UFRJ

COS624 - Tóp. Esp. em Banco de Dados

Uso da linguagem R e Tidyverse para CTA de dados

Naya da S Nascimento - DRE 119160428
Barbara Rodrigues dos Santos Cerqueira - DRE 117198425
Maria Luiza Sousa Martins Americo - DRE: 122168778
Barbara Varela Bonfim - DRE: 120130698
Sávio Barreto T da Silva - DRE: 120037175

Nota: Todos os arquivos do trabalho estão disponíveis no repositório: https://github.com/BarbaraCerqueira/COS624-Avaliacao02/tree/analise1-ajustes

1 Análise 1 – Perfis de Eleitorado do TSE

O principal objetivo deste trabalho é explorar o perfil dos eleitores de Espírito Santo e Santa Catarina e realizar uma comparação entre eles, utilizando critérios como número de eleitores, gênero, faixa etária, estado civil, grau de escolaridade, entre outros. A análise foi feita usando a linguagem R e o pacote Tidyverse, enquanto os dados foram obtidos a partir do site do TSE (Tribunal Superior Eleitoral).

1.1 Carga dos dados

Os estados escolhidos para análise foram Santa Catarina e Espírito Santo, e portanto os arquivos utilizados da base do TSE foram:

- perfil_eleitor_secao_ATUAL_SC.csv
- perfil_eleitor_secao_ATUAL_ES.csv

Os arquivos foram carregados utilizando a função read_delim do pacote readr (parte do Tidyverse). As configurações utilizadas foram delimitador ";", a codificação de caracteres "Latin1" para suportar caracteres especiais em português, o idioma "pt" e o separador decimal como ",".

Exemplo de carga do arquivo do Espírito Santo:

```
dados_eleitorado_es <- read_delim(file = "data/eleitorado/es/perfil_eleitor_secao_ATUAL_ES.csv", delim = ";", locale = locale("pt", decimal_mark = ",", encoding = "Latin1"))
```

Para cada estado, foi computado o número de linhas, colunas e células com dados. Os valores obtidos foram os seguintes:

Espírito Santo:

- Número de Linhas: 1571604
- Número de Colunas: 30
- Número de Células com Dados: 47148120

Santa Catarina:

- Número de Linhas: 2977591
- Número de Colunas: 30
- Número de Células com Dados: 89327730

1.2 Transformações de Pré-processamento

Aqui, as colunas irrelevantes para a análise foram eliminadas. A seleção foi feita utilizando a função select do pacote dplyr. As colunas mantidas foram:

- NM_MUNICIPIO
- CD_GENERO, DS_GENERO
- CD_ESTADO_CIVIL, DS_ESTADO_CIVIL
- CD_FAIXA_ETARIA, DS_FAIXA_ETARIA
- CD_GRAU_ESCOLARIDADE, DS_GRAU_ESCOLARIDADE
- QT_ELEITORES_PERFIL

Após isso, verificamos o número de células com dados do dataset após a seleção. Para SC o novo valor foi 29775910, enquanto para o ES foi 15716040.

Em seguida, os dados inválidos foram removidos utilizando a função filter do pacote dplyr. Foi considerado inválidos registros que tivessem o valor “Inválido” na coluna DS_FAIXA_ETARIA, ou valores “NÃO INFORMADO” nas outras colunas:

```
dados_filtrados_es <- filter(dados_selecionados_es,
                             DS_FAIXA_ETARIA != "Inválido",
                             NM_MUNICIPIO != "NÃO INFORMADO",
                             DS_GENERO != "NÃO INFORMADO",
                             DS_ESTADO_CIVIL != "NÃO INFORMADO",
                                     DS_GRAU_ESCOLARIDADE != "NÃO
INFORMADO")
```

Após a filtragem, o novo tamanho dos datasets era:

Espírito Santo:

- Número de Linhas Após Transformações: 1568860
- Número de Colunas Após Transformações: 10
- Número de Células com Dados Após Transformações: 15688600

Santa Catarina:

- Número de Linhas Após Transformações: 2977286
- Número de Colunas Após Transformações: 10
- Número de Células com Dados Após Transformações: 29772860

1.3 Análises

1.3.1 Seleções de linhas que atendam critérios específicos

Foi feita a seleção de eleitores homens, com idade entre 35 e 39 anos, casados,

com nível médio completo.

```
criterios_es <- filter(dados_filtrados_es,
                       DS_GENERO == "MASCULINO",
                       DS_FAIXA_ETARIA == "35 a 39 anos",
                       DS_ESTADO_CIVIL == "CASADO",
                           DS_GRAU_ESCOLARIDADE == "ENSINO MÉDIO
COMPLETO")


total_eleitores_criterios_es                                    <-
sum(criterios_es$QT_ELEITORES_PERFIL)
total_eleitores_geral_es                                        <-
sum(dados_filtrados_es$QT_ELEITORES_PERFIL)
percentagem_es    <-    round((total_eleitores_criterios_es    /
total_eleitores_geral_es) * 100, 2)
```

A análise retornou os seguintes resultados:

Espírito Santo:

- Número de Eleitores que Atendem ao Critério: 15211
- Número Total de Eleitores: 2992116
- O critério escolhido representa 0.51 % dos eleitores do Espírito Santo.

Santa Catarina:

- Número de Eleitores que Atendem ao Critério: 29706
- Número Total de Eleitores: 5628787
- O critério escolhido representa 0.53 % dos eleitores de Santa Catarina.

1.3.2 Tabelas com as contagens de eleitores por cada atributo categórico

Foram feitas tabelas de contagem de eleitores por gênero, município e grau de escolaridade. Foi feita separadamente também uma tabela combinada, com contagem por gênero e faixa etária.

A seguir está um exemplo de como foi feita a tabela de contagem de eleitores por gênero, unindo os dados de ambos os estados em uma só tabela. As outras tabelas foram feitas seguindo essa mesma ideia.

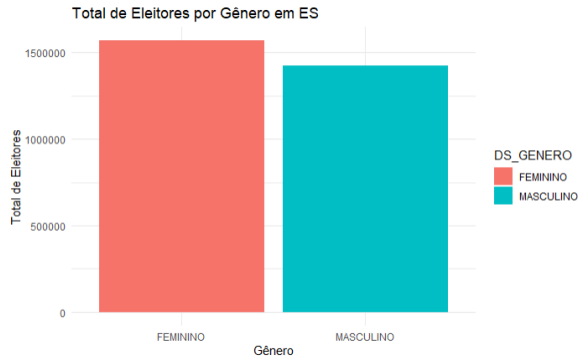
```
# Calcular total de eleitores por gênero no ES

contagem_genero_es <- dados_filtrados_es %>%

  group_by(DS_GENERO) %>%

     summarise(TOTAL_ELEITORES_ES  =  sum(QT_ELEITORES_PERFIL,
na.rm = TRUE))


# Calcular total de eleitores por gênero no SC

contagem_genero_sc <- dados_filtrados_sc %>%

  group_by(DS_GENERO) %>%

     summarise(TOTAL_ELEITORES_SC  =  sum(QT_ELEITORES_PERFIL,
na.rm = TRUE))


contagem_genero       <-       full_join(contagem_genero_es,
contagem_genero_sc, by = "DS_GENERO")

print(contagem_genero)
```

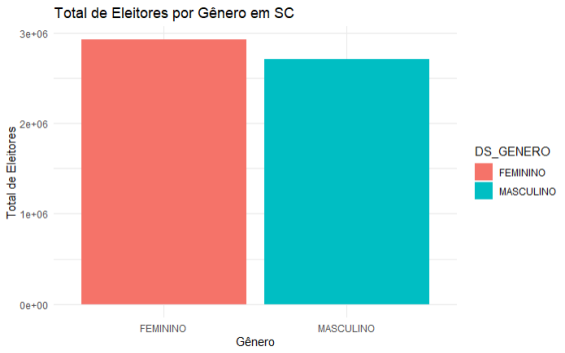
A tabela obtida com esse trecho de código foi:

DS_GENERO <chr>	TOTAL_ELEITORES_ES <dbl>	TOTAL_ELEITORES_SC <dbl>
FEMININO	1568263	2921782
MASCULINO	1423853	2707005

Para esse critério específico também fizemos alguns gráficos usando o pacote ggplot2 do Tidyverse.

```
ggplot(contagem_genero,    aes(x    =    DS_GENERO,    y    =
TOTAL_ELEITORES_ES, fill = DS_GENERO)) +

  geom_bar(stat = "identity") +

  labs(title = "Total de Eleitores por Gênero em ES",

       x = "Gênero",

       y = "Total de Eleitores") +

  theme_minimal()
```





Também foi feito um mapa de calor para a contagem de eleitores por gênero e faixa etária.

Tabela Gênero x Faixa Etária x Número de Eleitores:

```
contagem_genero_idade_es <- dados_filtrados_es %>%
  group_by(DS_GENERO, DS_FAIXA_ETARIA) %>%
  summarise(TOTAL_ELEITORES_ES = sum(QT_ELEITORES_PERFIL)) %>%
  arrange(DS_GENERO, DS_FAIXA_ETARIA)


contagem_genero_idade_sc <- dados_filtrados_sc %>%
  group_by(DS_GENERO, DS_FAIXA_ETARIA) %>%
  summarise(TOTAL_ELEITORES_SC = sum(QT_ELEITORES_PERFIL)) %>%
  arrange(DS_GENERO, DS_FAIXA_ETARIA)


contagem_genero_idade <- full_join(contagem_genero_idade_es, contagem_genero_idade_sc, by = c("DS_GENERO", "DS_FAIXA_ETARIA"))

print(contagem_genero_idade)
```

DS_GENERO <chr>	DS_FAIXA_ETARIA <chr>	TOTAL_ELEITORES_ES <dbl>	TOTAL_ELEITORES_SC <dbl>
FEMININO	100 anos ou mais	1224	2622
FEMININO	16 anos	3832	7425
FEMININO	17 anos	7265	13132
FEMININO	18 anos	15510	29213
FEMININO	19 anos	19992	37179
FEMININO	20 anos	22886	42353
FEMININO	21 a 24 anos	104985	192207
FEMININO	25 a 29 anos	150962	284522
FEMININO	30 a 34 anos	148525	294015
FEMININO	35 a 39 anos	161961	301290

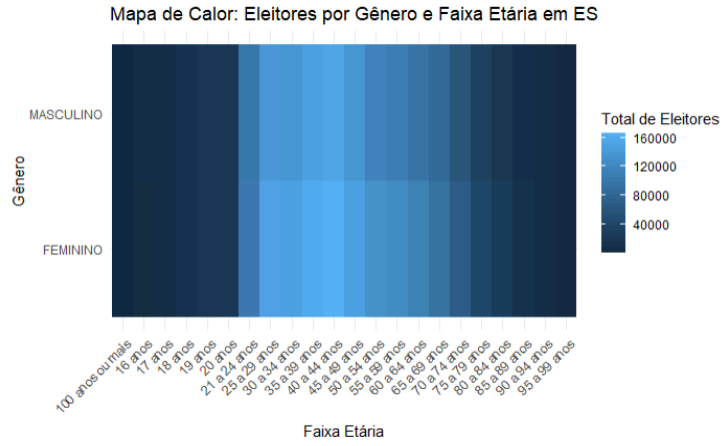
1-10 of 44 rows

Previous 1 2 3 4 5 Next

Mapa de Calor:

```
ggplot(contagem_genero_idade, aes(x = DS_FAIXA_ETARIA, y =
```

```
DS_GENERO, fill = TOTAL_ELEITORES_ES)) +
  geom_tile() +
   labs(title = "Mapa de Calor: Eleitores por Gênero e Faixa Etária em ES",
        x = "Faixa Etária",
        y = "Gênero",
        fill = "Total de Eleitores") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



1.3.3 Estatísticas Básicas

Os datasets de Eleitorado são predominantemente formados por colunas categóricas, e portanto é difícil fazer análises estatísticas nesses dados. Sendo assim, aqui fizemos algumas análises voltadas especificamente para os municípios de cada estado, como por exemplo a média de eleitores ao longo dos municípios, o número máximo e mínimo de eleitores em um município, dentre outros. Para isso, foi reaproveitado a contagem de eleitores por município realizada na análise anterior.

```
estatisticas_es <- eleitores_por_municipio_es %>%
  summarise(
    media_eleitores = mean(TOTAL_ELEITORES_ES, na.rm = TRUE),
      mediana_eleitores = median(TOTAL_ELEITORES_ES, na.rm = TRUE),
    max_eleitores = max(TOTAL_ELEITORES_ES, na.rm = TRUE),
```

```
    min_eleitores = min(TOTAL_ELEITORES_ES, na.rm = TRUE),
    desvio_padrao = sd(TOTAL_ELEITORES_ES, na.rm = TRUE)
  ) %>%
  mutate(estado = "ES")

estatisticas_sc <- eleitores_por_municipio_sc %>%
  summarise(
    media_eleitores = mean(TOTAL_ELEITORES_SC, na.rm = TRUE),
      mediana_eleitores = median(TOTAL_ELEITORES_SC, na.rm =
TRUE),
    max_eleitores = max(TOTAL_ELEITORES_SC, na.rm = TRUE),
    min_eleitores = min(TOTAL_ELEITORES_SC, na.rm = TRUE),
    desvio_padrao = sd(TOTAL_ELEITORES_SC, na.rm = TRUE)
  ) %>%
  mutate(estado = "SC")

estatisticas <- bind_rows(estatisticas_es, estatisticas_sc)
print(estatisticas)
```

Tabela resultante:

media_eleitores <dbl>	mediana_eleitores <dbl>	max_eleitores <dbl>	min_eleitores <dbl>	desvio_padrao <dbl>	estado <chr>
38360.46	15561.5	361362	4947	69499.31	ES
19080.63	6692.0	433953	1534	45073.72	SC

A tabela mostra a média, mediana e desvio padrão de eleitores por município, assim como o número de eleitores máximo e mínimo dentre todos os municípios de ambos os estados, Santa Catarina e Espírito Santo.

1.3.4 Respondendo às perguntas sugeridas

Na proposta do trabalho foram sugeridas algumas análises que respondessem a perguntas específicas. São elas:

- Qual o município com menos eleitores?
- Qual o município com mais eleitores, depois da capital?

- Qual a faixa de variação da quantidade de eleitores por município?
- Qual a faixa de variação da quantidade de eleitores por gênero?

Fizemos essas análises para ambos os estados escolhidos e obtemos os seguintes resultados:

Espírito Santo:

- O município do Espírito Santo com menos eleitores é DIVINO DE SÃO LOURENÇO, com um total de 4947 eleitores.
- O município do Espírito Santo com mais eleitores excluindo a capital é SERRA , com um total de 361362 eleitores. Importante notar que este seria o município com mais eleitores independentemente de excluir a capital Vitória, visto que esta última tem menos eleitores.
- A faixa de variação da quantidade de eleitores por município em Espírito Santo é 356415
- A faixa de variação da quantidade de eleitores por gênero em Espírito Santo é 144410

Santa Catarina:

- O município de Santa Catarina com menos eleitores é SANTIAGO DO SUL , com um total de 1534 eleitores.
- O município de Santa Catarina com mais eleitores excluindo a capital é JOINVILLE , com um total de 433953 eleitores. Importante notar que este seria o município com mais eleitores independentemente de excluir a capital Florianópolis, visto que esta última tem menos eleitores.
- A faixa de variação da quantidade de eleitores por município em Santa Catarina é 432419
- A faixa de variação da quantidade de eleitores por gênero em Santa Catarina é 214777

2 Análise 2 – Dados INMET

2.1 Pré-processamento dos dados

A estação meteorológica escolhida em ambos os anos foi a da praia de Copacabana, no Rio de Janeiro. Os arquivos foram adicionados a pasta Dados_Estacao_Copacabana do repositório, e importados no R utilizando a função “read_delim”, do pacote “readr”. Os parâmetros utilizados foram o caminho do arquivo, o delimitador, no caso ";" e a localização, contendo o idioma pt, o separador decimal “,” e o encoding, o UTF-8.

Para verificarmos a estrutura do dataframe, importado a partir do arquivo, utilizamos o comando `str(dados_clima_2023),` o qual nos fornece informações de linhas, colunas e o tipo de dado de cada coluna. Inicialmente tínhamos 8.760 linhas e 20 colunas em cada um dos dataframes. Para verificarmos as células válidas utilizamos a função `sum(!is.na(dados_clima_2022))` que retorna um objeto lógico do tamanho do dataframe, onde células vazias são tratadas como false e as preenchidas como true, já que temos a negação, em seguida esse objeto lógico é somado, e cada true é tratado como 1. Ao fim, tivemos 159.588 células válidas em 2022 e 143.669 em 2023.

Dentre as 20 colunas, mantivemos as 10 especificadas e outras duas, Data e Hora UTC. Para fazer a filtragem, definimos um vetor com as colunas e em seguida fizemos para ambos os anos: `dados_clima_202X <- dados_clima_202X[, colunas_desejadas],` ou seja, mantivemos todas as linhas e selecionamos apenas as colunas no vetor. Após essa remoção, ficamos com 103.795 células válidas em 2022 e 105.099 em 2023.

A última etapa de pré-processamento consiste em remover todas as linhas com algum campo em branco, e para isso utilizamos a função filter do pacote dplyr, `dados_clima_2022 <- dados_clima_2022 %>% filter(complete.cases(.))`. Nesse caso, complete.cases(.) indica uma llinha cheia, portanto, estamos filtrando aquelas com algum campo vazio. Ao fim, terminamos com 8292 linhas em 2022 e 8755 em 2023.

2. 2 Análise 1

Nessa análise, pensamos em verificar qual seria o ano mais quente na parte da tarde, compreendendo 12-18h. Para isso, transformamos o campo Hora UTC em um número, utilizando:

```
dados_clima_2022 <- dados_clima_2022 %>%
  mutate(`Hora UTC` = as.numeric(substr(`Hora UTC`, 1, 2)))
```

Esse trecho utiliza a função mutate do dplyr, que altera a coluna Hora UTC, extraindo o valor da hora em si da string e transformando ela em um número. Com

essa coluna transformada, podemos usar facilmente a função filter e criar um dataframe auxiliar apenas com essa faixa horária, e em seguida calcular as estatísticas da coluna `TEMPERATURA DO AR - BULBO SECO, HORARIA (°C)` com a função summarise.

```
slice_df <- dados_clima_2022 %>%
  filter(`Hora UTC` >= 9 & `Hora UTC` <= 15)

resultados <- slice_df %>%
  summarise(
    Minimo = min(`TEMPERATURA DO AR - BULBO SECO, HORARIA (°C)`, na.rm = TRUE),
    Maximo = max(`TEMPERATURA DO AR - BULBO SECO, HORARIA (°C)`, na.rm = TRUE),
    Media = mean(`TEMPERATURA DO AR - BULBO SECO, HORARIA (°C)`, na.rm = TRUE)
  )
```

Nesse código, criamos uma tabela resultados com 3 colunas, cada uma delas com um único valor, o máximo, mínimo e a média da coluna.

O resultado em 2023 foi:

Minimo	Máximo	Media
15.8	38.3	25.0

Já em 2022:

Minimo	Máximo	Media
15.4	36.8	24.3

Indicando que 2023 foi um ano mais quente nesse período.

2.3 Análise 2

Utilizando a função summarise o processo de cálculo é bastante semelhante ao anterior. Precisamos apenas definir uma função a parte para o cálculo da moda. A função consiste em retornar os valores únicos dessa coluna, mapear os valores da coluna para índices nesse mesmo vetor e em seguida verificamos o valor de índice que mais ocorre e por fim retornamos o valor do vetor no índice.

```
estatisticas <- dados_clima_2022 %>%
   select(where(is.numeric)) %>%   # Seleciona apenas colunas numéricas
  summarise(across(
```

```
      everything(),
      list(
      Media = ~ mean(.x, na.rm = TRUE),
      Mediana = ~ median(.x, na.rm = TRUE),
      Moda = ~ calcula_moda(.x),
      Minimo = ~ min(.x, na.rm = TRUE),
      Maximo = ~ max(.x, na.rm = TRUE),
      Intervalo = ~ max(.x, na.rm = TRUE) - min(.x, na.rm =
TRUE),
      DesvioPadrao = ~ sd(.x, na.rm = TRUE)
      ),
      .names = "{.col}_{.fn}"
  )) %>%
  pivot_longer(cols = everything(),
               names_to = c("Variavel", "Estatistica"),
               names_sep = "_",
               values_to = "Valor") %>%
  pivot_wider(names_from = Estatistica, values_from = Valor)
```

Nesse caso, utilizamos a função summarise juntamente com a função across(), que recebe como parâmetro todas as colunas, everything(), e uma lista de funções anônimas a serem aplicadas em todas as colunas. Esse processo gera 6 colunas para cada uma, tornando a visualização péssima. Para evitar isso, definimos o nome final da coluna como nome original_estatística, em seguida passamos essa tabela como argumento para a função pivot_longer, que transforma o dataframe original em 3 colunas, a primeira recebe as colunas, a segunda as estatísticas de cada coluna e a terceira os valores, essa separação é feita usando o "_". Por fim, passamos esta última tabela para a função `pivot_wider`, que transforma os valores da coluna Estatistica em colunas e os valores da coluna Valor em seus respectivos valores.

A tabela com o resultado pode ser observada no arquivo analise-clima.html.

2.4 Análise 3

Para realizar os agrupamentos temporais, utilizamos o pacote lubridate e a função floor_date, que "arredonda a data para baixo", ou seja, se definirmos o limiar como um 1 mês, todos os valores dentro do mês serão reduzidos até o dia 01, possibilitando um agrupamento a partir desses valores.

```
agrupamento_diario_2023 <- dados_clima_2023 %>%
```

```
  group_by(dia = floor_date(datetime, "day")) %>%
    summarise(across(where(is.numeric), \(x) mean(x, na.rm =
TRUE)))

agrupamento_semanal_2023 <- dados_clima_2023 %>%
  group_by(semana = floor_date(datetime, "week")) %>%
    summarise(across(where(is.numeric), \(x) mean(x, na.rm =
TRUE)))

agrupamento_mensal_2023 <- dados_clima_2023 %>%
  group_by(mes = floor_date(datetime, "month")) %>%
    summarise(across(where(is.numeric), \(x) mean(x, na.rm =
TRUE)))

agrupamento_bimestral_2023 <- dados_clima_2023 %>%
    group_by(bimestre = cut(datetime, breaks = "2 months",
labels = FALSE)) %>%
    summarise(across(where(is.numeric), \(x) mean(x, na.rm =
TRUE)))

agrupamento_semestral_2023 <- dados_clima_2023 %>%
   group_by(semestre = semester(datetime)) %>%   # Define o
semestre com lubridate
    summarise(across(where(is.numeric), \(x) mean(x, na.rm =
TRUE)))
```

Em todos os casos a lógica a seguir foi a mesma, a única coisa diferente foi o valor do parâmetro do intervalo de tempo.

2.5 Análise 4

Para calcularmos as estatísticas dos agrupamentos, bastava utilizar o mesmo processo com o summarise, mas como estávamos dispostos a realizar comparações de temperatura entre os anos, juntamos os dataframes dos agrupamentos diários de ambos os anos e em seguida calculamos as estatísticas básicas:

```
agrupamento_diario <- bind_rows(agrupamento_diario_2022,
agrupamento_diario_2023)

estatisticas_diarias <- agrupamento_diario %>%
  group_by(ano) %>%
  summarise(
    Media_Temperatura = mean(`TEMPERATURA DO AR - BULBO SECO,
```

```
HORARIA (°C)`, na.rm = TRUE),
    Mediana_Temperatura = median(`TEMPERATURA DO AR - BULBO
SECO, HORARIA (°C)`, na.rm = TRUE),
    Minimo_Temperatura = min(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Maximo_Temperatura = max(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Desvio_Temperatura = sd(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    .groups = "drop"
  )

estatisticas_diarias
```

ano	Media_Temperatura	Mediana_Temperatura	Minimo_Temperatura	Maximo_Temperatura	Desvio_Temperatura
2022	23.32376	23.86726	18.95688	27.19643	1.949284
2023	24.06108	24.07857	20.08393	27.17143	1.784709

2.6 Análise 5

Para calcular os valores mínimo, máximo e médio em cada agrupamento, bastou discriminarmos quais colunas seriam criadas a partir da coluna de temperatura, utilizando os mesmos intervalos:

```
agrupamento_diario_2023 <- dados_clima_2023 %>%
  group_by(dia = floor_date(datetime, "day")) %>%
  summarise(
    Media_Temperatura = mean(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Maxima_Temperatura = max(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Minima_Temperatura = min(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    .groups = "drop"
  )

agrupamento_semanal_2023 <- dados_clima_2023 %>%
  group_by(semana = floor_date(datetime, "week")) %>%
  summarise(
    Media_Temperatura = mean(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Maxima_Temperatura = max(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
```

```
    Minima_Temperatura = min(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    .groups = "drop"
  )

agrupamento_mensal_2023 <- dados_clima_2023 %>%
  group_by(mes = floor_date(datetime, "month")) %>%
  summarise(
    Media_Temperatura = mean(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Maxima_Temperatura = max(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Minima_Temperatura = min(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    .groups = "drop"
  )
agrupamento_bimestral_2023 <- dados_clima_2023 %>%
   group_by(bimestre = cut(datetime, breaks = "2 months",
labels = FALSE)) %>%
  summarise(
    Media_Temperatura = mean(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Maxima_Temperatura = max(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Minima_Temperatura = min(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    .groups = "drop"
  )
agrupamento_semestral_2023 <- dados_clima_2023 %>%
   group_by(semestre = semester(datetime)) %>%   # Define o
semestre com lubridate
  summarise(
    Media_Temperatura = mean(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Maxima_Temperatura = max(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    Minima_Temperatura = min(`TEMPERATURA DO AR - BULBO SECO,
HORARIA (°C)`, na.rm = TRUE),
    .groups = "drop"
  )

agrupamento_diario_2023
```

2.7 Análise 6

Com base nos resultados da análise 4, verificamos que em média o ano de 2023 foi mais quente. De posse dos últimos agrupamentos mensais de ambos os anos, foi possível verificar que neles os meses de fevereiro e março foram os mais quentes. Já em 2023 os meses de julho e agosto foram os mais frios, em 2022 foram setembro e junho, as tabelas podem ser facilmente verificadas no arquivo analise-clima.html. O próximo passo seria verificar o volume de chuva, ou seja, verificar os meses mais chuvosos do ano, e para isso faremos basicamente o mesmo processo da análise 5, mas usando a coluna PRECIPITAÇÃO TOTAL, HORÁRIO (mm):

```
agrupamento_mensal_2022_precipitacao <- dados_clima_2022 %>%
  group_by(mes = floor_date(datetime, "month")) %>%
  summarise(
    Media_Precipitacao = mean(`PRECIPITAÇÃO TOTAL, HORÁRIO
(mm)`, na.rm = TRUE),
    Maxima_Precipitacao = max(`PRECIPITAÇÃO TOTAL, HORÁRIO
(mm)`, na.rm = TRUE),
    Minima_Precipitacao = min(`PRECIPITAÇÃO TOTAL, HORÁRIO
(mm)`, na.rm = TRUE),
    .groups = "drop"
  )

agrupamento_mensal_2023_precipitacao <- dados_clima_2023 %>%
  group_by(mes = floor_date(datetime, "month")) %>%
  summarise(
    Media_Precipitacao = mean(`PRECIPITAÇÃO TOTAL, HORÁRIO
(mm)`, na.rm = TRUE),
    Maxima_Precipitacao = max(`PRECIPITAÇÃO TOTAL, HORÁRIO
(mm)`, na.rm = TRUE),
    Minima_Precipitacao = min(`PRECIPITAÇÃO TOTAL, HORÁRIO
(mm)`, na.rm = TRUE),
    .groups = "drop"
  )
```

A partir dos resultados gerados, verificamos que em 2022 o mês mais chuvoso foi abril, enquanto em 2023 foi fevereiro.