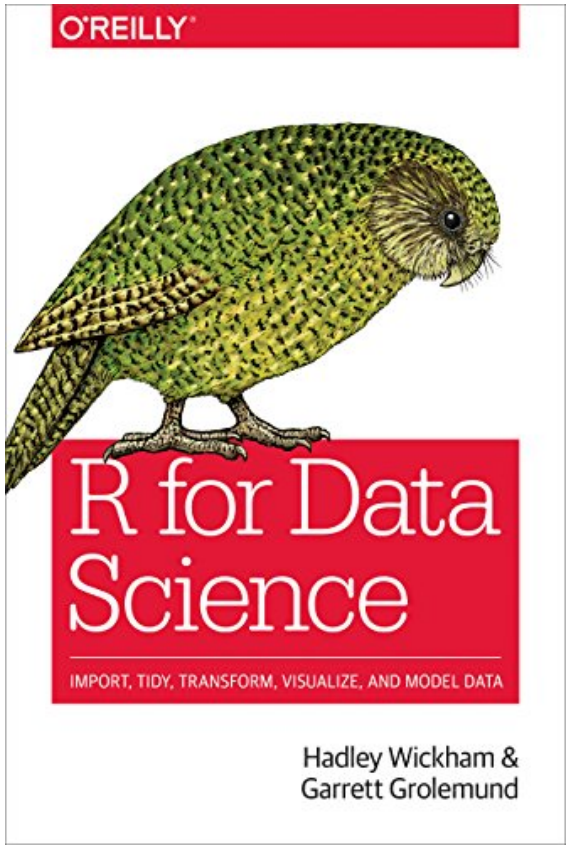
Conhecendo o Tidyverse

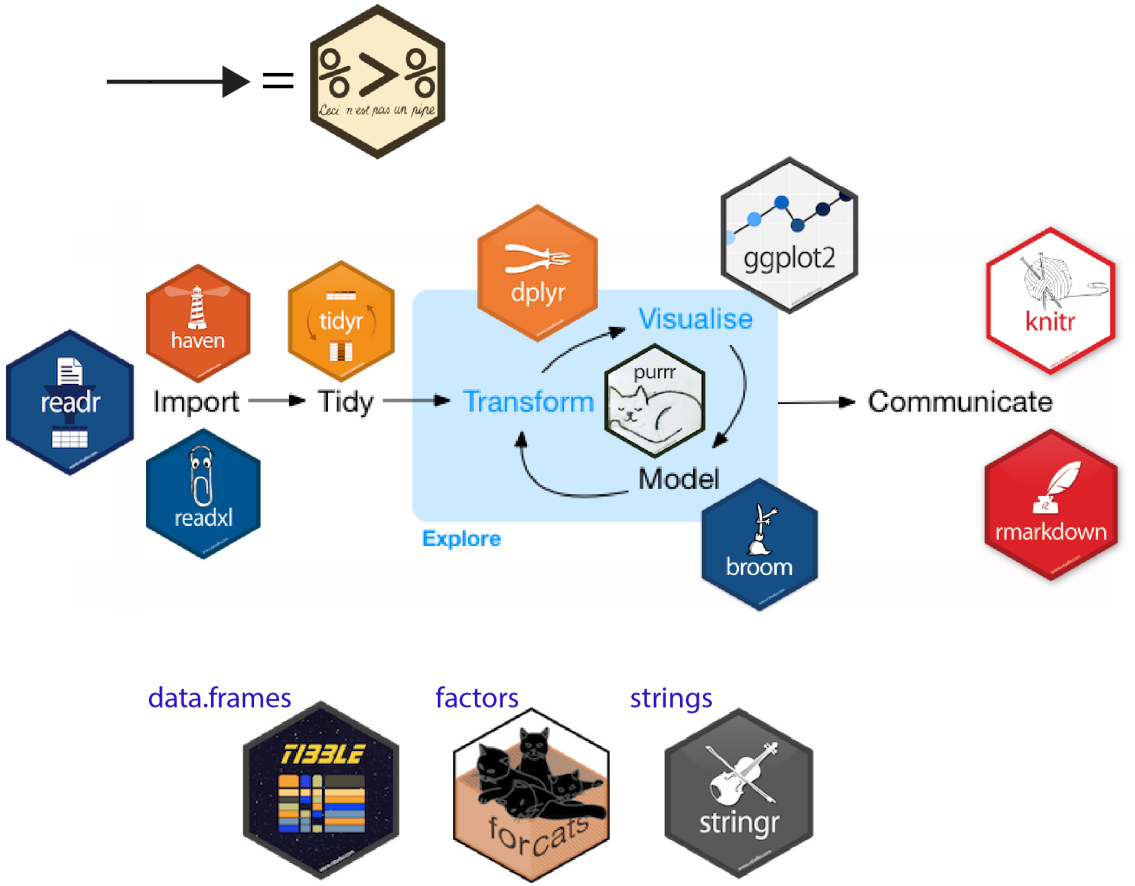
stringr, forcats e lubridate

Fernando de Pol Mayer (LEG/DEST/UFPR)
2022-03-29

R for Data Science



R for Data Science, a principal referência sobre o emprego da linguagem R em ciência de dados.



Workflow de ciência de dados com o {tidyverse}. Fonte: https://oliviergimenez.github.io/intro_tidyverse/#7

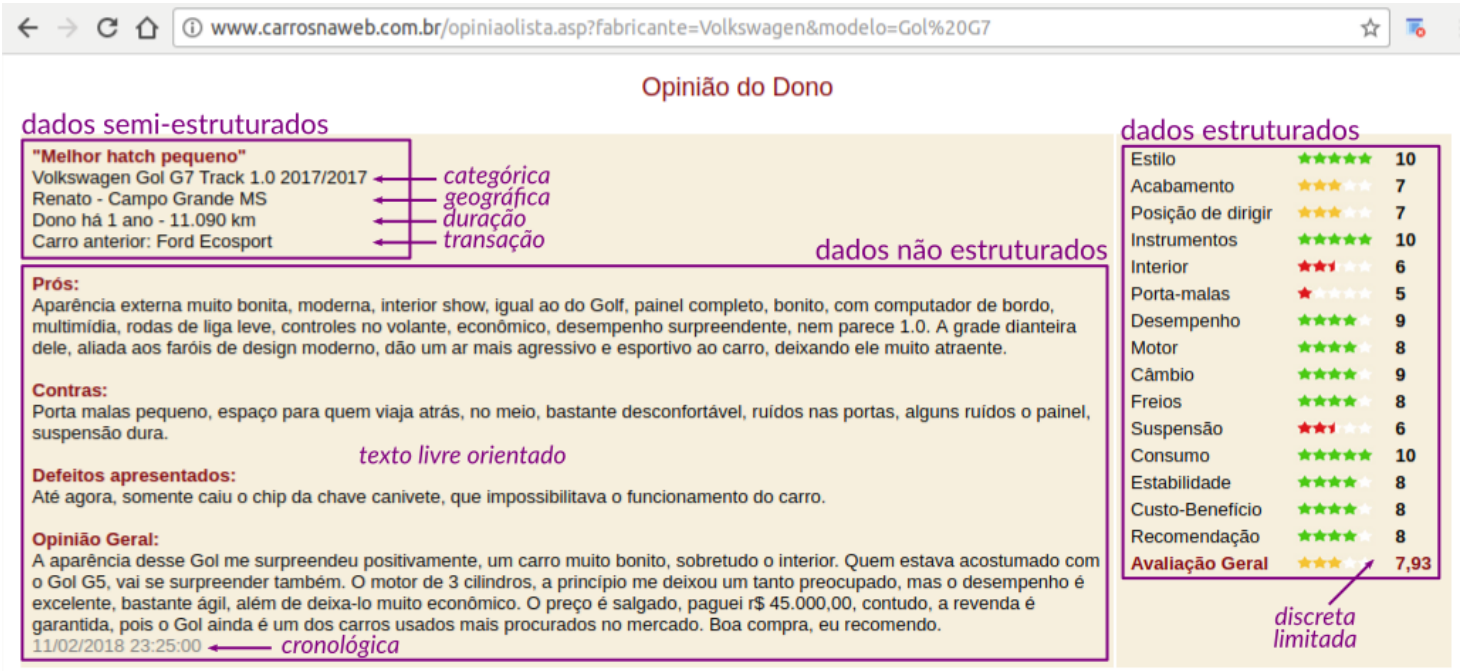
{stringr}

Usos de regex

- Padrões em cadeias de caracteres.
- Detectar.
- Extrair.
- Eliminar.
- Substituir.

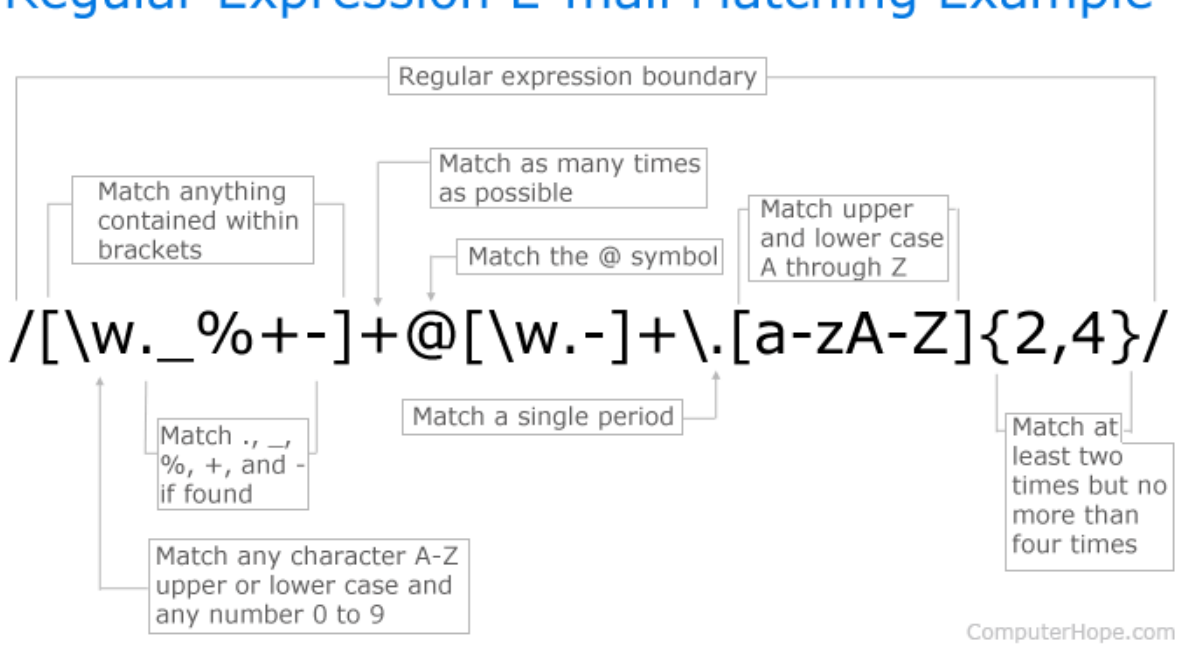
Exemplos:

- Telefone.
- Email.
- Data.
- CPF/CNPJ.
- Endereço.



Avaliação online de um veículo do site carros na web.
Fonte: https://github.com/leg-ufpr/hackathon/blob/master/datasets.md.

Expressões regulares



Expressão regular para verificar emails.
Fonte: https://paulvanderlaken.com/2017/10/03/regular-expressions-in-r-part-1-introduction-and-base-r-functions/.

Regex online

- https://regexr.com/.
- https://regex101.com/.
- https://www.regextester.com/.
- https://regex-generator.olafneumann.org/.

Conteúdo do pacote

- Página do pacote: https://stringr.tidyverse.org/.
- Vinheta de uso: https://stringr.tidyverse.org/articles/regular-expressions.html.
- Cheatsheet: https://github.com/rstudio/cheatsheets/raw/master/strings.pdf

Principais funções

- `str_detect()`: detectar padrões para usar em filtros.
- `str_remove*()`: remove um padrão.
- `str_extract*()`: extrai um padrão.
- `str_replace*()`: Substitui um padrão.
- `str_c()`: concatena strings.

A maioria das funções do pacote tem o prefixo `str_`.

```
%>%             str_glue_data   str_starts
boundary        str_interp      str_sub
coll            str_length      str_sub<-
fixed           str_locate      str_subset
fruit           str_locate_all  str_to_lower
invert_match    str_match       str_to_sentence
regex           str_match_all   str_to_title
sentences       str_order       str_to_upper
str_c           str_pad         str_trim
str_conv        str_remove      str_trunc
str_count       str_remove_all  str_view
str_detect      str_replace     str_view_all
str_dup         str_replace_all str_which
str_ends        str_replace_na  str_wrap
str_extract     str_sort        word
str_extract_all str_split       words
str_flatten     str_split_fixed
str_glue        str_squish
```

{forcats}

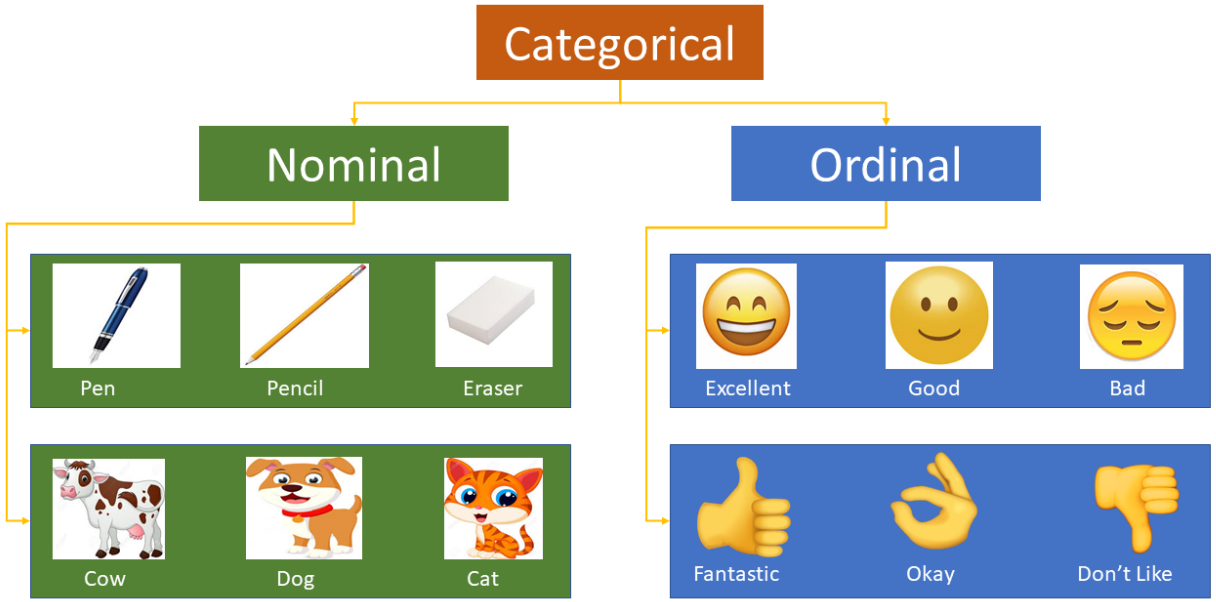
O que são fatores?

Variáveis qualitativas

- Nominais ou ordinais.
- São propriedades ou atributos que expressam qualidade.
- Nominais: a ordem não importa.
- Ordinais: existe ordenação natural.

Exemplos:

- Nota para avaliação de produto/serviço.
- Escala likert.
- Unidades geográficas como UF.
- Unidades cronológicas como meses.
- Tipo de produto/departamento.



Tipos de variáveis qualitativas. Fonte: https://morioh.com/p/c7649e4e463f.

Conteúdo do pacote

- Página do pacote: https://forcats.tidyverse.org/.
- Cheatsheet: https://github.com/rstudio/cheatsheets/blob/main/factors.pdf

Principais funções

- `fct_reorder()`: **reordena** os níveis baseado em outra variável. Ex: reordenar UFs pelo número de habitantes.
- `fct_relevel()`: troca os níveis do fator de **posição**. Ex: passar o nível "desconhecido" para última posição.
- `fct_relabel()`: para **rerotular** os níveis. Ex: São Paulo → SP, Mato Grosso do Sul → MS.
- `fct_recode()`: para **recodificar** os níveis do fator. Ex: {PR, SC, RS} → Sul, {MS, MT, GO} → Centro-oeste.
- `fct_lump*()`: **aglutinar** níveis conforme critérios. Ex: agrupar os níveis de menor ocorrência sob o rótulo "outros".

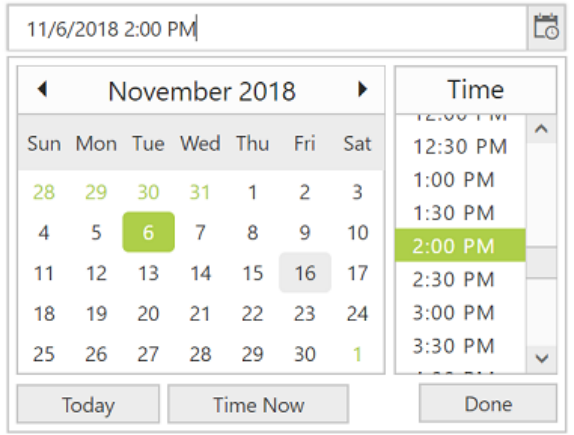
A maioria das funções do pacote tem o prefixo `fct_`.

```
%>%              fct_other
as_factor        fct_recode
fct_anon         fct_relabel
fct_c            fct_relevel
fct_collapse     fct_reorder
fct_count        fct_reorder2
fct_cross        fct_rev
fct_drop         fct_shift
fct_expand       fct_shuffle
fct_explicit_na  fct_unify
fct_infreq       fct_unique
fct_inorder      first2
fct_inseq        gss_cat
fct_lump         last2
fct_lump_lowfreq lvls_expand
fct_lump_min     lvls_reorder
fct_lump_n       lvls_revalue
fct_lump_prop    lvls_union
fct_match
```

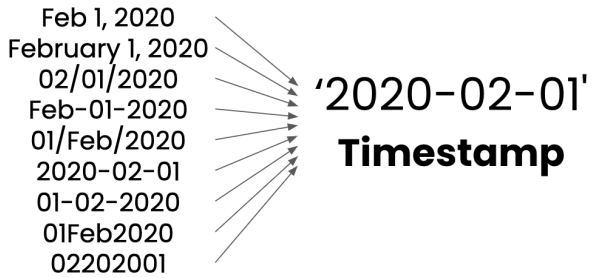
{lubridate}

O que são variáveis de data e tempo?

- Datas e tempos registram o instante, duração, período de eventos.
- São complicados por serem irregulares:
  - Convenções de representação dependem do idioma.
  - Existem os fusos horários e horários de verão.
  - Anos podem ser bissextos.
  - Existem feriados que mudam com os países e estados.
  - Meses de tamanhos diferentes.
  - Horas e minutos não usam base decimal.
- Recursos consistentes e acessíveis para trabalhar com datas e tempos são muito bem-vindos.
- Página do pacote: https://lubridate.tidyverse.org/.
- Cheatsheet: https://github.com/rstudio/cheatsheets/blob/main/lubridate.pdf



Seletor de data e hora.



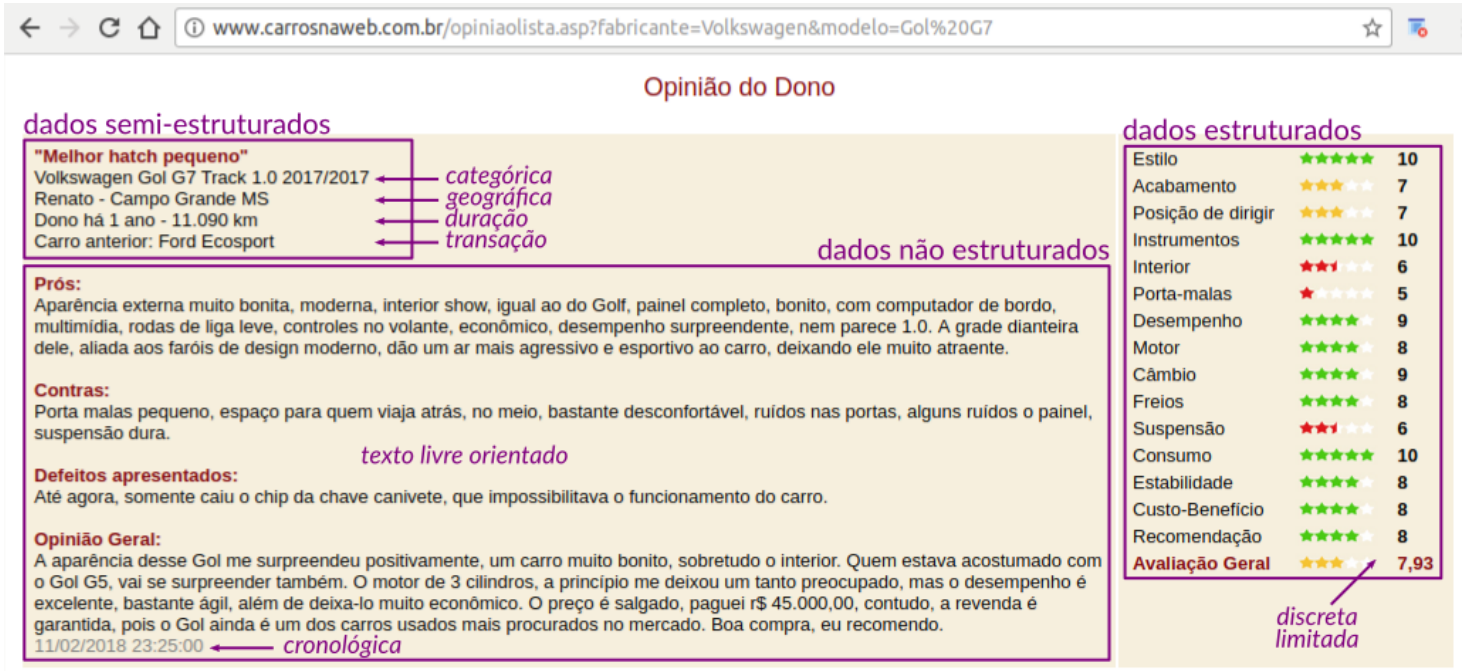
Tipos de representação de datas. Fonte: https://www.dataindependent.com/pandas/pandas-to-datetime/.

Conteúdo do pacote

```
%--%                dmilliseconds     hours             leap_year         period             tz<-
%m-%                dminutes          int_aligns        local_time        period_to_seconds  union
%m+%                dmonths           int_diff          make_date         picoseconds        wday
%within%            dmy               int_end           make_datetime     pm                 wday<-
add_with_rollback   dmy_h             int_end<-         make_difftime     POSIXct            week
am                  dmy_hm            int_flip          mday              pretty_dates       week<-
Arith               dmy_hms           int_length        mday<-            qday               weeks
as_date             dnanoseconds      int_overlaps      mdy               qday<-             with_tz
as_datetime         dpicoseconds      int_shift         mdy_h             quarter            yday
as.difftime         dseconds          int_standardize   mdy_hm            reclass_date       yday<-
as.duration         dst               int_start         mdy_hms           reclass_timespan   ydm
as.interval         duration          int_start<-       microseconds      rollback           ydm_h
as.period           dweeks            intersect         milliseconds      rollbackward       ydm_hm
ceiling_date        dyears            interval          minute            rollforward        ydm_hms
Compare             dym               is.Date           minute<-          round_date         year
cyclic_encoding     epiweek           is.difftime       minutes           second             year<-
date                epiyear           is.duration       month             second<-           years
Date                fast_strptime     is.instant        month<-           seconds            ym
date_decimal        fit_to_timeline   is.interval       ms                seconds_to_period  ymd
date<-              floor_date        is.period         my                semester           ymd_h
day                 force_tz          is.POSIXct        myd               setdiff            ymd_hm
day<-               force_tzs         is.POSIXlt        NA_Date_          show               ymd_hms
days                format_ISO8601    is.POSIXt         NA_POSIXct_       stamp              yq
days_in_month       guess_formats     is.timepoint      nanoseconds       stamp_date
ddays               hm                is.timespan       now               stamp_time
decimal_date        hms               isoweek           origin            time_length
dhours              hour              isoyear           parse_date_time   today
dmicroseconds       hour<-            lakers            parse_date_time2  tz
```

Conderações finais

- Dados como datas e fatores são muito frequentes, bem como a necessidade de tratar cadeias de caracteres.
- O {stringr} oferece recursos para emprego de regex na manipulação de campos de texto.
- O {forcats} reúne várias funções de conveniência para trabalhar com fatores.
- O {lubridate} traz funcionalidades para trabalhar variáveis de datas.



Avaliação online de um veículo do site carros na web.
Fonte: https://github.com/leg-ufpr/hackathon/blob/master/datasets.md.