

# Trabalhando com dados amostrais

## Aula 05

Frederico Bertholini

# Preparação para a aula

- ▶ Baixe os dados da pasta exercícios (ou faça pull do GitHub)
- ▶ Configure o GitHub na sua máquina – Versionamento ->  
<https://www.curso-r.com/blog/2017-07-17-rstudio-e-github/> –  
Instruções adicionais de instalação  
<http://r-bio.github.io/git-installation/>
- ▶ Rode todos os pacotes (usando o macetinho) - baixe exercício 5
- ▶ Repositório no GitHub [https://github.com/fredbsr/aulas\\_ENAP/tree/master/CADS2018](https://github.com/fredbsr/aulas_ENAP/tree/master/CADS2018)

## Exercício

- ▶ Qual juiz julga a maior proporção de processos que tratam de drogas
- Dica: construa um `data.frame` contendo as colunas `juiz`, `n_processos_drogas`, `n_processos_n_drogas` e `total_processos`, remodelando os dados para haver um juiz por linha e utilizando `spread()`

## Resolução

```
decisoes %>%  
  filter(!is.na(txt_decisao)) %>%  
  mutate(txt_decisao = tolower(txt_decisao),  
         droga = str_detect(txt_decisao,  
                             "droga|entorpecente|psicotr[óo]pico|maconha|haxixe|coca  
         droga=case_when(  
           droga==TRUE ~ "droga",  
           droga==FALSE ~ "n_droga"  
         )) %>%  
  group_by(juiz,droga) %>%  
  summarise(n=n()) %>%  
  spread(droga,n,fill = 0) %>%  
  mutate(total=droga+n_droga,  
         proporcao=droga/total)
```

## Exercício

- ▶ Qual quantidade mensal de decisões por juiz?
- ▶ Dica: use `data_decisao` `dmy()` e `month()`

## Resolução

```
decisoes %>%
  filter(!is.na(txt_decisao)) %>%
  mutate(txt_decisao = tolower(txt_decisao),
         droga = str_detect(txt_decisao,
                           "droga|entorpecente|psicotr[óo]pico|maconha|haxixe|coca
         droga=case_when(
           droga==TRUE ~ "droga",
           droga==FALSE ~ "n_droga"
         )) %>%
  group_by(juiz,droga) %>%
  summarise(n=n()) %>%
  spread(droga,n,fill = 0) %>%
  mutate(total=droga+n_droga,
         proporcao=droga/total)
```

## Resultado

```
## # A tibble: 65 x 5
## # Groups:   juiz [65]
##   juiz          droga n_droga total proporcao
##   <chr>         <dbl>   <dbl> <dbl>      <dbl>
## 1 Airton Vieira      23     131   154      0.149
## 2 Alcides Malossi Junior 23      72    95      0.242
## 3 Alexandre Almeida    41     122   163      0.252
## 4 Amaro Thom         36      96   132      0.273
## 5 Andrade Sampaio     35      79   114      0.307
## 6 Ang lica de Almeida   2       6    8      0.25
## 7 Antonio Tadeu Ottoni  0       1    1      0
## 8 Bandeira Lins        0       2    2      0
## 9 Camargo Aranha Filho 32     109   141      0.227
## 10 Camilo L ellis      32     133   165      0.194
## # ... with 55 more rows
```

## Exemplo para o ggplot



## Unindo e separando colunas

- ▶ `unite` junta duas ou mais colunas usando algum separador (`_`, por exemplo).
- ▶ `separate` faz o inverso de `unite`, e uma coluna em várias usando um separador.

## Exemplo de separação de colunas

- ▶ Olhe os valores da variável `classe_assunto`

## Exemplo de separação de colunas

- ▶ Vamos separar a coluna classe\_assunto em duas colunas
- ▶ coluna classe e coluna assunto
- ▶ Existe separador? -> sim, /
- ▶ Usei count apenas em assunto

## Em ação

```
decisoes %>%  
  select(n_processo, classe_assunto) %>%  
  separate(classe_assunto, c('classe', 'assunto'), sep = ' ',  
           extra = 'merge', fill = 'right') %>%  
  count(assunto, sort = TRUE)  
  
## count é um jeito resumido de usar group_by() %>% summar
```

## Em ação

```
## # A tibble: 152 x 2
##   assunto                                n
##   <chr>                                <int>
## 1 Tráfico de Drogas e Condutas Afins  2441
## 2 Pena Privativa de Liberdade         1106
## 3 Roubo Majorado                      1093
## 4 Furto Qualificado                   838
## 5 Roubo                               780
## 6 Progressão de Regime                 607
## 7 Furto                               450
## 8 Receptação                          353
## 9 Homicídio Qualificado                329
## 10 Crimes de Trânsito                  322
## # ... with 142 more rows
```

## List columns: `nest()` e `unnest()`

`nest()` e `unnest()` são operações inversas e servem para tratar dados complexos, como o que temos em `processos`

```
d_partes <- processos %>%  
  select(n_processo, partes) %>%  
  unnest(partes)
```

As list columns são uma forma condensada de guardar dados que estariam em múltiplas tabelas. Por exemplo, uma alternativa à colocar as partes numa list column seria guardar a tabela d\_partes separadamente.

```
glimpse(d_partes)
```

```
## Observations: 37,579
## Variables: 5
## $ n_processo <chr> "00000003-71.2016.8.26.0073", "00000003
## $ id          <int> 1, 1, 2, 1, 1, 2, 1, 1, 2, 1, 1, 2, 1
## $ name        <chr> "JOSE MARIA JUSTINO NETO", "Defensor
## $ part        <chr> "Apelante", "Apelante", "Apelado", "A
## $ role        <chr> "Apelante", "Apelante", "Apelado", "A
```

## Duplicatas

Para retirar duplicatas, utilizar `distinct`. Ele considera apenas a primeira linha em que encontra um padrão para as combinações de variáveis escolhidas e descarta as demais.

```
decisoes %>%  
  distinct(municipio)
```

```
## # A tibble: 315 x 1  
##   municipio  
##   <chr>  
## 1 Cosmópolis  
## 2 São Paulo  
## 3 Ribeirão Preto  
## 4 Araçatuba  
## 5 Presidente Prudente  
## 6 Bertioga  
## 7 Taubaté  
## 8 Aparecida  
## 9 ...
```



## Por coluna

Para manter as demais colunas, use `.keep_all=`:

```
decisoes %>%  
  distinct(municipio, camara,  
           .keep_all = TRUE)
```

```
## # A tibble: 2,760 x 9
```

```
##   id_decisao n_processo classe_assunto municipio ca  
##   <chr>      <chr>      <chr>      <chr>      <chr>  
## 1 11094999 0057003-20~ Habeas Corpus / ~ Cosmópolis 3  
## 2 11093733 0052762-03~ Habeas Corpus / ~ São Paulo 3  
## 3 11093677 0055169-79~ Habeas Corpus / ~ Ribeirão 3  
## 4 11093270 9000580-82~ Agravo de Execu~ Araçatuba 8  
## 5 11093374 0052938-79~ Mandado de Segur~ São Paulo 8  
## 6 11093320 9000723-79~ Agravo de Execu~ Presiden~ 8  
## 7 11091506 0003276-86~ Apelação / Tráfi~ Bertioga 8  
## 8 11093326 9000298-11~ Agravo de Execu~ Taubaté 8  
## 9 11092475 0004653-39~ Apelação / Tráfi~ Aparecida 8
```

## janitor::get\_dupes()

Use `janitor::get_dupes()` para averiguar os casos em que há repetição de combinações de colunas.

```
decisoes %>%  
  get_dupes(n_processo)
```

```
## # A tibble: 114 x 10
```

```
##   n_processo   dupe_count id_decisao classe_assunto
```

```
##   <chr>          <int> <chr>      <chr>
```

```
## 1 0000276-86.~      2 11051087  Apelação / Tráfico
```

```
## 2 0000276-86.~      2 11093633  Embargos de Decla
```

```
## 3 0000358-10.~      2 11108278  Embargos de Decla
```

```
## 4 0000358-10.~      2 11028129  Apelação / Roubo
```

```
## 5 0002236-18.~      2 11041351  Apelação / Contra
```

```
## 6 0002236-18.~      2 11041352  Apelação / Contra
```

```
## 7 0004453-20.~      2 11041132  Apelação / Tráfico
```

```
## 8 0004453-20.~      2 11093635  Embargos de Decla
```

```
## 9 0004636-51.~      3 11032094  Apelação / Tráfico
```

## Joins

# Dados relacionais

- ▶ Hadley Wickham <http://r4ds.had.co.nz/relational-data.html>

# Principais funções

Para juntar tabelas, usar `inner_join`, `left_join`, `anti_join`, etc.

Visualizando

## Exemplo de inner join:

```
decisoes %>%  
  filter(data_registro == "18/01/2018", !is.na(id_decisao))  
  select(id_decisao, n_processo) %>%  
  inner_join(processos, "n_processo")
```

```
## # A tibble: 169 x 5
```

##		id_decisao	n_processo	infos
##		<chr>	<chr>	<list>
##	1	11109089	0003779-93.2015.8.26.0597	<tibble [14 x 2]
##	2	11109088	3001293-25.2013.8.26.0510	<tibble [13 x 2]
##	3	11108246	0063566-45.2015.8.26.0050	<tibble [14 x 2]
##	4	11108245	0003528-84.2015.8.26.0400	<tibble [14 x 2]
##	5	11109087	0008470-76.2015.8.26.0072	<tibble [14 x 2]
##	6	11109086	0013767-62.2012.8.26.0624	<tibble [14 x 2]
##	7	11109085	3019561-54.2013.8.26.0405	<tibble [14 x 2]
##	8	11108348	0003072-91.2017.8.26.0521	<tibble [11 x 2]
##	9	11108725	0009578-41.2017.8.26.0050	<tibble [12 x 2]
##	10	11108347	3001116-52.2013.8.26.0028	<tibble [12 x 2]

## Exemplo de right join:

```
decisoes %>%  
  filter(data_registro == "18/01/2018", !is.na(id_decisao))  
  select(id_decisao, n_processo) %>%  
  right_join(processos, "n_processo")
```

```
## # A tibble: 11,638 x 5
```

##	id_decisao	n_processo	infos
##	<chr>	<chr>	<list>
## 1	<NA>	0000003-71.2016.8.26.0073	<tibble [11 x 2]
## 2	<NA>	0000004-09.2017.8.26.0142	<tibble [12 x 2]
## 3	<NA>	0000004-34.2016.8.26.0630	<tibble [12 x 2]
## 4	<NA>	0000004-59.2015.8.26.0633	<tibble [14 x 2]
## 5	<NA>	0000004-62.2014.8.26.0611	<tibble [14 x 2]
## 6	<NA>	0000006-04.2017.8.26.0651	<tibble [12 x 2]
## 7	<NA>	0000006-06.2015.8.26.0576	<tibble [12 x 2]
## 8	<NA>	0000006-63.2017.8.26.0599	<tibble [12 x 2]
## 9	<NA>	0000006-74.2010.8.26.0125	<tibble [14 x 2]
## 10	<NA>	0000006-82.2016.8.26.0604	<tibble [11 x 2]



## Exercício

- ▶ Crie um objeto contendo informações sobre os tamanhos das bancadas dos partidos (arquivo `bancadas.rds`), suas respectivas coligações eleitorais para 2018 (arquivo `coligacoes.xlsx`) e o grau de concordância com a agenda do Gov Temer (arquivo `governismo_temer.xlsx`).
- ▶ Crie uma coluna unindo partido e candidato, sem excluir as originais
- ▶ Bônus: use `group_by` e `summarise` para identificar qual candidato tem a coligação com menor média de concordância e qual candidato tem a coligação com maior soma da proporção total de assentos.

survyr

# Referências

- ▶ Survey <http://r-survey.r-forge.r-project.org/survey/>
- ▶ srvyr <https://cran.r-project.org/web/packages/srvyr/vignettes/srvyr-vs-survey.html>

