

**UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO
BACHARELADO EM CIÊNCIAS DE DADOS**

ANDRÉ THIAGO GOMES TABOADA RA 2003616

ANDREIA MARIA DA SILVA RAMOS RA 2015234

BIANCA DE SOUZA VIEIRA RA 2011157

CARLOS ADALBERTO SILVA MOREIRA JUNIOR RA 2015232

GABRIEL MERÊNCIO DOS SANTOS RA2000440

LAIZ HIROMI KODAIRA RA 2002395

LORENZO NATHANIEL NO RA 2001230

LUCAS RODRIGUES ZANFORLIN RA 2015835

**Disponibilização e visualização dos dados de sistemas
produtores da SABESP via dashboard e API**

SÃO PAULO, SP

Outubro, 2023

**UNIVERSIDADE VIRTUAL DO ESTADO DE SÃO PAULO
BACHARELADO EM CIÊNCIAS DE DADOS**

ANDRÉ THIAGO GOMES TABOADA RA 2003616

ANDREIA MARIA DA SILVA RAMOS RA 2015234

BIANCA DE SOUZA VIEIRA RA 2011157

CARLOS ADALBERTO SILVA MOREIRA JUNIOR RA 2015232

GABRIEL MERÊNCIO DOS SANTOS RA2000440

LAIZ HIROMI KODAIRA RA 2002395

LORENZO NATHANIEL NO RA 2001230

LUCAS RODRIGUES ZANFORLIN RA 2015835

**Disponibilização e visualização dos dados de sistemas
produtores da SABESP via dashboard e API**

Pré-projeto de pesquisa submetido como requisito parcial para aprovação na disciplina Trabalho de Conclusão de Curso do curso de Bacharelado em Ciências de Dados da Universidade Virtual do Estado de São Paulo.

São Paulo, SP

Outubro, 2023.

TABOADA, André Thiago Gomes; RAMOS, Andreia Maria da Silva; VIEIRA, Bianca de Souza; JUNIOR, Carlos Adalberto Silva Moreira; SANTOS, Gabriel Merêncio dos; KODAIRA, Laiz Hiromi; NO, Lorenzo Nathaniel; ZANFORLIN, Lucas Rodrigues. **Disponibilização e visualização de dados de sistemas produtores da SABESP via dashboard e API.** Trabalho de Conclusão de Curso (Bacharelado em Ciência de Dados). Universidade Virtual do Estado de São Paulo, São Paulo, 2023.

Resumo

Este trabalho propõe a disponibilização e visualização dos dados de sistemas produtores da SABESP via *dashboard* e API. O objetivo geral é tornar o acesso aos dados mais democrático, comprehensível e acessível por meio de uma API e *dashboard*. A SABESP disponibiliza dados sobre os sistemas produtores que abastecem o estado de São Paulo em um site próprio. No entanto, o acesso a esses dados é dificultado por alguns problemas, como a seleção lenta de intervalos grandes de tempo e a falta de formatos padronizados e convenientes, além da ausência de gráficos e uma apresentação acessível ao público geral. Para resolver esses problemas, o trabalho coleta os dados da SABESP e os disponibiliza em formatos padronizados e convenientes, além de disponibilizar um *dashboard* com os dados em gráficos interativos para o público geral. O trabalho contribui para a transparência e o controle social na gestão de recursos hídricos. A disponibilização dos dados de forma mais democrática, comprehensível e acessível permite que o público geral monitore a situação em sua região e que desenvolvedores, profissionais de dados, acadêmicos e outros públicos especializados possam contribuir com novas ferramentas, soluções, modelos e *insights*. Após a construção da aplicação, usando metodologias como *design thinking* e CRISP-DM, distribuímos um questionário ao público geral e obtivemos avaliações positivas.

Palavras-chave: disponibilização e visualização dos dados, análise de dados, dados públicos SABESP

Lista de ilustrações

Figura 1 – Erro de <i>Gateway Timeout</i> na página da SABESP.....	5
Figura 2 – Arquivo XML exportado na página da SABESP.....	6
Figura 3 – Tentativa de importar os dados da SABESP com a biblioteca <i>Pandas</i>	6
Figura 4 – Tabulação original dos dados disponibilizados pela SABESP.....	7
Figura 5 – Gráfico mostrando a vazão observada e prevista de um ponto de controle no SPHM-PJC.....	10
Figura 6 – Página inicial do protótipo do <i>dashboard</i>	14
Figura 7 – Página do Sistema Cantareira no protótipo do <i>dashboard</i>	14
Figura 8 – Tabela com os dados selecionados e opção de <i>download</i> no protótipo do <i>dashboard</i>	15
Figura 9 – Requisição do portal da SABESP aos dados dos sistemas.....	16
Figura 10 – Código de tratamento dos arquivos exportados pelo site da SABESP..	17
Figura 11 – Amostra dos dados tratados de represas do Sistema Cantareira.....	18
Figura 12 – Concatenação dos <i>Dataframes</i> e ordenação do <i>DataFrame</i> resultante.....	19
Figura 13 - Verificação dos tipos do <i>DataFrame</i>	20
Figura 14 - Conversão da coluna ‘Data’	20
Figura 15 - Contagem dos valores nulos.....	21
Figura 16 - Verificação de duplicidade nos nomes e valores das colunas.....	22
Figura 17 - Verificação e remoção de duplicidade.....	23
Figura 18 - Função com retorno dos <i>outliers</i> superior e inferior.....	24
Figura 19 - Visualização do <i>boxplot</i>	24
Figura 20 - Colunas da Represa Cachoeira da França.....	26
Figura 21 - Código teste de Shapiro-Wilk.....	27
Figura 22 - Resultado do referente ao P Valor.....	28
Figura 23 – Histograma dos valores do teste.....	28
Figura 24 – Etapas do CRISP-DM.....	30
Figura 25 – <i>Dashboard</i> final após melhorias.....	35
Figura 26 – Página “Sobre os dados” na aplicação final.....	36
Figura 27 – Página “API” na aplicação final.....	36
Figura 28 – <i>Endpoints</i> da API na documentação.....	37

Figura 29 – Documentação de um <i>endpoint</i> da API.....	38
Figura 30 – Resultado de um teste de um <i>endpoint</i> da API.....	38
Figura 31 – Descrições dos dados fornecidos pela API na documentação.....	39
Figura 32 – Resumo das respostas às perguntas 1 e 2 do questionário de avaliação.....	41
Figura 33 – Resumo das respostas às perguntas 3 e 4 do questionário de avaliação.....	42
Figura 34 – Resumo das respostas às perguntas 4 e 5 do questionário de avaliação.....	43
Figura 35 – Resumo das respostas às perguntas 6 e 7 do questionário de avaliação.....	44
Figura 36 – Resumo das respostas às perguntas 8 e 9 do questionário de avaliação.....	45
Figura 37 – Resumo das respostas à pergunta 10 do questionário de avaliação....	45

Lista de tabelas

Tabela 1 - Sistema Produtores e seus respectivos reservatórios.....	25
Tabela 2 - Descrição das colunas.....	26
Tabela 3 – Cronograma do trabalho.....	33

Sumário

1 Introdução.....	1
2 Objetivos.....	3
2.1 Objetivos gerais.....	3
2.2 Objetivos específicos.....	3
3 Justificativa.....	4
4 Referencial teórico.....	7
4.1 Catalogação de referências.....	7
4.2 Trabalhos relacionados.....	9
4.3 Diferencial competitivo.....	11
5 Metodologia.....	13
5.1 Coleta de dados.....	15
5.2 Tratamento dos dados.....	16
5.3 Análise exploratória dos dados.....	25
5.4 Construção do dashboard.....	30
5.5 Construção da API.....	31
6 Cronograma.....	33
7 Resultados e discussões.....	35
8 Conclusões.....	46
9 Referências.....	48
10 Anexos.....	51
Anexo A - Trechos do Projeto Integrador IV referente à previsão de volume do Sistema Cantareira.....	51

1 Introdução

O acesso à informação é uma questão relevante e de suma importância quando consideramos nossos direitos como cidadãos. Podemos encontrar na Lei de Acesso à Informação, lei de nº 12.527/2011, as garantias do nosso direito constitucional de solicitar e obter informações de órgãos e entidades públicas. O livre acesso aos dados e informações permitem que cidadãos possam cobrar e participar de forma mais ativa e, consequentemente, melhorar a gestão das instituições públicas. Podemos chamar isso de controle social, também assegurado pela Constituição Federal de 1988 (BEZERRA, 2021).

Desta forma, a partir da experiência do grupo em um projeto integrador anterior (anexo A), verificou-se uma deficiência na disponibilização e acesso aos dados da Companhia de Saneamento Básico do Estado de São Paulo (SABESP). A SABESP é uma sociedade anônima de economia mista, sendo o estado de São Paulo o maior acionista, e por isso disponibiliza publicamente seus dados, serviços e prestação de contas. Contradicoratoriamente, apesar de conseguirmos acessar os dados da SABESP em seu próprio site, verificamos uma certa dificuldade na extração, manipulação e visualização dos dados, resultando em uma barreira para grande parte da população.

Somado a isso, temos que a mera disponibilização dos dados não garante à população acesso à informação. Como mencionado por Bezerra (2021):

A participação ativa do cidadão no controle social pressupõe a transparência das ações governamentais. O governo deve propiciar ao cidadão a possibilidade de entender os instrumentos de gestão, para que ele possa influenciar no processo de tomada de decisões. O acesso do cidadão à informação simples e compreensível é o ponto de partida para uma maior transparência. (BEZERRA, 2021)

O dado por si só não significa informação; assim como descrito por Jamil (2005), considera-se o dado como um fato, código ou sinal que pode ser obtido por observação ou medição, mas, isoladamente, sua compreensão é complexa. Para que haja informação, é necessário contextualização e dados suficientes para uma análise.

Em síntese, podemos considerar que a disponibilização dos dados, meramente dispostos em uma tabela, não nos gera informação. É necessário todo um trabalho de extração, tratamento e carregamento dos dados para posteriormente ser possível realizar uma análise e contextualização, gerando a informação, para em seguida ser disponibilizado de maneira que seja compreensível e mais acessível à população.

O foco do trabalho também foram os dados de represas e mananciais gerenciados, monitorados e preservados pela SABESP. São cinco sistemas produtores de água sob sua supervisão: Rio Claro, Alto Cotia, Guarapiranga, Alto Tietê e Cantareira. A SABESP é responsável por tratar e distribuir água potável para mais de 14 milhões de moradores da capital e grande São Paulo (SABESP, s.d.). Essas 14 milhões de pessoas são diretamente afetadas, tanto positivamente quanto negativamente, pelas decisões tomadas sobre a gestão desses recursos. Por isso, é de suma importância que haja uma boa disponibilização dos dados e informações para que mais pessoas possam colaborar com *insights* e sugestões de melhorias nas mais diversas áreas de atuação da SABESP.

Desta forma, o trabalho a seguir realizou uma análise técnica com relação a maneira com que o acesso dos dados da SABESP se faz disponível à população. Tendo como base esta análise, desenvolvemos uma solução visando facilitar a democratização dos dados, tornando-o mais acessível a pesquisadores e estudiosos. Por fim, com o objetivo de atingir o público leigo, elaboramos um *dashboard* interativo, no qual é possível visualizar os dados referentes aos sistemas produtores da Sabesp.

2 Objetivos

2.1 Objetivos gerais

O objetivo geral do trabalho foi tornar o acesso aos dados de sistemas produtores da SABESP mais democrático, comprehensível e acessível por meio de uma API e *dashboard*. O resultado final desejado foi um produto de dados de simples acesso que permita a desenvolvedores, profissionais de dados e o público geral monitorarem os dados e construírem trabalhos com base neles.

2.2 Objetivos específicos

Os objetivos específicos deste trabalho foram:

- Verificar e identificar a forma com que a SABESP atualmente disponibiliza os dados e gera informação;
- Disponibilizar os dados atualizados dos sistemas produtores da SABESP em formatos padronizados e convenientes em plataformas como GitHub e Kaggle;
- Construir uma API de acesso aos dados para desenvolvedores;
- Construir um *dashboard* com os dados de sistemas produtores em gráficos interativos para o público geral;

3 Justificativa

A Companhia de Saneamento Básico do Estado de São Paulo (SABESP) disponibiliza dados sobre os sistemas produtores que abastecem o estado de São Paulo em um site próprio¹. A disponibilização desses dados é vital para a participação ativa da população na gestão de recursos hídricos, permitindo que o público geral monitore a situação em sua região e que desenvolvedores, profissionais de dados, acadêmicos e outros públicos especializados possam contribuir com novas ferramentas, soluções, modelos e *insights*.

O monitoramento desses dados é particularmente relevante no contexto de crises hídricas, como a crise de abastecimento de 2014 em São Paulo. Além dos problemas de saneamento básico e saúde pública, a crise gerou consequências econômicas devido à redução do ritmo de produção e queda de produtividade em indústrias, produzindo milhares de demissões (Redação RBA, 2014). Dado que o estado de São Paulo corresponde a um terço do Produto Interno Bruto do Brasil, a crise também se mostrou uma ameaça à recuperação econômica do país (Dezem, 2014).

Souza-Fernandes (2016) destaca a relevância da participação da sociedade na gestão de recursos hídricos:

A água é um problema de segurança nacional e como tal merece a adoção de estratégias direcionadas para cada um de seus aspectos particulares, todos eles de relevância para o desenvolvimento dos povos, aí compreendida a saúde pública. E todos devem se tornar partícipes nesta gestão. (SOUZA-FERNANDES, 2016, p. 95)

Considerando a participação da sociedade, há alguns problemas e pontos fracos no modo como a SABESP disponibiliza os dados de sistemas produtores, dificultando o acesso e uso dos dados. Do ponto de vista do público geral, os dados estão disponíveis apenas em formato tabular, o que não é conveniente para visualização, principalmente para o acompanhamento dos indicadores ao longo do tempo. Isso acaba sendo um ruído na questão da transparência e controle social,

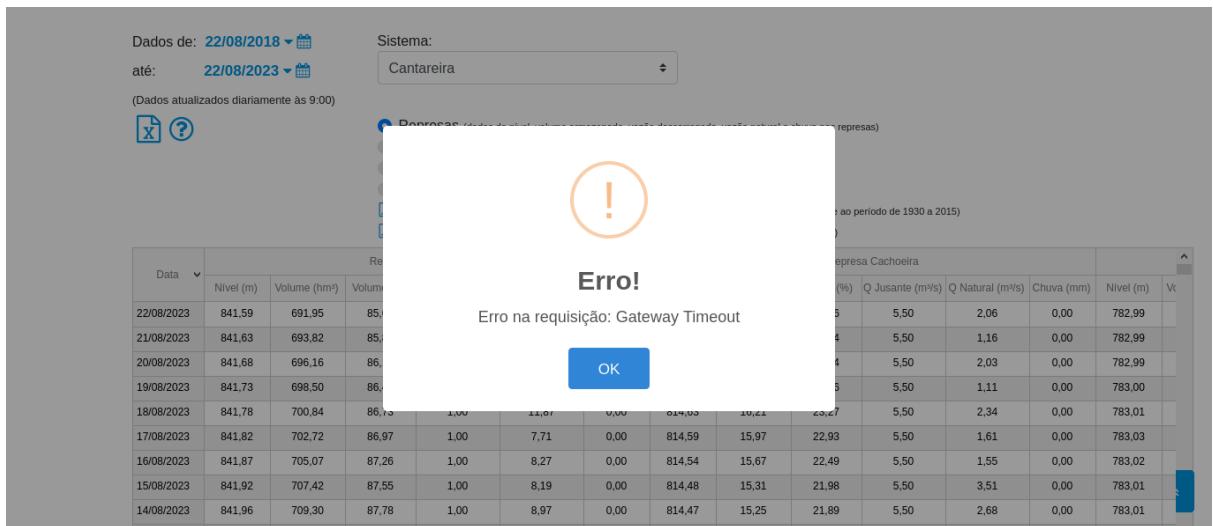
¹ <https://mananciais.sabesp.com.br/HistoricoSistemas?SistemaId=0>

uma vez que a disponibilização dos dados não se dá de maneira comprehensível e facilitada.

Do ponto de vista de desenvolvedores e profissionais de dados, identificamos três problemas:

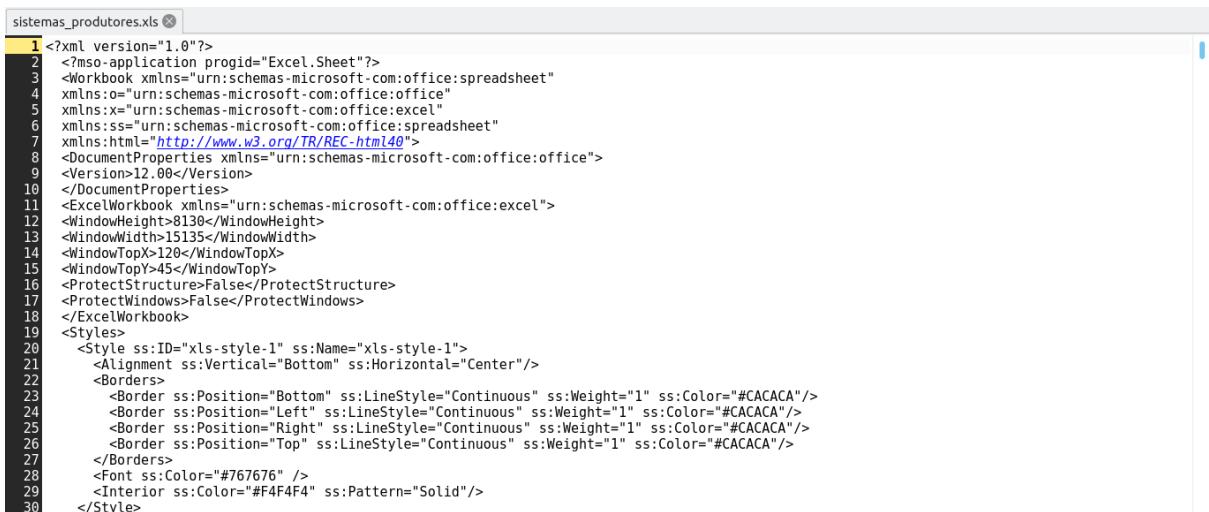
1. A seleção de intervalos grandes de tempo é lenta e pode falhar por completo, resultando em um *Gateway Timeout*, como no exemplo da figura 1;
2. Embora o arquivo com os dados exportados tenha a extensão .xls e possa ser aberto normalmente no Excel e outros editores de planilhas, ele é na verdade um arquivo XML (*Extensible Markup Language*) com uma estrutura própria, como exibido na figura 2. Isso dificulta a abertura e manipulação do arquivo em linguagens de programação como o *Python*. A figura 3 mostra uma tentativa inexitosa de abrir o arquivo exportado pelo portal da SABESP com a biblioteca *Pandas* para ilustrar essa dificuldade; o erro persiste independentemente da *engine* especificada, pois o formato não é reconhecido.
3. O formato da tabela com os dados por represa também é inconveniente para bibliotecas como o Pandas. Conforme mostra a figura 4, cada represa aparece como *header* e os dados de cada represa em um determinado dia aparecem em uma mesma linha. Normalmente é mais conveniente ter a represa como uma coluna, com dados de apenas uma represa por linha.

Figura 1 – Erro de *Gateway Timeout* na página da SABESP



Fonte: Portal dos Mananciais Sabesp².

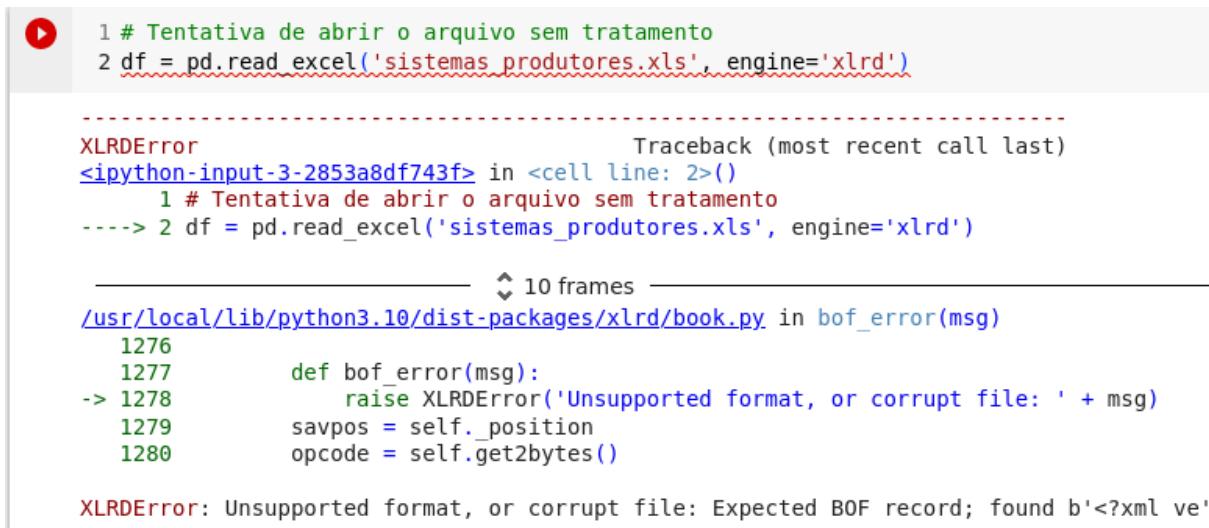
Figura 2 – Arquivo XML exportado na página da SABESP



```
sistemas_produtores.xls
1 <?xml version="1.0"?>
2 <?mso-application progid="Excel.Sheet"?>
3 <Workbook xmlns="urn:schemas-microsoft-com:office:spreadsheet"
4 xmlns:o="urn:schemas-microsoft-com:office:office"
5 xmlns:x="urn:schemas-microsoft-com:office:excel"
6 xmlns:ss="urn:schemas-microsoft-com:office:spreadsheet"
7 xmlns:html="http://www.w3.org/TR/REC-html40">
8 <DocumentProperties xmlns="urn:schemas-microsoft-com:office:office">
9 <Version>12.00</Version>
10 </DocumentProperties>
11 <ExcelWorkbook xmlns="urn:schemas-microsoft-com:office:excel">
12 <WindowHeight>8130</WindowHeight>
13 <WindowWidth>15135</WindowWidth>
14 <WindowTopX>120</WindowTopX>
15 <WindowTopY>45</WindowTopY>
16 <ProtectStructure>False</ProtectStructure>
17 <ProtectWindows>False</ProtectWindows>
18 </ExcelWorkbook>
19 <Styles>
20   <Style ss:ID="xls-style-1" ss:Name="xls-style-1">
21     <Alignment ss:Vertical="Bottom" ss:Horizontal="Center"/>
22     <Borders>
23       <Border ss:Position="Bottom" ss:LineStyle="Continuous" ss:Weight="1" ss:Color="#CACACA"/>
24       <Border ss:Position="Left" ss:LineStyle="Continuous" ss:Weight="1" ss:Color="#CACACA"/>
25       <Border ss:Position="Right" ss:LineStyle="Continuous" ss:Weight="1" ss:Color="#CACACA"/>
26       <Border ss:Position="Top" ss:LineStyle="Continuous" ss:Weight="1" ss:Color="#CACACA"/>
27     </Borders>
28     <Font ss:Color="#767676" />
29     <Interior ss:Color="#F4F4F4" ss:Pattern="Solid"/>
30   </Style>
```

Fonte: Elaborada pelos autores (2023).

Figura 3 – Tentativa de importar os dados da SABESP com a biblioteca Pandas



```
# Tentativa de abrir o arquivo sem tratamento
df = pd.read_excel('sistemas_produtores.xls', engine='xlrd')

-----
XLRError                                     Traceback (most recent call last)
<ipython-input-3-2853a8df743f> in <cell line: 2>()
      1 # Tentativa de abrir o arquivo sem tratamento
----> 2 df = pd.read_excel('sistemas_produtores.xls', engine='xlrd')

----- 10 frames -----
/usr/local/lib/python3.10/dist-packages/xlrd/book.py in bof_error(msg)
1276
1277     def bof_error(msg):
-> 1278         raise XLRError('Unsupported format, or corrupt file: ' + msg)
1279         savpos = self._position
1280         opcode = self.get2bytes()

XLRError: Unsupported format, or corrupt file: Expected BOF record; found b'<?xml ve'
```

Fonte: Elaborada pelos autores (2023).

Figura 4 – Tabulação original dos dados disponibilizados pela SABESP

² <https://mananciais.sabesp.com.br/HistoricoSistemas?SistemaId=0>

Data	Represa Jaguari/Jacareí						Represa Cachoeira					
	Nível (m)	Volume (hm³)	Volume (%)	Q Jusante (m³/s)	Q Natural (m³/s)	Chuva (mm)	Nível (m)	Volume (hm³)	Volume (%)	Q Jusante (m³/s)	Q Natural (m³/s)	Chuva (mm)
21/08/2023	841,63	693,82	85,86	1,00	5,05	0,00	814,65	16,33	23,44	5,50	1,16	0,00
20/08/2023	841,68	696,16	86,15	1,00	4,81	0,00	814,65	16,33	23,44	5,50	2,03	0,00
19/08/2023	841,73	698,50	86,44	1,00	5,68	0,00	814,64	16,27	23,36	5,50	1,11	0,00
18/08/2023	841,78	700,84	86,73	1,00	11,87	0,00	814,63	16,21	23,27	5,50	2,34	0,00
17/08/2023	841,82	702,72	86,97	1,00	7,71	0,00	814,59	15,97	22,93	5,50	1,61	0,00
16/08/2023	841,87	705,07	87,26	1,00	8,27	0,00	814,54	15,67	22,49	5,50	1,55	0,00
15/08/2023	841,92	707,42	87,55	1,00	8,19	0,00	814,48	15,31	21,98	5,50	3,51	0,00
14/08/2023	841,96	709,30	87,78	1,00	8,97	0,00	814,47	15,25	21,89	5,50	2,68	0,00
13/08/2023	842,00	711,19	88,01	1,00	6,18	0,20	814,46	15,19	21,81	5,50	1,33	0,80
12/08/2023	842,05	713,55	88,31	1,00	7,74	0,00	814,43	15,01	21,55	5,50	1,02	4,80
11/08/2023	842,10	715,92	88,60	1,00	4,50	0,00	814,38	14,71	21,13	5,50	0,80	0,00
10/08/2023	842,15	718,28	88,89	1,00	5,27	0,00	814,37	14,65	21,04	5,50	1,23	0,00
09/08/2023	842,20	720,65	89,18	1,00	5,03	0,00	814,35	14,54	20,87	5,50	1,02	0,00

Fonte: Portal dos Mananciais Sabesp³.

4 Referencial teórico

4.1 Catalogação de referências

Buscamos três referências em bases de dados para nortear o trabalho, sobretudo na parte técnica, nas bases de dados do Google Acadêmico.

Ficha de Catalogação da Pesquisa Bibliográfica				
Base de dados/Biblioteca Digital	Google Acadêmico			
Estratégia de busca (OR, NOT, AND)	Python AND Tratamento e Análise de dados			
Quantidade de registros	Identificados:	172	Relevantes:	3
Trabalhos Relevantes				
Título	Python para análise de dados: Tratamento de dados com pandas, numpy e ipython			
Autor	Wes Mckinney			
Tipo	Livro técnico especializado			
Link				

³ <https://mananciais.sabesp.com.br/HistoricoSistemas?SistemaId=0>

Título	Análise de dados com Python e Pandas
Autor	Daniel Y. Chen
Tipo	Livro técnico especializado
Link	
Título	Python na análise de dados: estudo de caso com dados de acidentes aéreos no Brasil
Autor	Philipe de Araujo Fernandes Formigoni
Tipo	Trabalho de conclusão de curso
Link	https://app.uff.br/riuff/handle/1/23160

Tomamos como referencial teórico os três livros aqui elencados com foco em tratamento e análise de dados utilizando a linguagem *Python*.

Wes McKinney, escritor do livro “*Python para análise de dados*” é uma figura proeminente na comunidade de análise de dados com *Python*, colaborandoativamente e de forma voluntária em melhorias e desenvolvimento de softwares open sources tendo como foco a computação analítica (MCKINNEY, 2023). Ele é o criador da biblioteca *Pandas*, uma das mais conhecidas e utilizadas bibliotecas para análise e tratamento de dados. Além de utilizar a biblioteca pandas em seu livro, ele também faz uso de outras bibliotecas *Python* para a melhor manipulação e tratamento dos dados, tais como: *Numpy*, *Matplotlib*, *Scikit-learn*, entre outras. De maneira prática e didática, o livro aborda desde as etapas de carga e armazenagem de dados até a visualização e modelagem de dados, incluindo também um capítulo com exemplos práticos com dados reais.

Daniel Y Chen é um engenheiro de dados e pesquisador no Laboratório de Decisões Análíticas e Sociais do Instituto de Biocomplexidade da *Virginia Tech* nos Estados Unidos da América, além de professor de Ciências de Dados na plataforma educacional *DataCamp*. Seu livro, “Análise de dados com *Python* e *Pandas*”, possui a mesma temática e abordagem da nossa primeira referência, mas possui um foco

maior e abrangente quando se trata de tratamento e visualização de dados, tornando-se um incremento em nossas pesquisas de base teórica e prática.

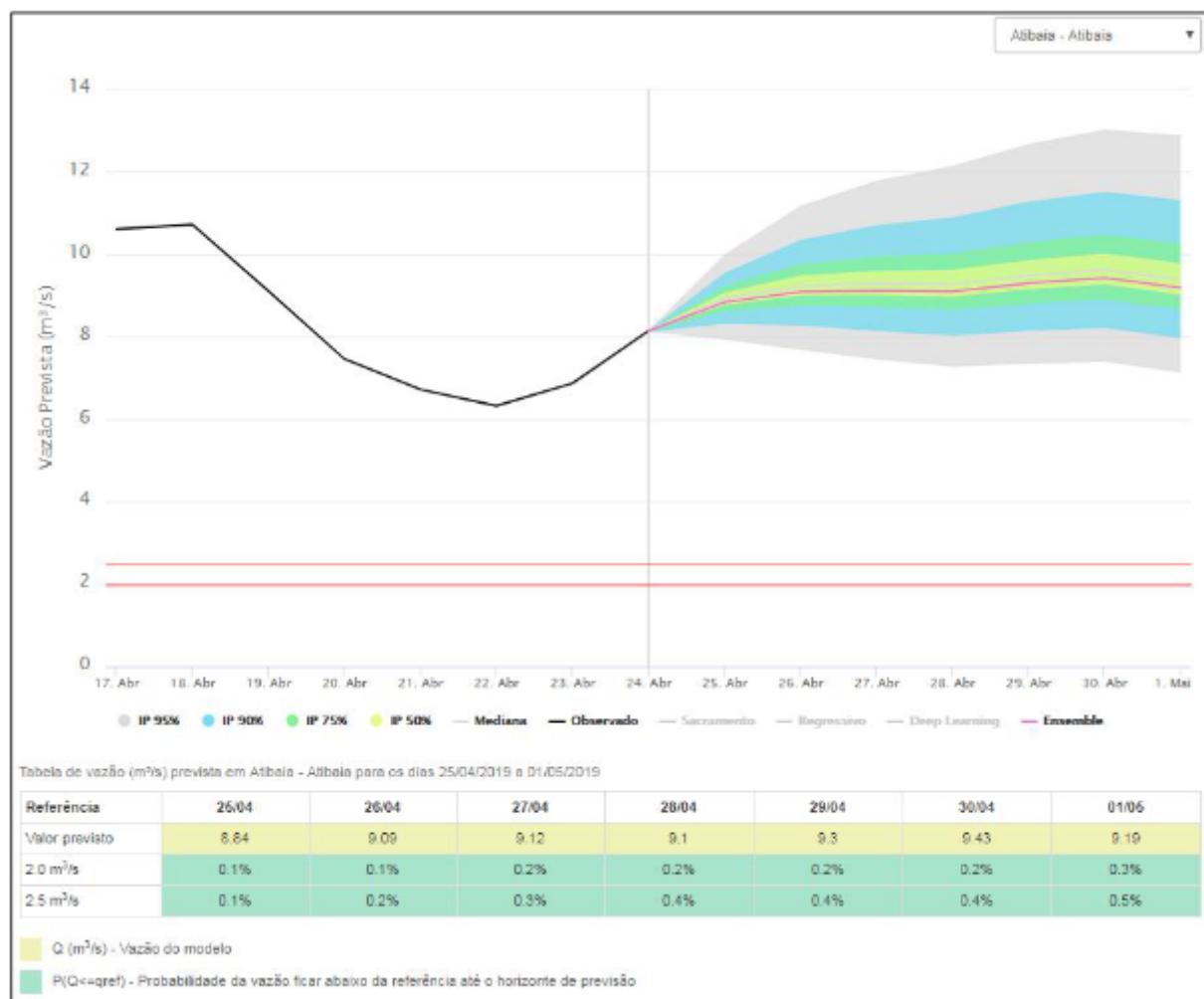
No Trabalho de Conclusão de Curso do curso de Engenharia de Produção da Universidade Federal Fluminense, Philipe Formigoni desenvolve uma análise de dados fazendo uso de Python e da metodologia CRISP-DM (*Cross Industry Standard Process for Data Mining*). Por se tratar de um Trabalho de Conclusão de Curso recente e com uma proposta próxima de nosso trabalho, achamos relevante considerar algumas das abordagens e aprendizados de Formigoni para enriquecer nossa análise de dados.

4.2 Trabalhos relacionados

Além do referencial teórico usado como base, também pesquisamos trabalhos relacionados à disponibilização de dados da SABESP na literatura, em repositórios do GitHub e na internet em geral usando palavras-chave como “SABESP API”, “SABESP dashboard”, “SABESP visualização”, etc. Identificamos quatro trabalhos relevantes que estudamos para entender o que já foi feito, quais métodos foram empregados, quais são suas limitações e como o nosso trabalho pode se diferenciar.

Almeida et. al (2019) descrevem o Sistema de Previsão Hidrometeorológico das Bacias PCJ (SPHM-PJC), um sistema de visualização e previsão disponível para consulta dos técnicos da Agência das Bacias PCJ e da CT-MH dos Comitês PCJ. O sistema não é disponível para o público geral, além de ser restrito apenas às Bacias PCJ do Sistema Cantareira. A figura 5 mostra uma visualização do sistema.

Figura 5 – Gráfico mostrando a vazão observada e prevista de um ponto de controle no SPHM-PJC



Fonte: Almeida et. al (2019).

O app Sabesp Mananciais RMSP⁴ é um aplicativo móvel de monitoramento dos mananciais da Região Metropolitana de São Paulo disponibilizado pela SABESP ao público geral para dispositivos Android e iOS. Uma limitação é não haver uma versão web que funciona em qualquer dispositivo, o que diminui a acessibilidade. Também não é possível especificar um intervalo de tempo.

Presbiteris (2015) apresenta uma API que disponibiliza os dados dos reservatórios da SABESP. Contudo, além do fato de que a API não está mais disponível, ela disponibiliza apenas os dados de um dia por vez, não permitindo o acesso aos dados de um determinado intervalo. A API também parece disponibilizar apenas os dados do Sistema Cantareira.

⁴ <https://mananciais.sabesp.com.br/appsabesp>

Milz (s.d.) apresenta um pacote escrito na linguagem R que disponibiliza os dados de volume e pluviometria dos sistemas monitorados pela SABESP. Embora a base de dados completa esteja disponível para *download*, só é possível fazer requisições caso o desenvolvedor use a linguagem R. Também não há dados de reservatórios específicos.

Além desses quatro trabalhos, mencionamos também o trabalho desenvolvido por alguns dos integrantes deste grupo como parte da disciplina Projeto Integrador IV, cujo foco foi a modelagem preditiva do Sistema Cantareira (como descrito no anexo A). Uma das dificuldades enfrentadas foi a etapa de extração e tratamento dos dados, que demandou mais tempo do que o previsto e foi uma das motivações para este trabalho. Também não conseguimos implementar a atualização dos dados e das previsões diariamente de acordo com os novos dados fornecidos pela SABESP devido à limitação de tempo.

4.3 Diferencial competitivo

Analizando os trabalhos relacionados, notamos algumas lacunas que nosso trabalho poderia preencher. Por exemplo, alguns dos sistemas não estão disponíveis ao público, como o descrito por Almeida et. al (2019), ou não são mais funcionais, como o descrito por Presbiteris (2015). Além disso, de modo geral, os projetos são focados na disponibilização de uma API ou na visualização dos dados, mas não em ambos.

Como diferencial competitivo, decidimos disponibilizar dados completos e atualizados via API, incluindo dados dos sistemas e dos reservatórios. Esses dados são disponibilizados via requisições HTTP à API, permitindo que desenvolvedores accessem os dados através de qualquer linguagem de programação. Em relação aos usuários, decidimos disponibilizar o *dashboard* via ambiente *web*, tornando-o acessível através de qualquer dispositivo (ao contrário, por exemplo, do aplicativo da SABESP, que não é compatível com todos os dispositivos).

O projeto desenvolvido como parte do Projeto Integrador IV (presente no anexo A) mostra uma das potenciais aplicações deste trabalho, pois a existência de um repositório ou API com os dados da SABESP atualizados agiliza o desenvolvimento de projetos de análise de dados e aprendizado de máquina, permitindo que pesquisadores e profissionais da área foquem na modelagem em vez da extração e tratamento dos dados.

5 Metodologia

Nosso trabalho passou pelas etapas de revisão bibliográfica (descrita anteriormente), coleta dos dados de sistemas produtores da SABESP, construção do *dashboard* e construção da API. Como metodologia para o desenvolvimento do produto final, optamos pelo *design thinking*.

Segundo Brown e Wyatt (2010), o *design thinking* prevê processos interativos formados por três etapas, não necessariamente sequenciais: inspiração, ideação e implementação. A inspiração começa com os requisitos básicos do projeto (por exemplo, segmento de mercado e tecnologias disponíveis) e parte para o entendimento das necessidades do público-alvo. A ideação consiste na geração de ideias para o desenvolvimento do produto ou à resolução do problema. Na implementação, elabora-se um protótipo que é testado e refinado até a construção do produto final. A vantagem do *design thinking* sobre os processos tradicionais é que as etapas não são rigidamente definidas e abrem espaço para a criatividade e *feedback* contínuo.

O grupo seguiu as etapas de inspiração e ideação para definir o tema, os objetivos e os problemas que desejamos resolver através de discussões internas e conversas com a orientadora. Em seguida, desenvolvemos um protótipo do *dashboard* com uma ideia inicial do *design*. A figura 6 mostra a página inicial do protótipo, exibindo uma mensagem introdutória, com um menu lateral que permite ao usuário selecionar um sistema, acessar mais informações sobre os dados e ser direcionado à documentação da API.

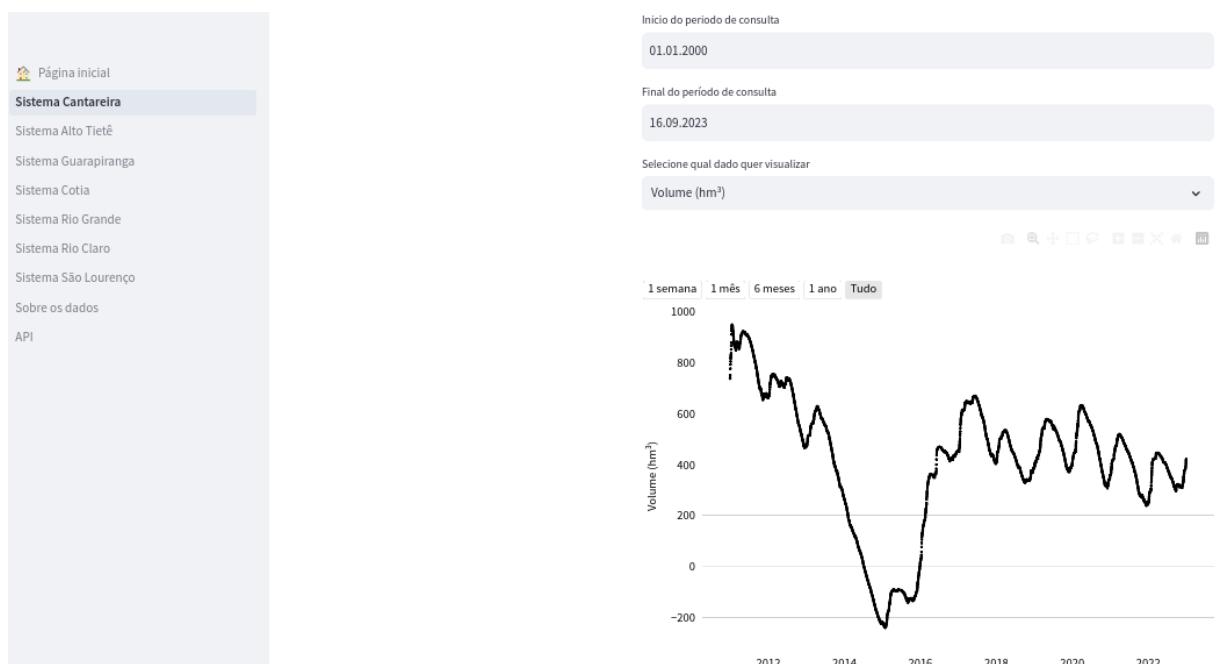
Figura 6 – Página inicial do protótipo do *dashboard*



Fonte: Elaborada pelos autores (2023).

A figura 7 mostra a página do Sistema Cantareira, que é semelhante às outras páginas de sistemas produtores; nela, o usuário pode selecionar um período de consulta e qual dado deseja visualizar, gerando um gráfico interativo. Logo abaixo, como mostra a figura 8, o usuário encontra os dados em formato tabular e pode fazer *download* dos mesmos em formato CSV.

Figura 7 – Página do Sistema Cantareira no protótipo do *dashboard*



Fonte: Elaborada pelos autores (2023).

Figura 8 – Tabela com os dados selecionados e opção de download no protótipo do dashboard

	Data	Volume (hm³)	Volume (%)	Chuva (mm)	Vazão natural (m³/s)	Vazão a jusante (m)
0	2011-01-01 00:00:00	735.9763	74.9412	0.1	30.346	
1	2011-01-02 00:00:00	739.4875	75.2987	36.25	70.475	
2	2011-01-03 00:00:00	749.6823	76.3368	50.1	147.125	4
3	2011-01-04 00:00:00	774.7399	78.8883	43.95	321.239	2
4	2011-01-05 00:00:00	791.3338	80.578	8.2	223.21	
5	2011-01-06 00:00:00	802.947	81.7605	13.35	168.225	
6	2011-01-07 00:00:00	814.6222	82.9493	13.15	168.489	
7	2011-01-08 00:00:00	822.6097	83.7627	3.8	125.485	2
8	2011-01-09 00:00:00	829.3022	84.4441	8.9	110.511	
9	2011-01-10 00:00:00	834.6583	84.9895	9.4	95.058	

[Baixar os dados como .csv](#)

Fonte: Elaborada pelos autores (2023).

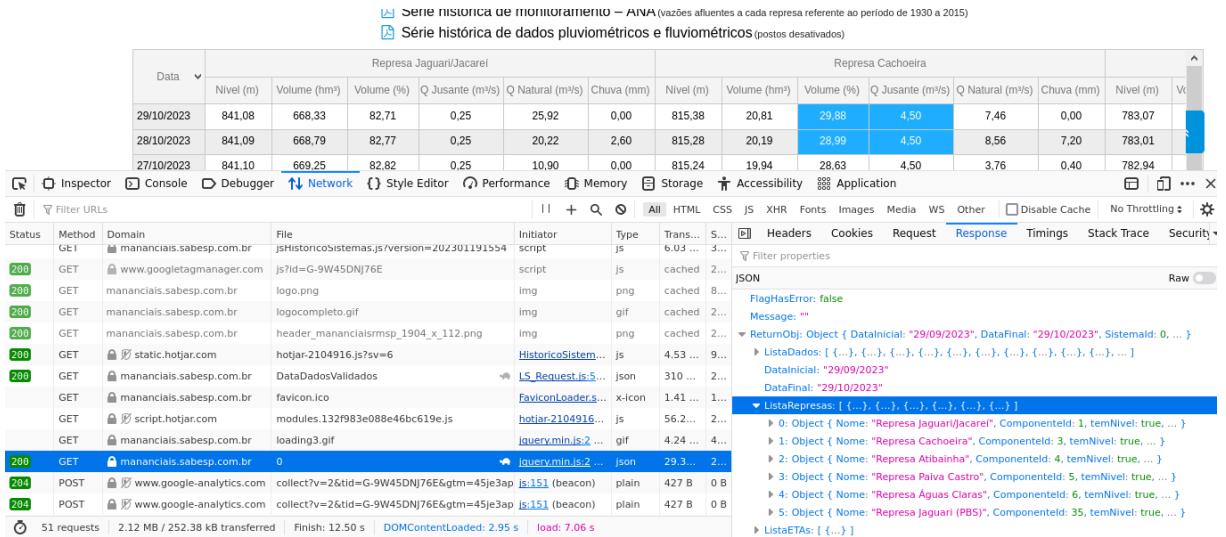
Os próximos passos do trabalho foram o refinamento sucessivo do protótipo e a implementação dos requisitos técnicos necessários para a validação final do trabalho.

5.1 Coleta de dados

O processo de coleta de dados foi conduzido por meio da prática de *web scraping*, um método amplamente adotado no âmbito da recuperação de informações online. De acordo com Mitchell (2019), o *web scraping* envolve o uso de programas automatizados para acessar páginas da web, requisitar dados e extrair informações. Essa técnica ganhou popularidade na primeira década dos anos 2000 com a popularização de navegadores baseados em um *web-kits opensource* e o surgimento de frameworks como o BeautifulSoup, os quais possibilitam a análise de arquivos HTML para a extração de dados de maneira mais eficiente.

Explorando o portal da SABESP com as ferramentas do navegador, notamos que a página faz uma requisição ao servidor para obter os dados, como exibido na figura 9. Assim, para simplificar a extração dos dados, criamos um programa que simula as requisições feitas pelo portal. Caso o modo como o portal obtenha os dados mude no futuro, podemos alterar o código ou usar um *framework* de automação como o Selenium.

Figura 9 – Requisição do portal da SABESP aos dados dos sistemas.



Fonte: Elaborada pelos autores (2023).

Compilamos todos os dados desde o ano 2000 e implementamos uma rotina de extração diária através do GitHub Actions⁵, uma plataforma que permite a execução automática de tarefas de acordo com as condições impostas. Configuramos um *script* para verificar novos dados aproximadamente às 09:00, 10:00, 15:00 e 20:00 no horário de Brasília como medida de contingência, já que o *script* pode falhar ou dados podem não estar disponíveis no horário indicado. Não há custos associados à extração, já que o plano gratuito do GitHub é suficiente para este projeto.

5.2 Tratamento dos dados

A etapa de tratamento de dados é fundamental para todo projeto de análise de dados. Segundo Costa (2020), é uma das etapas mais demoradas de um projeto, que envolve atividades como limpeza, transformação e até a criação de novas variáveis com a finalidade de nos trazer mais informações de maneira eficiente. Costa (2020) também ressalta a importância desta etapa pelo fato de que, ao trabalharmos com diferentes fontes de dados, é comum nos depararmos

⁵ <https://github.com/features/actions>

com erros, falta de padronização, valores nulos, redundâncias, etc. McKinney, também comenta que:

Durante a análise e a modelagem dos dados, um período significativo de tempo é gasto em sua preparação: carga, limpeza, transformação e reorganização. Sabe-se que essas tarefas em geral ocupam 80% ou mais do tempo de um analista. Às vezes, o modo como os dados são armazenados em arquivos ou em banco de dados não constituem um formato correto para uma tarefa em particular (MCKINNEY, 2017, p.358).

Como discutido anteriormente e ilustrado na figura 2, o arquivo exportado pelo site da SABESP está no formato XML e precisa passar por algum tratamento para que os dados possam ser lidos. A figura 10 mostra o código usado para esse tratamento, que recebe um caminho de arquivo e retorna um objeto *DataFrame* da biblioteca de manipulação de dados *Pandas*. De acordo com MCKINNEY(2017), um *DataFrame* :

[...] representa uma tabela de dados retangular e contém uma coleção ordenada de colunas, em que cada uma pode ter um tipo de valor diferente(numérico, string, booleano, etc.) O DataFrame tem índice tanto para linha quanto para coluna; pode ser imaginado como um dicionário de Series, todos compartilhando o mesmo índice. Internamente, os dados são armazenados como um ou mais blocos dimensionais em vez de serem armazenados como uma lista, um dicionário ou outra coleção de arrays unidimensionais(MCKINNEY, 2017, p. 248)

Figura 10 – Código de tratamento dos arquivos exportados pelo site da SABESP

```
1 def parse_xml(filepath):
2     df = None
3     with open(filepath,'r') as xml_file:
4         soup = BeautifulSoup(xml_file.read(), 'xml')
5         for sheet in soup.findAll('Worksheet'):
6             sheet_as_list = []
7             for row in sheet.findAll('Row'):
8                 sheet_as_list.append([cell.Data.text if cell.Data else '' for cell in row.findAll('Cell')])
9             df = pd.DataFrame(sheet_as_list)
10            df.columns = df.iloc[1]
11            df.drop([0, 1], inplace=True)
12            df.reset_index(drop=True, inplace=True)
13            df['Data'] = pd.to_datetime(df['Data'], dayfirst=True)
14    return df
```

Fonte: Elaborada pelos autores (2023).

No caso das represas individuais, devido ao formato inconveniente apontado na figura 3, precisamos acrescentar uma coluna “Represa” indicando a

represa à qual os dados se referem. A figura 11 mostra o resultado desse tratamento para alguns registros referentes a represas do Sistema Cantareira, em um arquivo CSV.

Figura 11 – Amostra dos dados tratados de represas do Sistema Cantareira

	Data	Represa	Nível (m)	Volume (hm³)	Volume (%)	Q Jusante (m³/s)	Q Natural (m³/s)	Chuva (mm)
1	2023-04-01	Jaguari/Jacareí	842.73	745.9688920815145	92.31790709462861	0.25	21.332377776593635	0
2	2023-04-01	Cachoeira	817.48	34.799797145691024	49.964276510791905	0.5	3.6913948468352515	11
3	2023-04-01	Atibainha	782.92	19.12603995065382	19.870760289778314	1,5	3.32554589490428	11.2
4	2023-04-01	Paiva Castro	744.13	1.2915293135742445	16.966837071325614	0,1	1.5.697079795496492	13
5	2023-04-02	Jaguari/Jacareí	842.73	745.9688920815145	92.31790709462861	0.25	31.25	0
6	2023-04-02	Cachoeira	817.53	35.15359675558181	50.47224905626355	0.5	2.3889028922544613	0.2
7	2023-04-02	Atibainha	782.98	20.25533066764399	21.04402277342258	1,5	9.62494409608443	0.4
8	2023-04-02	Paiva Castro	744.11	1.212089003448309	15.923228703605167	0,1	3.313551966135004	0.2
9	2023-04-03	Jaguari/Jacareí	842.72	745.4879739395055	92.25839073030647	0.25	25.18381780082255	0
10	2023-04-03	Cachoeira	817.57	35.43735617791514	50.87966046100965	0.5	2.115252573302463	0
11	2023-04-03	Atibainha	783.02	21.009755445188034	21.827822971996074	1,1	1.5307682586116336	0
12	2023-04-03	Paiva Castro	744.12	1.2517901789658765	16.444783552935306	0,1	3.2245043462681444	0
13	2023-04-04	Jaguari/Jacareí	842.71	745.00718025165	92.19888976789599	0.25	25.185258242412424	0
14	2023-04-04	Cachoeira	817.6	35.65059606869297	51.18582250044192	0.5	1.3480542914100466	0
15	2023-04-04	Atibainha	783.08	22.143738665180933	23.005960673017636	1,3	9.9888057869548526	0
16	2023-04-04	Paiva Castro	744.12	1.2517901789658765	16.444783552935306	0,1	3.5989999999999999	0
17	2023-04-05	Jaguari/Jacareí	842.7	744.5265110539269	92.13940421184978	0.25	24.886699100427712	0
18								

Fonte: Elaborada pelos autores (2023).

Adicionalmente, como é impossível extrair todos os dados de uma vez por limitação do site da SABESP, foi preciso criar um *DataFrame* por ano e posteriormente foi realizada sua concatenação. Para melhor visualização dos dados no arquivo CSV bruto, realizou-se a ordenação dos dados tendo como parâmetro a coluna ‘Data’. A figura 12 mostra um exemplo de código que concatena e ordena três *DataFrames* com dados do Sistema Cantareira de 2011 a 2023 e o *DataFrame* resultante.

Figura 12 – Concatenação dos Dataframes e ordenação do DataFrame resultante

```

1 df_final = pd.concat([df1, df2, df3])
2 df_final.sort_values(by=['Data'], inplace=True)
3 df_final

```

	Data	Volume (hm³)	Volume (%)	Chuva (mm)	Vazão natural (m³/s)	Vazão a jusante (m³/s)
1460	2011-01-01	735.97625	74.94117	0.1	30.346	5.5
1459	2011-01-02	739.48751	75.29871	36.25	70.475	5.5
1458	2011-01-03	749.68231	76.3368	50.1	147.125	4.48
1457	2011-01-04	774.73991	78.8883	43.95	321.239	2.27
1456	2011-01-05	791.33379	80.57798	8.2	223.21	2
...
4	2022-12-28	391.29407	39.84373	29.7	60.422	1.17
3	2022-12-29	399.4439	40.67359	24.5	114.401	0.85
2	2022-12-30	407.63398	41.50754	38.6	115.444	0.85
1	2022-12-31	414.72548	42.22964	12.7	104.118	0.85
0	2023-01-01	420.23007	42.79015	10.05	86.009	0.85

4384 rows × 6 columns

Fonte: Elaborada pelos autores (2023).

Com todos os dados desde 2000 até a data da extração compilados em arquivos CSV, verificamos a consistência e integridade dos dados e realizamos tratamentos adicionais quando necessário. Um exemplo desse tratamento é o que pode ser percebido na figura 13, na qual foi utilizada a função *info* para obter a descrição dos formatos de dados contidos no *DataFrame*.

Figura 13 - Verificação dos tipos do DataFrame

▼ Verificação dos tipos

```

[ ] df_alto_tiete.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8684 entries, 0 to 8683
Data columns (total 6 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Data              8684 non-null   object  
 1   Volume (hm³)      8684 non-null   float64
 2   Volume (%)        8684 non-null   float64
 3   Chuva (mm)        6857 non-null   float64
 4   Vazão natural (m³/s) 8684 non-null   float64
 5   Vazão a jusante (m³/s) 8684 non-null   float64
dtypes: float64(5), object(1)
memory usage: 407.2+ KB

```

Fazer conversão das colunas de data e verificar novamente para todos os DataFrames.

Fonte: Elaborada pelos autores (2023).

Após essa verificação foi preciso, então, realizar a conversão dos dados na coluna ‘Data’, que se apresentou no formato *object*. Como podemos observar na figura 14, foi realizada a conversão da coluna para o formato adequado através da função *datetime* e, posteriormente, uma nova verificação para confirmar a efetividade da ação.

Figura 14- Conversão da coluna ‘Data’

```
[ ] dfs = [df_alto_tiete, df_alto_tiete_res, df_cantareira, df_cantareira_res,\n         df_cotia, df_cotia_res, df_guarapiranga, df_guarapiranga_res, df_rio_claro,\n         df_rio_claro_res, df_rio_grande, df_rio_grande_res, df_sao_lourenco,\n         df_sao_lourenco_res]\n\n[ ] for df in dfs:\n    df['Data'] = pd.to_datetime(df['Data'], dayfirst=True)\n    df.info()\n\n[ ] <class 'pandas.core.frame.DataFrame'>\nRangeIndex: 8684 entries, 0 to 8683\nData columns (total 6 columns):\n #   Column           Non-Null Count  Dtype \n--- \n 0   Data             8684 non-null   datetime64[ns]\n 1   Volume (hm³)     8684 non-null   float64\n 2   Volume (%)       8684 non-null   float64\n 3   Chuva (mm)       6857 non-null   float64\n 4   Vazão natural (m³/s) 8684 non-null   float64\n 5   Vazão a jusante (m³/s) 8684 non-null   float64\n dtypes: datetime64[ns](1), float64(5)\n memory usage: 407.2 KB\n<class 'pandas.core.frame.DataFrame'>\nRangeIndex: 48048 entries, 0 to 48047\nData columns (total 8 columns):\n #   Column           Non-Null Count  Dtype \n--- \n 0   Data             48048 non-null   datetime64[ns]\n 1   Nível            46061 non-null   float64\n 2   Volume (hm³)     39606 non-null   float64\n 3   Volume (%)       39606 non-null   float64\n 4   Chuva (mm)       41593 non-null   float64\n 5   Vazão natural (m³/s) 39606 non-null   float64\n 6   Vazão a jusante (m³/s) 39639 non-null   float64\n 7   Reservatório     48048 non-null   object\n dtypes: datetime64[ns](1), float64(6), object(1)\n memory usage: 2.9+ MB
```

Fonte: Elaborada pelos autores (2023).

Assim como descrito por Costa (2020) em toda análise de dados se faz necessário a verificação da presença de dados ausentes para que possamos garantir “qualidade, completude, veracidade e integridade dos fatos que aqueles

dados representem”. Tal etapa pode ser percebida na figura 14, onde podemos identificar na coluna “*Non-null Count*” a contagem das linhas que possuem valores não nulos em seu conteúdo.

Chen (2018) cita que dificilmente encontraremos um conjunto de dados sem dados ausentes ou nulos, sendo que elas podem se apresentar como *null*, *nan*, *na* e até como strings vazias ‘’. Desta forma, o tratamento dos valores nulos dependerá de uma análise específica dos dados, buscando identificar os motivos, problemas ou possíveis distorções (MCKINNEY, 2017, p. 360). Especificamente ao nosso projeto, após análise de cada caso, foi verificado a ausência de valores nulos, conforme demonstra a figura 15.

Figura 15 - Contagem dos valores nulos.

```
df_nulls.isnull().sum()
```

```
VrNivel      0
VlVolumeHm   0
VrVolume%    0
VlQJusante   0
VlQNatural   0
VrChuva     0
DdDia        0
MmMes        0
AaAno        0
dtype: int64
```

Fonte: elaborada pelos autores (2023).

Para que não haja duplicidade de dados, foram utilizadas as funções *duplicate* e *drop_duplicate*, sendo o primeiro para verificar se há duplicidade e o segundo para remoção dos dados duplicados (MCKINNEY, 2017). Na figura 16, foi realizado a verificação de duplicidade nos nomes e valores das colunas e na figura 17 temos o desenvolvimento de uma função para nos retornar a quantidade de linhas do *DataFrame* original e a quantidade de linhas do *DataFrame* após verificação e remoção de dados duplicados. Desta forma, temos ciência da quantidade de dados duplicados e removidos.

Figura 16 - Verificação de duplicidade nos nomes e valores das colunas

```
[ ] # Verificar se não há nomes de colunas duplicados
for df in dfs:
    colunas_nomes_iguais = (df.columns.duplicated() == False).all()
    if colunas_nomes_iguais:
        print(f'Não há nomes de colunas duplicados em {df.name}')
    else:
        print(f'Há nomes de colunas duplicados em {df.name}')

Não há nomes de colunas duplicados em df_alto_tiete
Não há nomes de colunas duplicados em df_alto_tiete_res
Não há nomes de colunas duplicados em df_cantareira
Não há nomes de colunas duplicados em df_cantareira_res
Não há nomes de colunas duplicados em df_cotia
Não há nomes de colunas duplicados em df_cotia_res
Não há nomes de colunas duplicados em df_guarapiranga
Não há nomes de colunas duplicados em df_guarapiranga_res
Não há nomes de colunas duplicados em df_rio_claro
Não há nomes de colunas duplicados em df_rio_claro_res
Não há nomes de colunas duplicados em df_rio_grande
Não há nomes de colunas duplicados em df_rio_grande_res
Não há nomes de colunas duplicados em df_sao_lourenco
Não há nomes de colunas duplicados em df_sao_lourenco_res
```

```
# Verificar coluna por coluna para verificar se há colunas com valores iguais

for df in dfs:
    colunas_iguais = False
    for i in range(len(df.columns)):
        for j in range(i + 1, len(df.columns)):
            if (df.iloc[:, i] == df.iloc[:, j]).all():
                colunas_iguais = True
                print(f'As colunas de índice {i} e {j} têm os mesmos valores em {df.name}')

    if not colunas_iguais:
        print(f'Não há colunas com os mesmos valores em {df.name}')
```

```
Não há colunas com os mesmos valores em df_alto_tiete
Não há colunas com os mesmos valores em df_alto_tiete_res
Não há colunas com os mesmos valores em df_cantareira
Não há colunas com os mesmos valores em df_cantareira_res
Não há colunas com os mesmos valores em df_cotia
Não há colunas com os mesmos valores em df_cotia_res
Não há colunas com os mesmos valores em df_guarapiranga
Não há colunas com os mesmos valores em df_guarapiranga_res
Não há colunas com os mesmos valores em df_rio_claro
Não há colunas com os mesmos valores em df_rio_claro_res
Não há colunas com os mesmos valores em df_rio_grande
Não há colunas com os mesmos valores em df_rio_grande_res
Não há colunas com os mesmos valores em df_sao_lourenco
Não há colunas com os mesmos valores em df_sao_lourenco_res
```

Fonte: Elaborada pelos autores (2023).

Figura 17 - Verificação e remoção de duplicidade.

```
[ ] # Criar DataFrame com registros duplicados removidos
for df in dfs:
    df_sem_duplicadas = df.drop_duplicates()
    print(f'{df.name}: O DataFrame original tem tamanho {len(df)}. O DataFrame sem duplicadas tem tamanho {len(df_sem_duplicadas)}.')

df_alto_tiete: O DataFrame original tem tamanho 8684. O DataFrame sem duplicadas tem tamanho 8684.
df_alto_tiete_res: O DataFrame original tem tamanho 48048. O DataFrame sem duplicadas tem tamanho 48048.
df_cantareira: O DataFrame original tem tamanho 8684. O DataFrame sem duplicadas tem tamanho 8684.
df_cantareira_res: O DataFrame original tem tamanho 51993. O DataFrame sem duplicadas tem tamanho 51993.
df_cotia: O DataFrame original tem tamanho 8684. O DataFrame sem duplicadas tem tamanho 8684.
df_cotia_res: O DataFrame original tem tamanho 26020. O DataFrame sem duplicadas tem tamanho 26020.
df_guarapiranga: O DataFrame original tem tamanho 8684. O DataFrame sem duplicadas tem tamanho 8684.
df_guarapiranga_res: O DataFrame original tem tamanho 26007. O DataFrame sem duplicadas tem tamanho 26007.
df_rio_claro: O DataFrame original tem tamanho 8684. O DataFrame sem duplicadas tem tamanho 8684.
df_rio_claro_res: O DataFrame original tem tamanho 8684. O DataFrame sem duplicadas tem tamanho 8684.
df_rio_grande: O DataFrame original tem tamanho 8684. O DataFrame sem duplicadas tem tamanho 8684.
df_rio_grande_res: O DataFrame original tem tamanho 14042. O DataFrame sem duplicadas tem tamanho 14042.
df_sao_lourenco: O DataFrame original tem tamanho 8591. O DataFrame sem duplicadas tem tamanho 8591.
df_sao_lourenco_res: O DataFrame original tem tamanho 8684. O DataFrame sem duplicadas tem tamanho 8684.
```

Conclusão: não há registros idênticos.

Fonte: Elaborada pelos autores (2023).

Por fim, foi verificado também a existência de *outliers*, dados fora do padrão ou discrepantes, em nossos conjuntos de dados. Acerca dos *outliers*, Costa (2020), diz que:

[...] outliers são instâncias ou valores de um atributo que são significativamente inconsistentes ou diferentes, ou seja, valores fora do padrão de um determinado conjunto de dados. Portanto, detectar desvios objetiva identificar mudanças em padrões que já foram percebidos. A identificação de outliers, ou detecção de desvios, pode ajudar a solucionar problemas da empresa. (COSTA, 2020, p. 109)

Para tal verificação, foi utilizado o gráfico de caixa, também conhecido como *boxplot*. Sobre o *boxplot* e os *outliers*, Galarnyk (2018), citado por Costa (2020, p.111), diz que “esse tipo de gráfico *boxplot* organiza os dados em grupos por quartis, bem como traça a variabilidade abaixo ou acima dos quartis inferiores e superiores, respectivamente. Além disso, coloca os outliers como pontos fora da caixa“. Desta forma, a verificação da presença de *outliers* se faz necessária, principalmente para evitar que não haja distorções futuras no desenvolvimento da análise dos dados.

Na figura 18, temos a criação de uma função que nos retorna os *outliers* baseado nos limites superior e inferior. Na figura 19 temos a representação visual do *boxplot*.

Figura 18 - Função com retorno dos *outliers* superior e inferior.

```
[ ] # Função que retorna os outliers de uma Series baseado nos limites superior e
# inferior
def outliers(series):
    df_desc = series.describe()
    q1 = df_desc['25%']
    q3 = df_desc['75%']
    aiq = q3 - q1
    li = q1 - 1.5 * aiq
    ls = q3 + 1.5 * aiq

    filtro = (series < li) | (series > ls)
    return series[filtro]
```

➊ # Verificar a quantidade de outliers em cada DataFrame
for df in dfs:
 qtd_outliers = len(outliers(df['Volume (hm³)']))
 pct_outliers = qtd_outliers/len(df['Volume (hm³)'])
 print(f'Quantidade de outliers em {df.name}: {qtd_outliers}. Isso representa {pct_outliers:.2f}% do total.')

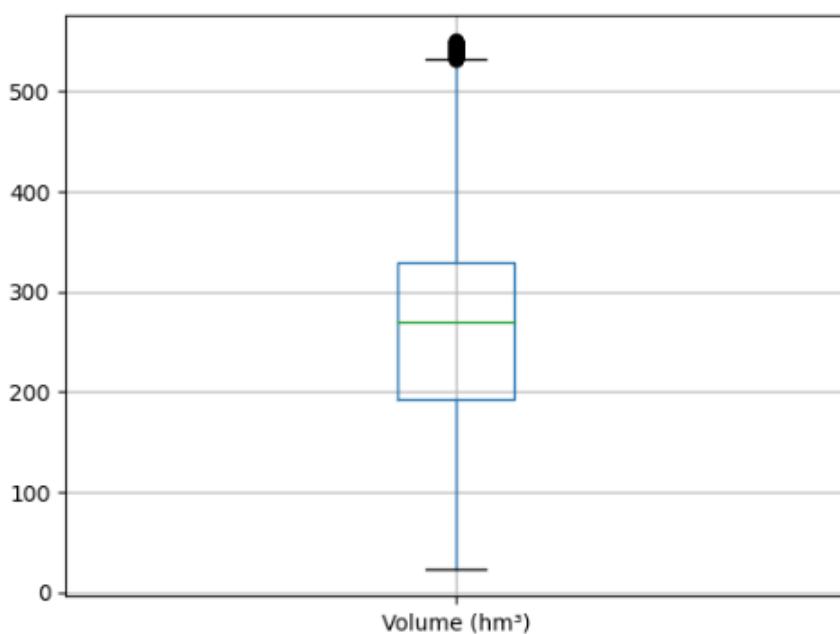
➋ Quantidade de outliers em df_alto_tiete: 77. Isso representa 0.01% do total.
Quantidade de outliers em df_alto_tiete_res: 5881. Isso representa 0.12% do total.
Quantidade de outliers em df_cantareira: 280. Isso representa 0.03% do total.
Quantidade de outliers em df_cantareira_res: 3020. Isso representa 0.06% do total.
Quantidade de outliers em df_cotia: 0. Isso representa 0.00% do total.
Quantidade de outliers em df_cotia_res: 0. Isso representa 0.00% do total.
Quantidade de outliers em df_guarapiranga: 0. Isso representa 0.00% do total.
Quantidade de outliers em df_guarapiranga_res: 1277. Isso representa 0.05% do total.
Quantidade de outliers em df_rio_claro: 0. Isso representa 0.00% do total.
Quantidade de outliers em df_rio_claro_res: 0. Isso representa 0.00% do total.
Quantidade de outliers em df_rio_grande: 47. Isso representa 0.01% do total.
Quantidade de outliers em df_rio_grande_res: 47. Isso representa 0.00% do total.
Quantidade de outliers em df_sao_lourenco: 140. Isso representa 0.02% do total.
Quantidade de outliers em df_sao_lourenco_res: 128. Isso representa 0.01% do total.

Fonte: Elaborada pelos autores (2023).

Figura 19 - Visualização do *boxplot*.

```
[ ] df_alto_tiete.boxplot(column="Volume (hm³)")
```

<Axes: >



Fonte: Elaborada pelos autores (2023).

5.3 Análise exploratória dos dados

O objetivo inicial da modelagem era a criação de modelos que predissessem o nível de água nos sistemas de abastecimento da SABESP. Ao analisar o site dos sistemas produtores da SABESP, identificamos que temos um total de sete sistemas de abastecimento: Alto Tietê, Cantareira, Cotia, Guarapiranga, Rio Claro, Rio Grande e São Lourenço. Cada sistema é composto por uma certa quantidade de represas que fluem e se unificam no sistema de abastecimento compondo o seguinte sistema:

Tabela 1 - Sistema Produtores e seus respectivos reservatórios

Sistemas Produtores	Reservatórios
Alto Tietê	Paratinga, Ponte Nova, Biritiba, Jundiaí, Taiaçupeba e Biritiba
Cantareira	Jaguari/Jacareí, Cachoeira, Atibainha, Paiva Castro, Águas Claras e Jaguari (JBS)
Cotia	Pedro Beichte, Represa da Graça e Isolina
Guarapiranga	Guarapiranga, Capivari e Billings
Rio Claro	Ribeirão do Campo
Rio Grande	Rio Grande, Ribeirão da Estiva
São Lourenço	Cachoeira da França

Fonte: elaborada pelos autores (2023).

Ao realizar a análise inicial das tabelas, percebemos que algumas não continham todas as *features* para a elaboração de um modelo de Machine Learning. Dessa forma, para uma análise minuciosa, optamos por realizar a modelagem dos dados para o Sistema São Lourenço, que é composto por apenas um reservatório. Caso a escolha fosse os demais sistemas, teríamos que criar um modelo por represa dado que cada uma possui suas particularidades específicas.

Figura 20 - Colunas da Represa Cachoeira da França.

Data	Represa Cachoeira do França					
	Nível (m)	Volume (hm ³)	Volume (%)	Q Jusante (m ³ /s)	Q Natural (m ³ /s)	Chuva (mm)

Fonte: Portal das Mananciais Sabesp.

Destacamos ainda que foi incluído um padrão para nomeação das colunas sendo mnemônicos no início da cara coluna do Dataset e CamelCase. As colunas foram transformadas para:

"VrNivel", "VlVolumeHm", "PcVolume%", "VzQJusante",
 "VzQNatural", "VrChuva", "DdDia", "MmMes", "AaAno".

Tabela 2 - Descrição das colunas

Mnemonics	Significado
Vr	Colunas onde a informação do seu dado se refere a algum tipo de valor, seja int, float etc.
Vz	Colunas onde a informação remete a vazão
Pc	Percentual % da informação
VI	Volume hm ³ (Hectare cúbico)

Fonte: Elaborada pelos autores (2023).

Na coluna data foi realizado um split para quebra em 3 nas colunas, AaAno, MmMes e DdDia.

Para a análise exploratória dos dados, a ideia inicial era a criação de um modelo paramétrico de regressão linear. Esses modelos possuem determinadas premissas que precisam ser contempladas. Portanto, logo na EDA além de observarmos *outliers*, nulos, missings e duplicados.

Nesse caso, de acordo com Miot, H. A. (2017), uma solução apropriada é o teste de Shapiro-Wilk, que permite verificar se um conjunto de dados segue uma distribuição normal, o que é fundamental em estatística. A distribuição normal é frequentemente pressuposta por muitas técnicas estatísticas, e a validade dessa suposição é essencial para a interpretação precisa dos resultados. Além disso, o teste de Shapiro-Wilk ajuda a identificar valores atípicos e anomalias nos dados, auxilia na escolha do método estatístico correto e melhora a qualidade das previsões em várias aplicações. Em resumo, é uma ferramenta crucial para garantir a confiabilidade das análises estatísticas e a tomada de decisões informadas.

Dessa maneira, submetemos o código ao teste para compreender a distribuição dos dados e logo foi identificado que nenhuma das features possuía uma distribuição normal; para essa afirmação utilizados o plot de histogramas e o teste de Shapiro-Wilk que revelou valor P inferior a 0.05 para todas as variáveis conforme demonstrado nas seguintes imagens:

Figura 21 - Código teste de Shapiro-Wilk.



```
def test_normal(df, cols, new_col_names):
    df = create_abt(df, cols, new_col_names)
    for col in df:
        if shapiro(df[col]).pvalue >= 0.05:
            print(f"coluna {col} {shapiro(df[col]).pvalue}. É Normal")
        else:
            print(f"coluna {col} {shapiro(df[col]).pvalue}. Não Normal")
```

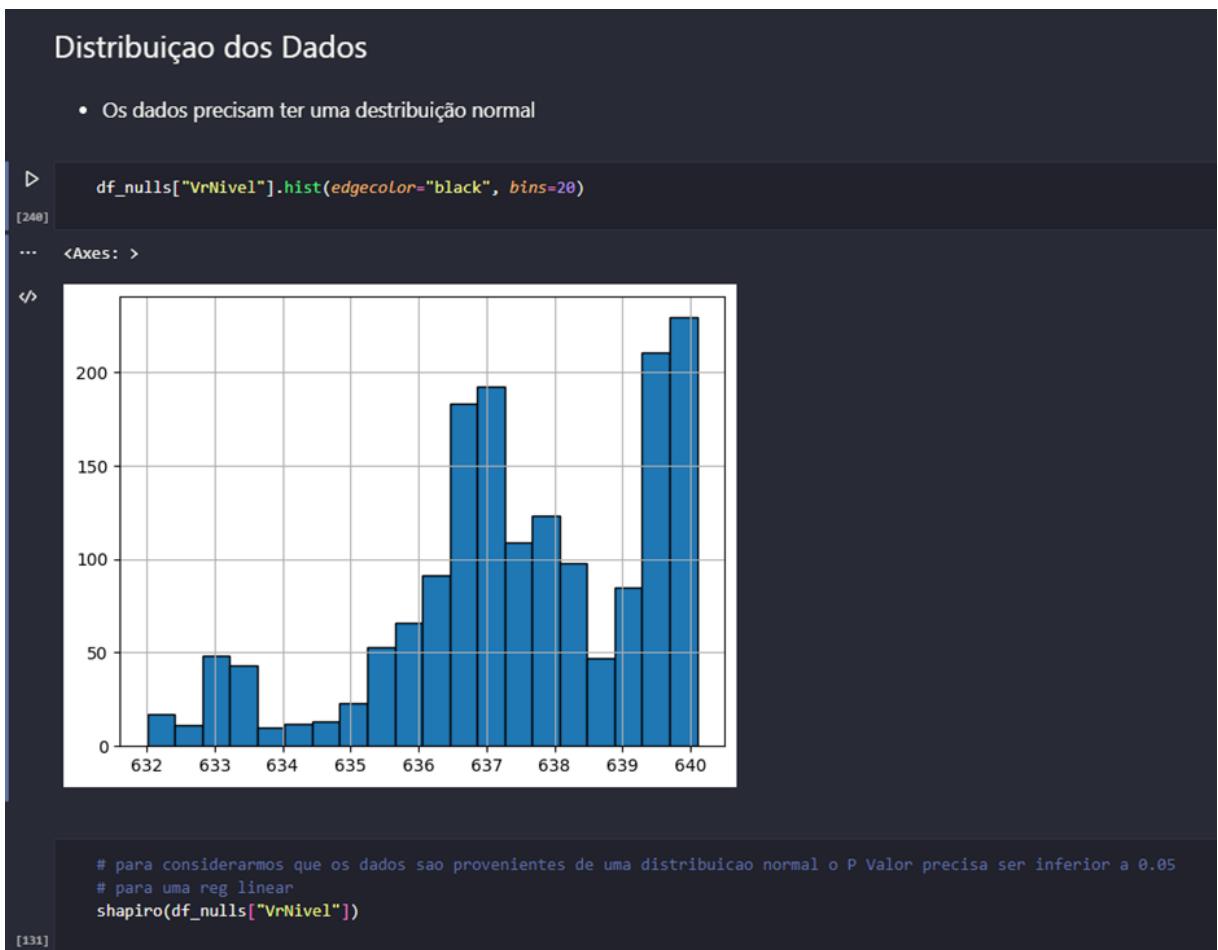
Fonte: Elaborada pelos autores (2023).

Figura 22 - Resultado do referente ao P Valor.

```
coluna VrNivel 2.5411407243235336e-27. Não Normal
coluna VlVolumeHm 1.0541436849865518e-24. Não Normal
coluna VrVolume% 1.0536989646512335e-24. Não Normal
coluna VlQJusante 1.0517226950027382e-30. Não Normal
coluna VlQNatural 1.6562250020170563e-28. Não Normal
coluna VrChuva 0.0. Não Normal
```

Fonte: Elaborada pelos autores (2023).

Figura 23 – Histograma dos valores do teste.



Fonte: Elaborada pelos autores (2023).

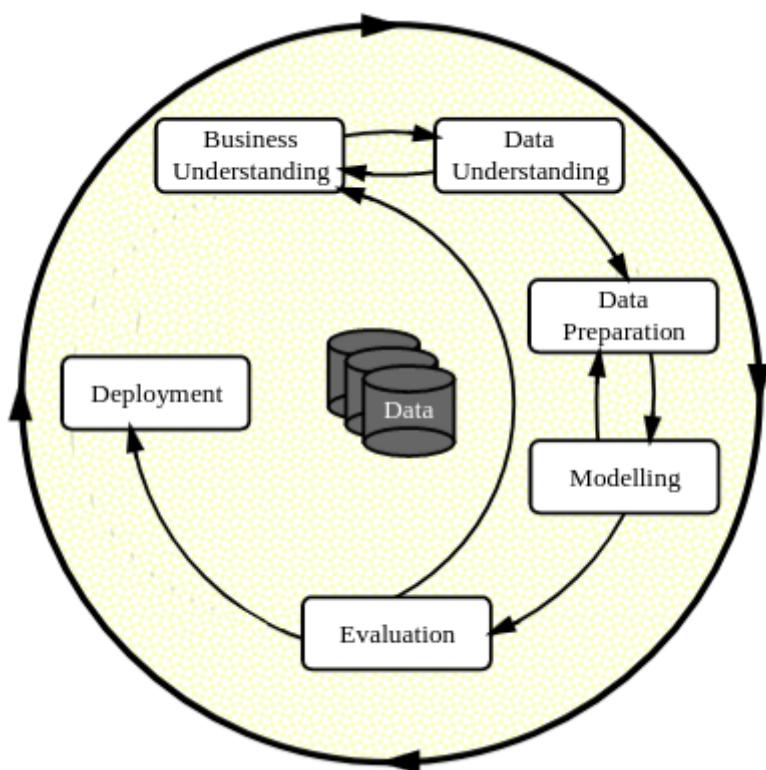
De acordo com os resultados, a base de dados apresenta um resultado de valor P inferior a 0,05 no teste de Shapiro-Wilk, o que indica que os dados não seguem uma distribuição normal. Ou seja, será necessário um tratamento adequado

dos dados como por exemplo, normalização de dados, dessa forma, ajustando e padronizando a distribuição. Posteriormente, serão implementadas ADE a fim de implementação de modelos preditivos no sistema.

5.4 Construção do dashboard

Dentre as metodologias usadas em projetos de dados, como a SEMMA e o KDD, optamos pelo CRISP-DM. O CRISP-DM é um modelo para processos de mineração de dados e descoberta de conhecimento que contempla seis etapas principais: entendimento do negócio, entendimento dos dados, preparação dos dados, modelagem, avaliação e implementação, conforme ilustrado na figura 24 (WIRTH; HIPP, 2000).

Figura 24 – Etapas do CRISP-DM



Fonte: Wirth e Hipp (2000).

No nosso trabalho, a fase de entendimento do negócio do CRISP-DM coincidiu com a etapa de inspiração do *design thinking*: buscamos entender o contexto, procurar trabalhos relacionados e elaborar requisitos. Na fase de entendimento dos dados, consultamos o dicionário de dados disponibilizado pela SABESP que descreve as colunas e pensamos na melhor forma de exibir esses dados.

A preparação dos dados envolveu a transformação dos dados em um formato mais conveniente. A modelagem, no nosso contexto, foi a elaboração do dashboard a partir do refinamento do protótipo construído anteriormente. A avaliação foi realizada através de um questionário destinado a habitantes da Região Metropolitana de São Paulo. Por fim, o *dashboard* foi implementado através do framework *Streamlit*⁶, que permite a criação de aplicações *web* voltadas à visualização de dados de forma simples através da linguagem *Python*, e será atualizado diariamente com os dados fornecidos pela API. O código-fonte está disponível em um repositório no GitHub⁷.

5.5 Construção da API

Uma API (Interface de Programação de Aplicação, do inglês *Application Programming Interface*) é uma interface entre duas aplicações, permitindo o uso de recursos e transferência de dados através de um protocolo de comunicação definido. Quando esses dados são fornecidos via *web* por uma aplicação autônoma, definimos essa aplicação como um *web service*. Há vários padrões e tecnologias de *web services*, como SOAP, CORBA, DCOM e RMI, mas a tecnologia mais usada no mundo é o modelo REST, também conhecido como RESTful (PEREIRA *et. al.*, 2018).

Segundo Pereira *et. al* (2018), o modelo REST identifica recursos através de um nome definido (normalmente uma ID), usa operadores de métodos padronizados através do protocolo HTTP e representa recursos em formatos como XML e JSON. Trata-se de uma arquitetura de *web service* simples que abstrai a arquitetura da *World Wide Web*, o que contribui para sua popularidade.

No nosso trabalho, construímos um *web service* com sua respectiva API REST através do framework *Flask*⁸ da linguagem *Python*, escolhido devido à simplicidade da ferramenta e familiaridade dos membros com a linguagem. O código-fonte está disponível em um repositório no GitHub⁹.

⁶ <https://streamlit.io/>

⁷ <https://github.com/GMerencio/sabesp-dashboard>

⁸ <https://flask.palletsprojects.com/en/2.3.x/>

⁹ <https://github.com/GMerencio/sabesp-api>

Com relação aos recursos e custos, hospedamos a API na plataforma *cloud* Render¹⁰, que disponibiliza um plano gratuito que atende as necessidades do trabalho. Caso o plano seja cancelado ou alterado, podemos procurar um serviço alternativo. Adicionalmente, como o projeto é de código aberto, desenvolvedores interessados no projeto hospedar a API em outras plataformas.

¹⁰ <https://render.com/>

6 Cronograma

Tabela 3 – Cronograma do trabalho

ETAPAS/MÊS	AGO	SET	OUT	NOV	RESPONSÁVEIS
Escolha do tema	X				Todos
Delimitação do tema	X				Todos
Definição dos objetivos	X				Todos
Levantamento bibliográfico inicial		X			Gabriel
Elaboração do protótipo inicial		X			Gabriel
Elaboração do documento para a primeira entrega		X			Gabriel, Carlos, Laiz
Primeira entrega parcial		X			Todos
Catalogação de referências		X			Carlos, Lucas
Aprimoramento do referencial teórico		X			Andreia, Bianca, Lorenzo
Extração dos dados da SABESP		X			Gabriel
Tratamento dos dados			X		Gabriel, Laiz
Construção do dashboard		X	X		Gabriel
Construção da API			X		Gabriel
Formulário para avaliação do dashboard			X		André, Bianca
Discussão dos resultados			X		Andreia, Lorenzo, Lucas
Conclusões			X		Bianca, Carlos, Lorenzo

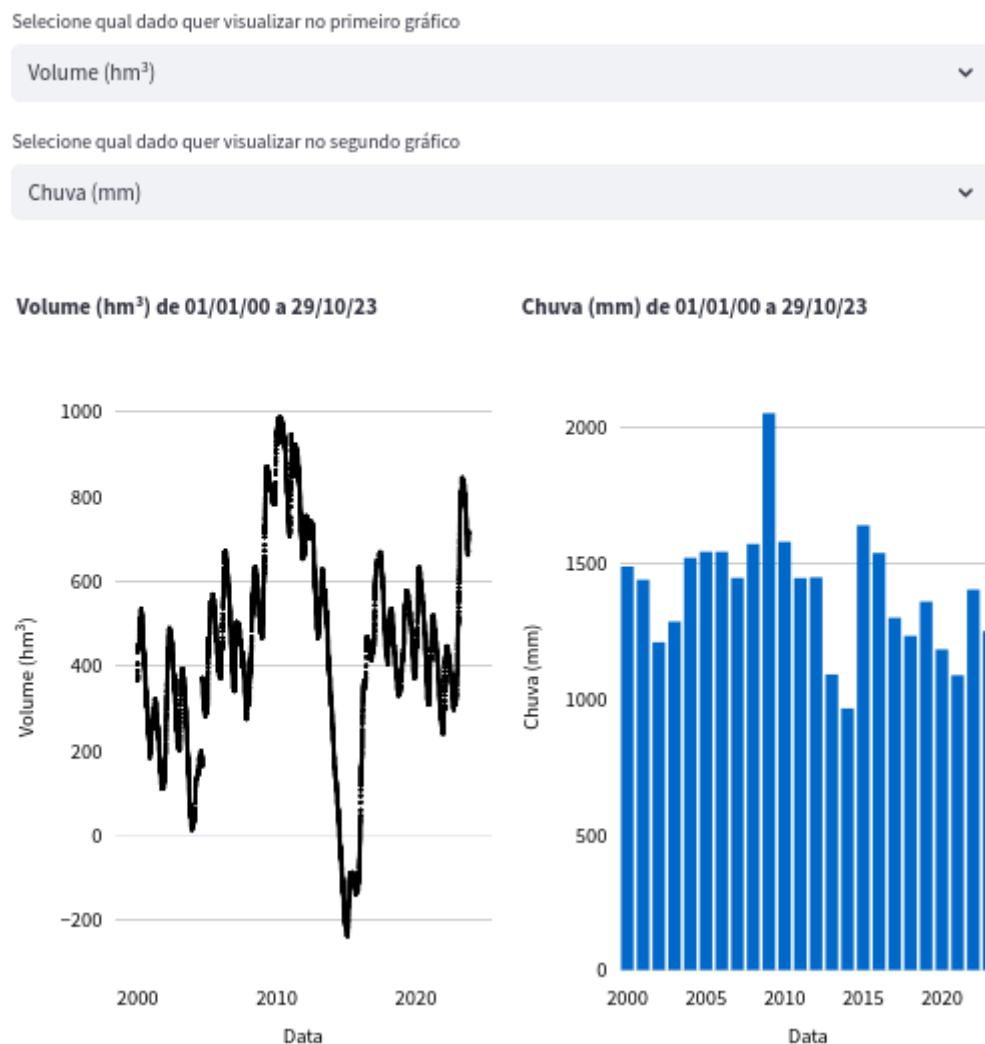
Revisão do trabalho para a segunda entrega parcial			X		André, Laiz
Segunda entrega parcial			X		Todos
Revisão do trabalho para a entrega final			X		André, Laiz
Roteiro do vídeo			X		André
Gravação do vídeo			X		Todos
Edição do vídeo			X		André
Entrega final			X		Todos

Fonte: Elaborada pelos autores (2023).

7 Resultados e discussões

A partir do protótipo do *dashboard*, implementamos as funcionalidades e refinamos o *design* de acordo com as discussões internas, publicando a versão final da aplicação¹¹. Em particular, seguindo as sugestões da orientadora, optamos por usar um gráfico de barras em vez de um gráfico de dispersão para os dados de chuva, facilitando a visualização, e incluímos a opção de visualizar dois gráficos ao mesmo tempo para permitir a comparação entre variáveis em um determinado período de tempo, como exibido na figura 25.

Figura 25 – Dashboard final após melhorias



Fonte: Elaborada pelos autores (2023).

¹¹ <https://sabesp-dashboard.streamlit.app/>

Também finalizamos a página “Sobre os dados”, contendo informações básicas sobre a fonte dos dados e o que cada coluna representa (figura 26) e a página “API”, que direciona o usuário à página de documentação e repositório do GitHub da API (figura 27).

Figura 26 – Página “Sobre os dados” na aplicação final

Sobre os dados

A fonte dos dados do projeto é a [Sabesp](#), que disponibiliza dados referentes ao sistemas produtores. A nível de sistemas, as seguintes informações estão disponíveis:

- Data: data do registro, no formato AAAA-MM-DD (por exemplo, 2000-12-31);
- Volume (hm^3): corresponde ao volume armazenado, em hectômetros cúbicos;
- Vazão natural (m^3/s): representa a vazão média diária afluente, em metros cúbicos por segundo, sem considerar as alterações antrópicas;
- Vazão a jusante (m^3/s): indica a vazão média diária descarregada, em metros cúbicos por segundo, para a jusante da barragem, ou seja, a quantidade de água liberada que segue pelo rio ou canal após o barramento;
- Chuva (mm): refere-se à precipitação acumulada nas últimas 24 horas no local do barramento, em milímetros.

Os dados são atualizados diariamente através de um script e disponibilizados via API. Você pode acessar o código-fonte e os dados, incluindo os dados dos reservatórios de cada sistema, [neste repositório do GitHub](#).

Fonte: Elaborada pelos autores (2023).

Figura 27 – Página “API” na aplicação final

API

Desenvolvedores podem acessar a [API REST](#) que disponibiliza os dados de sistemas e reservatórios, atualizados diariamente. [O repositório do projeto pode ser acessado aqui](#).

Fonte: Elaborada pelos autores (2023).

Paralelamente ao desenvolvimento do *dashboard*, finalizamos a API e a disponibilizamos através da plataforma *cloud Render*¹². A API fornece quatro endpoints, conforme exibido na figura 28:

- **/sistemas/{inicio}&{fim}&{sistema}** (GET): Fornece os dados de um determinado sistema no período especificado;
- **/sistemas/{sistema}** (GET): Fornece os dados de hoje de um determinado sistema;
- **/reservatorios/{inicio}&{fim}&{sistema}** (GET): Fornece os dados dos reservatórios de um determinado sistema no período especificado;
- **/reservatorios/{sistema}** (GET): Fornece os dados de hoje dos reservatórios de um determinado sistema.

Figura 28 – Endpoints da API na documentação

The screenshot shows the API SABESP 1.0 documentation. At the top, it says [Base URL: /] /swagger.json. Below that, a note states: API que disponibiliza os dados de reservatórios e sistemas da SABESP, atualizados diariamente. A verificação de novos dados ocorre, aproximadamente, às 09:00, 10:00, 15:00 e 20:00 no horário de Brasília.

sistemas Dados de sistemas

- GET** /sistemas/{inicio}&{fim}&{sistema} Dados do sistema no período especificado
- GET** /sistemas/{sistema} Dados de hoje do sistema especificado, caso haja

reservatorios Dados de reservatórios

- GET** /reservatorios/{inicio}&{fim}&{sistema} Dados dos reservatórios do sistema no período especificado
- GET** /reservatorios/{sistema} Dados de hoje dos reservatórios do sistema especificado, caso haja

Fonte: Elaborada pelos autores (2023).

A documentação da API descreve cada *endpoint*, com os parâmetros esperados e respostas (conforme a figura 29), e permite o teste de cada um deles (conforme a figura 30).

¹² <https://sabesp-api.onrender.com/>

Figura 29 – Documentação de um endpoint da API

GET /sistemas/{sistema} Dados de hoje do sistema especificado, caso haja

Parameters

Name	Description
sistema <small>* required string (path)</small>	Nome do sistema. Opções: cantareira, alto_tiete, rio_claro, rio_grande, guarapiranga, cotia, sao_lourenco sistema

Responses

Code	Description
200	Success
	Example Value: Model
	{ "Data": "2000-12-31", "Volume (hm)": 120, "Volume (%)": 50.5, "Chuva (mm)": 0, "Vazão natural (m³/s)": 20.8, "Vazão a jusante (m³/s)": 17.2 }
400	Solicitação inválida. Verifique a sintaxe da requisição
404	Os dados de hoje ainda não foram atualizados

Fonte: Elaborada pelos autores (2023).

Figura 30 – Resultado de um teste de um endpoint da API

Responses

Curl

```
curl -X 'GET' \
  'https://sabesp-api.onrender.com/sistemas/cantareira' \
  -H 'accept: application/json'
```

Request URL

<https://sabesp-api.onrender.com/sistemas/cantareira>

Server response

Code	Details
200	Response body <pre>[{ "Data": "2023-10-29", "Volume (hm)": 712.52388, "Volume (%)": 72.55312, "Chuva (mm)": 0, "Vazão natural (m³/s)": 44.593, "Vazão a jusante (m³/s)": 8.35 }]</pre> <p>Copy Download</p>

Response headers

```
alt-svc: h3=":443"; ma=86400
cf-cache-status: DYNAMIC
cf-ray: 81dd162a0bbe4f0-GRU
content-encoding: br
content-type: application/json
date: Sun, 29 Oct 2023 17:19:59 GMT
rndr-id: 63cbfedf-4bf4-4023
server: cloudflare
vary: Accept-Encoding
x-firefox-early-data: accepted
x-firefox-http3: h3
x-render-origin-server: unicorn
```

Fonte: Elaborada pelos autores (2023).

A documentação da API também traz informações dos dados, incluindo os tipos e descrições de cada campo, como exibe a figura 31.

Figura 31 – Descrições dos dados fornecidos pela API na documentação

Models	
Sistema ▾ {	<pre> Data string(\$date) example: 2000-12-31 Data do registro Volume (hm³) number example: 120 Volume armazenado, em hectômetros cúbicos Volume (%) number example: 50.5 Volume armazenado, em percentual do volume total Chuva (mm) number example: 10 Precipitação acumulada das últimas 24 horas, em milímetros Vazão natural (m³/s) number example: 20.8 Vazão média diária afluente, em metros cúbicos por segundo Vazão a jusante (m³/s) number example: 17.2 Vazão média diária descarregada, em metros cúbicos por segundo } </pre>
Reservatório ▾ {	<pre> Data string(\$date) example: 2000-12-31 Data do registro </pre>

Fonte: Elaborada pelos autores (2023).

Com a API e o *dashboard* finalizados, criamos uma pesquisa de avaliação da aplicação através do *Google Forms* e a distribuímos entre 20/10 e 25/10 por meio de grupos de *WhatsApp*, *Telegram*, redes sociais e outros meios de comunicação, focando no público residente na Região Metropolitana de São Paulo. Fizemos as seguintes perguntas:

1. Você reside na Região Metropolitana de São Paulo?
2. Como você avalia a facilidade de navegação na aplicação?
3. (Opcional) Por que você deu essa nota à facilidade de navegação na aplicação?
4. Como você avalia a facilidade de interagir com os gráficos (selecionando o período, o dado a ser exibido, etc.)?
5. (Opcional) Por que você deu essa nota à facilidade de interagir com os gráficos?
6. Como você avalia a velocidade da aplicação (tempo que as páginas demoram para carregarem e atualizarem)?
7. O quanto útil você considera a aplicação?
8. Por favor justifique sua avaliação da utilidade da aplicação.
9. Você encontrou algum erro ou bug ao navegar pela aplicação?

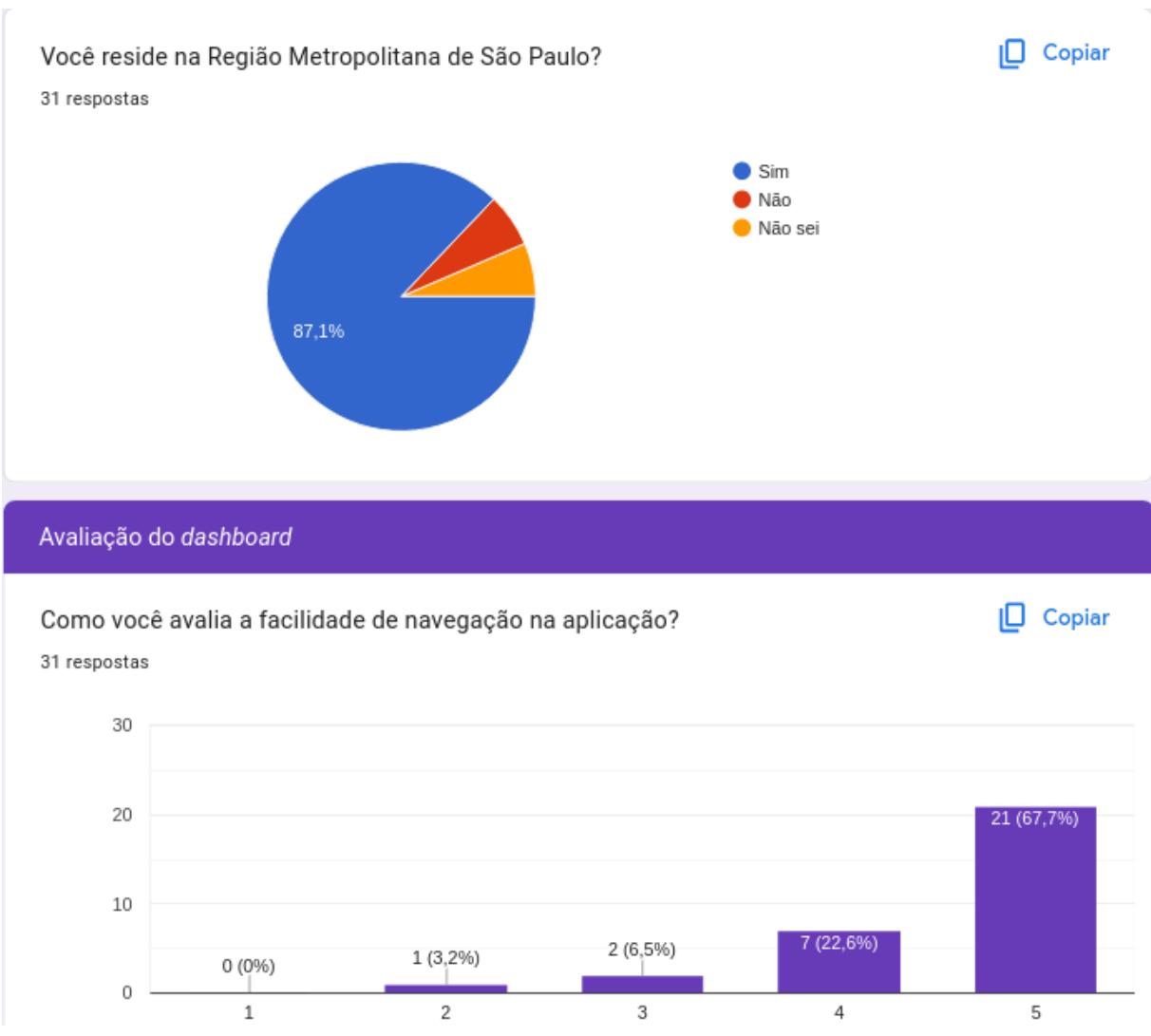
10.(Opcional) Caso tenha encontrado algum erro ou bug ao navegar pela aplicação, por favor descreva-o(s) aqui.

11. (Opcional) Caso tenha algum comentário adicional, por favor insira aqui.

Coletamos um total de 31 respostas. De modo geral, vimos um *feedback* positivo em relação à facilidade de navegação, facilidade de interação com os gráficos, velocidade da aplicação e utilidade. As figuras 32 ao 37 mostram o resumo das respostas. Analisando as respostas, notamos os seguintes pontos:

- Alguns usuários notaram dificuldades em acessar a aplicação pelo celular;
- Tentamos reproduzir os *bugs* relatados, sem êxito. Vale ressaltar que o *Streamlit* ocasionalmente exibe mensagens de erro que não afetam a aplicação, bem como comportamentos inesperados temporários;
- Alguns usuários indicaram dificuldade em navegar pelo gráfico, inclusive sugerindo instruções;
- Outras sugestões incluem uma funcionalidade que indique qual sistema abastece o local de residência do usuário, um aplicativo móvel e notificações de volume baixo nos sistemas.

Figura 32 – Resumo das respostas às perguntas 1 e 2 do questionário de avaliação



Fonte: Elaborada pelos autores (2023).

Figura 33 – Resumo das respostas às perguntas 3 e 4 do questionário de avaliação

(Opcional) Por que você deu essa nota à facilidade de navegação na aplicação?

3 respostas

Layout simples, intuitivo

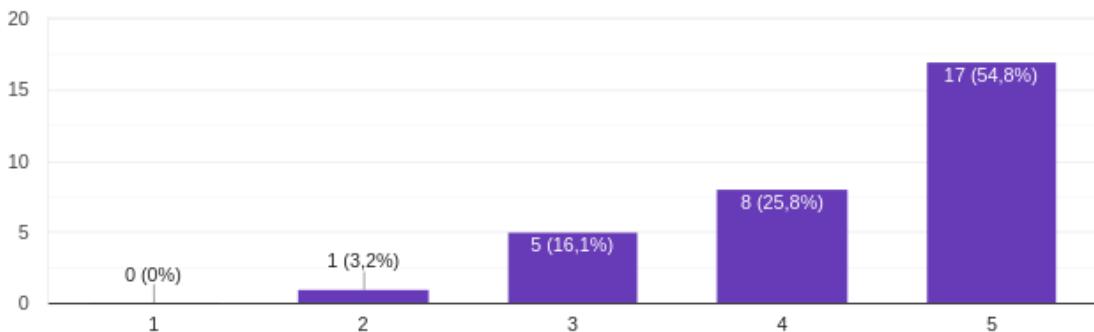
Instruções claras

Não achei fácil de usar no celular

Como você avalia a facilidade de interagir com os gráficos (selecionando o período, o dado a ser exibido, etc.)?

 Copiar

31 respostas



Fonte: Elaborada pelos autores (2023).

Figura 34– Resumo das respostas às perguntas 4 e 5 do questionário de avaliação
(Opcional) Por que você deu essa nota à facilidade de interagir com os gráficos?

5 respostas

Poderia ter instruções.

Os filtros são intuitivos e funcionam como esperado, mas não entendi como interagir com os gráficos clicando neles

Fácil e rápido pra escolher os dados

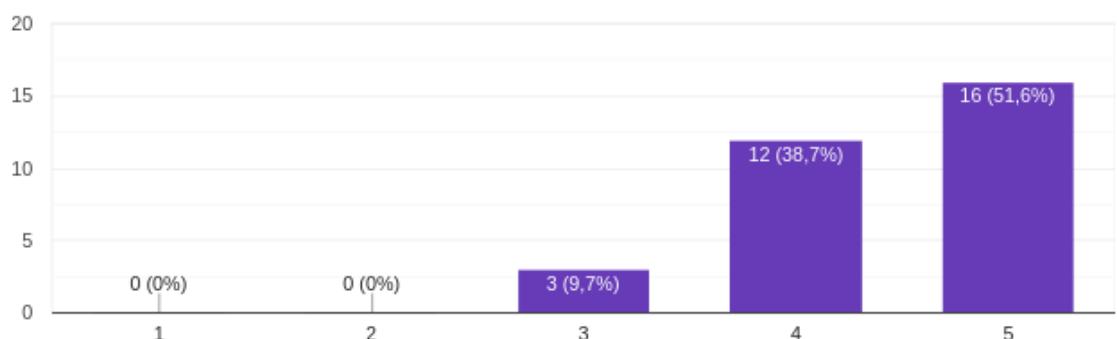
não sei mexer com gráficos

Ruim de mexer no celular

Como você avalia a velocidade da aplicação (tempo que as páginas demoram para carregarem e atualizarem)?

31 respostas

 Copiar



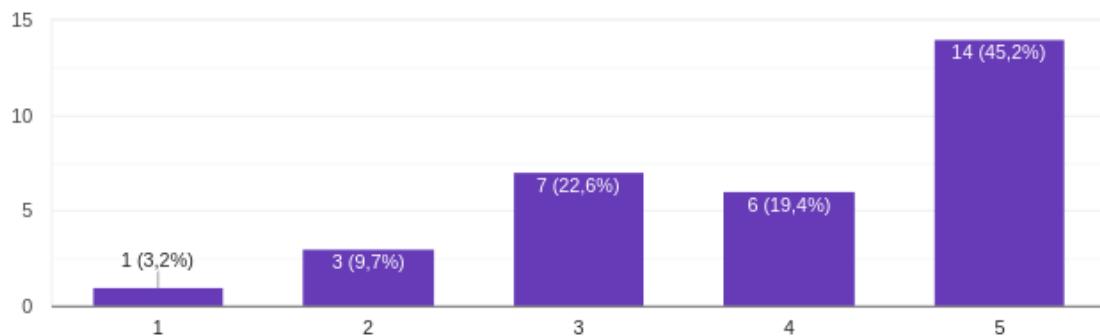
Fonte: Elaborada pelos autores (2023).

Figura 35 – Resumo das respostas às perguntas 6 e 7 do questionário de avaliação

O quanto útil você considera a aplicação?

 Copiar

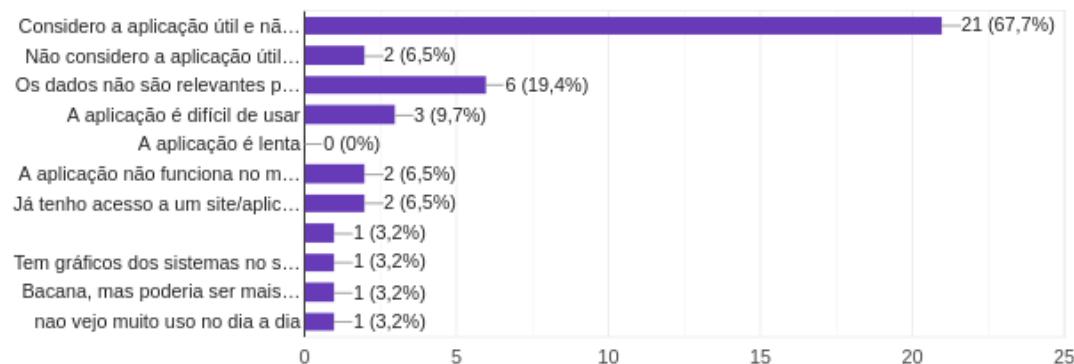
31 respostas



Por favor justifique sua avaliação da utilidade da aplicação.

 Copiar

31 respostas



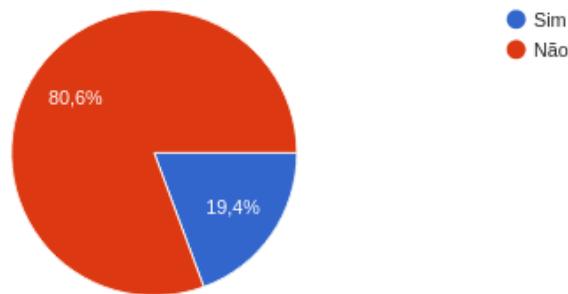
Fonte: Elaborada pelos autores (2023).

Figura 36 – Resumo das respostas às perguntas 8 e 9 do questionário de avaliação

Você encontrou algum erro ou bug ao navegar pela aplicação?

31 respostas

 Copiar



(Opcional) Caso tenha encontrado algum erro ou bug ao navegar pela aplicação, por favor descreva-o(s) aqui.

3 respostas

Mensagem "Bad message format: s is undefined"

site n carregou no celular

Às vezes o gráfico não muda quando vai de uma página pra outra

Fonte: Elaborada pelos autores (2023).

Figura 37 – Resumo das respostas à pergunta 10 do questionário de avaliação

(Opcional) Caso tenha algum comentário adicional, por favor insira aqui.

4 respostas

não sei que sistema abastece minha região... seria bom ter algo que mostrasse isso

Explicar como funciona a interatividade com os gráficos e talvez bolar um sistema de notificação de quando o nível do reservatório está baixo, etc.

Muito bom, mas não moro em São Paulo

Poderia ter um app

Fonte: Elaborada pelos autores (2023).

8 Conclusões

O desenvolvimento desse trabalho de conclusão de curso teve como base o acesso facilitado e transparente a dados e informações sobre recursos hídricos, assunto de suma importância à sociedade. Diante disso, foram selecionados os dados disponibilizados das represas administradas pela SABESP (Companhia de Saneamento Básico de São Paulo), responsável pelo abastecimento de água do estado de São Paulo.

Analizando outros projetos com propósitos em comum, foram encontradas algumas deficiências, como a falta de disponibilização pública das informações trabalhadas, a ausência de expansão dos dados abrangendo mais represas e mais períodos, versão *web* para consulta, delimitando o acesso a apenas dispositivos móveis, inexistência da possibilidade da extração de dados atualizados do site e acesso limitado apenas a uma estrutura de programação.

Com base nos aspectos levantados e identificação dessas lacunas, e levando em conta interesses da população e os dados encontrados no site da empresa SABESP, foram determinados dois objetivos: o primeiro foi construir uma base de acesso atualizada com os dados através de uma API para pesquisadores, desenvolvedores, técnicos e curiosos sobre o assunto. E o segundo foi a elaboração de um *dashboard* dinâmico e interativo para o público em geral.

Partindo da metodologia de *design thinking*, foram definidas as etapas, iniciando pela escolha das ferramentas para extração e tratamento de dados. Os dados foram coletados através de *web scraping* com a linguagem *Python*, usando a plataforma Github Actions para automatizar a coleta, e o tratamento de dados foi realizado utilizando a biblioteca *Pandas*. Para a API, a arquitetura REST foi selecionada, resultando em um *web service* desenvolvido com o *framework Flask* em *Python*. O *dashboard* foi estruturado empregando a estrutura de modelagem CRISP-DM e implementado através da biblioteca *Streamlit*.

Executamos os objetivos que foram propostos com a criação de um ambiente completo, com acesso a todo material disponibilizado pela SABESP sobre todas as suas represas e sistemas. Através da combinação da extração automatizada de dados, fornecimento dos dados através da API e visualização por meio do *dashboard*, promovemos a ampliação do acesso público à informação de maneira mais contextualizada, focada, dinâmica e clara. Verificamos a satisfação da sociedade através do questionário aplicado, constatando um retorno positivo e interesse da população.

A API, acessível através de métodos HTTP, pode ser usada em qualquer linguagem de programação. Além disso, com os dados atualizados e disponibilizados em um formato padrão, pesquisadores, desenvolvedores e outras partes interessadas nesses dados podem trabalhar apenas na modelagem e no desenvolvimento de soluções, em vez de se preocuparem excessivamente com a extração e tratamento dos dados, o que contribui para a facilitação de futuros projetos e ideias.

Por ser uma aplicação *web*, o *dashboard* é acessível por qualquer tipo de dispositivo. O *dashboard* também oferece opções interativas para comparar variáveis e períodos, trazendo melhorias na forma de visualização desses dados por parte da população em geral em comparação ao formato tabular fornecido pelo portal da SABESP. Os gráficos permitem atrelar os dados a possíveis causas ou impactos, como por exemplo a associação de baixos volumes em reservatórios a fenômenos naturais, como o baixo volume de chuva.

Perante o desenvolvido, o projeto pode vir a contribuir para a gestão de recursos hídricos com a participação da sociedade. Tratando-se de um projeto de código aberto, há a possibilidade de melhorias pela comunidade, como a expansão de dados coletados, desenvolvimento de modelos preditivos e aprimoramento do *dashboard* a partir do *feedback* colhido.

9 Referências

ALMEIDA, A. S. *et al.* Sistema de previsão hidrometeorológico para subsidiar a operação do sistema Cantareira na gestão das bacias PCJ. In: Simpósio Brasileiro de Recursos Hídricos - SBRH, 23, 2019, Foz do Iguaçu, PR. **Anais eletrônicos...** Florianópolis, SC: ABRHidro, 2019. Disponível em: <<https://files.abrhidro.org.br/Eventos/Trabalhos/107/XXIII-SBRH0640-1-20190812-223215.pdf>>. Acesso em: 29 ago. 2023.

BEZERRA, F. **Controle Social, Democracia e Administração Pública.** Controladoria Geral da União - CGU, fev. 2021, Brasília. Disponível em: <<https://www.gov.br/cgu/pt-br/assuntos/controle-social/artigos/controle-social-democracia-e-administracao-publica>>. Acesso em 13 de set. 2023.

BRASIL. **Lei Nº12.527** de 18 de nov. de 2011. Regula o acesso a informações previsto no inciso XXXIII do art. 5º , no inciso II do § 3º do art. 37 e no § 2º do art. 216 da Constituição Federal; altera a Lei nº 8.112, de 11 de dezembro de 1990; revoga a Lei nº 11.111, de 5 de maio de 2005, e dispositivos da Lei nº 8.159, de 8 de janeiro de 1991; e dá outras providências. Diário Oficial da União, Brasília, 18 nov. de 2011. Disponível em: <https://www.planalto.gov.br/ccivil_03/_ato2011-2014/2011/lei/l12527.htm>. Acesso em 16 set. 2023.

BROWN, T.; WYATT, J. **Design thinking for Social Innovation.** Stanford Social Innovation Review. California: Leland Stanford Jr. University, 2010.

CHEN, D. Y. **Análise de dados com Python e Pandas.** Tradução de Lúcia A. Kinoshita. 1ª ed. São Paulo: Novatec editora, 2018

COSTA, S. E. da. **Preparação e Análise Exploratória de Dados.** Indaial: Uniasselvi, 2020.

DEZEM, V. **Falta d'água paralisa fábricas e ameaça o crescimento da economia de São Paulo.** UOL Economia, out. 2014. Disponível em: <<https://economia.uol.com.br/noticias/bloomberg/2014/10/24/falta-dagua-paralisa-unidades-da-rhodia-e-afeta-negocios-em-sp.htm>>. Acesso em: 29 ago. 2023.

MCKINNEY, W. **Python para Análise de dados:** Tratamento de dados com pandas, numpy e ipython. Tradução de Lúcia A. Kinoshita. 1ª ed. São Paulo: Novatec, 2017.

MILZ, B. **beatrizmilz/mananciais:** Base de dados sobre volume operacional em mananciais de abastecimento público na Região Metropolitana de São Paulo (SP -

Brasil). Disponível em: <<https://github.com/beatrizmilz/mananciais>>. Acesso em: 29 ago. 2023.

MITCHELL, R. **Web Scraping com Python**. São Paulo: Novatec Editora, 2019.

PEREIRA, A. L K. *et al.* Abordagem Restful Em Serviços Web. **Revista Gestão em Foco**, n. 10, p. 460-465, 2018. Disponível em: <<https://portal.unisepe.com.br/unifia/wp-content/uploads/sites/10001/2018/12/032-ABORDAGEM-RESTFUL-EM-SERVI%C3%87OS-WEB.pdf>>. Acesso em: 29 ago. 2023.

PRESBITERIS, R. **Obtendo atualizações dos reservatórios de água da SABESP**. Portal iMasters, abr. 2015. Disponível em: <<https://imasters.com.br/apis-microservicos/obtendo-atualizacoes-dos-reservatorios-de-agua-da-sabesp>>. Acesso em: 29 ago. 2023.

REDAÇÃO RBA. **Falta de água em São Paulo já causa 3 mil demissões**. Rede Brasil Atual, jul. 2014. Disponível em: <<https://www.redebrasilitual.com.br/cidadania/falta-de-agua-em-sao-paulo-ja-causa-3-mil-demissoes-8539/>>. Acesso em: 29 ago. 2023.

SABESP. s.d. **De Onde Vem?**. Disponível em: <<https://site.sabesp.com.br/site/interna/Default.aspx?secaold=31>>. Acesso em 16 set. 2023.

SOUZA-FERNANDES, L. C. A escassez de água no Sistema Cantareira. In: SADDY, A.; BRANDÃO, C.; AVZARADEL, P. C. S. (Orgs.). **Constituição, Crise Hídrica, Energia e Mineração na América Latina**. Rio de Janeiro: Lumen Juris, 2016. p. 89-117.

WIRTH, R.; Hipp, J. CRISP-DM: towards a standard process model for data mining. In: International Conference and Exhibition on the Practical Application of Knowledge Discovery and Data Mining, 4, 2000, Manchester, UK. **Anais...** Blackpool, UK: Practical Application Company, 2000, p. 29-40.

Miot, H. A. (2017). Avaliação da normalidade dos dados em estudos clínicos e experimentais. 16(2), pp. 88-91. doi.org/10.1590/1677-5449.041117.

10 Anexos

Anexo A - Trechos do Projeto Integrador IV referente à previsão de volume do Sistema Cantareira

VIEIRA, B. S.; SIMÕES, C. H.; SANTOS, G. M.; FILHO, J. G. S.; NO, L. N.; SANTOS, N. S.; RIBEIRO, R. S.; SILVA, S. Q. **Previsão de volume do Sistema Cantareira com modelos de aprendizado de máquina.** 61f. Relatório Técnico-Científico. Cursos de Ciência de Dados e Engenharia da Computação – **Universidade Virtual do Estado de São Paulo.** Tutor: Nícolas Rosalem. Polos CEU Jambeiro, Quinta do Sol, Alto Alegre, Feitiço da Vila, Parque Bristol, Capão Redondo e Parque São Carlos, 2023.

RESUMO

O Sistema Cantareira é responsável por abastecer mais de 9 milhões de habitantes da Região Metropolitana de São Paulo. Isso correspondente a 35% da demanda dessa região, que possui baixa disponibilidade hídrica e densidade populacional alta. Esses fatores trazem desafios à disponibilidade de água, exigindo medidas de gerenciamento e prevenção de crises hídricas, como importação de água de outros sistemas, instituição de bônus por economia no consumo de água e campanhas de conscientização à população. Em 2014, ocorreu uma das maiores crises hídricas em São Paulo, devido ao período de seca que afetou o Sistema Cantareira, afetando os setores do agronegócio, saneamento e energia. As consequências do período de estiagem foram o aumento do preço de alimentos, gerando redução e restrição no consumo; e menor de geração de energia hidrelétrica, gerando a necessidade de fontes alternativas e a importação de energia de outros países. Com isso, a inflação, o Produto Interno Bruto e os investimentos do país foram afetados. Este projeto teve como objetivo desenvolver e disponibilizar um modelo preditivo do volume do Sistema Cantareira, utilizando modelos de aprendizado de máquina a partir de bases de dados confiáveis, e explorar correlações entre o volume do sistema com os níveis de precipitação no mesmo período. Utilizamos a metodologia do *design thinking*, envolvendo a comunidade externa no processo de construção e validação da solução. Após a construção da solução final, obtivemos avaliações positivas da comunidade e construímos um modelo com uma boa acurácia.

PALAVRAS-CHAVE: Sistema Cantareira; Previsão de dados; Aprendizado de máquina; Visualização de dados.

A.1 Introdução

O acesso a água potável é indispensável à vida humana (FERREIRA, 2011). A Resolução 64/A/RES/64/292 de 28/07/2010 “declarou a água limpa e segura e o saneamento um direito humano essencial para gozar plenamente a vida e todos os outros direitos humanos”.

O relatório do IPEA (s.d), no Desenvolvimento Sustentável (ODS) 6, indica como meta “Até 2030, alcançar o acesso a saneamento e higiene adequados e equitativos para todos...”. Sachs (2005) associa o fim da pobreza com a importância do acesso a água e saneamento pela população em geral.

Para Oliveira *et al* (2018), a distribuição de água para uma população deve levar em consideração dois importantes aspectos: “a situação dos mananciais e o estudo da pluviometria que incide sobre estes”.

Segundo a ANA (2021), a Região Metropolitana de São Paulo (RMSP) possui 39 municípios. É uma região com uma grande quantidade de pessoas e sua localização tem baixa disponibilidade hídrica, o que traz complexidade a disponibilidade de água e “exige a importação de água de bacias adjacentes, como ocorre no Sistema Canteira”.

O Sistema Cantareira é um sistema de abastecimento de água localizado na região metropolitana de São Paulo, responsável por atender 35% da demanda da RMSP. Seus principais mananciais são: Reservatório Jaguari, Jacareí, Atibainha, Cachoeira e Paiva Castro, conforme Figura 1 (ANA, 2021), além de túneis, canais de interligação, estação elevatória, reservatório e uma Estação de Tratamento de Água (WHATELY; CUNHA, 2006).

Figura 1 – Sistema Cantareira

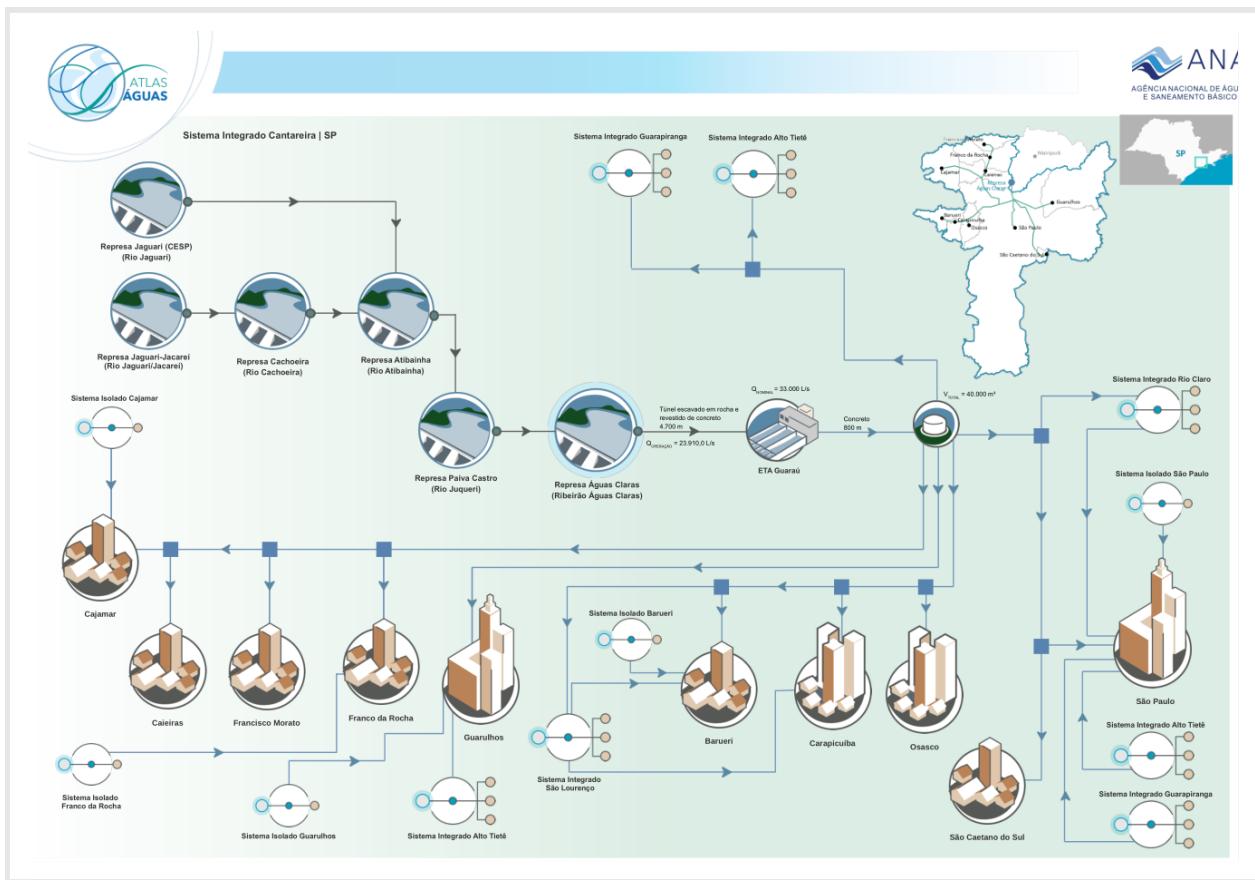


Fonte: <https://www.gov.br/ana/pt-br/sala-de-situacao/sistema-cantareira/sistema-cantareira-saiba-mais>. Acesso em: 15/04/2023

De acordo com o Atlas das Águas da ANA (2021), existem dois tipos de sistemas de abastecimento de água: os sistemas isolados, que atendem apenas um município, e os sistemas integrados, que atendem mais de um município. Os sistemas integrados são maioria nas regiões com maior densidade populacional.

O Sistema Cantareira é um exemplo de sistema integrado, conforme demonstrado na Figura 2. Possui uma área produtora de aproximadamente 227.950 hectares e produz 33 m³/s (WHATELY; CUNHA, 2006).

Figura 2 – Croqui do Sistema Integrado Cantareira



Fonte: https://portal1.snhr.gov.br/arquivos/Croquis_SNIRH/3CC_SP_INT_07_Cantareira.pdf. Acesso em: 15/04/2023

Diariamente, o site da Sabesp¹³ disponibiliza dados referentes ao Sistemas Produtores, onde é possível escolher o Sistema Canteira e ter informações sobre o Nível (m), Volume (hm³), Volume (%), Q Jusante (m³/s), Q Natural (m³/s) e Chuva (mm) separados por Represa.

De acordo com este mesmo site, temos a definição dos itens disponibilizados:

- Nível (m): refere-se ao nível da água no reservatório, em metros;
- Volume (hm³): corresponde ao volume armazenado na represa, em hectômetros cúbicos;
- Q Natural (m³/s): representa a vazão média diária afluente ao reservatório, em metros cúbicos por segundo, sem considerar as alterações antrópicas;

¹³ <https://mananciais.sabesp.com.br/HistoricoSistemas?SistemaId=0>

- Qjus (m^3/s): indica a vazão média diária descarregada, em metros cúbicos por segundo, para a jusante da barragem, ou seja, a quantidade de água liberada que segue pelo rio ou canal após o barramento;
- Chuva (mm): refere-se à precipitação acumulada nas últimas 24 horas no local do barramento da represa, em milímetros.

A base de dados disponível neste site remonta ao ano 2000; assim, é possível ter acesso a 22 anos de informação, ou seja, uma amostra representativa e ampla, possibilitando a detecção de tendências e padrões.

Dessa forma, após algumas reuniões entre o grupo, ficou decidido que neste Projeto Integrador iríamos nos concentrar em recursos hídricos, restringindo o escopo ao Sistema Cantareira.

A ideia inicial era trabalhar com funcionários da Sabesp, órgãos governamentais relacionados e especialistas em áreas relevantes; no entanto, diante das dificuldades de contato, optamos por desenvolver um formulário com perguntas que buscam identificar as principais necessidades e problemas enfrentados pelos cidadãos do estado de São Paulo, preferencialmente usuários do Sistema Cantareira.

Os resultados apontam que o gerenciamento e a prevenção de crises hídricas são questões de grande significado para a população. Vale também ressaltar que modelos de previsão e sistemas de monitoramento, de modo geral, não são acessíveis ao público geral, embora a pesquisa tenha revelado um claro interesse por tais informações. Conclui-se, portanto, que há uma demanda significativa não atendida.

Dessa forma, o objetivo principal deste projeto foi desenvolver um modelo preditivo do volume do Sistema Cantareira que permitisse antecipar possíveis crises hídricas e subsidiar as tomadas de decisão pelos órgãos responsáveis. Para isso, foram utilizados algoritmos de aprendizagem de máquina, com base na análise dos dados dos reservatórios desde o ano 2000 até 2023. O modelo resultante está acessível através de uma interface *web*, tornando as informações disponíveis ao público em geral. Assim, o tema específico cumpriu os requisitos de análise de dados em escala com um conjunto de dados existente, uso de aprendizagem de

máquina e uma interface para visualização dos resultados, além de atender às demandas identificadas junto à comunidade local.

A.2 Desenvolvimento

A.2.1 Objetivos

O objetivo principal do projeto foi o desenvolvimento e disponibilização de um modelo preditivo do volume do Sistema Cantareira. Como objetivos secundários, tentamos também desenvolver um modelo preditivo relativo à precipitação e explorar correlações entre precipitação e volume.

Nossos objetivos específicos foram:

- Identificar bases de dados relevantes e confiáveis;
- Extrair, limpar e transformar os dados para análise (etapa de pré-processamento);
- Entender e caracterizar os dados através de análises exploratórias;
- Estudar modelos de aprendizado de máquina apropriados para a tarefa de previsão do volume do Sistema Cantareira;
- Implementar os modelos e medir sua performance, identificando o modelo com os melhores resultados;
- Disponibilizar o modelo em uma interface *web* disponível para o público.

A.2.2 Justificativa e delimitação do problema

A água é um bem essencial à vida, e os problemas relacionados à sua escassez vêm cada vez mais sendo tema presente nos debates políticos e no cotidiano da população mundial. Dos 75% de água que existem no planeta, apenas 3% equivalem à água doce, sendo encontrada em geleiras, subsolo ou captadas pela chuva. Apenas 1% dessa água chega aos nossos reservatórios para ser distribuída, e com o consumo elevado por parte da agricultura e indústria, pouco sobra para a população. As crises hídricas causam diversos impactos econômicos e sociais, pois afetam diretamente setores como a agricultura e indústrias, resultando no aumento do custo e diminuição da oferta por água, alimentos e energia, além de prejudicar o saneamento básico, atingindo toda a estrutura econômica da região.

No Brasil, apesar de o país liderar o *ranking* com a maior reserva hidrológica do planeta, há muitos anos a crise hídrica vem afetando o território brasileiro, em especial regiões como o Sudeste, no qual se concentra grande densidade demográfica e industrial, e poucas reservas hídricas. Em 2014, houve uma das maiores crises com a falta de água; São Paulo foi um dos

estados mais afetados pela falta de chuva, em especial o Sistema Cantareira, que pela primeira vez teve que usar seus recursos abaixo do volume de suas comportas.

Nesse período, os principais setores afetados foram o agronegócio, saneamento e energia. Com o extenso período de estiagem, o setor do agronegócio perdeu sua capacidade produtiva e elevou seus custos, gerando assim aumento de preços, baixo consumo e restrição de alimentos. O baixo nível nos reservatórios e má conservação dos mananciais afetaram o tratamento dos esgotos, gerando assim casos de doenças. No caso do setor de energia, como a principal fonte de energia elétrica brasileira são as hidroelétricas, sem elas as alternativas são as usinas termelétricas, à base de carvão, e importação de energia de outros países, ambas resultando no aumento de preço. Isso afetou diretamente a inflação, PIB e investimentos no país. Em São Paulo, que tem um dos maiores polos industriais do Brasil, a falta de água afetou principalmente o setor industrial: fábricas desativaram unidades por não conseguirem captar água de rios que as abasteciam. Em Paulínia, por exemplo, a fábrica Solvay desativou cerca de 22 unidades (DEZEM, 2014). Foram constatadas mais de três mil demissões no setor na região (BORGES, 2014).

Segundo Souza-Fernandes (2016), o tratamento e abordagem sobre o assunto tem a seguinte definição e relevância: "A água é um problema de segurança nacional e como tal merece a adoção de estratégias direcionadas para cada um de seus aspectos particulares, todos eles de relevância para o desenvolvimento dos povos, aí compreendida a saúde pública. E todos devem se tornar partícipes nesta gestão." Diante da importância do tema, e visando a participação e entendimento maior da população, que é uma das principais afetadas pelo assunto e que hoje não tem um acesso consolidado aos dados acerca dos reservatórios, o objetivo deste projeto integrador foi tornar mais acessíveis as informações a respeito do volume do Sistema Cantareira, que abastece grande parte do estado de São Paulo, através de um modelo de análise preditiva e uma interface de fácil utilização, possibilitando assim um melhor gerenciamento, planejamento e alívio de futuras crises hídricas.

A.2.3 Fundamentação teórica

A.2.3.1 Séries temporais

Segundo Morettin e Toloi (2006), qualquer conjunto de observações ordenadas no tempo é caracterizado como uma série temporal. Exemplos incluem índices diários da Bolsa de Valores, valores mensais de temperaturas, precipitação atmosférica anual e registro de marés em um porto.

Mais formalmente, uma série temporal $Z(t_1), Z(t_2), \dots, Z(t_n)$ é um conjunto formado pelas observações nos instantes t_1, t_2, \dots, t_n de um processo estocástico, que é uma família de variáveis aleatórias definidas num mesmo espaço de probabilidade (MORETTIN; TOLOI, 2006).

Dentre os diversos modelos para previsão de séries temporais, optamos inicialmente por usar os modelos ARIMA e *Prophet*. Modelos ARIMA podem ser caracterizados da seguinte forma:

Os modelos ARIMA são modelos estatísticos lineares para análise de séries temporais. A abreviação ARIMA em inglês significa "Auto-Regressive Integrated Moving Average Model", ou seja, auto-regressivo, integrado e médias móveis. Os termos auto-regressivos correspondem a desfasagens da série e as médias móveis são as desfasagens dos erros aleatórios. Os modelos ARIMA (p, d, q) são em teoria, a classe mais geral de modelos para a previsão de uma série temporal. (XAVIER, 2016, p. 23)

Contudo, devido às limitações de tempo e dificuldades em configurar modelos complexos como ARIMA e redes neurais, optamos por não incluir o ARIMA na solução final.

O *Prophet*, em contrapartida, é um modelo de regressão aditivo com uma tendência de curva de crescimento linear ou logístico. Trata-se de um algoritmo disponível como um pacote para as linguagens de programação R e Python e é capaz de detectar automaticamente padrões sazonais de uma série, tornando-o bastante acessível (SILVA *et al.*, 2016).

A.2.3.2 Aprendizado de máquina

Aprendizado de máquina é uma subárea de inteligência artificial que faz uso de algoritmos, matemática e estatística para resolver problemas como reconhecimento facial, detecção de fraude, filtragem de *spam* em *e-mails*, projeções de dados, etc. De modo geral, as tarefas a quais o aprendizado de máquina se propõe a resolver são divididas em preditivas e descritivas, e o processo de aprendizado pode ser descrito como supervisionado, não supervisionado, por reforço, ativo e semissupervisionado (FACELI *et al.*, 2021).

A categoria relevante para este trabalho é a de problemas de regressão, classificados como tarefas preditivas:

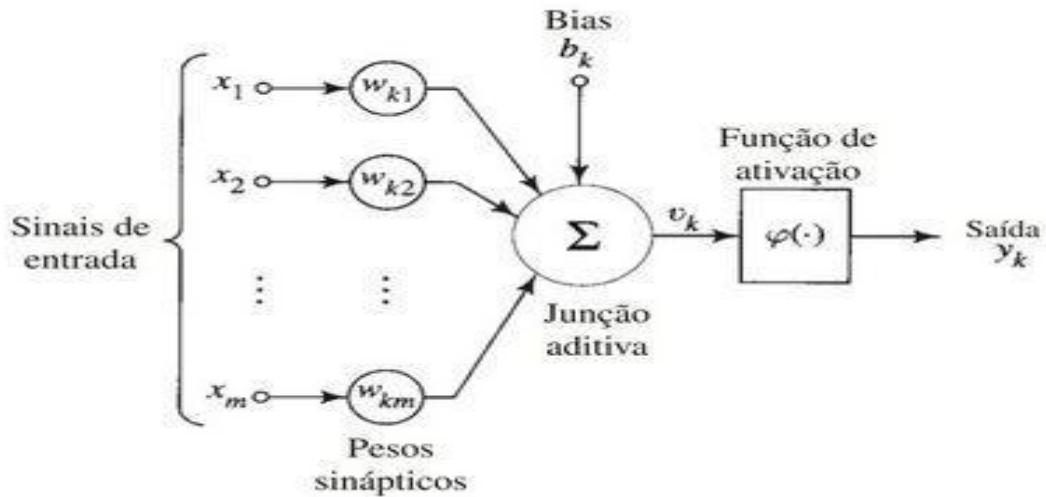
Um algoritmo de AM preditivo é uma função que, dado um conjunto de exemplos rotulados, constrói um estimador. O rótulo ou etiqueta toma valores em um domínio conhecido. Se esse domínio for um conjunto de valores nominais, tem-se um problema de classificação, também conhecido como aprendizado de conceitos, e o estimador gerado é um classificador. Se o domínio for um conjunto infinito e ordenado de valores, tem-se um problema de regressão, que induz um regressor. Um classificador (ou regressor), por sua vez, também é uma função, que, dado um exemplo não rotulado, atribui esse exemplo a uma das possíveis classes (ou a um valor real). (FACELI *et al.*, 2021, p. 49)

Assim, podemos entender a tarefa de regressão como o problema de encontrar uma função que se adequa aos dados para que possamos prever valores futuros. Os modelos ARIMA e *Prophet*, descritos anteriormente, são modelos de regressão.

A.2.3.3 Redes neurais

Redes neurais são algoritmos de aprendizado de máquina baseados no funcionamento do cérebro humano, utilizando estruturas computacionais para representar neurônios conectados entre si. A cada conexão é associado um peso sináptico, de modo que o processo de resolução de problemas (análogo ao processo de aprendizagem em humanos) envolve a modificação desses pesos para atingir os objetivos desejados. As diferentes arquiteturas de redes neurais descrevem maneiras distintas de organizar os neurônios e suas conexões, cada uma com suas vantagens e desvantagens (HAYKIN, 2001).

Figura 3 – Modelo de um neurônio

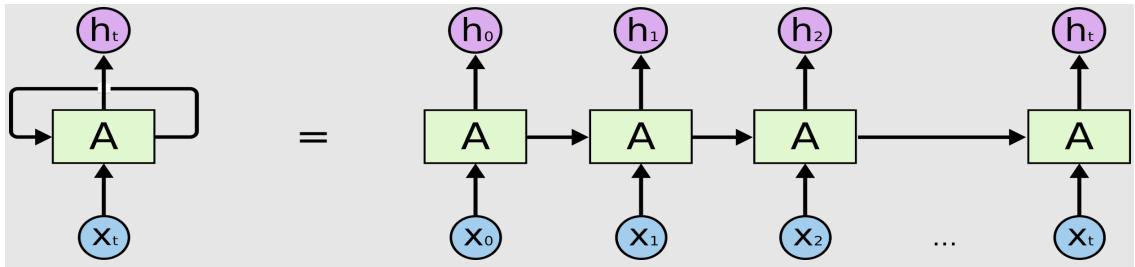


Fonte: HAYKIN, 2001, p. 36.

A Figura 3 ilustra um modelo de neurônio utilizado em algoritmos de redes neurais. Nela, vemos sinais de entrada, representados por x_1, x_2, \dots, x_m , que indicam os dados recebidos; os pesos sinápticos associados a cada conexão, representados por $w_{k1}, w_{k2}, \dots, w_{km}$; o termo de bias, b_k , utilizado como parâmetro adicional para calibrar a rede; a junção das entradas, pesos e bias na forma de uma soma ponderada; e a função de ativação, que leva como entrada a soma ponderada e retorna um número como saída (y_k).

Redes neurais tradicionais processam os dados de maneira independente, sem considerar dados anteriores ou o estado da rede. Isso impede o tratamento de dados sequenciais, como séries temporais, uma vez que previsões anteriores devem informar previsões posteriores. Redes neurais recorrentes resolvem esse problema ao incorporar laços de realimentação, de modo que, por exemplo, a saída da rede para um sinal de entrada é usada como entrada juntamente com o próximo sinal de entrada (OLAH, 2015). Na prática, isso resulta em uma rede neural com múltiplos módulos, conforme ilustrado na Figura 4, que mostra as conexões entre as entradas x_0, x_1, \dots, x_t e as saídas h_0, h_1, \dots, h_t em uma rede neural recorrente.

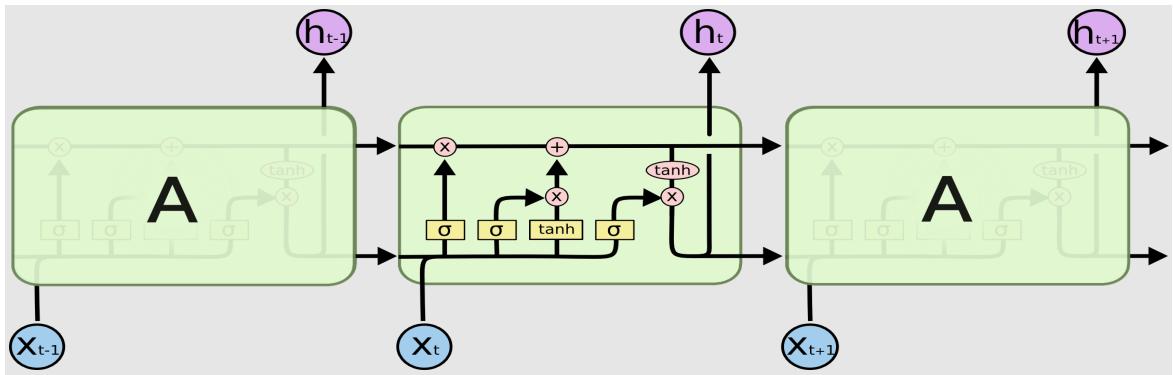
Figura 4 – Esquema geral de uma rede neural recorrente



Fonte: OLAH, 2015

Neste trabalho, além do modelo *Prophet* descrito anteriormente, também utilizamos uma arquitetura de rede neural recorrente conhecida como *Long Short-Term Memory* (LSTM). Em LSTMs, cada módulo contém quatro camadas de neurônios, chamadas de *gates*, que regulam o estado da rede, garantindo que memórias de longo prazo sejam retidas; isso é algo que redes neurais recorrentes tradicionais têm dificuldade de fazer com sucesso, pois memórias muito antigas não são retidas (PEREIRA, 2017). A Figura 5 ilustra o funcionamento de uma LSTM em linhas gerais, com três *gates* σ e um *gate* tanh (os nomes dos *gates* correspondem às suas funções de ativação), bem como as interações entre eles. O estado da rede é representado pela seta horizontal superior.

Figura 5 – Esquema geral de uma rede LSTM



Fonte: OLAH, 2015

A.2.3.4 Trabalhos semelhantes

Identificamos na literatura quatro trabalhos relevantes que utilizamos como referência, observando as abordagens distintas empregadas pelos autores e suas limitações.

O trabalho de Almeida *et al.* (2019) descreve o Sistema de Previsão Hidromeeteorológico das Bacias PCJ (SPHM-PJC), um sistema *web* de monitoramento e previsão (meteorológica e de

vazão) para gestão das bacias dos rios Piracicaba, Capivari e Jundiaí. Apesar da robustez dos modelos e do sistema, uma limitação notável é que as bacias monitoradas abastecem apenas os reservatórios Jaguari, Jacareí, Cachoeira e Atibainha, deixando de lado o reservatório Paiva Castro, abastecido pelo rio Juqueri. Adicionalmente, o sistema não é disponível ao público geral, justificando o desenvolvimento de um sistema mais amplo e acessível.

Em Oliveira *et al.* (2018), encontramos um projeto com escopo semelhante ao nosso, mas focado na previsão de precipitação do Sistema Cantareira em vez do volume, um dado mais imediatamente relevante ao público geral. Vale ressaltar que os autores utilizaram um único modelo, uma rede neural artificial baseada na arquitetura *Multilayer Perceptron* (MLP). Similarmente, Tercini, Porto e Júnior (2017) utilizaram um modelo baseado apenas no Método de Monte Carlo para prever o volume armazenado do Sistema Cantareira. Em nosso trabalho, exploramos múltiplos modelos, além de fornecer uma interface de visualização e disponibilizá-la via *web*.

Belotti (2019) traz a exploração mais abrangente do tema, aplicando 18 modelos ao problema de previsão de vazões afluentes com foco em usinas hidrelétricas. O modelo com a melhor acurácia foi um ensemble, ou seja, uma combinação de vários modelos. Esses resultados ressaltam a importância de utilizar múltiplos modelos para realizar previsões mais precisas. Embora não tenha sido possível testar um grande número de modelos devido a restrições de tempo, adotamos uma abordagem semelhante em nosso sistema.

Após a análise dos trabalhos citados, verificamos que nossa proposta, a de desenvolver e disponibilizar ao público um sistema de previsão de volume do Sistema Cantareira, explorando vários modelos de aprendizado de máquina, é distinta e possui relevância acadêmica, com o potencial de contribuir a projetos futuros.

A.2.5 Metodologia

O *design thinking* foi a metodologia usada como base para o nosso projeto. Embora seja frequentemente usada em contextos empresariais como parte de processos de inovação, o *design thinking* é, na verdade, uma abordagem ampla para a resolução de problemas com ênfase em experimentação, flexibilidade e conhecimento tácito. Em particular, empregamos o *design thinking* através do modelo HCD (*Human-Centered Design*), que possui três etapas: ouvir, criar e implementar (AMORIM, 2013).

Como parte da primeira etapa (“ouvir”), elaboramos um formulário *online* como forma de pesquisa de interesse que foi distribuído para habitantes do estado de São Paulo via grupos de *Telegram*, *WhatsApp* e outros canais de comunicação. Incluímos as seguintes perguntas:

1. Sua residência ou local de trabalho são abastecidos pelo sistema Cantareira?
2. Você já sofreu problemas de desabastecimento de água?
3. Você tem interesse em um sistema que prevê os volumes dos reservatórios do sistema Cantareira?
4. Além do volume, gostaria de ter acesso a outros dados?
5. De que forma(s) gostaria de acessar o sistema?
6. Caso conheça sistemas semelhantes, por favor mencione-os e dê uma breve avaliação de cada. Que problemas e possíveis melhorias você identifica?
7. Caso tenha outras sugestões e recomendações para o sistema, por favor escreva aqui.
8. Por que você não tem interesse em um sistema que prevê os volumes dos reservatórios do sistema Cantareira? (Caso o respondente tenha declarado não ter interesse na pergunta 3.)
9. Qual ou quais sistemas semelhantes você conhece? (Caso o respondente tenha declarado conhecer sistemas semelhantes na pergunta 8.)
10. De modo geral, como você avalia o(s) sistema(s)? (Caso o respondente tenha declarado conhecer sistemas semelhantes na pergunta 8.)

O sumário das respostas encontra-se no Apêndice A. Destacamos que 67% dos participantes que informaram residir ou trabalhar em um local abastecido pelo Sistema Cantareira indicaram que sofreram problemas de desabastecimento de água. Adicionalmente, notamos um alto nível de interesse em dados de reservatórios, com 91% dos participantes respondendo afirmativamente à pergunta 4.

Tendo confirmado o interesse da comunidade e colhido sugestões, nossos próximos passos envolveram a coleta e tratamento dos dados para a posterior elaboração e disponibilização de um modelo preditivo (como parte da etapa “criar”), conforme descrito na solução inicial.

A etapa final (“implementar”) envolveu o aprimoramento da solução a partir do *feedback* coletado através de um segundo formulário, também distribuído para habitantes do estado de São Paulo via grupos de *Telegram*, *WhatsApp* e outros canais de comunicação entre 15/03/2023 e 22/03/2023. Incluímos as seguintes perguntas:

1. Sua residência ou local de trabalho são abastecidos pelo Sistema Cantareira?
2. Por favor avalie a clareza das instruções no site. (Nota de 1 a 5.)
3. Por que você deu essa nota à clareza das instruções no site? (Resposta opcional.)
4. Na sua opinião, o quanto fácil é navegar pelos gráficos? (Nota de 1 a 5.)
5. Por que você deu essa nota à facilidade de navegação? (Resposta opcional.)
6. Como você avalia o tempo que o site demora para carregar? (Nota de 1 a 5.)
7. Como você avalia a performance do site durante o uso? (Nota de 1 a 5.)
8. Como você, pessoalmente, avalia a utilidade do site? (Nota de 1 a 5.)
9. Por favor justifique sua avaliação da utilidade do site.
10. Você encontrou algum erro ou bug ao navegar pelo site?
11. Caso tenha encontrado algum erro ou bug ao navegar pelo site, por favor descreva-o(s) aqui. (Resposta opcional.)
12. Caso tenha algum comentário adicional, por favor insira aqui. (Resposta opcional.)

O sumário das respostas encontra-se no Apêndice B. Embora o número de respondentes (15 no total) tenha sido menor, dado que a pesquisa de avaliação exigiu mais tempo e esforço em relação à pesquisa de interesse, notamos um alto nível de satisfação: todas as categorias avaliadas tiveram uma média acima de 4. Ao final da pesquisa, organizamos as sugestões e procedemos ao desenvolvimento da solução final.

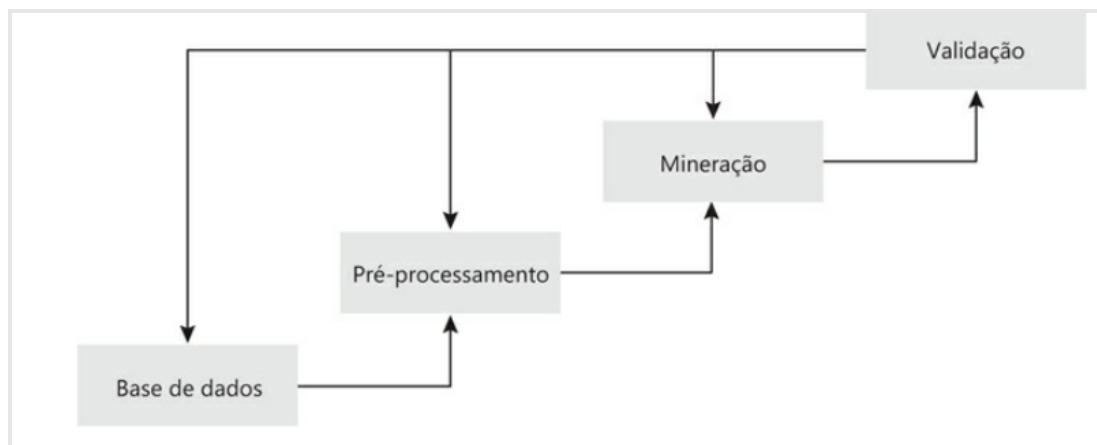
A.3 Resultados

A.3.1 Solução inicial

A.3.1.1 KDD (Knowledge Discovery in Databases)

Em 1995, na primeira conferência internacional sobre KDD, realizada na cidade de Montreal, Canadá, foi proposto que a terminologia *descoberta de conhecimentos em bases de dados* se referisse a todo o processo de extração de conhecimentos a partir de dados. Foi proposto também que a terminologia mineração de dados fosse empregada exclusivamente para a etapa de descoberta do processo de KDD, que inclui a seleção e integração das bases de dados, a limpeza da base, a seleção e transformação dos dados, a mineração e a avaliação dos dados (DE CASTRO; FERRARI, 2016).

Figura 6 - Processo de descoberta de conhecimento em bases de dados



Fonte: DE CASTRO; FERRARI, 2016

A.3.1.1.1 Extração (base de dados)

Com base nas etapas do KDD, inicialmente cogitamos extrair os dados brutos do site Portal dos Mananciais SABESP - Dados dos Sistemas Produtores¹⁴ através da técnica de *web scraping*, que consiste em uma etapa específica da mineração de dados para extração de dados de sites da web. Em seguida, esses dados são convertidos em informação estruturada para posterior análise. Após uma análise prévia, identificamos que o site foi estruturado de tal

¹⁴ <https://mananciais.sabesp.com.br/HistoricoSistemas?SistemaId=0>

maneira que demandaria um tempo excessivo na etapa de automatização do processo de extração dos dados. Devido ao prazo do projeto, optamos pela extração manual dos dados.

Após o início das extrações, nos deparamos com outro inconveniente: devido a restrições do site, os registros só podem ser extraídos em intervalos de no máximo 6 meses; neste caso, a extração dos dados de 2000 até o primeiro trimestre de 2023 corresponde a um total de 47 arquivos no formato .xls.

Após a extração e formatação das colunas, realizamos o agrupamento dos dados em um único *dataset*.

A.3.1.1.2 Pré-processamento dos dados

O pré-processamento, também conhecido como preparação da base de dados, manipula e transforma os dados brutos de maneira que o conhecimento neles contido possa ser mais fácil e corretamente obtido.

Após a obtenção do *dataset*, a base foi carregada no *Jupyter Notebook* para análise de cada atributo. O *dataset* está disponível no repositório do *GitHub*¹⁵ que criamos para centralizar o processo de análise e desenvolvimento. Em seguida, foi gerado o *dataframe* df_SABESP para análise de cada atributo.

A.3.1.1.3 Mineração

A análise descritiva de dados (ADD) foi utilizada para descrever, simplificar ou sumarizar as principais características de uma base de dados, fundamentais para qualquer análise quantitativa de dados.

A base de dados df_SABESP consiste em 33.964 linhas e 8 atributos, conforme mostra a Figura 7.

¹⁵ <https://github.com/GMerencio/previsao-sistema-cantareira>

Figura 7 - Base de dados df_SABESP

```
In [5]: df_SABESP.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 33964 entries, 0 to 33963
Data columns (total 8 columns):
 #   Column           Non-Null Count  Dtype  
---  --  
 0   Data              33964 non-null   datetime64[ns]
 1   Represa           33964 non-null   object  
 2   Nível (m)         33964 non-null   float64 
 3   Volume (hm³)     33964 non-null   float64 
 4   Volume (%)        33964 non-null   float64 
 5   Q Jusante (m³/s) 33964 non-null   float64 
 6   Q Natural (m³/s) 33964 non-null   float64 
 7   Chuva (mm)        33964 non-null   float64 
dtypes: datetime64[ns](1), float64(6), object(1)
memory usage: 2.1+ MB
```

Fonte: Elaborada pelos autores (2023)

Também não foram identificados valores ausentes, conforme a Figura 8.

Figura 8 - Inexistência de valores ausentes na base de dados df_SABESP

```
In [110]: # índices das linhas que contém valores NaN
ValNaN = pd.isnull(df_SABESP).any(axis=1).to_numpy().nonzero()

# imprime apenas as linhas com valores ausentes
display(df_SABESP.iloc[ValNaN])
```

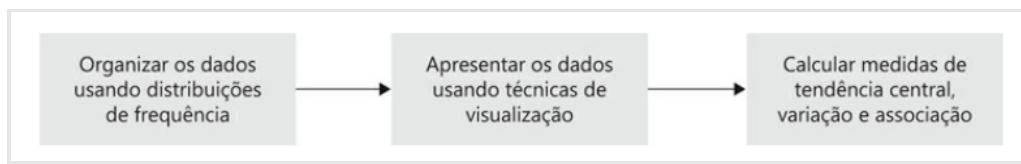
Data	Represa	Nível (m)	Volume (hm³)	Volume (%)	Q Jusante (m³/s)	Q Natural (m³/s)	Chuva (mm)

Fonte: Elaborada pelos autores (2023)

A análise descritiva de dados é um processo que pode ser desmembrado em três passos principais (conforme a Figura 9):

1. Organizar os dados usando distribuições de frequência;
2. apresentar os dados usando técnicas de visualização; e
3. calcular medidas de tendência central, variação e associação.

Figura 9 - Processo de análise descritiva dos dados



Fonte: DE CASTRO; FERRARI, 2016

A Figura 10 mostra as principais estatísticas da base de dados.

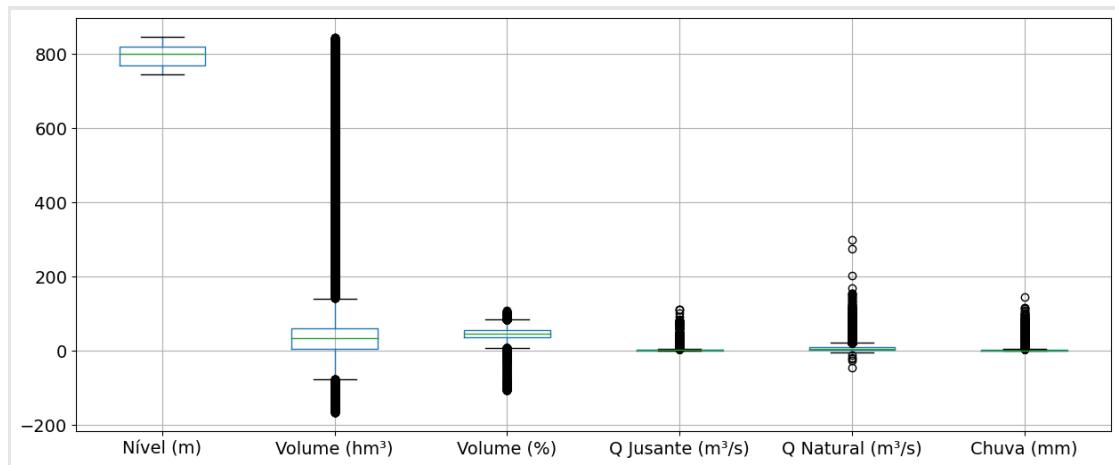
Figura 10 - Principais estatísticas da base de dados

	Nível (m)	Volume (hm³)	Volume (%)	Q Jusante (m³/s)	Q Natural (m³/s)	Chuva (mm)
count	33964.000000	33964.000000	33964.000000	33964.000000	33964.000000	33964.000000
mean	794.796559	107.918554	43.989486	1.475058	8.143131	3.890495
std	34.216316	185.010730	21.144810	4.172573	11.630960	9.628289
min	743.970000	-167.310740	-106.811763	0.000000	-46.991017	0.000000
25%	767.747500	4.569826	35.016609	0.250000	2.110000	0.000000
50%	798.530000	33.534372	45.740448	1.000000	4.220000	0.000000
75%	818.570000	58.876226	54.219413	1.500000	9.323181	2.000000
max	844.700000	843.085059	109.176203	110.000000	298.206401	143.200000

Fonte: Elaborada pelos autores (2023)

O *box plot* ilustrado no Gráfico 1 indica que, com exceção do atributo **Nível (m)**, todos os demais atributos possuem anomalias ou valores discrepantes, também conhecidos como outliers (círculos pretos), o que pode prejudicar o desempenho de vários métodos de aprendizado de máquina, pois trata-se de amostras com valores de atributos incorretos.

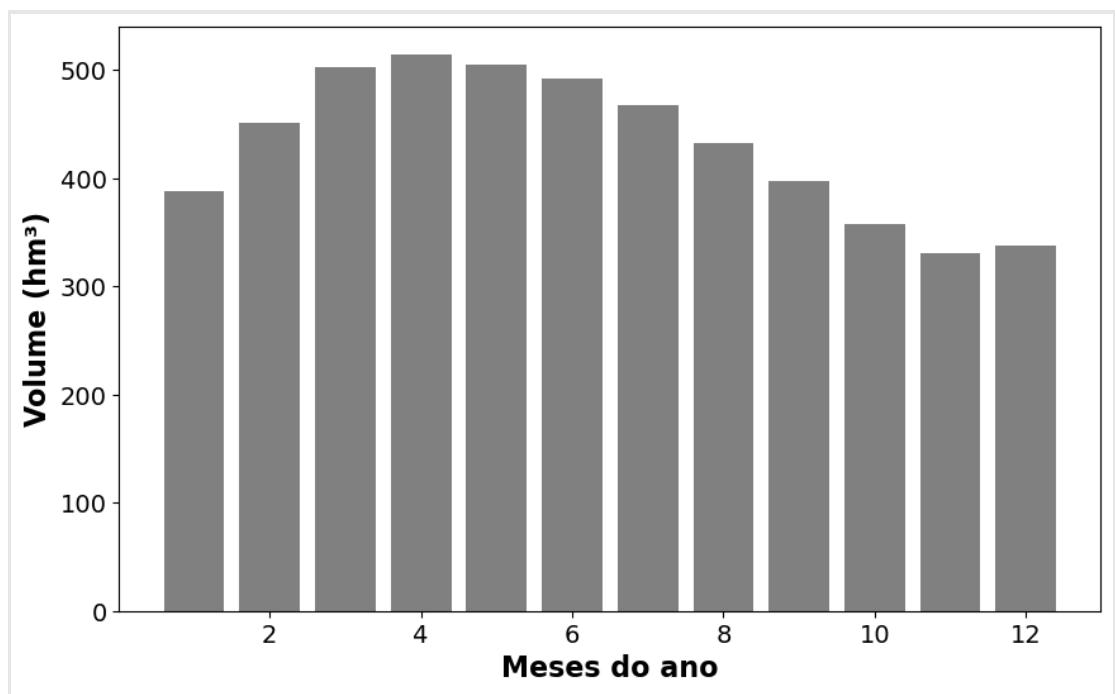
Gráfico 1 - Box plot de cada atributo do dataframe df_SABESP



Fonte: Elaborada pelos autores (2023)

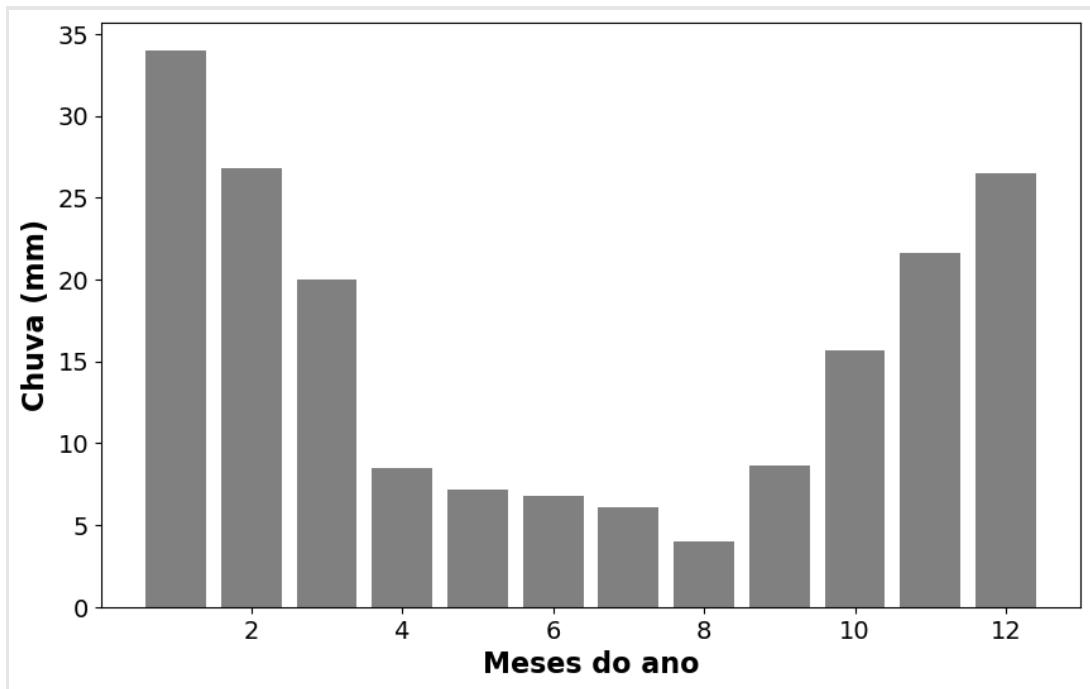
Vamos observar como se comportou o volume do sistema e a média pluviométrica (chuvas) ao longo da série temporal. Ambos os atributos são alvos de nossa análise.

Gráfico 2 - Variação média do volume do Sistema Cantareira por meses do ano (2000 até março de 2023)



Fonte: Elaborada pelos autores (2023)

Gráfico 3 - Variação média da chuva do Sistema Cantareira por meses do ano (2000 até março de 2023)



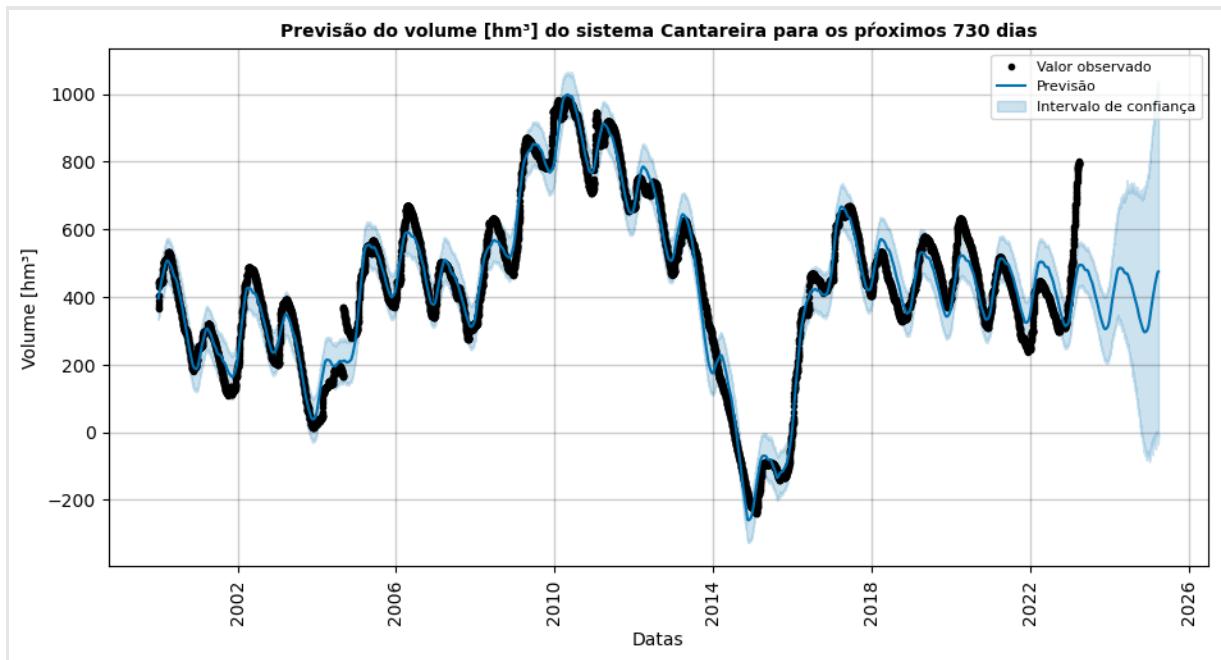
Fonte: Elaborada pelos autores (2023)

Nos gráficos 2 e 3, é possível observar que os meses de máximo volume do reservatório correspondem aos menores índices pluviométricos. A hipótese é de que essa variação pode estar relacionada ao período de racionamento do consumo de água ou até mesmo uma diminuição do consumo devido ao período de inverno no hemisfério sul. Tais hipóteses podem ser validadas posteriormente.

A.3.1.2 Modelo preditivo preliminar

Nessa primeira etapa do modelo, utilizamos a ferramenta *Prophet* para elaboração do modelo inicial e, nessa etapa, os dados não foram separados entre dados de treino e teste, que teria como objetivo validar a eficácia do modelo. O aprofundamento da previsão, bem como a utilização do modelo de rede neural e o estudo de correlação entre o volume e as chuvas no Sistema Cantareira. O Gráfico 4 mostra os resultados do modelo.

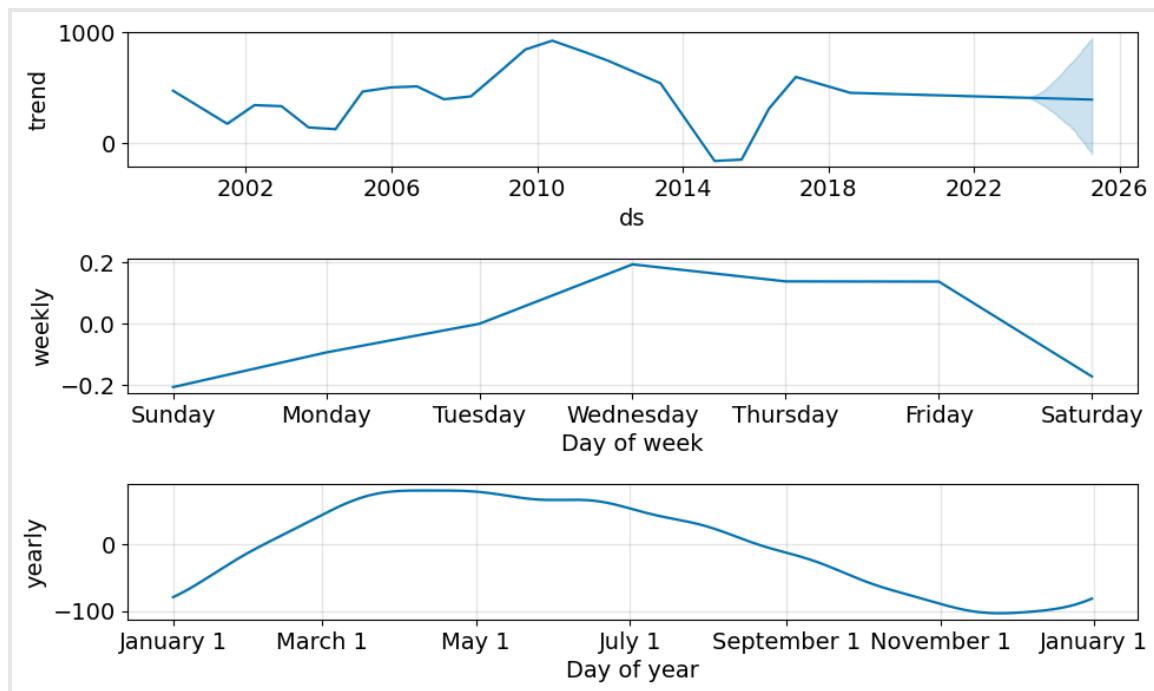
Gráfico 4 - Modelo de previsão do volume do Sistema Cantareira



Fonte: Elaborada pelos autores (2023)

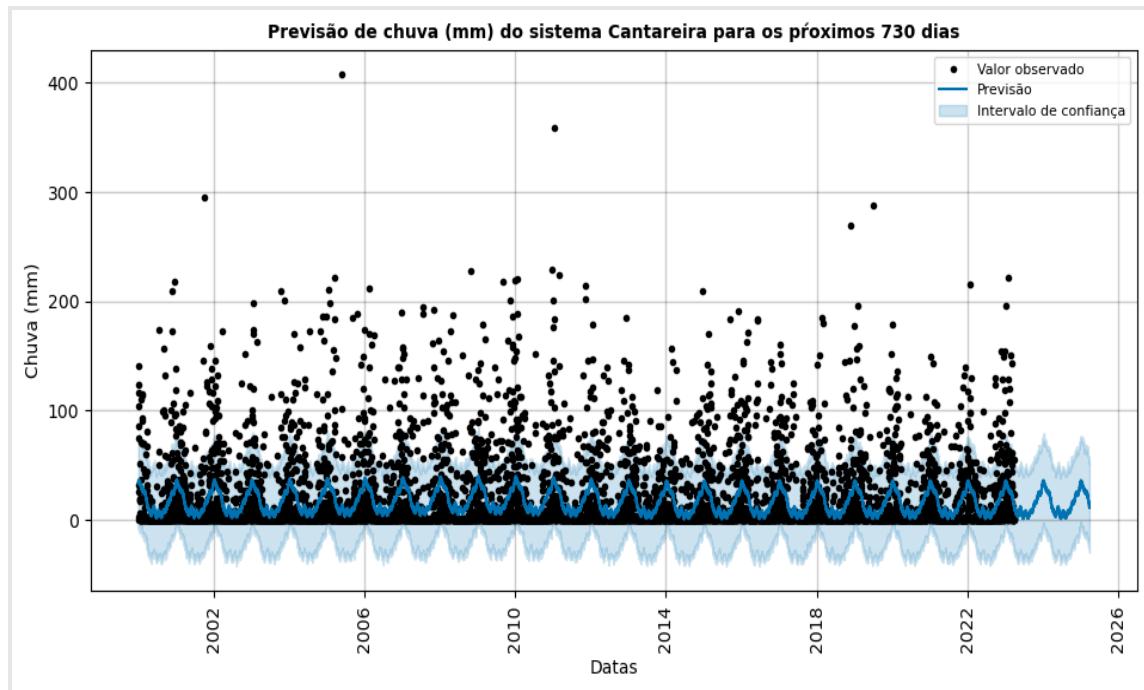
Notamos que durante o período de 16/05/2014 a 15/05/2017, durante a crise hídrica, o Sistema Cantareira operou com o volume de sua reserva técnica. Como o *Prophet* é um modelo aditivo, ele gera uma curva complexa que pode ser decomposta em suas constituintes, ou componentes, conforme o Gráfico 5. Os gráficos 6 e 7 mostram análises semelhantes para o atributo chuva.

Gráfico 5 - Decomposição do modelo de volume do *Prophet* em seus componentes



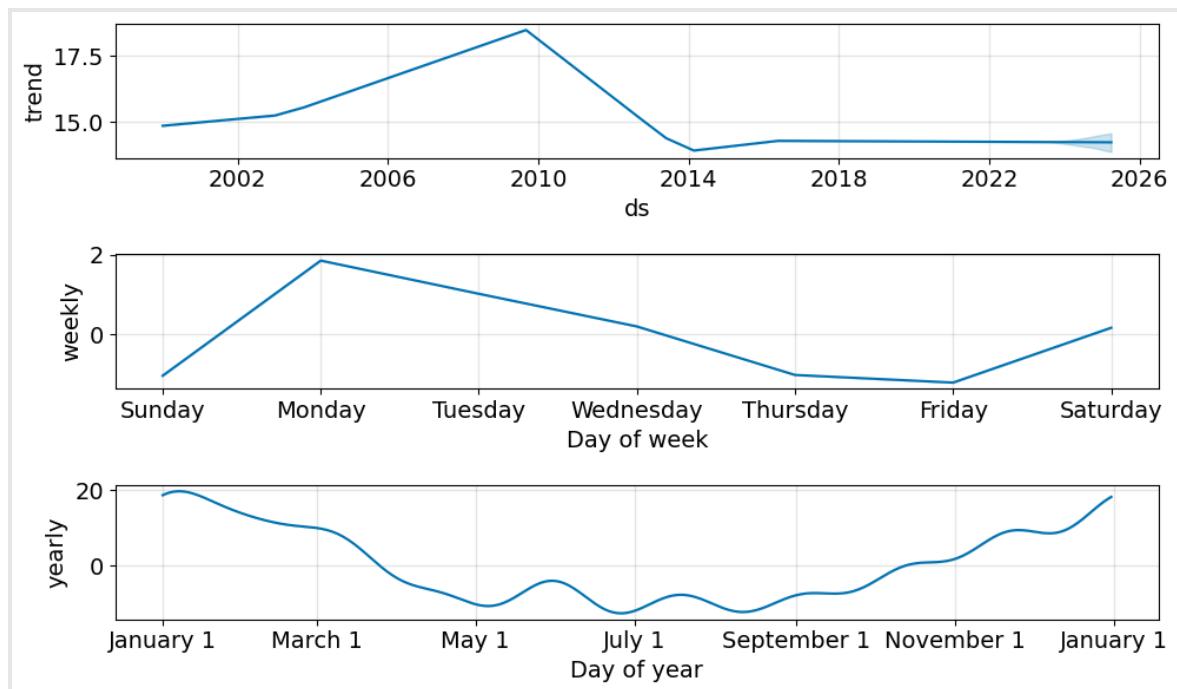
Fonte: Elaborada pelos autores (2023)

Gráfico 6 - Modelo de previsão da chuva do sistema Cantareira



Fonte: Elaborada pelos autores (2023)

Gráfico 7 - Decomposição do modelo de chuva do *Prophet* em seus componentes



Fonte: Elaborada pelos autores (2023)

Os pontos dos gráficos 4 e 6 representam os valores reais observados, a linha sólida azul representa os valores calculados no modelo e o sombreado claro ao redor da linha sólida na região prevista representa o intervalo e incerteza inferior e superior.

No modelo preliminar não foram levados em consideração os outliers identificados na etapa de mineração (ilustrados no Gráfico 1) e também não foi feita a validação para verificar se os dados para treinar o modelo são aderentes aos dados de teste, não selecionados nesta etapa.

A.3.1.3 Interface de visualização

Após a confecção do modelo, utilizamos o *framework Streamlit*¹⁶ e a biblioteca *Plotly*¹⁷ para desenvolver e disponibilizar uma interface de visualização através da linguagem *Python*. Escolhemos o *Streamlit* devido à facilidade de prototipação, permitindo a rápida criação de aplicações web voltadas a dados sem ter que criar e hospedar um website inteiro do zero. O *Plotly* disponibiliza gráficos interativos e tem boa integração com o *Streamlit*.

¹⁶ <https://streamlit.io/>

¹⁷ <https://plotly.com/python/>

A aplicação¹⁸, disponível no *Streamlit Cloud*, serviço de hospedagem do *Streamlit*, consiste em três páginas. Na página inicial, temos uma breve introdução ao projeto e uma seção com instruções em texto e em vídeo para navegar pelo gráfico, conforme a Figura 11.

Figura 11 - Página inicial da aplicação

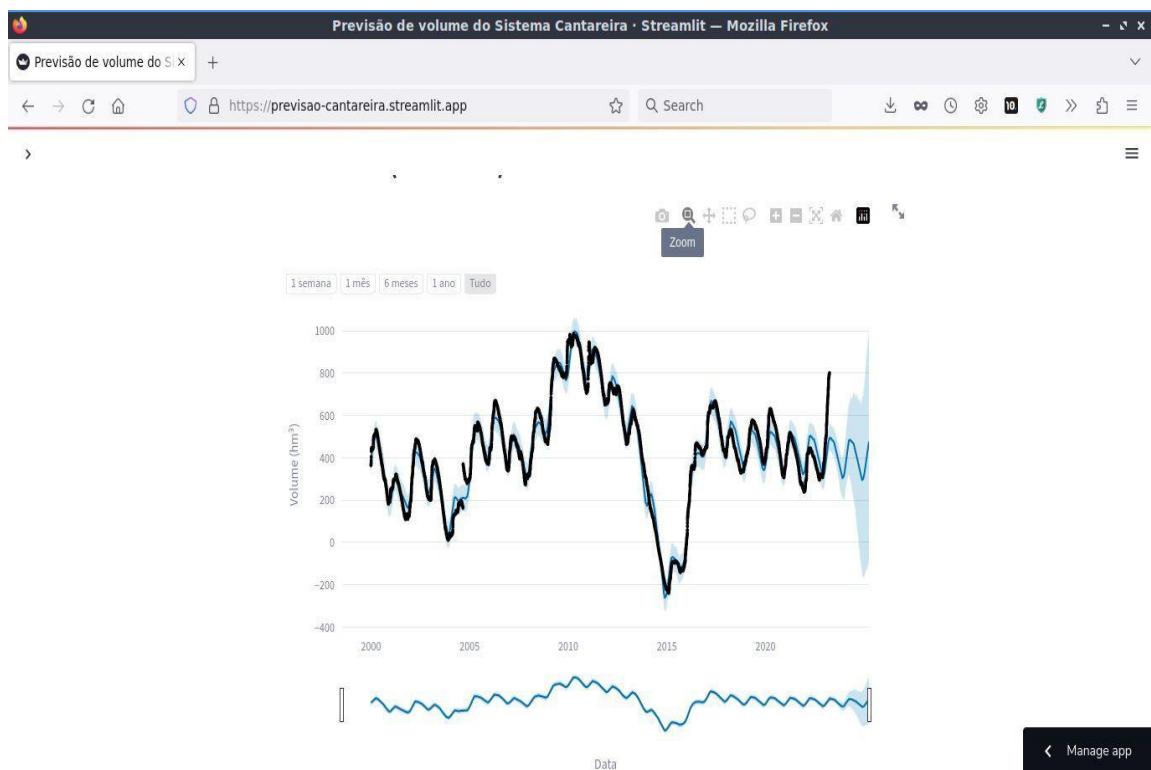


Fonte: Elaborada pelos autores (2023)

As Figuras 12 e 13 mostram os gráficos interativos de volume e chuva, com os dados históricos e as previsões do modelo. Os dados em preto representam os valores observados e os dados em azul representam os valores previstos pelo nosso modelo. A área azul acima e abaixo dos pontos representa o intervalo de confiança da previsão. Essas informações estão descritas nas legendas dos gráficos.

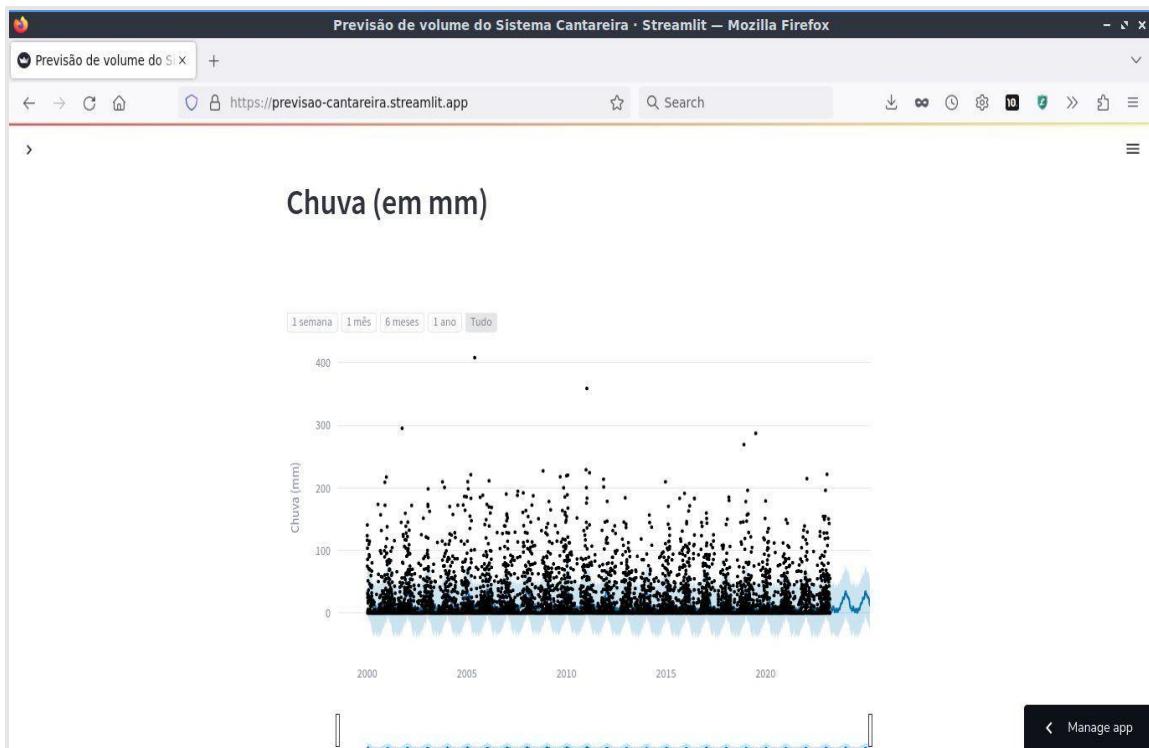
¹⁸ <https://previsao-cantareira.streamlit.app/>

Figura 12 - Gráfico interativo do volume na aplicação



Fonte: Elaborada pelos autores (2023)

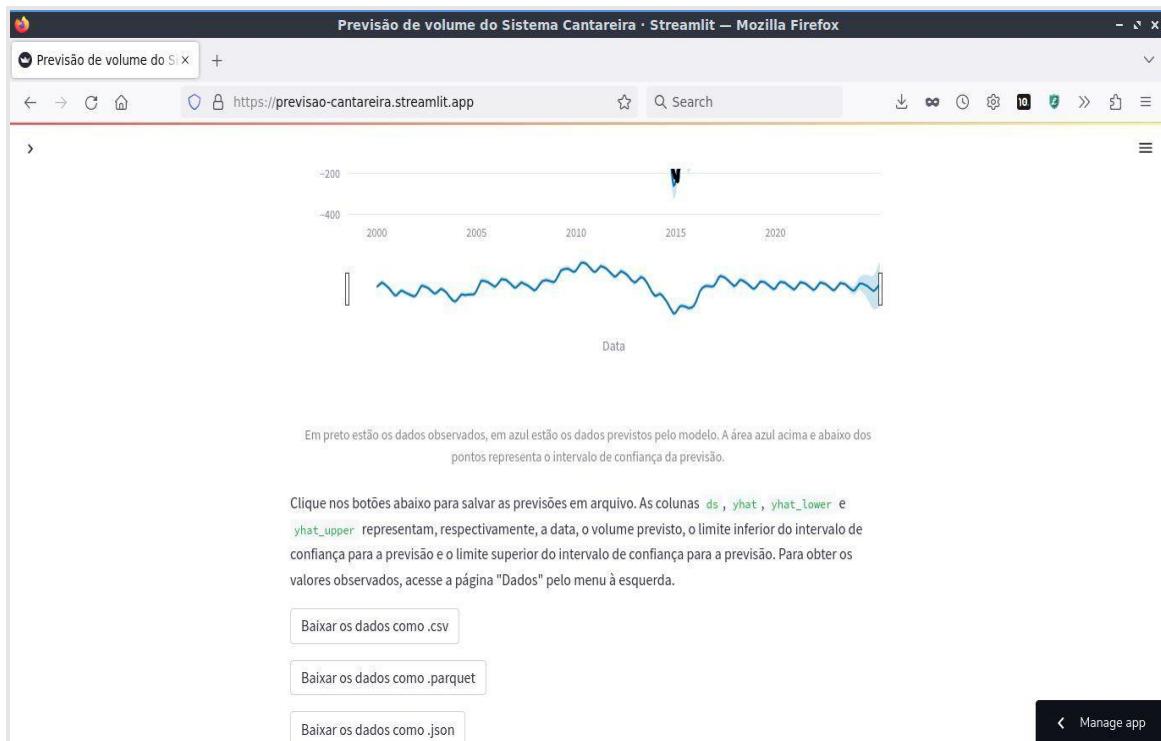
Figura 13 - Gráfico interativo da chuva na aplicação



Fonte: Elaborada pelos autores (2023)

O usuário pode navegar pelos gráficos de várias formas: selecionando um período para ampliar, escolhendo um intervalo pré-determinado (uma semana, um mês, seis meses ou um ano), arrastando o gráfico para avançar ou retroceder no tempo, etc. Para reiniciar a visão, basta clicar no ícone de casa. Abaixo do gráfico estão botões para baixar as previsões em formatos diferentes, como mostra a Figura 14.

Figura 14 - Botões para baixar os dados previstos na aplicação



Fonte: Elaborada pelos autores (2023)

Pelo menu lateral, podemos acessar a página “Dados”, que informa a fonte dos dados, a descrição dos campos e o procedimento de coleta, além de fornecer e opções para baixar a base de dados que coletamos e tratamos, conforme as Figuras 15 e 16.

Figura 15 - Página de informações sobre os dados na aplicação

A fonte dos dados do projeto é a [Sabesp](#), que disponibiliza dados referentes ao sistemas produtores. É possível ter informações sobre o Nível (m), Volume (hm³), Volume (%), Q Jusante (m³/s), Q Natural (m³/s) e Chuva (mm). De acordo com este mesmo site, temos a definição dos itens disponibilizados:

- Nível (m): refere-se ao nível da água no reservatório, em metros;
- Volume (hm³): corresponde ao volume armazenado na represa, em hectômetros cúbicos;
- Q Natural (m³/s): representa a vazão média diária afluente ao reservatório, em metros cúbicos por segundo, sem considerar as alterações antrópicas;
- Qjus (m³/s): indica a vazão média diária descarregada, em metros cúbicos por segundo, para a jusante da barragem, ou seja, a quantidade de água liberada que segue pelo rio ou canal após o barramento;
- Chuva (mm): refere-se à precipitação acumulada nas últimas 24 horas no local do barramento da represa, em milímetros.

Obtivemos os dados desde 2000 até o primeiro trimestre de 2023, incluindo apenas as represas principais (Jaguari, Jacareí, Atibaína, Cachoeira e Paiva Castro). Abaixo estão os dados de cada represa e os dados agrupados, disponíveis para download.

	Data	Represa	Nível (m)	Volume (hm ³)	Volume (%)	Q Jusante (m ³ /s)	Q Natur
0	2006-06-30 00:00:00	Jaguari/Jacareí	837.45	509.9987	63.1153	2	
1	2006-06-29 00:00:00	Jaguari/Jacareí	837.49	511.648	63.3194	2	
2	2006-06-28 00:00:00	Jaguari/Jacareí	837.52	512.8863	63.4726	2	

[Manage app](#)

Fonte: Elaborada pelos autores (2023)

Figura 16 - Botões para baixar os dados coletados na aplicação

	Data	Represa	Nível (m)	Volume (hm ³)	Volume (%)	Q Jusante (m ³ /s)	Q Natur
0	2006-06-30 00:00:00	Jaguari/Jacareí	837.45	509.9987	63.1153	2	
1	2006-06-29 00:00:00	Jaguari/Jacareí	837.49	511.648	63.3194	2	
2	2006-06-28 00:00:00	Jaguari/Jacareí	837.52	512.8863	63.4726	2	
3	2006-06-27 00:00:00	Jaguari/Jacareí	837.56	514.5393	63.6772	2	
4	2006-06-26 00:00:00	Jaguari/Jacareí	837.54	513.7125	63.5749	2	
5	2006-06-25 00:00:00	Jaguari/Jacareí	837.59	515.7804	63.8308	2	
6	2006-06-24 00:00:00	Jaguari/Jacareí	837.64	517.8517	64.0871	2	
7	2006-06-23 00:00:00	Jaguari/Jacareí	837.69	519.9262	64.3438	2	
8	2006-06-22 00:00:00	Jaguari/Jacareí	837.74	522.0042	64.601	2	
9	2006-06-21 00:00:00	Jaguari/Jacareí	837.78	523.6689	64.807	2	

[Baixar os dados como .csv](#)
 [Baixar os dados como .parquet](#)
 [Baixar os dados como .json](#)

[Manage app](#)

Fonte: Elaborada pelos autores (2023)

Por fim, a página "Sobre o modelo" descreve os modelos que utilizamos e traz considerações de análises e justificativas de escolhas de modelos.

Figura 17 - Página de informações sobre o modelo preditivo na aplicação

The screenshot shows a Streamlit application window titled "Sobre o modelo". On the left, there is a sidebar with three items: "Página inicial", "Dados", and "Sobre o modelo", with "Sobre o modelo" being the active tab. The main content area has a title "Sobre o modelo" with a book icon. Below it, there is text about the Prophet model and a link to its GitHub repository.

Nesta primeira etapa do modelo, utilizamos o [Prophet](#) para a elaboração do modelo inicial e, nesta etapa, os dados não foram separados entre dados de treino e teste, que teria como objetivo validar a eficácia do modelo. O aprofundamento da previsão, bem como a utilização de outros modelos e o estudo de correlação entre o volume e as chuvas no Sistema Cantareira, serão aprofundados no trabalho final.

O Prophet é um modelo de regressão aditivo com uma tendência de curva de crescimento linear ou logístico. Trata-se de um algoritmo disponível como um pacote para as linguagens de programação R e Python e é capaz de detectar automaticamente padrões sazonais de uma série, tornando-o bastante acessível.

[Você pode acessar o modelo no GitHub do projeto.](#)

Fonte: Elaborada pelos autores (2023)

A.3.2 Solução final

Organizando as sugestões colhidas na pesquisa de avaliação, observamos os seguintes pontos de melhoria:

1. Um usuário relatou problemas de legibilidade nos gráficos: "As cores escuras dificultam a leitura do gráfico".
2. Alguns usuários apontaram dificuldade em uso no celular. De fato, recomendamos o acesso em ambiente *desktop* devido às limitações da biblioteca de visualização utilizada (*Plotly*), que não possui adaptações para usuários de dispositivos móveis. A correção envolveria um extenso trabalho de customização, utilização de outra biblioteca ou o desenvolvimento de uma solução própria. Infelizmente, nenhuma das alternativas é viável considerando as restrições de tempo.
3. Similarmente, um usuário sugeriu: "Um *app* seria melhor pra mostrar notificações relevantes do volume ao invés de ter que acessar o site todo dia". Esse é um aspecto relevante, mas foge ao escopo deste trabalho, que se limita à visualização e disponibilização do modelo preditivo.

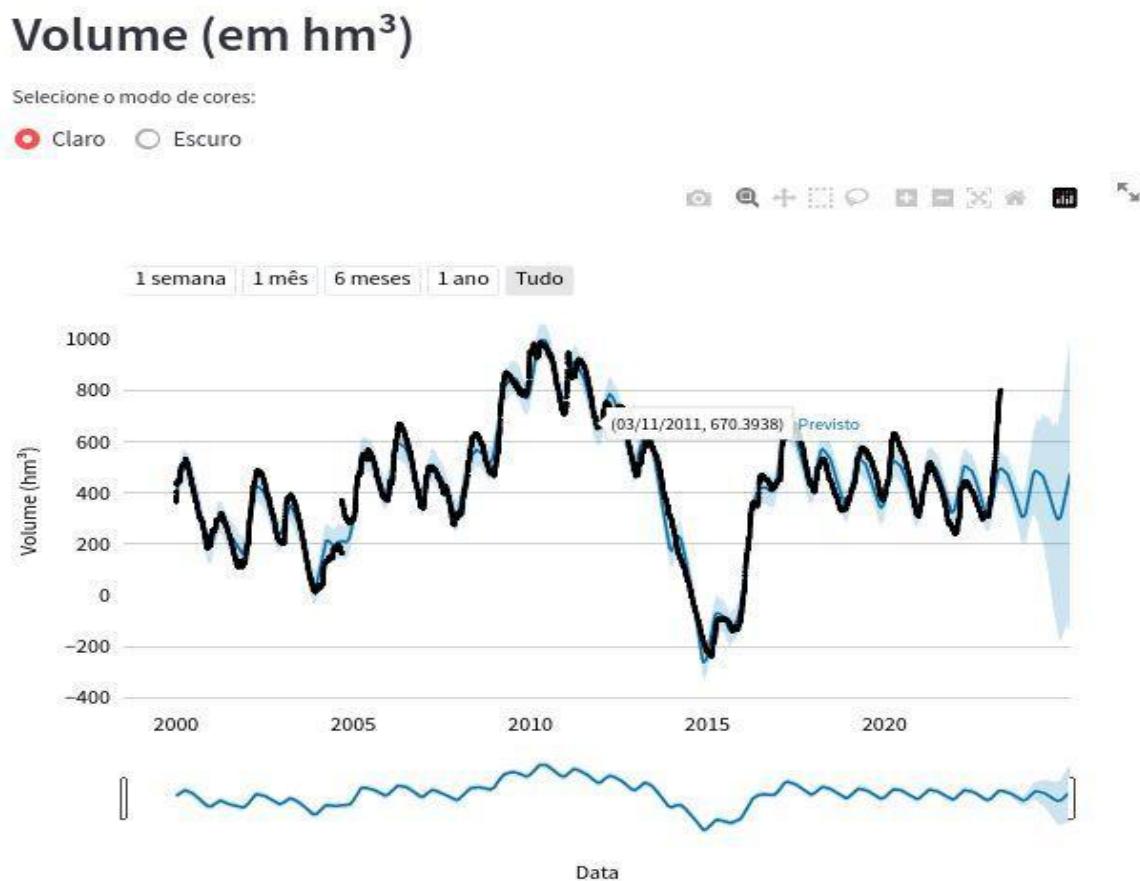
4. Houve também a sugestão de atualizar os dados diariamente, o que envolve uma série de desafios, desde a extração automatizada dos dados até a geração e monitoramento do modelo, dado que modelos preditivos precisam ser ajustados ao longo do tempo. Desse modo, deixamos a atualização diária dos dados para um trabalho futuro.

Assim, para a solução final, optamos por focar no primeiro aspecto levantado e continuar desenvolvendo o modelo.

A.3.2.1 Interface de visualização

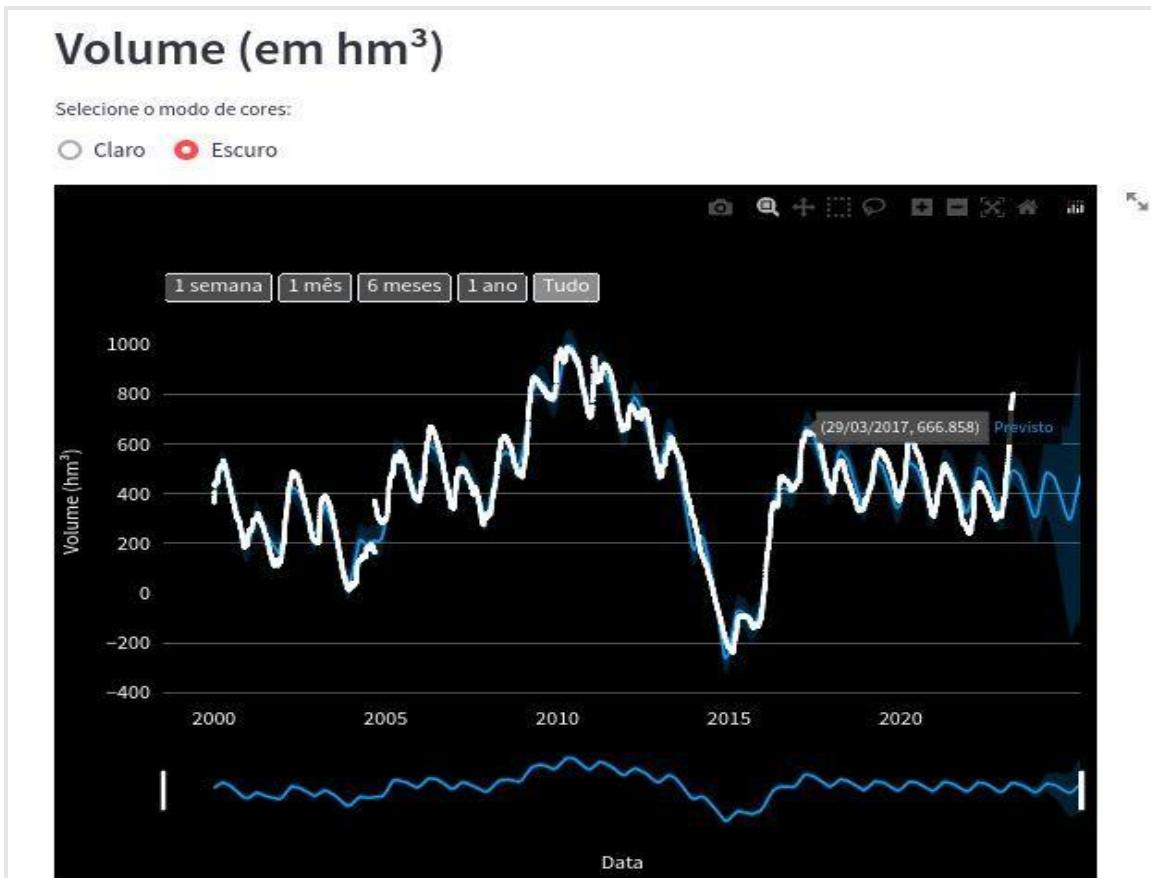
Para melhorar a legibilidade dos gráficos, aumentamos o tamanho do texto e optamos por usar cores com maior contraste. O resultado pode ser visto na Figura 18. Adicionalmente, incluímos uma opção que permite ao usuário selecionar o modo de cor: o modo claro, selecionado por padrão, e o modo escuro, exibido na Figura 19.

Figura 18 - Novo gráfico com ajustes de legibilidade na aplicação



Fonte: Elaborada pelos autores (2023)

Figura 19 - Gráfico no modo escuro na aplicação



Fonte: Elaborada pelos autores (2023)

A.3.2.2 Modelo preditivo final

Para o desenvolvimento do modelo preditivo final, optamos pelo uso da arquitetura LSTM. Uma rede LSTM é um tipo especial de arquitetura de rede neural recorrente (RNN) que é amplamente utilizada para modelagem e previsão de séries temporais. Ao contrário das RNNs tradicionais, as LSTMs são projetadas para capturar dependências de longo prazo em sequências de dados (BROWNLEE, 2018).

As LSTMs são especialmente eficazes para lidar com o desafio do "gradiente desvanecente" em RNNs, onde os gradientes usados para atualizar os pesos da rede diminuem drasticamente à medida que são propagados ao longo do tempo. Isso dificulta a aprendizagem de dependências de longo prazo em sequências temporais.

A principal diferença entre uma LSTM e uma RNN tradicional é a adição de células de memória (*memory cells*) na LSTM. Cada célula de memória é responsável por armazenar

informações relevantes e esquecer informações irrelevantes durante o processamento sequencial.

Em resumo, o funcionamento básico de uma rede LSTM consiste em:

1. Entrada (*Input*): A LSTM recebe uma sequência de entrada, que pode ser uma série temporal com várias etapas.
2. Células de Memória (*Memory Cells*): As células de memória são unidades que armazenam informações ao longo do tempo. Cada célula de memória possui três componentes principais:
 - o Estado da célula (*Cell State*): Representa a memória de longo prazo da LSTM. A informação relevante é armazenada e passada adiante por meio do estado da célula.
 - o Portão de Esquecimento (*Forget Gate*): Decide quais informações antigas devem ser esquecidas na célula de memória com base na entrada atual.
 - o Portão de Entrada (*Input Gate*): Decide quais informações devem ser atualizadas no estado da célula com base na entrada atual.
3. Atualização do Estado da Célula: Através do portão de esquecimento e do portão de entrada, a LSTM atualiza o estado da célula, combinando informações antigas e novas.
4. Saída (*Output*): A LSTM produz uma saída com base no estado atual da célula de memória. Essa saída pode ser usada para fazer previsões ou alimentar a próxima etapa da sequência.

Pelos motivos acima, as LSTMs são capazes de capturar dependências temporais de longo prazo em séries temporais, permitindo que o modelo aprenda padrões complexos e faça previsões mais precisas.

A implementação da rede LSTM foi feita na linguagem *Python* usando as bibliotecas *TensorFlow*, *Keras* e *PyTorch*, que são amplamente utilizadas para previsão em séries temporais. Essas bibliotecas fornecem interfaces simples para construir e treinar modelos LSTM, além de oferecer recursos avançados para o processamento de séries temporais, como sequências com múltiplas variáveis, janelas deslizantes e modelos com camadas LSTM empilhadas ou bidirecionais.

Nessa primeira etapa do modelo, utilizamos uma rede neural LSTM que possui uma camada visível com 1 entrada, uma camada oculta com 4 blocos LSTM ou neurônios e uma camada de saída que faz uma previsão de valor único. A função de ativação sigmoide padrão é usada para os blocos LSTM. A rede é treinada por 100 épocas e um tamanho de lote de 1 é usado.

Para validar a precisão do modelo, utilizamos o RMSE (*Root Mean Square Error*), uma métrica muito utilizada para avaliar a precisão de um modelo de regressão, comparando as previsões feitas pelo modelo com os valores reais do conjunto de dados.

O RMSE é calculado como a raiz quadrada da média dos erros ao quadrado. Ele fornece uma medida da dispersão dos erros de previsão em relação aos valores verdadeiros. Quanto menor o valor do RMSE, mais preciso é o modelo.

A fórmula para calcular o RMSE é a seguinte:

$$RMSE = \sqrt{\left(\frac{1}{n} \times \sum_{i=1}^n (y_i - \hat{y}_i)^2 \right)}$$

Onde:

- n é o número de exemplos no conjunto de dados
- y_i é o valor verdadeiro do exemplo i
- \hat{y}_i é o valor previsto pelo modelo para o exemplo i

O RMSE para os dados de treino e teste para o modelo de previsão do volume (hm^3) e de precipitação de chuva (mm) estão descritos na tabela 1 a seguir.

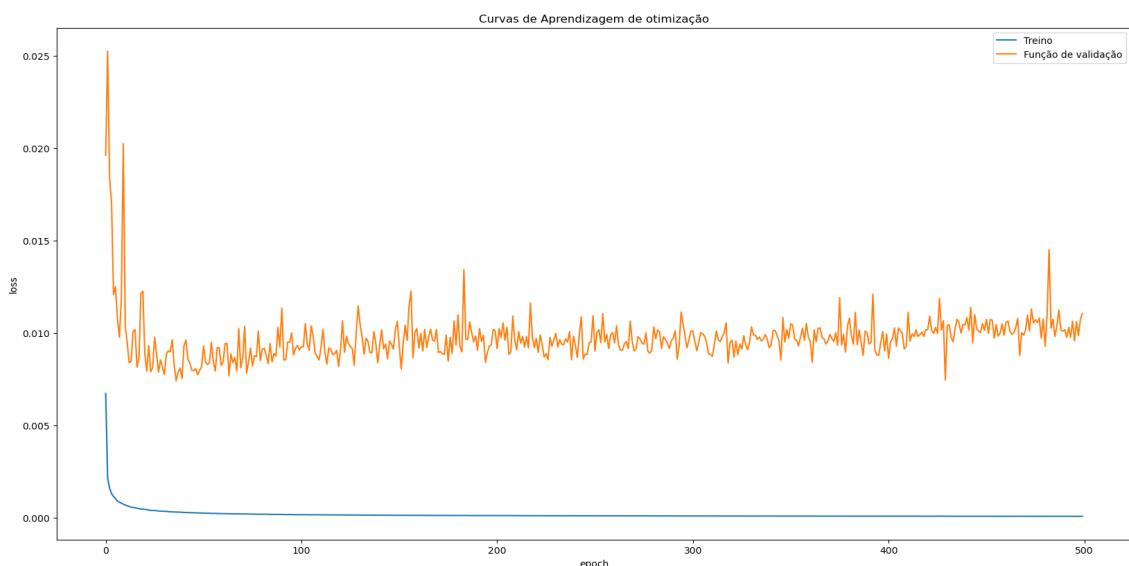
Tabela 1 – Valores RMSE para os dados de treino e teste do modelo de previsão do volume e precipitação dos reservatórios do Sistema Cantareira

	TREINO	TESTE
Volume (hm ³)	35.52	17.75
Chuva (mm)	30.23	28.44

Fonte: Elaborada pelos autores (2023)

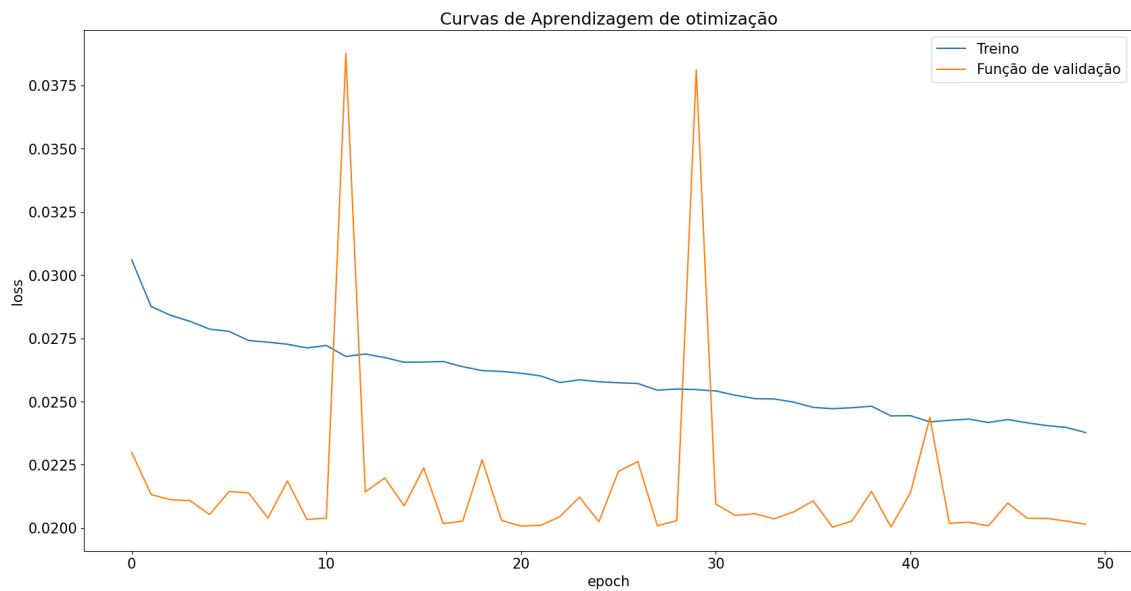
Analizando a tabela 1, é possível concluir que ambos os modelos estão aceitáveis, mas com uma precisão baixa, o que pode dificultar uma tomada de decisão mais assertiva acerca da validade do modelo.

Gráfico 8 - Curva de treino x função de validação, volume do Sistema Cantareira



Fonte: Elaborada pelos autores (2023)

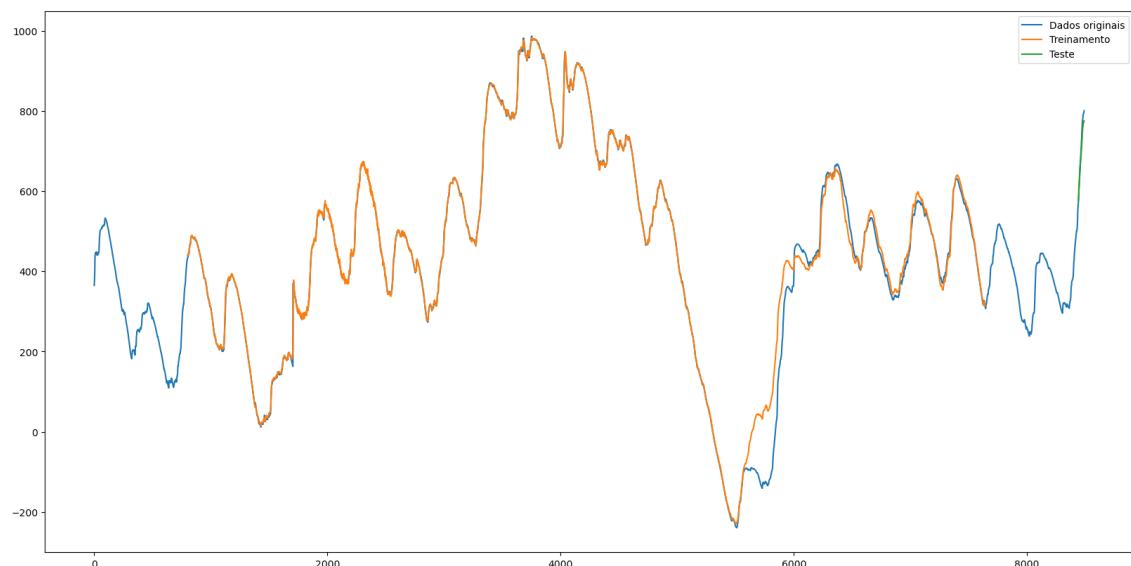
Gráfico 9 - Curva de treino x função de validação, precipitação do Sistema Cantareira



Fonte: Elaborada pelos autores (2023)

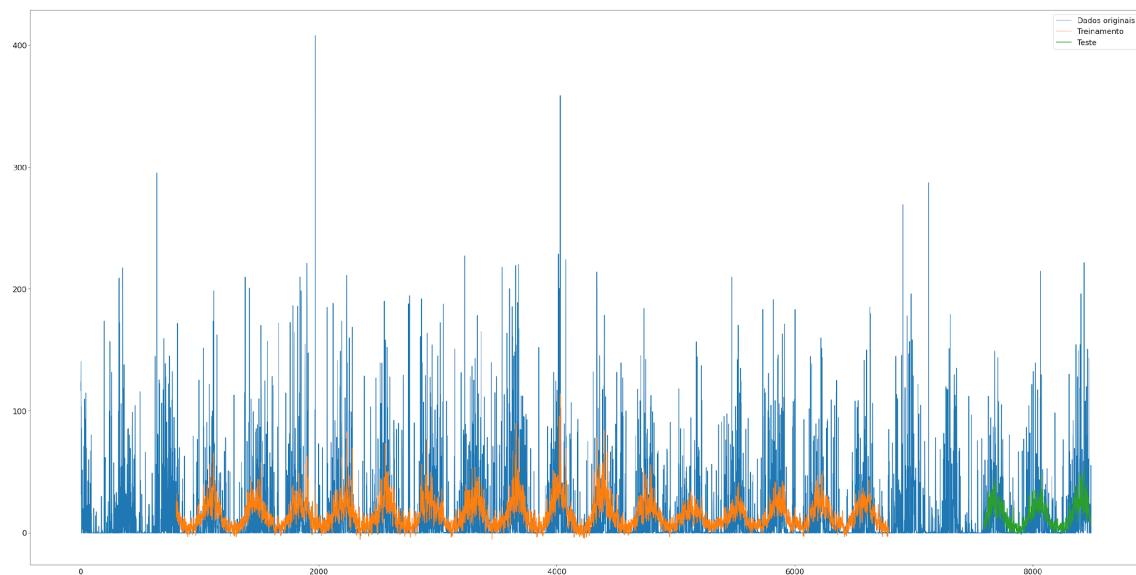
Embora as curvas de treinamento e validação não tenham convergido, podemos observar que elas estão tendendo a se aproximar à medida que o modelo avança nas épocas de treinamento. Neste caso, foram utilizadas 500 épocas para treino. É possível que, com um número maior de iterações, o desempenho do modelo melhore significativamente.

Gráfico 10 - Validação do modelo, volume do Sistema Cantareira



Fonte: Elaborada pelos autores (2023)

Gráfico 11 - Validação do modelo, precipitação do Sistema Cantareira

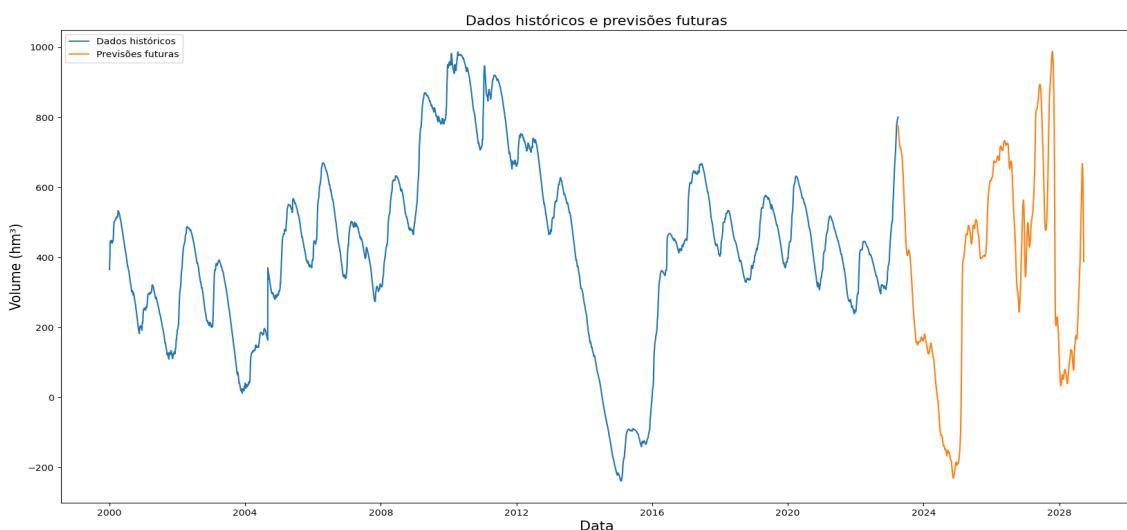


Fonte: Elaborada pelos autores (2023)

Analizando os gráficos 10 e 11, podemos observar que os modelos (linha laranja) ficaram bem aderentes aos dados originais (linha azul) e o teste conseguiu reproduzir de maneira satisfatória o comportamento esperado no dado original.

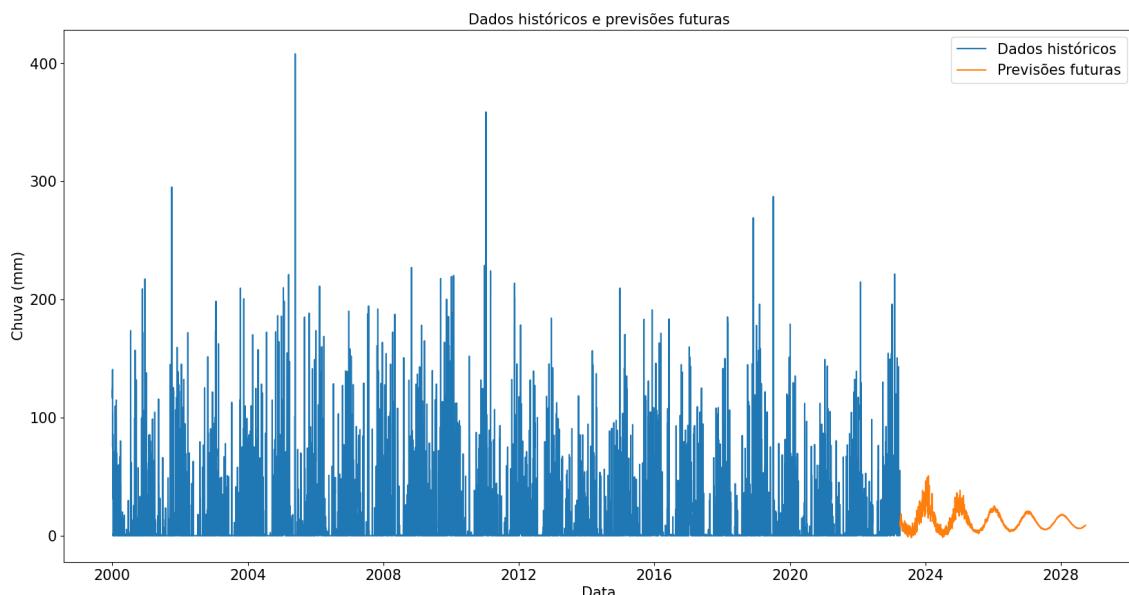
Após a validação do modelo, ele foi empregado para realizar previsões dos próximos 2.000 dias.

Gráfico 12 - Modelo de previsão do volume (hm^3) do Sistema Cantareira para 2.000 dias



Fonte: Elaborada pelos autores (2023)

Gráfico 13 - Modelo de previsão da precipitação (mm) do Sistema Cantareira para 2.000 dias



Fonte: Elaborada pelos autores (2023)

Ao analisar os gráficos 12 e 13, podemos observar que a rede LSTM exibe uma capacidade de aprendizado superior em comparação ao *Prophet*, embora o modelo *Prophet* possa ser muito útil como complemento às análises, juntamente com outros modelos de previsão de séries temporais não descritas nesse trabalho. Além disso, a rede LSTM é capaz de representar detalhadamente as series futuras com base na série histórica, incluindo as partes não sazonais.

A.4 CONSIDERAÇÕES FINAIS

O acesso à água potável, suas fontes, logística de distribuição e, principalmente, o controle para prever e enfrentar possíveis períodos de crises hídricas através das informações sobre volume dos reservatórios para tomada de decisões antecipadas ao evento, são fatores importantes para a manutenção das estruturas responsáveis por manter a vitalidade da sociedade.

O Sistema Cantareira é responsável pelo abastecimento de grande parcela da população metropolitana de São Paulo, e através de questionário distribuído aos moradores de diversas regiões da Capital, foi possível verificar a demanda e interesse por acesso facilitado a informações relacionadas ao volume das represas contempladas pelo sistema. Analisando as necessidades e sugestões apresentadas na pesquisa de interesse, foi desenvolvido um modelo preditivo do volume que contempla esse complexo, tendo por base dados extraídos desde os anos 2000 até 2023. O modelo e os dados foram disponibilizados através de uma aplicação *web* acessível ao público, que foi bem avaliada.

A aplicação teve sua interface gráfica desenvolvida em *Streamlit* e linguagem *Python*, possibilitando ao usuário interagir com os gráficos através da seleção do período e dando acesso também à base de dados original. Após o *feedback* da comunidade externa, o *design* e cores foram ajustados para maior facilidade na visualização e identificação dos dados. Foi elaborado um modelo preditivo utilizando rede neural LSTM, por ser um método mais assertivo e com previsões mais precisas. Foi possível projetar um gráfico envolvendo volume e precipitação dos próximos 2.000 dias, contemplando assim o objetivo central do projeto. Infelizmente, não pudemos atender a algumas sugestões da comunidade neste trabalho, como o aprimoramento do acesso através de dispositivos móveis e atualizações em tempo real da base de dados. Também não exploramos a fundo a relação entre chuva e volume na nossa análise, de modo que o objetivo secundário foi parcialmente cumprido.

Devido à importância do Sistema Cantareira e as consequências de crises hídricas já vistas historicamente em todo o Brasil, principalmente na região Sudeste, o projeto tem impactos sociais importantes. Através dos modelos disponibilizados, com acesso facilitado e entendimento dos dados por meio de uma interface, a população tem a possibilidade de participar de forma mais efetiva e assídua na gestão de recursos hídricos.

Quanto às contribuições acadêmicas, destacamos que, embora não tenhamos conseguido explorar tantos modelos quanto alguns trabalhos anteriores, trouxemos uma contribuição distinta devido ao nosso foco no Sistema Cantareira e à disponibilização de uma interface de visualização ao público. Adicionalmente, o modelo e a aplicação são de código aberto e publicados em um repositório público, então podem ser aprimorados por pesquisadores e desenvolvedores.

Para aprimorar o desempenho dos modelos, é fundamental considerar uma variedade de fatores, tanto quantitativos quanto qualitativos. Isso inclui a avaliação do impacto ambiental na cobertura florestal da bacia hidrográfica e nos mananciais que compõem o Sistema Cantareira, bem como as mudanças climáticas e a manutenção regular dos sensores de coleta de dados.

Além disso, é importante destacar o papel das ações sociais, como a conscientização da população acerca da conservação dos mananciais e uso consciente dos recursos hídricos. Essas ações podem desempenhar um papel significativo na melhoria do cenário do Sistema Cantareira. Ao considerar todas essas medidas e implementá-las de forma integrada, é possível obter avanços significativos na compreensão e gestão desse importantíssimo sistema. Isso permitirá uma abordagem mais eficaz para garantir a segurança e sustentabilidade dos recursos hídricos.