormal
##  [7] pesoNormal pesoNormal pesoNormal pesoNormal pesoNormal pesoNormal
## [13] pesoNormal pesoNormal pesoNormal
## Levels: pesoNormal
```

```
levels(df$info) <- c("pesoNormal","pesoAbaixo","pesoAcima")
```

(l) Execute novamente a função `summary()` sobre o data frame `df`. Qual é a diferença?

```
summary(df)
```

```
##      altura           peso            imc                info
##  Min.   :1.470   Min.   :52.00   Min.   :22.10   pesoNormal:15
##  1st Qu.:1.560   1st Qu.:56.50   1st Qu.:22.15   pesoAbaixo: 0
##  Median :1.650   Median :61.00   Median :22.40   pesoAcima : 0
##  Mean   :1.651   Mean   :61.93   Mean   :22.71
##  3rd Qu.:1.740   3rd Qu.:67.00   3rd Qu.:23.20
##  Max.   :1.830   Max.   :74.00   Max.   :24.10
```

(m) Qual o efeito dos comandos `df[-1,]` e `df[,-ncol(df)]`?

```
df[-1,]
```

```
##    altura peso  imc       info
## 2    1.50   53 23.6 pesoNormal
## 3    1.52   54 23.4 pesoNormal
## 4    1.55   56 23.3 pesoNormal
## 5    1.57   57 23.1 pesoNormal
## 6    1.60   59 23.0 pesoNormal
## 7    1.63   60 22.6 pesoNormal
## 8    1.65   61 22.4 pesoNormal
## 9    1.68   63 22.3 pesoNormal
## 10   1.70   64 22.1 pesoNormal
## 11   1.73   66 22.1 pesoNormal
## 12   1.75   68 22.2 pesoNormal
## 13   1.78   70 22.1 pesoNormal
## 14   1.80   72 22.2 pesoNormal
## 15   1.83   74 22.1 pesoNormal
```

```
df[,-ncol(df)]
```

```
##    altura peso  imc
## 1    1.47   52 24.1
## 2    1.50   53 23.6
## 3    1.52   54 23.4
## 4    1.55   56 23.3
## 5    1.57   57 23.1
## 6    1.60   59 23.0
## 7    1.63   60 22.6
## 8    1.65   61 22.4
## 9    1.68   63 22.3
## 10   1.70   64 22.1
## 11   1.73   66 22.1
## 12   1.75   68 22.2
## 13   1.78   70 22.1
## 14   1.80   72 22.2
## 15   1.83   74 22.1
```

(n) Adicione ao final do data frame duas linhas: uma com a altura = 1.74 e o peso = 50 e outra com altura = 1.50 e o peso = 74. Para tal, aceda ou índice da linha que está adicionar com `nrow(df) + 1`. Em alternativa, pode também usar a função `rbind()`.

```
df[nrow(df)+1,] <- list(1.74,50,round(50/1.74^2,1),calcIMC(round(50/1.74^2,1)))
df[nrow(df)+1,] <- list(1.50,74,round(74/1.50^2,1),calcIMC(round(74/1.50^2,1)))
df
```

```
##    altura peso  imc       info
## 1    1.47   52 24.1 pesoNormal
## 2    1.50   53 23.6 pesoNormal
## 3    1.52   54 23.4 pesoNormal
## 4    1.55   56 23.3 pesoNormal
## 5    1.57   57 23.1 pesoNormal
## 6    1.60   59 23.0 pesoNormal
## 7    1.63   60 22.6 pesoNormal
## 8    1.65   61 22.4 pesoNormal
## 9    1.68   63 22.3 pesoNormal
## 10   1.70   64 22.1 pesoNormal
## 11   1.73   66 22.1 pesoNormal
## 12   1.75   68 22.2 pesoNormal
## 13   1.78   70 22.1 pesoNormal
## 14   1.80   72 22.2 pesoNormal
## 15   1.83   74 22.1 pesoNormal
## 16   1.74   50 16.5 pesoAbaixo
## 17   1.50   74 32.9  pesoAcima
```

(o) Usando uma estratégia semelhante, adicione uma nova coluna ao data frame com a informação do sexo dos indíviduos a que estes dados dizem respeito: `"f"`. Em alternativa, pode também usar a função `cbind()`.

```
df[,ncol(df)+1] <- list(factor(rep("f",nrow(df)),levels=c("f","m")))
colnames(df)[ncol(df)] <- "sexo"
df
```

```
##    altura peso  imc       info sexo
## 1    1.47   52 24.1 pesoNormal    f
## 2    1.50   53 23.6 pesoNormal    f
## 3    1.52   54 23.4 pesoNormal    f
## 4    1.55   56 23.3 pesoNormal    f
## 5    1.57   57 23.1 pesoNormal    f
## 6    1.60   59 23.0 pesoNormal    f
## 7    1.63   60 22.6 pesoNormal    f
## 8    1.65   61 22.4 pesoNormal    f
## 9    1.68   63 22.3 pesoNormal    f
## 10   1.70   64 22.1 pesoNormal    f
## 11   1.73   66 22.1 pesoNormal    f
## 12   1.75   68 22.2 pesoNormal    f
## 13   1.78   70 22.1 pesoNormal    f
## 14   1.80   72 22.2 pesoNormal    f
## 15   1.83   74 22.1 pesoNormal    f
## 16   1.74   50 16.5 pesoAbaixo    f
## 17   1.50   74 32.9  pesoAcima    f
```