EMISSÃO DE CO_2 DO SOLO EM ÁREAS AGRÍCOLAS: ABORDAGEM EM APRENDIZADO DE MÁQUINA ESTATÍSTICO

Autor: Prof. Dr. Alan Rodrigo Panosso (https://www.fcav.unesp.br/#!/alan)

E-mail: alan.panosso@unesp.br (mailto:alan.panosso@unesp.br)

Departamento de Engenharia e Ciências Exatas

UNESP - Câmpus de Jaboticabal

Objetivo

O objetivo do repositório `tese-fco2-ml-2023` é promover a transparência, a reprodutibilidade e a colaboração em pesquisa. Você é incentivado a explorar o código-fonte, utilizar os dados e contribuir com melhorias, se desejar. Sinta-se à vontade para entrar em contato caso tenha alguma dúvida ou precise de mais informações sobre minha pesquisa.

Contribuições

Contribuições são bem-vindas! Se você deseja colaborar com melhorias nos códigos, correções de erros ou qualquer outro aprimoramento, sinta-se à vontade para abrir uma solicitação de `pull request` .

Licença

Base de dados

Apresentação do pacote `fco2r` construído para facilitar a divulgação e análise dos resultados obtidos ao longo de mais de 20 anos de ensaios em campo. Este pacote, permite a visualização dos dados, a execução de análises estatísticas avançadas e a geração de gráficos interativos para tornar os resultados mais acessíveis e compreensíveis para a comunidade científica.

Instalação

Você pode instalar uma versão de desenvolvimento do pacote `fco2r` a partir do GitHub (https://github.com/) com os seguintes comandos:

```
# install.packages("devtools")
# devtools::install_github("arpanosso/fco2r")
```

Problemas na instalação:

Possíveis problemas na instalação do pacote podem ser sanados com os seguintes comandos:

```
# Sys.getenv("GITHUB_PAT")
# Sys.unsetenv("GITHUB_PAT")
# Sys.getenv("GITHUB_PAT")
```

Carregando os pacotes

```
library(fco2r)
library(tidyverse)
library(patchwork)
library(ggspatial)
library(readxl)
library(skimr)
library(tidymodels)
library(ISLR)
library(modeldata)
library(vip)
library(ggpubr)
```

Conhecendo a base de dados de emissão de CO_2 do solo

Base proveniente de ensaios de campo.

```
help(data_fco2)
glimpse(data_fco2)
#> Rows: 15,397
#> Columns: 39
#> $ experimento       <chr> "Espacial", "Espacial", "Espacial", "Espacial", "Esp…
#> $ data              <date> 2001-07-10, 2001-07-10, 2001-07-10, 2001-07-10, 200…
#> $ manejo            <chr> "convencional", "convencional", "convencional", "con…
#> $ tratamento        <chr> "AD_GN", "AD_GN", "AD_GN", "AD_GN", "AD_GN", "AD_GN"…
#> $ revolvimento_solo <lgl> FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FALSE, FAL…
#> $ data_preparo      <date> 2001-07-01, 2001-07-01, 2001-07-01, 2001-07-01, 200…
#> $ conversao         <date> 1970-01-01, 1970-01-01, 1970-01-01, 1970-01-01, 197…
#> $ cobertura         <lgl> TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE, TRUE…
#> $ cultura           <chr> "milho_soja", "milho_soja", "milho_soja", "milho_soj…
#> $ x                 <dbl> 0, 40, 80, 10, 25, 40, 55, 70, 20, 40, 60, 10, 70, 3…
#> $ y                 <dbl> 0, 0, 0, 10, 10, 10, 10, 10, 20, 20, 20, 25, 25, 30,…
#> $ longitude_muni    <dbl> 782062.7, 782062.7, 782062.7, 782062.7, 782062.7, 78…
#> $ latitude_muni     <dbl> 7647674, 7647674, 7647674, 7647674, 7647674, 7647674…
#> $ estado            <chr> "SP", "SP", "SP", "SP", "SP", "SP", "SP", "SP", "SP"…
#> $ municipio         <chr> "Jaboticabal", "Jaboticabal", "Jaboticabal", "Jaboti…
#> $ ID                <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 1…
#> $ prof              <chr> "0-0.1", "0-0.1", "0-0.1", "0-0.1", "0-0.1", "0-0.1"…
#> $ FCO2              <dbl> 1.080, 0.825, 1.950, 0.534, 0.893, 0.840, 1.110, 1.8…
#> $ Ts                <dbl> 18.73, 18.40, 19.20, 18.28, 18.35, 18.47, 19.10, 18.…
#> $ Us                <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ pH                <dbl> 5.1, 5.1, 5.8, 5.3, 5.5, 5.7, 5.6, 6.4, 5.3, 5.8, 5.…
#> $ MO                <dbl> 20, 24, 25, 23, 23, 21, 26, 23, 25, 24, 26, 20, 25, …
#> $ P                 <dbl> 46, 26, 46, 78, 60, 46, 55, 92, 55, 60, 48, 71, 125,…
#> $ K                 <dbl> 2.4, 2.2, 5.3, 3.6, 3.4, 2.9, 4.0, 2.3, 3.3, 3.6, 4.…
#> $ Ca                <dbl> 25, 30, 41, 27, 33, 38, 35, 94, 29, 36, 37, 29, 50, …
#> $ Mg                <dbl> 11, 11, 25, 11, 15, 20, 16, 65, 11, 17, 15, 11, 30, …
#> $ H_Al              <dbl> 31, 31, 22, 28, 27, 22, 22, 12, 31, 28, 28, 31, 18, …
#> $ SB                <dbl> 38.4, 43.2, 71.3, 41.6, 50.6, 60.9, 55.0, 161.3, 43.…
#> $ CTC               <dbl> 69.4, 74.2, 93.3, 69.6, 77.9, 82.9, 77.0, 173.3, 74.…
#> $ V                 <dbl> 55, 58, 76, 60, 65, 73, 71, 93, 58, 67, 67, 58, 82, …
#> $ Ds                <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ Macro             <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ Micro             <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ VTP               <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ PLA               <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ AT                <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ SILTE             <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ ARG               <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
#> $ HLIFS             <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, …
```

Vamos conhecer, um pouco mais a nossa base de dados.

```
skimr::skim(data_fco2)
```

Name	data_fco2
Number of rows	15397
Number of columns	39
_______________________	
Column type frequency:	
character	7
Date	3
logical	2
numeric	27
________________________	
Group variables	None

Data summary

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
experimento	0	1	8	8	0	2	0
manejo	0	1	6	15	0	10	0
tratamento	0	1	2	10	0	21	0

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
cultura	0	1	4	14	0	11	0
estado	0	1	2	2	0	2	0
municipio	0	1	7	20	0	6	0
prof	0	1	5	7	0	2	0

Variable type: Date

skim_variable	n_missing	complete_rate	min	max	median	n_unique
data	0	1	2001-07-10	2019-12-01	2014-07-12	205
data_preparo	0	1	1986-03-01	2019-04-01	2002-01-01	14
conversao	0	1	1970-01-01	2009-07-03	1986-03-01	11

Variable type: logical

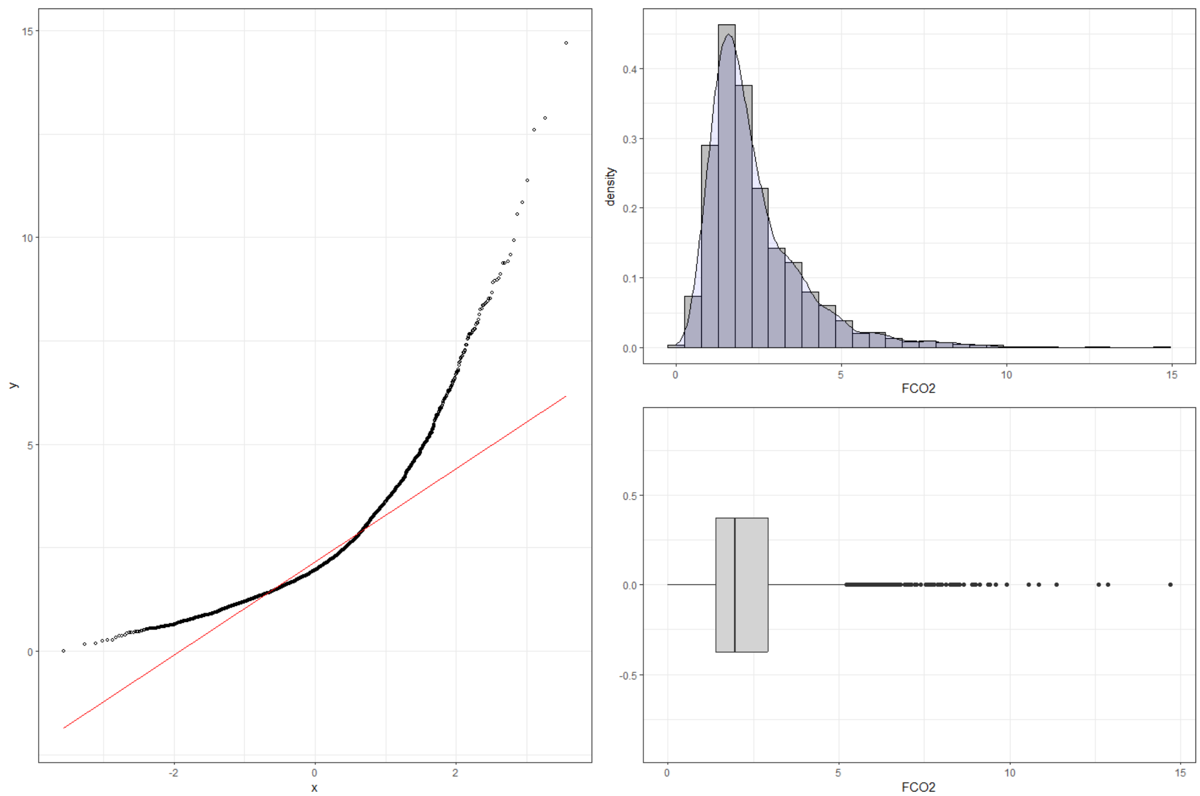
skim_variable	n_missing	complete_rate	mean	count
revolvimento_solo	0	1	0	FAL: 15397
cobertura	0	1	1	TRU: 15397

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	p25	p50	p75	p100	hist
x	0	1.00	1392083.56	2923710.70	0.00	0.00	30.00	100.00	7749472.16	
y	0	1.00	495854.97	1722529.75	0.00	0.00	27.00	80.00	7630525.47	
longitude_muni	0	1.00	1067926.05	1796771.47	456798.63	458447.46	458447.46	792043.56	7638196.06	
latitude_muni	0	1.00	7231328.21	1754220.76	795907.06	7635356.70	7749398.84	7749821.85	7758831.37	
ID	0	1.00	40.52	31.52	1.00	13.00	35.00	60.00	141.00	
FCO2	110	0.99	2.78	2.08	-3.42	1.30	2.16	3.75	46.93	
Ts	317	0.98	21.84	6.76	1.00	19.33	22.50	26.15	195.63	
Us	1754	0.89	16.31	8.93	0.00	10.00	14.06	22.00	89.00	
pH	2802	0.82	4.64	1.13	3.50	4.00	4.50	5.15	52.00	
MO	1355	0.91	21.59	12.60	1.35	12.00	23.00	29.00	61.26	
P	1355	0.91	20.95	24.74	1.00	6.00	15.48	27.36	253.00	
K	1348	0.91	2.40	2.21	0.03	0.90	1.70	3.40	34.00	
Ca	1376	0.91	17.20	14.57	1.10	6.00	11.00	26.00	94.00	
Mg	1376	0.91	10.13	5.65	0.32	7.00	10.00	13.00	65.00	
H_Al	1362	0.91	46.89	29.38	0.00	26.00	42.29	72.00	121.00	
SB	1376	0.91	29.69	20.10	1.54	15.60	23.80	42.00	161.30	
CTC	1369	0.91	77.10	32.99	4.62	59.23	83.40	103.20	173.30	
V	1383	0.91	41.68	20.05	4.96	22.00	43.00	58.00	100.00	
Ds	3284	0.79	1.38	0.17	0.88	1.24	1.38	1.52	1.86	
Macro	3277	0.79	8.55	7.85	-45.30	0.15	8.13	13.64	49.77	
Micro	3298	0.79	25.30	17.13	0.07	0.37	33.86	38.30	52.42	
VTP	3298	0.79	42.34	15.65	-4.68	40.81	46.25	51.32	87.80	
PLA	3438	0.78	29.57	11.80	-47.30	21.27	32.41	38.15	79.80	
AT	8083	0.48	1013.33	1358.81	11.72	236.00	593.62	816.00	4542.73	
SILTE	8048	0.48	229.26	336.37	1.26	50.87	73.65	188.00	1395.00	
ARG	8055	0.48	995.41	1560.32	27.19	173.27	403.69	609.50	5244.76	
HLIFS	10872	0.29	14590.11	17253.55	158.39	1110.15	2409.80	29707.78	84692.90	

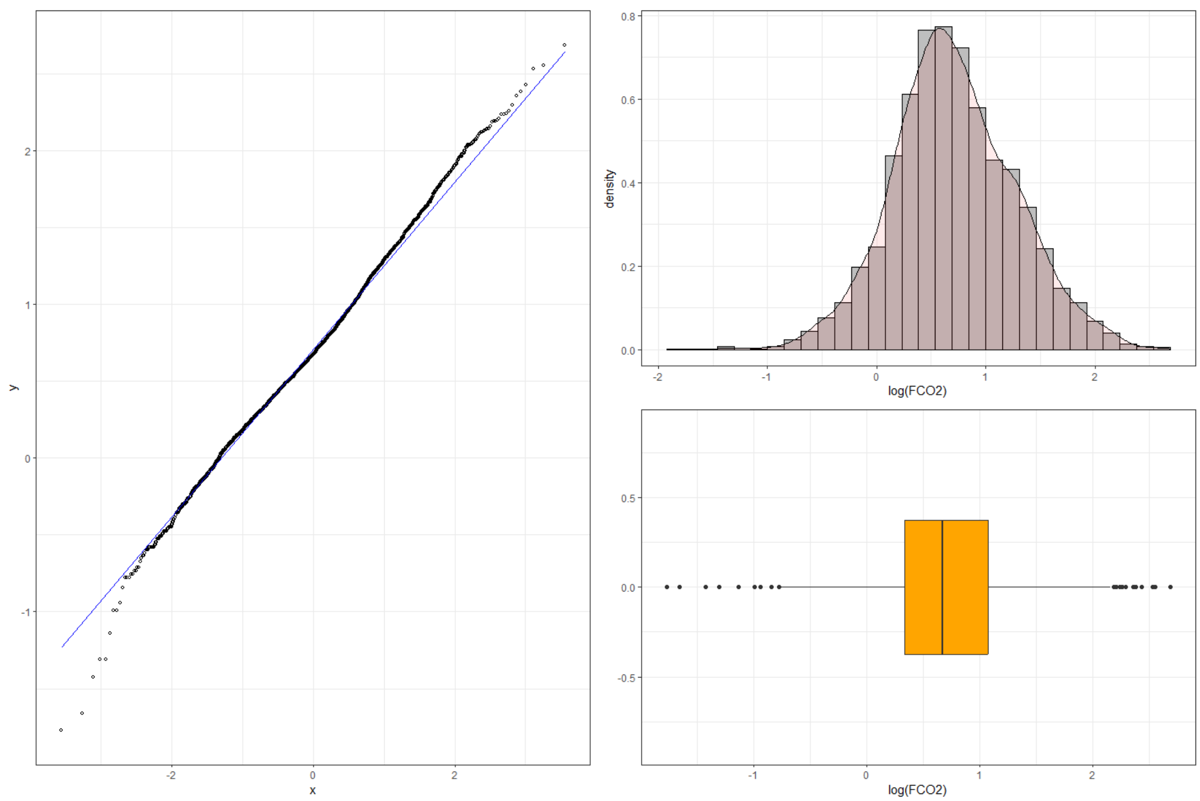
Alguns gráficos a respeito de nossa variável alvo, emissão de CO_2 do solo (FCO_2).

```
theme_set(theme_bw())
fco2_histograma <- data_fco2 %>%
  drop_na() %>%
  ggplot(aes(x=FCO2, y=..density..)) +
  geom_histogram(col="black",fill="gray") +
  geom_density(fill="blue",alpha=.08)

fco2_boxplot <- data_fco2 %>%
  drop_na() %>%
  ggplot(aes(x=FCO2)) +
  geom_boxplot(fill="lightgray") +
  coord_cartesian(ylim=c(-.9,.9))

fco2_qqplot <- data_fco2 %>%
  drop_na() %>%
  ggplot(aes(sample=FCO2)) +
  stat_qq(shape=1,size=1,color="black")+
  stat_qq_line(col="red")

fco2_qqplot | (fco2_histograma)/(fco2_boxplot)
```



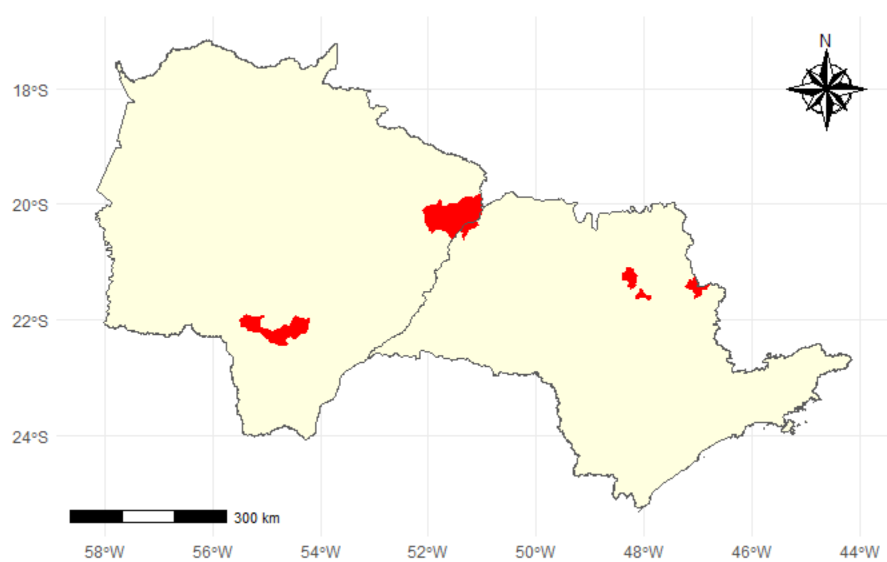
Aplicando a transformação logarítmica nos dados de FCO_2

```
ggplot2::theme_set(theme_bw())

fco2_histograma <- data_fco2 %>%
  drop_na() %>%
  ggplot(aes(x=log(FCO2), y=..density..)) +
  geom_histogram(col="black",fill="gray") +
  geom_density(fill="red",alpha=.08)

fco2_boxplot <- data_fco2 %>%
  drop_na() %>%
  ggplot(aes(x=log(FCO2))) +
  geom_boxplot(fill="orange") +
  coord_cartesian(ylim=c(-.9,.9))

fco2_qqplot <- data_fco2 %>%
  drop_na() %>%
  ggplot(aes(sample=log(FCO2))) +
  stat_qq(shape=1,size=1,color="black")+
  stat_qq_line(col="blue")

fco2_qqplot | (fco2_histograma)/(fco2_boxplot)
```




```
# brasil_geobr <- geobr::read_country()
# estados <- read_state(code_state = "all")
# write_rds(estados,"data/estados.rds")
# write_rds(brasil_geobr,"data/brasil_geobr.rds")
estados <- read_rds("data/estados.rds")
```

```r
# muni <- read_municipality()
# write_rds(muni,"data/municipios.rds")
muni <- read_rds("data/municipios.rds")

sp <- muni %>%
  filter(abbrev_state == "SP")

ms <- muni %>%
  filter(abbrev_state == "MS")

sp_ms <- muni %>%
  filter(abbrev_state == "SP" | abbrev_state == "MS")

fsp<-if_else(sp$name_muni == "Jaboticabal" |
             sp$name_muni == "Guariba" |
             sp$name_muni == "Padrópolis" |
             sp$name_muni == "Rincão"|
             sp$name_muni == "Mococa"|
             sp$name_muni == "Ilha Solteira"
             ,"red","lightyellow")

fms<-if_else(ms$name_muni == "Aparecida Do Taboado" |
             ms$name_muni == "Selvíria"|
             ms$name_muni == "Dourados"
             ,"red","lightyellow")

sp_ <- estados %>%
      filter(abbrev_state == "SP")

ms_ <- estados %>%
      filter(abbrev_state == "MS")

ggplot(sp_ms) +
  geom_sf(fill="lightyellow")+
  theme_minimal() +
  annotation_scale(location="bl")+
  annotation_north_arrow(location="tr",
                         style = north_arrow_nautical(),
                         width = unit(2,"cm"),
                         height = unit(2,"cm")) +
  geom_sf(data= sp, fill=fsp,col=fsp) +
  geom_sf(data=sp_,fill="transparent") +
  geom_sf(data= ms, fill=fms,col=fms) +
  geom_sf(data=ms_,fill="transparent")
```



Conhecendo a base de dados de concentração de CO_2 atmosférico, oriundo do

sensor NASA-OCO2.

```
help(oco2_br)
glimpse(oco2_br)
#> Rows: 37,387
#> Columns: 18
#> $ longitude                                              <dbl> -70.5, -…
#> $ longitude_bnds                                         <chr> "-71.0:-…
#> $ latitude                                               <dbl> -5.5, -4…
#> $ latitude_bnds                                          <chr> "-6.0:-5…
#> $ time_yyyymmddhhmmss                                    <dbl> 2.014091…
#> $ time_bnds_yyyymmddhhmmss                               <chr> "2014090…
#> $ altitude_km                                            <dbl> 3307.8, …
#> $ alt_bnds_km                                            <chr> "0.0:661…
#> $ fluorescence_radiance_757nm_uncert_idp_ph_sec_1_m_2_sr_1_um_1 <dbl> 7.272876…
#> $ fluorescence_radiance_757nm_idp_ph_sec_1_m_2_sr_1_um_1       <dbl> 2.537127…
#> $ xco2_moles_mole_1                                      <dbl> 0.000394…
#> $ aerosol_total_aod                                      <dbl> 0.148579…
#> $ fluorescence_offset_relative_771nm_idp                 <dbl> 0.016753…
#> $ fluorescence_at_reference_ph_sec_1_m_2_sr_1_um_1       <dbl> 2.615319…
#> $ fluorescence_radiance_771nm_idp_ph_sec_1_m_2_sr_1_um_1       <dbl> 3.088582…
#> $ fluorescence_offset_relative_757nm_idp                 <dbl> 0.013969…
#> $ fluorescence_radiance_771nm_uncert_idp_ph_sec_1_m_2_sr_1_um_1 <dbl> 5.577878…
#> $ XCO2                                                   <dbl> 387.2781…
```

Breve resumo do banco de dados de X_{CO2}

```
skimr::skim(oco2_br)
```

Name	oco2_br
Number of rows	37387
Number of columns	18
________________________	
Column type frequency:	
character	4
numeric	14
________________________	
Group variables	None

Data summary

Variable type: character

skim_variable	n_missing	complete_rate	min	max	empty	n_unique	whitespace
longitude_bnds	0	1	11	11	0	39	0
latitude_bnds	0	1	7	11	0	38	0
time_bnds_yyyymmddhhmmss	0	1	29	29	0	1765	0
alt_bnds_km	0	1	11	20	0	64	0

Variable type: numeric

skim_variable	n_missing	complete_rate	mean	sd	p0	
longitude	0	1	-5.120000e+01	8.280000e+00	-7.350000e+01	-5.65000
latitude	0	1	-1.179000e+01	7.850000e+00	-3.250000e+01	-1.75000
time_yyyymmddhhmmss	0	1	2.016952e+13	1.564571e+10	2.014091e+13	2.01602
altitude_km	0	1	3.123200e+03	1.108800e+02	2.555700e+03	3.05635
fluorescence_radiance_757nm_uncert_idp_ph_sec_1_m_2_sr_1_um_1	0	1	8.520719e+17	5.599367e+18	-9.999990e+05	6.32325
fluorescence_radiance_757nm_idp_ph_sec_1_m_2_sr_1_um_1	0	1	-1.358150e+18	1.946775e+20	-3.400736e+22	7.73515
xco2_moles_mole_1	0	1	0.000000e+00	0.000000e+00	0.000000e+00	0.00000
aerosol_total_aod	0	1	4.828100e+02	7.848572e+04	2.000000e-02	1.10000
fluorescence_offset_relative_771nm_idp	0	1	-4.814400e+02	2.193698e+04	-9.999990e+05	1.00000

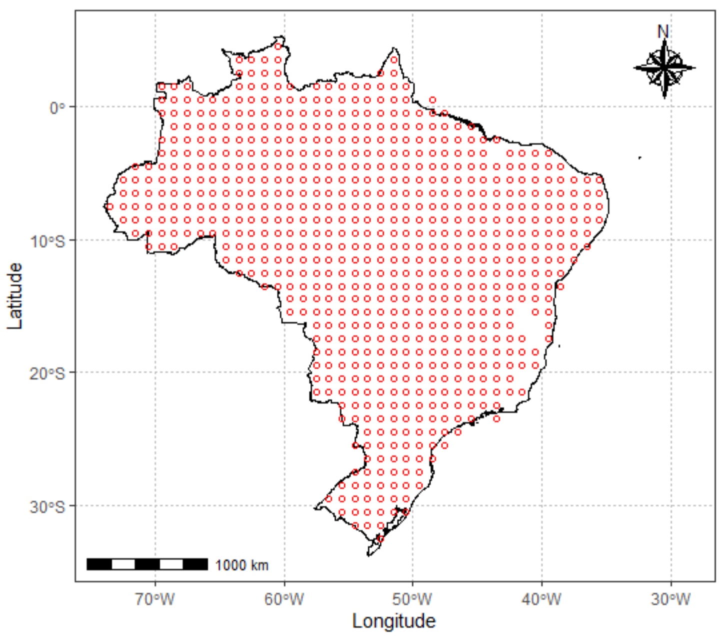
skim_variable	n_missing	complete_rate	mean	sd	p0
fluorescence_at_reference_ph_sec_1_m_2_sr_1_um_1	0	1	1.296932e+18	2.245185e+18	-8.394901e+19	2.01456
fluorescence_radiance_771nm_idp_ph_sec_1_m_2_sr_1_um_1	0	1	1.904438e+18	2.236381e+18	-8.453983e+19	9.69470
fluorescence_offset_relative_757nm_idp	0	1	-3.744400e+02	1.934763e+04	-9.999990e+05	1.00000
fluorescence_radiance_771nm_uncert_idp_ph_sec_1_m_2_sr_1_um_1	0	1	5.235574e+17	7.580471e+16	-9.999990e+05	4.69546
XCO2	0	1	3.858900e+02	3.120000e+00	3.383400e+02	3.84410

Manipulando a base `oco2_br` para criação das variáveis temporais e ajuste de unidade de xco2.

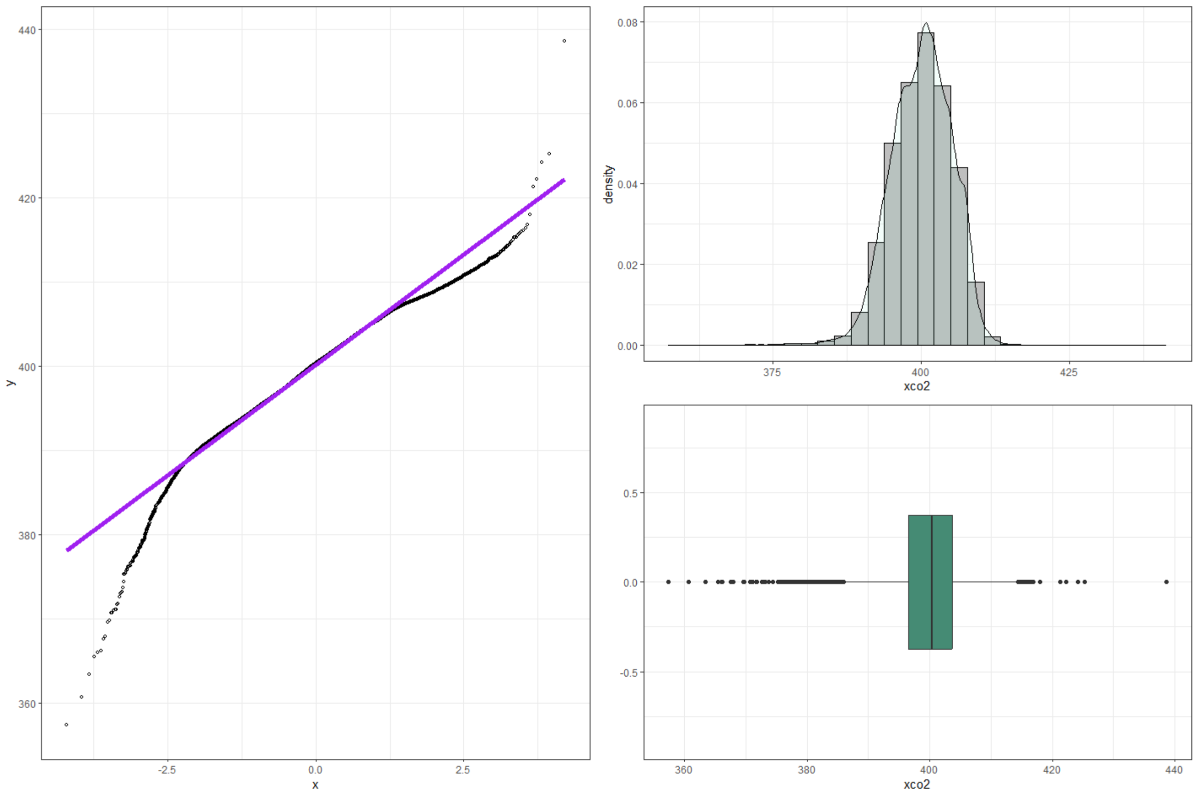
```
oco2_br <- oco2_br  %>%
        mutate(
           xco2 = xco2_moles_mole_1*1e06,
           data = ymd_hms(time_yyyymmddhhmmss),
           ano = year(data),
           mes = month(data),
           dia = day(data),
           dia_semana = wday(data))
```

Mapa das leituras do satélite OCO2-NASA

```
source("R/graficos.R")
brasil_geobr <- read_rds("data/brasil_geobr.rds")
brasil_geobr %>%
  ggplot() +
  geom_sf(fill="white", color="black",
          size=.15, show.legend = FALSE) +
  tema_mapa() +
  geom_point(data=oco2_br %>%
                          sample_n(20000) ,
             aes(x=longitude,y=latitude),
             shape=1,
             col="red",
             alpha=01)+
  labs(x="Longitude",y="Latitude")
```

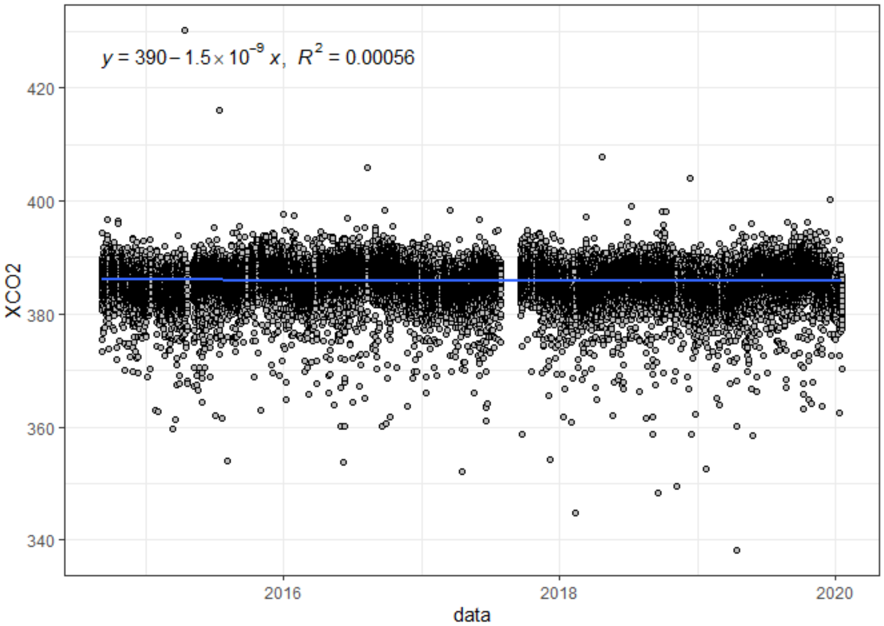


```
xco2_histograma <- oco2_br %>%
  drop_na() %>%
  ggplot(aes(x=xco2, y=..density..)) +
  geom_histogram(col="black",fill="gray") +
  geom_density(fill="aquamarine2",alpha=.08)

xco2_boxplot <- oco2_br %>%
  drop_na() %>%
  ggplot(aes(x=xco2)) +
  geom_boxplot(fill="aquamarine4") +
  coord_cartesian(ylim=c(-.9,.9))

xco2_qqplot <- oco2_br %>%
  drop_na() %>%
  ggplot(aes(sample=xco2)) +
  stat_qq(shape=1,size=1,color="black")+
  stat_qq_line(col="purple",lwd=2)

xco2_qqplot | (xco2_histograma)/(xco2_boxplot)
```

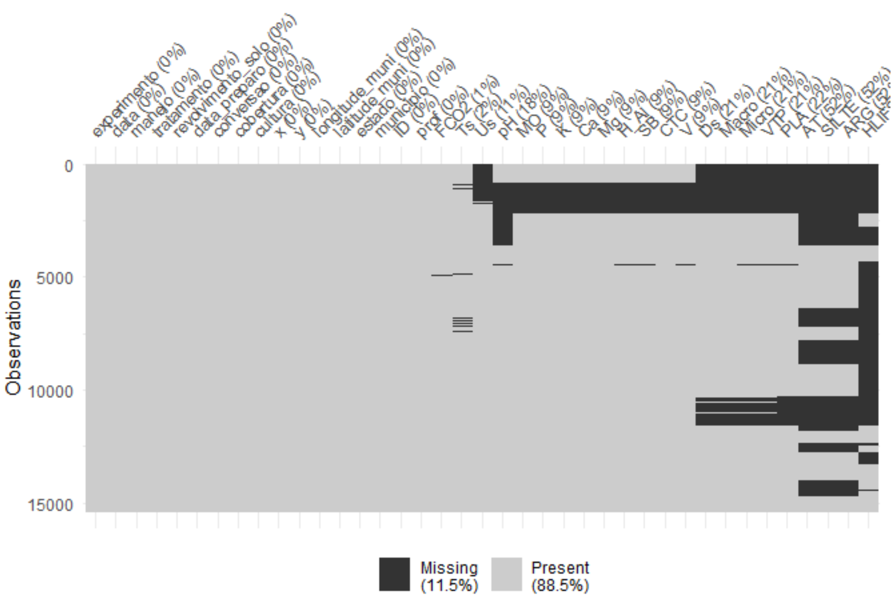


```
# Definindo o plano de multisession
future::plan("multisession")
```

```
oco2_br  %>%
  ggplot(aes(x=data,y=XCO2)) +
  geom_point(shape=21,color="black",fill="gray") +
  geom_smooth(method = "lm") +
  stat_regline_equation(ggplot2::aes(
  label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")))
```




```
visdat::vis_miss(data_fco2)
```



Listando as datas dos arquivos

```
lista_data_fco2 <- unique(data_fco2$data)
lista_data_oco2 <- unique(oco2_br$data)
datas_fco2 <- paste0(lubridate::year(lista_data_fco2),"-",lubridate::month(lista_data_fco2)) %>% unique()

datas_oco2 <- paste0(lubridate::year(lista_data_oco2),"-",lubridate::month(lista_data_oco2)) %>% unique()
datas <- datas_fco2[datas_fco2 %in% datas_oco2]
```

Chaves para mesclagem

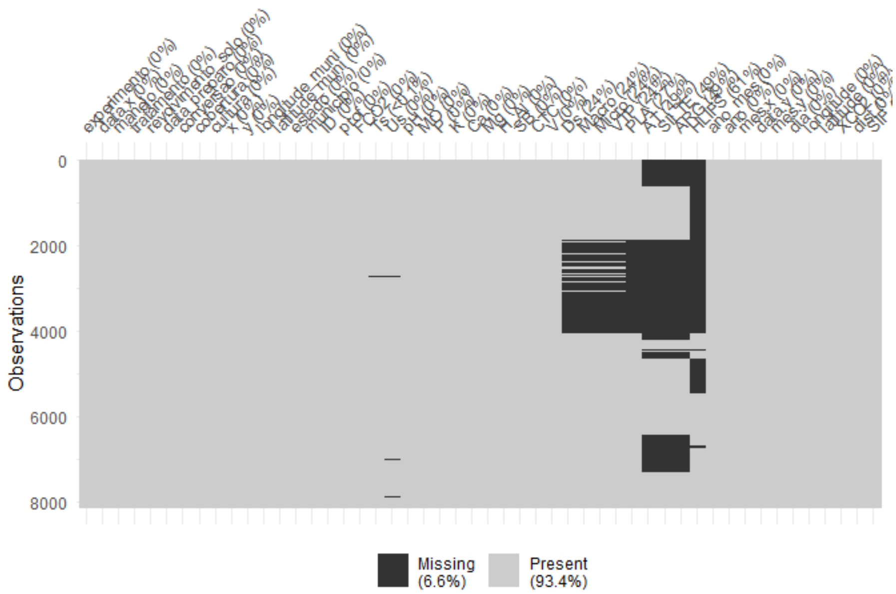
```
fco2 <- data_fco2 %>%
  mutate(ano_mes = paste0(lubridate::year(data),"-",lubridate::month(data))) %>%
  dplyr::filter(ano_mes %in% datas)

xco2 <- oco2_br %>%
  mutate(ano_mes=paste0(ano,"-",mes)) %>%
  dplyr::filter(ano_mes %in% datas)
```

Coordenadas das cidades

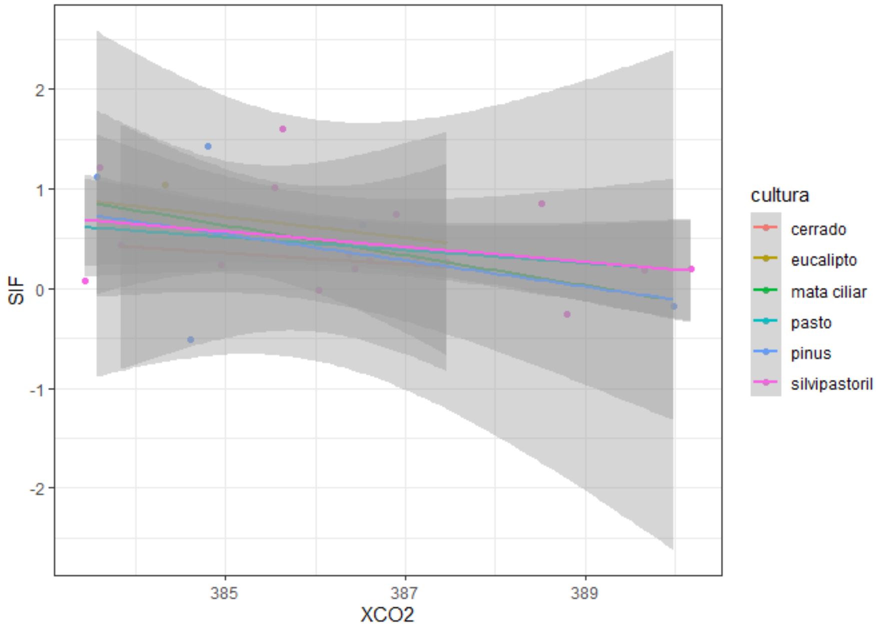
```
unique(xco2$ano_mes)[unique(xco2$ano_mes) %>% order()] ==
unique(fco2$ano_mes)[unique(fco2$ano_mes) %>% order()]
#>  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
#> [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
data_set <- left_join(fco2 %>%
            mutate(ano = lubridate::year(data),
                   mes = lubridate::month(data)
                   ),
          xco2 %>%
            select(data,mes,dia,longitude,latitude,XCO2,fluorescence_radiance_757nm_idp_ph_sec_1_m_2_sr_1_um_1,fl
uorescence_radiance_771nm_idp_ph_sec_1_m_2_sr_1_um_1, ano_mes), by = "ano_mes") %>%
  mutate(dist = sqrt((longitude-(-51.423519))^2+(latitude-(-20.362911))^2),
         SIF = (fluorescence_radiance_757nm_idp_ph_sec_1_m_2_sr_1_um_1*2.6250912*10^(-19)  + 1.5*fluorescence_rad
iance_771nm_idp_ph_sec_1_m_2_sr_1_um_1* 2.57743*10^(-19))/2)


data_set<-data_set %>%
  select(-fluorescence_radiance_757nm_idp_ph_sec_1_m_2_sr_1_um_1, -fluorescence_radiance_771nm_idp_ph_sec_1_m_2_s
r_1_um_1 )  %>%
  filter(dist <= .16, FCO2 <= 30 )
```

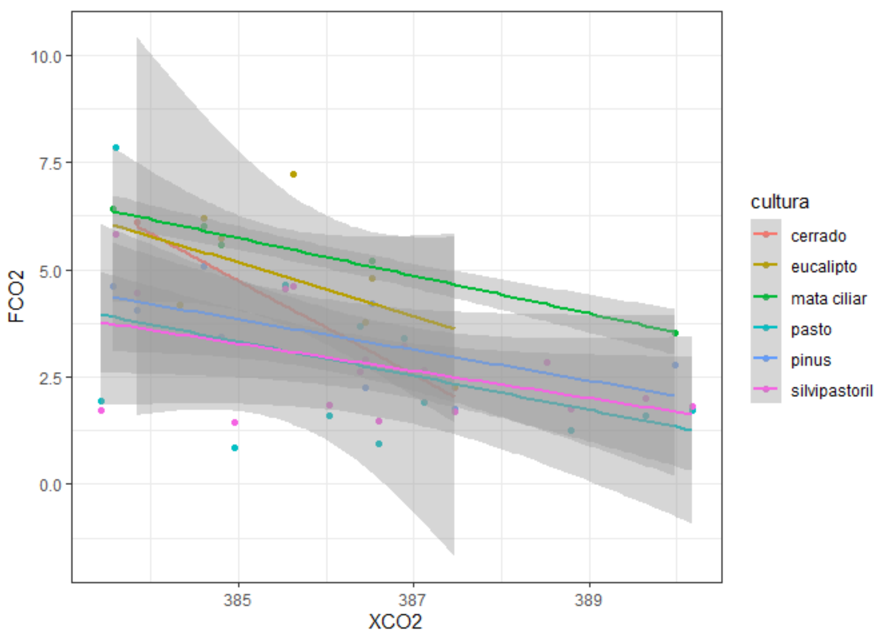
```
visdat::vis_miss(data_set)
```



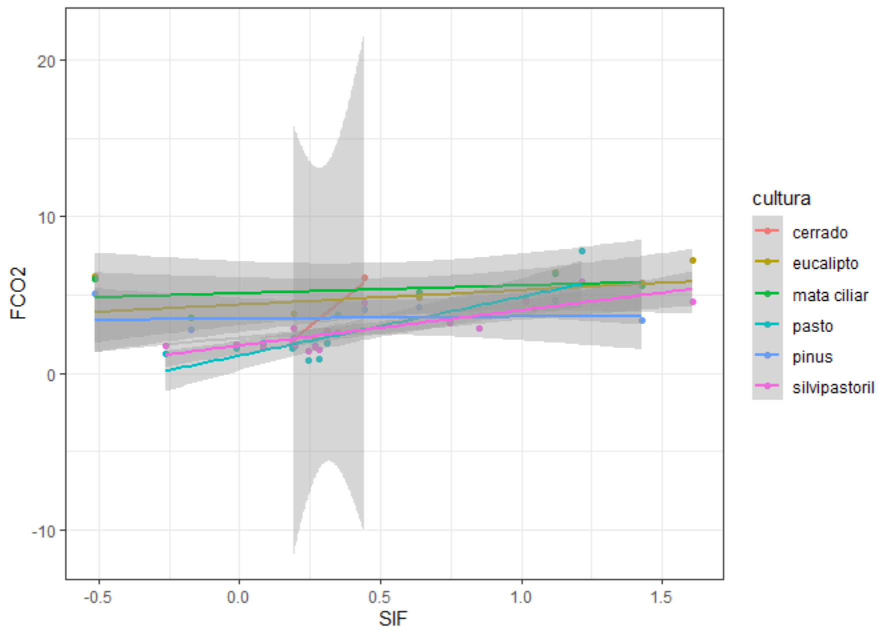

```
tab_medias <- data_set %>%
  # mutate(SIF = ifelse(SIF <=0, mean(data_set$SIF, na.rm=TRUE),SIF)) %>%
  group_by(ano_mes, cultura) %>%
  summarise(FCO2 = mean(FCO2, na.rm=TRUE),
            XCO2 = mean(XCO2, na.rm=TRUE),
            SIF = mean(SIF, na.rm=TRUE))

tab_medias %>%
  ggplot(aes(x=XCO2, y=SIF,color=cultura)) +
  geom_point()+
  geom_smooth(method = "lm")+
  theme_bw()
```




```
tab_medias %>%
  ggplot(aes(x=XCO2, y=FCO2,col=cultura)) +
  geom_point()+
  geom_smooth(method = "lm")+
  theme_bw()
```




```
tab_medias %>%
  ggplot(aes(y=FCO2, x=SIF, color=cultura)) +
  geom_point()+
  geom_smooth(method = "lm") +
  theme_bw()
```



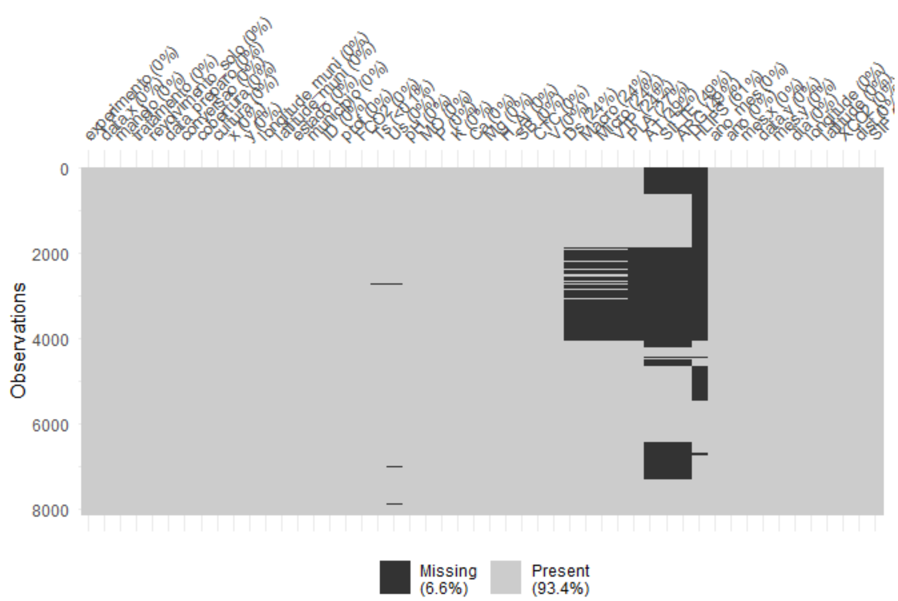

```
data_set_temporal <- data_set %>%
  filter(experimento == "Temporal")

data_set_espacial <- data_set %>%
  filter(experimento == "Espacial")
```

Segunda Aproximação

- Alvo: FCO2
- restrição dados após 2014
- Features: Atributos do Solo + Xco2 e SIF
- Modelo mais simples e geral
- Testar 3 métodos baseados em árvores de decisão

visualização do banco de dados

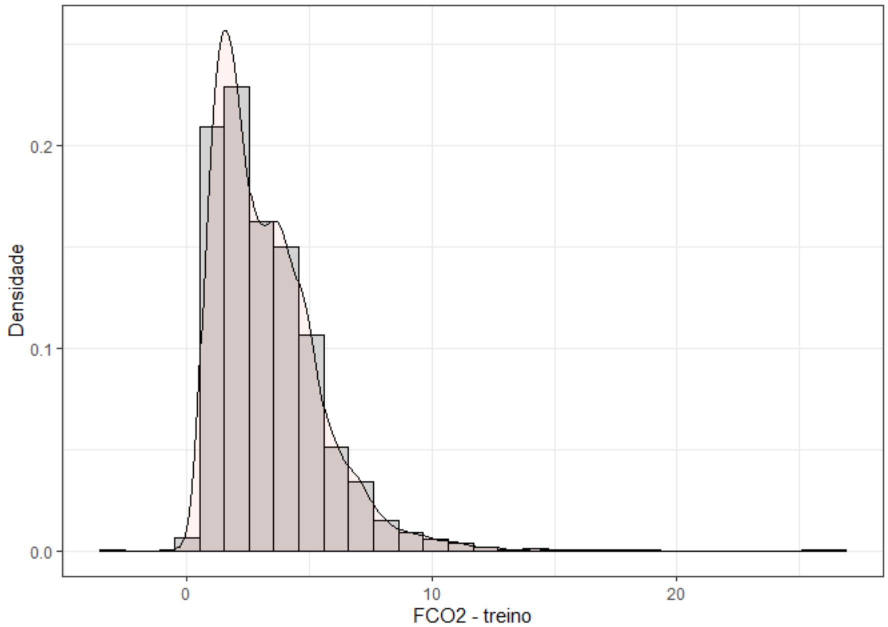
```
visdat::vis_miss(data_set)
```



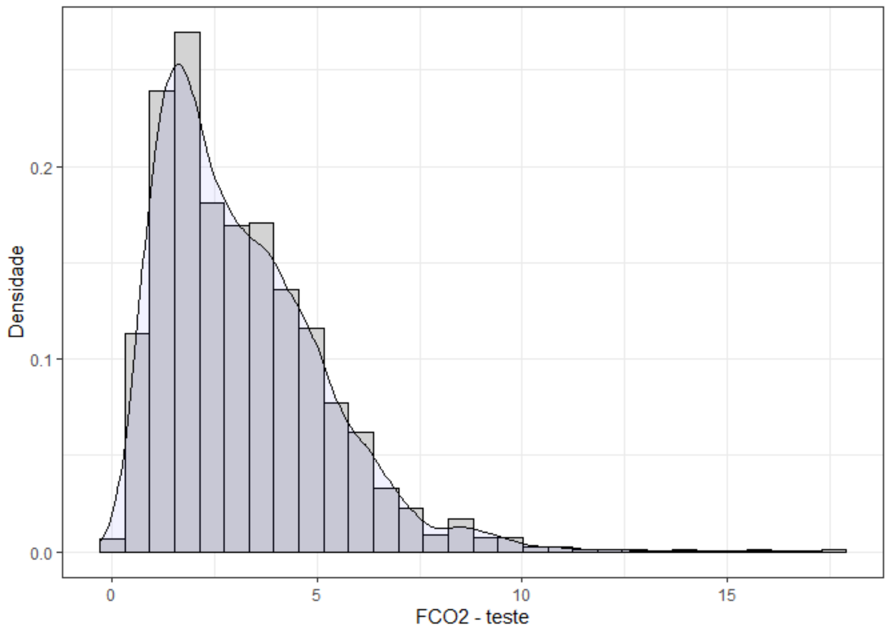
Definindo a Base de treino e teste

```
data_set_ml <- data_set # %>% sample_n(100) # <-------
fco2_initial_split <- initial_split(data_set_ml, prop = 0.75)
```

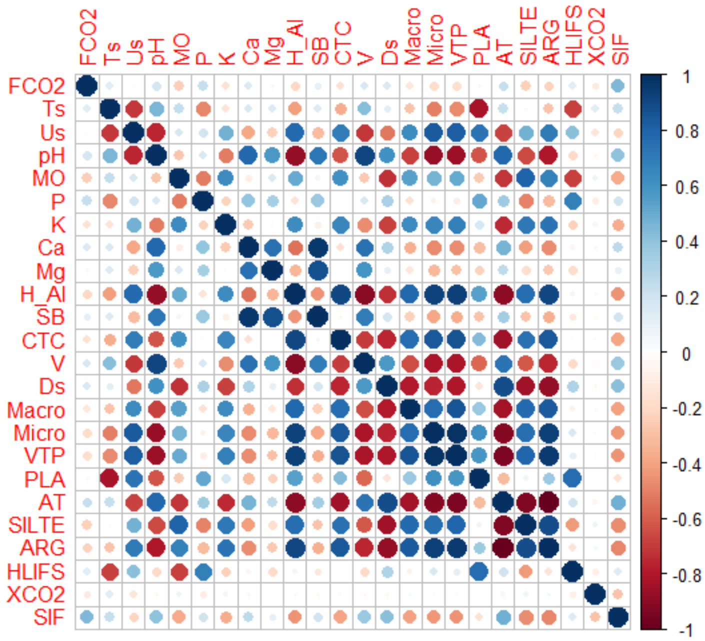
```
fco2_train <- training(fco2_initial_split)
# fco2_test <- testing(fco2_initial_split)
# visdat::vis_miss(fco2_test)
fco2_train  %>%
  ggplot(aes(x=FCO2, y=..density..))+
  geom_histogram(bins = 30, color="black",  fill="lightgray")+
  geom_density(alpha=.05,fill="red")+
  theme_bw() +
  labs(x="FCO2 - treino", y = "Densidade")
```




```
fco2_testing <- testing(fco2_initial_split)
fco2_testing  %>%
  ggplot(aes(x=FCO2, y=..density..))+
  geom_histogram(bins = 30, color="black",  fill="lightgray")+
  geom_density(alpha=.05,fill="blue")+
  theme_bw() +
  labs(x="FCO2 - teste", y = "Densidade")
```



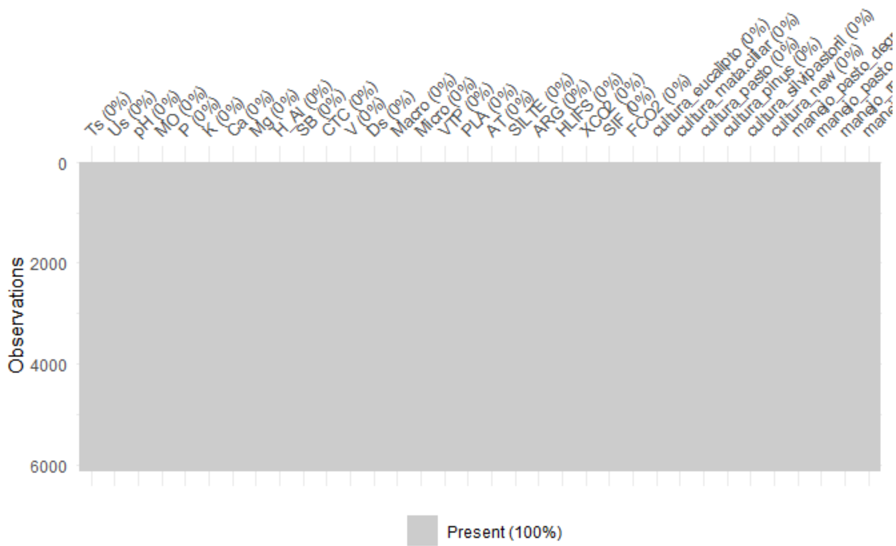

```
fco2_train   %>%    select(FCO2:HLIFS,XCO2,SIF) %>%
  select(where(is.numeric)) %>%
  drop_na() %>%
  cor()  %>%
  corrplot::corrplot()
```



data prep

```
fco2_recipe <- recipe(FCO2 ~ .,
                      data = fco2_train %>%
                        select(cultura, manejo, cobertura, FCO2:HLIFS,XCO2,SIF)
) %>%
  step_normalize(all_numeric_predictors())  %>%
  step_novel(all_nominal_predictors()) %>%
  step_zv(all_predictors()) %>%
  #step_naomit(c(Ts, Us)) %>%
  step_impute_median(where(is.numeric)) %>% # inputação da mediana nos numéricos
  #step_poly(c(Us,Ts), degree = 2)  %>%
  step_dummy(all_nominal_predictors())
bake(prep(fco2_recipe), new_data = NULL)
#> # A tibble: 6,104 × 34
#>         Ts      Us      pH      MO       P       K      Ca     Mg    H_Al
#>      <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>   <dbl>  <dbl>   <dbl>
#>  1 -3.02    1.56   -0.0522 -1.50    2.83   -0.124   1.46    2.46  -0.104
#>  2  0.396  -0.491   0.315   0.0804  0.320  -0.859   0.0468  1.63  -1.38
#>  3 -2.84    0.749  -0.157  -1.31    2.13   -0.294  -0.141  -0.229  0.344
#>  4  0.618  -1.10    0.0528  0.279  -0.484  -0.520   0.892   0.597 -0.456
#>  5 -0.167  -0.406  -0.157   1.07   -0.182   0.0455 -0.0471  0.184  0.600
#>  6 -0.106  -0.0172  0.473  -0.415   0.923  -0.916   0.141   0.597 -1.38
#>  7  0.0206 -0.136   0.315  -0.613  -0.0819 -0.859   0.0468  0.391 -1.32
#>  8  0.583  -0.693  -0.157   1.27   -0.384  -0.181  -0.0471 -0.229  0.0879
#>  9  0.583   1.23    0.105   0.0804 -0.384  -0.350   0.704   1.84  -0.616
#> 10 -1.14    0.960  -0.157   0.972  -0.585  -0.237  -0.517  -0.229  0.856
#> # ℹ 6,094 more rows
#> # ℹ 25 more variables: SB <dbl>, CTC <dbl>, V <dbl>, Ds <dbl>, Macro <dbl>,
#> #   Micro <dbl>, VTP <dbl>, PLA <dbl>, AT <dbl>, SILTE <dbl>, ARG <dbl>,
#> #   HLIFS <dbl>, XCO2 <dbl>, SIF <dbl>, FCO2 <dbl>, cultura_eucalipto <dbl>,
#> #   cultura_mata.ciliar <dbl>, cultura_pasto <dbl>, cultura_pinus <dbl>,
#> #   cultura_silvipastoril <dbl>, cultura_new <dbl>,
#> #   manejo_pasto_degradado <dbl>, manejo_pasto_renovado <dbl>, …
```

```
visdat::vis_miss(bake(prep(fco2_recipe), new_data = NULL))
```



Reamostragem definida e será padrão para todos os modelos

```
fco2_resamples <- vfold_cv(fco2_train, v = 5) # 10 fold
grid <- grid_regular(
  penalty(range = c(-8, 0)),
  levels = 20
)
```

Árvore de Decisão

Definição do modelo

```
fco2_dt_model <- decision_tree(
  cost_complexity = tune(),
  tree_depth = tune(),
  min_n = tune()
)  %>%
  set_mode("regression")  %>%
  set_engine("rpart")
```

Workflow

```
fco2_dt_wf <- workflow()   %>%
  add_model(fco2_dt_model) %>%
  add_recipe(fco2_recipe)
```

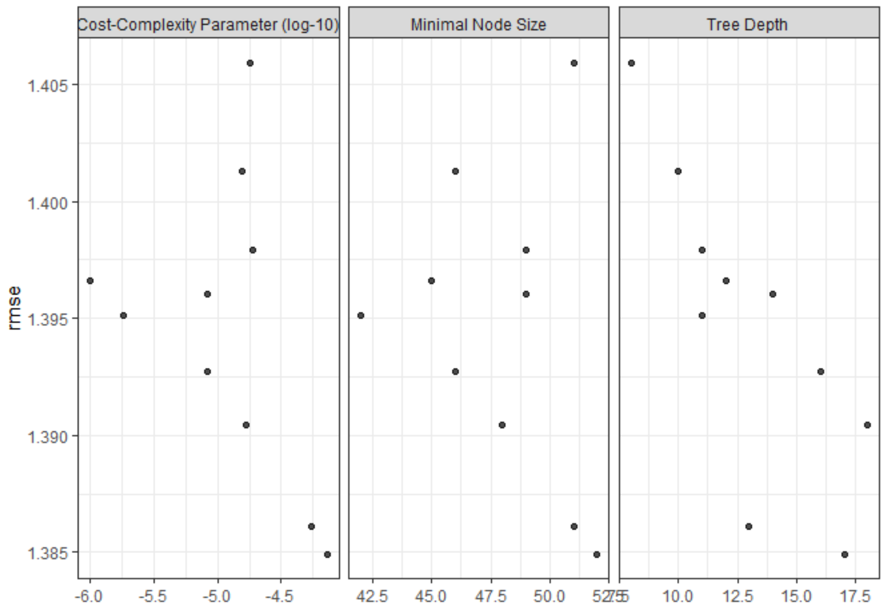
Criando a matriz (grid) com os valores de hiperparâmetros a serem testados

```
grid_dt <- grid_random(
  cost_complexity(c(-6, -4)),
  tree_depth(range = c(8, 18)),
  min_n(range = c(42, 52)),
  size = 10 # <-----------------------------
)
glimpse(grid_dt)
#> Rows: 10
#> Columns: 3
#> $ cost_complexity <dbl> 8.372060e-06, 1.000888e-06, 1.782748e-05, 5.409058e-05…
#> $ tree_depth      <int> 16, 12, 8, 13, 14, 11, 10, 17, 18, 11
#> $ min_n           <int> 46, 45, 51, 51, 49, 42, 46, 52, 48, 49
```

Tuning de hiperparâmetros

```
fco2_dt_tune_grid <- tune_grid(
  fco2_dt_wf,
  resamples = fco2_resamples,
  grid = grid_dt,
  metrics = metric_set(rmse)
)
```

```
autoplot(fco2_dt_tune_grid)
```



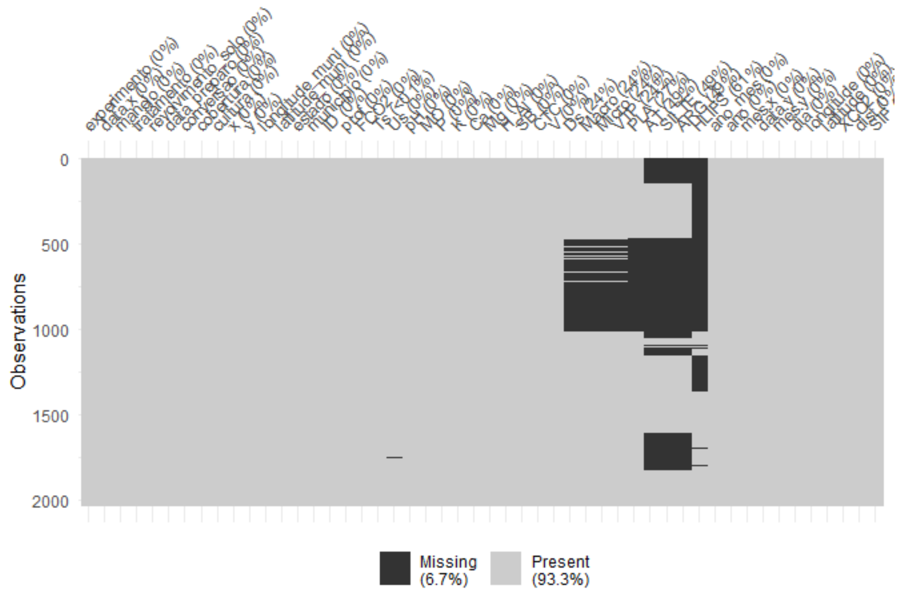

```
collect_metrics(fco2_dt_tune_grid)
#> # A tibble: 10 × 9
#>    cost_complexity tree_depth min_n .metric .estimator  mean     n std_err
#>              <dbl>      <int> <int> <chr>   <chr>      <dbl> <int>   <dbl>
#>  1      0.00000837         16    46 rmse    standard    1.39     5  0.0406
#>  2      0.00000100         12    45 rmse    standard    1.40     5  0.0428
#>  3      0.0000178           8    51 rmse    standard    1.41     5  0.0434
#>  4      0.0000541          13    51 rmse    standard    1.39     5  0.0427
#>  5      0.00000836         14    49 rmse    standard    1.40     5  0.0420
#>  6      0.00000183         11    42 rmse    standard    1.40     5  0.0431
#>  7      0.0000155          10    46 rmse    standard    1.40     5  0.0427
#>  8      0.0000720          17    52 rmse    standard    1.38     5  0.0427
#>  9      0.0000166          18    48 rmse    standard    1.39     5  0.0412
#> 10      0.0000188          11    49 rmse    standard    1.40     5  0.0431
#> # ℹ 1 more variable: .config <chr>
```

Desempenho dos modelos finais

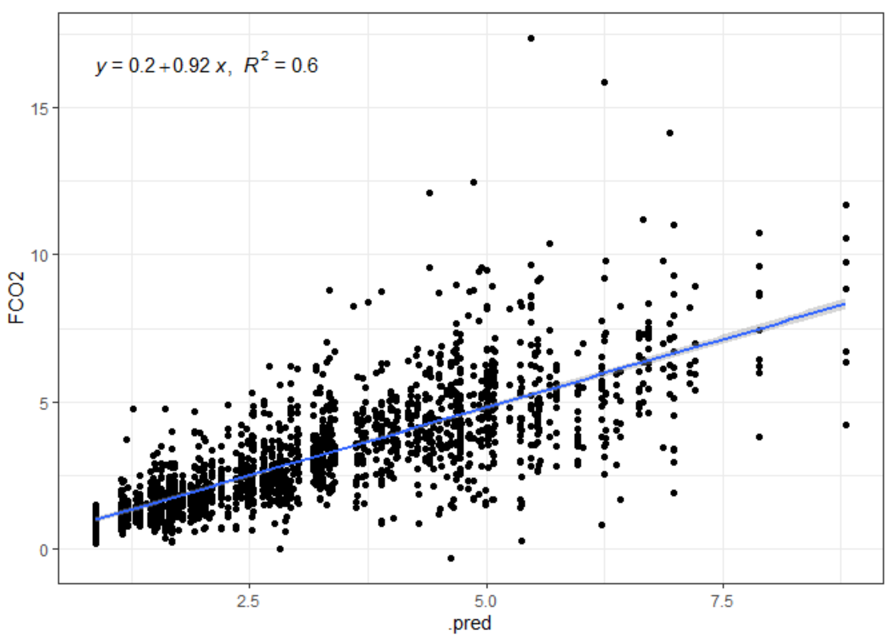
```
fco2_dt_best_params <- select_best(fco2_dt_tune_grid, "rmse")
fco2_dt_wf <- fco2_dt_wf %>% finalize_workflow(fco2_dt_best_params)
fco2_dt_last_fit <- last_fit(fco2_dt_wf, fco2_initial_split)
```

Criando os preditos

```
fco2_test_preds <- bind_rows(
  collect_predictions(fco2_dt_last_fit)  %>%   mutate(modelo = "dt")
)

fco2_test <- testing(fco2_initial_split)
visdat::vis_miss(fco2_test)
```

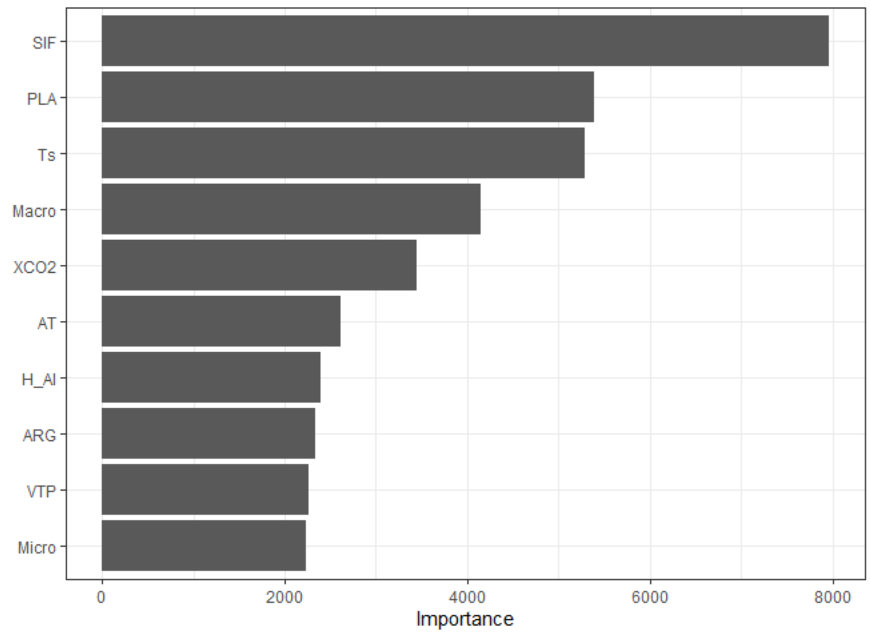



```
fco2_test_preds %>%
  ggplot(aes(x=.pred, y=FCO2)) +
  geom_point()+
  theme_bw() +
  geom_smooth(method = "lm") +
  stat_regline_equation(ggplot2::aes(
  label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")))
```



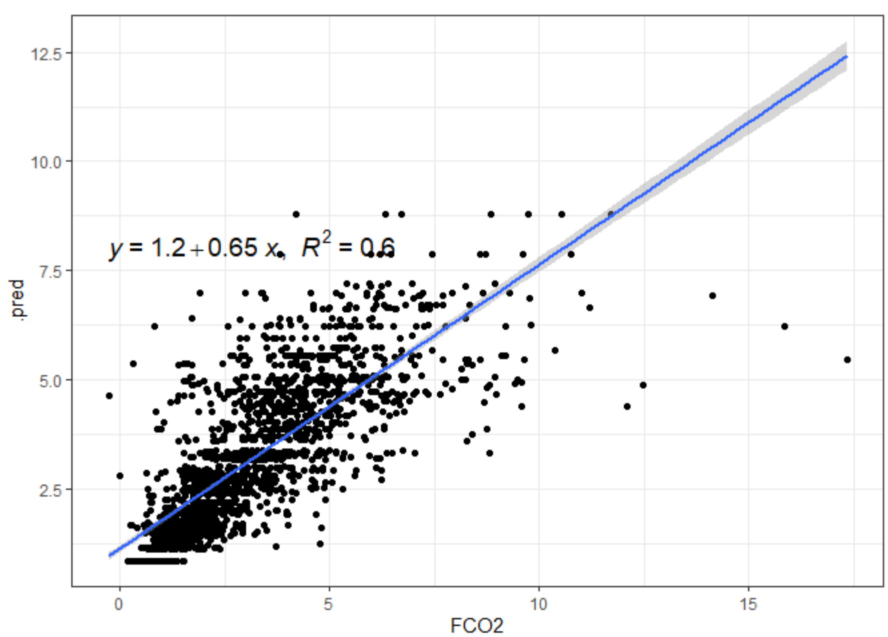
Importância

```
fco2_dt_last_fit_model <-fco2_dt_last_fit$.workflow[[1]]$fit$fit
vip(fco2_dt_last_fit_model)
```



Métricas

```
da <- fco2_test_preds %>%
  filter(FCO2 > 0, .pred>0 )

my_r <- cor(da$FCO2,da$.pred)
my_r2 <- my_r*my_r
my_mse <- Metrics::mse(da$FCO2,da$.pred)
my_rmse <- Metrics::rmse(da$FCO2,
                         da$.pred)
my_mae <- Metrics::mae(da$FCO2,da$.pred)
my_mape <- Metrics::mape(da$FCO2,da$.pred)*100


fco2_test_preds %>%
  ggplot(aes(x=FCO2,y=.pred))+
  geom_point()+
  geom_smooth(method = "lm")+
  stat_regline_equation(ggplot2::aes(
    label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")),size=5)
```



```
  # ggplot2::annotate('text',x=10.4,y=16.7,label=paste0('RMSE = ',round(my_rmse,2),', MAPE = '
  #                                                       ,round(my_mape,2),'%'),size=5)+
  # theme_bw()
vector_of_metrics <- c(r=my_r, R2=my_r2, MSE=my_mse, RMSE=my_rmse, MAE=my_mae, MAPE=my_mape)
print(data.frame(vector_of_metrics))
#>      vector_of_metrics
#> r            0.7767997
#> R2           0.6034178
#> MSE          1.6054010
#> RMSE         1.2670442
#> MAE          0.8313771
#> MAPE        32.6851327
```

Random Forest

Definição do modelo

```
fco2_rf_model <- rand_forest(
  min_n = tune(),
  mtry = tune(),
  trees = tune()
)   %>%
  set_mode("regression")  %>%
  set_engine("randomForest")
```

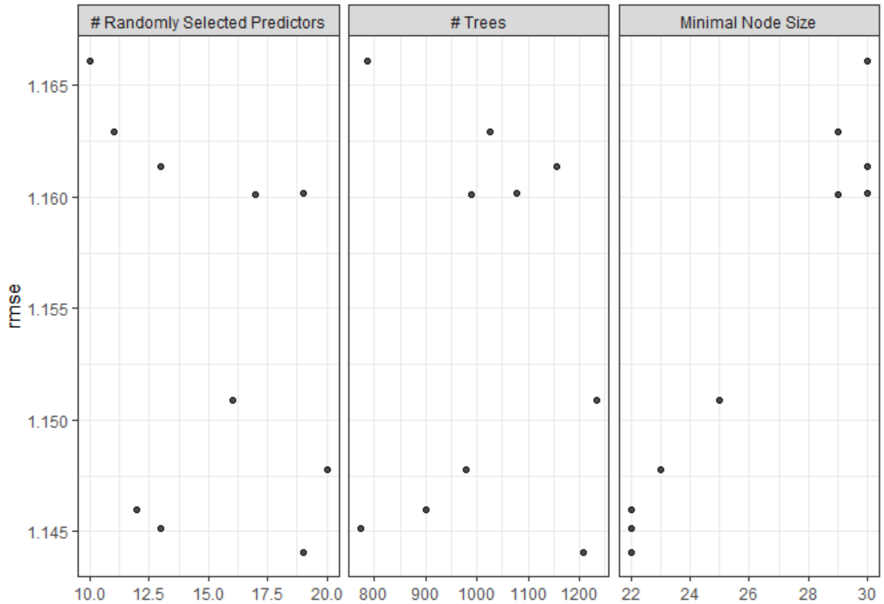
Workflow

```
fco2_rf_wf <- workflow()   %>%
  add_model(fco2_rf_model) %>%
  add_recipe(fco2_recipe)
```

Tune

```
grid_rf <- grid_random(
  min_n(range = c(20, 30)),
  mtry(range = c(10,20)),
  trees(range = c(769,1500) ),
  size = 10 #<-----------------------
)
```

```
fco2_rf_tune_grid <- tune_grid(
 fco2_rf_wf,
  resamples = fco2_resamples,
  grid = grid_rf,
  metrics = metric_set(rmse)
)
autoplot(fco2_rf_tune_grid)
```



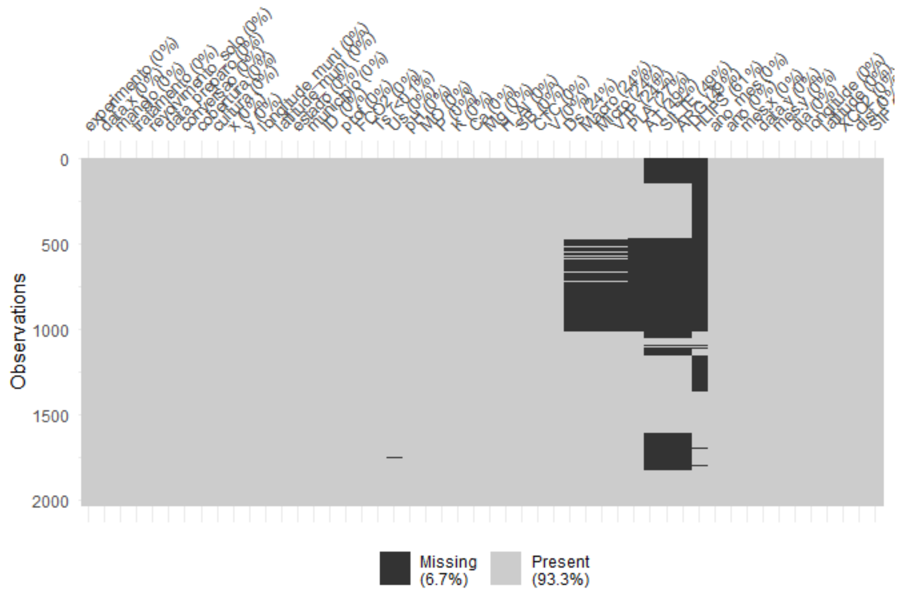

```
collect_metrics(fco2_rf_tune_grid)
#> # A tibble: 10 × 9
#>     mtry trees min_n .metric .estimator  mean     n std_err .config
#>    <int> <int> <int> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
#>  1    13   772    22 rmse    standard    1.15     5  0.0601 Preprocessor1_Model…
#>  2    16  1235    25 rmse    standard    1.15     5  0.0599 Preprocessor1_Model…
#>  3    20   979    23 rmse    standard    1.15     5  0.0591 Preprocessor1_Model…
#>  4    10   785    30 rmse    standard    1.17     5  0.0585 Preprocessor1_Model…
#>  5    19  1078    30 rmse    standard    1.16     5  0.0584 Preprocessor1_Model…
#>  6    11  1025    29 rmse    standard    1.16     5  0.0584 Preprocessor1_Model…
#>  7    12   900    22 rmse    standard    1.15     5  0.0589 Preprocessor1_Model…
#>  8    17   990    29 rmse    standard    1.16     5  0.0587 Preprocessor1_Model…
#>  9    19  1208    22 rmse    standard    1.14     5  0.0598 Preprocessor1_Model…
#> 10    13  1155    30 rmse    standard    1.16     5  0.0588 Preprocessor1_Model…
```

Desempenho modelo final

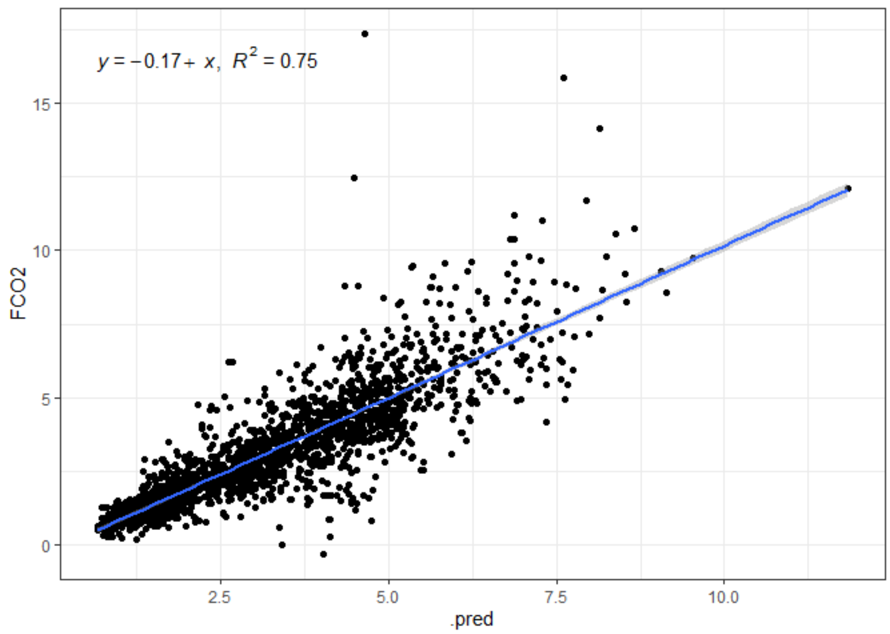
```
fco2_rf_best_params <- select_best(fco2_rf_tune_grid, "rmse")
fco2_rf_wf <- fco2_rf_wf %>% finalize_workflow(fco2_rf_best_params)
fco2_rf_last_fit <- last_fit(fco2_rf_wf, fco2_initial_split)
```

Criando os preditos

```
fco2_test_preds <- bind_rows(
  collect_predictions(fco2_rf_last_fit)  %>%   mutate(modelo = "rf")
)

fco2_test <- testing(fco2_initial_split)
visdat::vis_miss(fco2_test)
```

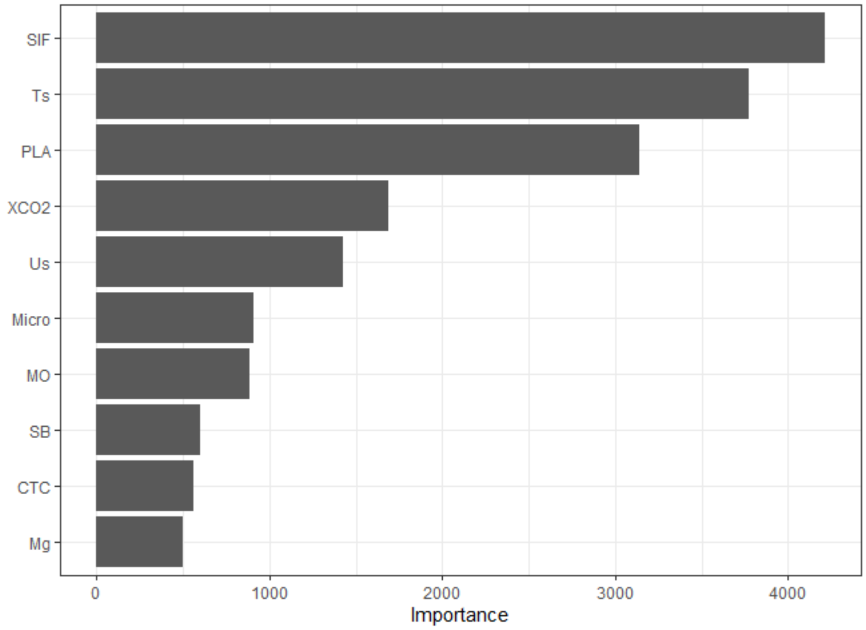



```
fco2_test_preds %>%
  ggplot(aes(x=.pred, y=FCO2)) +
  geom_point()+
  theme_bw() +
  geom_smooth(method = "lm") +
  stat_regline_equation(ggplot2::aes(
  label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")))
```



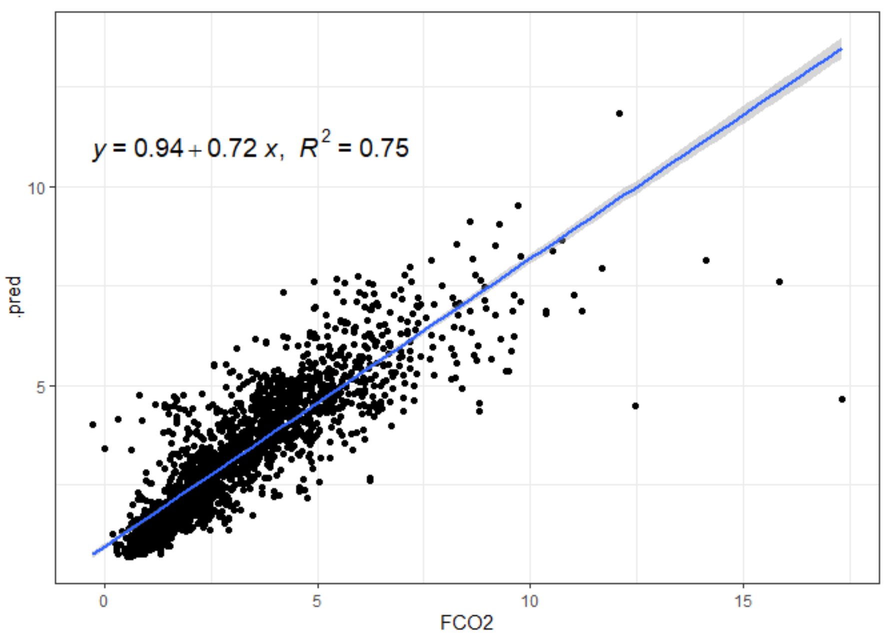
Importância

```
fco2_rf_last_fit_model <-fco2_rf_last_fit$.workflow[[1]]$fit$fit
vip(fco2_rf_last_fit_model)
```



Métricas

```
da <- fco2_test_preds %>%
  filter(FCO2 > 0, .pred>0 )

my_r <- cor(da$FCO2,da$.pred)
my_r2 <- my_r*my_r
my_mse <- Metrics::mse(da$FCO2,da$.pred)
my_rmse <- Metrics::rmse(da$FCO2,
                         da$.pred)
my_mae <- Metrics::mae(da$FCO2,da$.pred)
my_mape <- Metrics::mape(da$FCO2,da$.pred)*100


fco2_test_preds %>%
  ggplot(aes(x=FCO2,y=.pred))+
  geom_point()+
  geom_smooth(method = "lm")+
  stat_regline_equation(ggplot2::aes(
    label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")),size=5)
```



```
  # ggplot2::annotate('text',x=10.4,y=16.7,label=paste0('RMSE = ',round(my_rmse,2),', MAPE = '
  #                                                     ,round(my_mape,2),'%'),size=5)+
  # theme_bw()
vector_of_metrics <- c(r=my_r, R2=my_r2, MSE=my_mse, RMSE=my_rmse, MAE=my_mae, MAPE=my_mape)
print(data.frame(vector_of_metrics))
#>      vector_of_metrics
#> r            0.8668851
#> R2           0.7514899
#> MSE          1.0020021
#> RMSE         1.0010005
#> MAE          0.6459571
#> MAPE        25.9283214
```

Boosting gradient tree (xgb)

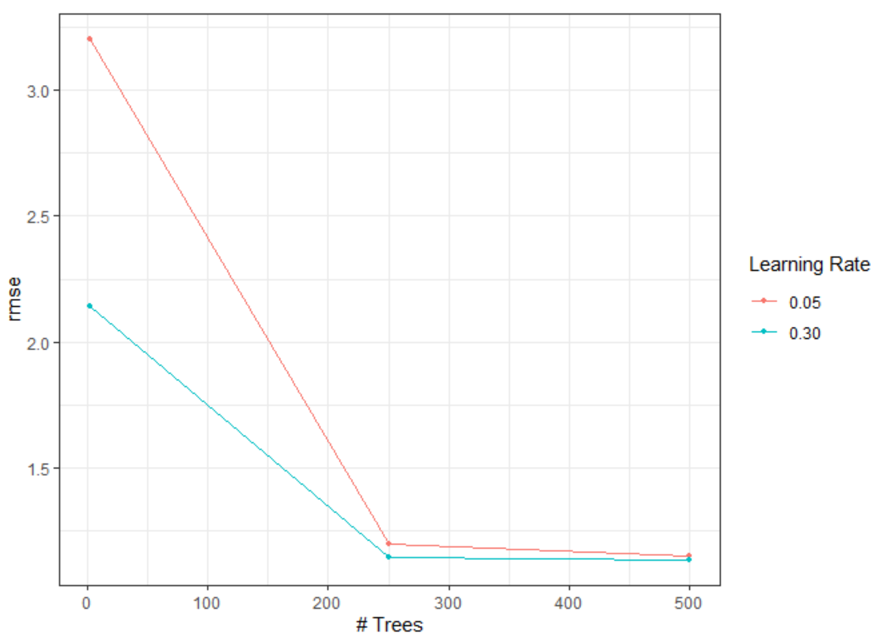
```
cores = 4
fco2_xgb_model <- boost_tree(
  mtry = 0.8,
  trees = tune(), # <---------------
  min_n = 5,
  tree_depth = 4,
  loss_reduction = 0, # lambda
  learn_rate = tune(), # epsilon
  sample_size = 0.8
)  %>%
  set_mode("regression")  %>%
  set_engine("xgboost", nthread = cores, counts = FALSE)
```

```
fco2_xgb_wf <- workflow()  %>%
  add_model(fco2_xgb_model) %>%
  add_recipe(fco2_recipe)
```

```
grid_xgb <- expand.grid(
  learn_rate =  c(0.05, 0.3),
  trees = c(2, 250, 500)
)
```

Passo 1

```
fco2_xgb_tune_grid <- tune_grid(
 fco2_xgb_wf,
  resamples = fco2_resamples,
  grid = grid_xgb,
  metrics = metric_set(rmse)
)
autoplot(fco2_xgb_tune_grid)
```

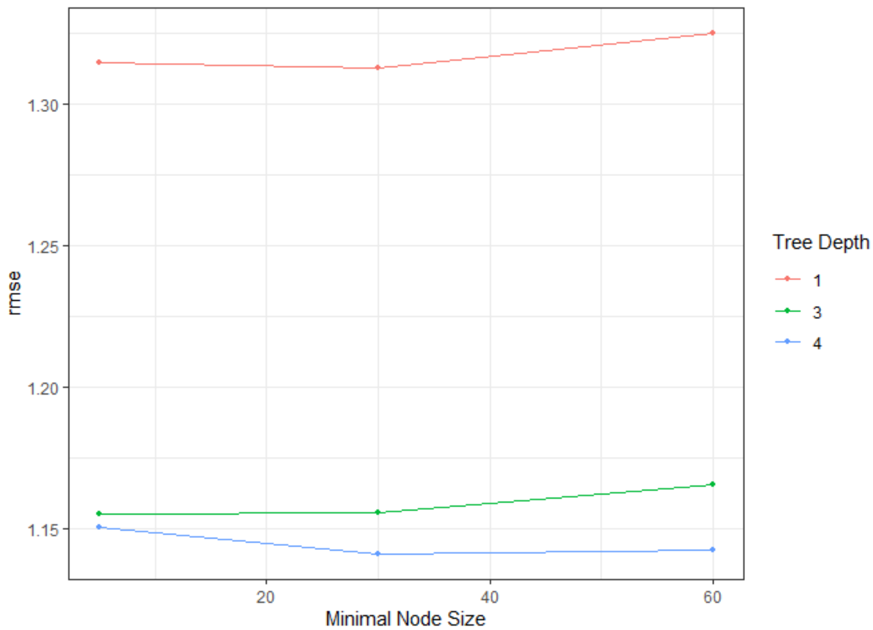


```
fco2_xgb_tune_grid   %>%   show_best(metric = "rmse", n = 6)
#> # A tibble: 6 × 8
#>   trees learn_rate .metric .estimator  mean     n std_err .config
#>   <dbl>      <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
#> 1   500       0.3  rmse    standard    1.14     5  0.0555 Preprocessor1_Model6
#> 2   250       0.3  rmse    standard    1.14     5  0.0541 Preprocessor1_Model5
#> 3   500       0.05 rmse    standard    1.15     5  0.0636 Preprocessor1_Model3
#> 4   250       0.05 rmse    standard    1.20     5  0.0603 Preprocessor1_Model2
#> 5     2       0.3  rmse    standard    2.14     5  0.0580 Preprocessor1_Model4
#> 6     2       0.05 rmse    standard    3.20     5  0.0537 Preprocessor1_Model1
```

```
fco2_xgb_select_best_passo1 <- fco2_xgb_tune_grid %>%
  select_best(metric = "rmse")
fco2_xgb_select_best_passo1
#> # A tibble: 1 × 3
#>   trees learn_rate .config
#>   <dbl>      <dbl> <chr>
#> 1   500        0.3 Preprocessor1_Model6
```

Passo 2

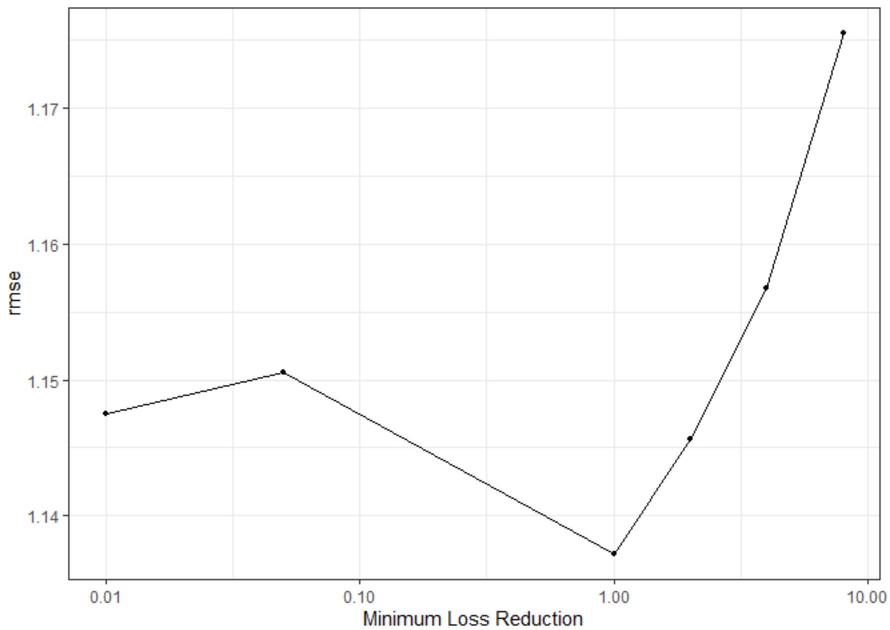
```
fco2_xgb_model <- boost_tree(
  mtry = 0.8,
  trees = fco2_xgb_select_best_passo1$trees,
  min_n = tune(),
  tree_depth = tune(),
  loss_reduction = 0,
  learn_rate = fco2_xgb_select_best_passo1$learn_rate,
  sample_size = 0.8
) %>%
  set_mode("regression")  %>%
  set_engine("xgboost", nthread = cores, counts = FALSE)

#### Workflow
fco2_xgb_wf <- workflow() %>%
    add_model(fco2_xgb_model)   %>%
    add_recipe(fco2_recipe)

#### Grid
fco2_xgb_grid <- expand.grid(
  tree_depth = c(1, 3, 4),
  min_n = c(5, 30, 60)
)

fco2_xgb_tune_grid <- fco2_xgb_wf   %>%
  tune_grid(
    resamples =fco2_resamples,
    grid = fco2_xgb_grid,
    control = control_grid(save_pred = TRUE, verbose = FALSE, allow_par = TRUE),
    metrics = metric_set(rmse)
  )

#### Melhores hiperparâmetros
autoplot(fco2_xgb_tune_grid)
```




```
fco2_xgb_tune_grid  %>%   show_best(metric = "rmse", n = 5)
#> # A tibble: 5 × 8
#>   min_n tree_depth .metric .estimator  mean     n std_err .config
#>   <dbl>      <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
#> 1    30          4 rmse    standard    1.14     5  0.0631 Preprocessor1_Model6
#> 2    60          4 rmse    standard    1.14     5  0.0599 Preprocessor1_Model9
#> 3     5          4 rmse    standard    1.15     5  0.0626 Preprocessor1_Model3
#> 4     5          3 rmse    standard    1.16     5  0.0611 Preprocessor1_Model2
#> 5    30          3 rmse    standard    1.16     5  0.0596 Preprocessor1_Model5
fco2_xgb_select_best_passo2 <- fco2_xgb_tune_grid  %>%   select_best(metric = "rmse")
fco2_xgb_select_best_passo2
#> # A tibble: 1 × 3
#>   min_n tree_depth .config
#>   <dbl>      <dbl> <chr>
#> 1    30          4 Preprocessor1_Model6
```

Passo 3

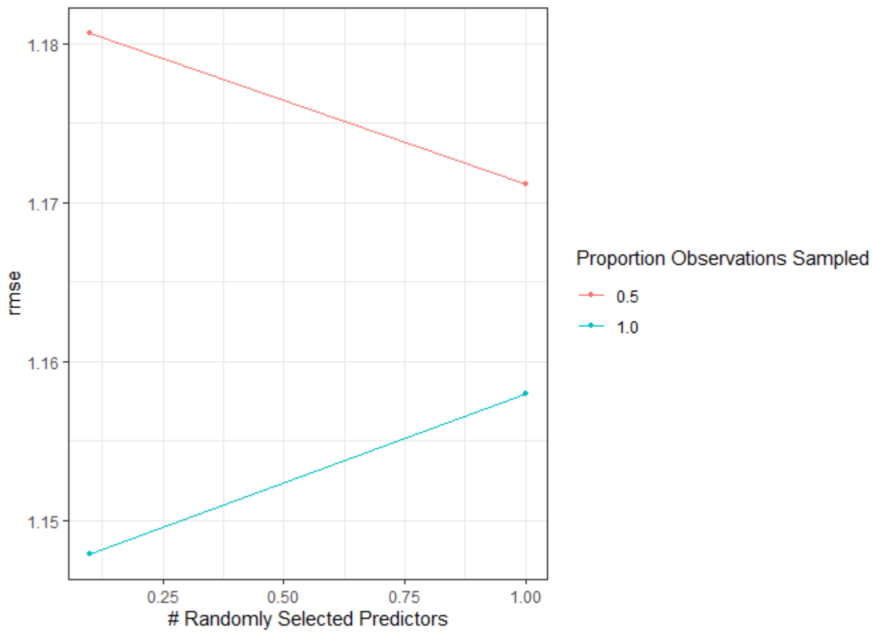
```
fco2_xgb_model <- boost_tree(
  mtry = 0.8,
  trees = fco2_xgb_select_best_passo1$trees,
  min_n = fco2_xgb_select_best_passo2$min_n,
  tree_depth = fco2_xgb_select_best_passo2$tree_depth,
  loss_reduction =tune(),
  learn_rate = fco2_xgb_select_best_passo1$learn_rate,
  sample_size = 0.8
)  %>%
  set_mode("regression")  %>%
  set_engine("xgboost", nthread = cores, counts = FALSE)

#### Workflow
fco2_xgb_wf <- workflow()  %>%
    add_model(fco2_xgb_model)  %>%
    add_recipe(fco2_recipe)

#### Grid
fco2_xgb_grid <- expand.grid(
  loss_reduction = c(0.01, 0.05, 1, 2, 4, 8)
)

fco2_xgb_tune_grid <- fco2_xgb_wf   %>%
  tune_grid(
    resamples = fco2_resamples,
    grid = fco2_xgb_grid,
    control = control_grid(save_pred = TRUE, verbose = FALSE, allow_par = TRUE),
    metrics = metric_set(rmse)
  )

#### Melhores hiperparâmetros
autoplot(fco2_xgb_tune_grid)
```




```
fco2_xgb_tune_grid   %>%   show_best(metric = "rmse", n = 5)
#> # A tibble: 5 × 7
#>   loss_reduction .metric .estimator  mean     n std_err .config
#>            <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
#> 1           1    rmse    standard    1.14     5  0.0567 Preprocessor1_Model3
#> 2           2    rmse    standard    1.15     5  0.0575 Preprocessor1_Model4
#> 3           0.01 rmse    standard    1.15     5  0.0624 Preprocessor1_Model1
#> 4           0.05 rmse    standard    1.15     5  0.0625 Preprocessor1_Model2
#> 5           4    rmse    standard    1.16     5  0.0650 Preprocessor1_Model5
fco2_xgb_select_best_passo3 <- fco2_xgb_tune_grid %>% select_best(metric = "rmse")
fco2_xgb_select_best_passo3
#> # A tibble: 1 × 2
#>   loss_reduction .config
#>            <dbl> <chr>
#> 1              1 Preprocessor1_Model3
```

Passo 4

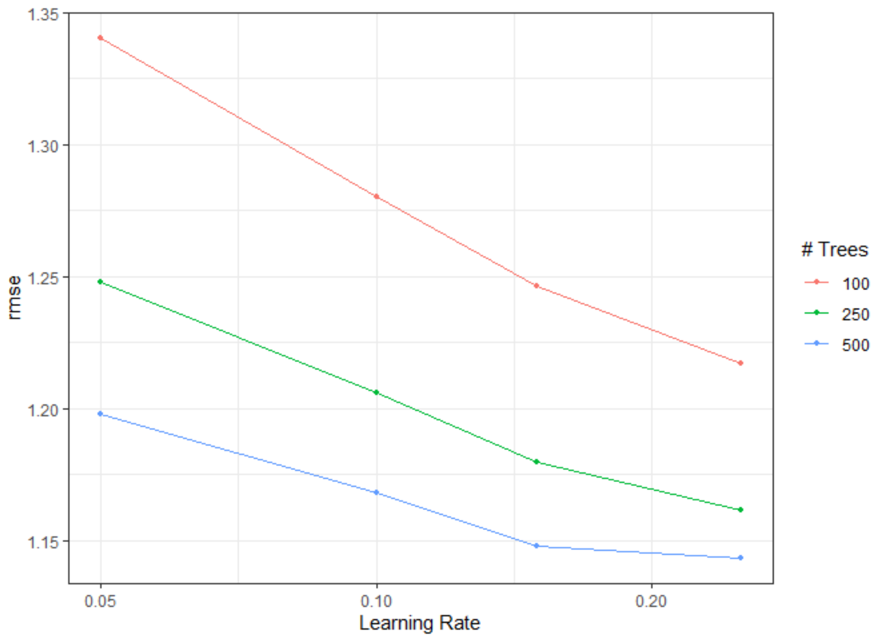
```
fco2_xgb_model <- boost_tree(
  mtry = tune(),
  trees = fco2_xgb_select_best_passo1$trees,
  min_n = fco2_xgb_select_best_passo2$min_n,
  tree_depth = fco2_xgb_select_best_passo2$tree_depth,
  loss_reduction = fco2_xgb_select_best_passo3$loss_reduction,
  learn_rate = fco2_xgb_select_best_passo1$learn_rate,
  sample_size = tune()
)%>%
  set_mode("regression")  |>
  set_engine("xgboost", nthread = cores, counts = FALSE)

#### Workflow
fco2_xgb_wf <- workflow()  %>%
    add_model(fco2_xgb_model)  %>%
    add_recipe(fco2_recipe)

#### Grid
fco2_xgb_grid <- expand.grid(
    sample_size = seq(0.5, 1.0, length.out = 2), ## <---
    mtry = seq(0.1, 1.0, length.out = 2) ## <---
)

fco2_xgb_tune_grid <- fco2_xgb_wf   %>%
  tune_grid(
    resamples = fco2_resamples,
    grid = fco2_xgb_grid,
    control = control_grid(save_pred = TRUE, verbose = FALSE, allow_par = TRUE),
    metrics = metric_set(rmse)
  )

autoplot(fco2_xgb_tune_grid)
```




```
fco2_xgb_tune_grid  |>  show_best(metric = "rmse", n = 5)
#> # A tibble: 4 × 8
#>    mtry sample_size .metric .estimator  mean     n std_err .config
#>   <dbl>       <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
#> 1   0.1           1   rmse     standard    1.15      5  0.0616 Preprocessor1_Model2
#> 2   1             1   rmse     standard    1.16      5  0.0604 Preprocessor1_Model4
#> 3   1             0.5 rmse     standard    1.17      5  0.0628 Preprocessor1_Model3
#> 4   0.1           0.5 rmse     standard    1.18      5  0.0628 Preprocessor1_Model1
fco2_xgb_select_best_passo4 <- fco2_xgb_tune_grid   %>%   select_best(metric = "rmse")
fco2_xgb_select_best_passo4
#> # A tibble: 1 × 3
#>    mtry sample_size .config
#>   <dbl>       <dbl> <chr>
#> 1   0.1           1 Preprocessor1_Model2
```

Passo 5

```
fco2_xgb_model <- boost_tree(
  mtry = fco2_xgb_select_best_passo4$mtry,
  trees = tune(),
  min_n = fco2_xgb_select_best_passo2$min_n,
  tree_depth = fco2_xgb_select_best_passo2$tree_depth,
  loss_reduction = fco2_xgb_select_best_passo3$loss_reduction,
  learn_rate = tune(),
  sample_size = fco2_xgb_select_best_passo4$sample_size
) |>
  set_mode("regression")  %>%
  set_engine("xgboost", nthread = cores, counts = FALSE)

#### Workflow
fco2_xgb_wf <- workflow() %>%
    add_model(fco2_xgb_model)  %>%
    add_recipe(fco2_recipe)

#### Grid
fco2_xgb_grid <- expand.grid(
    learn_rate = c(0.05, 0.10, 0.15, 0.25),
    trees = c(100, 250, 500)
)

fco2_xgb_tune_grid <- fco2_xgb_wf   %>%
  tune_grid(
    resamples = fco2_resamples,
    grid = fco2_xgb_grid,
    control = control_grid(save_pred = TRUE, verbose = FALSE, allow_par = TRUE),
    metrics = metric_set(rmse)
  )

#### Melhores hiperparâmetros
autoplot(fco2_xgb_tune_grid)
```




```
fco2_xgb_tune_grid  %>%   show_best(metric = "rmse", n = 5)
#> # A tibble: 5 × 8
#>   trees learn_rate .metric .estimator  mean     n std_err .config
#>   <dbl>      <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
#> 1   500       0.25 rmse    standard    1.14     5  0.0623 Preprocessor1_Model12
#> 2   500       0.15 rmse    standard    1.15     5  0.0621 Preprocessor1_Model09
#> 3   250       0.25 rmse    standard    1.16     5  0.0608 Preprocessor1_Model11
#> 4   500       0.1  rmse    standard    1.17     5  0.0579 Preprocessor1_Model06
#> 5   250       0.15 rmse    standard    1.18     5  0.0590 Preprocessor1_Model08
fco2_xgb_select_best_passo5 <- fco2_xgb_tune_grid   %>%   select_best(metric = "rmse")
fco2_xgb_select_best_passo5
#> # A tibble: 1 × 3
#>   trees learn_rate .config
#>   <dbl>      <dbl> <chr>
#> 1   500       0.25 Preprocessor1_Model12
```

Desempenho dos modelos finais

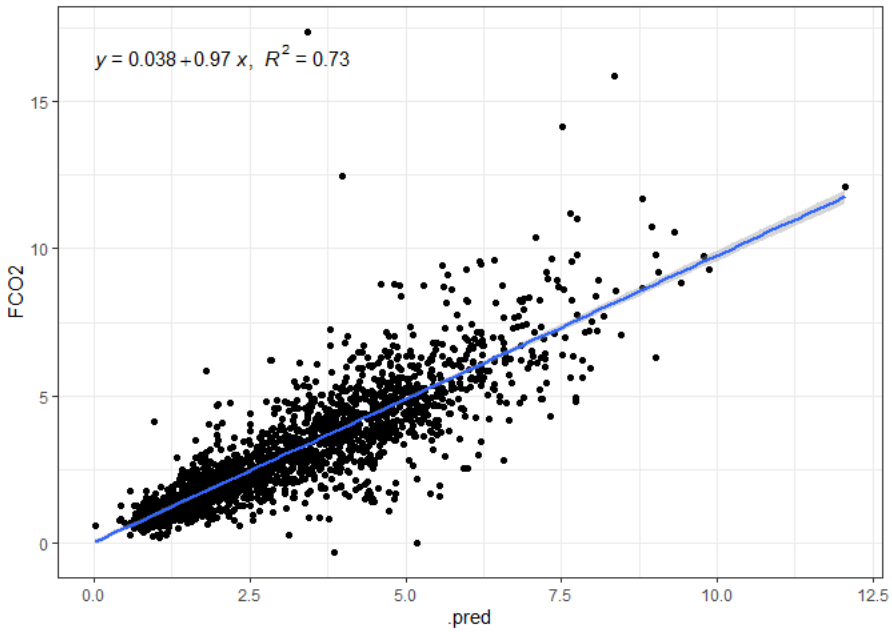
```
fco2_xgb_model <- boost_tree(
  mtry = fco2_xgb_select_best_passo4$mtry,
  trees = fco2_xgb_select_best_passo5$trees,
  min_n = fco2_xgb_select_best_passo2$min_n,
  tree_depth = fco2_xgb_select_best_passo2$tree_depth,
  loss_reduction = fco2_xgb_select_best_passo3$loss_reduction,
  learn_rate = fco2_xgb_select_best_passo5$learn_rate,
  sample_size = fco2_xgb_select_best_passo4$sample_size
) %>%
  set_mode("regression")  %>%
  set_engine("xgboost", nthread = cores, counts = FALSE)
```

```
df <- data.frame(
  mtry = fco2_xgb_select_best_passo4$mtry,
  trees = fco2_xgb_select_best_passo5$trees,
  min_n = fco2_xgb_select_best_passo2$min_n,
  tree_depth = fco2_xgb_select_best_passo2$tree_depth,
  loss_reduction = fco2_xgb_select_best_passo3$loss_reduction,
  learn_rate = fco2_xgb_select_best_passo5$learn_rate,
  sample_size = fco2_xgb_select_best_passo4$sample_size
)
fco2_xgb_wf <- fco2_xgb_wf %>% finalize_workflow(df) # <------
fco2_xgb_last_fit <- last_fit(fco2_xgb_wf, fco2_initial_split) # <--------
```

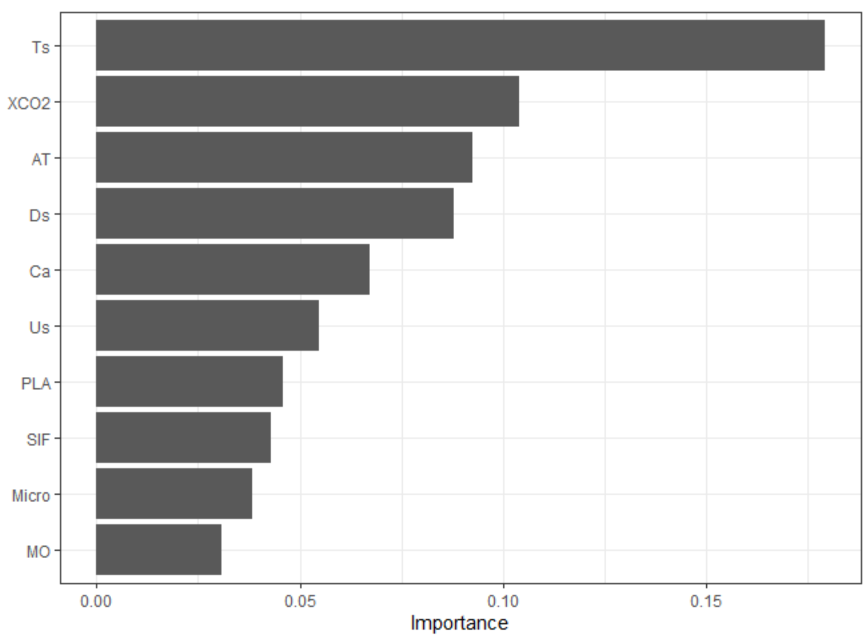
Criar Preditos

```
fco2_test_preds <- bind_rows(
  collect_predictions(fco2_xgb_last_fit)  %>%   mutate(modelo = "xgb")
)
```

```
fco2_test_preds %>%
  ggplot(aes(x=.pred, y=FCO2)) +
  geom_point()+
  theme_bw() +
  geom_smooth(method = "lm") +
  stat_regline_equation(ggplot2::aes(
  label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")))
```

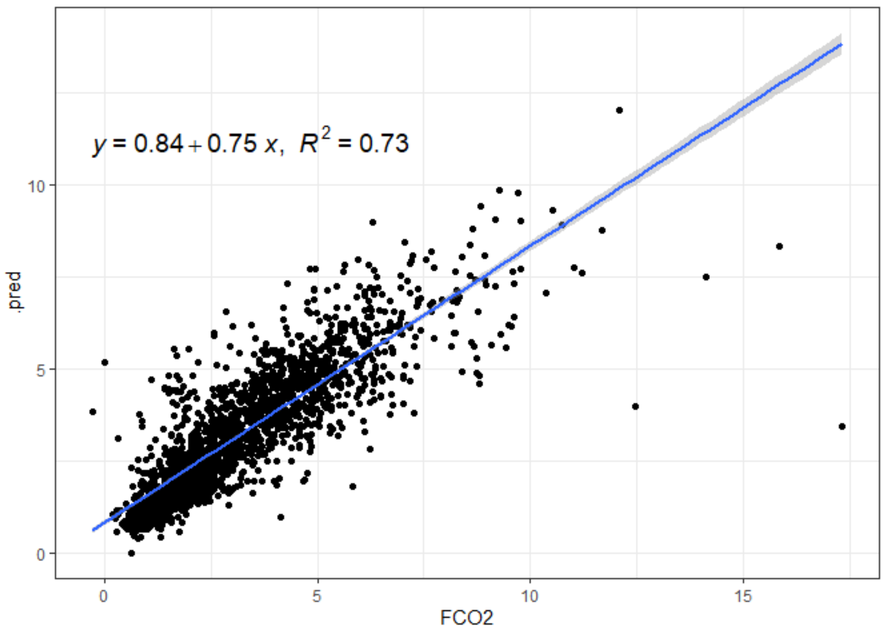



```
fco2_xgb_last_fit_model <-fco2_xgb_last_fit$.workflow[[1]]$fit$fit
vip(fco2_xgb_last_fit_model)
```



Métricas

```
da <- fco2_test_preds %>%
  filter(FCO2 > 0, .pred>0 )

my_r <- cor(da$FCO2,da$.pred)
my_r2 <- my_r*my_r
my_mse <- Metrics::mse(da$FCO2,da$.pred)
my_rmse <- Metrics::rmse(da$FCO2,
                         da$.pred)
my_mae <- Metrics::mae(da$FCO2,da$.pred)
my_mape <- Metrics::mape(da$FCO2,da$.pred)*100


fco2_test_preds %>%
  ggplot(aes(x=FCO2,y=.pred))+
  geom_point()+
  geom_smooth(method = "lm")+
  stat_regline_equation(ggplot2::aes(
    label =  paste(..eq.label.., ..rr.label.., sep = "*plain(\",\")~~")),size=5)
```




```
  # ggplot2::annotate('text',x=10.4,y=16.7,label=paste0('RMSE = ',round(my_rmse,2),', MAPE = '
  #                                                    ,round(my_mape,2),'%'),size=5)+
  # theme_bw()
vector_of_metrics <- c(r=my_r, R2=my_r2, MSE=my_mse, RMSE=my_rmse, MAE=my_mae, MAPE=my_mape)
print(data.frame(vector_of_metrics))
#>      vector_of_metrics
#> r            0.8572155
#> R2           0.7348184
#> MSE          1.0655782
#> RMSE         1.0322685
#> MAE          0.6680929
#> MAPE        26.7237981
```