Introdução à Linguagem R - Workshop

Encontro 3

Davi Moreira

01 de Novembro, 2018

Sumário

1 Encontro 3

1.1 Dúvidas e revisão do conteúdo do encontro prévio

- 15 minutos serão reservados para dúvidas e revisão do conteúdo do encontro prévio.

1.2 Estrutura do encontro 3

3. REQUISITANDO DADOS DA WEB

- Extraindo conteúdo da página: rvest, xml2, css e xpath;
- Baixando arquivos da web: pdf, csv, excel;

Até o final do encontro o aluno deverá ser capaz de:

- Obter e organizar conteúdo da web de forma automatizada
- Desenvolver programas capazes de acessar páginas, realizar consultas e fazer download de arquivos

2 `for`, `while` e nossas próprias funções

As funções `for()` e `while()` implementam o controle de fluxo no `R`. A escolha de qual usar vai depender do contexto e objetivo do código.

2.1 `for`

```
# for()

for (i in 1:10) {
  print (i)
}
```

2.2 `while`

```
# while ()
x = 1

while (x <= 10 )  {
  print (x)
  x = x + 1
}
```

2.3 Criando funções

```
soma_dois <- function(x) { x + 2 }

soma_dois(4)

obj <- 1:15
```

```
soma_dois(obj)
```

3 purrr package

Para uma boa introdução sobre o pacote, veja o seguinte material:

- Curso R - Purrr
- Happy R Users Purrr – Tutorial
- Purrr Tutorial

```
install.packages("tidyverse")
library(purrr)
```

```
soma_dois <- function(x) { x + 2 }
obj <- 1:15

obj <- map(obj, soma_dois)
obj

# como a funcao map retorna uma lista, podemos usar sufixos para retornar um tipo
# de vetor específico

map_dbl(obj, soma_dois)
```

4 Web Scraping

Web Scraping é uma técnica de extração de dados utilizada para coletar conteúdo publicado na internet por meio de procedimentos automatizados.

4.1 Tipos de conteúdo disponível

4.1.1 Código fonte

É possível conhecer o código fonte de um site ao clicar com o botão direito do mouse no conteúdo da página.

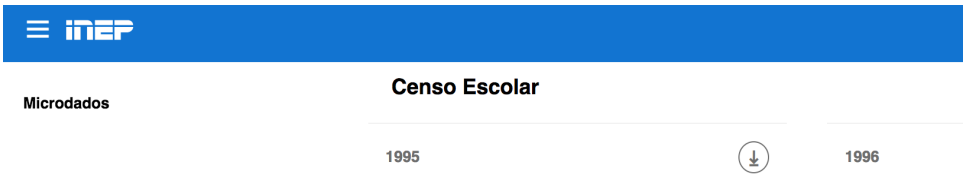
- Wikipedia

- Deputados

4.1.2 Arquivos para download

Além do conteúdo diretamente publicado na página, pode ser de interesse fazer o download de arquivos disponíveis.

- Censo Escolar



- TCE

4.1.3 Web services

Os Web services são componentes que permitem às aplicações enviar e receber dados. Um dos motivos que tornam os Web Services atrativos é o fato deste modelo ser baseado em tecnologias standards, em particular XML e HTTP (Hypertext Transfer Protocol). Os Web Services são utilizados para disponibilizar serviços interativos na Web, podendo ser acessados por outras aplicações. O objetivo dos Web Services é a comunicação de aplicações através da Internet.

- Web service da Câmara dos Deputados

4.2 Pacotes para raspagem de dados

Há diversos pacotes para raspagem de dados com o `R`. Abaixo segue um lista com os principais. Para referências sobre seu uso, consulte os links indicados, este tutorial sobre o 'rvest' e este capitulo sobre web scraping.

- 'httr'
- 'xml2'
- 'rvest'

Como o site Curso-R destaca, esses pacotes não são suficientes para acessar todo tipo de conteúdo da web. Páginas com conteúdo produzido na linguagem `javascript`, por exemplo, precisam de outras ferramentas para acesso a seu conteúdo. Nesses casos, é necessário "simular" um navegador que acessa a página web e realiza consultas. Uma das melhores ferramentas para isso é o selenium, abaixo indicado.

- 'RSelenium'

5 Obtendo conteúdo

5.1 Etapas para raspagem de dados na web

1. Conhecer detalhadamente o caminho para acesso aos dados
2. Armazenar todos os caminhos de acesso aos dados de forma amigável ao programa
3. Obter os dados
4. Processar os dados obtidos

5.2 Conteúdo de páginas

5.2.1 Código Fonte:

Podemos facilmente obter o código fonte de um endereço na internet com o uso da função `readLines`.

```
if(require(tidyverse) == F) install.packages('tidyverse'); require(tidyverse)
if(require(rvest) == F) install.packages('rvest'); require(rvest)
if(require(httr) == F) install.packages('httr'); require(httr)
if(require(xml2) == F) install.packages('xml2'); require(xml2)

################################################################################
# ETAPA 1. Conhecer detalhadamente o caminho para acesso aos dados
# ETAPA 2. Armazenar todos os caminhos de acesso aos dados de forma amigável ao programa

# defnindo endereço da web
```

```r
link <- "https://pt.wikipedia.org/wiki/Lista_de_munic%C3%ADpios_do_Brasil_por_IDH"

################################################################################
# ETAPA 3. Obter os dados
conteudo <- readLines(link)  # obtem o codigo fonte
head(conteudo)

# Vamos verificar a posição de Fenando de Noronha no vetor 'conteudo'.
grep("Fernando", conteudo)
conteudo[769 + 4]  # linha com o IDH de Fernando de Noronha

conteudo[769 + 9]  # Próximo município
conteudo[769 + 9 + 9] # Parece haver um padrão

# Com o objeto 'conteudo' já seria possível obter os dados para criacão do data frame
# com o IDH dos municípios.

################################################################################
# ETAPA 4. Processar os dados obtidos
# vamos selecionar todas linhas que apresentem os nomes dos municipios

grep("São Caetano", conteudo)  # 94
grep("Santa Maria", conteudo)  # 1066

indice <- 94
nomes_munic <- NULL
i <- 1

while(indice < 1066){
      if(i==1){
            nomes_munic[i] <- conteudo[indice]
      } else{
        nomes_munic[i] <- conteudo[indice+9]
      }
          indice <- indice + 9
          i <- i + 1
}

nomes_munic

?regex

nomes_munic <- gsub("[[:print:]]+\">", "", nomes_munic)
nomes_munic <- gsub("</a>", "", nomes_munic)
nomes_munic <- gsub("</b>", "", nomes_munic)
nomes_munic <- gsub("<b>", "", nomes_munic)

nomes_munic

# Poderíamos realizar procedimento semelhante para obter os IDHs municipais, os
# nomes da UFs e assim construir nosso data.frame
```

5.2.1.1 Atividade prática:

```r
# Identifique em qual linha do vetor 'conteudo' está Pernambuco. Adapte o exemplo visto
# para obter um vetor com os nomes das UFs.
```

5.2.2 Obtendo tabelas em html:

```r
##############################################################################################
# ETAPA 1. Conhecer detalhadamente o caminho para acesso aos dados
# ETAPA 2. Armazenar todos os caminhos de acesso aos dados de forma amigável ao programa
link <- "https://pt.wikipedia.org/wiki/Lista_de_munic%C3%ADpios_do_Brasil_por_IDH"


##############################################################################################
# ETAPA 3. Obter os dados
# ETAPA 4. Processar os dados obtidos

bd <- link %>%
  httr::GET() %>%
  xml2::read_html() %>%
  rvest::html_node('table') %>%
  rvest::html_table(header = TRUE)

bd
class(bd)

# E quando o link possui mais de uma tabela?

##############################################################################################
# ETAPA 1. Conhecer detalhadamente o caminho para acesso aos dados
# ETAPA 2. Armazenar todos os caminhos de acesso aos dados de forma amigável ao programa
link <- "https://pt.wikipedia.org/wiki/Lista_de_campe%C3%B5es_do_futebol_brasileiro"

##############################################################################################
# ETAPA 3. Obter os dados
bd <- link %>%
  httr::GET() %>%
  xml2::read_html() %>%
  rvest::html_nodes('table') %>%  # veja que utilizamos outra função
  rvest::html_table(header = TRUE)

class(bd)

##############################################################################################
# ETAPA 4. Processar os dados obtidos
bd1 <- bd[[1]]
bd2 <- bd[[2]]
bd3 <- bd[[3]]
```

5.2.2.1 Atividade prática:

```r
# Com o link abaixo, desenvolva um progrma que obtenha o endereço das páginas de todos
# os deputados federais da atual legislatura alocando-os num vetor.
```

```
link <- "http://www.camara.leg.br/internet/deputado/DepNovos_Lista.asp?
          Legislatura=54&Partido=QQ&SX=QQ&Todos=None&UF=QQ&condic=QQ&forma=lista&
          nome=&ordem=nome&origem="
```

5.3 Download de arquivos

```
################################################################################
# ETAPA 1. Conhecer detalhadamente o caminho para acesso aos dados
link <- "https://www.tce.pe.gov.br/internet/index.php/relatorios-de-gestao-fiscal-2"

################################################################################
# ETAPA 2. Armazenar todos os caminhos de acesso aos dados de forma amigável ao programa
link_relatorios <- link %>% read_html %>% html_nodes("a") %>% html_attr('href')
link_relatorios <- link_relatorios[grep("rdg", link_relatorios)]

################################################################################
# ETAPA 3. Obter os dados
mainDir <- paste(getwd(), "/dados/", sep = "")
subDir <- "relatorios_tce"

dir.create(file.path(mainDir, subDir), showWarnings = FALSE)

setwd("./dados/relatorios_tce/")

for( i in 1:length(link_relatorios)){
   download.file(link_relatorios[i], paste0(as.character(c(2017:2006))[i], ".pdf"), mode="auto")
}

download.file(link_relatorios)

################################################################################
# ETAPA 4. Processar os dados obtidos
if(require(pdftools) == F) install.packages('pdftools'); require(pdftools)

setwd("./relatorios_tce/")
rdg2017 <- pdf_text("2017.pdf")

length(rdg2017)
head(rdg2017)

# Preparando data.frame para nuvem de palavras
# Instalando pacotes
if(require(tm) == F) install.packages('tm'); require(tm)
if(require(SnowballC) == F) install.packages('SnowballC'); require(SnowballC)
if(require(wordcloud) == F) install.packages('wordcloud'); require(wordcloud)
if(require(RColorBrewer) == F) install.packages('RColorBrewer'); require(RColorBrewer)

docs <- Corpus(VectorSource(rdg2017))

# convertendo o texto em caixa baixa
docs <- tm_map(docs, content_transformer(tolower))
```

```r
# removendo números
docs <- tm_map(docs, removeNumbers)

# removendo stopwords (artigos, preposições, etc.)
docs <- tm_map(docs, removeWords, stopwords("portuguese"))

# removendo pontuação
docs <- tm_map(docs, removePunctuation)

# eliminando espaços
docs <- tm_map(docs, stripWhitespace)

# stemming
docs <- tm_map(docs, stemDocument, language = "portuguese")

# document term matrix - matriz de documentos e termos
dtm <- TermDocumentMatrix(docs)

# criando data.frame
m <- as.matrix(dtm)
v <- sort(rowSums(m), decreasing=TRUE)
d <- data.frame(word = names(v),freq=v)
head(d)
d <- d[-3,]  # removendo caracter especial
head(d)

# nuvem de palavras
set.seed(1234)
wordcloud(words = d$word, freq = d$freq, min.freq = 1,
          max.words=100, random.order=FALSE, rot.per=0.35,
          colors=brewer.pal(8, "Dark2"))
```

5.3.0.1 Atividade prática:

```
# Crie um novo diretório e desenvolva um programa que faça o download dos arquivos
# em formato .zip na página de dados do Censo Escolar (http://inep.gov.br/microdados).
# Para otimizar nosso tempo, selecione apenas os microdados do Censo Escolar de
# 1996, 2006, 2016.
```

5.4 Web service

```
if(require(httr) == F) install.packages('httr'); require(httr);
if(require(XML) == F) install.packages('XML'); require(XML);
if(require(xml2) == F) install.packages('xml2'); require(xml2);

#############################################################################
# ETAPA 1. Conhecer detalhadamente o caminho para acesso aos dados
# ETAPA 2. Armazenar todos os caminhos de acesso aos dados de forma amigável ao programa

link <- paste0("http://www.camara.leg.br/SitCamaraWS/Deputados.asmx/ObterDeputados")

#############################################################################
# ETAPA 3. Obter os dados
response <- GET(link)

#############################################################################
# ETAPA 4. Processar os dados obtidos
data <- xmlParse(response, encoding = "UTF-8")
ls <- xmlToList(data)

names(ls$deputado)

ideCadastro <- NULL
condicao <- NULL
matricula <- NULL
idParlamentar <- NULL
nome <- NULL
nomeParlamentar <- NULL
urlFoto <- NULL
sexo <- NULL
uf <- NULL
partido <- NULL
email <- NULL

for(i in 1:length(ls)){
  ideCadastro[i] <- ls[[i]]$ideCadastro
  condicao[i] <- ls[[i]]$condicao
  matricula[i] <- ls[[i]]$matricula
  idParlamentar[i] <- ls[[i]]$idParlamentar
  nome[i] <- ls[[i]]$nome
  nomeParlamentar[i] <- ls[[i]]$nomeParlamentar
  urlFoto[i] <- ls[[i]]$urlFoto
  sexo[i] <- ls[[i]]$sexo
  uf[i] <- ls[[i]]$uf
  partido[i] <- ls[[i]]$partido
```

```
  email[i] <- ls[[i]]$email
}

bd <- data.frame(ideCadastro, condicao, matricula, idParlamentar, nome,
                 nomeParlamentar, urlFoto, sexo, uf, partido, email)

head(bd)
```

5.4.0.1 Atividade prática:

Com a base de dados obtida, utilize a variável ideCadastro para obter detalhes dos Deputados Federais que representam o Estado de Pernambuco, confome permitido pelo link ObterDetalhesDeputado

6 Atividade Prática - Encontro 2

- Com os dados do Censo Escolar de 2016, construa uma base de dados municipal que apresente o número de turmas, docentes e matrículas por município. Em seguida faça a união dessa base com o Atlas dos Municípios (atlas2013_dadosbrutos_pt.xlsx), utilizando os dados de 2010 presentes na aba "MUN 91-00-10".

7 Atividade Prática - Encontro 3

```
# Utilizando o vetor de endereços das páginas dos deputados federais obtido com o link abaixo,

link <- "http://www.camara.leg.br/internet/deputado/DepNovos_Lista.asp?
        Legislatura=54&Partido=QQ&SX=QQ&Todos=None&UF=QQ&condic=QQ&forma=lista&
        nome=&ordem=nome&origem="

# Acesse cada uma das páginas e monte uma base de dados que tenha as seguintes
# informações biográficas de cada deputado: Nome, Data de Nascimento, Naturalidade,
# Profissao, Filiacao e Escolaridade.

# Qual a proporção de Deputados Federais com ensino superior?
```

8 Links úteis para o próximo encontro

- Análise exploratória
- Regressão Linear
- Pacote ggplot2
- Data Visualisation
- CursoR: ggplot2