UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA HIDRÁULICA E AMBIENTAL
PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA CIVIL

ANA CINTHYA MARIANO DE SOUSA

PROPOSTAS DE MODELOS PARA DIMENSIONAMENTO DE DISPOSITIVOS DE CONTROLE DE PERCOLAÇÃO INTERNA - ABRAÇOS - EM BARRAGENS DE TERRA POR MEIO DE TÉCNICAS DE APREDIZADO DE MÁQUINA

FORTALEZA
2025

ANA CINTHYA MARIANO DE SOUSA

PROPOSTAS DE MODELOS PARA DIMENSIONAMENTO DE DISPOSITIVOS DE CONTROLE DE PERCOLAÇÃO INTERNA - ABRAÇOS - EM BARRAGENS DE TERRA POR MEIO DE TÉCNICAS DE APREDIZADO DE MÁQUINA

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Civil da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia Civil. Área de concentração: Geotecnia.
Orientador: Prof. Dr. Silvrano Adonias Dantas Neto.

FORTALEZA

2025

ANA CINTHYA MARIANO DE SOUSA

PROPOSTAS DE MODELOS PARA DIMENSIONAMENTO DE DISPOSITIVOS DE CONTROLE DE PERCOLAÇÃO INTERNA - ABRAÇOS - EM BARRAGENS DE TERRA POR MEIO DE TÉCNICAS DE APREDIZADO DE MÁQUINA

Dissertação apresentada ao curso de Mestrado Acadêmico em Geotecnia do Programa de Pós-Graduação em Engenharia Civil do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Geotecnia. Área de Concentração: Geotecnia.

Aprovado em: _____/_____/_____.

BANCA EXAMINADORA

Prof. Dr. Silvrano Adonias Dantas Neto (Orientador)
Universidade Federal do Ceará

Profa. Dr. Mariana Vella Silveira
Universidade Federal do Ceará

Prof. Dr. Adriano Frutuoso da Silva
Universidade Federal de Roraima

AGRADECIMENTOS

Gostaria de iniciar expressando minha profunda gratidão à minha família, que esteve ao meu lado em todos os momentos, oferecendo apoio incondicional e incentivo constante. À minha mãe, Socorro, e à minha avó, Fransquinha, por todo o amor e força que me deram, e por me inspirarem a nunca desistir. Às minhas tias, Mauis, Fatinha e Ivonete, e aos meus primos, Myrian e Matheus, pelo carinho e apoio contínuos. Ao meu irmão, David, que sempre foi um exemplo de dedicação e perseverança, inspirando-me a seguir firme em minha trajetória acadêmica.

Agradeço imensamente ao meu orientador, Professor Silvrano Dantas, pela constante disponibilidade, paciência e orientação valiosa ao longo de todo o processo de pesquisa. Sua dedicação e comprometimento foram essenciais para o desenvolvimento deste trabalho.

Registro aqui minha gratidão aos professores Alfran Sampaio, Francisco Chagas, Mariana Vella e Anderson Borghetti, pelo conhecimento compartilhado e pelas experiências que contribuíram significativamente para o meu crescimento profissional.

À Universidade Federal do Ceará (UFC) e ao Departamento de Engenharia Hidráulica e Ambiental (DEHA), meu sincero agradecimento pela estrutura e pelas oportunidades oferecidas ao longo da minha formação acadêmica. Esta instituição foi fundamental para o meu desenvolvimento intelectual e pessoal.

Aos meus amigos Samuel Mariano, Victor Mariano, Bruna Lemos, Thais Vasconcelos e Clarisse Vasconcelos, agradeço pelo companheirismo, amizade e apoio em todos os momentos. Vocês foram essenciais para enfrentar os desafios dessa jornada.

Por fim, agradeço à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo apoio financeiro que possibilitou a realização desta pesquisa.

A todos, o meu mais sincero obrigado.

Slow down, youre doing fine,

You cant be everything you wanna be before your time. (Billy Joel)

RESUMO

Esta pesquisa desenvolve modelos preditivos voltados ao dimensionamento otimizado de dispositivos de controle de percolação interna em barragens de terra, comumente denominados abraços, os quais desempenham papel fundamental na mitigação de processos erosivos ao longo da interface solo/estrutura. Para tal, foi adotada uma abordagem integrada que combina simulações computacionais tridimensionais baseadas no Método dos Elementos Finitos (MEF) com técnicas de aprendizado de máquina, possibilitando a análise aprofundada do comportamento do fluxo hidráulico em diferentes cenários geotécnicos, hidráulicos e geométricos. A modelagem numérica considerou uma série de variações paramétricas, incluindo diferentes valores de condutividade hidráulica, graus de saturação, características físicas das fundações, além de geometrias distintas para os dispositivos de controle. Como resultado, foi gerado um banco de dados robusto e representativo, o qual serviu como base para o treinamento de diferentes algoritmos de aprendizado de máquina, notadamente Redes Neurais Artificiais, Árvores de Decisão, Florestas Aleatórias e o modelo XGBoost. Dentre os modelos avaliados, o XGBoost apresentou o desempenho mais satisfatório, alcançando um coeficiente de determinação (R^2) de 96.58 % no conjunto de dados de treinamento e 90,58 % no conjunto de teste, aliado a métricas de erro reduzidas (Erro Médio Absoluto - MAE = 7.03; Raiz do Erro Quadrático Médio - RMSE = 16.38; Erro Percentual Absoluto Médio - MAPE = 13.7 %), o que evidencia a capacidade do modelo em realizar predições do comprimento ideal dos dispositivos. A utilização desses modelos permitiu identificar o comprimento mais eficiente do dispositivo abraço, promovendo a redução de gradientes hidráulicos críticos e, consequentemente, a prevenção de mecanismos de erosão interna. Assim, os resultados obtidos fornecem subsídios técnicos relevantes para o aprimoramento de projetos de barragens de terra, especialmente em contextos onde há limitação de dados experimentais, configurando-se como uma solução eficiente e economicamente viável no campo da engenharia geotécnica aplicada à segurança de obras hidráulicas.

Palavras-chave: Estrutura de abraço; Aprendizado de máquina; Modelagens numéricas; Fluxo tridimensional; Barragens de terra.

ABSTRACT

This research develops predictive models aimed at the optimized design of internal seepage control devices in earth dams, commonly referred to as wraparound, which play a key role in mitigating erosion processes along the soilstructure interface. To achieve this, an integrated approach was adopted, combining three-dimensional computational simulations based on the Finite Element Method (FEM) with machine learning techniques, enabling an in-depth analysis of hydraulic flow behavior across a wide range of geotechnical, hydraulic, and geometric conditions. The numerical modeling accounted for a series of parametric variations, including different values of hydraulic conductivity, degrees of saturation, foundation characteristics, and diverse geometries of the control devices. As a result, a robust and representative dataset was generated and used to train various machine learning algorithms, notably Artificial Neural Networks, Decision Trees, Random Forests, and the XGBoost model. Among the models evaluated, XGBoost exhibited the most satisfactory performance, achieving a coefficient of determination (Rš) of 96.58% for the training dataset and 90.58% for the test dataset, along with low error metrics (Mean Absolute Error MAE = 7.03; Root Mean Square Error RMSE = 16.38; Mean Absolute Percentage Error MAPE = 13.7%), demonstrating the models strong predictive capability in estimating the optimal device length. The use of these models enabled the identification of the most effective abraço configurations, leading to a reduction in critical hydraulic gradients and, consequently, the prevention of internal erosion mechanisms. Therefore, the findings provide valuable technical insights for improving earth dam design, particularly in contexts with limited experimental data, establishing a reliable, efficient, and cost-effective solution within geotechnical engineering applied to the safety of hydraulic structures.

Keywords: Embracement wraparound; Machine learning; Numerical modeling; Three-dimensional flow; Earth dams.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

As barragens constituem infraestruturas críticas para o abastecimento hídrico, atendendo às demandas populacionais quanto à quantidade, qualidade, localização e temporalidade dos recursos, sendo fundamentais para o desenvolvimento sustentável. A Lei n 12.334/2010 estabelece diretrizes para garantir a segurança estrutural dessas obras, visando à proteção de vidas, saúde pública, patrimônio e meio ambiente, além de definir as responsabilidades dos empreendedores e órgãos fiscalizadores, fomentando uma cultura de segurança, gestão de riscos e participação social no monitoramento dessas estruturas (BRASIL, 2010).

A percolação hídrica em barragens de terra representa um fator crítico que afeta diretamente a segurança e desempenho dessas estruturas. Como demonstrado por GAIOTO (1992), o incremento do grau de saturação do solo reduz substancialmente sua resistência ao cisalhamento, predispondo à ocorrência de processos erosivos, particularmente na região de jusante. Este mecanismo pode comprometer tanto a estabilidade global da barragem quanto sua integridade estrutural. Complementarmente, SOWERS (1962) alerta que a falta de controle adequado da percolação pode gerar dois problemas críticos: (i) perdas hídricas excessivas por vazamento, reduzindo a eficiência do reservatório, e (ii) gradientes hidráulicos elevados, que aumentam as forças de percolação e os riscos de instabilidade. Portanto, o controle eficaz da percolação deve implementar estratégias que simultaneamente mitiguem esses efeitos adversos e garantam a estabilidade estrutural e funcionalidade da barragem.

Na engenharia de barragens de terra, uma estratégia consolidada para controle de gradientes hidráulicos na interface solo-estrutura consiste na implementação de dispositivos denominados "abraços". Esses elementos estruturais, projetados como extensões integradas aos muros de concreto dos vertedouros, atuam aumentando o caminho de percolação, promovendo a dissipação gradativa de energia hidráulica e mitigando os riscos de erosão na zona de contato entre o maciço terroso e as estruturas rígidas. No entanto, a literatura especializada evidencia uma significativa lacuna técnica quanto aos critérios de dimensionamento desses dispositivos. Essa carência normativa gera incertezas nos projetos executivos, podendo comprometer tanto a eficácia funcional quanto a segurança estrutural das barragens. Diante desse cenário, torna-se imperativo o desenvolvimento de pesquisas aplicadas que avancem nas metodologias de dimensionamento, visando a otimização técnica e a garantia de soluções mais eficientes, seguras e

confiáveis para esses elementos críticos.

Diversos estudos têm demonstrado a eficácia de modelos de regressão e técnicas de machine learning na análise e previsão de comportamentos críticos em barragens, evidenciando que tais abordagens constituem ferramentas essenciais para lidar com a complexidade inerente a essas estruturas (TAYFUR *et al.*, 2005; TOPOK; CIGIZOGLU, 2008; UNES, 2010; XUE *et al.*, 2014). Os modelos numéricos baseados no Método dos Elementos Finitos (MEF) são particularmente eficazes na estimativa de deslocamentos e tensões em barragens; contudo, sua precisão está intimamente relacionada à correta caracterização dos materiais, especialmente no que diz respeito às propriedades mecânicas e hidráulicas, que frequentemente apresentam incertezas significativas (SALAZAR; al., 2015). A aplicação desses modelos requer, ainda, a adoção de simplificações e suposições quanto à geometria estrutural, às condições de contorno e aos parâmetros de entrada, fatores que podem comprometer a confiabilidade dos resultados obtidos. Nesse contexto, a integração de técnicas de machine learning com dados históricos e variáveis externas tem se mostrado promissora, ao complementar as metodologias tradicionais e proporcionar maior robustez à modelagem de barragens.

Com base nesse contexto, a presente pesquisa tem como objetivo principal desenvolver uma metodologia preditiva voltada ao dimensionamento otimizado de dispositivos de controle de percolação interna em barragens de terra, com foco específico na interface solo/estrutura, visando mitigar gradientes hidráulicos críticos e prevenir a ocorrência de mecanismos de erosão interna. A proposta busca suprir uma lacuna identificada na literatura técnica, ao oferecer um embasamento científico robusto que auxilie engenheiros na concepção, dimensionamento e implementação de soluções eficazes para o controle da percolação. Para tal, foram realizadas simulações computacionais tridimensionais de fluxo, por meio da aplicação do Método dos Elementos Finitos (MEF), as quais permitiram a construção de um banco de dados estruturado, abrangendo uma ampla variedade de condições geotécnicas, geométricas e hidráulicas. A partir desse banco, empregaram-se técnicas de aprendizado de máquina, reconhecidas por sua capacidade de modelar fenômenos não lineares e de elevada complexidade, com o propósito de desenvolver modelos preditivos aptos a estimar o comprimento ideal do dispositivo de controle, abraço. A abordagem integrada proposta revela-se especialmente promissora para o aprimoramento da segurança e estabilidade de barragens de terra, sobretudo em contextos nos quais há limitação de dados experimentais disponíveis.

1.1 Objetivos

1.1.1 Objetivo principal

Comparar métodos de aprendizado de máquina Redes Neurais Artificiais (RNAs), Árvores de Decisão, Florestas Aleatórias (Random Forest) e XGBoost para identificar o modelo mais eficaz na previsão do comprimento do dispositivo de controle de percolação (abraço) em barragens de terra, considerando métricas de desempenho, robustez e eficiência computacional.

1.1.2 Objetivos específicos

- **Avaliar o desempenho preditivo dos modelos:**
  Aplicar e comparar RNAs, Árvores de Decisão, Florestas Aleatórias e XGBoost utilizando métricas consolidadas (Rš, MAE, RMSE e MAPE) para identificar o algoritmo com maior precisão na estimativa do comprimento do abraço.

- **Analisar a capacidade de generalização, distribuição dos resíduos, robustez e custo computacional dos modelos:**
  Avaliar as curvas de aprendizado para identificar overfitting, comparar histogramas de resíduos para medir viés, precisão e presença de outliers, aplicar validação cruzada k-fold para verificar a estabilidade dos resultados e analisar o custo computacional em relação ao desempenho e à viabilidade prática de cada modelo.

- **Recomendar o modelo ideal para dimensionamento do abraço:**
  Selecionar o modelo que apresente o maior Rš e os menores erros (MAE, RMSE, MAPE) no conjunto de teste, com resíduos bem comportados (distribuição aproximadamente normal e ausência de viés), baixa propensão ao overfitting e custo computacional compatível com aplicações práticas em projetos de engenharia.

1.2 Estrutura do trabalho

Esta dissertação está organizada em cinco capítulos, cada um abordando aspectos específicos da pesquisa. O Capítulo 1 introduz o tema de estudo, apresentando sua relevância, motivação e objetivos. São definidos claramente o objetivo geral - desenvolver uma metodologia preditiva para dimensionamento de dispositivos de controle de percolação - e os objetivos

específicos, que incluem a análise computacional do fenômeno e a aplicação de técnicas de machine learning.

O Capítulo 2 realiza uma revisão bibliográfica abrangente, dividida em três eixos principais. Primeiramente, aborda os fundamentos teóricos da percolação em barragens de terra e os mecanismos de erosão interna. Em seguida, detalha os dispositivos de controle existentes, com ênfase nos "abraços". Por fim, apresenta os conceitos de machine learning relevantes para a pesquisa, incluindo Redes Neurais Artificiais, Árvores de Decisão, Florestas Aleatórias e XGBoost, destacando suas aplicações em problemas geotécnicos.

No Capítulo 3, é descrita a metodologia adotada. Esta seção detalha o processo de modelagem computacional tridimensional utilizando o Método dos Elementos Finitos (MEF), incluindo a parametrização dos modelos e as condições de contorno consideradas. Também são apresentados os procedimentos para geração do banco de dados e as técnicas de pré-processamento aplicadas. A seção final do capítulo explica a implementação dos algoritmos de machine learning e os critérios de validação adotados.

O Capítulo 4 apresenta e discute os resultados obtidos. Inicialmente, são mostrados os dados das simulações numéricas, com análise dos campos de fluxo e gradientes hidráulicos. Na sequência, são comparados os desempenhos dos diferentes modelos preditivos, utilizando métricas como coeficiente de determinação (R^2) e erro médio absoluto (MAE). O capítulo inclui ainda uma análise de sensibilidade dos parâmetros mais relevantes para o dimensionamento dos dispositivos.

O Capítulo 5 consolida as principais conclusões desta pesquisa, ressaltando suas contribuições metodológicas e aplicadas no campo da engenharia geotécnica, ao mesmo tempo em que delineia recomendações estratégicas para investigações futuras, com vistas à continuidade e aprofundamento das discussões aqui iniciadas.

2 REVISÃO DE LITERATURA

Este capítulo apresenta uma revisão da literatura pertinente, organizada em duas partes principais. A primeira parte aborda os fundamentos das equações hidrodinâmicas que descrevem o fluxo de água nos solos, incluindo a Lei de Darcy, o conceito de gradiente hidráulico e a equação de continuidade. Também é discutido o fenômeno da erosão interna, com destaque para os mecanismos que o caracterizam como escoamento concentrado, erosão regressiva, erosão de contato e sufusão e a importância do tratamento das interfaces, como a solo-concreto.

A segunda parte explora o uso de técnicas de Aprendizado de Máquina como uma abordagem complementar na análise de problemas geotécnicos. São revisados os conceitos e as aplicações de diversas metodologias de Aprendizado de Máquina, como Redes Neurais Artificiais (RNAs), Árvores de Decisão, Florestas Aleatórias e métodos de Boosting com árvores de regressão, que se têm mostrado eficazes na modelagem de sistemas complexos e na previsão de fenômenos geotécnicos.

Ao integrar os fundamentos da geotecnia com as capacidades das técnicas de Aprendizado de Máquina, este capítulo visa fornecer o embasamento teórico necessário para o desenvolvimento de metodologias na análise e mitigação dos riscos associados à percolação e à erosão interna em estruturas geotécnicas.

2.1 Equações hidrodinâmicas

A estrutura do solo, sendo um sistema trifásico, responde de maneira altamente não linear às mudanças no teor de água e nas condições ambientais, o que, juntamente com sua natureza anisotrópica e heterogênea, torna o estudo e a previsão de seu comportamento um desafio significativo. Essa complexidade limita o desenvolvimento de soluções analíticas e numéricas para diversos problemas geotécnicos (BAGHBANI *et al.*, 2022). Nesse contexto, a percolação da água no solo, entendida como o processo de movimentação da água através dos vazios interconectados, é um fenômeno fundamental para a análise da estabilidade das estruturas de contenção, como barragens. A análise detalhada da perda de carga e do gradiente hidráulico é fundamental para o controle da percolação, sendo um fator crítico na garantia da estabilidade, durabilidade e segurança das infraestruturas de engenharia geotécnica (DAS; SOBHAN, 2007; TERZAGHI *et al.*, 1969).

SOWERS (1962) destaca que a percolação descontrolada pode ocasionar problemas significativos, como o aumento de vazamentos e a elevação do gradiente hidráulico, afetando diretamente a segurança das estruturas hidráulicas. Os autores identificam quatro principais efeitos da percolação: (i) a velocidade do fluxo de água pode causar o desagregamento das partículas e promover o rearranjo das partículas finas do solo, (ii) o fluxo pode induzir condições de liquefação, semelhantes a areia movediça, (iii) a pressão interna da água pode reduzir a coesão do solo, levando ao seu enfraquecimento, e (iv) o aumento dos gradientes hidráulicos pode desencadear a erosão interna, processo conhecido como piping.

2.1.1 Lei de Darcy

A Lei de Darcy é um dos fundamentos centrais na análise de fluxo de água através de solos saturados, sendo amplamente aplicada em engenharia geotécnica, como em projetos de drenagem, controle de recalques e monitoramento da percolação em barragens de terra. A equação da Lei de Darcy estabelece uma relação linear entre a velocidade de percolação e o gradiente hidráulico, expressa pela Equação 1 (DAS; SOBHAN, 2007; PINTO, 2002).

$$v = -k\nabla h \tag{1}$$

Onde v é a velocidade de descarga, k é a condutividade hidráulica do solo, e ∇h é o gradiente hidráulico, que representa a variação da altura do potencial hidráulico h ao longo do meio.

A velocidade de descarga $v$refere-se à quantidade de água que atravessa uma unidade de área transversal do solo em um intervalo de tempo. Contudo, a velocidade efetiva de percolação v_e, que considera apenas os espaços vazios no solo, é geralmente maior que v. A relação entre essas duas velocidades pode ser expressa pela Equação 2 (LAMBE; WHITMAN, 1979).

$$v_e = \frac{v}{n} \tag{2}$$

Onde v_e é a velocidade efetiva de percolação e n é a porosidade do solo.

A permeabilidade k é a propriedade do solo que permite o escoamento da água, sendo influenciada por fatores como granulometria, estrutura do solo e viscosidade do fluido. O coeficiente k é determinado experimentalmente por ensaios de permeabilidade, como os realizados em laboratório ou in situ.

A Lei de Darcy apresenta limitações técnicas que devem ser cuidadosamente consideradas em determinadas condições (DAS; SOBHAN, 2007; TERZAGHI *et al.*, 1969). Essas

restrições decorrem das premissas idealizadas que embasam a teoria e podem impactar sua aplicabilidade prática:

- **Fluxo constante e incompressibilidade do solo:** A Lei de Darcy pressupõe que o fluxo é constante e que o solo é incompressível. Entretanto, em condições dinâmicas, onde as propriedades do solo ou do fluxo variam com o tempo, essas premissas podem não se aplicar, comprometendo a precisão dos resultados.

- **Restrição a fluxos laminares:** A validade da Lei de Darcy é limitada a regimes de fluxo laminar, ou seja, velocidades de escoamento abaixo do limite crítico que marca a transição para o fluxo turbulento. Em situações onde o fluxo se torna turbulento, a teoria deixa de ser aplicável, exigindo abordagens mais complexas.

- **Condição de saturação:** A teoria foi desenvolvida para solos saturados, sendo inadequada para modelar diretamente o fluxo em solos parcialmente saturados. Nessas condições, efeitos adicionais, como capilaridade e variações no grau de saturação, devem ser considerados para uma análise realista.

- **Dependência de condições de contorno:** A aplicação prática da Lei de Darcy requer informações precisas sobre as condições de contorno, como os níveis de entrada e saída da água. Em situações de campo, obter essas informações com a precisão necessária pode ser um desafio, introduzindo incertezas no modelo.

Essas limitações destacam a necessidade de avaliação crítica ao utilizar a Lei de Darcy em projetos geotécnicos, especialmente em contextos onde o fluxo apresenta características dinâmicas, turbulentas ou envolve solos não saturados. A compreensão dessas restrições permite uma aplicação mais consciente e fundamentada da teoria, garantindo maior confiabilidade nas análises de fluxo (BEAR, 1972).

2.1.2 Gradiente hidráulico

A tensão efetiva do solo é diretamente afetada pelo fluxo de água em seus vazios, com o gradiente hidráulico desempenhando um papel central nesse processo, especialmente em situações de fluxo ascendente. O gradiente hidráulico é determinado pela relação entre a perda de carga e o comprimento do percurso de percolação, conforme apresentado na Equação 3, posicionando-se como um dos parâmetros mais relevantes para a modelagem e o controle da

percolação em aplicações geotécnicas (TERZAGHI *et al.*, 1969).

$$i = \frac{\Delta H}{L} \tag{3}$$

Quando a força de percolação gerada por um fluxo ascendente no solo se iguala ao peso das partículas sólidas submersas, as tensões efetivas no solo tornam-se nulas, especialmente em solos arenosos de granulação fina. Nessa condição, ocorre a liquefação do material, caracterizada pela perda da resistência ao cisalhamento devido à ausência de contato efetivo entre as partículas. Esse fenômeno está intimamente associado ao gradiente hidráulico crítico, o qual representa o limite de estabilidade do solo na presença de fluxos ascendentes (OLIVEIRA, 2012).

WANG *et al.* (2016) conduziram estudos experimentais aprofundados que analisaram minuciosamente a interação entre o processo de erosão, o gradiente hidráulico crítico e as propriedades mecânicas de distintos tipos de solos. Os resultados obtidos reforçam os achados de FELL *et al*. (2004b), mencionados pela USBR (2019), destacando três aspectos principais:

1. **Solos altamente porosos:** Esses solos apresentam baixa resistência ao fluxo ascendente devido à sua elevada permeabilidade e à menor interação entre partículas. Como resultado, a erosão se inicia em gradientes hidráulicos críticos baixos, tornando-os mais suscetíveis a processos de piping e liquefação.

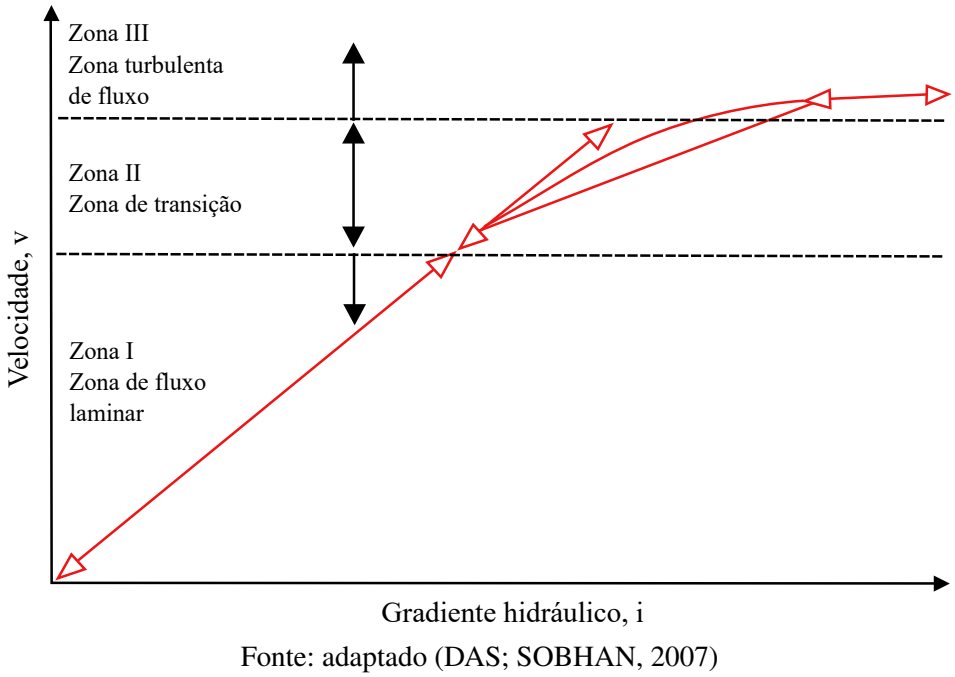
2. **Solos com frações de finos argilosos:** Solos contendo proporções significativas de partículas argilosas tendem a resistir melhor à erosão, devido às forças eletroquímicas entre partículas que aumentam a coesão. Nesses casos, o gradiente hidráulico crítico necessário para iniciar o deslocamento de partículas é relativamente alto, indicando maior estabilidade.

3. **Solos compactados e de maior densidade:** Solos densos e bem compactados apresentam menor porosidade e maior interação partícula-partícula, o que resulta em maior resistência à ação de gradientes hidráulicos críticos. Isso reflete a importância de práticas de compactação no controle de percolação e na estabilidade de barragens e fundações permeáveis.

REYNOLDS (1883) identificaram que a relação linear entre o gradiente hidráulico e a velocidade média do fluxo apresenta uma transição significativa quando o gradiente atinge um valor crítico, indicando uma mudança no comportamento do fluxo. De acordo com DAS e SOBHAN (2007), a variação da velocidade (v) em função do gradiente hidráulico (iii) pode ser

analisada de acordo com três regimes distintos de fluxo, conforme ilustrado na Figura 1. A figura divide o fluxo em três zonas: Zona I, correspondente ao fluxo laminar; Zona II, representando a zona de transição; e Zona III, associada ao fluxo turbulento. Quando o gradiente hidráulico é aumentado gradualmente, o fluxo permanece laminar nas Zonas I e II, sendo que a velocidade v apresenta uma relação linear com o gradiente hidráulico. Contudo, ao atingir um gradiente hidráulico mais elevado, o fluxo transita para o regime turbulento, caracterizando a Zona III. Caso o gradiente hidráulico seja reduzido, as condições de fluxo laminar retornam, mas restritas apenas à Zona I. Esse comportamento reflete a mudança nas características do fluxo em função do gradiente hidráulico, sendo crucial para a análise de processos de percolação e de transporte de fluidos em meios porosos.

Figura 1 – Regimes de fluxo



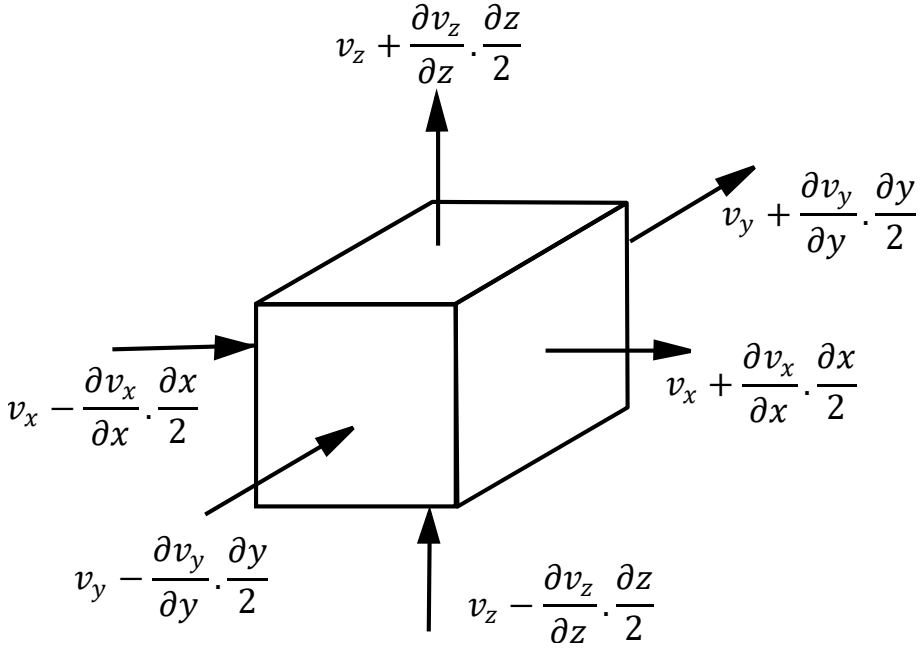
Fonte: adaptado (DAS; SOBHAN, 2007)

2.1.3 Equação de continuidade

A equação de continuidade garante que o fluxo de água seja consistente dentro de um elemento de volume no solo. Ela expressa a conservação da massa, ou seja, a quantidade de água que entra ou sai de um elemento de solo deve ser igual à variação do volume de água armazenado nesse elemento de solo ao longo do tempo (BEAR, 1972).

Para um elemento de volume de dimensões dx, dy e dz, representando na Figura 2 a quantidade de fluxo de água nas direções x, y e z pode ser expressa como:

Figura 2 – Representação do fluxo volumétrico nas direções *x*, *y* e *z*



Fonte: adaptado (TERZAGHI *et al.*, 1969)

- Vazão na direção *x* : $v_x\ dydz$
- Vazão na direção *y* : $v_y\ dxdz$
- Vazão na direção *z* : $v_z\ dxdy$

Se considerarmos as variações de fluxo nas direções *x*, *y* e *z* ao longo de um pequeno deslocamento, temos:

- Vazão de entrada na direção *x*:

$$\left(v_x - \frac{\delta v_x}{\delta x}\ \frac{dx}{2}\right)$$

- Vazão de saída na direção *x*:

$$\left(v_x + \frac{\delta v_x}{\delta x}\ \frac{dx}{2}\right)$$

Similarmente, para as direções *y* e *z*:

- Vazão de entrada e saída na direção *y*: $\frac{\delta v_y}{\delta y}\ dxdy$
- Vazão de entrada e saída na direção *z*: $\frac{\delta v_z}{\delta z}\ dxdy$

A equação da continuidade, que descreve a variação temporal do volume de água armazenado em um sistema, é representada pela Equação 4:

$$\left(\frac{\delta v_x}{\delta x} + \frac{\delta v_y}{\delta y} + \frac{\delta v_z}{\delta z}\, dx\, dy\, dz = -\frac{\delta(n\, dx\, dy\, dz)}{\delta t} \right) \tag{4}$$

Onde $dx\ dy\ dz$ é o volume infinitesimal de solo.

Esta fórmula descreve como o fluxo de água no solo está relacionado à variação no volume de água armazenado. O termo à esquerda, que é a soma das derivadas parciais das componentes de velocidade, representa a quantidade de água fluindo para dentro ou para fora do elemento de solo nas três direções x, y e z. O termo à direita expressa a variação temporal do volume de água no elemento, que é afetada pela porosidade do solo (BEAR, 1972).

Quando o fluxo é estacionário, ou seja, em condições nas quais não ocorre variação temporal no volume de água armazenado no solo, a equação de continuidade se reduz à Equação 5:

$$\frac{\delta v_x}{\delta x} + \frac{\delta v_y}{\delta y} + \frac{\delta v_z}{\delta z} = 0 \tag{5}$$

Esta equação implica que, no regime estacionário, a água que entra em um volume de solo em um certo intervalo de tempo deve ser igual à água que sai desse volume, garantindo que não haja acúmulo ou depleção de água no elemento de solo. Portanto, a soma das taxas de variação das velocidades nas três direções é zero (BEAR, 1972).

Se o solo for isotrópico, isto é, quando a condutividade hidráulica apresenta o mesmo valor em todas as direções ($kx = ky = kz = k$), a equação de continuidade pode ser simplificada e é representada pela Equação 6:

$$k \left(\frac{\delta^2 h}{\delta x^2} + \frac{\delta^2 h}{\delta y^2} + \frac{\delta^2 h}{\delta z^2} \right) = 0 \tag{6}$$

Ao introduzir o potencial de velocidade, a equação de fluxo pode ser reescrita na forma apresentada pela Equação 7.

$$\frac{\delta^2 \Phi}{\delta x^2} + \frac{\delta^2 \Phi}{\delta y^2} + \frac{\delta^2 \Phi}{\delta z^2} = 0 \tag{7}$$

Esta equação, conhecida como equação de Laplace, descreve o comportamento do fluxo de água em solos isotrópicos, onde a condutividade hidráulica não varia em função da direção. O potencial de velocidade é uma variável que reflete o comportamento do fluxo de água com base na altura do potencial hidráulico h e na condutividade hidráulica k (BEAR, 1972).

Em um meio poroso saturado, o comportamento do fluxo de água é modelado por uma combinação das equações de Darcy, continuidade e as equações de condutividade. Essas equações fornecem uma descrição matemática robusta para o movimento de água em solos, sendo

fundamentais para a análise e projeto de sistemas geotécnicos, como barragens e fundações. O modelo assume que o fluxo é laminar, a densidade da água é constante e a condutividade hidráulica é a principal variável que regula o fluxo (BEAR, 1972; CHAPUIS; AUBERTIN, 2003).

2.2 Erosão interna

O controle da percolação em estruturas geotécnicas, como barragens e fundações permeáveis, é essencial para prevenir problemas críticos como subpressões excessivas, instabilidade do talude de jusante, erosão interna (piping) e migração de partículas através de juntas ou fissuras. A percolação descontrolada pode desencadear processos de degradação progressiva, frequentemente iniciados na zona de descarga, ampliando os caminhos de fluxo e comprometendo a integridade estrutural. Esses processos, se não mitigados, podem evoluir de incidentes menores a falhas catastróficas (FELL, 2005; ICOLD, 2013).

Para mitigar os riscos associados à percolação em estruturas geotécnicas, é fundamental o dimensionamento e a implementação eficaz de dispositivos de vedação e drenagem, como cortinas de injeção, tapetes impermeáveis e trincheiras de alívio, que garantem a estabilidade e segurança da estrutura ao longo de sua vida útil (LEONHARDT, 2003; FELL, 2005; ICOLD, 2013). Segundo SARÉ (2003), o piping geralmente se origina na região de jusante e avança para montante, sendo intensificado por trajetórias preferenciais em interfaces entre materiais com propriedades hidromecânicas distintas.

A erosão interna desenvolve-se de forma discreta e gradual, progredindo silenciosamente até atingir proporções detectáveis, como brechas visíveis ou irregularidades identificadas por monitoramento especializado (FERDOS, 2016). Esse fenômeno ocorre em etapas bem definidas, apresentadas na Tabela 1.

2.2.1 Mecanismos de erosão interna

Segundo ICOLD (2013), a erosão em barragens se inicia por meio de quatro mecanismos principais: vazamentos concentrados, erosão regressiva, erosão de contato e sufusão. Cada um desses mecanismos possui características distintas que influenciam o comportamento e a estabilidade das estruturas. A seguir, detalham-se esses processos e suas particularidades.

Tabela 1 – Etapas principais do processo de erosão interna em barragem

Fase	Assuntos considerados
Carregamento	Carga hidrostática.
Localização do início da erosão	Maciço, fundação ou do maciço para a fundação.
Início	Mecanismo de erosão: erosão laminar, retroerosão tubular regressiva, erosão de contato ou sufusão.
Continuação (Filtração)	Ausência de filtros e zonas preferenciais para percolação. Para erosão laminar e retroerosão tubular regressiva: o tubo continua aberto?
Progressão	Para erosão de contato e sufusão: o gradiente crítico ou a velocidade de percolação atingem valores significativos para a progressão da erosão?
Detecção	Processo de tubulação, tipo de monitoramento e frequência de inspeção.
Intervenção	Equipe qualificada, equipamentos disponíveis, taxa de erosão e impactos ocorridos.
Brecha	Alargamento agravante do tubo, galgamento e instabilidade na estrutura.

adaptado de ICOLD (2013), citado por SOUSA (2021)

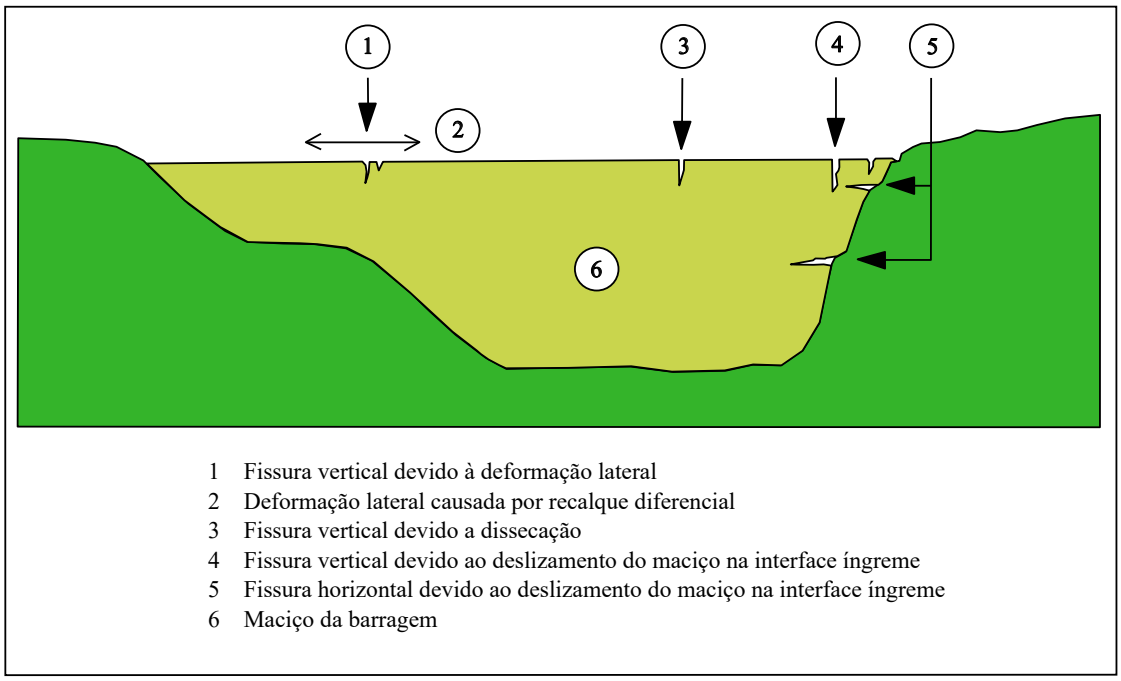
2.2.1.1 Escoamento concentrado

O escoamento concentrado (Figura 3) ocorre quando se formam aberturas em solos plásticos ou, em alguns casos, em siltes não saturados, areia siltosa ou cascalhos silto-arenosos, o que facilita a erosão das laterais dessas aberturas. Essas aberturas podem ter várias origens, como: (i) fissuras causadas por assentamentos diferenciais durante a construção ou operação da barragem, (ii) fraturas hidráulicas resultantes de tensões baixas em torno de conduítes ou na parte superior da barragem, (iii) dessaturação nos níveis superiores do aterro, (iv) ação de geadas que provocam rachaduras na coroa da barragem, (v) assentamentos de colapso em aterros mal compactados, especialmente ao redor de conduítes e paredes, e (vi) interferência de animais escavadores ou raízes de árvores em decomposição. A ocorrência desses mecanismos contribui para o agravamento da erosão e compromete a estabilidade da barragem (ICOLD, 2013).

2.2.1.2 Erosão regressiva

O fenômeno de piping é caracterizado pelo deslocamento progressivo de partículas do solo, causado pelas forças trativas geradas pelo fluxo intergranular de água, conforme detalhado por TERZAGHI (1943) e SHERARD (1963). A erosão interna inicia-se no ponto de saída do fluxo, onde as forças erosivas se intensificam devido à concentração da água, levando ao transporte das partículas do solo ao longo do caminho de percolação, conforme apresentado

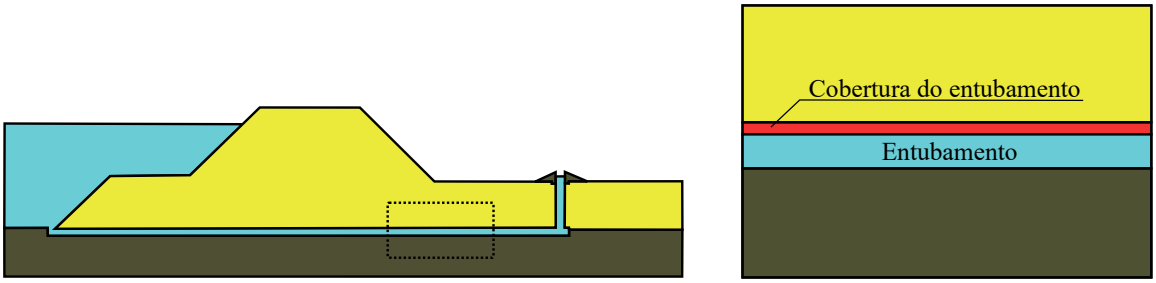
Figura 3 – Tipos de erosões laminares



Fonte: adaptado (ICOLD, 2013)

na Figura 4. De acordo com FELL (2005), em seu manual sobre engenharia geotécnica de barragens, para que a erosão interna e o piping se desenvolvam, quatro condições essenciais devem ser atendidas: (i) a presença de um caminho de fluxo de percolação e uma fonte de água; (ii) a existência de material erodível ao longo do caminho de fluxo, passível de ser transportado pela percolação; (iii) uma saída desprotegida pela qual o material erodido possa escapar; e (iv) a capacidade do material sendo transportado, ou do material situado acima, de formar e sustentar o "teto"do tubo, permitindo a formação de um canal de erosão contínuo. A compreensão dessas condições é crucial para a análise e prevenção do piping em barragens de terra, especialmente em projetos que visam garantir a estabilidade e a segurança das estruturas.

Figura 4 – Erosão regressiva (piping)

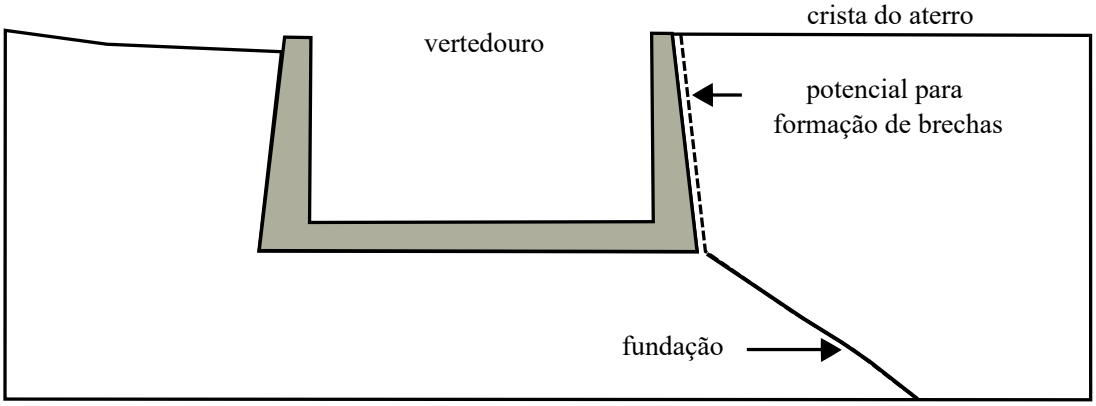


Fonte: adaptado (SOUSA, 2021)

2.2.1.3 Erosão de contato

As interfaces entre materiais, caracterizadas por propriedades hidromecânicas contrastantes, configuram zonas de elevada suscetibilidade à percolação descontrolada, favorecendo o desenvolvimento de erosão por contato, conforme apresentado na Figura 5 (LIN *et al.*, 2003; SELLMEIJER *et al.*, 2011; XIE *et al.*, 2018; HUANG *et al.*, 2021; FLORES-BERRONES *et al.*, 2011). Essas discrepâncias nas propriedades dos materiais, combinadas com a influência de tensões elevadas e gradientes hidráulicos críticos, constituem fatores primordiais para a geração de tensões de cisalhamento significativas. Essas tensões promovem fluxos preferenciais concentrados, que podem comprometer a integridade e a segurança estrutural da barragem (LUO *et al.*, 2013). Estudos estatísticos indicam que falhas decorrentes de erosão interfacial representam 8,46% do total de ocorrências reportadas, conforme o CHINA WATER CONSERVANCY YEARBOOK COMPILATION COMMITTEE (2008). Eventos históricos, como o colapso da Barragem de Teton nos Estados Unidos, evidenciado por SEED e DUNCAN (1987), destacam a gravidade e os riscos associados às falhas de percolação em interfaces solo-estrutura, reforçando a necessidade de medidas preventivas e análises detalhadas durante o projeto e a operação dessas estruturas.

Figura 5 – Erosão de contato



Fonte: adaptado (FELL *et al.*, 2008)

2.2.1.4 Sufusão

A sufusão é um processo associado à migração seletiva de partículas finas em solos granularmente instáveis, especialmente aqueles com distribuição granulométrica ampla ou descontínua (gap-graded) (ICOLD, 2013). Segundo o manual, esse fenômeno ocorre principalmente em solos não plásticos e materiais utilizados em barragens, como filtros ou aterros, quando

apresentam excesso de partículas finas ou características que favorecem a instabilidade interna. Durante o fluxo de percolação, a água exerce forças trativas que deslocam as partículas menores, permitindo que elas atravessem os poros formados pelas partículas mais grosseiras, que permanecem estruturalmente estáveis e suportam as tensões efetivas. Esse transporte de partículas finas pode modificar significativamente as condições hidráulicas do solo, resultando em aumento da permeabilidade e intensificação dos gradientes hidráulicos, com potenciais implicações na estabilidade da estrutura.

Embora em alguns casos o processo alcance um equilíbrio temporário, no qual parte das partículas finas se acomoda e estabiliza, eventos como ciclos de variação de cargas hidráulicas ou flutuações no nível de reservatórios podem reativar o transporte de partículas. Quando a sufusão ocorre no núcleo ou na fundação de uma barragem, o deslocamento progressivo de partículas pode causar recalques localizados, comprometendo a integridade estrutural do maciço. Filtros projetados com materiais internamente instáveis também representam um risco significativo, pois a perda gradual de partículas finas os torna menos eficazes na contenção da erosão interna. Além disso, a segregação de solos durante a construção, particularmente em materiais amplamente graduados, pode criar zonas localmente vulneráveis, mesmo que a graduação média seja considerada estável. Tais condições destacam a importância de uma engenharia criteriosa na escolha e compactação de materiais, bem como na avaliação detalhada das condições hidráulicas e mecânicas dos solos, para evitar a progressão da sufusão e seus impactos adversos (ICOLD, 2013).

2.2.2 Interface concreto/solo

Sempre que dois materiais distintos são justapostos, forma-se uma interface (Figura 6) caracterizada por uma descontinuidade física e mecânica, decorrente das diferenças inerentes de comportamento tensão-deformação, resistência e permeabilidade. Essa interface promove descontinuidades nos campos de tensões e deformações, além de contrastes significativos nos coeficientes de permeabilidade, os quais influenciam diretamente o regime de escoamento na zona de contato. A transferência de tensões entre os materiais adjacentes ocorre em função das discrepâncias de rigidez e resistência mecânica, podendo, em certas situações, ser agravada pela má conformação ou acomodação entre as superfícies, o que propicia a formação de vazios localizados. Tais vazios passam a atuar como condutos preferenciais de percolação, intensificando o fluxo hidráulico na interface e comprometendo, em última instância, a segurança e o

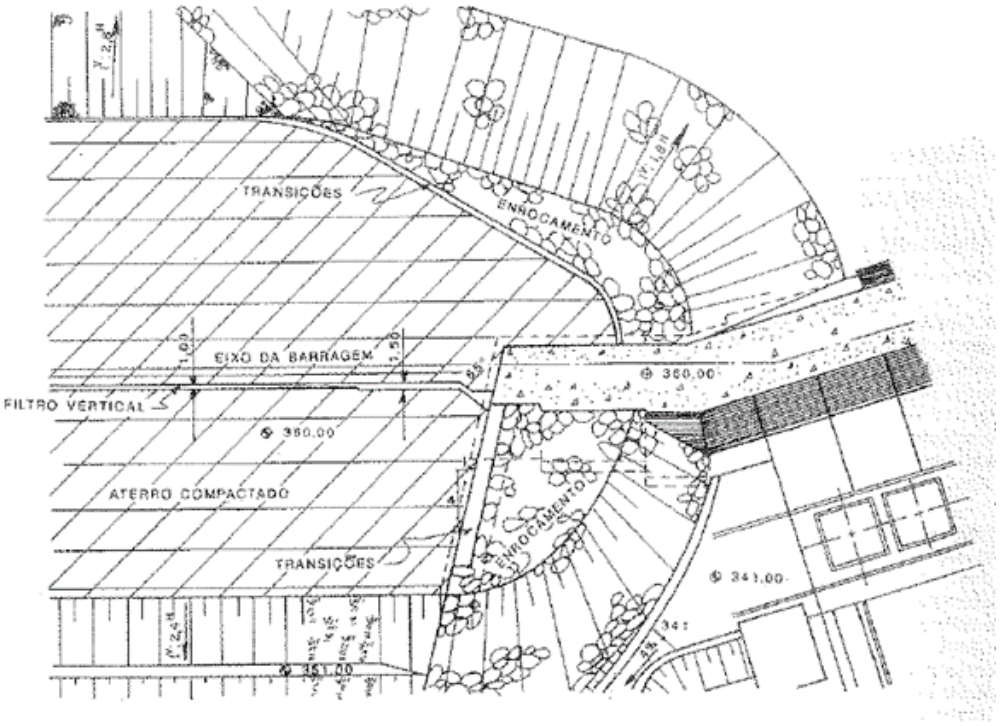
desempenho estrutural da barragem (CRUZ, 1996).

As interfaces estabelecidas entre materiais com características hidromecânicas contrastantes constituem zonas críticas e potencialmente susceptíveis à percolação não controlada, favorecendo a ocorrência de processos erosivos localizados, com destaque para os mecanismos de erosão de contato (LIN *et al.*, 2003; SELLMEIJER *et al.*, 2011; XIE *et al.*, 2018; HUANG *et al.*, 2021; FLORES-BERRONES *et al.*, 2011). As divergências nas propriedades hidráulicas e mecânicas dos materiais adjacentes, aliadas à ação de tensões elevadas e à presença de gradientes hidráulicos críticos, favorecem o desenvolvimento de tensões de cisalhamento na zona de interface, condição que culmina na formação de fluxos concentrados capazes de comprometer significativamente a estabilidade global da estrutura de contenção (LUO *et al.*, 2013). Estimativas estatísticas reportadas pelo CHINA WATER CONSERVANCY YEARBOOK COMPILATION COMMITTEE (2008) indicam que as falhas associadas à erosão interfacial correspondem a aproximadamente 8,46% do total de incidentes catalogados. A relevância desse tipo de falha é evidenciada por acidentes históricos, como a ruptura da Barragem de Teton, nos Estados Unidos, cujas causas foram diretamente relacionadas a processos de percolação na interface solo-estrutura (SEED; DUNCAN, 1987).

O dimensionamento adequado de estruturas localizadas na interface entre materiais distintos exige, de forma imprescindível, a consideração dos esforços induzidos pelo empuxo do solo, do enrocamento e da água, bem como a implementação de medidas eficazes de controle do fluxo, incluindo dispositivos de vedação e sistemas de drenagem. Com o objetivo de mitigar os efeitos adversos decorrentes dos acentuados contrastes nas propriedades hidráulicas e mecânicas dos materiais em contato, recomenda-se a inclusão de uma camada intermediária, usualmente denominada de transição, cuja função é promover a compatibilização entre os meios. Essa camada deve ser criteriosamente projetada para apresentar propriedades intermediárias de permeabilidade, resistência à erosão, deformabilidade e resistência mecânica, assegurando, assim, a continuidade estrutural e hidráulica do sistema e contribuindo para a estabilidade e segurança da obra de engenharia (CRUZ, 1996).

O encontro entre a barragem de terra e a estrutura de concreto associada pode ser classificado, predominantemente, em dois arranjos distintos: o tipo frontal, no qual o maciço entra em contato direto com o paramento rígido do vertedouro ou da tomada dágua, e o tipo abraço, no qual a barragem envolve um muro especialmente projetado para viabilizar a conexão entre os materiais inconsolidados do maciço e as estruturas em concreto. Neste segundo caso, as

Figura 6 – Barragem de Água Vermelha - planta da estrutura de contato entre as barragens de terra e enrocamento e de concreto



Fonte: (CRUZ, 1996)

estruturas de concreto permanecem isoladas da massa de terra, assegurando uma interface controlada e bem definida. Importa destacar que, quando adotado o arranjo em abraço, o respectivo muro encontra-se submetido a empuxos atuantes em ambas as faces montante e jusante , o que implica desafios adicionais no dimensionamento estrutural. A consideração do empuxo de jusante, entretanto, depende de hipóteses relacionadas ao comportamento deformacional da estrutura, as quais, por vezes, não se encontram claramente especificadas e, em alguns contextos, são tratadas de forma inadequada, comprometendo a precisão dos modelos de análise (CRUZ, 1996).

Nesse contexto, uma solução amplamente consolidada e empregada no controle de gradientes hidráulicos em barragens de terra consiste na incorporação de dispositivos denominados abraços, que se configuram como extensões projetadas a partir das estruturas de concreto, com a finalidade de ampliar o trajeto de percolação ao longo da interface solo-estrutura. A principal função desses dispositivos é promover o aumento do percurso hidráulico, favorecendo a dissipação progressiva da energia do fluxo subterrâneo, o que contribui para a redução significativa do gradiente hidráulico e, consequentemente, para a mitigação dos riscos associados à erosão interna e ao desenvolvimento do fenômeno de piping (CRUZ, 1996).

Sob a ótica hidrodinâmica, o abraço altera substancialmente a configuração do problema de percolação, promovendo sua transição de uma condição bidimensional para um comportamento tridimensional. Nessa nova condição, o fluxo passa a ocorrer por diferentes planos e direções, ampliando a zona de contato entre o solo e o concreto e dificultando o avanço do fluxo preferencial em direção ao talude de jusante. Essa modificação geométrica e funcional contribui, também, para a atenuação de pressões negativas que poderiam comprometer a estabilidade do contato entre os materiais, especialmente em fundações permeáveis submetidas a solicitações variáveis.

O desempenho efetivo desse tipo de dispositivo está intrinsecamente condicionado à qualidade da interação solo-estrutura, à resistência ao deslizamento ao longo da interface e ao comportamento do meio em condições de fluxo não saturado aspectos estes que apresentam elevada complexidade física e numérica. Não obstante sua relevância, tais interações são, com frequência, negligenciadas nas etapas de projeto e dimensionamento, o que compromete a funcionalidade e a segurança global da estrutura, sobretudo em cenários caracterizados por eventos extremos, como chuvas intensas, enchentes súbitas ou flutuações abruptas do nível do reservatório.

Dessa forma, o dimensionamento das estruturas de contato tipo abraço deve considerar, de forma criteriosa e integrada: (i) a estimativa dos empuxos resultantes da ação combinada do solo, do enrocamento e da lâmina dágua; (ii) o controle do fluxo de percolação, tanto por meio de dispositivos de vedação quanto de sistemas de drenagem eficazes; e (iii) a avaliação dos deslocamentos potenciais da estrutura, bem como suas implicações nos esforços atuantes e na efetividade das soluções adotadas para o controle hidráulico e a estabilidade da barragem (CRUZ, 1996).

2.2.2.1 Tratamento da interface na Barragem Castanhão

A Barragem Castanhão, localizada no rio Jaguaribe, no estado do Ceará, configura-se como a maior obra hídrica do Nordeste brasileiro, assumindo papel estratégico no abastecimento da Região Metropolitana de Fortaleza, bem como no controle de cheias e na regularização do regime hidrológico da bacia. Sua concepção estrutural adota uma configuração mista, composta por um trecho central em concreto compactado a rolo (CCR) e ombreiras laterais constituídas por barragens zonadas de terra com núcleo argiloso. O segmento em CCR compreende o intervalo entre as estacas 5+000 e 36+10, totalizando aproximadamente 310 metros de

extensão, e corresponde ao dispositivo de conexão entre o maciço de terra e as estruturas rígidas, constituindo o dispositivo abraço, conforme apresentado na Figura 7 (CEARÁ (Estado), 1999a; CEARÁ (Estado), 1999c).

Nas regiões de transição entre o CCR e o maciço de terra, o paramento de montante do bloco de concreto apresenta inclinação de 1:10 (H:V), especialmente nas estacas de extremidade (5+000 a 9+7,34 e 36+10 a 36+7,94). Essa geometria foi adotada com o intuito de favorecer a compactação adequada do solo contra o concreto, assegurando uma interface com desempenho mecânico e hidráulico satisfatório. O núcleo argiloso, executado com material proveniente da jazida Facetra, foi projetado de modo a envolver parcialmente o corpo em CCR, promovendo uma transição gradual entre materiais de rigidez diferenciada, o que contribui para o controle de deslocamentos relativos e para a integridade da interface ao longo do tempo(CEARÁ (Estado), 1999a; CEARÁ (Estado), 1999c).

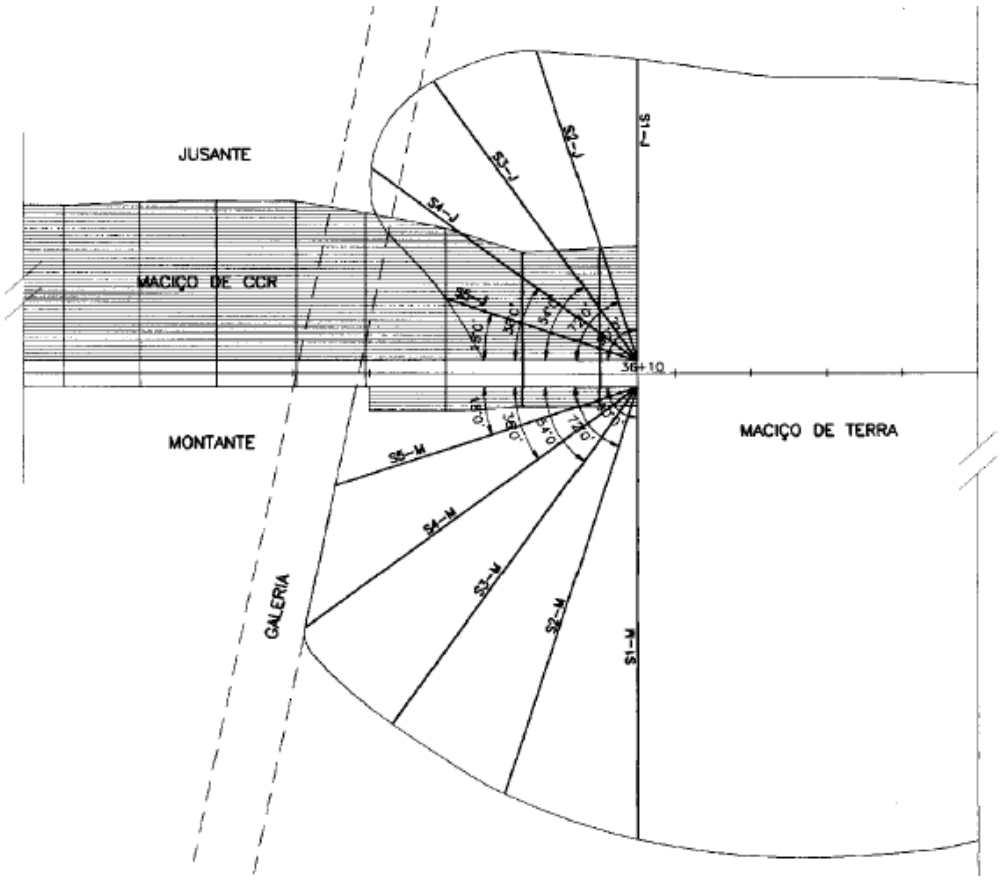
Com vistas à otimização do desempenho hidráulico da junção solo-concreto, o projeto incorpora filtros verticais e horizontais de areia limpa, camadas de enrocamento estrategicamente posicionadas, juntas de construção seladas com perfis de PVC do tipo Fugenband, além de uma faixa de concreto convencional de 0,50 m de espessura executada na face de montante. Esses elementos desempenham papel fundamental na redução da percolação ao longo da interface, na dissipação de pressões intersticiais e na contenção de eventuais trajetórias preferenciais de fluxo (CEARÁ (Estado), 1999a; CEARÁ (Estado), 1999c).

Adicionalmente, o arranjo estrutural da barragem contempla um conjunto de estruturas hidráulicas complementares, tais como vertedouro, tomada dágua e descarregador de fundo, rigidamente integradas ao corpo da barragem. Essa configuração assegura a continuidade construtiva e a estanqueidade global do sistema, ao mesmo tempo em que contribui para a estabilidade das interfaces críticas, mesmo sob condições extremas de carregamento hidrodinâmico ou solicitantes induzidas por variações no nível do reservatório (CEARÁ (Estado), 1999a; CEARÁ (Estado), 1999c).

2.3 Aprendizado de máquina

Nas últimas três décadas, métodos de modelagem baseados em inteligência artificial (IA) têm se destacado na engenharia geotécnica como uma alternativa eficiente e inovadora às abordagens convencionais. Sua principal vantagem reside na capacidade de modelar processos complexos e não lineares sem a necessidade de pressuposições prévias sobre as relações entre

Figura 7 – Barragem Castanhão - dispositivo abraço no contato entre a barragen de terra e de concreto



Fonte: (CEARÁ (Estado), 1999b)

variáveis de entrada e saída. Aplicações dessas técnicas abrangem predição de comportamentos, monitoramento estrutural, seleção de parâmetros críticos e identificação de fenômenos complexos, demonstrando elevada precisão, mesmo em contextos marcados por incertezas ou ausência de modelos físicos detalhados. Tal versatilidade posiciona a IA como uma ferramenta indispensável para solucionar desafios geotécnicos modernos (BELLO *et al.*, 2015; JANG; TOPAL, 2013; SARKER, 2021).

Uma das características mais notáveis das técnicas de aprendizado de máquina é sua capacidade de processar e analisar grandes volumes de dados, identificando padrões relevantes mesmo quando as relações subjacentes entre variáveis são desconhecidas ou não intuitivas. Essa habilidade é particularmente valiosa na engenharia geotécnica, onde os dados disponíveis frequentemente apresentam limitações, como escassez, incompletude ou presença de ruído. Adicionalmente, essas ferramentas podem preencher lacunas em conjuntos de dados por meio de estimativas confiáveis, aumentando a qualidade e a utilidade dos dados disponíveis. As técnicas de aprendizado de máquina são geralmente classificadas em aprendizado supervisionado,

onde os modelos são treinados com dados rotulados, e aprendizado não supervisionado, que identifica padrões em dados não rotulados (GOH, 1995; KAVZOGLU; COLKESEN, 2009).

Pesquisas recentes têm demonstrado a superioridade dos modelos de inteligência artificial na previsão da percolação e da poropressão em barragens, superando as abordagens estatísticas e numéricas tradicionais. LEI *et al.* (2023) evidenciaram que o modelo IGBO-ELM, ao integrar técnicas avançadas de otimização, aprimora significativamente a precisão e a capacidade de generalização na previsão da percolação em barragens de concreto. Da mesma forma, BILALI *et al.* (2022) mostraram que modelos de aprendizado de máquina superam as abordagens estatísticas na previsão da poropressão diária em barragens de aterro, permitindo uma detecção mais eficaz de anomalias e um entendimento mais profundo do comportamento hidráulico dessas estruturas.

2.3.1 Aprendizado supervisionado

No aprendizado supervisionado, o objetivo central é desenvolver modelos que realizem o mapeamento das variáveis de entrada (x) para as variáveis de saída (y), com base em um conjunto de dados rotulado ($\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$), onde cada par de entrada e saída é conhecido. Esses pares podem abranger desde características simples, como altura e peso, até representações mais complexas, como imagens ou grafos. A variável de saída (y) pode assumir diferentes formas, sendo categórica, como nos problemas de classificação, ou contínua, como nos problemas de regressão (HASTIE, 2009; BISHOP, 2006).

2.3.1.1 Classificação

A classificação é uma tarefa fundamental no aprendizado supervisionado, onde o objetivo principal é estimar uma função $f(x)$ que relacione as variáveis de entrada (x) a uma variável de saída discreta (y), utilizando um conjunto de dados rotulados para treinar o modelo. Essa função deve generalizar bem, ou seja, ser capaz de fazer previsões precisas para dados novos, além do conjunto de treinamento. A tarefa de classificação pode ser categorizada em diferentes tipos, dependendo da natureza das saídas esperadas (HASTIE, 2009; BISHOP, 2006; MITCHELL, 1997).

I. **Classificação binária**: neste caso, o número de classes (C) é igual a 2, e a variável de saída (y) pode assumir apenas dois valores distintos, como “sim” ou “não”, “0” ou “1”.

Exemplos típicos incluem problemas de detecção de spam ou diagnóstico médico (presença ou ausência de uma doença).

II. **Classificação multiclasse**: quando $C > 2$, a saída y pode pertencer a uma entre várias classes mutuamente exclusivas. Por exemplo, em um problema de classificação de imagens de animais, o modelo pode ser treinado para identificar entre três classes distintas: "gato", "cachorro" e "coelho". Se a entrada for uma imagem de um cachorro, a saída y será a classe "cachorro". Aqui, a relação de exclusão mútua garante que a imagem não pode ser simultaneamente "gato" e "cachorro".

III. **Classificação multilabel**: aqui, um único exemplo pode pertencer simultaneamente a múltiplas classes, tornando-se útil em contextos como rotulação de imagens contendo múltiplos objetos. Por exemplo, em um sistema de classificação de imagens, uma única foto pode conter tanto um "gato" quanto um "cachorro". Nesse caso, o modelo atribui as duas classes à mesma imagem, ou seja, a saída y seria um vetor com múltiplas etiquetas, como $y = [\text{gato}, \text{cachorro}]$.

Uma característica essencial da classificação é a necessidade de prever probabilidades associadas a cada possível classe, especialmente em situações onde os dados são ambíguos. A probabilidade condicional $p(y \mid x, D)$, que descreve a probabilidade de y dado x e o conjunto de treinamento D, é fundamental no processo de inferência. A partir dessa distribuição, o melhor palpite para o rótulo verdadeiro é obtido por meio da maximização da probabilidade, ou seja, identificando o valor de y que maximiza a função $p(y \mid x, D)$. Esse procedimento visa encontrar a estimativa mais provável de y, dado o valor observado de x e os dados fornecidos no treinamento, fundamentando-se nos princípios da máxima verossimilhança (BISHOP, 2006).

2.3.1.2 Regressão

A regressão é uma tarefa de aprendizado supervisionado que se assemelha à classificação, mas se diferencia pelo fato de a variável de resposta ser contínua. Essa abordagem busca modelar a relação entre as variáveis de entrada ($x_i \in \mathbb{R}$) e a variável de saída ($y_i \in \mathbb{R}$) por meio de funções que melhor descrevam os padrões presentes nos dados (HASTIE, 2009).

Além disso, problemas de maior complexidade frequentemente surgem em aplicações práticas, como no caso de entradas com alta dimensionalidade, presença de *outliers* que podem comprometer a acurácia do ajuste do modelo, ou quando a relação entre as variáveis de entrada

e saída é não linear e apresenta descontinuidade. Essas dificuldades exigem abordagens avançadas para garantir a eficácia e a precisão dos modelos, uma vez que tais fatores podem afetar significativamente o desempenho e a generalização dos algoritmos de aprendizado (HASTIE, 2009).

2.3.2 Aprendizado não supervisionado

O aprendizado não supervisionado é uma abordagem poderosa dentro do aprendizado de máquina, onde se trabalha com dados não rotulados, ou seja, não há associação explícita entre entradas e saídas. O objetivo central é identificar padrões, estruturas ou regularidades ocultas nos dados. Diferentemente do aprendizado supervisionado, onde o foco está em prever uma saída desejada com base em entradas conhecidas, o aprendizado não supervisionado busca modelar a distribuição incondicional dos dados $p(x_i \mid \theta)$, em vez de uma densidade condicional $p(y_i \mid x_i, \theta)$ (HASTIE, 2009).

Uma das principais diferenças em relação ao aprendizado supervisionado é que no aprendizado não supervisionado as variáveis de interesse (x_i) são geralmente vetores multivariados de características, exigindo o uso de modelos probabilísticos multivariados para capturar suas relações. Por outro lado, no aprendizado supervisionado, a variável de saída (y_i) frequentemente é univariada, o que simplifica os modelos a serem empregados. Esse contraste ressalta a complexidade inerente ao aprendizado não supervisionado, especialmente em cenários de alta dimensionalidade, onde identificar estruturas relevantes nos dados é mais desafiador (HASTIE, 2009).

Conforme discutido por HASTIE (2009), alguns exemplos clássicos de aprendizado não supervisionado incluem aplicações como o agrupamento (clustering), que visa identificar grupos ou categorias naturais nos dados, como segmentar clientes com base em padrões de comportamento ou detectar comunidades em redes sociais; a redução de dimensionalidade, que busca resumir os dados mantendo suas características essenciais, como no uso da Análise de Componentes Principais (PCA) para compressão de informações ou visualização em espaços de menor dimensão; a modelagem de mistura gaussiana, utilizada para aproximar distribuições complexas dos dados com uma combinação de distribuições normais; e os modelos de tópicos, empregados para explorar estruturas latentes em dados textuais, como identificar tópicos principais em coleções de documentos.

Essa abordagem é amplamente aplicável, especialmente em cenários onde os dados ro-

tulados são escassos ou caros de obter, permitindo que insights valiosos sejam extraídos diretamente da riqueza das informações contidas nos próprios dados.

2.3.3 Subajuste (*underfitting*) e sobreajuste (*overfitting*)

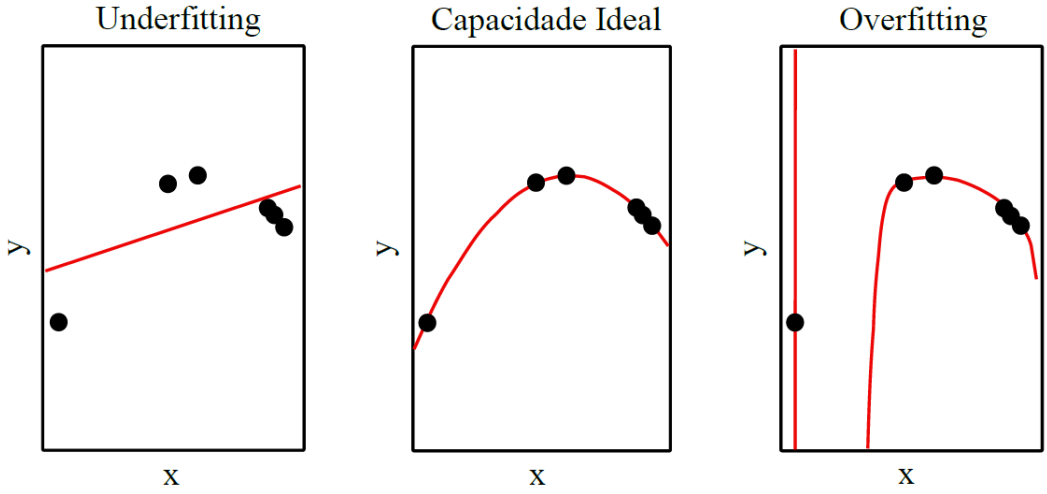
No contexto do aprendizado de máquina, o objetivo fundamental é desenvolver um modelo capaz de obter desempenho satisfatório em dados novos e previamente desconhecidos, extrapolando os limites do conjunto de dados utilizado durante o treinamento. Essa capacidade, denominada generalização, é essencial para assegurar que o modelo seja efetivamente aplicável a problemas do mundo real. A generalização reflete a habilidade do modelo em realizar previsões precisas sobre entradas inéditas, evidenciando que o aprendizado vai além da simples memorização dos padrões observados no conjunto de treinamento, alcançando uma compreensão mais profunda e flexível das relações subjacentes aos dados (BISHOP, 2006; HASTIE, 2009).

O treinamento de um modelo em aprendizado de máquina busca minimizar o erro de treinamento, caracterizando um problema de otimização. Contudo, o principal desafio reside em garantir um erro reduzido em novos dados, o chamado erro de generalização. Segundo GOODFELLOW *et al.* (2016), tanto os dados de treinamento quanto os de teste são amostras de uma mesma distribuição probabilística subjacente, fundamentada nas hipóteses i.i.d. (independentes e identicamente distribuídas). Esse arcabouço matemático conecta os erros de treinamento e teste, fornecendo uma base para analisar a capacidade de generalização dos modelos.

No aprendizado de máquina, os parâmetros do modelo não são fixos; eles são ajustados iterativamente com base nos dados disponíveis, visando minimizar uma função de perda predefinida. Entretanto, o erro esperado no conjunto de teste tende a ser maior, ou, no melhor cenário, equivalente ao erro observado no conjunto de treinamento. Esse comportamento está diretamente relacionado a dois problemas clássicos do aprendizado de máquina: *underfitting* e *overfitting*, apresentado na Figura 8 (HASTIE, 2009; GOODFELLOW *et al.*, 2016).

O *underfitting* ocorre quando o modelo não é capaz de capturar os padrões fundamentais nos dados de treinamento, resultando em baixa capacidade preditiva e erros elevados tanto no treinamento quanto na validação. Esse problema geralmente decorre de modelos excessivamente simples ou de insuficiência de dados representativos. Por outro lado, o *overfitting* ocorre quando o modelo se ajusta excessivamente aos dados de treinamento, aprendendo não apenas os padrões gerais, mas também os ruídos e particularidades específicas irrelevantes. Como conse-

Figura 8 – Subajuste (underfitting) e superajuste (overfitting)



Fonte: adaptado (GOODFELLOW *et al.*, 2016)

quência, o modelo apresenta um erro de treinamento reduzido, mas um desempenho significativamente pior em dados não vistos, comprometendo sua capacidade de generalização (HASTIE, 2009; GOODFELLOW *et al.*, 2016).

2.3.4 Normalização dos dados

A normalização dos dados constitui uma etapa fundamental na preparação para o treinamento de modelos de aprendizado de máquina, assegurando que todas as variáveis independentes sejam escaladas de maneira uniforme, eliminando desequilíbrios que possam comprometer o desempenho do modelo (IPNET, 2023).

A normalização consiste no ajuste dos valores das variáveis para intervalos predefinidos, como $[0,1]$ ou $[-1,1]$ no caso de variáveis que incluem valores negativos, preservando as proporções relativas dentro de cada faixa. Embora este processo não elimine completamente o impacto de *outliers* (valores extremos), ele desempenha um papel crucial na padronização das escalas. Essa uniformização reduz significativamente o risco de que variáveis com magnitudes desproporcionais dominem ou comprometam o processo de aprendizado do modelo, contribuindo para maior estabilidade e eficiência nos cálculos (IPNET, 2023).

As técnicas de normalização podem ser classificadas em:

I. **Min-Max Scaling**: O redimensionamento pelo método Min-Max ajusta os valores das variáveis para um intervalo definido, geralmente entre $[0,1]$, utilizando a fórmula:

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}} \tag{8}$$

Onde X é o valor original da variável, X_{min} é o menor valor observado da variável no conjunto de dados, X_{max} é o maior valor observado da variável no conjunto de dados e X_{norm} é o valor normalizado da variável.

Esse método garante que todos os valores sejam redimensionados proporcionalmente, mantendo a distribuição relativa dos dados dentro do intervalo especificado, sem alterar a relação entre os valores originais.

II. **Z-score Normalization**: A normalização pelo método Z-score centraliza os dados em torno de sua média e ajusta-os em função de sua variabilidade, conforme a equação:

$$X_{\text{norm}} = \frac{X - \mu}{\sigma} \tag{9}$$

Onde μ é a média da variável no conjunto de dados, σ é o desvio padrão da variável e X_{norm} é o valor normalizado da variável.

Esse método transforma os valores em unidades de desvio padrão a partir da média, resultando em uma distribuição com média zero e variância unitária, preservando a estrutura relativa dos dados enquanto reduz a influência de magnitudes desproporcionais.

2.3.5 Métricas e gráficos de desempenho

Nos modelos de regressão fundamentados em técnicas de aprendizado de máquina, as métricas de desempenho têm um papel crucial na avaliação da eficácia do modelo, sendo essenciais para a quantificação do desvio entre as previsões geradas e as observações reais. Essas métricas são instrumentos fundamentais para analisar a capacidade do modelo em generalizar e fornecer resultados preditivos de alta precisão, refletindo a qualidade do ajuste realizado. A utilização de métricas de desempenho para tal análise é amplamente documentada na literatura científica, com uma longa trajetória histórica que sustenta seu uso em diferentes contextos de modelagem preditiva (PLEVRIS *et al.*, 2022).

Entre as diversas métricas amplamente utilizadas para avaliar o desempenho de modelos de regressão, destacam-se o erro quadrático médio (EQM, ou MSE Mean Squared Error), sua raiz (REQM, ou RMSE Root Mean Squared Error), o erro absoluto médio (EAM, ou MAE Mean Absolute Error), o coeficiente de correlação de Pearson (R) e o coeficiente de determinação (Rš). Essas métricas são amplamente reconhecidas por sua capacidade de capturar tanto a magnitude dos erros quanto a relação entre as variáveis preditoras e as observadas, fornecendo uma visão robusta da acurácia do modelo. Conforme enfatizado por PLEVRIS *et al.* (2022), a

escolha adequada dessas métricas é essencial para garantir uma avaliação precisa, permitindo a otimização dos modelos de regressão e a comparação entre diferentes abordagens preditivas.

2.3.5.1 Erro Absoluto Médio (EAM)

O Erro Absoluto Médio (EAM) é uma métrica estatística utilizada para avaliar a precisão de modelos preditivos. Seu objetivo principal é mensurar a magnitude média dos erros cometidos pelo modelo de previsão, independentemente da direção desses erros (se positivos ou negativos). Essa abordagem é particularmente útil em situações onde o foco está na amplitude do erro, sem a necessidade de distinguir entre subestimações e superestimações. Ao utilizar valores absolutos, o EAM oferece uma maneira clara e simples de analisar a acurácia do modelo, sem ser influenciado por erros que possam se cancelar entre si (MONTGOMERY *et al.*, 2012).

Matematicamente, o EAM é expresso pela Equação 10:

$$\text{EAM} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{10}$$

onde:

n é o número total de observações;

y_i é o valor real observado;

$\hat{y}_i$ é o valor previsto pelo modelo;

$|y_i - \hat{y}_i|$ representa o erro absoluto entre a observação e a previsão.

2.3.5.2 Raiz do Erro Quadrático Médio (EQM)

A Raiz do Erro Quadrático Médio (EQM) é uma das métricas mais comuns na avaliação de modelos preditivos, especialmente em problemas de regressão. Ela reflete a magnitude das diferenças entre os valores preditos pelo modelo e os valores reais observados, fornecendo uma medida agregada da precisão do modelo (MONTGOMERY *et al.*, 2012).

O RMSE é particularmente útil porque penaliza de maneira mais severa grandes desvios entre os valores previstos e observados. Isso ocorre porque os erros são elevados ao quadrado antes de serem somados, o que aumenta o impacto de *outliers* ou grandes erros em relação aos erros menores. Como resultado, o RMSE tende a ser mais sensível a valores discrepantes do que outras métricas, como o Erro Absoluto Médio (EAM)(MONTGOMERY *et al.*, 2012).

A formulação matemática do EQM é expressa pela Equação 11:

$$\text{EQM} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(p_i - r_i)^2} \tag{11}$$

Onde N é o número total de observações; p_i é o valor previsto pelo modelo; r_i é o valor real observado; e $(p_i - r_i)^2$ representa o erro ao quadrado associado a cada observação.

A extração da raiz quadrada no final garante que a métrica tenha as mesmas unidades dos dados originais, facilitando sua interpretação e comparação.

2.3.5.3 Coeficiente de Correlação de Pearson (R)

O Coeficiente de Correlação de Pearson (R) é uma métrica estatística que quantifica a força e a direção da relação linear entre duas variáveis. O valor do coeficiente de correlação varia entre -1 e 1 (MONTGOMERY *et al.*, 2012).

- $R = 1$ indica uma correlação linear positiva perfeita;
- $R = -1$ indica uma correlação linear negativa perfeita;
- $R = 0$ indica ausência de correlação linear entre as variáveis.

A formulação matemática do Coeficiente de Correlação de Pearson é dada pela Equação 12:

$$R = \frac{\sum_{i=1}^{n}(p_i - \bar{p})(r_i - \bar{r})}{\sqrt{\sum_{i=1}^{n}(p_i - \bar{p})^2} \cdot \sqrt{\sum_{i=1}^{n}(r_i - \bar{r})^2}} \tag{12}$$

Nessa expressão, p_i e r_i são os valores preditos e observados, respectivamente, para o i-ésimo item da amostra; $\bar{p}$ e $\bar{r}$ são as médias dos valores preditos e observados; e n é o número total de observações.

Essa fórmula calcula a soma dos produtos das diferenças entre cada valor e suas respectivas médias, normalizada pela raiz quadrada do produto das somas dos quadrados dessas diferenças. Esse cálculo reflete como as variações nas duas variáveis estão associadas linearmente (MONTGOMERY *et al.*, 2012).

2.3.5.4 Coeficiente de Determinação (R^2)

O Coeficiente de Determinação (R^2) é uma métrica utilizada para avaliar a qualidade do ajuste de um modelo preditivo aos dados observados. Ele mede a proporção da variância total dos valores reais que é explicada pelas previsões do modelo (MONTGOMERY *et al.*, 2012).

O valor de R^2 varia entre 0 e 1, sendo que:

- $R^2 = 1$ indica que o modelo explica perfeitamente a variabilidade dos dados;
- $R^2 = 0$ indica que o modelo não explica nenhuma variabilidade.

A formulação matemática do coeficiente de determinação é dada pela Equação 13:

$$R^2 = (R)^2 \tag{13}$$

Onde R é o Coeficiente de Correlação de Pearson entre os valores preditos e observados. O R^2 é simplesmente o quadrado desse coeficiente, refletindo a fração da variabilidade nos dados observados que é explicada pelo modelo.

Em outras palavras, R^2 é a razão entre a soma dos quadrados explicada pelo modelo e a soma total dos quadrados em relação à média dos dados observados. Um valor elevado de R^2 indica que o modelo consegue capturar uma parte significativa da variabilidade dos dados, enquanto um valor baixo sugere que o modelo é ineficaz na explicação dessa variabilidade (MONTGOMERY *et al.*, 2012).

2.3.5.5 Curvas de aprendizado

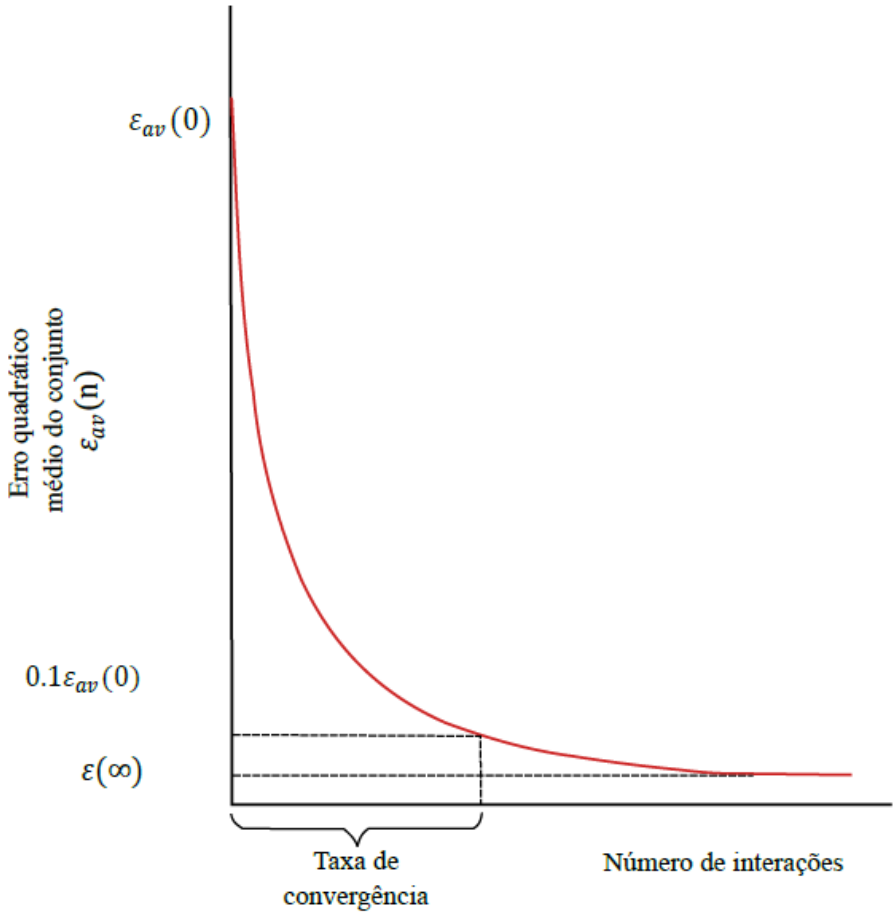
A curva de aprendizado (Figura 9) é uma ferramenta fundamental na análise de modelos de aprendizado de máquina, especialmente no que se refere ao monitoramento da evolução do erro durante o processo de treinamento. Conforme descrito por HAYKIN (2001), essa curva é gerada ao se plotar o erro quadrático médio de estimativa $\xi(n)$. Ela fornece uma visão clara sobre como o erro do modelo muda à medida que o número de iterações aumenta, permitindo a análise da taxa de convergência do algoritmo.

A curva de aprendizado auxilia na compreensão de se o modelo está melhorando de forma contínua ou se encontra estagnado, além de fornecer insights sobre o comportamento do erro de treinamento e validação. Ao analisar essa curva, é possível identificar características importantes do modelo, como a taxa de aprendizado e a capacidade de generalização (HAYKIN, 2001).

Se o erro de treinamento diminui enquanto o erro de validação permanece elevado, isso pode indicar um problema de *overfitting*, no qual o modelo está excessivamente ajustado aos dados de treinamento. Por outro lado, se ambos os erros permanecem elevados, o modelo pode estar sofrendo de *underfitting*, ou seja, não está aprendendo suficientemente bem os padrões contidos nos dados (HAYKIN, 2001).

A análise da curva de aprendizado também é essencial para a seleção de hiperparâmetros, permitindo a escolha dos parâmetros que maximizam a capacidade de generalização do modelo, garantindo um desempenho satisfatório em dados não vistos durante o treinamento (HAYKIN, 2001).

Figura 9 – Curva de aprendizado: evolução do erro quadrático médio ao longo das iterações



Fonte: adaptado (HAYKIN, 2001)

2.3.5.6 Gráfico de resíduos

A normalidade dos resíduos (Figura 10) constitui um pressuposto crucial na análise de Regressão Linear Simples (RLS) e Regressão Linear Múltipla (RLM), quando aplicadas por meio do Método dos Mínimos Quadrados Ordinários (MMQO). Este pressuposto é essencial

não apenas para a validade do modelo de regressão em si, mas também para a confiabilidade dos testes formais que o acompanham, complementando a análise visual realizada por gráficos de dispersão.

Sua importância reside no fato de que, caso os resíduos não apresentem uma distribuição normal, a confiabilidade dos intervalos de confiança e das estimativas dos coeficientes de regressão é comprometida, tornando-os imprecisos e potencialmente ineficazes (JUNIOR; PRETTI, 2025).

2.3.6 Validação cruzada

A validação cruzada é uma técnica utilizada para avaliar a capacidade de generalização de modelos de aprendizado de máquina em dados não vistos. Diferentemente da simples divisão entre conjuntos de treinamento e teste, essa abordagem promove uma análise mais abrangente do desempenho do modelo. Por meio da segmentação do conjunto de dados original em múltiplos subconjuntos, a validação cruzada assegura que o modelo seja testado em diferentes cenários, reduzindo o risco de vieses provenientes de divisões específicas (KOHAVI, 1995).

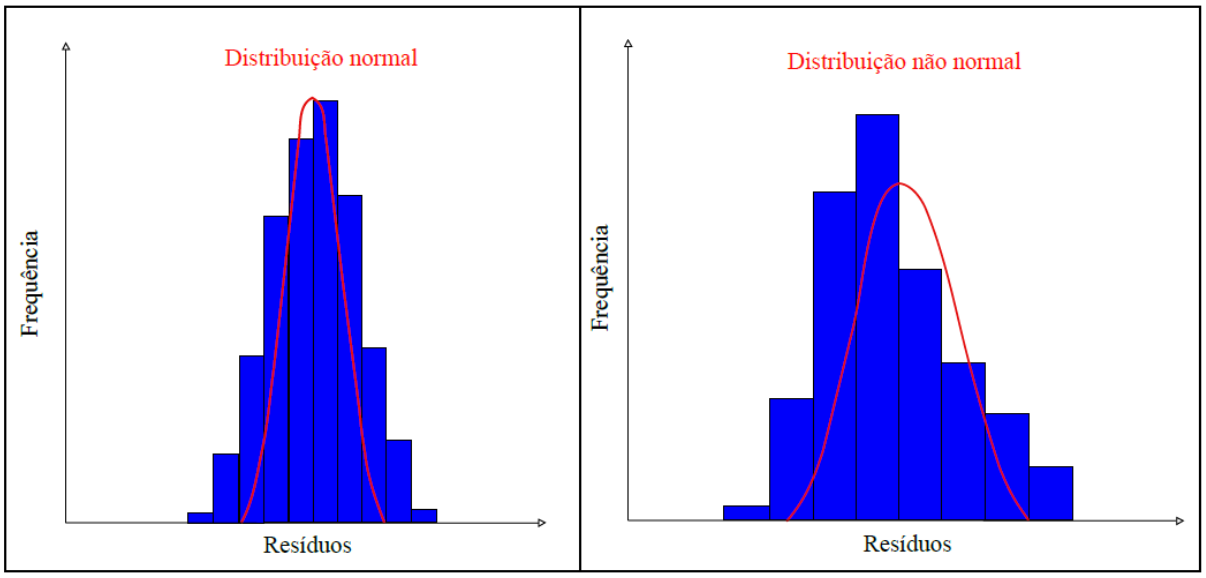
A validação cruzada começa com a divisão do conjunto de dados, onde o conjunto original é particionado em k subconjuntos, conhecidos como *folds*, geralmente de tamanho semelhante (Figura 11). Cada *fold* representa uma fração distinta dos dados, garantindo que todas as amostras sejam utilizadas tanto no treinamento quanto na validação em diferentes iterações. Esse processo assegura que nenhuma amostra seja excluída da análise, promovendo uma avaliação mais abrangente do modelo (KOHAVI, 1995).

Na etapa de treinamento e teste iterativo, o modelo é treinado em $k-1$ *folds* e avaliado no *fold* restante, que serve como conjunto de teste. Esse procedimento é repetido k vezes, permitindo que cada *fold* seja utilizado exatamente uma vez para validação. Essa abordagem iterativa reduz a influência de particionamentos específicos nos resultados, aumentando a confiabilidade da avaliação (KOHAVI, 1995).

2.4 Redes Neurais Artificiais (RNAs)

Um neurônio artificial é uma estrutura matemática que simplifica o comportamento funcional de um neurônio biológico, conforme proposto por MCCULLOCH e PITTS (1943) e posteriormente aprofundado por DEMUTH *et al.* (2014). Essa abstração foi desenvolvida para emular os mecanismos de processamento de informações do sistema nervoso, permitindo que re-

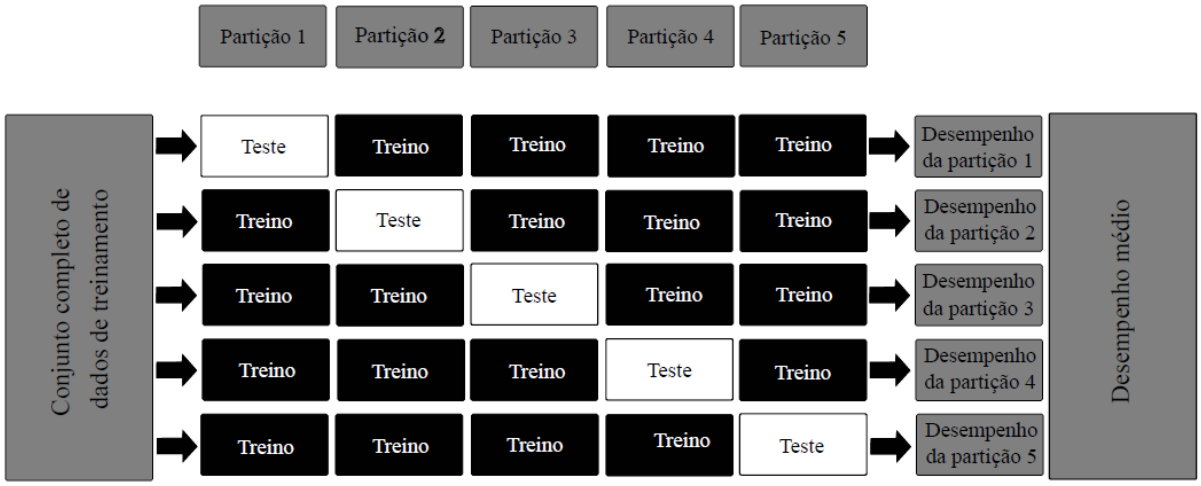
Figura 10 – Verificação da normalidade dos resíduos da regressão



Fonte: adaptado (JUNIOR; PRETTI, 2025)

des computacionais analisem e interpretem dados complexos. O modelo baseia-se na interação de múltiplos neurônios artificiais organizados em camadas, formando a base das Redes Neurais Artificiais (RNAs). Essas redes têm se destacado como ferramentas poderosas para identificar padrões e explorar relações não lineares em dados multidimensionais, eliminando a necessidade de compreensão explícita dos fenômenos físicos subjacentes. Tal eficiência deriva de suas propriedades intrínsecas, como a capacidade de modelar não linearidades, adaptabilidade a diferentes cenários, tolerância a ruídos e falhas, e a flexibilidade para aprendizado supervisionado e não supervisionado (JAIN *et al.*, 1996; MAIER; DANDY, 2000; DAWSON; WILBY, 2001; XU; LI, 2002).

Figura 11 – Avaliação do desempenho por validação cruzada



Fonte: Adaptado de IACOMCAFÉ (2024)

Pesquisas recentes demonstram a eficácia das redes neurais artificiais na previsão de fenômenos geotécnicos, superando abordagens tradicionais. POSO e JESUS (2022) mostraram que sua integração com otimização por enxame de partículas aprimora a previsão da estabilidade de taludes em barragens de terra homogêneas, fornecendo uma ferramenta valiosa para projetistas. BEIRANVAND e RAJAEE (2022) destacaram que esses modelos apresentam uma considerável precisão e aplicabilidade na predição da percolação e da poropressão em barragens. Além disso, SILVA *et al.* (2021) demonstraram que um modelo de rede neural artificial desenvolvido para estimar a resistência à tração e a eficácia erosiva de geotêxteis naturais obteve excelente desempenho preditivo, com alta precisão na previsão dos efeitos de redução do escoamento e sedimentos.

2.4.1 Arquitetura de uma RNA

Uma RNA caracteriza-se como um modelo computacional de regressão ou classificação, sendo tipicamente representada por uma arquitetura de rede composta por camadas interconectadas de neurônios artificiais. No contexto da regressão, o modelo é comumente projetado com $K = 1$, ou seja, com uma única unidade de saída Y_1, cuja função é mapear as entradas para um valor contínuo. Este tipo de modelo é amplamente utilizado quando se busca modelar relações entre variáveis de forma quantitativa e preditiva (HASTIE, 2009).

Entretanto, as RNAs podem ser expandidas para lidar com múltiplas saídas simultaneamente, o que implica na utilização de uma arquitetura adaptada para problemas multivariados. Essa flexibilidade permite que a rede aprenda e modele interações mais complexas entre as variáveis independentes e as múltiplas variáveis dependentes, tornando-a uma ferramenta poderosa na resolução de problemas de alta complexidade, como aqueles encontrados em cenários de múltiplos objetivos ou em sistemas com diversas variáveis de resposta (HASTIE, 2009; HAYKIN, 2001).

2.4.1.1 Neurônio artificial: estrutura e funcionamento

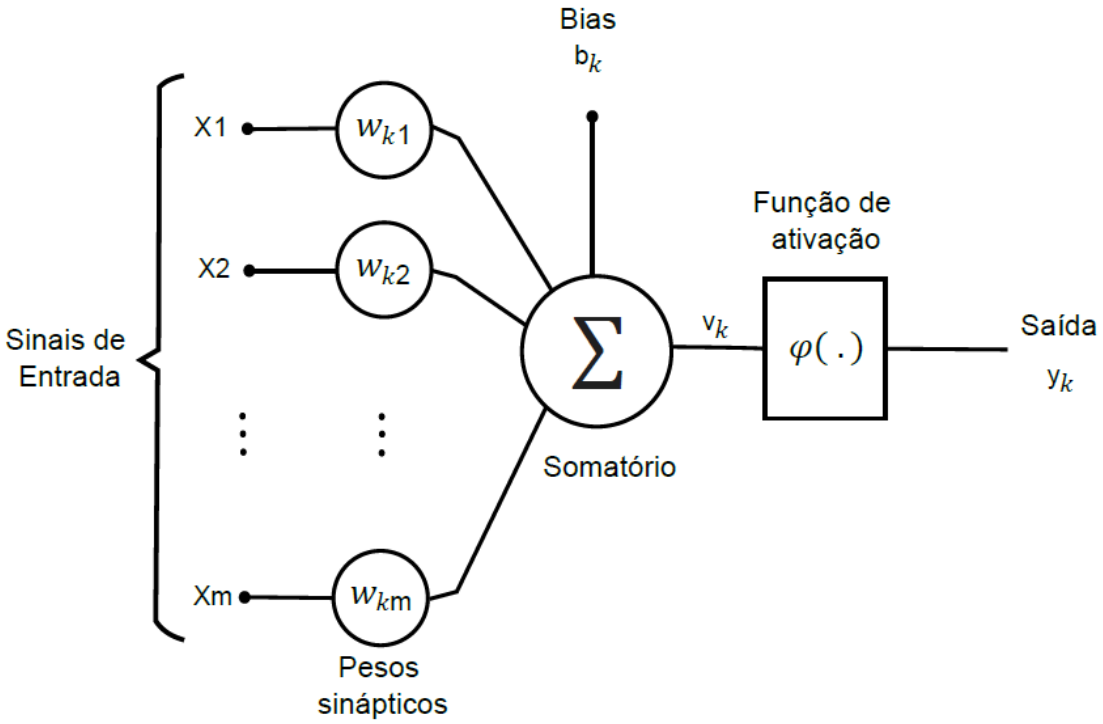
A arquitetura de um neurônio artificial, conforme detalhado por HAYKIN (2001), é composta por três componentes principais que operam de forma sequencial e interdependente (Figura 12):

I. Conjunto de sinapses:

As sinapses conectam o neurônio a outros neurônios ou a entradas externas, sendo ca-

racterizadas por um peso sináptico w_i. Este peso determina a intensidade e a polaridade da conexão, influenciando a amplitude e o sentido do sinal de entrada. Os pesos podem amplificar ($w_i > 1$), atenuar ($0 < w_i < 1$) ou inverter ($w_i < 0$) os sinais de entrada. A contribuição de cada entrada para o neurônio é dada pela multiplicação do sinal de entrada x_i pelo seu peso correspondente w_i.

Figura 12 – Representação gráfica de um neurônio artificial



Fonte: adaptado (HAYKIN, 2001)

II. Unidade de soma

O somador realiza a soma ponderada dos sinais de entrada, expressa pela Equação 14.

$$u_k = \sum_{j=1}^{m} w_{kj} \cdot x_j \tag{14}$$

onde $w_k = [w_{k1}, w_{k2}, \ldots, w_{km}]^T$ é o vetor de pesos e $x = [x_1, x_2, \ldots, x_m]^T$ é o vetor de entradas. Essa soma ponderada pode ser compactamente expressa pela Equação 15.

$$u_k = w_k^T x \tag{15}$$

onde $w_k^T x$ representa o produto escalar entre o vetor de pesos e o vetor de entradas. A variável u_k corresponde à projeção do vetor x na direção dos pesos w_k no espaço vetorial $\mathbb{R}^m$, sendo um passo crucial para determinar o alinhamento das entradas com os pesos.

III. Função de ativação

As funções de ativação desempenham um papel essencial nas redes neurais, pois são responsáveis por introduzir não-linearidades no modelo, permitindo que a rede aprenda padrões complexos e relações não lineares entre as entradas e saídas (HAYKIN, 2001). A seguir, discutem-se algumas das funções de ativação mais comuns e suas características:

A sigmoide (Figura 13) é uma função matemática diferenciável e limitada, que transforma os valores de entrada para um intervalo específico, como (0,1) ou (-1,1), o que a torna particularmente adequada para tarefas de classificação binária (Lederer, 2021). Sua forma sigmoidal em "S"confere-lhe suavidade e continuidade, facilitando a otimização dos parâmetros da rede por meio de métodos baseados em gradiente, como o gradiente descendente.

A tangente hiperbólica, um tipo de função sigmoide, é denotada por $f_{\text{tanh}} : \mathbb{R} \to (-1,1)$ e é definida pela Equação 16.

$$f_{\text{tanh}}(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{16}$$

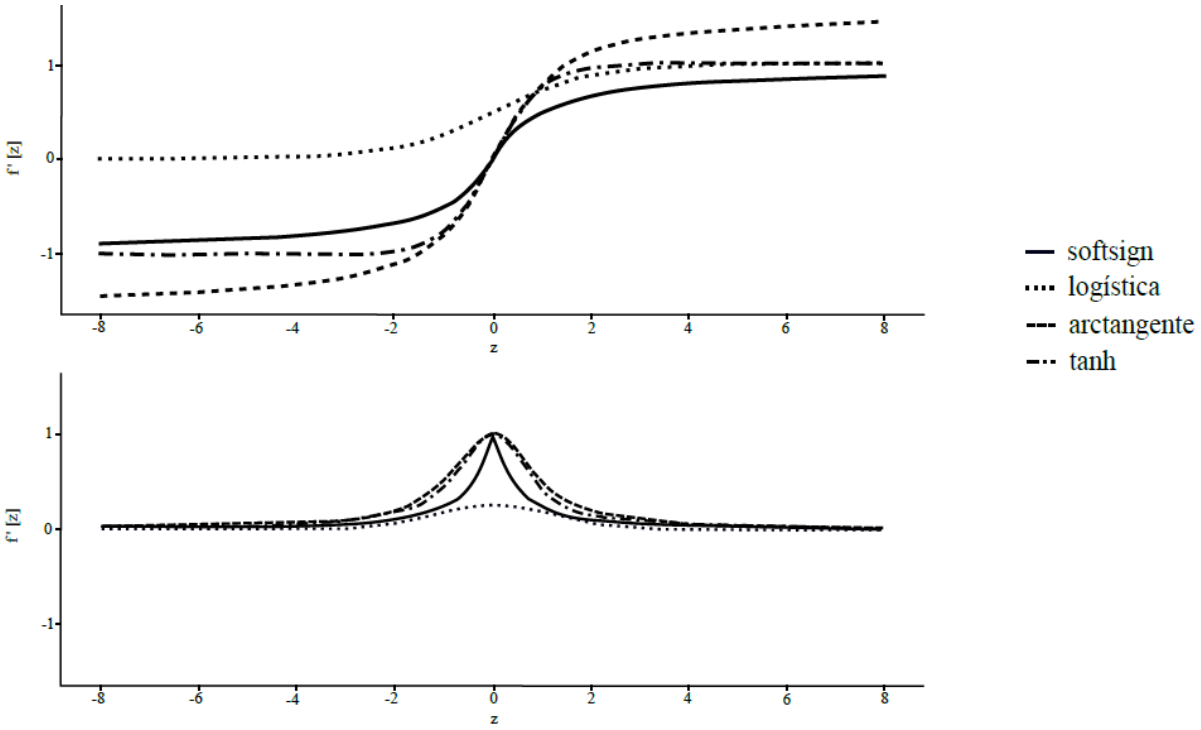
Esta função mapeia a reta real $\mathbb{R}$ para o intervalo (-1,1) e é amplamente utilizada em redes neurais devido às suas propriedades de suavização e diferenciação contínua, o que a torna eficaz em processos de otimização e em modelos preditivos. Apesar de suas qualidades, a sigmoide sofre com o problema de "vanishing gradients"(gradientes desaparecendo) quando utilizada em redes neurais profundas. Esse problema ocorre devido ao fato de que as derivadas da sigmoide são pequenas para valores de entrada extremos (positivos ou negativos), o que resulta em gradientes próximos de zero, dificultando o treinamento e a atualização eficiente dos pesos (LEDERER, 2021).

As funções de ativação linear por partes (Figura 14), como a ReLU (Rectified Linear Unit) e suas variantes, são compostas por segmentos lineares, proporcionando maior eficiência computacional em comparação com funções que dependem de operações mais pesadas, como exponenciais. A ReLU é definida pela Equação 17.

$$\text{ReLU}(x) = \max(0, x) \tag{17}$$

Essa função é amplamente utilizada em redes neurais profundas devido à sua simplicidade computacional e ao fato de que mantém uma derivada não nula para $x > 0$, o que

Figura 13 – Visualização de funções de ativação sigmoides



Fonte: adaptado (LEDERER, 2021)

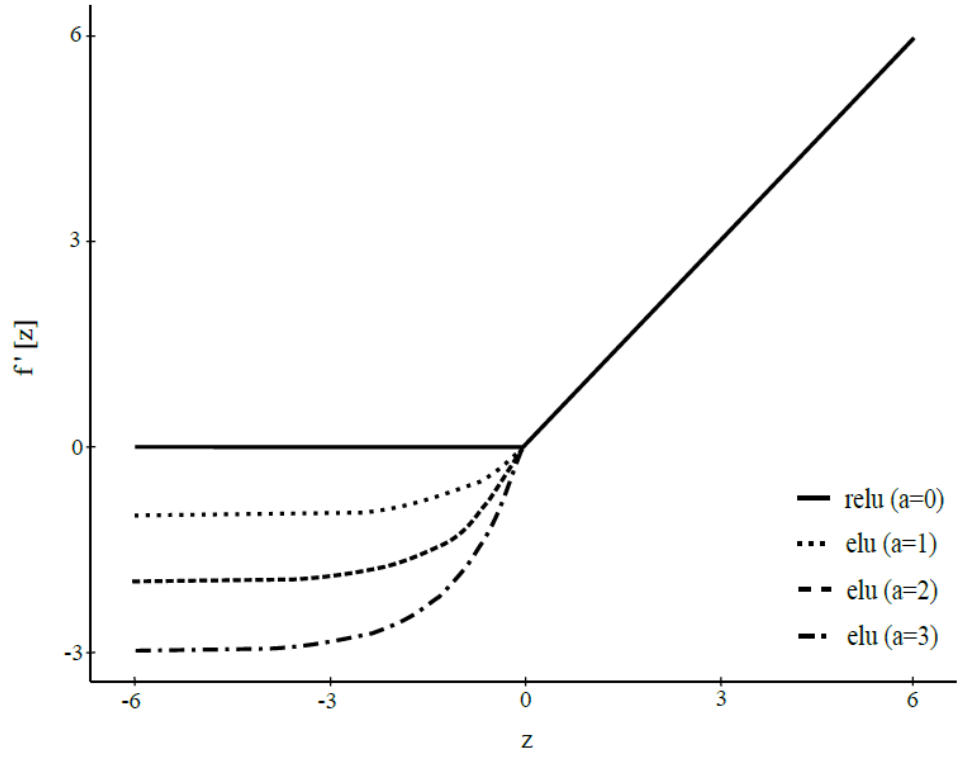
ajuda a resolver o problema de gradientes desaparecendo. No entanto, ela apresenta a desvantagem de resultar em "neurônios mortos"quando a entrada é negativa, ou seja, a saída da ReLU é zero, o que pode prejudicar o aprendizado em algumas situações (LEDERER, 2021).

2.4.2 Camadas de uma RNA

A estrutura de uma rede neural é composta por uma série de camadas, cada uma desempenhando um papel específico no processo de processamento e aprendizado dos dados. Essas camadas são organizadas de forma hierárquica, permitindo que a rede aprenda representações progressivamente mais complexas dos dados de entrada (HAYKIN, 2001; HASTIE, 2009).

A camada de entrada contém os neurônios responsáveis por receber os dados de entrada. Cada neurônio nesta camada está associado a uma variável específica do conjunto de dados, e sua função é simplesmente transmitir os valores dessas variáveis para as camadas subsequentes. A quantidade de neurônios na camada de entrada é diretamente proporcional ao número de características ou atributos dos dados de entrada. Os valores fornecidos aos neurônios de entrada não são processados ou modificados; eles servem como a base para os cálculos subsequentes realizados pela rede (HAYKIN, 2001; HASTIE, 2009).

Figura 14 – Visualização de funções de ativação com segmentação linear



Fonte: adaptado (LEDERER, 2021)

As camadas ocultas são camadas intermediárias situadas entre a camada de entrada e a camada de saída. Cada neurônio em uma camada oculta realiza a soma ponderada dos sinais de entrada recebidos de neurônios da camada anterior. Essa soma ponderada é então passada por uma função de ativação, que introduz não-linearidades essenciais para o aprendizado de padrões complexos. O número de camadas ocultas e o número de neurônios em cada camada dependem da complexidade do problema que a rede está tentando modelar. Redes neurais mais complexas, como as redes profundas (deep learning), podem ter várias camadas ocultas, permitindo que a rede capture padrões hierárquicos e de alta complexidade nos dados. A escolha da arquitetura (quantidade de camadas e neurônios) impacta diretamente no desempenho e capacidade de generalização da rede (HAYKIN, 2001; HASTIE, 2009).

A camada de saída é responsável por gerar a resposta final da rede neural, que pode ser uma previsão numérica ou uma classificação, dependendo do tipo de tarefa em questão. O número de neurônios na camada de saída está relacionado ao número de classes em uma tarefa de classificação ou ao número de valores contínuos a serem previstos em uma tarefa de regressão. As saídas dos neurônios da camada final são obtidas após a realização dos cálculos nas camadas anteriores, e elas representam as predições ou decisões da rede com base no aprendizado realizado durante o treinamento (HAYKIN, 2001; HASTIE, 2009).

Essas camadas funcionam de maneira sequencial, com cada uma contribuindo para o processamento dos dados de forma progressiva e cada vez mais abstrata. O treinamento de uma rede neural envolve a atualização dos pesos das conexões entre os neurônios, de forma a minimizar o erro entre a previsão da rede e o valor real desejado, através de métodos de otimização (HAYKIN, 2001; HASTIE, 2009).

2.4.3 Tipos de RNAs

As redes neurais podem ser classificadas em diversos tipos, cada uma com características específicas que a tornam adequada para diferentes tipos de problemas e aplicações. A seguir, são descritos os tipos mais comuns de redes neurais, com ênfase nas suas particularidades e áreas de aplicação.

As redes de propagação direta representam a arquitetura neural mais básica e amplamente utilizada. Nessas redes, a informação flui de forma unidirecional, ou seja, dos neurônios da camada de entrada, passando pelas camadas ocultas, até chegar à camada de saída, sem feedback entre os neurônios. Este modelo é ideal para tarefas de classificação e regressão onde a entrada não depende de estados passados ou de sequências temporais. A simplicidade dessa estrutura torna as redes de propagação direta eficientes, mas limitadas em aplicações que exigem memória de longo prazo ou considerações temporais (HAYKIN, 2001).

Ao contrário das redes propagação direta, as redes neurais recorrentes (RNRs) possuem conexões cíclicas, o que permite que a informação seja retroalimentada para os neurônios anteriores. Isso cria um tipo de "memória"que é útil para lidar com sequências de dados, como texto ou séries temporais. O mecanismo de retroalimentação nas RNNs permite que essas redes capturem dependências temporais, sendo amplamente aplicadas em tarefas como processamento de linguagem natural, tradução automática e previsão de séries temporais (HASTIE, 2009). Contudo, as RNNs tradicionais podem sofrer com o problema de "gradientes desaparecendo"durante o treinamento de sequências longas, o que limita sua eficácia.

As redes convolucionais (CNNs) são especialmente projetadas para processar dados com uma estrutura de grade, como imagens e vídeos. Elas utilizam operações de convolução, que aplicam filtros (ou kernels) sobre as entradas, permitindo a extração hierárquica de características espaciais, como bordas, texturas e formas. Essa capacidade de hierarquização torna as CNNs ideais para tarefas de visão computacional, como reconhecimento de objetos, segmentação de imagens e detecção de faces. Além disso, as CNNs utilizam a propriedade de invariância

espacial, onde as características extraídas são invariantes a translações, escalas e rotações, o que as torna poderosas para o processamento de imagens complexas (LECUN *et al.*, 1998).

As redes de memória de longo prazo (LSTM) são uma variante das RNRs, desenvolvidas para superar as limitações das redes neurais recorrentes tradicionais, particularmente o problema de gradientes desaparecendo, que impede que as redes aprendam com sequências longas. As LSTMs utilizam células de memória especializadas que podem manter informações por períodos mais longos, permitindo à rede aprender dependências de longo prazo dentro de sequências. Esse modelo é amplamente utilizado em tarefas que exigem uma longa memória temporal, como previsão de séries temporais, tradução automática e geração de texto (HOCHREITER; SCHMIDHUBER, 1997).

2.4.4 Treinamento das RNAs

O treinamento de redes neurais constitui um processo iterativo de extrema importância, com o objetivo de otimizar o desempenho do modelo por meio do ajuste refinado dos parâmetros internos, denominados pesos. O vetor de parâmetros da rede é composto por diversos elementos, sendo os pesos os principais componentes ajustados durante o treinamento. Esses parâmetros variam em estrutura e valor conforme o número de camadas e neurônios presentes na rede, refletindo a complexidade e a arquitetura específica do modelo adotado. A definição precisa e a escolha adequada desses parâmetros são cruciais para assegurar uma aprendizagem eficiente e uma alta capacidade de generalização, visto que são responsáveis por determinar a propagação e o processamento dos sinais nas diferentes camadas da rede (HASTIE, 2009).

2.4.4.1 Função de custo

A função de custo, também denominada função de erro, é uma métrica fundamental utilizada para quantificar a discrepância entre as previsões geradas pelo modelo e os valores reais observados. A função de custo desempenha um papel fundamental no processo de treinamento, pois é responsável por orientar o ajuste do modelo, indicando a direção e a magnitude das modificações necessárias para aprimorar suas previsões. Em problemas de regressão, a função de custo mais utilizada é a soma dos erros quadrados, também conhecida como Erro Quadrático Médio (EQM), que quantifica a média dos quadrados das diferenças entre os valores previstos e os reais. Em contraste, em problemas de classificação, especialmente nas tarefas de classificação multi-classe, a função de custo frequentemente adotada é a entropia cruzada, que avalia a diver-

gência entre as distribuições de probabilidade das classes previstas e as reais, proporcionando uma medida eficiente de desajuste entre essas distribuições (HASTIE, 2009).

Durante o treinamento da rede neural, o objetivo principal é minimizar essa função de custo, ajustando iterativamente os pesos da rede para melhorar a capacidade de previsão do modelo. A minimização da função de erro é realizada por meio de algoritmos de otimização que ajustam os parâmetros da rede para reduzir a discrepância entre as previsões da rede e os valores reais (HASTIE, 2009).

2.4.4.2 Algoritmo de retropropagação

A minimização da função de erro em redes neurais é comumente realizada por meio de um algoritmo denominado retropropagação (*backpropagation*), que utiliza a técnica de gradiente descendente. O algoritmo de retropropagação consiste em um processo iterativo que envolve duas passagens principais, essenciais para o ajuste dos pesos da rede neural e, consequentemente, a melhoria da precisão do modelo (HAYKIN, 2001; HASTIE, 2009).

I. **Passo para adiante (Forward Pass)**: Nesta etapa, os dados de entrada são propagados por todas as camadas da rede neural, a partir da camada de entrada até a camada de saída. O objetivo é calcular as saídas previstas pela rede, baseadas nas ativações dos neurônios e nas ponderações atuais dos pesos.

II. **Passo para trás (Backward Pass)**: Após a obtenção da saída da rede, calcula-se o erro, que é a diferença entre as previsões da rede e os valores reais. Esse erro é então retropropagado para as camadas anteriores, começando pela camada de saída e seguindo em direção à camada de entrada. Durante esse processo, os gradientes do erro em relação aos pesos da rede são computados e utilizados para ajustar os pesos, de forma a minimizar a discrepância entre as previsões e os valores reais.

   A atualização dos pesos em cada iteração $r+1$ é dada pela Equação 18.

$$\beta_{km}^{(r+1)} = \beta_{km}^{(r)} - \gamma^{(r)} \frac{1}{N} \sum_{i=1}^{N} \frac{\partial R_i}{\partial \beta_{km}^{(r)}} \tag{18}$$

   Onde $\beta_{km}^{(r)}$ representa o peso entre o neurônio m na camada k na iteração r; $\gamma^{(r)}$ é a taxa de aprendizado na iteração r, que controla a magnitude da atualização dos pesos; N é o

número total de exemplos no conjunto de dados; e $\frac{\partial R_i}{\partial \beta_{km}^{(r)}}$ é o gradiente da função de erro R_i em relação ao peso $\beta_{km}^{(r)}$ no exemplo i.

Esse processo de retropropagação e atualização dos pesos é repetido de forma iterativa até que a função de erro atinja um valor mínimo aceitável ou até que um número máximo de iterações seja alcançado. No entanto, em redes neurais profundas, onde o número de camadas e parâmetros é muito grande, a retropropagação pode se tornar computacionalmente custosa e lenta. Nesse contexto, métodos adicionais, como a otimização estocástica do gradiente (SGD) ou variantes como Adam, podem ser empregados para acelerar o processo de convergência, melhorando a eficiência do treinamento e ajudando a superar problemas como o gradiente desaparecendo (HAYKIN, 2001; HASTIE, 2009).

2.4.4.3 Otimizadores

O gradiente descendente é uma das técnicas clássicas de otimização utilizadas no treinamento de redes neurais. Seu funcionamento baseia-se na atualização iterativa dos parâmetros da rede (pesos) na direção oposta ao gradiente da função de custo. Ou seja, a cada iteração, os parâmetros são ajustados de forma a reduzir a função de custo, com a intenção de minimizar o erro da rede. O algoritmo é simples, mas pode ser computacionalmente dispendioso, principalmente em redes neurais profundas ou quando o número de parâmetros é grande, já que o cálculo do gradiente para cada amostra pode ser muito demorado. Além disso, o desempenho do gradiente descendente pode ser sensível à escolha da taxa de aprendizado, sendo necessário um cuidadoso ajuste para garantir a convergência eficiente (HAYKIN, 2001; HASTIE, 2009).

O algoritmo LBFGS (Limited-memory BroydenFletcherGoldfarbShanno) é uma técnica de otimização baseada em gradientes, eficiente em problemas com grande número de parâmetros, como no treinamento de redes neurais profundas. Pertencente à classe dos métodos quasi-Newton, o LBFGS aproxima a inversa da matriz Hessiana, melhorando a taxa de convergência sem armazenar a matriz completa, o que o torna adequado para problemas de alta dimensionalidade (HAYKIN, 2001; ALI, 2021).

A atualização dos parâmetros no algoritmo LBFGS é dada pela fórmula:

$$\theta_{t+1} = \theta_t - \alpha \nabla C(\theta_t) \tag{19}$$

onde θ_t é o vetor de parâmetros da rede neural na iteração t; α é a taxa de aprendizado, que controla a magnitude da atualização; e $\nabla C(\theta_t)$ é o gradiente da função de custo $C(\theta_t)$ em

relação aos parâmetros θ_t.

Entre as técnicas mais recentes e amplamente utilizadas, o algoritmo Adam (Adaptive Moment Estimation) se destaca por combinar os benefícios do gradiente descendente com métodos baseados em momentos. O Adam ajusta a taxa de aprendizado para cada parâmetro com base em estimativas adaptativas da média e da variância dos gradientes. A Equação 20 fornece o algoritmo Adam para atualizar os parâmetros θ (KINGMA; BA, 2015).

$$\theta_{t+1} = \theta_t - \alpha \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \varepsilon} \tag{20}$$

onde θ_t é o vetor de parâmetros na iteração t, α é a taxa de aprendizado, $\hat{m}_t$ é a estimativa corrigida do primeiro momento (média dos gradientes), $\hat{v}_t$ é a estimativa corrigida do segundo momento (variância dos gradientes) e ε é um pequeno valor para evitar divisões por zero.

O Adam combina a eficiência computacional com uma convergência estável, sendo amplamente empregado em aplicações práticas devido à sua robustez em problemas com grandes conjuntos de dados ou parâmetros (KINGMA; BA, 2015).

2.5 Árvores de Decisão

As árvores de decisão, amplamente utilizadas em aprendizado de máquina, podem ser classificadas em dois tipos principais: classificação e regressão, sendo ambas empregadas na construção de modelos preditivos baseados em dados (LOH, 2011). Esses modelos são construídos por meio de uma divisão recursiva do espaço de dados, em que cada subdivisão está vinculada a um modelo preditivo simplificado ajustado aos dados do respectivo subconjunto. O processo resulta em uma estrutura hierárquica que pode ser visualizada como uma árvore de decisão, proporcionando uma visão clara e organizada das etapas envolvidas na tomada de decisão.

As árvores de decisão são algoritmos altamente flexíveis, aptos a desempenhar tanto tarefas de classificação quanto de regressão, além de resolver problemas com múltiplas variáveis de saída. Sua adaptabilidade, aliada à simplicidade conceitual, faz delas ferramentas robustas para modelagem, permitindo ajustar-se a conjuntos de dados complexos e extrair informações relevantes de maneira eficaz. Elas se destacam pela facilidade de interpretação, uma vez que suas decisões são organizadas em uma estrutura hierárquica e intuitiva. Por esse motivo, são frequentemente classificadas como modelos de “caixa branca”, pois permitem aos usuários compreender claramente sua lógica interna. Em contraste, abordagens mais sofisticadas, como Florestas

Aleatórias (Random Forests) e redes neurais, são tipicamente vistas como modelos de "caixa preta", devido à complexidade de seus mecanismos internos e à dificuldade de interpretar como as previsões são geradas (GÉRON, 2022).

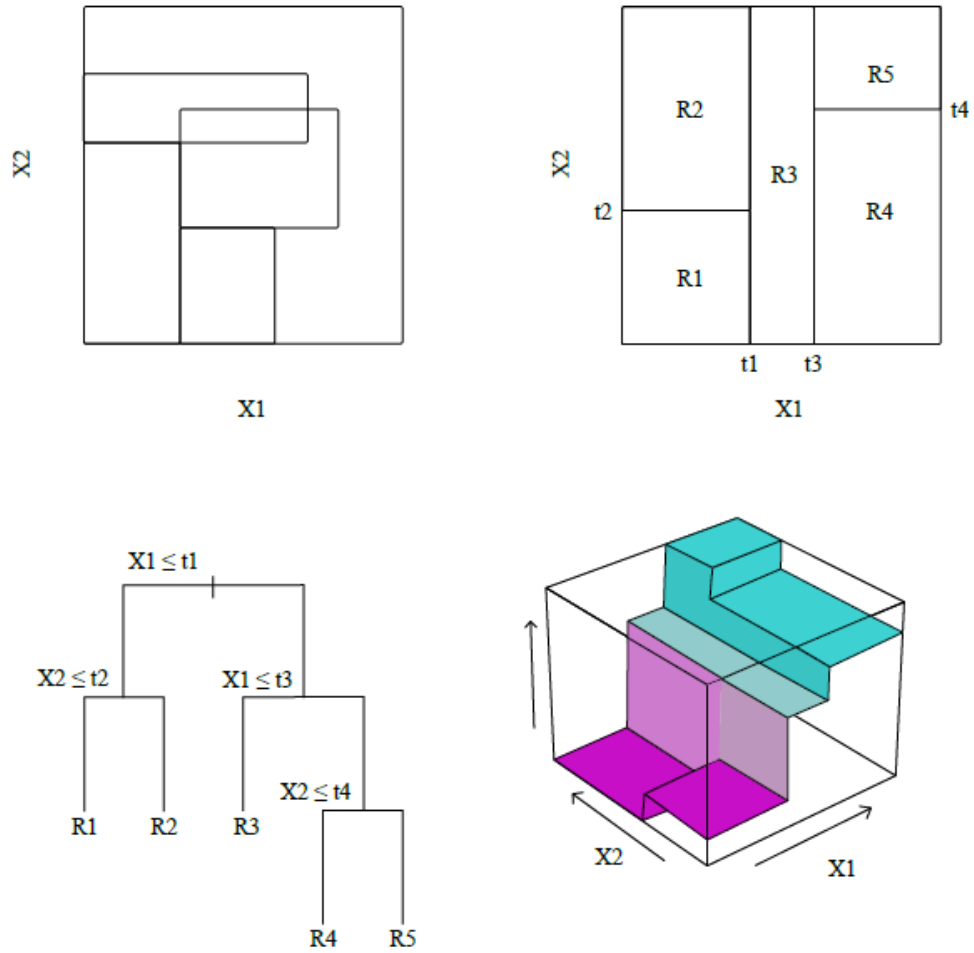
Pesquisas recentes têm demonstrado a eficiência das abordagens de soft computing (computação flexível), como as árvores de decisão, na previsão da profundidade de erosão em estruturas hidráulicas. SAMADI *et al.* (2014) mostraram que métodos como as árvores de modelo M5 (M5 Model Tree uma árvore de decisão com modelos lineares nas folhas) e as árvores de classificação e regressão (CART Classification and Regression Trees) superam fórmulas empíricas, proporcionando equações mais precisas e práticas, com destaque para o modelo M5 em termos de acurácia. GOYAL e OJHA (2011) compararam redes neurais artificiais (RNA) (Artificial Neural Networks ANN), Máquinas de Vetores de Suporte (Support Vector Machines SVM) e a árvore de modelo M5, concluindo que SVM e M5 são mais eficazes na estimativa da profundidade, largura e comprimento da cavidade de erosão abaixo de vertedouros tipo ski-jump. Além disso, esses métodos fornecem expressões explícitas para aplicação prática em projetos de engenharia. Já SAMADI *et al.* (2020) aplicaram as técnicas MARS (Multivariate Adaptive Regression Splines Splines de Regressão Adaptativa Multivariada), que utilizam funções por partes para capturar relações não lineares entre variáveis, e CART, uma abordagem baseada em árvores que pode ser usada tanto para regressão quanto para classificação, para estimar a profundidade de erosão induzida por ondas em grupos de estacas.

Dado um conjunto de dados $D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$, em que cada $x_i \in \mathbb{R}^p$ representa um vetor de atributos de entrada com p dimensões, e $y_i \in \mathbb{R}$ corresponde ao valor contínuo de saída associado, o conjunto caracteriza um modelo de dados para análise supervisionada, onde os vetores x_i descrevem as variáveis preditoras e y_i denotam as variáveis dependentes. O objetivo é particionar o espaço de entrada $\mathbb{R}^p$ em M regiões $R_1, R_2, \ldots, R_M$, de forma a minimizar o erro preditivo total. Em cada região R_j, ajusta-se um modelo constante c_j, sendo c_j o valor que minimiza a soma dos erros quadráticos na respectiva região (Figura 15).

2.5.1 Regressão

Árvores de decisão para regressão constituem uma abordagem amplamente empregada em aprendizado de máquina para a previsão de variáveis contínuas, destacando-se pela simplicidade estrutural e alta interpretabilidade. Fundamentadas em um processo hierárquico, essas árvores particionam recursivamente o espaço de entrada em sub-regiões menores, de modo a

Figura 15 – Visualização da construção de uma árvore de decisão em duas dimensões



Fonte: adaptado (HASTIE, 2009)

minimizar um critério de erro pré-definido. Cada divisão ou nó de decisão é selecionado com base em uma métrica que avalia o ganho informacional, maximizando a homogeneidade dos dados dentro das sub-regiões resultantes (HASTIE, 2009).

Considerando um conjunto de dados composto por p variáveis de entrada e uma variável de resposta, para cada uma das N observações, temos o par $(\mathbf{x}_i, y_i)$, onde $\mathbf{x}_i = (x_{i1}, x_{i2}, \ldots, x_{ip})$ e y_i representa o valor de resposta associado à observação i, para $i = 1, 2, \ldots, N$ (HASTIE, 2009).

O algoritmo de construção da árvore deve ser capaz de decidir de forma autônoma não apenas as variáveis de divisão, mas também os pontos de divisão, bem como a topologia (ou estrutura) da árvore. Em termos práticos, isso implica que o algoritmo deve determinar como particionar o espaço de entrada $\mathbb{R}^p$ de forma a minimizar a função de erro, levando em consideração as diferentes variáveis e seus valores de corte (HASTIE, 2009).

A função de predição é formalmente definida pela Equação 21.

$$f(\mathbf{x}) = \sum_{m=1}^{M} c_m \cdot \mathbb{I}(\mathbf{x} \in R_m), \tag{21}$$

onde c_m é um coeficiente associado a cada região R_m no domínio de $\mathbf{x}$, e $\mathbb{I}(\mathbf{x} \in R_m)$ é uma função indicadora que assume o valor 1 se $\mathbf{x}$ pertence à região R_m, e 0 caso contrário.

O particionamento das regiões R_m é realizado de forma recursiva, com cada divisão definida por condições de corte em uma dimensão x_j do espaço de entrada. A métrica de minimização, frequentemente adotada, é a soma dos erros quadráticos (SSE) dentro de cada região, representada pela Equação 22 (HASTIE, 2009).

$$SSE = \sum_{m=1}^{M} \sum_{\mathbf{x}_i \in R_m} (y_i - c_m)^2. \tag{22}$$

Essa abordagem gera um modelo altamente interpretável, no qual a estrutura da árvore reflete diretamente os critérios de decisão que particionam o espaço de entrada, garantindo previsões locais adaptadas às características dos dados. Estratégias como a poda ou o ajuste da profundidade da árvore são frequentemente utilizadas para evitar o sobreajuste e melhorar a capacidade de generalização do modelo (HASTIE, 2009).

As divisões em árvores de decisão para regressão são realizadas de forma iterativa, com o objetivo de minimizar o erro preditivo em cada etapa. O critério comumente utilizado para avaliar a qualidade das divisões é o Erro Quadrático Médio (MSE), definido pela Equação 23 (HASTIE, 2009).

$$MSE = \sum_{j=1}^{M} \sum_{\mathbf{x}_i \in R_j} (y_i - c_j)^2, \tag{23}$$

onde:

$$c_j = \frac{1}{|R_j|} \sum_{\mathbf{x}_i \in R_j} y_i. \tag{24}$$

A cada iteração, uma região R é dividida em duas sub-regiões, R_L (esquerda) e R_R (direita), com o objetivo de minimizar o erro total. O erro total após a divisão é dado por pela Equação 25 (HASTIE, 2009).

$$\text{Erro Total} = \sum_{\mathbf{x}_i \in R_L} (y_i - c_L)^2 + \sum_{\mathbf{x}_i \in R_R} (y_i - c_R)^2, \tag{25}$$

onde:

$$c_L = \frac{1}{|R_L|} \sum_{\mathbf{x}_i \in R_L} y_i, \quad c_R = \frac{1}{|R_R|} \sum_{\mathbf{x}_i \in R_R} y_i. \tag{26}$$

O processo de divisão busca maximizar a redução no erro total em cada passo. Para isso, identifica-se a melhor combinação de ponto de corte s e dimensão j que particionam a região R em R_L e R_R. A redução no erro, ou ganho, é calculada pela Equação 27 (HASTIE, 2009).

$$\Delta\text{Erro} = \text{Erro em } R - (\text{Erro em } R_L + \text{Erro em } R_R). \tag{27}$$

Essa estratégia permite ajustar localmente o modelo às características dos dados em cada sub-região, promovendo flexibilidade e eficiência no aprendizado. Entretanto, para evitar sobreajuste, técnicas como poda e a imposição de limites na profundidade da árvore são frequentemente aplicadas (HASTIE, 2009).

Além do Erro Quadrático Médio (MSE), outros critérios de avaliação de erro podem ser empregados, como o Erro Absoluto Médio (MAE), dado pela Equação 28 (HASTIE, 2009).

$$MAE = \sum_{j=1}^{M} \sum_{\mathbf{x}_i \in R_j} |y_i - c_j|, \tag{28}$$

onde c_j representa a média dos valores y_i na região R_j. O MAE tem a vantagem de ser menos sensível a outliers em comparação ao MSE, proporcionando uma métrica alternativa para otimização de modelos.

Para evitar o *overfitting*, que ocorre quando o modelo se ajusta excessivamente aos dados de treinamento, resultando em uma baixa capacidade de generalização, são aplicadas técnicas de regularização, como a poda por complexidade de custo. Esse procedimento visa balancear o ajuste do modelo com sua simplicidade, prevenindo a modelagem de ruídos nos dados (HASTIE, 2009).

A poda pode ser realizada de diferentes maneiras, como pós-poda e pré-poda. A pós-poda envolve a construção inicial de uma árvore de grande porte seguida pela remoção de nós de baixo valor preditivo, o que tende a melhorar a generalização do modelo. Já a pré-poda limita o crescimento da árvore durante a construção, restringindo a profundidade ou número de nós com base em critérios definidos anteriormente. Ambas as abordagens influenciam diretamente a complexidade do modelo, e a escolha entre elas depende do trade-off desejado entre a redução do overfitting e a perda de precisão preditiva (HASTIE, 2009).

A poda realizada após a construção de uma árvore de decisão inicial T_0 consiste na remoção de nós internos para simplificar o modelo, utilizando a Equação 29 como critério de complexidade de custo (HASTIE, 2009).

$$C_\alpha(T) = \sum_{m=1}^{|T|} N_m Q_m(T) + \alpha|T|, \tag{29}$$

onde N_m é o número de observações no nó m, $Q_m(T)$ é a soma dos quadrados dos erros no nó m, e α é o parâmetro de regularização que controla a complexidade do modelo, equilibrando o erro de ajuste e a penalização pela complexidade da árvore.

A constante α atua como um hiperparâmetro que regula o trade-off entre a complexidade da árvore e seu desempenho, sendo ajustado para evitar que a árvore cresça excessivamente, mantendo a capacidade de generalização do modelo (HASTIE, 2009).

Uma vez construída, a árvore de decisão realiza previsões atribuindo a uma nova entrada $\mathbf{x}$ o valor c_j correspondente à região R_j em que $\mathbf{x}$ se encontra. A previsão $\hat{y}$ é dada pela Equação 30 (HASTIE, 2009).

$$\hat{y} = c_j \quad \text{se } \mathbf{x} \in R_j. \tag{30}$$

O modelo resultante é altamente interpretável, permitindo a visualização das decisões tomadas em cada nível da árvore, o que facilita a compreensão dos fatores que influenciam as previsões (HASTIE, 2009).

2.5.1.1 Classificação

As árvores de decisão de classificação são um modelo de aprendizado supervisionado que organiza os dados em uma estrutura hierárquica, realizando divisões recursivas a partir de um conjunto de atributos, com o objetivo de classificar observações com base em uma variável-alvo categórica. A construção de uma árvore de decisão envolve o uso de diversos critérios para particionar os dados, com foco em maximizar a pureza dos nós, ou seja, minimizar a heterogeneidade das observações dentro de cada nó. Para isso, são empregadas métricas de impureza, que quantificam o grau de mistura das classes dentro de cada subconjunto gerado pela divisão (HAN *et al.*, 2011).

O principal objetivo das árvores de decisão de classificação é maximizar a pureza dos nós após a divisão dos dados. A pureza é medida por métricas como entropia e índice de Gini,

que avaliam a distribuição das classes dentro de cada nó e são fundamentais para a seleção dos melhores pontos de divisão (HAN *et al.*, 2011).

A entropia é uma medida da incerteza ou da desordem associada à distribuição das classes em uma partição de dados. Para um conjunto de dados S com K classes, a entropia é definida pela Equação 31 (HAN *et al.*, 2011).

$$H(S) = -\sum_{i=1}^{K} p_i \log_2(p_i) \tag{31}$$

Onde p_i representa a proporção de exemplos da classe i no conjunto S. A entropia atinge seu valor máximo quando as classes estão distribuídas de forma uniforme, indicando uma alta incerteza na classificação. À medida que a pureza de um nó aumenta (ou seja, quando as observações em um nó pertencem predominantemente a uma única classe), a entropia diminui.

O índice de Gini mede a probabilidade de um elemento ser classificado incorretamente, considerando a distribuição das classes em um nó. Para um conjunto S, o índice de Gini é definido pela Equação 32 (HAN *et al.*, 2011).

$$Gini(S) = 1 - \sum_{i=1}^{K} p_i^2 \tag{32}$$

Onde p_i é a proporção de exemplos da classe i no conjunto S. O valor do índice de Gini varia entre 0 (pureza máxima, onde todos os exemplos pertencem à mesma classe) e $1 - \frac{1}{K}$ (onde as classes estão igualmente distribuídas). Quanto menor o valor do índice de Gini, maior a pureza do nó.

O ganho de informação quantifica a redução na entropia resultante de uma divisão dos dados com base em um determinado atributo. Para um atributo A com possíveis valores v, o ganho de informação é calculado pela Equação 33 (HAN *et al.*, 2011).

$$IG(S,A) = H(S) - \sum_{v \in V} \frac{|S_v|}{|S|} H(S_v) \tag{33}$$

Onde S_v representa o subconjunto de S onde o atributo A assume o valor v, e $|S_v|$ é o número de exemplos neste subconjunto. O ganho de informação é maior quando a divisão resulta em subconjuntos mais puros, ou seja, quando a redução da incerteza após a divisão é significativa.

Para a construção da árvore de decisão, o algoritmo seleciona o atributo A^* que maximiza a separação dos dados, ou seja, aquele que resulta na menor impureza nos nós gerados

pela divisão. A escolha do melhor atributo é realizada por meio da maximização da métrica de impureza definida, que pode ser o ganho de informação ou a redução do índice de Gini, dependendo da estratégia adotada. Formalmente, a seleção do atributo ótimo é expressa pela Equação 34 (HAN *et al.*, 2011).

$$A^* = \arg\max_A \text{Métrica}(S, A) \tag{34}$$

Onde a métrica pode ser o ganho de informação $IG(S,A)$ ou a redução do índice de Gini, dependendo da implementação do modelo. A ideia central é escolher o atributo A^* que proporciona a maior melhoria na pureza dos nós, reduzindo assim a incerteza ou a probabilidade de erro na classificação dos dados. Esse processo é repetido recursivamente para cada nó até que sejam atingidos os critérios de parada estabelecidos, como a profundidade máxima da árvore ou o número mínimo de exemplos por nó (HAN *et al.*, 2011).

O crescimento da árvore de decisão é interrompido quando um dos seguintes critérios de parada é atendido (HASTIE, 2009; HAN *et al.*, 2011):

I. **Número mínimo de amostras por nó**: o crescimento da árvore é interrompido se o número de amostras $|S|$ em um nó for inferior a um valor mínimo predefinido $n_{\min}$. Este critério visa evitar que a árvore se torne excessivamente detalhada e sobreajustada, ajustando-se a ruídos ou variações triviais nos dados.

II. **Profundidade máxima da árvore**: caso a profundidade da árvore atinja o limite máximo predefinido $d_{\max}$, o algoritmo cessa o crescimento. A profundidade da árvore está diretamente relacionada à sua complexidade; esse critério é utilizado para controlar a capacidade de generalização do modelo, evitando o sobreajuste (*overfitting*).

III. **Pureza do nó**: o crescimento da árvore é interrompido quando a impureza do nó atinge o valor zero, ou seja, quando o nó se torna puramente homogêneo em relação à variável-alvo. Nesse ponto, todos os exemplos no nó pertencem à mesma classe, tornando-se desnecessária uma divisão adicional.

Esses critérios garantem que a árvore seja construída de maneira controlada, evitando um crescimento excessivo e promovendo um modelo que seja capaz de generalizar de forma eficiente (HASTIE, 2009; HAN *et al.*, 2011).

Após o treinamento do modelo, a classificação de uma nova observação é realizada por meio de um percurso na árvore de decisão, desde o nó raiz até um nó terminal. Esse percurso

segue as condições estabelecidas em cada nó intermediário com base nos valores dos atributos da observação (HASTIE, 2009; HAN *et al.*, 2011).

No nó terminal, a observação é atribuída à classe c que apresenta a maior proporção p_c de amostras pertencentes a essa classe no nó. Formalmente, a classe predita para uma observação x é determinada pela Equação 35 (HASTIE, 2009; HAN *et al.*, 2011).

$$y(x) = \arg\max_c p_c \tag{35}$$

Onde p_c é a proporção de amostras no nó terminal que pertencem à classe c, e $\arg\max_c$ indica que a classe escolhida é aquela que maximiza p_c.

Esse procedimento assegura que a predição seja consistente com a distribuição de classes observada nos dados de treinamento, refletindo as divisões hierárquicas da árvore (HASTIE, 2009; HAN *et al.*, 2011).

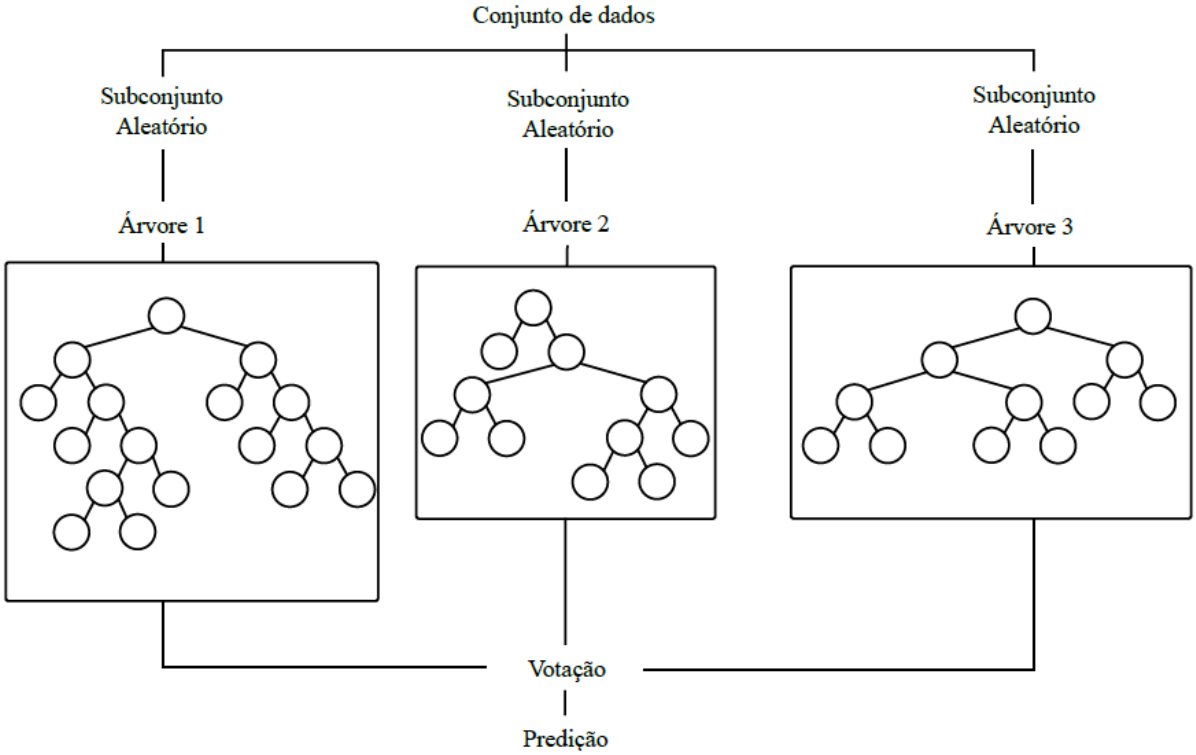
2.6 Florestas Aleatórias

As florestas aleatórias (*random forests*) constituem uma técnica avançada e amplamente empregada em aprendizado de máquina, aplicável tanto a problemas de classificação quanto de regressão. Este método se baseia no princípio de modelos de conjunto, no qual múltiplas árvores de decisão independentes são geradas e combinadas para produzir predições mais robustas e precisas (Figura 16).

Entre as pesquisas recentes que aplicaram florestas aleatórias na análise de barragens, GU *et al.* (2023) desenvolveram um modelo para previsão de deformação em barragens de arco super-altas, utilizando wavelet para redução de ruído e florestas aleatórias otimizado por um algoritmo de *salp swarm*, obtendo bons resultados em termos de desempenho e precisão. LI *et al.* (2023) propuseram um modelo para previsão de deformação em barragens de concreto faceado com enrocamento (CFRDs), aprimorando as árvores aleatórias com a substituição da árvore de decisão pelo algoritmo M5. Os resultados indicaram que essa abordagem apresentou melhor desempenho em comparação com modelos clássicos, como *Support Vector Regression* (SVR).

O conceito central por trás das florestas aleatórias é a aplicação do método de *bagging* (amostragem com reposição), introduzido por BREIMAN (2001). Esse método combina predições de múltiplos modelos ruidosos, mas aproximadamente não tendenciosos, para reduzir a variância global do modelo. Árvores de decisão são particularmente adequadas para essa abor-

Figura 16 – Diagrama esquemático da estrutura de florestas aleatórias



Fonte: adaptado (SU *et al.*, 2021)

dagem, pois conseguem capturar relações complexas e não lineares nos dados. Contudo, sua natureza inerentemente instável, caracterizada por alta sensibilidade às variações nos dados de entrada, as torna propensas a alta variância. O *bagging* mitiga esse problema, aumentando a robustez e a estabilidade do modelo.

No processo de *bagging*, cada árvore de decisão é construída a partir de um subconjunto do conjunto de dados original, gerado por amostragem com reposição. Essa estratégia assegura que cada árvore seja treinada em um conjunto de dados ligeiramente diferente, promovendo diversidade entre os classificadores individuais. Como resultado, a agregação (geralmente por média ou votação majoritária) das predições dessas árvores diversas produz um modelo final significativamente mais estável e preciso.

Um dos desafios do *bagging* é a correlação entre as árvores individuais, que pode limitar a efetividade na redução da variância do modelo. Para mitigar esse problema, as florestas aleatórias introduzem uma modificação fundamental: em cada nó da árvore, um subconjunto aleatório de variáveis preditoras é selecionado como candidato para a divisão, conforme proposto por BREIMAN (2001). Essa estratégia aumenta a diversidade estrutural entre as árvores, reduzindo a correlação entre elas e, consequentemente, melhorando a robustez do modelo agregado.

Formalmente, o impacto desse processo pode ser descrito da seguinte maneira:

- Quando m, o número de variáveis selecionadas aleatoriamente em cada nó, é reduzido, a correlação entre as árvores diminui devido à maior diversidade nas divisões realizadas.
- No entanto, valores excessivamente baixos de m podem comprometer a capacidade das árvores individuais de identificar padrões relevantes nos dados, prejudicando o desempenho geral do modelo.

Um compromisso adequado é frequentemente alcançado definindo $m = \sqrt{p}$, onde p é o número total de variáveis preditoras, ou utilizando valores próximos a essa proporção. Esse ajuste balanceia a diversidade estrutural das árvores com a preservação de sua capacidade preditiva, otimizando o desempenho das florestas aleatórias em uma ampla gama de aplicações.

Após a construção de B árvores no modelo de conjunto, a predição final é obtida por agregação. No caso de regressão, a predição final é dada pela média das saídas das árvores individuais, enquanto para classificação, utiliza-se o critério de votação majoritária.

Para regressão, o preditor de uma floresta aleatória é formalizado pela Equação 36.

$$\hat{f}_{\mathrm{rf}}^{(B)}(x) = \frac{1}{B}\sum_{b=1}^{B} T(x;\theta_b) \tag{36}$$

Onde $\hat{f}_{\mathrm{rf}}^{(B)}(x)$ é a predição agregada do modelo para a entrada x, B é o número total de árvores no modelo e $T(x;\theta_b)$ representa a predição da b-ésima árvore baseada nos parâmetros θ_b, que incluem as variáveis de divisão, os pontos de corte e os valores nos nós terminais.

Esse processo de agregação confere às florestas aleatórias sua robustez característica, ao reduzir a variância e evitar o sobreajuste, mesmo em cenários com um grande número de variáveis ou dados ruidosos.

Ao utilizarem subconjuntos amostrados com reposição para construir árvores independentes, as florestas aleatórias introduzem maior diversidade ao selecionar, em cada nó, um subconjunto aleatório de variáveis preditoras como candidatas à divisão (BREIMAN, 2001). Essa abordagem reduz a correlação entre as árvores, mitigando limitações típicas do *bagging*. O preditor final para regressão é calculado pela média das predições individuais, enquanto na classificação utiliza-se a votação majoritária, garantindo estabilidade mesmo em cenários de alta dimensionalidade e interações complexas. As florestas aleatórias destacam-se por sua robustez contra o *overfitting*, redução da variância por meio da agregação de múltiplos modelos e ampla

aplicabilidade em tarefas diversas, incluindo detecção de anomalias e modelagem de relações não lineares (HASTIE, 2009).

2.6.1 Convergência

Em Florestas Aleatórias, os classificadores individuais $h_1(X), h_2(X), \ldots, h_K(X)$ são compostos por Árvores de Decisão. A probabilidade de erro de generalização P_E^* pode ser analisada em termos de uma função chamada *margem* $mg(X,Y)$, que é definida como:

$$mg(X,Y) = \frac{1}{K}\sum_{k=1}^{K}\mathbb{I}(h_k(X)=Y) - \max_{j\neq Y}\frac{1}{K}\sum_{k=1}^{K}\mathbb{I}(h_k(X)=j) \tag{37}$$

onde $\mathbb{I}(\cdot)$ é a função indicadora, que assume o valor 1 se a condição for verdadeira e 0 caso contrário. A margem $mg(X,Y)$ mede a diferença entre a média dos votos atribuídos à classe correta Y e à classe incorreta j. Uma margem maior implica maior confiança na previsão do modelo, resultando em uma taxa de erro de generalização reduzida.

O erro de generalização P_E^* é a probabilidade de que a margem seja menor que zero:

$$P_E^* = P_{X,Y}(mg(X,Y) < 0) \tag{38}$$

À medida que o número de árvores K na floresta aumenta, a função de erro de generalização converge para um valor limite. Esse comportamento explica por que as florestas aleatórias não sofrem *overfitting* com o aumento do número de árvores, uma vez que o erro de generalização se estabiliza à medida que K cresce (BREIMAN, 1996).

2.6.2 Força e Correlação

A força de uma Floresta Aleatória, que é um dos principais determinantes de seu desempenho, é formalmente definida pela função da margem esperada s, dada pela Equação 39.

$$s = \mathbb{E}_{X,Y}[mr(X,Y)] \tag{39}$$

onde $mr(X,Y)$ é a margem da floresta aleatória, expressa como:

$$mr(X,Y) = P_\Theta(h(X,\Theta)=Y) - \max_{j\neq Y}P_\Theta(h(X,\Theta)=j) \tag{40}$$

A margem $mr(X,Y)$ quantifica a diferença entre a probabilidade de uma árvore classificar corretamente a instância X como pertencente à classe Y e a maior probabilidade de erro as-

sociada a qualquer outra classe j. A força s de um conjunto de classificadores é positiva quando a floresta possui uma boa capacidade de distinção entre as classes, o que contribui diretamente para um desempenho superior (BREIMAN, 2001).

Além disso, a correlação entre os classificadores individuais tem um papel crucial no desempenho global da floresta aleatória. A interação entre a força dos classificadores e a correlação entre eles influencia diretamente a variância da margem $mr(X,Y)$, que é dada pela Equação 41 (BREIMAN, 2001).

$$\text{var}(mr) = \mathbb{E}_{\Theta,\Theta'}\left[\text{cov}_{X,Y}\left(rmg(\Theta,X,Y)\cdot rmg(\Theta',X,Y)\right)\right] \tag{41}$$

onde $rmg(\Theta,X,Y)$ representa a função margem bruta, que indica se uma árvore individual acertou ou errou a classificação de Y. A correlação média entre as árvores, denotada por ρ, afeta a variância da margem. Quanto menor a correlação entre os classificadores, maior a redução da variância e, consequentemente, melhor será o desempenho da floresta aleatória, uma vez que a diversidade entre os classificadores contribui para a estabilidade e precisão do modelo.

A análise das interações entre a força dos classificadores e sua correlação leva à formulação de um limite superior para o erro de generalização. Utilizando a desigualdade de Chebyshev, é possível derivar a seguinte expressão para o erro de generalização P_E^*:

$$P_E^* \leq \frac{\rho(1-s^2)}{s^2} \tag{42}$$

Este limite fornece uma medida quantitativa de como o erro de generalização depende da correlação média ρ e da força s dos classificadores individuais. Um valor reduzido dessa razão implica um desempenho superior da floresta aleatória, pois indica que a correlação entre as árvores é baixa e sua força é alta. Em outras palavras, a floresta apresenta uma maior capacidade de discriminação entre as classes, com menores probabilidades de erro devido à diminuição da correlação entre os classificadores e ao aumento da sua capacidade de acerto (BREIMAN, 2001).

2.6.3 Amostras fora da amostra (Out-of-Bag)

Uma característica distintiva das florestas aleatórias é a utilização das amostras fora da amostra (OOB) para avaliar o desempenho do modelo durante o processo de treinamento. Em cada iteração, ao treinar uma árvore com um subconjunto dos dados, aproximadamente um terço das amostras são deixadas de fora. Essas amostras, denominadas amostras OOB, desempenham

um papel crucial na estimativa do erro de generalização do modelo, eliminando a necessidade de uma validação cruzada completa. A média dos preditores calculada para as amostras OOB fornece uma estimativa de erro altamente comparável à obtida por validação cruzada N-partida, tornando o processo de treinamento mais eficiente (BREIMAN, 2001). Esse método reduz a necessidade de um conjunto de validação separado e permite que o treinamento da floresta aleatória seja realizado de forma contínua, com o monitoramento do erro OOB à medida que o modelo é ajustado.

2.6.4 Importância das variáveis

Outro aspecto crucial das florestas aleatórias é a capacidade de avaliar a importância das variáveis no processo de predição. A avaliação da importância das variáveis é realizada por meio de duas abordagens principais. A primeira é baseada na melhoria do critério de divisão durante a construção das árvores. A cada divisão, a redução no critério de impureza (como o índice de Gini ou o erro quadrático médio) é acumulada para cada variável. Ao final do processo, é gerado um gráfico que reflete a importância relativa de cada variável no modelo.

A segunda abordagem utiliza as amostras fora da amostra (OOB) para calcular a importância. Após o treinamento, as variáveis são permutadas aleatoriamente, e a redução na precisão das predições é medida, fornecendo uma avaliação da contribuição de cada variável. Embora ambas as abordagens geralmente resultem em rankings semelhantes, a segunda abordagem pode ser considerada mais equilibrada, uma vez que reflete de forma mais uniforme o impacto de cada variável no desempenho do modelo (HASTIE, 2009).

2.6.5 Sobreajustee controle de complexidade

Embora as florestas aleatórias sejam amplamente reconhecidas por sua capacidade de mitigar o *overfitting*, elas podem, em determinadas condições, ser suscetíveis a esse problema. Isso é particularmente verdadeiro quando o número de variáveis relevantes é significativamente inferior ao número total de variáveis. SEGAL (2004) evidencia que, ao limitar a profundidade das árvores individuais dentro das florestas aleatórias, é possível observar melhorias no desempenho preditivo do modelo.

Contudo, ao ajustar parâmetros como a profundidade das árvores e o número total de árvores na floresta, é possível atenuar esse risco. A prática de utilizar árvores com profundidade total geralmente não resulta em impactos negativos substanciais, uma vez que a média das

previsões de múltiplas árvores tende a suavizar o modelo e prevenir o *overfitting*, desde que o número de árvores seja suficientemente elevado. Isso ocorre porque, por meio do processo de *bagging*, as florestas aleatórias reduzem a variância do modelo, sendo resilientes ao aumento do número de árvores (BREIMAN, 2001).

2.6.6 Viés

O viés de uma floresta aleatória pode ser analisado de forma análoga ao viés de uma árvore de decisão única, conforme expresso pela Equação 43.

$$\text{Bias}(x) = \mu(x) - \mathbb{E}_Z \left[\hat{f}_{\text{RF}}(x) \right] \tag{43}$$

onde $\mu(x)$ representa a função verdadeira e desconhecida, e $\mathbb{E}_Z[\hat{f}_{\text{RF}}(x)]$ é a expectativa do valor predito pela floresta aleatória para um ponto x. Essa equação ilustra que, embora a randomização e a redução do espaço amostral nas florestas aleatórias possam aumentar o viés em comparação a uma árvore de decisão sem poda, os ganhos em desempenho preditivo resultam principalmente da significativa redução na variância (BREIMAN, 2001; HASTIE, 2009)

Em problemas de regressão, especialmente quando se utiliza uma função de perda quadrática, as florestas aleatórias exibem um comportamento típico de *trade-off* entre viés e variância. Como os parâmetros das árvores de decisão em uma floresta são ajustados de forma independente, a média das predições provenientes de múltiplas árvores contribui para a redução da variância. No entanto, essa abordagem tende a aumentar o viés à medida que o número de árvores na floresta cresce. Esse padrão de comportamento é análogo ao observado na regressão *ridge*, onde a regularização atua na estabilização do modelo ao ajustar os coeficientes das variáveis, evitando assim o *overfitting* (HASTIE, 2009).

2.7 Boosting com árvores de regressão

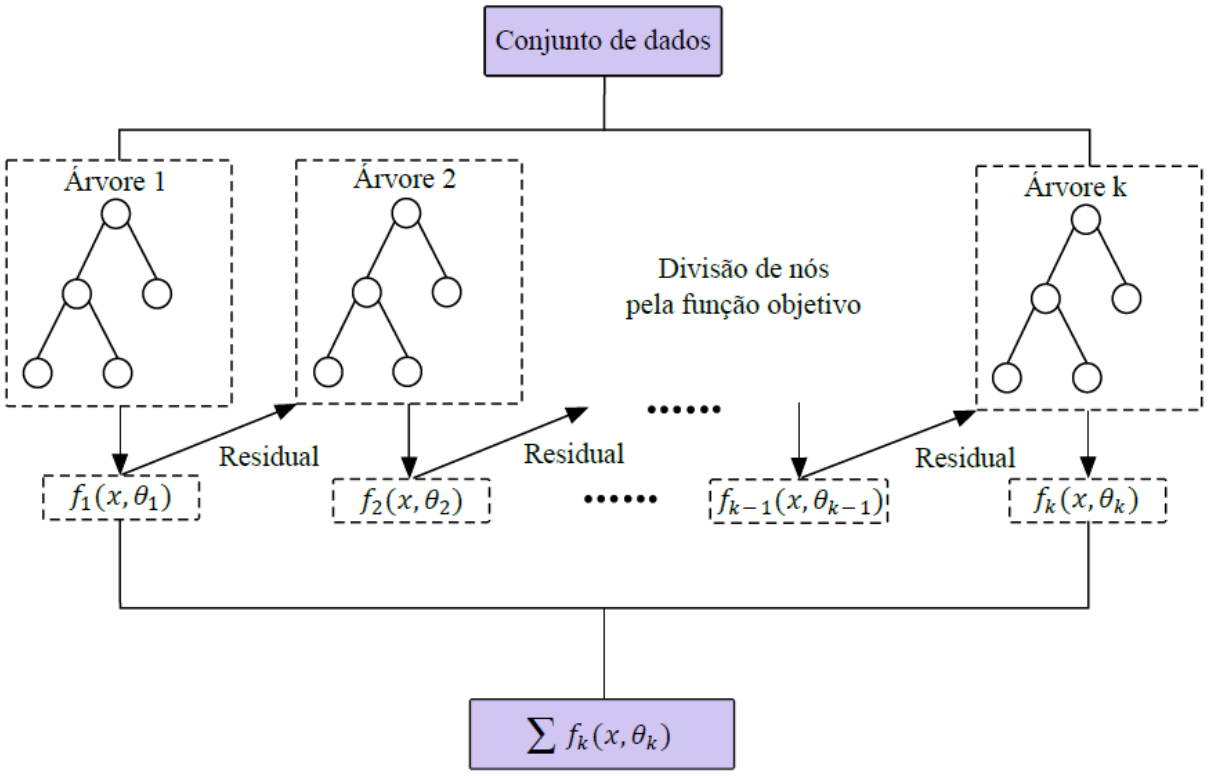
O *Boosting*, ou reforço, é uma técnica sofisticada de aprendizado de máquina, amplamente utilizada para aumentar a acurácia de modelos preditivos por meio da combinação sequencial de múltiplos modelos fracos. Esse método é fundamentado no princípio de iterativamente ajustar uma série de modelos em versões modificadas dos dados de treinamento, priorizando os erros cometidos pelos modelos anteriores. Em cada iteração, o modelo ajustado tenta corrigir os erros dos modelos anteriores, e as previsões finais são obtidas por meio de uma combinação

ponderada das saídas individuais desses modelos, com maior peso atribuído aos preditores que corrigem melhor os erros (SCHAPIRE, 2003).

A ideia central por trás do *Boosting* é que a combinação das previsões de vários modelos simples pode superar a precisão de um único modelo complexo. Esse processo, que enfatiza a correção dos erros por meio de modelos sequenciais, frequentemente resulta em um desempenho mais robusto, pois os erros de predição dos aprendizes fracos são suavizados quando agregados. Além disso, o *Boosting* é particularmente eficaz para problemas complexos, onde as interações não-lineares entre variáveis são difíceis de capturar por modelos isolados. A abordagem de ponderação das previsões, com base nos erros cometidos, permite que o modelo final seja altamente preciso e generalizável, muitas vezes superando modelos complexos que tentam capturar todos os padrões do conjunto de dados em uma única iteração (SCHAPIRE, 2003).

O algoritmo de *Boosting* com árvores de regressão é um processo iterativo que busca otimizar a previsão de um modelo preditivo combinando múltiplas árvores de regressão ajustadas em sequência (Figura 17). Cada árvore subsequente tenta corrigir os erros cometidos pelas árvores anteriores, resultando em uma previsão mais precisa e robusta. O processo de aprendizado pode ser detalhado pelos seguintes passos principais, conforme descrito por MICHELIS (2012):

Figura 17 – Construção de um modelo de boosting



Fonte: adaptado (OMARZAI, 2024)

1. **Inicialização:** O processo começa com uma previsão constante inicial, dada pela média das observações $\bar{y_i}$ no conjunto de treinamento. Esta previsão inicial serve como valor de referência para o modelo.

$$F_0(x_i) = \bar{y_i} \quad \text{para todo} \quad x_i \tag{44}$$

2. **Iteração ($m = 1$ até M):** Para cada iteração m, os seguintes passos são executados:

   a) **Cálculo do erro de previsão:** Para cada observação x_i do conjunto de treinamento, calcula-se o erro de previsão residual $\tilde{y_i}$, que é a diferença entre a observação real y_i e a previsão fornecida pelo modelo anterior $F_{m-1}(x_i)$:

$$\tilde{y_i} = y_i - F_{m-1}(x_i) \tag{45}$$

   b) **Extração da subamostra aleatória:** Uma subamostra aleatória dos dados de treinamento é extraída para garantir a diversidade e melhorar a generalização do modelo.

   c) **Ajuste de uma nova árvore de regressão:** Uma nova árvore de regressão é ajustada aos resíduos (erros) da subamostra extraída, ou seja, a árvore tenta modelar as diferenças entre as previsões anteriores e os valores reais.

   d) **Atualização da previsão do modelo:** A previsão do modelo final é atualizada somando a previsão da árvore ajustada $f_m(X)$ com a previsão do modelo anterior $F_{m-1}(X)$:

$$F_m(X) = F_{m-1}(X) + f_m(X) \tag{46}$$

3. **Modelo final:** O modelo final $F_M(X)$ é a soma de todas as árvores ajustadas ao longo do processo iterativo. Cada árvore subsequente ajusta os erros das anteriores, resultando em uma combinação otimizada de predições.

$$F_M(X) = F_0(X) + \sum_{m=1}^{M} f_m(X) \tag{47}$$

Esse processo de aprendizado em etapas permite que o modelo de Boosting com Árvores de Regressão melhore progressivamente sua precisão, corrigindo os erros cometidos em cada etapa, até que o modelo final atinja uma previsão robusta e altamente ajustada aos dados de treinamento MICHELIS (2012).

Embora o algoritmo de *Boosting* com Árvores de Regressão seja uma técnica extremamente eficaz, ele apresenta uma vulnerabilidade ao sobreajuste *(overfitting)*, uma vez que a

minimização do erro de treinamento ocorre de forma progressiva a cada iteração. Esse comportamento pode levar o modelo a se ajustar excessivamente aos dados de treinamento, resultando em uma baixa capacidade de generalização para novos dados. Para mitigar o risco de sobreajuste, é comum a introdução de um parâmetro de regularização, denominado ν, que controla a magnitude das atualizações feitas a cada iteração, limitando assim a quantidade de ajuste que o modelo pode fazer em cada ciclo de treinamento MICHELIS (2012)

A regularização com ν ajuda a equilibrar o aprendizado entre a redução do erro de treinamento e a manutenção da capacidade de generalização do modelo. Ao ajustar esse parâmetro, busca-se controlar a influência de cada árvore adicional no modelo final, evitando que o ajuste excessivo às flutuações do conjunto de dados leve a um modelo excessivamente complexo e propenso ao sobreajuste MICHELIS (2012).

Adicionalmente, técnicas de validação cruzada são frequentemente empregadas para determinar o número ideal de árvores a serem incluídas no modelo final. A validação cruzada permite avaliar a performance do modelo em diferentes subconjuntos dos dados de treinamento, ajudando a identificar a quantidade de árvores que otimiza a precisão sem comprometer a capacidade de generalização. Essa abordagem assegura que o número de árvores no modelo seja equilibrado, evitando a inclusão de árvores em excesso, o que poderia resultar em uma modelagem excessivamente ajustada e, portanto, menos robusta para novos dados MICHELIS (2012).

2.7.1 Objetivo de aprendizagem regularizado

Como descrito por CHEN e GUESTRIN (2016), um modelo de boosting com árvores de decisão, ou mais especificamente, um conjunto de árvores de regressão, tem como objetivo prever a saída $\hat{y}_i$ de maneira aditiva, utilizando uma combinação ponderada de várias funções preditivas. O modelo é composto por K funções aditivas $f_k(x_i)$, cada uma representando a previsão de uma árvore de regressão, e a soma das previsões de todas as árvores resulta na previsão final $\hat{y}_i$, representada pela Equação 48.

$$\hat{y}_i = \eta(x_i) = \sum_{k=1}^{K} f_k(x_i), \quad f_k \in F \tag{48}$$

Aqui, cada $f_k(x_i)$ é uma árvore de regressão, que pode ser vista como uma estrutura de decisão $q(x_i)$ (indicando a folha da árvore) associada a um valor de pontuação w. O treinamento do modelo de boosting visa minimizar uma função de perda regularizada, dada pela Equação 49.

$$L(\eta) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k) \tag{49}$$

No qual o primeiro termo $l(\hat{y}_i, y_i)$ é a função de perda, que mede a discrepância entre a previsão $\hat{y}_i$ e o valor real y_i, e o segundo termo $\Omega(f_k)$ atua como uma penalização à complexidade das árvores de regressão. A introdução desse termo de regularização tem como principal objetivo evitar o overfitting, incentivando modelos mais simples e com maior capacidade de generalização (CHEN; GUESTRIN, 2016).

A adição de um termo de regularização, como o que é implementado no modelo Regularized Greedy Forest (RGF), oferece benefícios adicionais, como a melhoria no desempenho do modelo, facilitando a paralelização do processo de treinamento e minimizando a tendência de ajuste excessivo aos dados de treinamento. Esse mecanismo de regularização, portanto, não só aprimora a robustez do modelo, mas também contribui para sua eficiência computacional, permitindo que ele seja treinado de maneira mais rápida e escalável, enquanto mantém a capacidade de generalização (CHEN; GUESTRIN, 2016).

2.7.2 *Gradient Tree Boosting* (impulsionamento por gradiente em árvore)

Como descrito no artigo de CHEN e GUESTRIN (2016), no gradient tree boosting, a adição de cada nova árvore de decisão é realizada de forma gradual, ou seja, a cada iteração é adicionada uma árvore f_t que visa minimizar a função de perda, ajustando os erros cometidos pelas árvores anteriores. O processo de atualização do modelo pode ser formalizado pela Equação 50.

$$L(t) = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t) \tag{50}$$

Onde $\hat{y}_i^{(t-1)}$ é a previsão da iteração anterior, e $f_t(x_i)$ representa a árvore adicionada na iteração t. A função de perda é minimizada utilizando uma aproximação de segunda ordem, que incorpora tanto o gradiente quanto a Hessiana da função de perda, conforme a Equação 51.

$$L(t) \approx \sum_{i=1}^{n} \left[l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i) \right] + \Omega(f_t) \tag{51}$$

Neste contexto, g_i e h_i são o gradiente e a Hessiana da função de perda, respectivamente, avaliados na iteração anterior. O gradiente g_i reflete a taxa de variação da função de perda em relação à previsão, enquanto a Hessiana h_i fornece uma medida da curvatura dessa variação, permitindo ao modelo ajustar as previsões de forma mais eficiente. Com isso, a árvore $f_t(x_i)$ é

construída de maneira a reduzir o erro residual, utilizando as informações de primeiro e segundo ordem derivadas da função de perda. A regularização, representada por $\Omega(f_t)$, penaliza a complexidade das árvores, evitando o sobreajuste e garantindo um modelo mais robusto (CHEN; GUESTRIN, 2016).

Este processo iterativo, ao combinar gradientes e Hessianas, proporciona uma atualização eficaz do modelo, permitindo que o Gradient Tree Boosting ajuste as previsões de forma mais precisa e controlada, minimizando erros e melhorando o desempenho do modelo em termos de generalização (CHEN; GUESTRIN, 2016).

2.7.3 Influência Relativa

A influência relativa de uma variável é uma métrica que quantifica a importância de uma variável dentro de um modelo de boosting, indicando o impacto dessa variável no desempenho do modelo. Essa medida é calculada com base na melhoria no ajuste do modelo obtida ao dividir os dados de treinamento em diferentes pontos, de acordo com a variável em questão. O valor da influência relativa J_j^2 de uma variável x_j é definido como a soma das melhorias empíricas geradas ao realizar divisões nos pontos correspondentes a essa variável, dada pela Equação 52.

$$J_j^2 = \sum_{\text{splits em } x_j} I_t^2 \tag{52}$$

Onde I_t^2 representa a melhoria da função de perda obtida pela divisão dos dados no ponto t com base na variável x_j. Essa melhoria reflete o impacto da variável x_j na redução do erro do modelo ao separar os dados em diferentes subconjuntos, contribuindo para um melhor ajuste das previsões.

A média dessa melhoria ao longo de todas as árvores geradas no processo de boosting fornece uma avaliação robusta e confiável da importância de cada variável. Quanto maior a influência relativa de uma variável, maior será sua contribuição para o modelo, indicando que ela desempenha um papel crucial na determinação das previsões do modelo final. Dessa forma, a análise da influência relativa permite identificar quais variáveis são mais determinantes para o desempenho do modelo, facilitando a interpretação e a explicabilidade dos resultados do boosting (CHEN; GUESTRIN, 2016).

2.7.4 XGBoost

O XGBoost (*Extreme Gradient Boosting*) é uma implementação eficiente e altamente escalável de um algoritmo de boosting baseado em árvores de decisão. Introduzido por CHEN e GUESTRIN (2016), o XGBoost é projetado para otimizar o desempenho de modelos preditivos e tem se tornado uma escolha predominante em tarefas de aprendizado supervisionado devido à sua notável capacidade de lidar com grandes volumes de dados e a eficiência computacional em problemas complexos de modelagem. O XGBoost trouxe uma série de inovações significativas em relação aos métodos tradicionais de boosting, oferecendo melhorias substanciais tanto na precisão quanto na velocidade de treinamento.

CAN R.and KOCAMAN e GÖKCEOLU (2021) avaliaram o desempenho do XGBoost na elaboração de mapas de suscetibilidade a deslizamentos na bacia superior da Barragem Ataturk, na Turquia, destacando a litologia, altitude e o índice de umidade topográfica como fatores-chave, com uma boa precisão de classificação. SHI *et al.* (2022) utilizaram o XGBoost para prever a estabilidade de represas de deslizamento de terra com base em um banco de dados global de 2783 casos, demonstrando maior precisão em comparação com métodos convencionais e aplicando o modelo em casos como Meilonggou, Usoi e Mount St. Helens. ZHANG *et al.* (2021) desenvolveram um modelo para monitoramento da percolação em barragens de concreto, combinando XGBoost com Hybrid Grey Wolf Optimization (HGWO), alcançando excelente desempenho na previsão da percolação e identificando a influência da defasagem entre o nível de água a montante e a precipitação.

2.7.4.1 Modelo de Boosting

O XGBoost, assim como outros métodos de boosting, segue o princípio de que um modelo robusto pode ser construído a partir da combinação de múltiplos modelos fracos, como as árvores de decisão. O processo de construção do modelo envolve uma abordagem sequencial, onde cada árvore subsequente corrige os erros residuais das árvores anteriores. Formalmente, a previsão final do modelo $f(x)$ é dada pela soma ponderada das previsões das árvores individuais. Essa expressão é representada pela Equação 53.

$$f(x) = \sum_{k=1}^{K} \alpha_k h_k(x) \tag{53}$$

Onde $f(x)$ é a previsão final do modelo, que combina os resultados das K árvores de decisão, α_k é o peso atribuído à k-ésima árvore no modelo final, esse peso é ajustado de forma

a minimizar o erro global, e $h_k(x)$ é a k-ésima árvore de decisão, que aprende a corrigir os erros das árvores anteriores.

Cada árvore no XGBoost é treinada para reduzir o erro de previsão, em particular o resíduo da soma ponderada das previsões das árvores anteriores, um processo que é frequentemente realizado utilizando o gradiente descendente. A sequência de árvores resulta em um modelo final que é uma combinação de previsões de árvores anteriores, com pesos otimizados para reduzir o erro (CHEN; GUESTRIN, 2016).

2.7.4.2 Inovações e melhorias no XGBoost

O XGBoost é caracterizado por várias inovações matemáticas e computacionais que o diferenciam de abordagens convencionais de boosting, tornando-o uma ferramenta poderosa em modelos de aprendizado em grande escala (CHEN; GUESTRIN, 2016).

- **Regularização:** o XGBoost introduz uma regularização explícita no modelo, que ajuda a evitar o overfitting ao penalizar modelos excessivamente complexos.
- **Otimização de segunda ordem:** o algoritmo utiliza uma abordagem de otimização de segunda ordem, que é mais eficiente na convergência do modelo em comparação com as abordagens de primeira ordem, como o gradient boosting.
- **Aprimoramento na construção das árvores:** o XGBoost implementa um método eficiente para construir as árvores, utilizando uma heurística baseada na otimização de uma função de custo específica.
- **Paralelização e processamento distribuído:** o XGBoost foi projetado para ser altamente eficiente, permitindo treinamento paralelo e distribuído, o que é essencial para lidar com grandes volumes de dados.

Essas melhorias tornam o XGBoost uma escolha popular para problemas de classificação e regressão em grandes conjuntos de dados, com um desempenho superior em muitas situações (CHEN; GUESTRIN, 2016).

3 METODOLOGIA

Este capítulo apresenta a metodologia adotada neste estudo, que abrange desde a modelagem numérica tridimensional do fluxo em barragens de terra até a aplicação de técnicas de aprendizado de máquina para a predição de parâmetros hidráulicos. Inicialmente, descreve-se a construção do banco de dados a partir de simulações computacionais realizadas por meio do Método dos Elementos Finitos (MEF), utilizando o software GeoStudio Seep/W 3D. Essa etapa enfatiza a análise da interação solo-estrutura e o desempenho do dispositivo de controle de percolação, denominado "abraço", na mitigação de fluxos preferenciais. Na sequência, são apresentados os parâmetros geométricos, hidráulicos e geotécnicos considerados nas modelagens, bem como as simplificações adotadas para garantir a eficiência computacional sem comprometer a representatividade dos resultados.

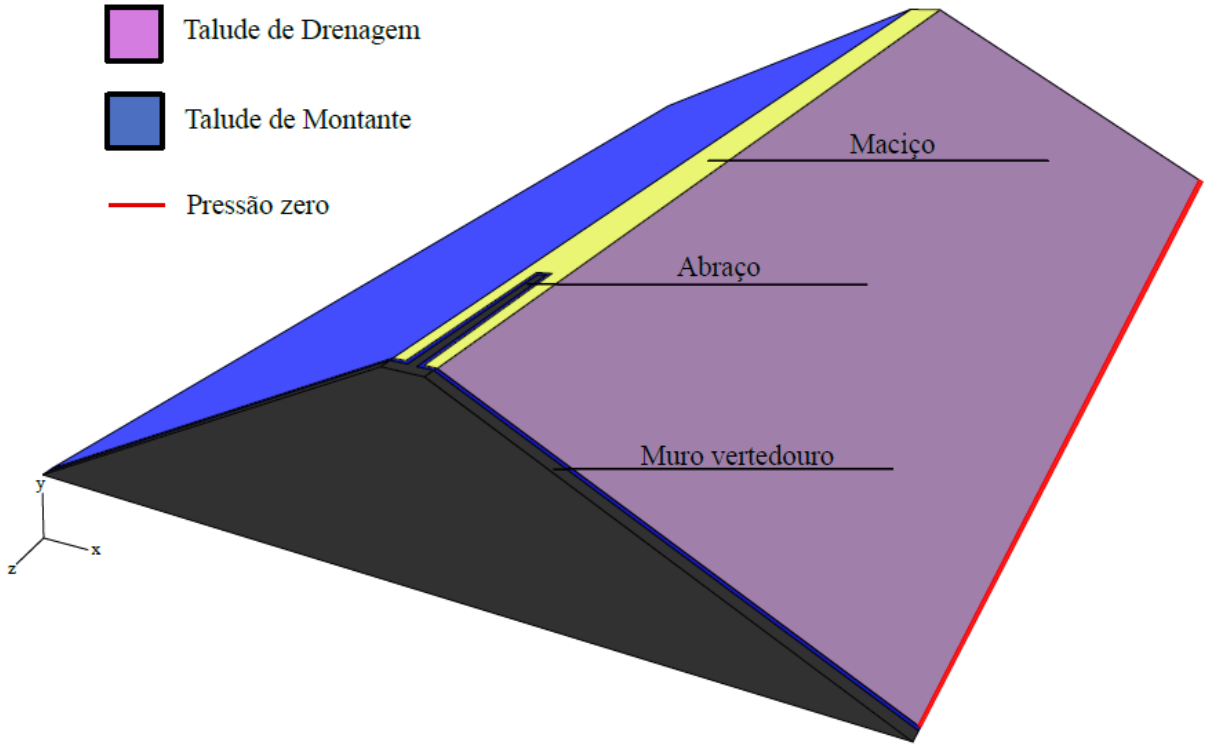
A segunda etapa da metodologia contempla a aplicação de algoritmos de aprendizado de máquina, incluindo Redes Neurais Artificiais, Árvores de Decisão, Random Forest e XGBoost. São detalhados os procedimentos de definição de hiperparâmetros, particionamento dos dados, métricas de desempenho e estratégias de validação cruzada adotadas. Por fim, realiza-se uma análise comparativa entre os modelos, fundamentada em gráficos de desempenho, distribuição dos resíduos e curvas de aprendizado, com o objetivo de identificar os algoritmos mais robustos para a predição do comprimento ideal do dispositivo "abraço" (L_a). A reprodutibilidade dos experimentos é assegurada por meio de scripts desenvolvidos em *Python*, disponibilizados nos apêndices deste trabalho.

3.1 Montagem do banco de dados a partir de simulações numéricas 3D do fluxo em barragens de terra

A etapa inicial do estudo consistiu em uma análise detalhada de barragens em perspectiva tridimensional (Figura 18), utilizando o Método dos Elementos Finitos (MEF). O MEF foi aplicado por meio da ferramenta computacional Seep/W 3D, do software GeoStudio (SEEQUENT, 2024), que permitiu a resolução numérica das equações de percolação e fluxo.

A adoção de modelagens tridimensionais mostrou-se imprescindível devido à influência determinante do dispositivo de controle de percolação, conhecido como abraço, na trajetória e no comportamento do fluxo. O abraço converte um problema originalmente bidimensional em

Figura 18 – Modelagem tridimensional para análise de fluxo com representação das variáveis envolvidas



Fonte: Autora (2025)

uma análise tridimensional ao alterar substancialmente a direção e a distribuição do fluxo na interface da barragem.

Essa modificação promove uma redistribuição da energia hidráulica ao longo de um percurso ampliado, contribuindo para a dissipação gradual da energia e a redução do gradiente hidráulico nas proximidades da interface. Como resultado, reduz-se significativamente o risco de processos de instabilidade, como a erosão interna, que representa uma ameaça crítica à integridade e estabilidade da barragem.

As simulações tridimensionais desempenharam um papel fundamental ao suprir a ausência de dados experimentais, fornecendo informações cruciais para a análise do comportamento hidráulico na interface solo-estrutura. Por meio dessas simulações, foi possível avaliar detalhadamente as interações entre o solo e a estrutura sob diferentes cenários hidrológicos e condições geotécnicas. Os resultados obtidos foram determinantes para validar o desempenho do abraço, com foco na sua eficácia na redução do gradiente hidráulico.

3.1.1 Parâmetros analisados

A presente análise teve como objetivo avaliar as condições de percolação na interface solo/estrutura de barragens, com foco na eficiência do elemento abraço. Foram consideradas três geometrias, com alturas de 30 m, 50 m e 70 m, representativas de diferentes escalas de projetos. Para cada configuração, parâmetros geométricos, geotécnicos e hidráulicos foram definidos de modo a abranger uma ampla gama de variáveis que influenciam a redução do gradiente hidráulico na interface solo/estrutura e sua interação com o dispositivo. A partir de simulações numéricas tridimensionais, gerou-se um conjunto de 9.261 dados brutos, sem tratamento prévio, que representam diversas configurações geométricas e hidráulicas. A Tabela 2 sumariza as observações, detalhando as variáveis consideradas em cada cenário.

As variações geométricas incluíram a largura da crista, a inclinação dos taludes e a presença de elementos de reforço, como enrocamentos e dispositivos drenantes. O abraço foi parametrizado em termos de largura e comprimento, permitindo avaliar sua eficácia no controle da percolação, com foco na redistribuição dos gradientes hidráulicos.

As fundações das barragens foram modeladas com substratos de diferentes profundidades e coeficientes de permeabilidade, permitindo uma análise aprofundada do impacto das condições geotécnicas e das intervenções de tratamento no comportamento do fluxo subterrâneo. A trincheira de vedação foi incorporado como um dos dispositivos de controle, com profundidade variável, ajustada de acordo com as características da barragem.

A parametrização adotada gerou um conjunto abrangente de cenários computacionais, possibilitando a análise detalhada da influência de cada variável no desempenho hidráulico e na segurança estrutural das barragens. O objetivo principal foi quantificar a redução percentual do gradiente hidráulico, comparando cenários com e sem o dispositivo de controle. Essa abordagem permitiu avaliar a efetividade do abraço na mitigação de riscos associados à percolação, fornecendo subsídios técnicos para seu dimensionamento otimizado em diferentes condições geotécnicas.

3.1.2 Modelagem da interação solo/estrutura e condições hidrológicas e geotécnicas

A fase inicial do estudo consistiu em uma análise tridimensional das barragens utilizando o Método dos Elementos Finitos (MEF) para resolver numericamente as equações governantes do fluxo hidráulico. Foram geradas diversas malhas de elementos finitos, compostas por diferentes números de elementos e nós, adaptadas às múltiplas geometrias e análises realizadas.

Tabela 2 – Parâmetros geométricos para diferentes tipos de barragem

Parâmetro	Barragem de 30 m (200 m extensão)	Barragem de 50 m (333 m extensão)	Barragem de 70 m (467 m extensão)
Largura da crista	6, 10, 14, 18, 22 m	10, 22, 36 m	14, 31, 50 m
Largura do abraço	1, 2, 3 m	1.67, 3.33, 5 m	2.34, 4.66, 7 m
Inclinação dos taludes	1/1, 1/2, 1/2.5, 1/3	1/1, 1/2, 1/2.5, 1/3	1/1, 1/2, 1/2.5, 1/3
Enrocamento no pé do talude	10, 15, 20, 25, 30 m	17, 25, 33 m	24, 35, 46 m
Tapete drenante com enrocamento	15, 20, 25, 30, 35, 40 m	25, 33, 42 m	35, 46, 59 m
Fundações	Substrato permeável de 25 m	Substrato permeável de 42 m	Substrato permeável de 59 m
Profundidade do cutoff	5, 8 e 10 m	8, 13 e 17 m	11, 18 e 24 m
Comprimento do abraço	10, 30, 50, 70, 90 m	50, 100, 117 m	70, 140, 187 m

Fonte: Autora (2025)

O tipo de elemento finito empregado foi o tetraédrico, que permitiu a discretização eficiente do domínio, especialmente em regiões com geometrias irregulares ou interfaces solo/estrutura. As condições de contorno incluíram a carga hidráulica a montante, que variou conforme as barragens testadas, e o talude de vazão zero, representando a impermeabilidade nas laterais do domínio. Essa abordagem permitiu uma representação do comportamento hidráulico e das características específicas de cada cenário analisado.

Com o intuito de otimizar a modelagem e facilitar a análise da interação entre a interface solo/estrutura, adotou-se a suposição de homogeneidade para o maciço das barragens simuladas. Essa abordagem foi escolhida para evitar a complexidade adicional de heterogeneidades no solo que poderiam dificultar a interpretação dos efeitos específicos do abraço no desempenho da barragem. Embora essa suposição represente uma simplificação das condições reais, ela permite um entendimento mais claro das interações fundamentais entre o solo e a estrutura.

Os parâmetros hidrológicos e geotécnicos para modelar o fluxo em solos não saturados foram obtidos com o software RETEC (RETEC, 2009), utilizando o modelo de Van Genuchten pela sua precisão em representar a curva de retenção de água em função da tensão matricial (GENUCHTEN, 1980). Esse modelo requer parâmetros como o teor de umidade na saturação, o teor de umidade residual, o parâmetro α (relacionado à pressão de entrada de ar), o expoente n (que define a forma da curva de retenção) e a condutividade hidráulica na saturação. Esses parâmetros permitem descrever a relação entre a sucção matricial e o teor de umidade, representando a retenção de água no solo e definindo a função de condutividade hidráulica não saturada,

essenciais para a análise do fluxo em meios porosos.

Na modelagem numérica, as descontinuidade geradas na interface solo/estrutura, como variações na permeabilidade e na condutividade hidráulica, foram representadas por uma camada de espessura reduzida, com permeabilidade 10 vezes maior do que a do maciço principal. Essa representação buscou reproduzir as características hidráulicas diferenciadas dessa interface, permitindo avaliar os padrões de redistribuição do fluxo. Essa abordagem forneceu subsídios para compreender os impactos das zonas de descontinuidade nos gradientes hidráulicos e, consequentemente, na estabilidade global da barragem.

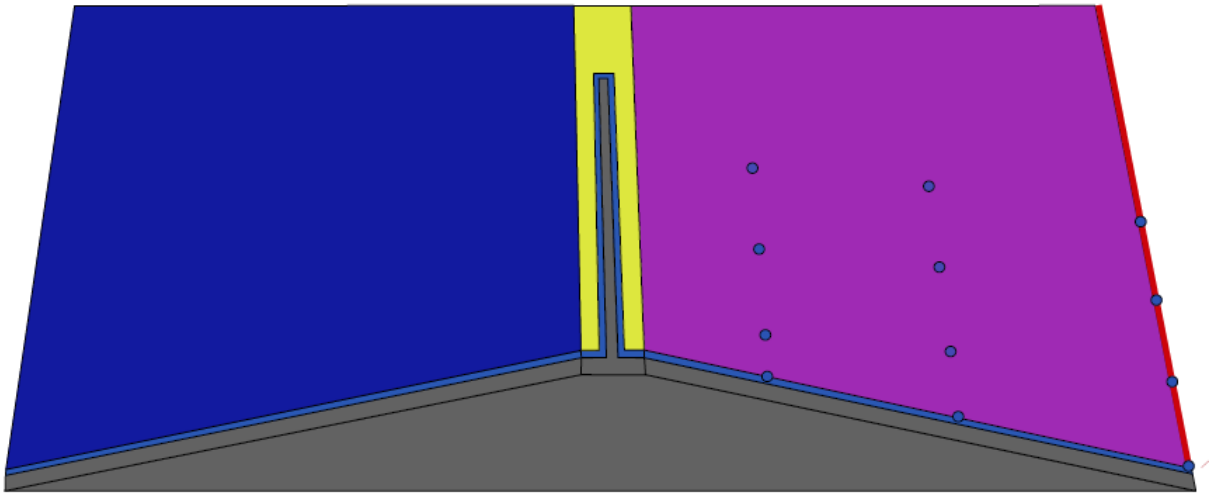
As condições de contorno adotadas nas simulações foram cuidadosamente definidas para reproduzir os fenômenos hidráulicos característicos das condições operacionais de barragens. A condição de impermeabilidade foi aplicada para impedir o fluxo de água através das superfícies designadas, enquanto a condição de superfície livre, determinada pelo nível do reservatório a montante, estabelece a altura da lâmina d'água. Essas condições são cruciais, pois regulam as pressões hidráulicas atuantes na estrutura, influenciando diretamente o comportamento do fluxo e a distribuição de tensões na barragem.

3.1.3 Análise dos gradientes hidráulicos

A metodologia adotada nesta dissertação envolveu a seleção de três localizações distintas dentro da barragem, com a definição de quatro pontos de análise em cada uma dessas localizações. O objetivo principal foi avaliar o comportamento dos gradientes hidráulicos, com foco na comparação entre a barragem com e sem o dispositivo de controle de percolação, denominado abraço, visando quantificar a redução percentual desses gradientes.

As localizações selecionadas para as análises, ilustradas na Figura 8, foram escolhidas de maneira estratégica, com o objetivo de capturar as variações nos gradientes hidráulicos em diferentes pontos ao longo da barragem, abrangendo tanto as zonas mais distais quanto as proximidades da interface entre o solo e a estrutura. Particularmente, a avaliação foi focada em pontos situados próximos à jusante da barragem, na região crítica da interface solo/estrutura (Figura 18), dada a sua maior vulnerabilidade a gradientes hidráulicos elevados e à concentração de esforços hidráulicos, como evidenciado por SOUSA (2013).

Figura 19 – Pontos analisados nas análises de fluxo



Fonte: Autora (2025)

3.1.4 Simplificações consideradas no modelo

As simplificações adotadas nas etapas finais de modelagem foram justificadas pela análise dos resultados numéricos e pela necessidade de otimizar o tempo de processamento computacional, sem comprometer a robustez e a confiabilidade dos modelos. A remoção de variáveis como a anisotropia do solo e as variações nas dimensões do enrocamento e do cutoff não afetou a representatividade dos resultados, uma vez que esses parâmetros apresentaram impacto insignificante nas respostas globais avaliadas, como os gradientes hidráulicos e o fluxo através da interface solo/estrutura.

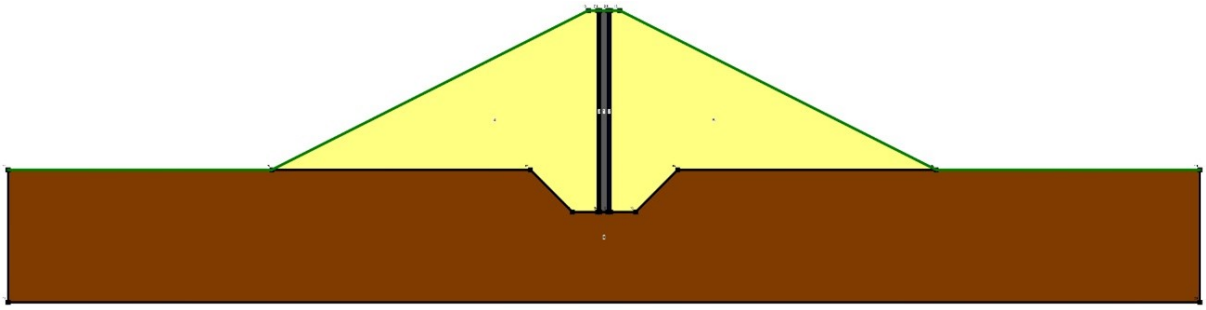
A anisotropia do solo, que descreve a variação da permeabilidade em diferentes direções, é um parâmetro frequentemente relevante para o comportamento hidráulico em diversas configurações de barragens. No entanto, os resultados obtidos nas simulações indicaram que sua influência foi mínima no contexto analisado. A comparação dos cenários com e sem a consideração da anisotropia demonstrou que os efeitos desse parâmetro não tiveram impacto significativo no comportamento hidráulico na interface solo/estrutura, independentemente da presença ou ausência do abraço.

Diante dessa constatação, decidiu-se desconsiderar a variação da anisotropia nas análises, mantendo-a constante em todos os cenários avaliados. Para representar as condições típicas do método construtivo de barragens, foi adotada uma razão de anisotropia de 0.1, onde k_h e k_v representam, respectivamente, a permeabilidade horizontal e vertical, de modo que $\frac{k_h}{k_v} = 0.1$.

As análises preliminares investigaram a influência das dimensões do enrocamento no pé do talude e das configurações do cutoff na fundação da barragem, com foco na redução de

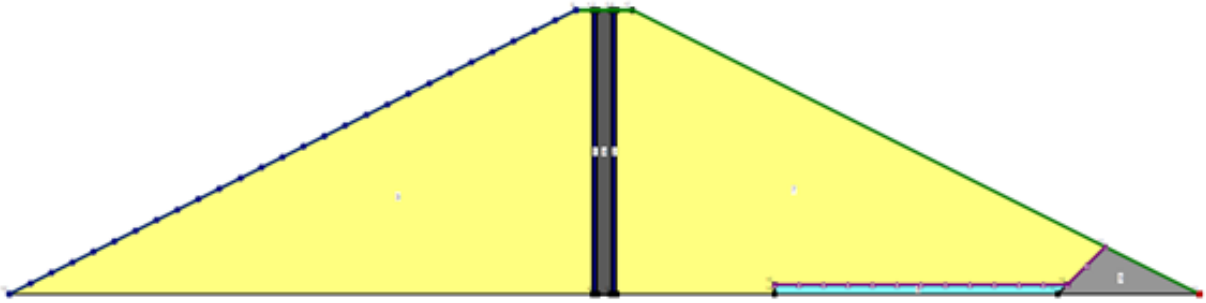
gradientes hidráulicos e no controle da percolação na interface solo/estrutura. As modelagens tridimensionais foram conduzidas conforme os parâmetros representados, de forma esquemática, nas Figuras 20 e 21, que ilustram cortes bidimensionais dos modelos utilizados. Os resultados mostraram que, em ambos os casos, a presença do enrocamento e do cutoff foi o fator predominante, independentemente de variações nas dimensões, profundidades ou localizações analisadas. Com base nessas conclusões, as etapas finais de modelagem foram simplificadas, considerando apenas a presença ou ausência desses elementos nos modelos numéricos. Essa abordagem racionalizou as análises e direcionou o foco para a eficácia desses dispositivos no controle hidráulico e estrutural.

Figura 20 – Modelo esquemático bidimensional da trincheira de vedação



Fonte: Autora (2025)

Figura 21 – Modelo esquemático bidimensional do enrocamento e tapete drenante



Fonte: Autora (2025)

Os parâmetros finais utilizados como variáveis de entrada e saída para a análise estão detalhados na Tabela 3, incluindo suas respectivas unidades e descrições.

3.2 Definição dos hiperparâmetros em modelos preditivos: redes neurais, árvore de decisão, random forest e XGBoost

Os procedimentos computacionais empregados para o treinamento, validação e avaliação dos modelos preditivos baseados em *Redes Neurais Artificiais*, *Árvores de Decisão*, *Flores-*

tas Aleatórias e *XGBoost* estão detalhados nos Apêndices A, B, C e D, respectivamente. Cada apêndice compreende o código-fonte completo em linguagem `Python`, estruturado em seções que abordam desde a importação das bibliotecas necessárias e o carregamento dos dados brutos até o pré-processamento, a definição dos modelos com seus hiperparâmetros, os processos iterativos de treinamento e avaliação, e a visualização dos resultados de desempenho. A execução desses scripts requer um ambiente de programação `Python` configurado com as bibliotecas científicas e de aprendizado de máquina pertinentes, incluindo, mas não se limitando a, `numpy`, `pandas`, `sklearn` (`scikit-learn`), `matplotlib` e `seaborn`, bem como a biblioteca `xgboost` para o modelo específico.

O fluxo de trabalho para a utilização dos códigos inicia-se com o carregamento dos dados brutos a partir de um arquivo no formato `.csv`, identificado nos scripts como `dados_10.csv`. É crucial que este arquivo esteja acessível no ambiente de execução e contenha a estrutura de colunas esperada pelos scripts, correspondente às variáveis independentes e dependente utilizadas no estudo. A etapa subsequente envolve o pré-processamento dos dados, que pode incluir a remoção de entradas com valores nulos e a normalização ou escalonamento dos dados de entrada e saída, conforme implementado no Apêndice A para as RNA, visando otimizar a performance dos algoritmos de aprendizado.

Após o pré-processamento, procede-se à fase de treinamento e avaliação do modelo. Os códigos nos apêndices correspondentes implementam laços de repetição para explorar diferentes combinações de hiperparâmetros predefinidos para cada tipo de modelo (como número de estimadores, profundidade máxima, funções de ativação, otimizadores, entre outros). Para cada combinação testada, o modelo é treinado com o conjunto de dados de treinamento e avaliado utilizando métricas de desempenho relevantes, como o Erro Médio Absoluto (MAE), a Raiz do Erro Quadrático Médio (RMSE), o Coeficiente de Determinação (R^2) e o Erro Percentual Absoluto Médio (MAPE). Os resultados de cada avaliação são armazenados para posterior análise comparativa, permitindo a identificação das configurações de hiperparâmetros que resultaram no melhor desempenho preditivo. As seções finais de cada apêndice contêm código para a geração de visualizações gráficas que auxiliam na interpretação dos resultados e na análise do comportamento do modelo.

A segunda etapa do estudo consistiu na aplicação de diferentes técnicas de aprendizado de máquina, com o objetivo de estabelecer funções matemáticas que correlacionem as variáveis de entrada, listadas na Tabela 3, com a variável de saída L_a, que representa o comprimento

ideal do dispositivo de controle de percolação. Esses modelos foram treinados para capturar relações complexas entre as variáveis de entrada e a variável de saída, utilizando abordagens supervisionadas que permitem a generalização e a predição de L_a.

Para a implementação dos modelos e o tratamento dos dados, foi utilizada a linguagem de programação Python, executada por meio da plataforma Google Colab, a qual oferece um ambiente interativo, baseado em nuvem, que facilita a execução de códigos e o compartilhamento dos resultados. Essa escolha permitiu maior flexibilidade na experimentação com diferentes bibliotecas de aprendizado de máquina, como scikit-learn, TensorFlow, Keras e XGBoost, além de garantir reprodutibilidade e acessibilidade aos experimentos computacionais realizados.

Tabela 3 – Parâmetros das variáveis analisadas

Tipo	Símbolo	Descrição	Mínimo	Máximo	Unidade
X1	km	Permeabilidade do solo do maciço	8.25E-05	9.50E-09	m/s
X2	H	Altura da barragem	30	70	m
X3	im	Inclinação do talude de montante	01/mar	01/jan	°
X4	i j	Inclinação do talude de jusante	01/mar	01/jan	°
X5	Bc	Largura da crista	4	36	m
X6	zs	Cota da soleira	17.33	60.67	m
X7	Wa	Largura do abraço	1	7	m
X8	Lt	Comprimento do tapete horizontal drenante	15	59	m
X9	E	Presença de enrocamento	0	1	-
X10	k f	Permeabilidade da fundação	0.001	8.25E-05	m/s
X11	T	Presença de trincheira de vedação (cutoff)	0	1	-
X12	X	Coordenada X	40	432	m
X13	Y	Coordenada Y	0	87	m
X14	Z	Coordenada Z	44	467	m
X15	Rg	Redução do gradiente hidráulico	0	0.98	%
Y1	La	Comprimento do abraço	7	210	m

Fonte: Autora (2025)

Os códigos completos utilizados nesta seção serão apresentados nos Apêndices A, B, C e D, proporcionando a possibilidade de replicação e validação do procedimento descrito. Os hiperparâmetros dos modelos de redes neurais, árvores de decisão, florestas aleatórias e *XGBoost*

desempenham um papel crucial na otimização do desempenho preditivo e na melhoria da capacidade de generalização dos modelos. A seguir, apresentam-se os principais hiperparâmetros considerados e avaliados para cada uma dessas abordagens.

Matematicamente, cada modelo pode ser expresso de forma genérica pela Equação 54:

$$L_a = f(\mathbf{X}; \theta) \tag{54}$$

onde $\mathbf{X}$ é o vetor de entrada composto pelas 15 variáveis descritas na Tabela 3, e θ representa o conjunto de parâmetros específicos de cada modelo, ajustados durante o processo de treinamento. Esses parâmetros variam conforme a técnica utilizada, podendo incluir pesos, vieses, regras de decisão ou outros componentes, dependendo da abordagem adotada. O objetivo foi explorar diferentes métodos para encontrar a melhor correlação entre as variáveis de entrada e a variável de saída L_a.

3.2.1 Organização dos Dados

Os dados utilizados neste estudo foram disponibilizados em formato `.csv` e, como primeira etapa do processo de análise, foram submetidos a um processo de pré-tratamento. Para realizar essa tarefa, foram empregadas as bibliotecas `Pandas` e `NumPy` (OLIPHANT, 2006; MCKINNEY, 2010), amplamente utilizadas na manipulação e análise de dados. Esse processo foi essencial para estruturar o conjunto de dados de forma apropriada, garantindo que estivesse adequado para as análises subsequentes. d A primeira ação realizada foi a identificação de valores nulos no conjunto de dados. Para isso, utilizou-se a função `.isnull()` da biblioteca `Pandas`, que permitiu detectar qualquer célula que não contivesse dados válidos. A presença de valores nulos é crítica, pois pode introduzir vieses significativos nas análises, prejudicando a acurácia dos resultados.

3.2.2 Redes Neurais Artificiais

Nesta investigação, o modelo adotado foi baseado em Redes Neurais Artificiais (RNA) do tipo perceptron multicamadas, implementadas por meio do regressor `MLPRegressor`, disponível na biblioteca `scikit-learn`, originalmente desenvolvida por (COURNAPEAU, 2007) e posteriormente aprimorada por (PEDREGOSA *et al.*, 2011). O MLP é uma rede neural multicamadas projetada para modelar relações não lineares complexas entre variáveis. Sua estrutura permite o ajuste iterativo dos pesos sinápticos e valores de *bias* durante o processo de treina-

mento, tornando-o uma ferramenta poderosa para problemas de predição em sistemas complexos.

Todo o processo, que abrange o pré-processamento, a definição das combinações de hiperparâmetros, o treinamento e a avaliação, está detalhado no Apêndice A, onde é possível acessar o código completo.

3.2.2.1 Divisão e Normalização dos Dados

A normalização dos dados é uma etapa fundamental na preparação para o treinamento de modelos de aprendizado de máquina. Com base nas informações obtidas durante a modelagem numérica, a técnica de normalização foi aplicada para ajustar as variáveis independentes a uma mesma escala, eliminando discrepâncias de magnitude que poderiam comprometer o desempenho dos algoritmos. Esse procedimento visa reduzir a influência de variações extremas, aprimorando tanto a eficácia quanto a precisão dos modelos desenvolvidos (IPNET, 2023).

Para a normalização, utilizou-se o método *Min-Max Scaling*, implementado por meio da classe `MinMaxScaler` da biblioteca `sklearn`. Essa abordagem transforma os valores das variáveis para um intervalo predefinido (por padrão, entre 0 e 1), preservando as proporções relativas dos dados originais, o que assegura a integridade das relações intrínsecas entre os valores.

Para os dados de entrada, o escalador `MinMaxScaler` foi ajustado ao conjunto de treinamento para calcular os parâmetros de normalização. Em seguida, tanto os conjuntos de treinamento quanto os de teste foram transformados para o intervalo definido. Da mesma forma, o escalador foi aplicado aos dados de saída (`y_treino` e `y_teste`), sendo necessário realizar o ajuste de forma para adequar os dados unidimensionais ao formato exigido pelo método. Esse procedimento garantiu a consistência na escala entre as variáveis de entrada e saída, otimizando o desempenho dos modelos de aprendizado de máquina.

Os dados foram organizados em dois subconjuntos distintos para os processos de treinamento e teste. A distribuição seguiu uma proporção de 80% do total de dados para o treinamento do modelo e 20% para os processos de teste e validação. Essa divisão foi implementada utilizando a função `train_test_split` da biblioteca `sklearn`, com o parâmetro `random_state` fixado em 0. A definição de `random_state` assegurou a reprodutibilidade dos resultados, garantindo que o particionamento dos dados permaneça consistente em todas as execuções do modelo.

3.2.2.2 Configuração das funções de ativação, otimizadores e camadas ocultas

Neste estudo, para avaliar o desempenho do modelo de Redes Neurais Artificiais (RNA), foram testadas diferentes combinações de funções de ativação, otimizadores e configurações de camadas ocultas. Essas configurações são cruciais para determinar a arquitetura mais eficiente para o problema em questão, considerando o equilíbrio entre desempenho preditivo e a complexidade computacional.

I. **Funções de ativação e otimizadores**

As funções de ativação desempenham um papel essencial ao introduzir não linearidade ao modelo, permitindo que a rede neural aprenda relações complexas entre as variáveis. Para este estudo, foram selecionadas três funções de ativação: **ReLU** (Rectified Linear Unit), **sigmoide** (logística) e **tangente hiperbólica** (tanh).

Os algoritmos de otimização selecionados para este estudo foram o **Adam** (Adaptive Moment Estimation) e o **LBFGS** (Limited-memory Broyden-Fletcher-Goldfarb-Shanno). O Adam destaca-se por sua adaptabilidade e rapidez no ajuste de parâmetros, enquanto o LBFGS é particularmente eficiente no processamento de grandes conjuntos de dados com uso limitado de memória.

II. **Configuração das camadas ocultas**

As camadas ocultas de uma rede neural desempenham um papel crucial na aprendizagem das representações intermediárias dos dados. Neste estudo, foram testadas diferentes configurações dessas camadas para avaliar sua influência no desempenho do modelo. Em todas as experimentações, os modelos foram treinados por 2000 iterações, garantindo uma análise consistente do impacto das variações nas camadas ocultas sobre a performance do modelo.

As configurações testadas foram as seguintes:

- A15-7-7-7-1: três camadas ocultas com 7 neurônios cada;
- A15-10-10-1: duas camadas ocultas com 10 neurônios cada;
- A15-30-20-1: duas camadas ocultas, sendo a primeira com 30 neurônios e a segunda com 20 neurônios;
- A15-50-50-1: duas camadas ocultas com 50 neurônios cada;

- A15-100-50-1: duas camadas ocultas, com 100 neurônios na primeira e 50 neurônios na segunda;
- A15-100-100-1: duas camadas ocultas, ambas com 100 neurônios;
- A15-100-1: uma camada oculta com 100 neurônios.

Essas configurações foram escolhidas para cobrir uma gama de possibilidades, desde redes mais rasas (com menos neurônios e camadas) até redes mais profundas e complexas. A análise dessas configurações ajudou a identificar a arquitetura mais eficiente para o problema de regressão proposto.

3.2.2.3 Avaliação e armazenamento dos resultados

O treinamento foi conduzido por meio de um processo iterativo estruturado em um *loop* triplo, que considerou de forma sistemática variações nas funções de ativação, algoritmos de otimização e configurações das camadas ocultas. Em cada iteração, um novo modelo era treinado e posteriormente avaliado com base em métricas de desempenho, incluindo o **Erro Médio Absoluto (MAE)**, o **Coeficiente de Determinação (R^2)**, a **Raiz do Erro Quadrático Médio (RMSE)** e o **Erro Percentual Absoluto Médio (MAPE)**.

Os critérios considerados incluíram:

- A função de ativação mais adequada ao problema específico;
- O algoritmo de otimização que garantisse maior estabilidade e convergência eficiente;
- A arquitetura da rede mais apropriada para capturar a complexidade do problema, evitando superdimensionamento ou perda de eficiência computacional.

Os resultados detalhados de cada combinação foram registrados e analisados minuciosamente, permitindo a identificação de padrões e tendências relevantes. Essa abordagem sistemática viabilizou a seleção da configuração mais eficiente para o problema em questão, estabelecendo uma base consistente para as etapas subsequentes do estudo.

3.2.3 Árvores de decisão

A separação dos dados foi realizada utilizando a função `train_test_split` da biblioteca `sklearn`, com o objetivo de dividir a base em conjuntos de treinamento e teste e avaliar

o desempenho do modelo em dados não vistos durante o treinamento. Para garantir uma avaliação robusta e balanceada, 30% dos dados foram alocados ao conjunto de teste, mantendo um equilíbrio adequado entre os conjuntos de treinamento e validação.

Foram definidos dois grupos principais de parâmetros para ajustar o modelo e otimizar seu desempenho. O primeiro grupo, referente à profundidade máxima (`max_depth`), diz respeito ao número máximo de níveis que uma árvore de decisão pode ter. Foram avaliados diferentes valores de profundidade, incluindo 4, 8, 10, 12, 14 e 16. Com esses valores, foi possível investigar a relação entre a profundidade da árvore e a performance do modelo, permitindo uma análise detalhada do impacto de diferentes níveis de complexidade no desempenho.

O segundo grupo de parâmetros envolveu sementes para randomização (`random_state`), que foi utilizado para controlar a aleatoriedade no treinamento e inicialização do modelo. Essas sementes garantiram que os procedimentos de embaralhamento e inicialização dos modelos fossem reprodutíveis. Foram testados diferentes valores de sementes, como 5, 10, 15 e 20, com o objetivo de avaliar o efeito da aleatoriedade sobre o desempenho e a estabilidade do modelo.

Todo o processo, que inclui as etapas de pré-processamento, definição das combinações de hiperparâmetros, treinamento e avaliação, está descrito em detalhes no Apêndice B. Nesse apêndice, também é disponibilizado o código completo para consulta.

3.2.3.1 Avaliação de configurações dos modelos

Esta etapa adota um processo iterativo para avaliar o desempenho do modelo `DecisionTreeRegress` visando identificar as combinações ótimas de parâmetros que maximizem as métricas de desempenho, como o coeficiente de determinação (R^2), erro absoluto médio (MAE), raiz do erro quadrático médio (RMSE), erro percentual absoluto médio (MAPE) e variância explicada. Essas métricas são essenciais para garantir que o modelo seja robusto e eficiente, demonstrando uma excelente capacidade de predição e generalização dos dados.

O processo segue uma estrutura de teste organizada em loops, nos quais são avaliadas diversas configurações do modelo, alterando dois parâmetros principais: `max_depth` e `random_state`.

I. **max_depth:** Este parâmetro controla a profundidade máxima da árvore de decisão, ou seja, o número máximo de divisões ou camadas que a árvore pode ter. A profundidade da árvore influencia diretamente a complexidade do modelo. Uma profundidade muito grande pode levar ao *overfitting*, onde o modelo se ajusta excessivamente aos dados de

treinamento, perdendo a capacidade de generalizar para novos dados. Por outro lado, uma profundidade muito pequena pode resultar em um modelo subajustado, incapaz de capturar as complexidades dos dados. O objetivo do ajuste de `max_depth` é encontrar o ponto de equilíbrio entre um modelo muito simples e um muito complexo.

II. **random_state:** Este parâmetro define a semente de aleatoriedade do processo de treinamento, garantindo que os resultados sejam consistentes e reprodutíveis. Ao definir um valor para `random_state`, o modelo é inicializado de forma determinística, permitindo que os experimentos possam ser replicados com os mesmos dados e configurações, o que é crucial para a análise e comparação dos resultados.

A metodologia de teste é organizada em dois loops encadeados. O primeiro loop varia as diferentes configurações de `max_depth`, e o segundo varia os valores de `random_state`. Combinando as configurações de ambos os parâmetros, cada par de valores é testado, permitindo avaliar o impacto de cada um no desempenho do modelo.

Durante o treinamento e avaliação do modelo, para cada combinação de parâmetros testados, o `DecisionTreeRegressor` é inicializado com os valores específicos de `max_depth` e `random_state`. O modelo é então ajustado aos dados de treinamento usando o método `fit()`, que ajusta os parâmetros internos da árvore de decisão com base nos dados fornecidos.

Após o ajuste, o modelo é utilizado para gerar previsões em dois conjuntos de dados: o conjunto de treinamento e o conjunto de teste. Essas previsões são comparadas com os valores reais de ambos os conjuntos para calcular as métricas de desempenho. A partir dessas comparações, é possível avaliar a precisão do modelo, sua capacidade de se ajustar aos dados de treinamento (precisão) e sua capacidade de generalizar para novos dados (generalização).

Ao final do processo, as combinações de parâmetros que maximizam as métricas de desempenho são identificadas, garantindo um modelo preditivo mais robusto e eficiente. O uso dessas métricas auxilia na escolha do modelo mais adequado para as previsões, equilibrando precisão e capacidade de generalização.

3.2.4 Florestas Aleatórias

O processo descrito envolve a construção e avaliação de modelos de Florestas Aleatórias, com foco na separação dos dados, ajuste de hiperparâmetros e análise comparativa dos resultados. Inicialmente, o conjunto de dados foi dividido em dois subconjuntos: 70% para trei-

namento, utilizados no ajuste do modelo, e 30% para teste, destinados à avaliação da capacidade de generalização.

Na etapa de ajuste dos hiperparâmetros, foi explorado o impacto de diferentes configurações nos resultados do modelo. Um dos parâmetros analisados foi o número de estimadores, representado pelo hiperparâmetro `n_estimators`, que define a quantidade de árvores na floresta. Foram avaliados os valores de 10, 20 e 40 árvores para determinar se o aumento no número de estimadores resulta em melhorias na precisão do modelo, sem impactar significativamente o custo computacional. Esse ajuste reveste-se de importância, uma vez que o incremento no número de árvores pode elevar tanto a estabilidade quanto o desempenho preditivo do modelo. No entanto, após um certo limite, os benefícios adicionais tendem a ser irrisórios, evidenciando a necessidade de um balanceamento criterioso entre a precisão alcançada e os recursos computacionais demandados.

Outro aspecto investigado foi a profundidade máxima das árvores de decisão, controlada pelo parâmetro `max_depth`. Foram testados valores entre 4 e 14 para identificar o equilíbrio ideal entre subajuste e sobreajuste. Árvores muito rasas podem não capturar adequadamente os padrões dos dados, enquanto árvores muito profundas podem se ajustar excessivamente ao conjunto de treinamento, prejudicando sua capacidade de generalização. Além disso, o parâmetro `random_state` foi utilizado para garantir a reprodutibilidade dos experimentos, permitindo que os mesmos resultados sejam obtidos em execuções subsequentes ao definir valores fixos como 5, 10, 15 e 20.

Por fim, foram avaliados os critérios de divisão `squarederror` e `absoluteerror`, responsáveis por medir a qualidade das divisões nas árvores de decisão. O critério `squarederror` minimiza a soma dos quadrados dos erros, penalizando grandes desvios com mais intensidade, mas sendo sensível a outliers. Já o critério `absoluteerror` mede o erro absoluto médio, sendo mais robusto em cenários com presença de valores extremos. A comparação entre esses critérios permitiu analisar como diferentes abordagens de penalização afetam a precisão e a robustez do modelo.

Todo o processo, incluindo pré-processamento, definição de combinações de hiperparâmetros, treinamento e avaliação, está detalhado no Apêndice C, onde é possível consultar o código completo.

3.2.4.1 Treinamento e Avaliação

O treinamento e a avaliação do modelo de Floresta Aleatória foram conduzidos por meio de um processo iterativo sistemático, utilizando um loop aninhado para testar todas as combinações possíveis dos hiperparâmetros definidos. Esse método permitiu uma exploração abrangente do espaço de parâmetros, garantindo que a configuração ideal fosse identificada com base no desempenho do modelo. Para cada conjunto de valores especificados para os hiperparâmetros `n_estimators`, `criterion`, `max_depth` e `random_state`, o modelo foi instanciado e preparado para o treinamento.

No treinamento, o modelo foi ajustado utilizando os dados de treinamento, compostos pelas variáveis independentes e a variável dependente. Esse processo consiste em permitir que o algoritmo aprenda padrões e relações entre as variáveis de entrada e a saída esperada, ajustando internamente os parâmetros de cada árvore para otimizar o critério de divisão especificado.

Após o ajuste, o modelo foi empregado para gerar previsões tanto para o próprio conjunto de treinamento quanto para o conjunto de teste, que contém dados previamente separados e não utilizados durante o aprendizado. Essa etapa é crucial para avaliar se o modelo é capaz de generalizar os padrões aprendidos para dados novos.

A avaliação do modelo foi conduzida com base nas métricas R^2 (coeficiente de determinação), MAE (erro absoluto médio), RMSE (erro quadrático médio) e MAPE (erro percentual absoluto médio), seguindo a mesma abordagem utilizada para os demais modelos, a fim de garantir consistência e comparabilidade nos resultados obtidos.

3.2.5 XGBoost

A avaliação e otimização do algoritmo XGBoost foram realizadas por meio de uma análise aprofundada dos principais hiperparâmetros, de forma semelhante à abordagem adotada para os demais modelos. Essa análise visou ajustar as configurações do modelo de forma a maximizar sua capacidade de generalizar padrões nos dados, minimizando erros e evitando o overfitting. O processo envolveu a definição de combinações sistemáticas de valores para os parâmetros mais relevantes e a aplicação de estratégias de validação para garantir a representatividade dos resultados obtidos.

O parâmetro `n_estimators`, que define o número de árvores que compõem o modelo, foi ajustado com valores de 100, 300 e 600, permitindo avaliar o impacto da quantidade de estimadores na precisão e no custo computacional. Já o parâmetro `max_depth`, que limita a

profundidade das árvores de decisão, teve valores variando entre 4 e 14, com incrementos que possibilitaram identificar o equilíbrio entre a captura de interações complexas e o risco de sobreajuste. Árvores mais profundas tendem a modelar relações mais sofisticadas, mas também aumentam o potencial de superestimativa nos dados de treino.

Outro parâmetro ajustado foi a `learning_rate`, responsável por controlar o impacto incremental de cada árvore no resultado final do modelo. Foram avaliadas taxas de aprendizado reduzidas, como 0.01, 0.03 e 0.06, com o objetivo de assegurar uma convergência mais estável, enquanto taxas mais elevadas, como 0.09, foram testadas para acelerar o processo de ajuste, ainda que com maior risco de oscilação no treinamento. Adicionalmente, foram exploradas diferentes funções de objetivo, como `reg:squarederror` e `reg:absoluteerror`, para identificar o critério de erro mais adequado ao problema. O erro quadrático médio (MSE) foi utilizado devido à sua sensibilidade a grandes desvios, oferecendo uma penalização mais rigorosa, enquanto o erro absoluto médio (MAE) foi considerado por sua robustez frente à influência de outliers.

Para garantir a reprodutibilidade dos experimentos e evitar viés, o parâmetro `random_state` foi configurado com valores de 5, 10, 15 e 20. Essa abordagem permitiu avaliar a robustez do modelo frente a diferentes amostras de treino e teste, introduzindo variações controladas na distribuição dos dados. A validação cruzada foi empregada como estratégia de avaliação, dividindo os dados em subconjuntos múltiplos para testar todas as combinações possíveis de hiperparâmetros.

O processo de otimização foi conduzido por meio de uma busca sistemática que avaliou todas as 480 combinações possíveis dos hiperparâmetros definidos. Essa estratégia permitiu identificar a configuração ideal, maximizando o desempenho do modelo.

Todo o processo, que abrange desde o pré-processamento até a definição das combinações de hiperparâmetros, treinamento e avaliação, está descrito no Apêndice D, onde o código completo pode ser consultado.

3.2.6 Treinamento e avaliação

A configuração do loop de treinamento foi estruturada para explorar de maneira sistemática todas as combinações possíveis dos hiperparâmetros do modelo XGBoost. Para isso, foi implementado um loop aninhado, onde o loop externo testava diferentes valores de `n_estimators`, enquanto o loop interno variava os parâmetros de `max_depth`, `learning_rate`, `objective` e

`random_state`. A cada iteração, uma configuração distinta de parâmetros era usada para treinar um modelo XGBoost, o que permitia avaliar como cada combinação afetava o desempenho do modelo. A metodologia garantiu que todas as possíveis interações entre os hiperparâmetros fossem analisadas, possibilitando a identificação da melhor configuração para o problema de regressão.

A criação do modelo foi realizada instanciando o `XGBRegressor`, desenvolvido por Chen e Guestrin (2016), para cada combinação específica de hiperparâmetros. O `XGBRegressor` é uma implementação do XGBoost voltada para tarefas de regressão, que aplica o algoritmo de boosting para construir um modelo preditivo. Para cada configuração de parâmetros, foi gerada uma nova instância do modelo, que foi ajustada utilizando os dados de treinamento. O processo de treinamento consistiu na adaptação do modelo às características dos dados, ajustando os pesos das árvores de decisão de maneira a minimizar o erro conforme os critérios definidos.

Após o treinamento, o modelo gerado foi utilizado para realizar previsões tanto no conjunto de treinamento quanto no conjunto de teste. A comparação entre as previsões e os valores reais possibilita a avaliação do desempenho do modelo por meio de métricas como R^2 (coeficiente de determinação), MAE (erro absoluto médio), RMSE (erro quadrático médio) e MAPE (erro percentual absoluto médio), seguindo a mesma abordagem utilizada para os demais modelos.

3.2.7 Análise dos resultados

Na etapa de análise de desempenho dos modelos, foi desenvolvido um processo rigoroso e sistemático para seleção e classificação dos resultados, fundamentado em métricas essenciais de avaliação de desempenho: MAPE (Mean Absolute Percentage Error), MAE (Mean Absolute Error), R^2 (Coeficiente de Determinação) e RMSE (Root Mean Square Error). O principal objetivo foi identificar as três melhores configurações dos modelos preditivos baseados em Redes Neurais Artificiais, Árvores de Decisão, Florestas Aleatórias e XGBoost, considerando cada métrica analisada. A seguir, apresenta-se a análise detalhada.

1. **Ordenação pelo MAPE:** Inicialmente, os modelos foram classificados em ordem crescente com base nos valores do MAPE, permitindo identificar as configurações com o menor desvio percentual médio entre os valores previstos e observados, destacando aquelas com maior precisão relativa.

2. **Ordenação pelo MAE:** Em seguida, os modelos foram classificados em ordem crescente com base nos valores do MAE, priorizando as configurações com os menores desvios absolutos médios. Essa análise permitiu avaliar a precisão preditiva em termos absolutos, independentemente da escala dos valores observados.

3. **Ordenação pelo RMSE:** Em seguida, foi realizada a ordenação com base nos valores de RMSE. Essa métrica penaliza mais intensamente os desvios maiores, sendo especialmente relevante para identificar configurações que apresentam menor erro quadrático médio.

4. **Análise do R^2:** Por fim, os modelos foram avaliados com base nos valores de R^2, tanto nos conjuntos de treinamento quanto de teste. Essa análise permitiu priorizar configurações que demonstraram consistência no desempenho preditivo, garantindo equilíbrio entre ajuste e capacidade de generalização.

Após a classificação individual dos modelos com base em cada métrica de desempenho, foi conduzida uma análise integrada e interpretativa dos resultados. Esse procedimento permitiu a seleção dos três modelos com melhor desempenho para uma avaliação aprofundada. Diversos tipos de gráficos foram utilizados para analisar o desempenho e o comportamento dos modelos preditivos. A seguir, apresentam-se os principais gráficos gerados.

3.2.7.1 Curvas de aprendizado

Neste estudo, utilizou-se as curvas de aprendizado dos três melhores modelos, analisando como seu desempenho evolui conforme mais dados são incorporados ao treinamento. A função `plot_learning_curve_multiple` foi desenvolvida para gerar as curvas de aprendizado de forma simultânea para vários modelos. Os parâmetros de entrada incluem uma lista dos modelos a serem avaliados, o número de divisões (folds) para validação cruzada e a métrica de avaliação. Para esta análise de regressão, a métrica de R^2 foi selecionada, considerando-se as implicações de custo computacional associadas ao cálculo de métricas adicionais.

O processo inicia-se com a utilização da função `learning_curve` do Scikit-learn, que calcula o desempenho de cada modelo em diferentes tamanhos de conjunto de treinamento. Para cada modelo, a função calcula as pontuações médias de erro tanto para o treinamento quanto para a validação, variando o tamanho do conjunto de dados utilizado. Posteriormente, as curvas de aprendizado são geradas, com o eixo X representando o número de exemplos no treinamento

e o eixo Y indicando o R^2. Todas as curvas de aprendizado dos modelos são plotadas no mesmo gráfico, proporcionando uma comparação visual clara de como o desempenho de cada modelo evolui à medida que o tamanho do conjunto de treinamento aumenta.

3.2.7.2 Gráfico R^2

O gráfico de R^2 tem como objetivo comparar os valores de R^2 obtidos pelos três melhores modelos nos conjuntos de treinamento e teste. Para essa avaliação, foi utilizado o método `score()` do scikit-learn, que calcula o valor de R^2 para um conjunto de dados específico, seja de treinamento ou de teste. Esse método possibilita uma comparação direta da capacidade de cada modelo em explicar a variabilidade dos dados, fornecendo insights sobre seu desempenho tanto no treinamento quanto na generalização para dados não vistos.

3.2.7.3 Análise dos resíduos

O histórico de resíduos é definido como a diferença entre os valores reais e os valores previstos. A distribuição dos resíduos foi analisada por meio de um histograma gerado com a função `sns.histplot()` da biblioteca Seaborn (Waskom, 2021), proporcionando uma visão clara da distribuição dos erros. Para facilitar a interpretação, uma curva de densidade Kernel (KDE) foi sobreposta ao histograma, suavizando a distribuição e destacando a tendência central e a dispersão dos erros. A configuração do gráfico foi realizada utilizando a biblioteca Matplotlib, utilizando o método `plt.scatter`.

3.2.7.4 Validação cruzada

A validação cruzada é uma técnica fundamental para avaliar a capacidade de generalização de modelos preditivos. Esse processo consiste em dividir o conjunto de dados em várias partições (ou folds), assegurando que todos os dados sejam utilizados tanto para treinamento quanto para validação. A técnica é aplicada de forma semelhante a diferentes algoritmos de aprendizado de máquina, como Redes Neurais Artificiais (RNA), Árvores de Decisão, Random Forest e XGBoost. A seguir, detalha-se o processo de validação cruzada, com ênfase na aplicação desses métodos e utilizando a métrica R^2 para avaliação de desempenho.

Embora outras métricas, como o Erro Médio Absoluto Percentual (MAPE) e a Raiz do Erro Quadrático Médio (RMSE), tenham sido discutidas previamente, optou-se por utilizar exclusivamente o coeficiente de determinação R^2 e o desvio padrão nesta análise de validação

cruzada, considerando o custo computacional associado. O cálculo de métricas adicionais em cada iteração da validação cruzada implicaria um aumento significativo no tempo de processamento.

Ao adotar o R^2 como métrica principal, foi possível realizar uma avaliação mais ágil e eficiente, sem comprometer o desempenho computacional. A validação cruzada com o método KFold é aplicado a todos os modelos para avaliar sua performance. Esse procedimento consiste em dividir o conjunto de dados em k subconjuntos (ou folds) e realizar múltiplas iterações de treinamento e validação, garantindo uma avaliação imparcial do desempenho. O parâmetro `shuffle=True` é utilizado para embaralhar aleatoriamente os dados antes da divisão, o que ajuda a reduzir potenciais vieses e assegura uma distribuição mais equilibrada dos dados entre os folds.

O objetivo central em todos os casos é avaliar a capacidade de generalização dos modelos. Embora o número de divisões k possa variar sendo 5 para Redes Neurais Artificiais (RNA) e 10 para Árvores de Decisão, Random Forest e XGBoost a metodologia de avaliação segue o mesmo princípio. O processo de validação para RNA é mais robusto e exige maior capacidade de processamento devido à sua complexidade, o que justifica a escolha de menos divisões (5 folds). Esse procedimento assegura uma análise imparcial e abrangente, minimizando possíveis vieses e proporcionando uma visão clara e confiável sobre a robustez preditiva de cada modelo.

4 RESULTADOS E DISCUSSÃO

Este capítulo apresenta os resultados obtidos e as discussões relacionadas ao desempenho dos modelos de aprendizado de máquina avaliados neste estudo, abrangendo Redes Neurais Artificiais, Árvores de Decisão, Florestas Aleatórias (Random Forest) e o algoritmo XGBoost. A análise foi conduzida com base em métricas consolidadas na literatura, como o coeficiente de determinação (R^2), o erro absoluto médio (MAE), o erro quadrático médio (RMSE) e o erro percentual absoluto médio (MAPE), permitindo uma avaliação criteriosa da capacidade de previsão, da robustez e da generalização de cada modelo.

Além dos indicadores numéricos, foram analisadas as curvas de aprendizado, com o objetivo de compreender a evolução do desempenho dos modelos ao longo do treinamento e identificar possíveis sinais de sobreajuste (overfitting) ou subajuste (underfitting). A validação cruzada foi utilizada para conferir maior confiabilidade aos resultados, evitando que a segmentação dos dados influenciasse indevidamente o desempenho observado.

Também foi avaliada a distribuição dos resíduos, permitindo identificar eventuais padrões sistemáticos de erro e verificar a consistência dos modelos nas diferentes faixas de valores previstos. A partir desses resultados, são discutidas as vantagens e limitações de cada técnica, considerando não apenas o desempenho estatístico, mas também aspectos como a sensibilidade a dados ruidosos, a interpretabilidade dos modelos e o esforço computacional envolvido.

Por fim, é realizada uma comparação entre os modelos testados, com foco na identificação da abordagem mais adequada para a tarefa proposta. Essa escolha considera, de forma integrada, a precisão das previsões, o custo computacional e a aplicabilidade prática em contextos reais de engenharia geotécnica. A análise busca, assim, oferecer subsídios para a adoção de modelos preditivos eficientes, confiáveis e compatíveis com as demandas do dimensionamento de dispositivos de controle de percolação em fundações permeáveis de barragens.

4.1 Redes Neurais Artificiais

A partir da relação matemática $La = f(X;\theta)$, foi possível identificar e quantificar as correlações entre o comprimento do *abraço* e as variáveis de entrada do modelo. Essas variáveis, que totalizam 15 características (*features*), representam os principais parâmetros de controle do sistema em estudo. A variável de saída corresponde ao comprimento do dispositivo de controle

de percolação (em metros), sendo essa a grandeza prevista pelo modelo.

4.1.1 Curvas de Aprendizado

Os resultados da curva de aprendizado (Figura 22) indicam que todos os modelos beneficiam-se de um aumento no número de exemplos de treinamento, com elevações progressivas no coeficiente de determinação (R^2). Esse comportamento reflete a capacidade dos modelos de aprender padrões mais complexos e ajustar-se melhor aos dados à medida que mais informações são fornecidas. No entanto, diferenças significativas no desempenho entre os conjuntos de treinamento e validação revelam variações na capacidade de generalização de cada modelo, evidenciando diferentes níveis de *overfitting*.

A arquitetura RN1 demonstra um desempenho consistente entre os conjuntos de treinamento e validação, evidenciando sua capacidade de generalizar de forma eficaz para novos dados. Esse equilíbrio sugere que o modelo foi capaz de aprender os padrões fundamentais presentes nos dados, sem incorrer em memorização excessiva ou perda de informações críticas.

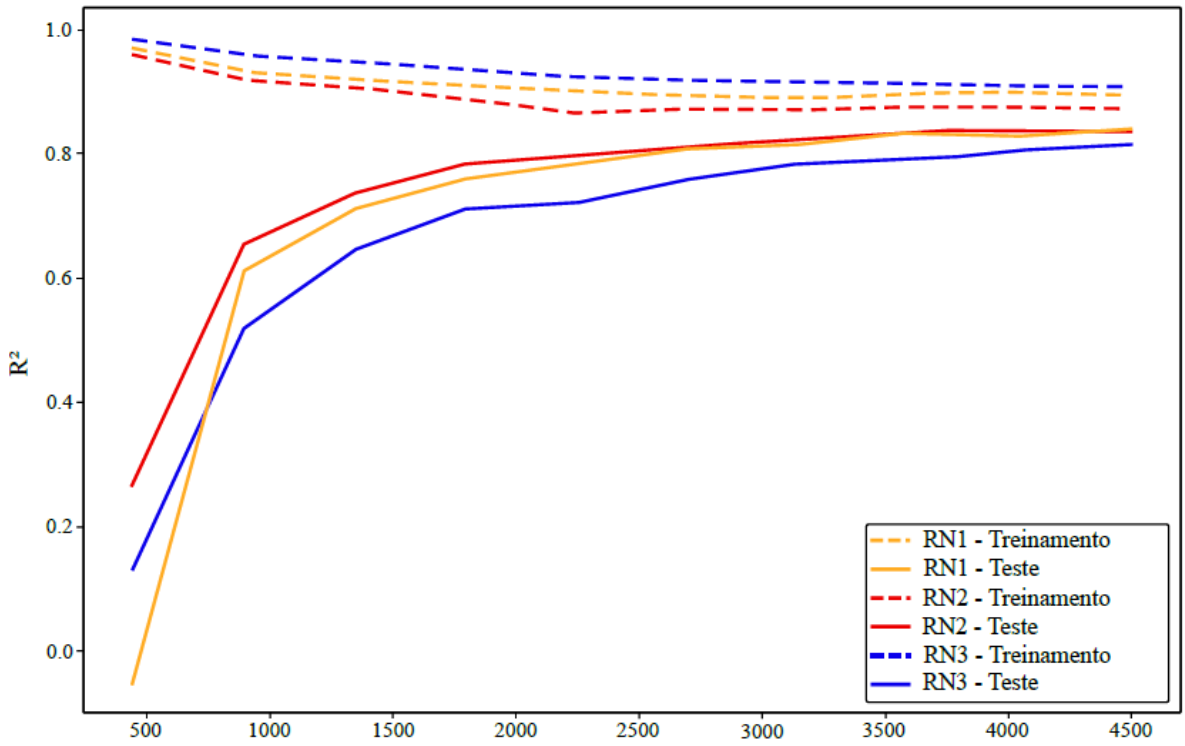
Por outro lado, a arquitetura RN2 apresenta uma discrepância mais pronunciada entre os desempenhos nos conjuntos de treinamento e validação, indicando sinais de sobreajuste moderado. Embora essa diferença não comprometa de forma significativa a aplicabilidade do modelo, ela aponta para a necessidade de ajustes nos hiperparâmetros ou da inclusão de um volume maior de dados de treinamento, o que poderia reduzir a discrepância e aprimorar sua capacidade de generalização.

Em contraste, a arquitetura RN3 revela a maior discrepância entre os desempenhos nos dois conjuntos, caracterizando um caso claro de sobreajuste (*overfitting*). Esse comportamento reflete a inclinação do modelo em memorizar os padrões específicos dos dados de treinamento, limitando sua aplicabilidade em cenários reais e comprometendo sua eficácia na predição de novos dados.

4.1.2 Comparação do R^2 (treinamento vs. teste)

A análise detalhada dos resultados obtidos, apresentada na Figura 23, indica que os valores do coeficiente de determinação (R^2) nos conjuntos de treinamento mantiveram-se elevados, posicionando-se de forma consistente em torno de 0,9 para todos os modelos avaliados. Este desempenho sugere que os modelos foram capazes de identificar e reproduzir, de maneira eficiente, os padrões subjacentes presentes nos dados utilizados para o treinamento. No entanto,

Figura 22 – Curvas de aprendizado para as arquiteturas RN1, RN2 e RN3



Fonte: Autora (2025)

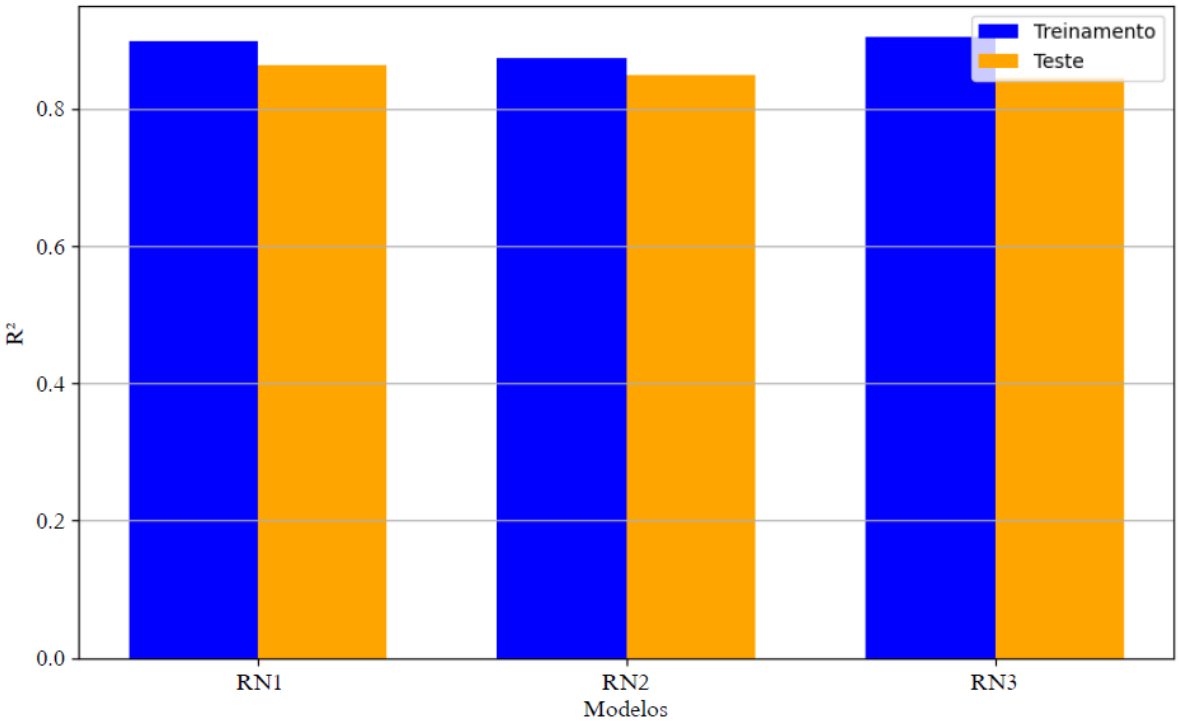
a discrepância observada entre os valores de R^2 nos conjuntos de treinamento e teste constitui uma métrica crítica para avaliar a capacidade de generalização de cada modelo. Diferenças acentuadas entre essas métricas podem indicar a presença de *overfitting*, o que comprometeria a eficiência dos modelos ao serem aplicados a novos dados. Esse comportamento evidencia a necessidade de ajustes no processo de modelagem, como a implementação de técnicas de regularização ou a revisão da complexidade do modelo, a fim de aprimorar seu desempenho preditivo em condições reais.

A arquitetura RN1 se destaca por apresentar a menor discrepância entre os valores de R^2 nos conjuntos de treinamento e teste. Por outro lado, as arquiteturas RN2 e RN3 exibem diferenças mais acentuadas, indicando a possibilidade de um leve sobreajuste. No entanto, apesar dessa tendência, ambos demonstram um desempenho consistente, indicando que, mesmo com uma maior especialização nos dados de treinamento, continuam a oferecer previsões de boa precisão.

4.1.3 Métricas de desempenho

A Tabela 4 apresenta os resultados de desempenho de três diferentes arquiteturas de redes neurais, avaliadas com base em diversas métricas de erro: coeficiente de determinação

Figura 23 – Comparação R^2 (treinamento vs teste) dos três melhores modelos - redes neurais



Fonte: Autora (2025)

(R^2), erro absoluto médio (MAE), erro quadrático médio (RMSE) e erro percentual absoluto médio (MAPE).

Os resultados demonstraram que a arquitetura RN1 teve o melhor desempenho geral, com $R^2 = 89.79\%$ no conjunto de treinamento e 86,39% no conjunto de teste. A arquitetura RN2 apresentou $R^2 = 87.38\%$ (treinamento) e 84.93% (teste), enquanto a arquitetura RN3 teve o maior R^2 no treinamento (90.50%), mas uma queda significativa no teste (84.43%), sugerindo *overfitting*.

Em termos de RMSE:

- RN1: $RMSE = 19.6$
- RN2: $RMSE = 20.62$
- RN3: $RMSE = 20.96$

A proporção do RMSE em relação à amplitude da variável dependente ($210 - 7 = 203$) foi:

$$\text{Proporção} = \frac{\text{RMSE}}{203} \times 100$$

- RN1: $\frac{19.6}{203} \times 100 \approx 9.66\%$
- RN2: $\frac{20.62}{203} \times 100 \approx 10.16\%$
- RN3: $\frac{20.96}{203} \times 100 \approx 10.33\%$

Quanto ao MAPE:

- RN1: 25.00%
- RN2: 25.53%
- RN3: 26.12%

Esses valores reforçam que a arquitetura RN1 é a mais equilibrada e confiável dentre as analisadas.

Tabela 4 – Métricas de desempenho para diferentes arquiteturas de redes neurais.

	RN1	**RN2**	**RN3**
Camadas ocultas	30, 20	100, 100	100, 50
Função de Ativação	tanh	tanh	relu
Otimização	lbfgs	lbfgs	lbfgs
Rš Treinamento	89.79%	87.38%	90.50%
Rš Teste	86.39%	84.93%	84.43%
MAE	13.5	14.39	14.82
RMSE	19.6	20.62	20.96
MAPE	25.00%	25.53%	26.12%

4.1.4 Validação cruzada

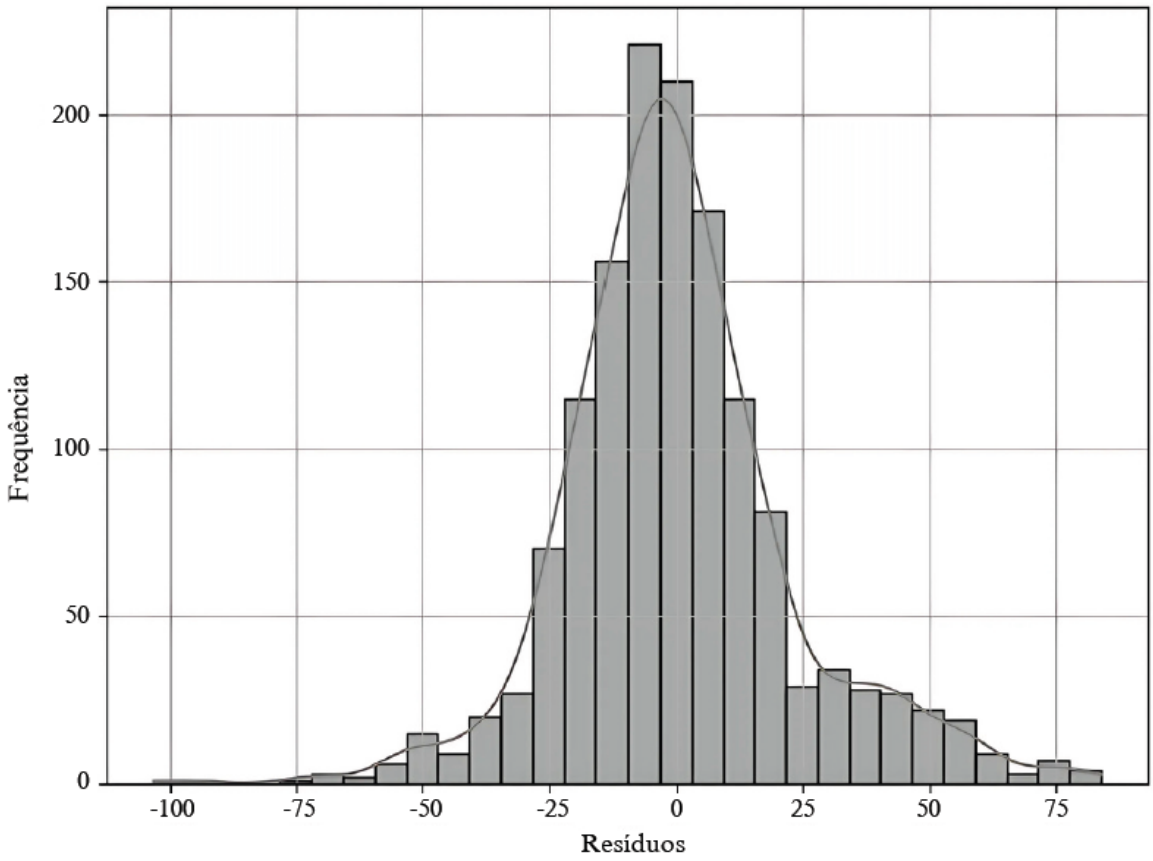
As métricas obtidas na análise de validação cruzada refletem o desempenho do melhor modelo identificado, o RN1. Esse modelo apresentou coeficiente de determinação médio $R^2 = 0.8377$, demonstrando capacidade de explicar, em média, 83,77% da variabilidade dos dados nos subconjuntos. O desvio padrão foi 0.0153, indicando baixa variabilidade e alta estabilidade entre os *folds*.

4.1.5 Análise dos resíduos

A análise do histograma (Figura 24) do modelo RN1 revela que os resíduos estão distribuídos de maneira aproximadamente simétrica e centralizados em torno de zero. A curva de

densidade sugere uma boa aproximação a uma distribuição normal, reforçando a adequação do modelo. No entanto, foram identificados alguns resíduos extremos, possivelmente associados a *outliers* nos dados ou limitações do modelo em capturar padrões complexos.

Figura 24 – Histograma de resíduos da modelo RN1



Fonte: Autora (2025)

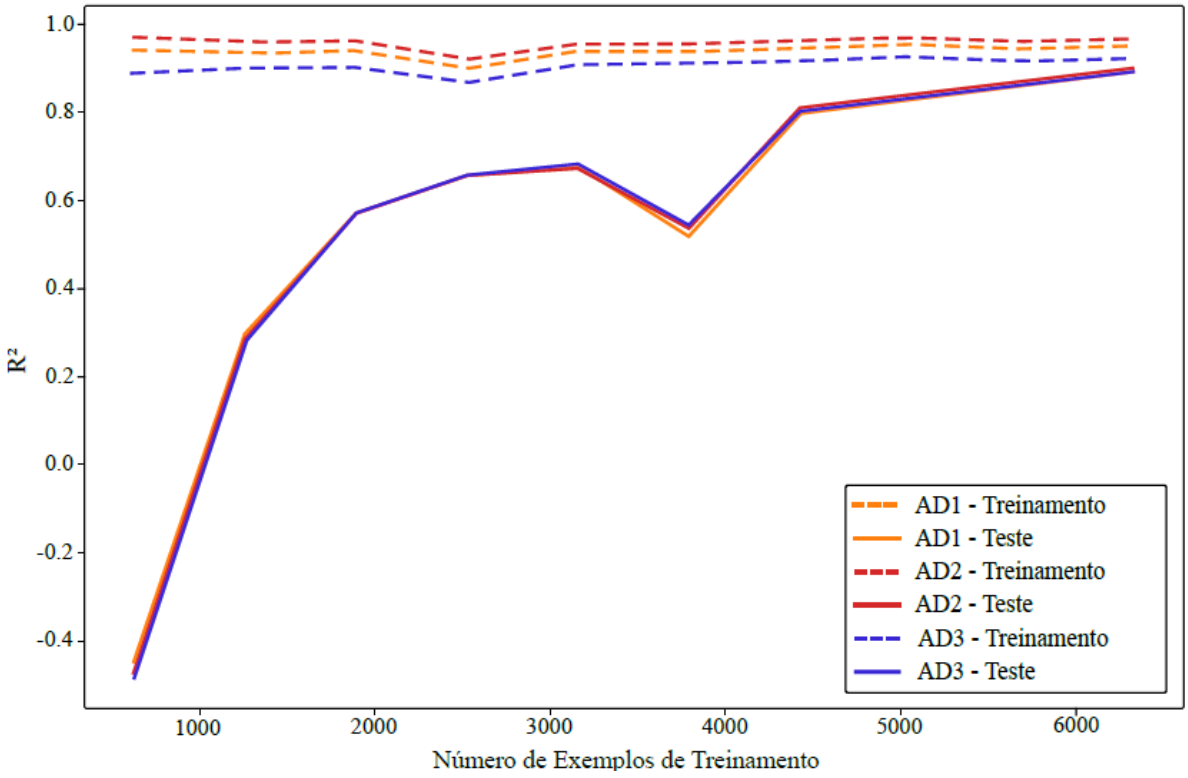
4.2 Árvores de Decisão

4.2.1 Curvas de aprendizado

A Figura 25 exibe as curvas de aprendizado combinadas, permitindo a comparação do desempenho de três modelos distintos (AD1, AD2 e AD3) em termos do coeficiente de determinação R^2, tanto para os dados de treinamento quanto para os de teste, em função do número de exemplos utilizados no treinamento.

A análise das curvas de desempenho dos modelos revelou diferenças relevantes em seus comportamentos. O modelo AD1 evidenciou leves sinais de overfitting, apresentando valores de R^2 próximos a 1 durante o treinamento, enquanto os resultados no conjunto de teste foram levemente inferiores. Em contrapartida, o modelo AD2 apresentou um desempenho mais equi-

Figura 25 – Curvas de aprendizado para as arquiteturas AD1, AD2 e AD3



Fonte: Autora (2025)

librado, com valores de R^2 elevados tanto no treinamento quanto no teste, embora a diferença entre as curvas sugira um leve overfitting. Por fim, o modelo AD3 registrou os menores valores de R^2 em ambas as fases, o que pode ser atribuído à baixa complexidade do modelo ou à influência de ruído nos dados, comprometendo sua capacidade de ajuste e predição.

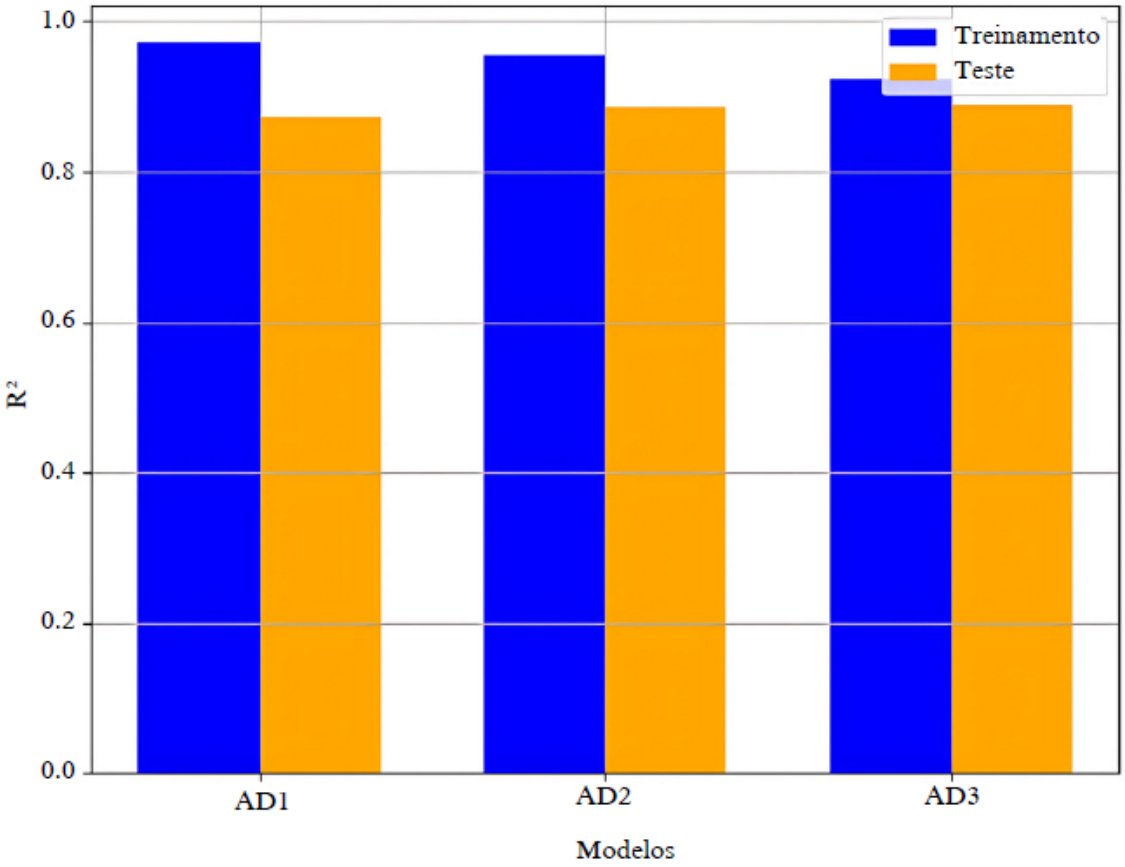
4.2.2 Gráfico comparativo R^2: treinamento vs. teste

Os valores de R^2 próximos de 1 para todos os modelos, tanto nos conjuntos de treinamento quanto nos de teste, indicam a eficácia dos modelos em explicar a variabilidade dos dados (Figura 26). A diferença mínima observada entre o desempenho nos conjuntos de treinamento e teste sugere a ausência de overfitting, evidenciando uma boa capacidade de generalização. Nesse sentido, o modelo AD1, embora apresente uma leve superioridade em relação aos outros, se destaca como o modelo mais eficiente.

4.2.3 Métricas de desempenho

A Tabela **??** apresenta as métricas de desempenho comparativo das três arquiteturas de maior desempenho, denominadas AD1, AD2 e AD3. Os resultados foram analisados com base

Figura 26 – Comparação R^2 (treinamento vs teste) dos três melhores modelos - árvores de decisão



Fonte: Autora (2025)

em um conjunto abrangente de métricas estatísticas, visando avaliar a eficácia preditiva de cada arquitetura. O coeficiente de determinação (R^2) variou de forma significativa entre os modelos analisados nas fases de treinamento e teste, evidenciando diferenças em suas capacidades de ajuste e generalização. Durante o treinamento, o modelo AD1 obteve o maior valor de R^2 (97.17%), indicando um excelente ajuste aos dados e uma notável habilidade em capturar as relações e padrões subjacentes, sem sinais aparentes de subajuste (underfitting). Em comparação, os modelos AD2 e AD3 apresentaram valores ligeiramente inferiores, de 95.40% e 95.39%, respectivamente, mas ainda demonstraram desempenho robusto na identificação dos padrões presentes no conjunto de treinamento. Esses resultados sugerem que, embora o modelo AD1 tenha se destacado em termos de ajuste, os modelos AD2 e AD3 mantiveram uma eficiência significativa no aprendizado, garantindo a assimilação adequada das características principais dos dados.

Por outro lado, na fase de teste, que é crítica para avaliar a capacidade de generalização do modelo para dados inéditos, o modelo AD2 se destacou com o maior valor de R^2 (88.65%),

Tabela 5 – Métricas de desempenho para diferentes arquiteturas de árvores de decisão.

	AD1	**AD2**	**AD3**
Profundidade máxima	12	10	10
Semente de aleatoriedade	10	5	20
Rš Treinamento	97.17%	95.40%	95.39%
Rš Teste	87.26%	88.65%	88.59%
MAE	7.08	7.16	7.16
RMSE	19.04	17.98	18.02
MAPE	14.19%	14.28%	14.34%

seguido de perto pelo modelo AD3 (88.59%) e, por fim, o modelo AD1 (87.26%). Embora o modelo AD1 tenha mostrado um ajuste excepcional durante o treinamento, sua leve diminuição na capacidade de generalização no conjunto de teste sugere uma leve tendência de sobreajuste (overfitting), onde o modelo pode ter memorizado características específicas do conjunto de treinamento que não são totalmente representativas do conjunto de teste. Em comparação, o modelo AD2 demonstrou a melhor capacidade de generalização, o que pode ser atribuído a um ajuste mais equilibrado, evitando a superespecialização nos dados de treinamento.

O modelo AD2 apresentou o menor RMSE (17.98), destacando-se como o mais eficaz na redução de grandes erros e evidenciando maior estabilidade nas previsões. O modelo AD3, com RMSE de 18.02, apresentou um desempenho bastante próximo ao de AD2. Já o modelo AD1, que obteve o maior RMSE (19.04), ainda demonstrou desempenho satisfatório, indicando capacidade preditiva, apesar de maior variação nos erros.

A proporção do RMSE em relação à amplitude foi calculada, fornecendo uma medida relativa para avaliar o desempenho dos modelos dentro da escala dos dados:

- Para o modelo AD1:

$$\text{Proporção (\%)} = \frac{19.04}{203} \times 100 \approx 9.38\%$$

- Para o modelo AD2:

$$\text{Proporção (\%)} = \frac{17.98}{203} \times 100 \approx 8.86\%$$

- Para o modelo AD3:

$$\text{Proporção (\%)} = \frac{18.02}{203} \times 100 \approx 8.88\%$$

Com base nesses cálculos, os valores de RMSE correspondem a cerca de 8.9% a 9.4% da amplitude total do conjunto de dados.

O modelo AD1 apresentou o menor valor de MAPE, igual a 14.19%, demonstrando que, em média, os desvios entre os valores previstos e os valores reais correspondem a 14.19% dos valores observados. Esse resultado destaca o AD1 como o melhor modelo em termos de erro relativo percentual. O modelo AD2 registrou um MAPE de 14.28%, indicando desempenho comparável ao de AD1, com uma diferença marginal, que reforça sua eficácia em aplicações práticas. O modelo AD3, por sua vez, alcançou um MAPE de 14.34%, permanecendo dentro de uma faixa aceitável de precisão, mas exibindo um desempenho ligeiramente inferior em relação aos outros dois modelos.

A análise das métricas de desempenho revelou que o modelo AD2 apresentou o melhor equilíbrio entre ajuste e generalização, com o maior R^2 no teste (88.65%) e o menor RMSE (17.98), demonstrando estabilidade e precisão nas previsões. Embora o modelo AD1 tenha alcançado o maior R^2 no treinamento (97.17%) e o menor MAPE (14.19%), sua maior variabilidade no RMSE (19.04) e leve redução no R^2 no teste indicam um leve sobreajuste. O modelo AD3 apresentou desempenho intermediário, com valores próximos aos de AD2, reforçando sua consistência em cenários práticos.

4.2.4 Validação cruzada

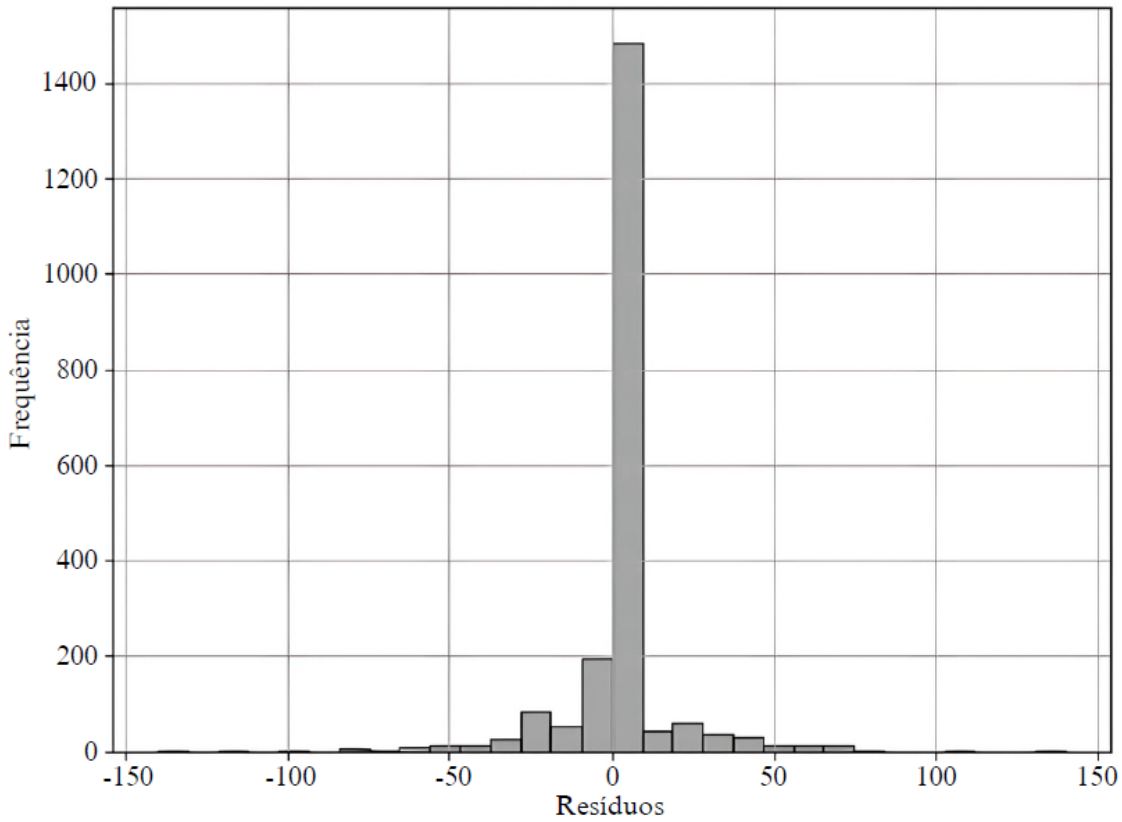
O modelo AD2 apresentou um coeficiente de determinação médio (R^2) de 0.8901 ao longo do processo de validação cruzada, o que indica que, em média, o modelo foi capaz de capturar 89% da variabilidade presente nos dados observados. Este desempenho sólido sugere uma forte relação entre as variáveis preditoras e a variável alvo, alinhando-se aos critérios de um modelo bem ajustado, dado que valores de R^2 próximos a 1 denotam alta capacidade explicativa. Além disso, o desvio padrão de R^2 foi de apenas 0.026, evidenciando uma baixa dispersão nos resultados obtidos entre as diferentes iterações da validação cruzada. Esse baixo desvio padrão é um indicador importante de consistência, uma vez que reflete a capacidade do modelo de manter desempenho estável independentemente das divisões específicas do conjunto de dados em subconjuntos de treinamento e teste.

4.2.5 Análise dos resíduos

Na análise apresentada, os resíduos do modelo AD2 (Figura 27) mostram uma leve assimetria à direita, com uma cauda mais longa para valores positivos. Essa característica sugere que o modelo apresenta uma tendência a subestimar os valores mais altos da variável alvo,

indicando a necessidade de ajustes para melhorar a precisão nessas regiões. Ainda assim, a maior parte dos resíduos está concentrada em torno de zero, o que evidencia um bom ajuste geral para a maioria dos dados analisados.

Figura 27 – Histograma de resíduos da modelo AD2



Fonte: Autora (2025)

Entretanto, a presença de outliers na extremidade direita do histograma sugere a existência de pontos influentes ou atípicos, que podem estar impactando o desempenho do modelo. Embora a média dos resíduos seja próxima de zero, indicando ausência de viés significativo, a distribuição dos erros não segue perfeitamente uma curva normal devido à assimetria e à variância observada em valores extremos.

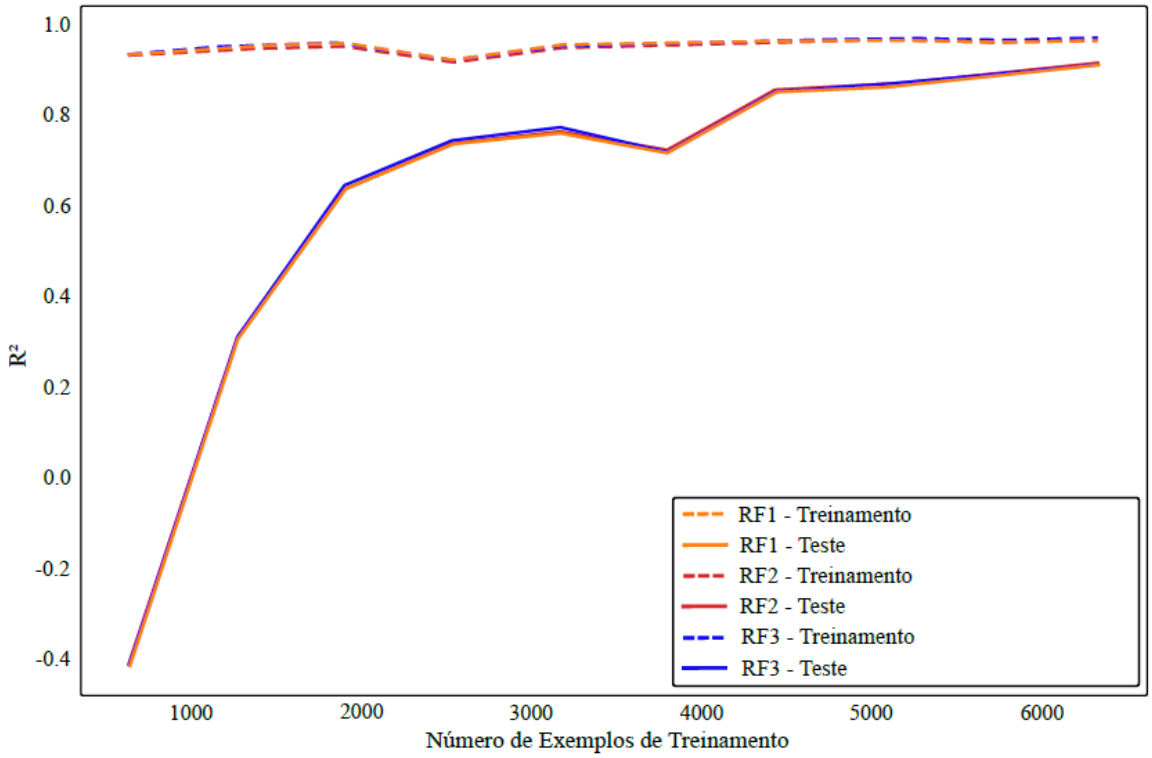
4.3 Florestas Aleatórias

4.3.1 Curvas de aprendizado

A Figura 28 ilustra as curvas de aprendizado para três modelos distintos, avaliados por meio do coeficiente de determinação (R^2), tanto para os dados de treinamento quanto de validação, em função do número de exemplos utilizados no treinamento. Os três modelos apresenta-

ram desempenhos similares, com diferenças praticamente imperceptíveis entre si, o que reflete um nível equivalente de eficiência nas predições realizadas.

Figura 28 – Curvas de aprendizado para as arquiteturas RF1, RF2 e RF3



Fonte: Autora (2025)

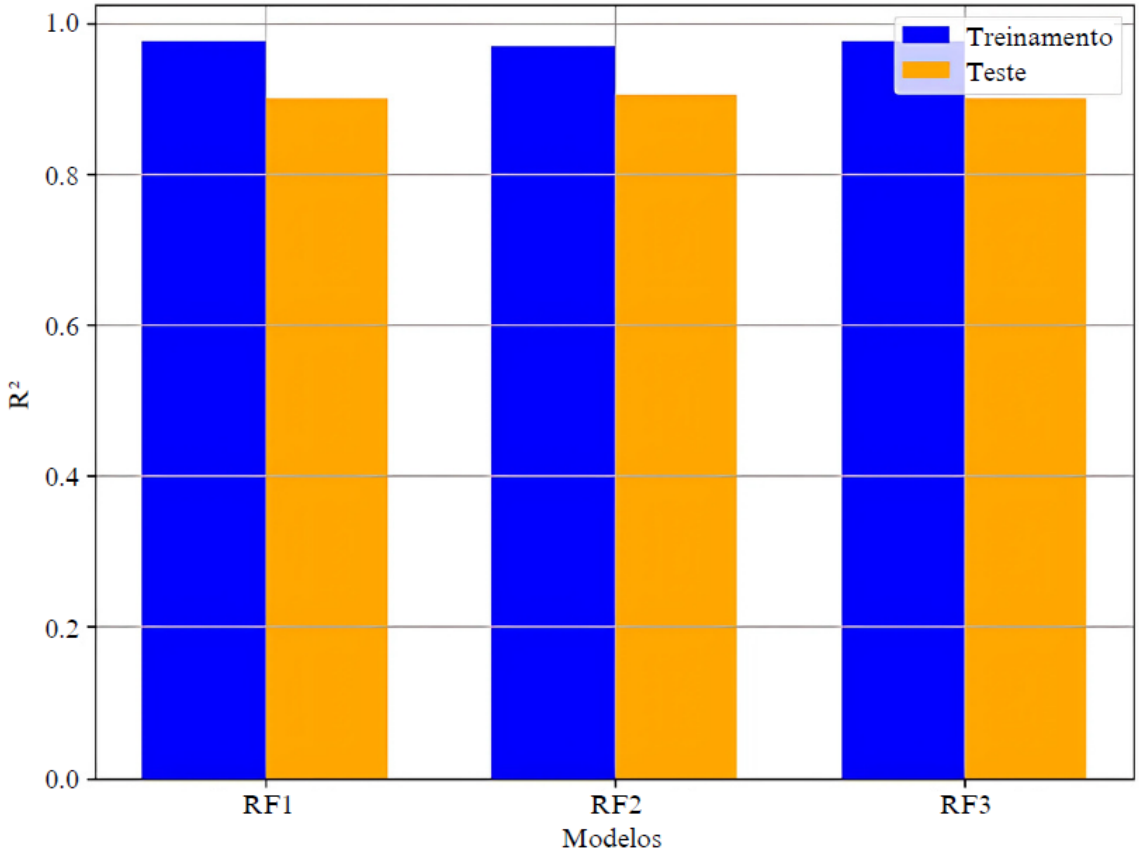
Observou-se uma leve queda nos valores de R^2 durante a validação no intervalo entre 3.000 e 4.000 exemplos de treinamento; contudo, as curvas estabilizaram em torno de 0.8 após esse ponto. Embora os modelos demonstrem um desempenho geral satisfatório, essa oscilação sugere limitações na validação, especialmente em capturar integralmente a complexidade dos dados nesse estágio específico, indicando a necessidade de ajustes finos ou maior volume de dados.

De modo geral, os modelos apresentaram um crescimento expressivo nos valores de R^2 no intervalo inicial, especialmente entre 500 e 2.000 exemplos de treinamento. Esse comportamento destaca a importância de um volume inicial adequado de dados para potencializar o aprendizado e aprimorar a eficácia do modelo. Após aproximadamente 4.000 exemplos, as curvas demonstraram estabilização, indicando que a inclusão de dados adicionais além desse ponto exerce um impacto irrisório na melhoria do desempenho, sugerindo que o modelo já alcançou um patamar próximo ao seu limite de aprendizado para o conjunto de dados em questão.

4.3.2 Gráfico comparativo R^2: treinamento vs. teste

A Figura 29 compara os coeficientes de determinação (R^2) para os dados de treinamento e teste em três modelos distintos, destacando o modelo RF1 com um R^2 próximo de 1, o que evidencia uma alta capacidade dos preditores em explicar a variância dos dados em ambos os conjuntos. O leve decréscimo no R^2 do treinamento para o teste indica uma boa generalização, com baixa tendência ao *overfitting*. Apesar de os demais modelos apresentarem desempenho satisfatório, seus resultados foram ligeiramente inferiores, reforçando a escolha do modelo RF1 como o mais adequado para as análises subsequentes.

Figura 29 – Comparação R^2 (treinamento vs teste) dos três melhores modelos - florestas aleatórias



Fonte: Autora (2025)

4.3.3 Métricas de desempenho

A seguir, são apresentados os resultados obtidos a partir da avaliação de três arquiteturas de modelos de *machine learning* (denominadas RF1, RF2 e RF3), com foco em métricas de desempenho aplicadas a um conjunto de dados. As métricas analisadas incluem o coeficiente

de determinação (R^2), o erro médio absoluto (MAE), a raiz quadrada do erro médio (RMSE) e o erro percentual absoluto médio (MAPE).

Nos dados de treinamento, todos os modelos apresentaram valores de R^2 elevados, com o modelo RF3 se destacando ligeiramente com 97.52%, seguido por RF1 (97.41%) e RF2 (96.86%), conforme apresentado na Tabela **??**. Esses valores indicam que os modelos são altamente eficazes em ajustar-se aos dados de treinamento, explicando a maior parte da variabilidade observada.

Tabela 6 – Métricas de desempenho para diferentes arquiteturas de florestas aleatórias.

	RF1	**RF2**	**RF3**
Nř de estimadores	20	20	40
Critério	squared_error	squared_error	squared_error
Profundidade máxima	14	12	14
Semente de aleatoriedade	20	20	10
Rš Treinamento	97.41%	96.86%	97.52%
Rš Teste	89.96%	90.45%	89.98%
MAE	6.96	6.91	7.07
RMSE	16.90	16.48	16.89
MAPE	13.80%	13.87%	13.97%

No entanto, ao avaliar o desempenho nos dados de teste, houve uma leve redução nos valores de R^2, o que pode sugerir um pequeno efeito de *overfitting*. Os resultados de R^2 para os modelos no teste variaram de 89.96% (RF1) a 90.45% (RF2), indicando que, apesar da diminuição, os modelos ainda mantêm uma boa capacidade de generalização.

Os resultados revelam que, apesar das pequenas variações nas métricas de desempenho, os modelos RF1, RF2 e RF3 apresentaram desempenhos bastante próximos entre si, com uma leve vantagem para o modelo RF3 em termos de R^2 durante o treinamento. No entanto, a diferença no desempenho entre os modelos nos dados de teste é mínima, não sendo suficiente para indicar uma superioridade clara de um modelo em relação ao outro.

Com base nos resultados, é possível observar que todos os modelos apresentaram um desempenho bastante próximo entre si, com uma ligeira vantagem para o modelo RF2, que obteve o menor valor de RMSE (16.48). Esse resultado sugere que o modelo RF2 foi o mais robusto em relação aos erros quadráticos, sendo menos impactado por grandes desvios nas previsões em comparação aos outros modelos.

A proporção do RMSE em relação à amplitude foi calculada, fornecendo uma medida relativa para avaliar o desempenho dos modelos dentro da escala dos dados:

- Para o modelo **RF1**:
  Proporção (%) $= \frac{16.9}{203} \times 100 \approx 8.33\%$

- Para o modelo **RF2**:
  Proporção (%) $= \frac{16.48}{203} \times 100 \approx 8.12\%$

- Para o modelo **RF3**:
  Proporção (%) $= \frac{16.89}{203} \times 100 \approx 8.32\%$

Com base nesses cálculos, os valores de RMSE correspondem a cerca de 8.12% a 8.33% da amplitude total do conjunto de dados.

A diferença entre os valores de MAPE de RF1, RF2 e RF3 é extremamente pequena, indicando que, em termos de precisão percentual, os modelos apresentam um desempenho praticamente idêntico. Essa consistência nas métricas de MAPE reforça a eficácia geral dos modelos na previsão, com todos eles apresentando erros percentuais muito baixos. No entanto, é importante destacar que o RF1 demonstrou uma leve vantagem em relação aos outros modelos, apresentando um MAPE ligeiramente inferior.

Os resultados indicam que, embora haja algumas variações nas métricas de desempenho, os modelos RF1, RF2 e RF3 apresentaram desempenhos muito semelhantes entre si, com uma leve vantagem para o modelo RF3 em termos de R^2 no conjunto de treinamento. No entanto, a diferença observada no desempenho de teste entre os modelos é mínima, o que não permite concluir com clareza a superioridade de um modelo sobre os outros. A escolha entre as arquiteturas pode, portanto, depender de outros fatores cruciais, como a complexidade do modelo, o tempo de treinamento necessário e a robustez frente a *outliers*, que podem influenciar de maneira significativa o desempenho em contextos específicos.

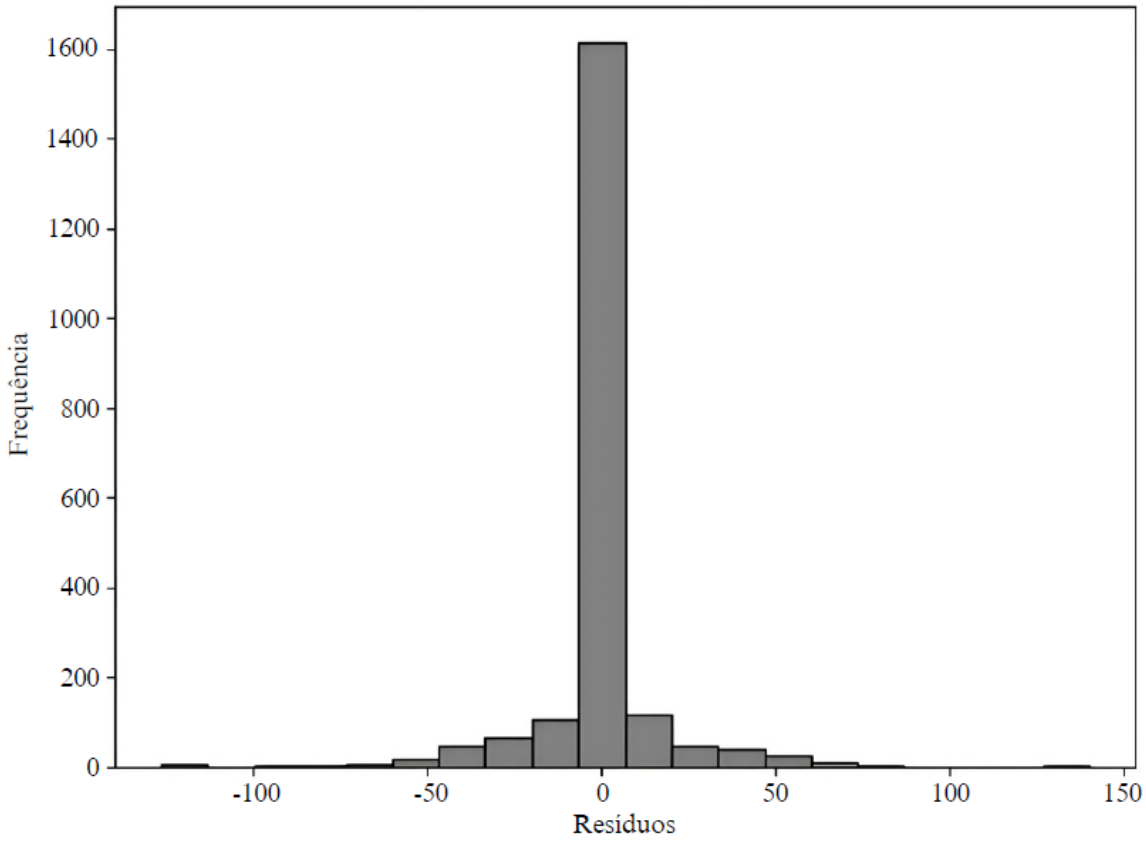
4.3.4 Validação cruzada

O modelo de Floresta Aleatória RF1 apresentou um bom desempenho na validação cruzada, com um coeficiente de determinação médio (R^2) de 0.9119, o que indica que cerca de 91.19% da variabilidade dos dados foi explicada pelo modelo. O desvio padrão do R^2 foi de 0,0150, mostrando que o modelo teve um desempenho estável e consistente ao longo das diferentes partições dos dados. Esses resultados sugerem que o RF1 é uma opção confiável para a tarefa em questão, com boa capacidade de generalização.

4.3.5 Análise dos resíduos

O histograma dos resíduos do modelo RF1 (Figura 30) apresenta uma distribuição predominantemente concentrada em torno de zero, indicando que as diferenças entre os valores reais e preditos são, em sua maioria, pequenas. Essa característica sugere que o modelo está bem ajustado, com erros residuais aleatórios e ausência de vieses significativos. Os poucos resíduos distantes da média zero podem estar relacionados a observações atípicas ou a limitações do modelo na captura de padrões específicos presentes nos dados.

Figura 30 – Histograma de resíduos da modelo RF1



Fonte: Autora (2025)

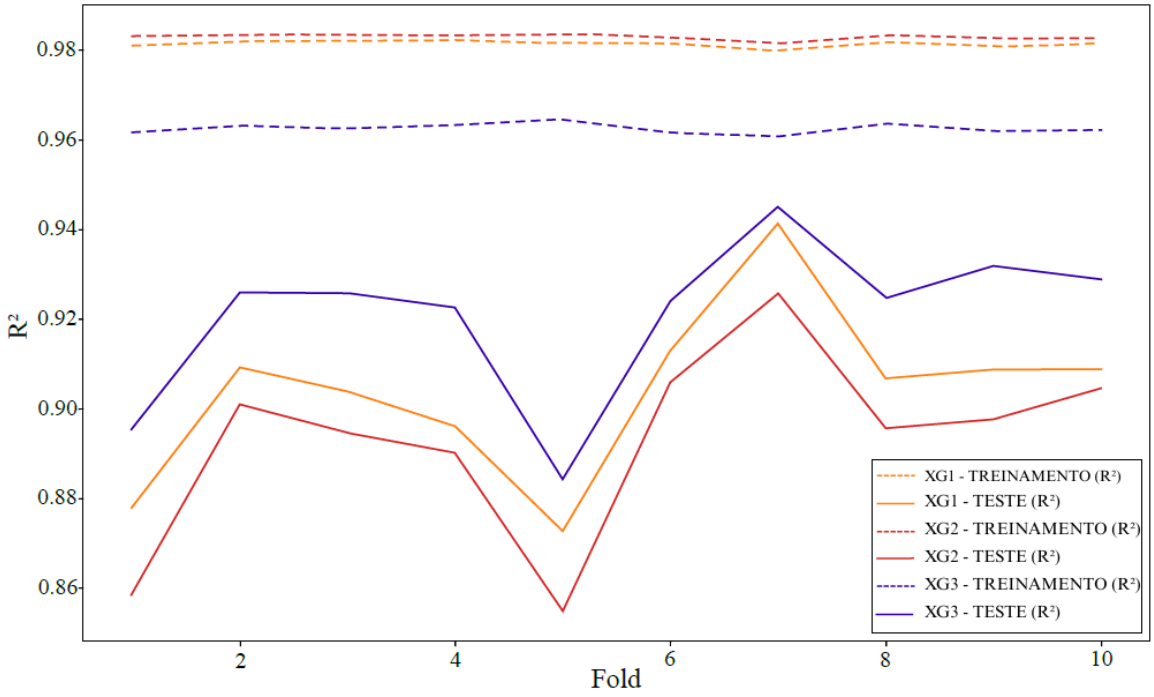
4.4 XGBoost

4.4.1 Curvas de aprendizado

A Figura 31 apresenta as curvas de aprendizado de três modelos distintos, avaliados com base no coeficiente de determinação (R^2), considerando tanto o conjunto de treinamento quanto o de teste. A análise foi realizada em função do número de *folds* na validação cruzada,

permitindo observar a evolução do desempenho preditivo de cada modelo à medida que os dados são particionados de maneira incremental.

Figura 31 – Curvas de aprendizado para as arquiteturas XG1, XG2 e XG3



Fonte: Autora (2025)

O modelo XG1 demonstrou um desempenho consistente, caracterizado pela proximidade entre os erros e a acurácia nas fases de treinamento e validação, além de flutuações mínimas durante a validação, refletindo sua estabilidade e robustez. Esse comportamento evidencia a eficácia do modelo na captura de padrões relevantes nos dados, reduzindo o risco de sobreajuste (*overfitting*) e garantindo uma boa capacidade de generalização para dados inéditos.

O modelo XG2 apresentou uma variância ligeiramente superior em comparação aos demais, embora ainda dentro de níveis aceitáveis em termos absolutos. Apesar da presença de picos e vales durante a validação, a amplitude dessas flutuações permaneceu moderada, refletindo um desempenho consistente e dentro de uma faixa adequada. A pequena diferença entre as curvas de treinamento e validação sugere que o modelo não sofre de sobreajuste significativo, demonstrando boa capacidade de generalização.

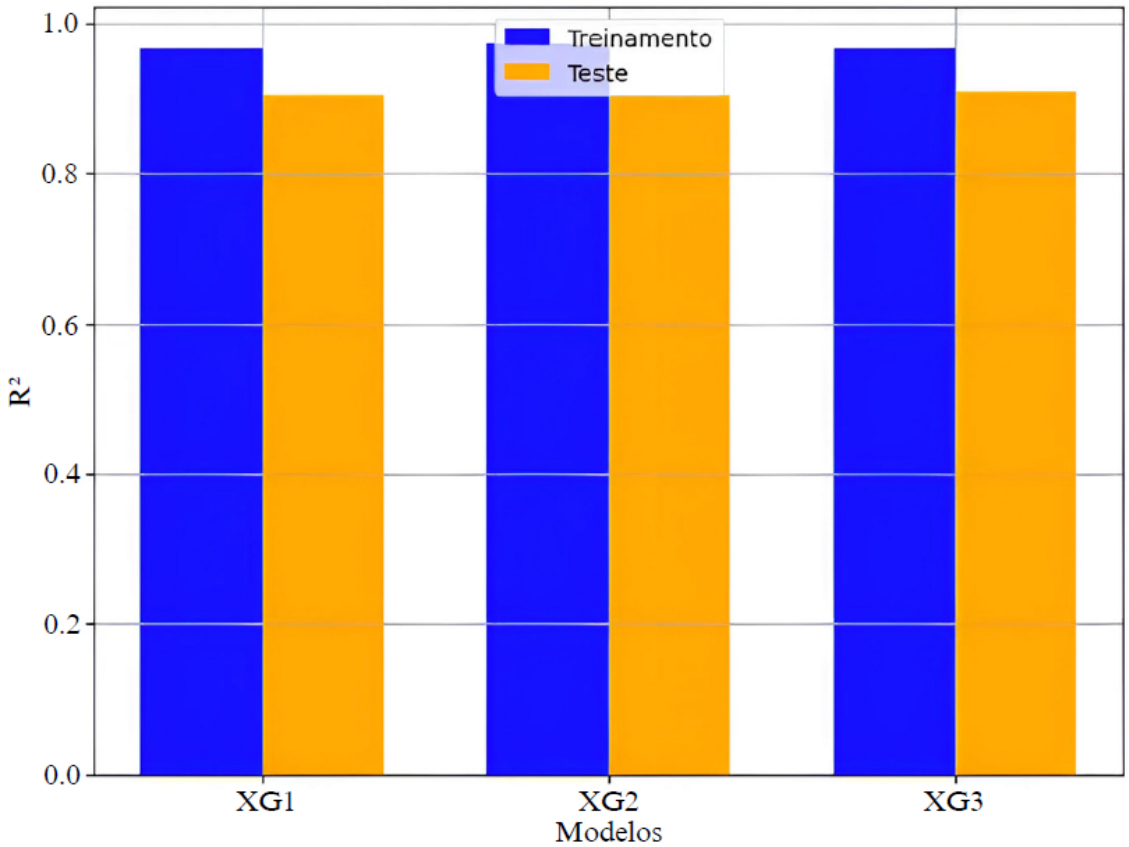
O modelo XG3 demonstrou um desempenho estável durante o treinamento, com variações na validação semelhantes às observadas nos demais modelos, caracterizadas por picos e vales de amplitude moderada. Apesar de apresentar resultados de validação próximos aos dos outros modelos, foi observada uma queda ligeiramente mais acentuada no desempenho em relação ao conjunto de treinamento, indicando um possível sobreajuste (*overfitting*). No entanto,

essa diferença é relativamente pequena e não compromete significativamente sua capacidade preditiva.

4.4.2 Gráfico comparativo R^2: treinamento vs. teste

A Figura 32 compara os coeficientes de determinação (R^2) obtidos pelos diferentes modelos durante o treinamento e o teste. Os valores do coeficiente de determinação indicam que todos os modelos apresentaram um ajuste de alta qualidade aos dados, com resultados próximos de 1, o que sugere uma excelente precisão na previsão, corroborando a robustez dos modelos em modelar a relação entre as variáveis de entrada e saída. O modelo XG1, que apresenta a menor

Figura 32 – Comparação R2 (treinamento vs teste) dos três melhores modelos - XGBoost



Fonte: Autora (2025)

diferença entre os valores de R^2 no treinamento e no teste, destaca-se como o mais equilibrado entre os avaliados. Esse comportamento sugere que o modelo conseguiu aprender os padrões subjacentes dos dados sem recorrer à memorização de informações específicas, mantendo sua consistência e capacidade de generalização ao lidar com dados não vistos.

O modelo XG2, por sua vez, apresentou um desempenho intermediário, com uma diferença ligeiramente maior entre as fases de treinamento e validação, o que indica uma tendência

moderada ao sobreajuste.

Já o modelo XG3 demonstrou maior diferença entre os valores de R^2, refletindo um claro comportamento de sobreajuste. Apesar de seu excelente desempenho no treinamento, a incapacidade de manter resultados semelhantes no teste indica que o modelo memorizou os dados de treinamento em vez de identificar padrões generalizáveis.

Assim, o modelo XG1 desponta como a melhor opção, combinando alta precisão e boa capacidade de generalização.

4.4.3 Métricas de desempenho

Os resultados da Tabela 7 referem-se à avaliação de três configurações distintas de uma rede XGBoost (XG1, XG2 e XG3), um algoritmo de aprendizado supervisionado baseado em árvores de decisão. A seguir, apresenta-se uma análise detalhada dos valores de cada métrica de desempenho, levando em consideração as especificidades de cada arquitetura de modelo. Essa análise tem como objetivo esclarecer de que maneira cada configuração impactou o desempenho do modelo, permitindo uma comparação dos resultados obtidos.

A arquitetura XG1 obteve um desempenho de 96.58% (R^2) no treinamento, indicando um bom ajuste aos dados. No entanto, o desempenho no conjunto de teste caiu para 90.58%, sugerindo uma leve perda na capacidade de generalização. Esse decréscimo, embora ainda indicativo de um bom desempenho, sugere que o modelo pode estar ligeiramente sobreajustado aos dados de treinamento.

Por outro lado, a arquitetura XG2 obteve um desempenho de 97.40% (R^2) no treinamento e 90.47% no teste, apresentando uma redução um pouco menor em relação ao modelo XG1. Isso indica uma leve melhoria na capacidade de generalização, embora o desempenho também tenha decaído no conjunto de teste.

Já a arquitetura XG3, que utilizou um maior número de estimadores (600) e uma taxa de aprendizado mais baixa (0.01), obteve um desempenho de 96.65% no treinamento e 90.94% no teste, com uma queda semelhante à dos modelos anteriores. Embora o desempenho no treinamento tenha sido ligeiramente superior, a redução na capacidade de generalização foi praticamente idêntica.

As arquiteturas avaliadas apresentaram variações nos valores de RMSE, evidenciando diferenças no desempenho preditivo. A arquitetura XG1 registrou um RMSE de 16.38, enquanto a arquitetura XG2 obteve um RMSE de 16.46, ligeiramente superior ao de XG1, indicando uma

Tabela 7 – Métricas de desempenho para diferentes modelos XGBoost

	XG1	**XG2**	**XG3**
Nř de estimadores	300	300	600
Taxa de aprendizado	0.06	0.06	0.01
Critério	reg:absoluteerror	reg:absoluteerror	reg:squarederror
Profundidade máxima	10	12	8
Semente de aleatoriedade	5	20	5
Rš Treinamento	96.58%	97.40%	96.65%
Rš Teste	90.58%	90.47%	90.94%
MAE	7.03	6.82	7.43
RMSE	16.38	16.46	16.06
MAPE	13.57%	13.82%	15.26%

leve redução na precisão das previsões. Já a arquitetura XG3 destacou-se como a mais eficiente, alcançando o menor RMSE, de 16.06, demonstrando maior capacidade de minimizar os erros preditivos em comparação às demais arquiteturas.

A proporção do RMSE em relação à amplitude foi calculada, fornecendo uma medida relativa para avaliar o desempenho dos modelos dentro da escala dos dados:

- Para o modelo **XG1**:
  Propoção (%) $\frac{16.38}{203} \times 100 \approx 8.06\%$
- Para o modelo **XG2**:
  Propoção (%) $\frac{16.46}{203} \times 100 \approx 8.10\%$
- Para o modelo **XG3**:
  Propoção (%) $\frac{16.06}{203} \times 100 \approx 7.92\%$

Com base nesses cálculos, os valores de RMSE para as arquiteturas XG1, XG2 e XG3 correspondem a cerca de 7.92% a 8.10% da amplitude total do conjunto de dados. Esses resultados indicam que as três arquiteturas apresentaram desempenhos relativamente próximos, com a XG3 tendo a menor proporção de erro em relação à amplitude total dos dados.

As arquiteturas analisadas apresentaram os seguintes valores de MAPE, refletindo o desempenho percentual relativo das previsões em relação aos valores reais. A arquitetura XG1 alcançou um MAPE de 13.57%, indicando uma boa precisão relativa. A arquitetura XG2 obteve um MAPE de 13.82%, um valor ligeiramente superior ao de XG1, sugerindo uma leve redução na precisão percentual das previsões. Já a arquitetura XG3 apresentou o maior MAPE,

de 15.26%, indicando um erro percentual relativamente mais alto em comparação aos demais modelos.

Em resumo, as arquiteturas XG1, XG2 e XG3 apresentaram bom desempenho, com valores de R^2 próximos a 90% no conjunto de teste. A XG1 teve uma queda de aproximadamente 6% em relação ao treinamento, sugerindo risco de overfitting, enquanto a XG2 mostrou uma redução menor. A XG3, com mais estimadores e uma taxa de aprendizado menor, também apresentou queda similar, mas se destacou com o menor RMSE (16.06) e a menor proporção de erro. No MAPE, a XG1 obteve o melhor desempenho (13.57%), enquanto a XG3 teve o maior (15.26%).

4.4.4 Validação cruzada

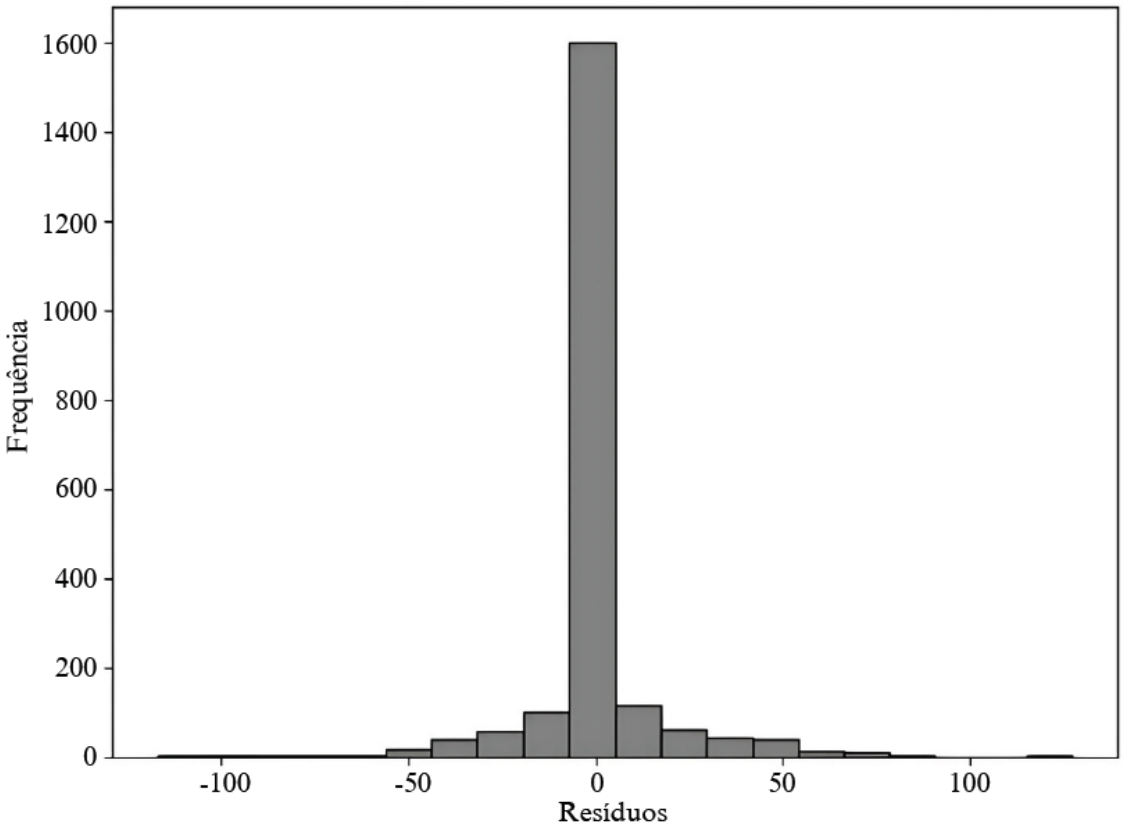
O modelo XG1, baseado em XGBoost, obteve o melhor desempenho entre todos os modelos avaliados, com um coeficiente de determinação (R^2) médio de 0.9140 na validação cruzada. Esse valor indica que aproximadamente 91.40% da variabilidade dos dados foi explicada pelo modelo, superando os demais métodos testados. Além disso, o desvio padrão do R^2 foi de 0.0134, um valor baixo que reflete a consistência e a estabilidade do modelo em diferentes conjuntos de dados. Esses resultados destacam o XG1 como a opção mais eficiente.

4.4.5 Análise dos resíduos

A distribuição dos resíduos (Figura 33) do modelo XG1 apresenta uma forma aproximadamente simétrica em torno de zero, indicando que os erros estão equilibrados e não há viés significativo, o que é desejável para a precisão do modelo. A curva de densidade sugere que os resíduos seguem uma distribuição normal, o que é uma premissa importante para a validade dos modelos de regressão.

Embora a maioria dos resíduos esteja concentrada em torno de zero, são observados alguns valores residuais maiores, o que indica previsões menos precisas em casos específicos. Esses valores elevados podem sugerir a presença de outliers ou de padrões não totalmente capturados pelo modelo, apontando para a necessidade de ajustes adicionais para melhorar a precisão em casos extremos.

Figura 33 – Histograma de resísuos do modelo XG1



Fonte: Autora (2025)

4.5 Comparação das técnicas de aprendizado de máquina

Com base nos resultados detalhados na Tabela 8, pode-se realizar uma análise técnica e abrangente do desempenho das arquiteturas avaliadas: Redes Neurais (RN1), Árvores de Decisão (AD2), Florestas Aleatórias (RF1) e XGBoost (XG1), utilizando as métricas de desempenho R^2, MAE, RMSE e MAPE, para os conjuntos de treinamento e teste.

O modelo de redes neurais (RN1) apresentou o menor desempenho entre as arquiteturas avaliadas. O coeficiente de determinação (R^2) foi de 89.79% no treinamento e 86.39% no teste, indicando uma capacidade moderada de capturar os padrões gerais nos dados, mas com limitação na generalização para novos conjuntos. Os valores elevados de erro (MAE = 13.5; RMSE = 19.6; MAPE = 25.00%) refletem uma dificuldade significativa em prever valores com precisão, principalmente em casos extremos. Esses resultados sugerem que a arquitetura (RN1) não foi capaz de modelar de maneira eficiente as complexidades e não linearidades dos dados analisados. Possíveis melhorias poderiam incluir ajustes na arquitetura da rede, como aumento no número de neurônios e camadas ocultas.

As árvores de decisão (AD2) apresentaram um desempenho significativamente melhor

Tabela 8 – Desempenho das arquiteturas avaliadas.

Arquiteturas	R^2		MAE	RMSE	MAPE
	Treinamento	Teste			
RN1 (Redes Neurais)	89.79%	86.39%	13.5	19.6	25.00%
AD2 (Árvores de Decisão)	95.40%	88.65%	7.16	17.98	14.28%
RF1 (Florestas Aleatórias)	97.41%	89.96%	6.96	16.9	13.80%
XG1 (XGBoost)	96.58%	90.58%	7.03	16.38	13.57%

que (RN1). O R^2 atingiu 95.40% no treinamento e 88.65% no teste, demonstrando uma capacidade razoável de generalização. Os erros observados (MAE = 7.16; RMSE = 17.98; MAPE = 14.28%) foram substancialmente menores que os de (RN1), indicando maior precisão preditiva e estabilidade. No entanto, os resultados sugerem que (AD2) ainda não alcançou o desempenho das arquiteturas mais avançadas, possivelmente devido à natureza determinística e à sensibilidade da árvore a ruídos nos dados. A combinação dessa abordagem com técnicas de *ensemble*, como florestas aleatórias ou boosting, pode melhorar significativamente o desempenho.

O modelo XGBoost (XG1) apresentou o melhor desempenho global. Com um R^2 de 96.58% no treinamento e 90,58% no teste, evidenciou a maior capacidade de generalização entre os modelos avaliados. Os erros também foram baixos (MAE = 7.03; RMSE = 16.38; MAPE = 13.7%), demonstrando alta precisão e eficácia na captura de padrões complexos nos dados. Sua eficácia pode ser atribuída à combinação de eficiência computacional e capacidade de lidar com dados complexos, destacando-o como uma solução ideal para cenários que priorizam a generalização.

O modelo de florestas aleatórias (RF1) também apresentou um desempenho notável, com o maior R^2 no treinamento (97.41%) e um excelente resultado no teste (89.96%). Seus erros foram ligeiramente inferiores aos do XGBoost (XG1) (MAE = 6.96; RMSE = 16.9; MAPE = 13.80%), reforçando sua eficácia no ajuste e na previsão. A abordagem de *ensemble* adotada pelo (RF1) reduz o risco de sobreajuste e melhora a estabilidade preditiva, tornando-o uma alternativa robusta, especialmente em situações em que a precisão nos dados de treinamento é crucial.

A Tabela 9 apresenta uma comparação entre diferentes modelos de aprendizado de máquina, considerando seu desempenho, o comportamento dos resíduos e o custo computacional associado a cada modelo.

Tabela 9 – Comparação das técnicas de aprendizado.

Modelo	Desempenho (R^2)	Resíduos	Processamento
Redes Neurais	Moderado	Moderado	Alto
Árvores de Decisão	Boa	Pouco	Baixo
Florestas Aleatórias	Excelente	Pouco	Moderado
XGBoost	Excelente	Pouco	Alto

4.5.1 Redes Neurais

As redes neurais artificiais (RNAs) utilizadas neste estudo demonstraram uma grande flexibilidade e competência na modelagem de dados, com valores de coeficiente de determinação (R^2) próximos de 0.9 durante o treinamento. Este valor sugere uma habilidade substancial do modelo em aprender padrões complexos presentes nos dados e em capturar as relações não lineares entre as variáveis independentes e dependentes. No entanto, uma análise mais detalhada das curvas de treinamento e teste revelou uma diferença significativa, especialmente nos modelos RN1 e RN2, indicando uma tendência ao *overfitting*.

O histograma dos resíduos do modelo RN1 revela uma distribuição que se aproxima da normalidade, com uma curva em forma de sino, sugerindo que a maioria dos resíduos (diferenças entre valores previstos e reais) está concentrada em torno de zero. Esse comportamento indica que o modelo, na maior parte das vezes, faz previsões que estão muito próximas dos valores reais. Contudo, ao analisar a dispersão dos resíduos, percebe-se uma maior variabilidade quando comparado aos outros modelos. Os resíduos variam de aproximadamente -100 a 75, o que significa que, embora a maioria dos erros seja pequena, há casos em que os erros são substancialmente elevados.

Essa variabilidade nos resíduos pode ser um reflexo da flexibilidade das RNAs. Embora elas tenham a capacidade de capturar padrões complexos, essa mesma flexibilidade pode torná-las mais suscetíveis ao *overfitting* e à sensibilidade a ruídos presentes nos dados, resultando em resíduos extremos em situações com valores atípicos ou dados mais ruidosos.

Em comparação com modelos mais simples, como as árvores de decisão ou as florestas aleatórias, as RNAs exigem maior poder de processamento, mais tempo de treinamento e, frequentemente, maior capacidade de memória. Embora sua capacidade de capturar padrões complexos seja um ponto forte, o alto custo computacional pode ser um fator limitante em casos com grandes volumes de dados ou em ambientes onde o tempo de processamento é crítico.

Embora o modelo RN1 tenha demonstrado eficácia em capturar padrões complexos, a presença de resíduos elevados sugere que ele pode não ser o modelo mais eficiente em to-

dos os cenários. A modelagem com RNAs, especialmente em casos com dados ruidosos ou com variabilidade significativa, requer cuidados na interpretação e aplicação do modelo, pois a capacidade de generalização pode ser comprometida por padrões excessivamente específicos aprendidos durante o treinamento.

4.5.2 Árvores de Decisão

Em contraste com as redes neurais artificiais, o modelo de árvore de decisão demonstrou um desempenho igualmente sólido, com valores de coeficiente de determinação (R^2) próximos de 0.9 tanto para o conjunto de treinamento quanto para o de teste. Esse resultado reflete uma excelente capacidade do modelo em aprender e generalizar os padrões subjacentes aos dados, sem sinais evidentes de *overfitting*.

Durante a validação, foi observada uma leve queda no desempenho, especialmente entre 3.000 e 4.000 exemplos. Esse declínio temporário foi seguido por uma estabilização das curvas, com valores de R^2 superiores a 0.8, o que sugere que o modelo manteve um bom ajuste geral, sem grandes desvios. A ausência de *overfitting* pode ser atribuída ao caráter não paramétrico das árvores de decisão, que tendem a ser mais resilientes ao ajuste excessivo aos dados.

O histograma dos resíduos do modelo AD2 apresenta uma grande concentração de resíduos em torno de zero, com elevada frequência na região central, indicando previsões consistentes e precisas, com poucos erros extremos. A dispersão dos resíduos é consideravelmente menor do que a observada no modelo RN1, variando de -50 a 50, o que sugere maior estabilidade.

Em termos de robustez e confiabilidade, o modelo AD2 é menos suscetível a *outliers*, pois as árvores de decisão dividem os dados em segmentos discretos, reduzindo o impacto de valores extremos. Por outro lado, essa rigidez reduz sua capacidade de adaptação a variações nos dados, tornando-o mais adequado para dados estruturados e com menor complexidade.

Em relação ao custo de processamento, o modelo de árvore de decisão apresenta vantagem significativa. São modelos simples e rápidos de treinar, com custo computacional reduzido, uma vez que funcionam por meio de divisões hierárquicas dos dados, ao contrário das RNAs que requerem ajustes iterativos de múltiplos parâmetros.

4.5.3 Florestas Aleatórias

As florestas aleatórias (*random forests*) representam uma alternativa robusta e eficaz aos modelos baseados em árvores de decisão isoladas, oferecendo maior capacidade de generaliza-

ção e menor risco de *overfitting*. O modelo combina múltiplas árvores de decisão e faz a média de seus resultados, o que reduz a influência de flutuações nos dados e melhora a estabilidade do modelo.

O histograma dos resíduos mostra alta concentração de valores próximos de zero, refletindo a consistência das previsões. Juntamente com o XGBoost, as florestas aleatórias apresentam a maior frequência de resíduos na região central, com dispersão baixa predominantemente entre -25 e 25 e poucos valores extremos, entre -100 e 100.

Esse comportamento é característico do modelo, que ao combinar previsões de múltiplas árvores, reduz o risco de *overfitting*, aumenta a robustez e melhora a capacidade de generalização. Além disso, as florestas aleatórias são menos sensíveis a ruídos e *outliers*, sendo mais precisas do que redes neurais em diversos cenários.

Apesar da eficiência, as florestas aleatórias requerem mais recursos computacionais do que modelos isolados, já que envolvem o treinamento simultâneo de múltiplas árvores. No entanto, em comparação com as RNAs, seu custo computacional é mais acessível, pois não exigem o ajuste intensivo de parâmetros nem processos de treinamento iterativo prolongado, tornando-as uma excelente escolha quando se deseja equilíbrio entre precisão e eficiência.

4.5.4 XGBoost

O modelo XGBoost se destaca como a solução mais avançada entre os avaliados, apresentando desempenho superior em diversas métricas. Utilizando a técnica de *boosting*, o modelo reduz iterativamente os erros residuais, permitindo a captura de padrões complexos de forma eficaz.

A generalização do modelo é notável, com curvas de teste acompanhando de perto as de treinamento, sem indícios de *overfitting* ou *underfitting*. Isso demonstra a capacidade do XGBoost de aprender padrões úteis que se aplicam a dados não vistos.

O histograma dos resíduos mostra alta concentração de resíduos em torno de zero, com dispersão extremamente baixa (entre -25 e 25). Isso confirma a eficácia do modelo em minimizar os erros e sua superioridade em previsões precisas, mesmo em cenários com variabilidade nos dados.

A robustez proporcionada pelo *boosting*, aliada à consistência dos resíduos, evidencia que o XGBoost é capaz de gerar previsões confiáveis e com poucos erros extremos, tornando-o uma das melhores alternativas em termos de precisão, generalização e estabilidade.

4.6 Limitações do modelo

O modelo desenvolvido ao longo deste estudo, embora tecnicamente robusto e metodologicamente consistente, apresenta uma série de limitações que devem ser cuidadosamente consideradas antes de sua aplicação em contextos práticos. As simplificações adotadas na modelagem numérica, como a hipótese de homogeneidade do maciço e a consideração de uma razão de anisotropia fixa, foram necessárias para viabilizar a construção da base de dados e a convergência dos modelos computacionais. No entanto, tais premissas implicam na perda de representatividade de situações onde a heterogeneidade do solo, camadas estratificadas ou condições de anisotropia complexas desempenham papel fundamental na condução do fluxo subterrâneo e, consequentemente, na eficácia dos dispositivos de controle de percolação.

Além disso, a total dependência de dados sintéticos, oriundos de simulações tridimensionais pelo método dos elementos finitos (GeoStudio Seep/W 3D), constitui uma limitação relevante. A ausência de validação empírica seja por meio de ensaios experimentais ou retroanálise com dados de barragens reais pode comprometer a confiabilidade dos modelos preditivos gerados por algoritmos de aprendizado de máquina. A representatividade das simulações numéricas é limitada por fatores como condições de contorno idealizadas, geometrias fixas e parametrizações discretas, o que reduz a capacidade do modelo de capturar fenômenos localizados, variações naturais ou situações atípicas, como eventos extremos ou falhas construtivas.

No âmbito do aprendizado de máquina, os algoritmos utilizados incluindo redes neurais artificiais, Random Forest e XGBoost possuem limitações inerentes. Embora o ajuste de hiperparâmetros e a validação cruzada tenham sido realizados de forma sistemática, permanece o risco de superajuste, especialmente em um conjunto de dados com estrutura gerada artificialmente e variações paramétricas pré-definidas. Outro desafio é a reduzida interpretabilidade desses modelos, que embora apresentem boa acurácia, funcionam como caixas-pretas, dificultando a extração de relações causais diretas entre variáveis de entrada e o desempenho hidráulico dos dispositivos de controle. Em engenharia geotécnica, essa dificuldade de interpretação representa um obstáculo prático para a aplicação direta dos resultados em projetos executivos e decisões de campo.

A parametrização do dispositivo abraço e dos elementos associados, como tapetes drenantes, enrocamento e cutoff, foi realizada com base em escalonamentos proporcionais à altura da barragem e intervalos discretos de análise. Embora essa abordagem seja metodologicamente adequada para a realização de estudos comparativos, ela não reflete a complexidade e variabili-

dade das condições encontradas em barragens reais. Em projetos de grande porte, a geometria da fundação e os parâmetros associados ao sistema de drenagem podem variar significativamente dependendo das condições locais, como a heterogeneidade do solo, a presença de camadas impermeáveis ou as características geotécnicas da fundação. A ausência de variações geométricas mais detalhadas, como ajustes adaptativos para diferentes tipos de solo e condições hidráulicas locais, limita a capacidade do modelo de representar de forma precisa a realidade de barragens com configurações mais complexas.

Além disso, o modelo não incorpora fatores dinâmicos essenciais que influenciam a performance do sistema de percolação em barragens operacionais, como os ciclos de enchente e rebaixamento, eventos climáticos extremos ou processos degenerativos, como erosão interna e fissuração do solo. Esses fenômenos podem alterar de maneira significativa a eficácia dos dispositivos de controle de percolação ao longo do tempo. A ausência desses elementos dinâmicos no modelo, juntamente com a falta de dados empíricos de monitoramento em tempo real, restringe a extrapolação dos resultados para cenários reais de operação. Assim, é fundamental realizar estudos adicionais que integrem dados de campo, permitindo uma validação mais robusta e a adaptação do modelo a situações práticas, melhorando sua aplicabilidade e precisão em barragens em funcionamento.

5 CONCLUSÕES E SUGESTÕES PARA PESQUISAS FUTURAS

5.1 Conclusões

A análise dos resultados alcançados nesta pesquisa evidenciou contribuições significativas no campo do dimensionamento do comprimento de dispositivos de controle de percolação interna -abraço - em barragens de terra. A metodologia proposta, que integra simulações computacionais tridimensionais utilizando o Método dos Elementos Finitos (MEF) com técnicas avançadas de aprendizado de máquina, demonstrou-se altamente eficaz na construção de modelos preditivos robustos para a determinação precisa do comprimento ideal dos dispositivos do tipo abraço. Esses dispositivos desempenham papel fundamental na mitigação de gradientes hidráulicos críticos em interfaces solo-estrutura, prevenindo processos erosivos, como a erosão interna, e promovendo a segurança estrutural das barragens.

Os modelos preditivos foram avaliados utilizando um conjunto de métricas de desempenho, incluindo o coeficiente de determinação (R^2), o erro quadrático médio (RMSE), o erro absoluto médio (MAE) e o erro percentual absoluto médio (MAPE), garantindo uma análise de sua precisão e capacidade preditiva. Adicionalmente, a avaliação foi complementada por análises gráficas, como gráficos de dispersão, histogramas de resíduos e curvas de aprendizado, que ofereceram perspectivas visuais sobre o desempenho dos modelos, evidenciando seu ajuste aos dados e sua capacidade de generalização. Essa abordagem combinada permitiu não apenas quantificar a eficácia dos modelos, mas também identificar padrões, tendências e limitações específicas de cada técnica, fornecendo uma base para comparações e melhorias futuras.

As Redes Neurais Artificiais (RNAs) demonstraram excelente capacidade de capturar padrões complexos e não lineares nas relações entre as variáveis, alcançando um coeficiente de determinação (R^2) próximo a 0,9 durante o treinamento, refletindo uma boa precisão. No entanto, análises qualitativas indicaram a ocorrência de overfitting em configurações específicas, como nos modelos RN1 e RN2, evidenciada por discrepâncias entre os desempenhos nos conjuntos de treinamento e validação. O histograma de resíduos apresentou uma distribuição quase normal, com resíduos concentrados em torno de zero, mas uma amplitude significativa (-100 a 75) destacou a sensibilidade das RNAs a ruídos e outliers, sugerindo a necessidade de ajustes adicionais para melhorar a generalização dos modelos.

As árvores de decisão demonstraram maior robustez contra overfitting devido à sua es-

trutura não paramétrica, apresentando coeficiente de determinação (R^2) elevado e desempenho estável tanto no treinamento quanto na validação. No entanto, enfrentaram limitações na modelagem de relações altamente não lineares, o que reduziu sua precisão em cenários de maior complexidade. O histograma de resíduos revelou uma variação mais estável (-50 a 50), destacando menor suscetibilidade a ruídos e valores atípicos, o que reforça a consistência dessas técnicas em ambientes com maior variabilidade nos dados.

As florestas aleatórias combinaram a robustez das árvores de decisão com uma maior capacidade de generalização, aproveitando a construção de múltiplas árvores para mitigar o impacto de outliers e ruídos nos dados. O desempenho geral foi consistente, com um coeficiente de determinação (R^2) elevado e diferenças reduzidas entre os conjuntos de treinamento e teste, indicando uma excelente capacidade de generalização. Além disso, a amplitude dos resíduos, variando de -40 a 40, foi significativamente menor em comparação às RNAs, destacando sua eficiência na modelagem de padrões complexos com menor sensibilidade a valores extremos.

O modelo XGBoost se destacou como a solução mais eficiente, superando as demais técnicas em precisão, generalização e capacidade de capturar padrões complexos, utilizando o método de boosting para ajustar iterativamente os erros residuais e refinar as previsões. As curvas de treinamento e teste acompanharam-se de perto, sem sinais de overfitting ou underfitting, evidenciando sua habilidade em aprender padrões robustos e generalizáveis. O histograma de resíduos apresentou alta concentração em torno de zero, com amplitude variando entre -25 e 25, refletindo previsões consistentes.

A integração de aprendizado de máquina com modelagens tridimensionais marca um avanço significativo no dimensionamento de dispositivos hidráulicos em barragens, ao combinar os resultados das simulações numéricas com a capacidade preditiva das técnicas de aprendizado de máquina. Essa abordagem se destaca especialmente em cenários onde há escassez de dados reais, permitindo realizar análises eficientes, otimizando o processo de dimensionamento mesmo na ausência de informações práticas obtidas diretamente de experimentos ou observações de campo. As redes neurais artificiais (RNAs) são indicadas para cenários com alta complexidade e relações não lineares, desde que o overfitting seja controlado, enquanto as árvores de decisão e as florestas aleatórias oferecem um bom desempenho e estabilidade em situações com alta variabilidade e custo computacional reduzido. Por sua vez, o XGBoost é ideal para aplicações que demandam alta precisão e generalização, embora exija um maior custo computacional devido à sua abordagem iterativa e ao ajuste fino de hiperparâmetros.

A pesquisa realizada é fundamental para o avanço no dimensionamento de dispositivos de controle de percolação interna em barragens de terra. A integração de simulações computacionais tridimensionais com aprendizado de máquina permite determinar de forma precisa o comprimento ideal dos dispositivos, como o abraço, para mitigar gradientes hidráulicos e prevenir erosão interna. Essa abordagem inovadora, utilizando técnicas como redes neurais artificiais e XGBoost, oferece uma solução eficiente, especialmente em cenários com dados limitados. Os resultados obtidos abrem caminho para futuras pesquisas que integrem dados de campo e experimentais, aprimorando a precisão dos modelos e sua aplicabilidade em condições reais de operação.

5.1.1 Recomendações para estudos futuros

1. A incorporação de dados de instrumentação em campo, como piezômetros e medidores de vazão, é fundamental para aprimorar a acuracidade dos modelos preditivos desenvolvidos. Esses dados empíricos são essenciais para a validação e calibração dos modelos, aumentando sua precisão e aplicabilidade prática. Além disso, permitem o monitoramento contínuo do desempenho do dispositivo abraço, proporcionando uma avaliação eficiente em diferentes condições operacionais.

2. Ensaios laboratoriais em escala reduzida, como modelos físicos, permitem a reprodução controlada dos fluxos nas interfaces solo-estrutura, incluindo testes de permeabilidade, simulações de erosão interna e a avaliação de dispositivos que simulam o abraço. Os dados obtidos nesses ensaios enriquecem as análises computacionais, possibilitando o ajuste refinado de parâmetros de entrada nos modelos preditivos.

3. Estudos futuros devem incluir condições mais complexas, como solos heterogêneos com anisotropias de permeabilidade, simulações de fluxo transiente em eventos extremos e avaliação de múltiplos dispositivos em série. Esses cenários expandem a aplicabilidade dos modelos para projetos de engenharia mais complexos.

4. Embora o XGBoost tenha mostrado bom desempenho, recomenda-se explorar outras técnicas, como deep learning para relações não lineares e grandes volumes de dados, algoritmos de ensemble avançados como LightGBM e CatBoost, e modelos probabilísticos para estimar a incerteza das previsões.

REFERÊNCIAS

ALI, M. M. Modified limited-memory broyden-fletcher-goldfarb-shanno algorithm for unconstrained optimization problem. **Indonesian Journal of Electrical Engineering and Computer Science**, Yogyakarta, Indonésia, v. 24, n. 2, p. 1027–1034, 2021.

BAGHBANI, A.; CHOUDHURY, T.; COSTA, S.; REINER, J. Application of artificial intelligence in geotechnical engineering: a state-of-the-art review. **Earth-Science Reviews**, Amsterdã, Holanda, v. 228, p. 103991, 2022.

BEAR, J. **Dynamics of fluids in porous media**. New York, EUA: Elsevier, 1972.

BEIRANVAND, B.; RAJAEE, T. Application of artificial intelligence-based single and hybrid models in predicting seepage and pore water pressure of dams: a state-of-the-art review. **Advances in Engineering Software**, Oxford, Reino Unido, v. 173, p. 103268, 2022.

BELLO, O.; HOLZMANN, J.; YAQOOB, T.; TEODORIU, C. Application of artificial intelligence methods in drilling system design and operations: a review of the state of the art. **Journal of Artificial Intelligence and Soft Computing Research**, Varsóvia, Polônia, v. 5, n. 2, p. 121–139, 2015.

BILALI, A. E.; MOUKHLISS, M.; TALEB, A.; NAFII, A.; ALABJAH, B.; BROUZIYNE, Y.; MAZIGH N.AND TEZNINE, K.; MHAMED, M. Predicting daily pore water pressure in embankment dam: empowering machine learning-based modeling. **Environmental Science and Pollution Research**, Berlim, Alemanha, v. 29, n. 31, p. 47382–47398, 2022.

BISHOP, C. M. **Pattern recognition and machine learning**. 1. ed. New York, EUA: Springer, 2006. ISBN 978-0-387-31073-2.

BRASIL. **Lei n 12.334, de 20 de setembro de 2010. Lei de segurança de barragens**. 2010. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2007-2010/2010/lei/l12334.htm>. Acesso em: 19 fev. 2026.

BREIMAN, L. Bagging predictors. **Machine Learning**, Nova Iorque, EUA, v. 24, n. 2, p. 123–140, 1996.

____. Random forests. **Machine Learning**, Nova Iorque, EUA, v. 45, n. 1, p. 5–32, 2001.

CAN R.AND KOCAMAN, S.; GÖKCEOLU, C. A comprehensive assessment of xgboost algorithm for landslide susceptibility mapping in the upper basin of atatürk dam, turkey. **Applied Sciences**, Basileia, Suíça, v. 11, n. 11, p. 4993, 2021.

CEARÁ (Estado). **Projeto executivo do paramento central da barragem Castanhão em concreto compactado a rolo: Volume 1: Relatório do projeto**. Fortaleza, 1999.

____. **Projeto executivo do paramento central da barragem Castanhão em concreto compactado a rolo: Volume 2: Desenhos**. Fortaleza, 1999.

____. **Projeto executivo do paramento central da barragem Castanhão em concreto compactado a rolo: Volume 5: Especificações técnicas**. Fortaleza, 1999.

CHAPUIS, R. P.; AUBERTIN, M. Predicting the coefficient of permeability of soils using the kozeny-carman equation. **Canadian Geotechnical Journal**, Ottawa, Canadá, v. 40, n. 3, p. 616–628, 2003.

CHEN, T.; GUESTRIN, C. Xgboost: a scalable tree boosting system. In: **Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining**. San Francisco, CA, EUA: [s.n.], 2016. p. 785–794.

CHINA WATER CONSERVANCY YEARBOOK COMPILATION COMMITTEE. **China water conservancy yearbook**. Pequim, China: China Water Conservancy and Hydropower Press, 2008.

COURNAPEAU, D. **Scikit-learn: machine learning in Python**. 2007. Google Summer of Code Project [projeto de código-fonte]. Disponível em: https://scikit-learn.org/stable/about.html. Acesso em: 26 fev. 2026.

CRUZ, P. T. d. **100 barragens brasileiras: casos históricos, materiais de construção, projeto**. São Paulo: Oficina de Textos, 1996. ISBN 978-85-8623-802-4.

DAS, B. M.; SOBHAN, K. **Fundamentos de engenharia geotécnica**. São Paulo: Thomson Learning, 2007.

DAWSON, C. W.; WILBY, R. L. Hydrological modelling using artificial neural networks. **Progress in Physical Geography**, [S.l.], v. 25, n. 1, p. 80–108, 2001.

DEMUTH, H. B.; BEALE, M. H.; JESS, O. D.; HAGAN, M. T. **Neural network design**. [S.l.]: Martin Hagan, 2014.

FELL, R. Seepage, internal erosion and piping. In: FELL, R.; MACGREGOR, P.; SPENCER, R. (Ed.). **Geotechnical engineering of dams**. Londres, Reino Unido: CRC Press, 2005. p. 181–236.

FELL, R.; FOSTER, M.; CYGANIEWICZ, J.; SILLS, G.; VROMAN, N.; DAVIDSON, R. **Risk analysis for dam safety: a unified method for estimating probabilities of failure of embankment dams by internal erosion and piping**. North Sydney, Austrália, 2008. 4-4 p.

FERDOS, F. **Internal erosion phenomena in embankment dams: throughflow and internal erosion mechanisms**. Tese (Tese (Doutorado em Engenharia Civil)) — Universidade Federal do Ceará, Fortaleza, CE, 2016. Defendida em 2016.

FLORES-BERRONES, R.; RAMIREZ-REYNAGA, M.; MACARI, E. J. Internal erosion and rehabilitation of an earth-rock dam. **Journal of Geotechnical and Geoenvironmental Engineering**, Reston, VA, EUA, v. 137, n. 2, p. 150–160, 2011.

GAIOTO, N. **Sistemas de controle de percolação de água em projetos de barragens de terra**. Tese (Tese (Livre-Docência em Engenharia Civil)) — Universidade Federal de São Carlos, São Carlos, SP, 1992.

GENUCHTEN, M. T. van. A closed-form equation for predicting the hydraulic conductivity of unsaturated soils. **Soil Science Society of America Journal**, [S.l.], v. 44, n. 5, p. 892–898, 1980.

GOH, A. T. C. Empirical design in geotechnics using neural networks. **Géotechnique**, Londres, Reino Unido, v. 45, n. 4, p. 709–714, 1995. DOI: 10.1680/geot.1995.45.4.709.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep learning**. Cambridge, MA, EUA: MIT Press, 2016. Disponível em: https://www.deeplearningbook.org. Acesso em: 17 jun. 2025.

GOYAL, M. K.; OJHA, C. Estimation of scour downstream of a ski-jump bucket using support vector and m5 model tree. **Water Resources Management**, Heidelberg, Alemanha, v. 25, n. 10, p. 2177–2195, 2011.

GU, C.; WU, B.; CHEN, Y. A high-robust displacement prediction model for super-high arch dams integrating wavelet de-noising and improved random forest. **Water**, Basileia, Suíça, v. 15, n. 7, p. 1271, 2023.

GÉRON, A. **Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow**. Sebastopol, CA, EUA: O'Reilly Media, Inc., 2022.

HAN, J.; KAMBER, M.; PEI, J. **Data mining: concepts and techniques**. 3. ed. Burlington, MA, EUA: Morgan Kaufmann, 2011.

HASTIE, T. **The elements of statistical learning: data mining, inference, and prediction**. Nova Iorque, EUA: Springer, 2009.

HAYKIN, S. **Redes neurais: princípios e prática**. Porto Alegre: Bookman, 2001.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural Computation**, Cambridge, MA, EUA, v. 9, n. 8, p. 1735–1780, 1997.

HUANG, D.; HUANG, W. B.; KE, C. Y.; SONG, Y. X. Experimental investigation on seepage erosion of the soil-rock interface. **Bulletin of Engineering Geology and the Environment**, [S.l.], v. 80, p. 3115–3137, 2021.

IACOMCAFÉ. **A importância da validação cruzada em machine learning**. 2024. Disponível em: <https://iacomcafe.com.br/importancia-validacao-cruzada-machine-learning/>. Acesso em: 16 jan. 2025.

ICOLD. **Internal erosion of existing dams, levees and dikes, and their foundations**. Paris, França: Comité International des Grands Barrages, 2013. (Bulletin 164, Volume 1: Internal erosion processes and engineering assessment).

IPNET. **A importância da normalização e padronização dos dados em Machine Learning**. 2023. Disponível em: https://medium.com/ipnet-growth-partner/padronizacao-normalizacao-dados-machine-learning-f8f29246c12. Acesso em: 9 jan. 2025.

JAIN, A. K.; MAO, J.; MOHIUDDIN, K. M. Artificial neural networks: a tutorial. **Computer**, Los Alamitos, CA, EUA, v. 29, n. 3, p. 31–44, 1996.

JANG, H.; TOPAL, E. Optimizing overbreak prediction based on geological parameters comparing multiple regression analysis and artificial neural network. **Tunnelling and Underground Space Technology**, Amsterdã, Holanda, v. 38, p. 161–169, 2013.

JUNIOR, R. N.; PRETTI, L. A. **Trabalho de avaliação: análise dos resíduos na regressão linear múltipla pelo método dos mínimos quadrados ordinários**. 2025.

KAVZOGLU, T.; COLKESEN, I. A kernel functions analysis for support vector machines for land cover classification. **International Journal of Applied Earth Observation and Geoinformation**, Amsterdã, Holanda, v. 11, n. 5, p. 352–359, 2009.

KINGMA, D. P.; BA, J. Adam: a method for stochastic optimization. In: **International Conference on Learning Representations (ICLR)**. San Diego, EUA: [s.n.], 2015. Disponível em: https://arxiv.org/abs/1412.6980. Acesso em: 20 jan. 2025.

KOHAVI, R. A study of cross-validation and bootstrap for accuracy estimation and model selection. In: **International Joint Conference on Artificial Intelligence (IJCAI)**. Montreal, Canadá: Morgan Kaufmann, 1995. p. 1137–1143.

LAMBE, T. W.; WHITMAN, R. V. **Mecânica dos solos**. São Paulo: McGraw-Hill, 1979.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, Londres, Reino Unido, v. 521, n. 7553, p. 436–444, 1998.

LEDERER, J. **Activation functions in artificial neural networks: a systematic overview**. 2021. Disponível em: https://arxiv.org/abs/2101.09957. Acesso em: 10 jan. 2025.

LEI, L.; ZHOU, Y.; HUANG, H.; LUO, Q. Extreme learning machine using improved gradient-based optimizer for dam seepage prediction. **Arabian Journal for Science and Engineering**, Springer, v. 48, n. 8, p. 9693–9712, 2023.

LEONHARDT, M. **Barragens de terra e enrocamento: projeto, construção e comportamento**. São Paulo: Oficina de Textos, 2003.

LI, Y.-L.; YIN, Q.-G.; ZHANG, Y.; ZHOU, H. Deformation prediction model of concrete face rockfill dams based on an improved random forest model. **Water Science and Engineering**, Amsterdã, Holanda, v. 16, n. 4, p. 390–398, 2023.

LIN, F.; ZHOU, C.; GAO, X. Study on anti-seepage treatment of bayi reservoir dam. **Chinese Journal of Rock Mechanics and Engineering**, Pequim, China, v. 22, n. S2, p. 2925–2928, 2003.

LOH, W.-Y. Classification and regression trees. **Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery**, Hoboken, NJ, EUA, v. 1, n. 1, p. 14–23, 2011.

LUO, Y.; ZHAN, M.; SHENG, J.; WU, Q. Hydro-mechanical coupling mechanism on joint of clay core-wall and concrete cut-off wall. **Journal of Central South University**, Springer, [S.l.], v. 20, n. 9, p. 2578–2585, 2013.

MAIER, H. R.; DANDY, G. C. Neural networks for the prediction and forecasting of water resources variables: a review of modelling issues and applications. **Environmental Modelling & Software**, Amsterdã, Holanda, v. 15, n. 1, p. 101–124, 2000.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The Bulletin of Mathematical Biophysics**, Nova Iorque, EUA, v. 5, p. 115–133, 1943.

MCKINNEY, W. Data structures for statistical computing in python. In: **Python in Science Conference (SciPy)**. Austin, TX, EUA: [s.n.], 2010. p. 51–56.

MICHELIS, A. **Traditional versus non-traditional boosting algorithms**. Dissertação (Dissertação (Mestrado em Ciência da Computação)) — University of Manchester, Manchester, Reino Unido, 2012. Defendida em 2012.

MITCHELL, T. M. **Machine learning**. Nova Iorque, EUA: McGraw-Hill, 1997. ISBN 978-0070428072.

MONTGOMERY, D. C.; PECK, E. A.; VINING, G. G. **Introduction to linear regression analysis**. 5. ed. Hoboken, NJ, EUA: Wiley, 2012. ISBN 978-0-470-54281-1.

OLIPHANT, T. E. **A guide to NumPy**. Austin, TX, EUA: Trelgol Publishing, 2006.

OLIVEIRA, P. H. A. **Problemas hidrogeológicos em barragens envolvendo o mecanismo de retroerosão tubular**. Tese (Tese (Doutorado em Engenharia Civil)) — Universidade Federal do Rio de Janeiro, Rio de Janeiro, 2012. Defendida em 2012.

OMARZAI, F. **XGBoost classification in depth**. 2024. Disponível em: https://medium.com/@fraidoonomarzai99/xgboost-classification-in-depth-979f11ef4bf9. Acesso em: 12 jan. 2025.

PEDREGOSA, F.; VAROQUAUX, G.; GRAMFORT, A.; MICHEL, V.; THIRION, B.; GRISEL, O.; BLONDEL, M.; PRETTENHOFER, P.; WEISS, R.; DUBOURG, V.; VANDERPLAS, J.; PASSOS, A.; COURNAPEAU, D.; BRUCHER, M.; PERROT, M.; DUCHESNAY, E. Scikit-learn: machine learning in python. **Journal of Machine Learning Research**, Cambridge, MA, EUA, v. 12, p. 2825–2830, 2011.

PINTO, C. S. **Curso básico de mecânica dos solos**. São Paulo: Oficina de Textos, 2002.

PLEVRIS, V.; SOLORZANO, G.; BAKAS, N. P.; SEGHIER, M. E. A. B. Investigation of performance metrics in regression analysis and machine learning-based prediction models. In: **European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS Congress)**. Oslo, Noruega: ECCOMAS, 2022.

POSO, F. D.; JESUS, K. L. M. d. Neural network-particle swarm optimization model for predicting slope stability of homogeneous earth dams. In: **IEEE 14th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)**. Manila, Filipinas: IEEE, 2022. p. 1–5.

RETEC. **Soil hydraulic and retention models, version 6.02**. [S.l.]: RETEC Software, 2009.

REYNOLDS, O. An experimental investigation of the circumstances which determine whether the motion of water shall be direct or sinuous, and of the law of resistance in parallel channels. **Philosophical Transactions of the Royal Society of London**, Londres, Reino Unido, v. 174, p. 935–982, 1883.

SALAZAR, F.; AL. et. An empirical comparison of machine learning techniques for dam behaviour modelling. **Structural Safety**, Amsterdã, Holanda, v. 56, p. 9–17, 2015.

SAMADI, M.; AFSHAR, M. H.; JABBARI, E.; SARKARDEH, H. Application of multivariate adaptive regression splines and classification and regression trees to estimate wave-induced scour depth around pile groups. **Iranian Journal of Science and Technology, Transactions of Civil Engineering**, Teerã, Irã, v. 44, p. 447–459, 2020.

SAMADI, M.; JABBARI, E.; AZAMATHULLA, H. M. Assessment of m5 model tree and classification and regression trees for prediction of scour depth below free overfall spillways. **Neural Computing and Applications**, Londres, Reino Unido, v. 24, p. 357–366, 2014.

SARKER, I. H. Machine learning: algorithms, real-world applications and research directions. **SN Computer Science**, Heidelberg, Alemanha, v. 2, n. 3, p. 160, 2021.

SARÉ, A. **Análise das condições de fluxo na barragem de Curuá-Una, Pará**. 167 p. Tese (Tese (Doutorado em Engenharia Civil)) — Pontifícia Universidade Católica do Rio de Janeiro, Departamento de Engenharia Civil, Rio de Janeiro, 2003. Defendida em 2003.

SCHAPIRE, R. E. The boosting approach to machine learning: an overview. In: **Nonlinear estimation and classification**. Nova Iorque, EUA: Springer, 2003. p. 149–171.

SEED, H. B.; DUNCAN, J. M. The failure of teton dam. **Engineering Geology**, Amsterdã, Holanda, v. 24, n. 1-4, p. 173–205, 1987.

SEEQUENT. Software, **GeoStudio: versão 24.2.1**. [S.l.]: Bentley Systems, 2024. Disponível em: https://www.seequent.com/products-solutions/geostudio/. Acesso em: 24 abr. 2025.

SEGAL, M. R. Machine learning benchmarks and random forest regression. **Bioinformatics**, Oxford, Reino Unido, v. 20, n. 11, p. 2007–2015, 2004.

SELLMEIJER, H.; CRUZ, J. L. d. l.; BEEK, V. M. v.; KNOEFF, H. Fine-tuning of the backward erosion piping model through small-scale, medium-scale and ijkdijk experiments. **European Journal of Environmental and Civil Engineering**, Abingdon, Reino Unido, v. 15, n. 8, p. 1139–1154, 2011.

SHERARD, J. L. [design and construction problems in earth and earth-rock dams]. In: **Earth and earth-rock dams**. Nova Iorque, EUA: John Wiley & Sons, 1963.

SHI, N.; LI, Y.; WEN, L.; ZHANG, Y. Rapid prediction of landslide dam stability considering the missing data using xgboost algorithm. **Landslides**, [S.l.], v. 19, n. 12, p. 2951–2963, 2022.

SILVA, D. L.; JESUS, K. L. M. d.; ADINA, E. M.; MANGROBANG, D. V.; ESCALANTE, M. D.; SUSI, N. A. M. Prediction of tensile strength and erosional effectiveness of natural geotextiles using artificial neural network. In: **International Conference on Computer and Automation Engineering (ICCAE)**. [S.l.]: IEEE, 2021. p. 121–127.

SOUSA, A. C. M. **Influência do gradiente hidráulico na erosão interna em interfaces solo/concreto**. Dissertação (Trabalho de Conclusão de Curso (Graduação em Engenharia Civil)) — Universidade Estadual do Vale do Acaraú, Centro de Ciências Exatas e Tecnologia, Sobral, CE, 2021. Defendido em 2021.

SOUSA, A. D. Análise do comportamento hidráulico em interfaces solo/estrutura em barragens de terra. **Revista de Engenharia Geotécnica**, [S.l.], v. 35, n. 2, p. 45–59, 2013.

SOWERS, G. F. **Earth and rockfill dam engineering**. Nova Deli, Índia: Asia Publishing House, 1962. 283 p.

SU, Y.; WENG, K.; LIN, C.; ZHENG, Z. An improved random forest model for the prediction of dam displacement. **IEEE Access**, [S.l.], v. 9, p. 9142–9153, 2021.

TAYFUR, G.; SWIATEK, D.; WITA, A.; SINGH, V. P. Case study: finite element and artificial neural network models for flow through jeziorsko earthfill dam in poland. **Journal of Hydraulic Engineering**, Reston, VA, EUA, v. 131, n. 6, p. 431–440, 2005.

TERZAGHI, K. **Theoretical soil mechanics**. Nova Iorque, EUA: John Wiley & Sons, 1943.

TERZAGHI, K.; PECK, R. B.; MESRI, G. **Soil mechanics in engineering practice**. Nova Iorque, EUA: John Wiley & Sons, 1969.

TOPOK, Z. F.; CIGIZOGLU, H. K. Predicting longitudinal dispersion coefficient in natural streams by artificial intelligence methods. **Hydrological Processes**, Chichester, Reino Unido, v. 22, n. 20, p. 4106–4129, 2008.

UNES, F. Dam reservoir level modeling by neural network approach: a case study. **Neural Network World**, Praga, República Tcheca, v. 20, n. 4, p. 461–472, 2010.

USBR. **Internal erosion risks for embankments and foundations**. [S.l.: s.n.]: [s.n.], 2019.

WANG, S.; CHEN, J.-S.; HE, H.-Q.; HE, W.-Z. Experimental study on piping in sandy gravel foundations considering effect of overlying clay. **Water Science and Engineering**, Amsterdã, Holanda, v. 9, n. 2, p. 165–171, 2016.

XIE, Q.; LIU, J.; HAN, B.; LI, H.; LI, Y.; LI, X. Critical hydraulic gradient of internal erosion at the soilstructure interface. **Processes**, MDPI, Basileia, Suíça, v. 6, n. 7, p. 92, 2018.

XU, Z. X.; LI, J. Y. Short-term inflow forecasting using an artificial neural network model. **Hydrological Processes**, [S.l.], v. 16, n. 12, p. 2423–2439, 2002.

XUE, X.; YANG, X.; CHEN, X. Estimating piping potential in earth dams and levees using generalized neural networks. **Acta Geotechnica Slovenica**, Liubliana, Eslovênia, v. 11, n. 2, p. 59–69, 2014.

ZHANG, K.; GU, C.; ZHU, Y.; CHEN, S.; DAI, B.; LI, Y.; SHU, X. A novel seepage behavior prediction and lag process identification method for concrete dams using hgwo-xgboost model. **IEEE Access**, [S.l.], v. 9, p. 23311–23325, 2021.

APÊNDICE A – CÓDIGO PARA TREINAMENTO E AVALIAÇÃO DE DIFERENTES CONFIGURAÇÕES DE REDES NEURAIS ARTIFICIAIS

A.1 Importação de Módulos e Carregamento de Dados

```
# Importar módulos necessários
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.preprocessing import RobustScaler, StandardScaler,
    MinMaxScaler
from sklearn.decomposition import PCA
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error,
    explained_variance_score, r2_score
from sklearn.model_selection import KFold, cross_val_score,
    GridSearchCV, train_test_split, learning_curve

# Carregar os dados
df2 = pd.read_csv('/content/drive/MyDrive/doutorado/dados_10.csv',
    sep=',', encoding='iso-8859-1')
df = df2.dropna()

# Separar variáveis independentes e dependentes
X = df.iloc[:, 0:16].values
y = df.iloc[:, 16].values
```

Listing A.1 – Importação de Módulos e Carregamento dos Dados

A.2 Pré-processamento de Dados

```
# Dividir os dados em treino e teste
```

```
x_treino, x_teste, y_treino, y_teste = train_test_split(X, y,
    test_size=0.2, random_state=42)

# Normalização das variáveis de entrada
x_scaler = MinMaxScaler()
x_treino_scaler = x_scaler.fit_transform(x_treino)
x_teste_scaler = x_scaler.transform(x_teste)

# Normalização das variáveis de saída
y_scaler = MinMaxScaler()
y_treino_scaler = y_scaler.fit_transform(y_treino.reshape(-1, 1))
y_teste_scaler = y_scaler.transform(y_teste.reshape(-1, 1))
```

Listing A.2 – Divisão e Normalização dos Dados

A.3 Treinamento e Avaliação dos Modelos

```
# Definir configurações de modelos
activation_functions = ['relu', 'logistic', 'tanh']
solvers = ['adam', 'lbfgs']
hidden_layer_configurations = [(7, 7, 7), (10, 10), (30, 20), (50,
    50), (100, 50), (100, 100), (100,)]

# Loop para treinar os modelos
results = []

for activation in activation_functions:
    for solver in solvers:
        for hidden_layer_sizes in hidden_layer_configurations:
            print(f"Treinando modelo com ativação={activation},
                solver={solver}, camadas={hidden_layer_sizes}")
            try:
                model = MLPRegressor(hidden_layer_sizes=
                    hidden_layer_sizes, activation=activation,
```

```
                                solver=solver, max_iter=2000,
                                    random_state=42)
            model.fit(x_treino_scaler, y_treino_scaler.ravel())

            # Avaliação
            r2_treino = model.score(x_treino_scaler,
                y_treino_scaler)
            r2_teste = model.score(x_teste_scaler, y_teste_scaler
                )
            previsoes_teste_scaler = model.predict(x_teste_scaler
                )
            previsoes_teste_inverse = y_scaler.inverse_transform(
                previsoes_teste_scaler.reshape(-1, 1))

            # Cálculo das métricas
            mae = mean_absolute_error(y_teste,
                previsoes_teste_inverse)
            rmse = np.sqrt(mean_squared_error(y_teste,
                previsoes_teste_inverse))
            mape = np.mean(np.abs((y_teste -
                previsoes_teste_inverse.flatten()) / y_teste)) *
                100
            explained_var = explained_variance_score(y_teste,
                previsoes_teste_inverse)

            # Armazenar resultados
            results.append({
                'activation': activation,
                'solver': solver,
                'hidden_layer_sizes': hidden_layer_sizes,
                'r2_train': r2_treino,
                'r2_test': r2_teste,
                'mae': mae,
                'rmse': rmse,
```

```
                    'mape': mape,
                    'explained_variance': explained_var
                })

            except Exception as e:
                print(f"Erro ao treinar modelo: {e}")
```

Listing A.3 – Treinamento de Modelos com Diferentes Configurações

A.4 Visualização e Análise dos Resultados

```
# Treinar o melhor modelo
melhor_modelo = MLPRegressor(hidden_layer_sizes=(30, 20), activation='tanh',
                             solver='lbfgs', max_iter=2000,
                                random_state=42)
melhor_modelo.fit(x_treino_scaler, y_treino_scaler.ravel())

# Previsões e resíduos
previsoes_teste_scaler = melhor_modelo.predict(x_teste_scaler)
previsoes_teste_inverse = y_scaler.inverse_transform(
    previsoes_teste_scaler.reshape(-1, 1))
residuos = y_teste - previsoes_teste_inverse.flatten()

# Gráfico de dispersão
plt.figure(figsize=(8,6))
plt.scatter(y_teste, previsoes_teste_inverse, alpha=0.7, color="blue")
plt.plot([y_teste.min(), y_teste.max()], [y_teste.min(), y_teste.max()], color="red", linestyle="--", linewidth=2)
plt.title("Valores Reais vs. Previstos (Melhor Modelo)")
plt.xlabel("Valores Reais")
plt.ylabel("Valores Previstos")
plt.grid()
plt.show()
```

```
20
21  # Histograma dos resíduos
22  plt.figure(figsize=(8,6))
23  sns.histplot(residuos, kde=True, bins=30, color='gray', alpha=0.7)
24  plt.title("Histograma dos Resíduos: Melhor Modelo")
25  plt.xlabel("Resíduos")
26  plt.ylabel("Frequência")
27  plt.grid()
28  plt.show()
29
30  # Comparação dos Rš dos três melhores modelos
31  modelos = ["Melhor modelo", "Segundo melhor modelo", "Terceiro melhor
        modelo"]
32  r2_train_vals = [
33      melhor_modelo.score(x_treino_scaler, y_treino_scaler),
34      segundo_modelo.score(x_treino_scaler, y_treino_scaler),
35      terceiro_modelo.score(x_treino_scaler, y_treino_scaler)
36  ]
37  r2_test_vals = [
38      melhor_modelo.score(x_teste_scaler, y_teste_scaler),
39      segundo_modelo.score(x_teste_scaler, y_teste_scaler),
40      terceiro_modelo.score(x_teste_scaler, y_teste_scaler)
41  ]
42
43  # Gráfico de barras
44  x = np.arange(len(modelos))
45  width = 0.35
46
47  plt.figure(figsize=(10,6))
48  plt.bar(x - width/2, r2_train_vals, width, label="Treinamento", color
        ="blue")
49  plt.bar(x + width/2, r2_test_vals, width, label="Teste", color="
        orange")
```

```
50 plt.title("Comparação Rš (Treinamento vs Teste) dos Três Melhores
       Modelos")
51 plt.xlabel("Modelos")
52 plt.ylabel("Rš")
53 plt.xticks(x, modelos)
54 plt.legend()
55 plt.grid(axis="y")
56 plt.show()
```

Listing A.4 – Gráficos de Análise do Melhor Modelo

APÊNDICE B – CÓDIGO PARA TREINAMENTO E AVALIAÇÃO DE ÁRVORES DE DECISÃO

B.1 Importação de Módulos e Carregamento de Dados

```
# Importação de bibliotecas
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.tree import DecisionTreeRegressor
from sklearn.model_selection import train_test_split, KFold,
   cross_val_score, learning_curve
from sklearn.metrics import (mean_absolute_error,
                             mean_squared_error,
                             mean_absolute_percentage_error,
                             explained_variance_score,
                             r2_score)

# Carregamento dos dados
df = pd.read_csv('/content/drive/MyDrive/doutorado/dados_10.csv',
                 sep=',',
                 encoding='iso-8859-1')

# Preparação dos dados
X = df.iloc[:, 0:16].values
y = df.iloc[:, 16].values

# Divisão treino-teste
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    test_size=0.3,
                                                    random_state=0)
```

Listing B.1 – Importação de Bibliotecas e Carregamento dos Dados

B.2 Treinamento e Avaliação do Modelo

```
# Parâmetros para teste
max_depth_options = [3, 5, 7, 10, 15, 20, None]
random_states = [0, 5, 10, 42]

# Estrutura para armazenar resultados
results = []

for depth in max_depth_options:
    for state in random_states:
        print(f"Treinando com max_depth={depth}, random_state={state}
            ")
        try:
            model = DecisionTreeRegressor(max_depth=depth,
                                          random_state=state)
            model.fit(X_train, y_train)

            # Métricas de avaliação
            y_pred = model.predict(X_test)
            metrics = {
                'max_depth': depth,
                'random_state': state,
                'r2_train': model.score(X_train, y_train),
                'r2_test': model.score(X_test, y_test),
                'mae': mean_absolute_error(y_test, y_pred),
                'rmse': np.sqrt(mean_squared_error(y_test, y_pred)),
                'mape': mean_absolute_percentage_error(y_test, y_pred
                    )*100,
                'explained_variance': explained_variance_score(y_test
                    , y_pred)
            }
            results.append(metrics)

        except Exception as e:
```

```
31                print(f"Erro na configuração {depth}-{state}: {str(e)}")
```

Listing B.2 – Configuração e Treinamento da Árvore de Decisão

B.3 Análise e Visualização dos Resultados

```
# Análise dos resultados
results_df = pd.DataFrame(results)

# Seleção dos melhores modelos
top_models_mape = results_df.sort_values('mape').head(5)
top_models_rmse = results_df.sort_values('rmse').head(5)

# Treinamento do melhor modelo
best_model = DecisionTreeRegressor(max_depth=10, random_state=42)
best_model.fit(X_train, y_train)

# Visualizações
plt.figure(figsize=(10,6))
plt.scatter(y_test, best_model.predict(X_test), alpha=0.6)
plt.plot([y.min(), y.max()], [y.min(), y.max()], 'r--')
plt.title("Valores Reais vs Previstos")
plt.xlabel("Real")
plt.ylabel("Previsto")
plt.grid()
plt.show()

# Curva de aprendizado
train_sizes, train_scores, test_scores = learning_curve(
    best_model, X, y, cv=5, scoring='r2', n_jobs=-1,
    train_sizes=np.linspace(0.1, 1.0, 10))

plt.plot(train_sizes, np.mean(train_scores, axis=1), label="Treino")
plt.plot(train_sizes, np.mean(test_scores, axis=1), label="Validação"
   )
```

```
plt.title("Curva de Aprendizado")
plt.xlabel("Tamanho do Conjunto de Treino")
plt.ylabel("RŠ Score")
plt.legend()
plt.grid()
plt.show()
```

Listing B.3 – Seleção e Visualização do Melhor Modelo

APÊNDICE C – CÓDIGO PARA TREINAMENTO E AVALIAÇÃO DE FLORESTAS ALEATÓRIAS

C.1 Configuração do Modelo

```
from sklearn.ensemble import RandomForestRegressor

# Parâmetros para otimização
param_grid = {
    'n_estimators': [50, 100, 200],
    'max_depth': [5, 10, 15, None],
    'min_samples_split': [2, 5, 10],
    'random_state': [42]
}

# Modelo base
rf_model = RandomForestRegressor()

# Busca de parâmetros
grid_search = GridSearchCV(estimator=rf_model,
                           param_grid=param_grid,
                           cv=5,
                           scoring='r2',
                           n_jobs=-1)
grid_search.fit(X_train, y_train)

# Melhores parâmetros
best_params = grid_search.best_params_
```

Listing C.1 – Implementação da Floresta Aleatória

C.2 Avaliação do Modelo

```
# Modelo final com melhores parâmetros
```

```
final_model = RandomForestRegressor(**best_params)
final_model.fit(X_train, y_train)

# Métricas de desempenho
y_pred = final_model.predict(X_test)
metrics = {
    'r2_train': final_model.score(X_train, y_train),
    'r2_test': final_model.score(X_test, y_test),
    'mae': mean_absolute_error(y_test, y_pred),
    'rmse': np.sqrt(mean_squared_error(y_test, y_pred)),
    'mape': mean_absolute_percentage_error(y_test, y_pred)*100
}

# Importância das features
feature_importance = pd.DataFrame({
    'feature': df.columns[:16],
    'importance': final_model.feature_importances_
}).sort_values('importance', ascending=False)
```

Listing C.2 – Avaliação da Floresta Aleatória

C.3 Visualização dos Resultados

```
# Gráfico de importância de features
plt.figure(figsize=(12,8))
sns.barplot(x='importance', y='feature', data=feature_importance)
plt.title("Importância das Variáveis")
plt.show()

# Validação cruzada
cv_scores = cross_val_score(final_model, X, y, cv=10, scoring='r2')
print(f"Rš médio: {np.mean(cv_scores):.4f} (š{np.std(cv_scores):.4f})
    ")

# Comparação com outros modelos
```

```
12 models = {
13     'Árvore': best_model,
14     'Floresta': final_model
15 }
16
17 for name, model in models.items():
18     y_pred = model.predict(X_test)
19     print(f"\n{name}:")
20     print(f"Rš: {r2_score(y_test, y_pred):.4f}")
21     print(f"MAE: {mean_absolute_error(y_test, y_pred):.4f}")
```

Listing C.3 – Visualização da Floresta Aleatória

APÊNDICE D – CÓDIGO COMPLETO PARA TREINAMENTO E AVALIAÇÃO DO XGBOOST

D.1 Configuração Inicial e Pré-processamento

```
# Bibliotecas essenciais
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import xgboost as xgb
from sklearn.model_selection import train_test_split, KFold,
    cross_val_score
from sklearn.metrics import (mean_absolute_error,
                             mean_squared_error,
                             explained_variance_score,
                             r2_score)

# Configuração de visualização
plt.style.use('seaborn')
plt.rcParams['figure.figsize'] = (12, 8)
pd.set_option('display.max_columns', 50)

# Carregamento e preparação dos dados
data = pd.read_csv('/content/drive/MyDrive/doutorado/dados_10.csv',
                   sep=',',
                   encoding='iso-8859-1')

# Pré-processamento
data_clean = data.dropna()
X = data_clean.iloc[:, 0:16].values
y = data_clean.iloc[:, 16].values

# Divisão dos dados
```

```
X_train, X_test, y_train, y_test = train_test_split(
    X, y,
    test_size=0.3,
    random_state=42
)

# Conversão para formato DMatrix do XGBoost
dtrain = xgb.DMatrix(X_train, label=y_train)
dtest = xgb.DMatrix(X_test, label=y_test)
```

Listing D.1 – Importação de Bibliotecas e Preparação dos Dados

D.2 Otimização de Hiperparâmetros

```
# Grade de parâmetros para otimização
param_grid = {
    'max_depth': [3, 5, 7, 9],
    'learning_rate': [0.01, 0.05, 0.1, 0.2],
    'n_estimators': [100, 200, 300],
    'gamma': [0, 0.1, 0.2],
    'min_child_weight': [1, 3, 5],
    'subsample': [0.6, 0.8, 1.0],
    'colsample_bytree': [0.6, 0.8, 1.0]
}

# Configuração do modelo base
xgb_model = xgb.XGBRegressor(
    objective='reg:squarederror',
    random_state=42,
    n_jobs=-1
)

# Busca em grade com validação cruzada
from sklearn.model_selection import RandomizedSearchCV

```

```
grid_search = RandomizedSearchCV(
    estimator=xgb_model,
    param_distributions=param_grid,
    n_iter=50,
    cv=5,
    scoring='r2',
    verbose=1,
    n_jobs=-1,
    random_state=42
)

grid_search.fit(X_train, y_train)

# Resultados da otimização
best_params = grid_search.best_params_
print(f"Melhores parâmetros encontrados: {best_params}")
print(f"Melhor R2: {grid_search.best_score_:.4f}")

# DataFrame com resultados
cv_results = pd.DataFrame(grid_search.cv_results_)
cv_results.sort_values(by='rank_test_score').head(5)
```

Listing D.2 – Busca Sistemática de Melhores Parâmetros

D.3 Treinamento do Modelo Final

```
# Parâmetros finais
final_params = {
    **best_params,
    'objective': 'reg:squarederror',
    'random_state': 42
}

# Treinamento com early stopping
final_model = xgb.XGBRegressor(**final_params)
```

```
10
11 eval_set = [(X_train, y_train), (X_test, y_test)]
12 final_model.fit(
13     X_train, y_train,
14     eval_set=eval_set,
15     eval_metric=['rmse'],
16     early_stopping_rounds=50,
17     verbose=10
18 )
19
20 # Salvar modelo
21 final_model.save_model('xgboost_model.json')
```

Listing D.3 – Implementação e Treinamento do Modelo Otimizado

D.4 Avaliação do Modelo

```
1  # Métricas de desempenho
2  def evaluate_model(model, X, y_true):
3      y_pred = model.predict(X)
4      return {
5          'R2': r2_score(y_true, y_pred),
6          'MAE': mean_absolute_error(y_true, y_pred),
7          'RMSE': np.sqrt(mean_squared_error(y_true, y_pred)),
8          'Explained Variance': explained_variance_score(y_true, y_pred
               )
9      }
10
11 # Avaliação no conjunto de treino e teste
12 train_metrics = evaluate_model(final_model, X_train, y_train)
13 test_metrics = evaluate_model(final_model, X_test, y_test)
14
15 # Resultados
16 metrics_df = pd.DataFrame({
17     'Conjunto': ['Treino', 'Teste'],
```

```
    'R2': [train_metrics['R2'], test_metrics['R2']],
    'MAE': [train_metrics['MAE'], test_metrics['MAE']],
    'RMSE': [train_metrics['RMSE'], test_metrics['RMSE']],
    'Explained Variance': [train_metrics['Explained Variance'],
                           test_metrics['Explained Variance']]
})

print(metrics_df)

# Validação cruzada
kfold = KFold(n_splits=10, shuffle=True, random_state=42)
cv_scores = cross_val_score(
    final_model, X, y,
    cv=kfold,
    scoring='r2',
    n_jobs=-1
)

print(f"\nValidação Cruzada (10 folds):")
print(f"R2 Médio: {np.mean(cv_scores):.4f}")
print(f"Desvio Padrão: {np.std(cv_scores):.4f}")
```

Listing D.4 – Avaliação Completa do Modelo

D.5 Visualização dos Resultados

```
# 1. Curva de aprendizado durante o treinamento
results = final_model.evals_result()
epochs = len(results['validation_0']['rmse'])
x_axis = range(0, epochs)

fig, ax = plt.subplots(figsize=(12, 6))
ax.plot(x_axis, results['validation_0']['rmse'], label='Treino')
ax.plot(x_axis, results['validation_1']['rmse'], label='Teste')
ax.legend()
```

```
plt.ylabel('RMSE')
plt.xlabel('Número de Árvores')
plt.title('Curva de Aprendizado do XGBoost')
plt.grid()
plt.show()

# 2. Importância das features
xgb.plot_importance(
    final_model,
    max_num_features=15,
    height=0.8,
    importance_type='weight',
    title='Importância das Features (Weight)'
)
plt.tight_layout()
plt.show()

# 3. Gráfico de dispersão valores reais vs previstos
y_pred = final_model.predict(X_test)

plt.figure(figsize=(10, 10))
plt.scatter(y_test, y_pred, alpha=0.6)
plt.plot([y.min(), y.max()], [y.min(), y.max()], 'r--', linewidth=2)
plt.xlabel('Valores Reais')
plt.ylabel('Valores Previstos')
plt.title('Valores Reais vs Previstos - XGBoost')
plt.grid()
plt.show()

# 4. Histograma de resíduos
residuals = y_test - y_pred

plt.figure(figsize=(12, 6))
sns.histplot(residuals, kde=True, bins=30)
```

```
plt.xlabel('Resíduos')
plt.ylabel('Frequência')
plt.title('Distribuição dos Resíduos')
plt.grid()
plt.show()
```

Listing D.5 – Visualização Completa dos Resultados

D.6 Análise Comparativa com Outros Modelos

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.tree import DecisionTreeRegressor
from sklearn.neural_network import MLPRegressor
from time import time

# Modelos para comparação
models = {
    'Decision Tree': DecisionTreeRegressor(max_depth=5, random_state
       =42),
    'Random Forest': RandomForestRegressor(n_estimators=100,
       random_state=42),
    'Neural Network': MLPRegressor(hidden_layer_sizes=(50,),
                        max_iter=1000,
                        random_state=42),
    'XGBoost': final_model
}

# Avaliação comparativa
results = []
for name, model in models.items():
    start_time = time()
    model.fit(X_train, y_train)
    train_time = time() - start_time

    y_pred = model.predict(X_test)
```

```
24
25      results.append({
26          'Model': name,
27          'R2': r2_score(y_test, y_pred),
28          'MAE': mean_absolute_error(y_test, y_pred),
29          'RMSE': np.sqrt(mean_squared_error(y_test, y_pred)),
30          'Training Time (s)': train_time
31      })
32
33  # DataFrame comparativo
34  comparison_df = pd.DataFrame(results)
35  comparison_df = comparison_df.sort_values('R2', ascending=False)
36  print(comparison_df)
37
38  # Visualização comparativa
39  metrics = ['R2', 'MAE', 'RMSE', 'Training Time (s)']
40  fig, axes = plt.subplots(2, 2, figsize=(16, 12))
41  axes = axes.flatten()
42
43  for i, metric in enumerate(metrics):
44      sns.barplot(
45          x=metric,
46          y='Model',
47          data=comparison_df,
48          ax=axes[i],
49          palette='viridis'
50      )
51      axes[i].set_title(f'Comparação de {metric}')
52      axes[i].set_xlabel(metric)
53      axes[i].set_ylabel('')
54
55  plt.tight_layout()
56  plt.suptitle('Comparação entre Modelos de Regressão', y=1.02)
57  plt.show()
```

Listing D.6 – Comparação Sistemática com Outros Algoritmos

D.7 Interpretabilidade do Modelo com SHAP

```
import shap

# Configuração do interpretador SHAP
explainer = shap.Explainer(final_model)
shap_values = explainer(X_test)

# 1. Summary plot
plt.figure(figsize=(12, 8))
shap.summary_plot(
    shap_values,
    X_test,
    feature_names=data_clean.columns[:16],
    plot_type='dot',
    max_display=15
)
plt.title('SHAP Summary Plot - Impacto das Features')
plt.tight_layout()
plt.show()

# 2. Dependence plot para a feature mais importante
shap.dependence_plot(
    shap_values.abs.mean(0).argsort()[-1],
    shap_values.values,
    X_test,
    feature_names=data_clean.columns[:16],
    interaction_index='auto'
)

# 3. Waterfall plot para uma observação específica
```

```
sample_idx = 42  # Exemplo específico
shap.plots.waterfall(shap_values[sample_idx], max_display=10)
plt.title(f'Waterfall Plot - Observação {sample_idx}')
plt.tight_layout()
plt.show()

# 4. Force plot para múltiplas observações
shap.plots.force(shap_values[0:100], matplotlib=True)
plt.title('Force Plot - Primeiras 100 Observações')
plt.tight_layout()
plt.show()
```

Listing D.7 – Análise de Importância com SHAP Values

D.8 Exportação dos Resultados

```
import json
import joblib

# 1. Salvar métricas em JSON
metrics_dict = {
    'best_params': best_params,
    'train_metrics': train_metrics,
    'test_metrics': test_metrics,
    'cv_mean_r2': np.mean(cv_scores),
    'cv_std_r2': np.std(cv_scores)
}

with open('xgboost_metrics.json', 'w') as f:
    json.dump(metrics_dict, f, indent=4)

# 2. Salvar DataFrame com previsões
predictions_df = pd.DataFrame({
    'Actual': y_test,
    'Predicted': y_pred,
```

```
    'Residual': residuals
})

predictions_df.to_csv('xgboost_predictions.csv', index=False)

# 3. Salvar objetos do modelo
joblib.dump(final_model, 'xgboost_model.pkl')
joblib.dump(explainer, 'shap_explainer.pkl')

# 4. Salvar gráficos
figures = [
    'learning_curve.png',
    'feature_importance.png',
    'actual_vs_predicted.png',
    'residuals_distribution.png',
    'model_comparison.png',
    'shap_summary.png'
]

for fig_name in figures:
    plt.savefig(fig_name, dpi=300, bbox_inches='tight')
```

Listing D.8 – Exportação dos Resultados para Análise