Como usar a API do CepespData para navegar dados eleitorais - ABRAJI 2019

Equipe CepespData/FGV

28 de junho de 2019

1. O CepespData

O CepespData é uma plataforma de acesso a dados eleitorais desenvolvida pelo Centro de Política e Economia do Setor Público (CEPESP) da Fundação Getulio Vargas (FGV) com apoio da FAPESP ao projeto As Instituições Políticas Subnacionais: Um Estudo Comparativo dos Estados Brasileiros (processo: 2013/15658-1) e do Tribunal Superior Eleitoral - TSE.

Hoje o CEPESPData possui bancos com informações das candidaturas e resultados das eleições brasileiras entre os anos de 1998 e 2018, dados de filiação partidária, da composição das secretarias estaduais entre 1998 e 2018 e da declaração de bens dos candidatos. Em breve, teremos dados da prestação de contas das campanhas, informações dos diretórios partidários dentre outros.

Os resultados eleitorais são aqueles disponibilizados pelo TSE no dia da apuração. Esses dado não são atualizados pelo TSE conforme decisões judiciais para indeferimento ou deferimento de candidaturas; portanto, são a representação mais próxima do que ocorreu nas eleições. Um persistente trabalho para reforçar a consistência dos dados garante que as diferenças de votos reportada pelas bases do Cepespdata /FGV nunca sejam superior a 1% aos resultados reportados pelo TSE, tomando como base a disponibilização dos resultados eleitorais por seção, no respositório de dados eleitorais.

Um dos principais diferenciais dos dados do CEPESPData é a a possibilidade de consultar os bancos de resultados eleitorais, candidatos e coligações - que hoje são disponibilizados separadamente pelo TSE - já integrados entre si. Nosso portal e nossa API permite ao usuário consultar os dados com diversos filtros e especificações de maneira automatizada e fácil. Além disso, os dados eleitorais já vêm com os códigos por município informado pelo TSE com seu equivalente para cada divisão administrativa do IBGE, ampliando as possibilidades de uso dos dados com outras fontes.

O cepespR é um pacote em linguagem R criado para auxiliar o acesso dos usuários à API do CEPESPData. Por meio dessa ferramenta, é possível realizar requisições de maneira mais rápida e estruturada aos dados eleitorais presentes no nosso repositório.

Esse workshop apresentará as principais funcionalidades do cepespR e como trabalhar com cada uma das requisições disponíveis. Também apresentaremos algumas operações simples com os dados com o pacote *dplyr*, em especial para mostrar como trabalhar com o dados do CepespData com outros bancos.

2. Instalando o cepespR

O pacote **cepespR** está hospedado em nosso github, então para instala-lo é preciso rodar o código abaixo apenas uma vez. *(apenas em caso de troca de computador é preciso instalar novamente o pacote, mas recomendamos atualiza-lo sempre que possível para garantir as atualizações de novos dados e requisições).* Também vamos instalar o pacote *dplyr* para nos auxiliar nas operações com os dados.

```
if (!require("devtools")) install.packages("devtools")
devtools::install_github("Cepesp-Fgv/cepesp-r")
install.packages("dplyr")
```

Uma vez instalado o CepespR, o pacote é ativado mediante a função `library`. Lembre-se que é preciso executar essa função **toda vez que iniciar o R**, senão as funções do **cepespR** não irão funcionar. Isso também vale para o pacote dplyr e qualquer outro que seja necessário em sua sessão.

```
library(cepespR)
library(dplyr)
```

3. Explorando as requisições via CepespR

Existem 7 requisições disponíveis hoje no pacote cepespR. Todas tem como parâmetros obrigatórios o ano (parâmetro *year*) e o cargo (parâmetro *position*) disputado e recebem como padrão os dados por município. Cada função recebe um data.frame com as colunas já formatadas no tipo de variável correto - por isso, é preciso indicar um objeto para salva-lo no seu ambiente no R. Opcionalmente, as funções recebem parâmetros que auxiliam no filtro dos dados. Os parâmetros podem ser indicados tanto em Português quanto Inglês. Para mais detalhes, consulte nossa documentação no GitHub

3.1 get_votes

Recupera as informações de quantos votos cada candidato recebeu em determinada eleição. É obrigatório informar o ano (*year*) e o cargo (*position*), recebendo como padrão a votação por município de todos os candidatos que receberam votos naquele ano e cargo. Opcionalmente, pode-se pedir os dados agregados

```
votos_pres18 <- get_votes(year=2018, #Obrigatório: ano da eleição
                          position="Presidente", #obrigatório: cargo disputado
                          candidate_number = 13, #opcional: filtra por candidato
                          regional_aggregation="Estado") #opcional: votos agregados por Estado.
```

3.2 get_candidates

Retorna os detalhes das candidaturas aptas registradas em cada eleição.

Exemplo: obter os candidatos eleitos a deputado federal pelo PT em 2002

```
deputadosPT2002 <- get_candidates(year=2002, #obrigatório: ano da eleição
                                  position="Deputado Federal", #obrigatório: cargo disputado
                                  only_elected = T, #opcional: receber apenas os eleitos
                                  party = 13) #opcional: receber apenas os candidatos do PT
```

3.3 get_coalitions

Retorna a composição das coligações deferidas que competiram em cada eleição.

Exemplo: obter as coligações partidárias para prefeito de 2004 com o DEM

```
coligacoes2004DEM <- get_coalitions(year=2004,#obrigatório: ano da eleição
                                    position = "Prefeito",#obrigatório: cargo disputado
                                    party=25)#opcional: receber apenas coligações com o DEM
```

3.4 get_elections

Função mais completa do cepespR, retorna todos os dados de resultados eleitorais, candidaturas e coligações juntos. Permite retornar os votos com diferentes agregações regionais (desde a soma do país todo até seção eleitoral) e políticas (do candidato à coligação). A agregação padrão é por município e por candidato.

Exemplo: obter o total de votos que os candidatos a prefeito eleitos pelo MDB no Rio de Janeiro

```
prefeitosMDBrio <- get_elections(year=2012,#obrigatório: ano da eleição
                                 position="Prefeito",#obrigatório: cargo disputado
                                 regional_aggregation="Estado", #opcional: votos agregados por Estado.
                                 political_aggregation="Partido", #opcional: votos agregados por partid
                                 state = "RJ", #opcional: receber apenas dados do estado do RJ
                                 party = 15, #opcional: receber apenas dados do MDB
                                 only_elected = T) #opcional: receber apenas os eleitos
```

3.5 get_assets

Recupera os bens declarados ao TSE pelos candidatos em cada eleição.

Exemplo: Bens declarados pelos candidatos do Piauí em 2018

```
bensPiaui2018 <- get_assets(year = 2018, #obrigatório: ano da eleição
                            state = "PI") #opcional: receber apenas dados do estado do Piauí
```

3.6 get_secretaries

Banco de dados inédito do Cepesp que reúne informações sobre ocupantes de cargos do primeiro escalão dos governos estaduais e do Distrito Federal. Para mais informações sobre os dados, clique aqui

Exemplo: Todas as secretárias e secretários estaduais de São Paulo entre 1998 e 2002

```
secSP <- get_secretaries(state="SP", #obrigatório: Estado.
                         name=NULL, #obrigatório: NULL para receber todos ou parte do nome para fitrar.
                         period = "1998-2002")#opcional: indicar o quadriênio de interesse
```

3.7 get_filiates

Retorna os dados dos filiados conforme declarado pelos partidos, atualizados em novembro de 2018. É preciso informar o Estado e o partido que deseja consultar a relação.

Exemplo: filiados ao partido NOVO no estado da Bahia

```
novoBA <- get_filiates(state="BA", #obrigatório. sigla do Estado
                       party = "NOVO") #obrigatório. Sigla do partido
```

4. Filtros, requisições avançadas e cache das consultas

Conforme visto nas funções acima, podemos agilizar a consulta fazendo filtros diretamente na requisição. Abaixo, explicamos cada uma delas.

4.1 Filtro por eleito(a)s

Nas funções get_candidates e get_elections, podemos limitar os resultados apenas àqueles que se elegeram incluindo o parâmetro "only_elected=T" na consulta:

```
prefeitoseleitos12 <- get_candidates(year=2012,
                                     position="Prefeito",
                                     only_elected = T) #opcional: receber apenas eleita(o)s
```

4.2 Filtro por partido

Também podemos limitar os resultados para apenas o partido que queremos. Basta informar o número. Está disponível nas funções get_candidates, get_coalitions, get_elections e get_filiates.

```
prefeitoseleitos12 <- get_candidates(year=2012,
                                     position="Prefeito",
                                     only_elected = T,
                                     party = 13) #opcional: receber apenas os candidatos do PT
```

4.3 Filtro pelo número do(a) candidato(a)

Semelhante ao filtro por partido, basta informar o número do candidato. Está disponível nas funções get_ votes, get_candidates e get_elections.

```
cand <- get_elections(year = 2016,
                      position="Vereador",
                      regional_aggregation="Municipio",
                      candidate_number = 25000) #selecionar dados de candidatos com o número 25000
```

4.4 Filtro por Estado

Retorna apenas as candidaturas do Estado indicado. Está disponível nas funções get_ votes e get_elections; nas funções get_assets, get_secretaries e get_filiates, é obrigatório.

```
cand <- get_elections(year = 2016,
                      position="Vereador",
                      regional_aggregation="Municipio",
                      state = "CE") #selecionar dados apenas do Ceará
```

4.5 Seleção das colunas do banco

É possível indicar quais colunas queremos na nossa consulta, caso contrário, a função devolverá todas as colunas disponíveis. É possível consultar quais são as colunas padrão de cada banco clicando aqui Essa opção está disponível para todas as funções.

Exemplo:

```r
#indicando uma lista com as colunas
colunas <- list("NUMERO_CANDIDATO","NOME_URNA_CANDIDATO","UF","QTDE_VOTOS")

presid14 <- get_elections(year = 2014,
                      position="Presidente",
                      regional_aggregation="Estado",
                      columns_list=colunas) #indicar aqui a lista criada
```

4.6 Informações para mais de um ano

Todas as requisições aceitam que se consulte mais de um ano de uma vez. Para isso, basta informar entre parenteses os anos a serem consultados e separa-los por vírgula - tomando o cuidado de informar anos eleitorais válidos.

Exemplo: Todos os prefeitos eleitos pelo PMDB no Rio de Janeiro entre 2008 e 2016:

```r
prefsPMDBrio <- get_elections(year="2008,2012,2016", #indica os três anos que queremos
                                  position="Prefeito",
                                  regional_aggregation="Municipality",
                                  political_aggregation="Candidate",
                                  state = "RJ",
                                  party = "15",
                                  only_elected = T)
```

4.7 Informações para mais de um cargo

Para conseguir os resultados para mais de um cargo, é preciso fazer um *for loop* para cada um dos cargos e empilhar os resultados num dataframe. Essa mesma lógica se aplica também para recuperar os dados de mais de um partido ou mais de uma UF na função get_filiates.

Exemplo: Todos os prefeitos e vereadores eleitos pelo PMDB no Rio de Janeiro entre 2008 e 2012:

```r
#crie um vetor com cada cargo requisitado separado por vírgula e entre aspas:
lista.cargos <- c("Vereador","Prefeito")

#crie um dataframe vazio para receber os dados:
bancocompleto <- data.frame()

#pedir para que a requisição seja feita para cargo da lista, um por vez, até o final da lista.
for(cargo in lista.cargos){
  #salvando a requisição num banco temporário
  bancotemporario <- get_elections(year="2008,2012,2016",
                        position=cargo, #preencherá com um cargo da lista de cargos por vez
                        regional_aggregation="Municipality",
                        political_aggregation="Candidate",
                        state = "RJ",
                        party = "15",
                        only_elected = T)

#empilhando os dados temporários no banco de dados completo
bancocompleto <- rbind(bancocompleto,bancotemporario)

#remove o banco temporário com os dados parciais
```

```
rm(bancotemporario)
}
```

4.8 Cache das consultas

A cada consulta feita na API, o banco de dados pedido será construído e baixado em sua máquina. Para limitar a banda consumida e agilizar as requisições mais comuns, é possível salvar uma cópia dos dados em sua máquina. Você poderá depois deletar esse dado manualmente caso queira atualizar a requisição.

Para isso, basta incluir o parâmetro "cached=T" ao final de qualquer uma das funções disponíveis. Assim, uma cópia dos dados será salva em "/static/cache" no seu diretório de trabalho e estará disponível automaticamente quando repetir a consulta. Exemplo:

```
pslAC<- get_filiates(state="AC",
                     party = "PSL",
                     cached = T) #parâmetro cached marcado como "TRUE"
```

5. Exercícios

Para cada um dos exercícios abaixo, responda:

- *Qual função do cepespR você utilizaria?*

- *Quais são os parâmetros que você deve informar?*

- *Seria preciso fazer outras operações no banco? Quais?*

1. Quantos votos na legenda o PSL teve em 2018 para Deputado Federal? E em 2014? Houve aumento?
2. Quantas governadoras foram eleitas nas últimas quatro eleições?
3. Quantas mulheres negras (pretas ou pardas) concorreram ao cargo de prefeita em 2016 no Brasil?
4. Quantas pessoas que eram filiadas ao PCO no Estado de Alagoas se desfiliaram do partido ou tiveram sua filiação cancelada?
5. Considerando os candidatos a vereador no Rio Grande do Sul em 2012, qual é o partido com o maior valor total de bens declarados?

6. Trabalhando com outras bases de dados a partir do código do IBGE

Uma das vantagens de utilizar os dados eleitorais do CepespData é já ter pronta a compatibilização do código do munícipio informado pelo TSE com o código das regiões administrativas do IBGE. Assim, a junção dos nossos dados eleitorais com qualquer outro que tenha o código do IBGE pode ser feita em poucas linhas de programação.

Nesse exemplo, vamos explorar como juntar os dados do Cepesp com os dados dos beneficiários do Programa Bolsa Família em 2016 e das estimativas populacionais do IBGE enviadas ao TCU de 2016.

Os dados pré-processados em .csv podem ser baixados clicando aqui

A pergunta que queremos responder é: existe correlação entre a porcentagem de votos do PT num município e a porcentagem de famílias do Programa Bolsa Família em 2016? *(lembrando que correlação não é a mesma coisa que causa! O foco aqui é o trabalho com os dados e não as conclusões. Para afirmar que o*

Programa Bolsa Família impacta positivamente no resultado eleitoral, precisaríamos de uma análise muito mais sofisticada que a proposta nesse exercício.)

Passo a passo:

- 1. Baixe os dados do Programa Bolsa Família (pbf_2016.csv) e abra no R. Dica: veja como utilizar a função *read.csv2()*. Inspecione quais são as variáveis no banco.
- 2. Baixe os dados da população do IBGE (pop_ibge_2016.csv)e abra no R. Inspecione quais são as variáveis no banco.
- 3. Utilizando o cepespR, construa um banco de dados que tenha a quantidade de votos por município por partido para o cargo de prefeito no ano de 2016
- 4. Crie uma variável que indique a porcentagem de votos recebida por partido por município.
- 5. Junte o banco de população com os dados do Bolsa Família utilizando o código do IBGE. *Dica: Verifique se o tipo da variável do código do IBGE nos dos bancos são iguais - caso contrário, uniformize-as. Também veja quantos dígitos o código possui - em alguns bancos, o código do IBGE vem com o dígito verificador, que pode ser descartado sem problemas. Se for preciso descarta-lo, procure como usar a função* substr()* e guarde numa nova variável.*
- 6. Verifique se o join foi feito corretamente explorando o banco com o comando *summary()*. Não podemos ter NA's!
- 7. Nesse novo banco, crie uma variável que indique o número de famílias benefiárias a cada 1000 habitantes
- 8. Agora, junte o banco com a variável do item 7 ao banco com as porcentagens de voto. *Dica: lembre-se de usar a variável do código do IBGE com o mesmo número de digitos*
- 9. Faça um gráfico de dispersão simples (*plot()*) entre as variáveis de famílias beneficiárias e a porcentagem de votos para candidatos do PT. Parece haver correlação?

FGV | EAESP EESP
CENTRO DE POLÍTICA E ECONOMIA DO SETOR PÚBLICO

Rua Itapeva, 286 - 10° andar - São Paulo/SP - CEP: 01332-000

Telefone:(11) 3799 - 3228

E-mail: cepesp@fgv.br / midia.cepesp@fgv.br

Twitter / Facebook / Instagram / GitHub