UNIVERSIDADE FEDERAL DE ALAGOAS
CENTRO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

JOÃO VICTOR FERRO

Machine Learning **na predição do acúmulo de carboidratos pela microalga** ***Chlorella vulgaris*** **em cultivo contínuo**

Maceió
2025

João Victor Ferro

Machine Learning na predição do acúmulo de carboidratos pela microalga *Chlorella vulgaris* em cultivo contínuo

Dissertação de mestrado apresentado ao Programa de Pós-graduação em Engenharia Química da Universidade Federal de Alagoas como requisito parcial para obtenção do título de Mestre em Engenharia Química.

Orientador: Prof. Dr. Carlos Eduardo de Farias Silva.

Coorientadora: Profa. Dra. Brígida Maria Villar da Gama.

Maceió

2025

Catalogação na Fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico

Bibliotecário: Marcelino de Carvalho Freitas Neto – CRB-4 – 1767

F395m Ferro, João Victor.

Machine Learning na predição de produção de carboidratos pela microalga *Chlorella vulgaris* em cultivo contínuo / João Victor Ferro. – 2025.

126 f. : il. color.

Orientador: Carlos Eduardo de Farias Silva.

Co-orientadora: Brígida Maria Villar da Gama.

Dissertação (Mestrado em Engenharia Química) – Universidade Federal de Alagoas. Centro de Tecnologia. Maceió, 2025.

Bibliografia: f. 79-93.

Apêndices: f. 94-126.

1. Aprendizado do computador. 2. *Chlorella vulgaris* - Cultivo contínuo. 3. Produção de carboidratos. 4. Regressão (Modelos lineares). I. Título.

CDU: 661.665

João Victor Ferro

Machine Learning na predição do acúmulo de carboidratos pela microalga *Chlorella vulgaris* em cultivo contínuo

Dissertação apresentada à Universidade Federal de Alagoas, como requisito para a obtenção do Título de Mestre em Engenharia Química.

Aprovada em: Maceió, 21 de fevereiro de 2025.

BANCA EXAMINADORA

gov.br Documento assinado digitalmente
CARLOS EDUARDO DE FARIAS SILVA
Data: 24/02/2025 21:36:45-0300
Verifique em https://validar.iti.gov.br

_____
Prof. Dr. Carlos Eduardo de Farias Silva (Orientador - PPGEQ/UFAL)

gov.br Documento assinado digitalmente
BRIGIDA MARIA VILLAR DA GAMA
Data: 25/02/2025 10:06:09-0300
Verifique em https://validar.iti.gov.br

_____
Profª. Drª. Brígida Maria Villar da Gama (Coorientadora – Pós-Doc/PPGEQ/UFAL)

gov.br Documento assinado digitalmente
WAGNER ROBERTO DE OLIVEIRA PIMENTEL
Data: 24/02/2025 22:05:16-0300
Verifique em https://validar.iti.gov.br

_____
Prof. Dr. Wagner Roberto De Oliveira Pimentel (PPGEQ/UFAL)

gov.br Documento assinado digitalmente
DAYANA DE GUSMAO COELHO
Data: 25/02/2025 14:34:30-0300
Verifique em https://validar.iti.gov.br

_____
Profª. Drª. Dayana De Gusmao Coelho (CTEC/UFAL)

AGRADECIMENTOS

A Deus, pela minha vida, e por me permitir ultrapassar todos os obstáculos encontrados ao longo da realização deste trabalho. A minha mãe, Margarida Ferro e meu irmão Lucas Ferro Brito, que me incentivaram nos momentos difíceis e compreenderam a minha ausência enquanto eu me dedicava à realização deste trabalho. A minha avó, Luiza Omena (*in memorian*) pelos ensinamentos. A todas minhas tias pelo carinho. A meu tio, Luiz, pela presença.

Aos amigos, Hugo Oliveira, Valdeir Araújo, Gamaliel Tavares que sempre estiveram ao meu lado, pela amizade e pelo apoio demonstrado ao longo de todo o período de tempo em que me dediquei na caminhada acadêmica. Ao professor Carlos Eduardo, por ter sido meu orientador e a professora Brígida Villar por ter sido minha coorientadora e terem desempenhado tais funções com dedicação. As professoras Aparecida da Silva, Carmem Zanta, Emanuella Gomes, Juliana Lima, Luciana Vieira, Manuela Fagundes (*in memorian*), e Valéria Malta; assim como aos professores Denys Carllos, Frede Carvalho, Jorge Brito, Magno Querino, Wagner Pimentel, pelas correções e ensinamentos que me permitiram apresentar um melhor desempenho no meu processo de formação profissional ao longo do curso. Aos meus colegas do Laboratório de Bioprocessos pelo suporte.

À Universidade Federal de Alagoas, essencial no meu processo de formação profissional, pela dedicação, e por tudo o que aprendi ao longo dos anos do curso. A todos que participaram, direta ou indiretamente do desenvolvimento deste trabalho de pesquisa, enriquecendo o meu processo de aprendizado. Às pessoas que convivi ao longo desses anos de curso, que me incentivaram e que certamente tiveram impacto na minha formação acadêmica.

"If I have seen further, it is by standing on the shoulders of giants."

— Isaac Newton

RESUMO

Neste estudo, foi aplicada a técnica de *Machine Learning* (ML) para prever a produção de carboidratos pela microalga *Chlorella vulgaris* em um cultivo contínuo, utilizando variáveis nutricionais (concentração de nitrogênio, 500-75 mg L-1, e fósforo, 200-50 mg L-1, em meio), ambientais (intensidade luminosa entre 150-450 µmol m-2 s-1 e a densidade óptica entre 0,5-20) e operacionais (tempo de residência entre 2,9-4,4 dias e tempo normalizado entre 0-1) como entradas para os modelos preditivos dana produtividade de biomassa e carboidratos e a % de acúmulo de carboidratos em biomassa. A análise do coeficiente de Pearson permitiu identificar correlações significativas entre as variáveis independentes e dependentes, destacando a influência da concentração de nutrientes e da intensidade luminosa na produção de carboidratos. Foi observada alta correlação entre nitrogênio e fósforo, o que pode comprometer a qualidade das regressões devido à colinearidade; como alternativa, a razão N/P foi utilizada para contornar esse problema. O histograma das variáveis evidenciou padrões e tendências nos dados, impactando as predições. Diferentes técnicas de regressão foram implementadas e comparadas, incluindo modelos lineares (Regressão Linear Multivariada, Ridge e LASSO) e não lineares (*Random Forest*, Redes Neurais e *Support Vector Regression* – SVR). Os modelos não lineares apresentaram melhor desempenho na previsão de todas as variáveis de saída, especialmente *Random Forest* e Redes Neurais, que capturaram relações complexas entre as variáveis. O melhor desempenho foi obtido com *Random Forest*, alcançando R^2 de 0,9347 e RMSE de 0,2556 para a produtividade de carboidratos, e R^2 de 0,8962 e RMSE de 0,3222 para a produtividade de biomassa. A otimização dos modelos foi realizada por meio da busca em grade (*grid search*), permitindo a definição dos melhores hiperparâmetros e melhorando significativamente a acurácia das previsões. Além disso, a validação cruzada foi empregada para evitar *overfitting* e garantir a generalização dos resultados, tornando os modelos mais robustos e confiáveis. Além da análise estatística dos modelos, foi realizada uma avaliação gráfica dos resíduos para verificar a adequação das predições. Observou-se que, apesar do bom desempenho numérico dos modelos não lineares, os resíduos indicaram um leve viés centralizado, sugerindo que melhorias podem ser alcançadas com a inclusão de novas variáveis de entrada ou ajustes nos pré-processamentos dos dados.

Palavras-chave: Redes neurais artificiais, *Chlorella vulgaris*, microalga, produção de carboidratos, amido, regressão, *Support vector regression.*

ABSTRACT

In this study, Machine Learning (ML) technique was applied to predict carbohydrate production by the microalgae Chlorella vulgaris in a continuous cultivation, using nutritional (nitrogen concentration, 500-75 mg L-1, and phosphorus, 200-50 mg L-1, in medium), environmental (light intensity between 150-450 µmol m-2 s-1 and optical density between 0.5-20) and operational (residence time between 2.9-4.4 days and normalized time between 0-1) variables as inputs for the predictive models on biomass and carbohydrate productivity and the % of carbohydrate accumulation in biomass. The analysis of Pearson's coefficient allowed to identify significant correlations between the independent and dependent variables, highlighting the influence of nutrient concentration and light on carbohydrate production. A high correlation was observed between nitrogen and phosphorus, which can compromise the quality of the regressions due to collinearity; as an alternative, the N/P ratio was used to circumvent this problem. The histogram of the variables showed patterns and trends in the data, impacting the predictions. Different regression techniques were implemented and compared, including linear models (Multivariate Linear Regression, Ridge and LASSO) and non-linear models (Random Forest, Artificial Neural Networks and Support Vector Regression – SVR). The non-linear models showed better performance in predicting all output variables, especially Random Forest and Neural Networks, which captured complex relationships between the variables. The best performance was obtained with Random Forest, reaching R^2 of 0.9347 and RMSE of 0.2556 for carbohydrate productivity, and R^2 of 0.8962 and RMSE of 0.3222 for biomass productivity. The optimization of the models was performed through grid search, allowing the definition of the best hyperparameters and significantly improving the accuracy of the predictions. Furthermore, cross-validation was used to avoid overfitting and ensure generalizability of the results, making the models more robust and reliable. In addition to the statistical analysis of the models, a graphical evaluation of the residuals was performed to verify the adequacy of the predictions. It was observed that, despite the good numerical performance of the nonlinear models, the residuals indicated a slight centralized bias, suggesting that improvements can be achieved with the inclusion of new input variables or adjustments in the data preprocessing.

Keywords: artificial neural networks, microalgae, starch production, regression, support vector regression.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ANN	*Artificial Neural Network*
RNA	Redes Neurais Artificiais
SVM	*Support Vector Machine*
SVR	*Support Vector Regression*
RF	*Random Forest*
SK-learn	*SciKit Learn*
RMSE	*Root Mean-Squared Error*
MLP	*Multi-Layer perceptron*
TN	Tempo Normalizado
TR	Tempo de Residência
LASSO	*Least Absolute Shrinkage and Selection Operator*
PAR	*Photosynthetically Active Radiation*
CNN	*Convolutional Neural Networks*
LM	*Levenberg-Marquardt*
GBR	*Gradient Boosting Regression*
RMSP	*Root Mean Square Propagation*
SGDM	*Stochastic Gradient Descent with Momentum*
OD	*Optical Density*
SRQ	Soma residual dos Quadrados
RBF	*Radial Basis Function*
EQM	Erro Quadrático Médio

SUMÁRIO

1. INTRODUÇÃO

A inteligência artificial tem revolucionado diversas áreas científicas e industriais, sendo o *Machine Learning* um de seus principais pilares. Essa abordagem se destaca pela capacidade de identificar padrões complexos e modelar sistemas não lineares, especialmente em contextos em que os métodos tradicionais de regressão, como por exemplo, a regressão linear, polinomial e linear múltipla apresentam limitações na predição das variáveis de interesse (Goodfellow et al., 2016).

No campo dos bioprocessos, a aplicação de *Machine Learning* tem mostrado potencial significativo na predição de carboidratos produzidos (David, et al., 2023; Ramandani, et al. 2025; Pääkkönen, et al. 2024; Sheik, et al. 2024.). Particularmente no cultivo de microalgas, como a *Chlorella vulgaris*, que é amplamente reconhecida por seu alto teor de carboidratos, proteínas e outros compostos bioativos (Safi et al., 2014; Pääkkönen, et al. 2024; Ahmad Sobri, et al. 2023).

As microalgas desempenham um papel crucial na biotecnologia devido à sua versatilidade em aplicações como produção de bioenergia, alimentos funcionais e tratamentos ambientais (Su et al., 2023). A *Chlorella vulgaris*, em particular, destaca-se por sua capacidade de produzir elevadas concentrações de biomassa e de acumular alto teor de carboidratos (até 60%) em condições específicas de cultivo, que pode otimizar sua produtividade em modo contínuo (Abdel-Latif et al., 2022). No entanto, a otimização do cultivo dessa microalga exige uma compreensão detalhada da interação entre variáveis nutricionais, ambientais e operacionais, uma vez que o acúmulo de carboidrato intracelular depende da interação dessas variáveis. Além disso, esses carboidratos podem ser utilizados para os mais diversos fins, como a produção de etanol (De Farias Silva e Sforza, 2016).

Ferramentas de aprendizado supervisionado, como regressão linear, Ridge, Lasso, Redes Neurais, *Suport Vector Regression* e Random Forest, são amplamente utilizadas para modelar sistemas complexos (Pedregosa et al., 2011). Essas metodologias permitem prever com precisão a produtividade e a composição bioquímica de culturas das microalgas, fornecendo insights valiosos para otimização de processos. Além disso, a normalização de variáveis e a aplicação de métricas como o coeficiente de Pearson garantem a robustez dos modelos preditivos (Fisher, *et al.* 2022)

Dado o crescente interesse em tecnologias sustentáveis, este trabalho visa aplicar técnicas avançadas de *Machine Learning* para prever a produção de carboidratos pela *Chlorella vulgaris* em cultivo contínuo. A abordagem proposta incluiu uma análise comparativa de métodos de regressão e aprendizado supervisionado, com o objetivo de identificar o modelo mais eficaz para a predição de variáveis-chave, como produtividade da biomassa e de carboidratos e o percentual de carboidrato em biomassa. Ao explorar a relação entre condições de cultivo e produção de biomassa, espera-se contribuir para o avanço do uso de microalgas em processos industriais.

2. OBJETIVOS

2.1. Geral

Prever a produção de carboidratos de Chlorella vulgaris em cultivo contínuo aplicando Machine learning a partir de variáveis nutricionais (concentração de fósforo e nitrogênio), ambientais (intensidade luminosa e, densidade óptica) e operacionais (tempo de residência e tempo normalizado).

2.2. Específicos

- Identificar relações entre variáveis independentes e dependentes usando métricas: o coeficiente de Pearson e histograma;
- Implementar e comparar diferentes métodos de regressão: Técnicas lineares (regressão linear multivariada, *Ridge* e LASSO). Técnicas não lineares (*Random Forest*, Redes Neurais e *Support Vector Regression* - SVR);
- Otimizar os modelos de *Machine Learning* ao definir os melhores hiperparâmetros para cada técnica;
- Realizar validação cruzada para evitar *overfitting* e garantir a generalização dos resultados;
- Validar os modelos preditivos, ao avaliar a precisão com métricas como R^2 e RMSE para cada variável de saída em dados de teste.

3. FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como escopo apresentar um levantamento bibliográfico que serviu como base para o presente trabalho, sendo descritas algumas definições referentes os modelos de regressão, método Pearson e outras análises estatísticas sobre a organização e distribuição dos dados, assim como as ferramentas disponibilizadas pelo *SK-Learn* em conjunto com o *Python*.

3.1. Python

Python é uma linguagem de programação de alto nível, conhecida por sua sintaxe simples e legível, o que a torna ideal tanto para iniciantes quanto para profissionais experientes. Seu vasto ecossistema de bibliotecas, permite o desenvolvimento de soluções eficientes para uma ampla gama de aplicações, incluindo análise de dados, aprendizado de máquina, automação e desenvolvimento web. Além disso, sua comunidade ativa e crescente contribui constantemente com novos recursos e ferramentas, tornando o *Python* uma das linguagens mais populares e versáteis do mundo da programação (Ranganathan, et al. 2019).

O *Scikit-learn* é uma biblioteca de aprendizado de máquina em *Python*, reconhecida por sua eficiência e facilidade de uso. Construída sobre *NumPy*, *SciPy* e *Matplotlib*, oferece algoritmos para regressão, classificação, *clustering* e redução de dimensionalidade, incluindo SVR, Random Forest e PCA. (Pedregosa et al., 2011). E é a biblioteca mais importante deste trabalho, pois, foi ela que todas as regressões foram construídas. Outras bibliotecas foram de grande importância, como Pandas e *Seaborn* que oferecem ferramentas de pré-processamento de dados e visualização. (Mckinney, 2025; Waskom, 2025).

Além de ferramentas para pré-processamento, validação cruzada e ajuste de hiperparâmetros, permite a automação do fluxo de trabalho por meio de *pipelines*. Um desafio comum no aprendizado de máquina é o *overfitting*, que prejudica a generalização dos modelos. Para evitá-lo, é essencial separar corretamente os dados de treinamento e teste (Ying, 2019).

O *Scikit-learn* é uma biblioteca amplamente adotada para aprendizado de máquina em *Python*, destacando-se pela sua facilidade de uso e eficiência. Construída sobre *NumPy, SciPy* e *Matplotlib*, ela fornece uma variedade de algoritmos para tarefas de regressão, classificação, *clustering* e redução de dimensionalidade.

Recursos como regressão linear, regressão Lasso, regressão Ridge, Redes Neurais, *SVR (Support Vector Regressor), RF (Random Forest),* e PCA (*Principal Component Analysis*) ajudam a resolver diferentes desafios de análise de dados. (Pedregosa et al., 2011)

3.2. Machine learning

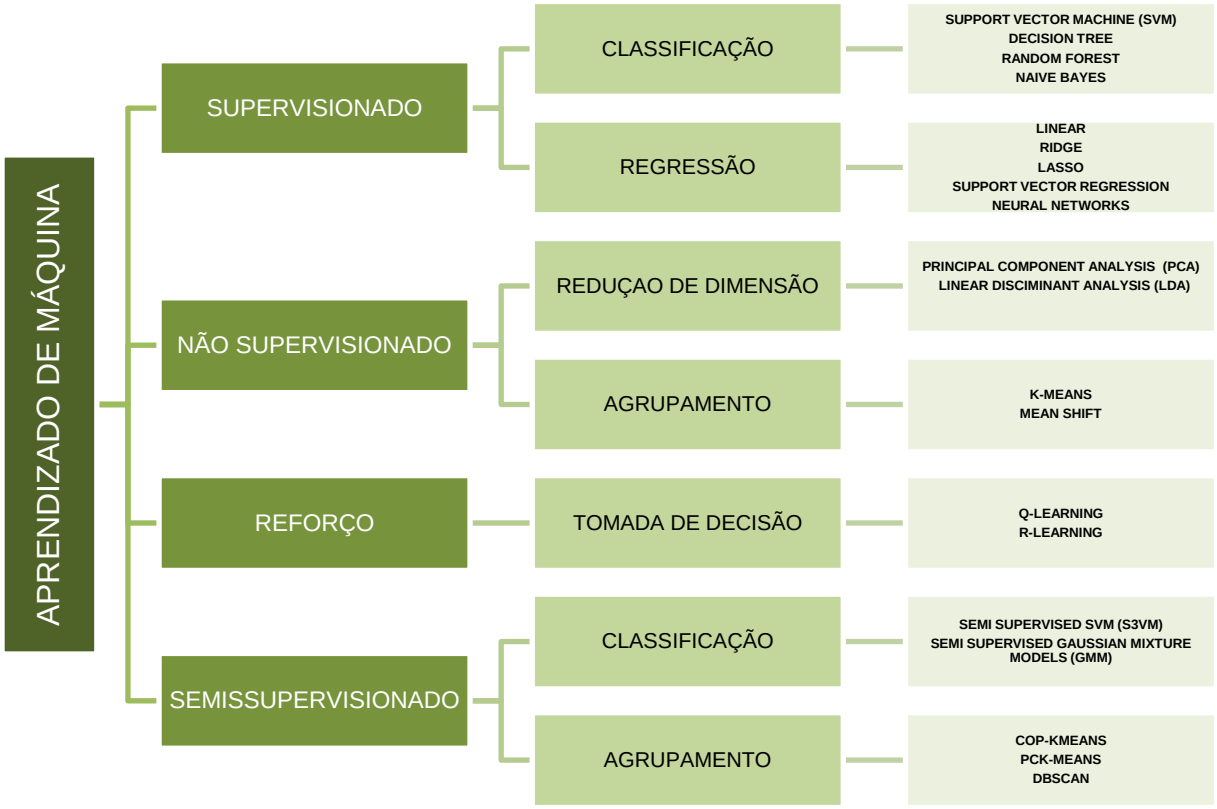
Machine Learning (ML), ou aprendizado de máquina, é uma subárea da inteligência artificial que desenvolve algoritmos capazes de aprender com dados e tomar decisões ou realizar previsões sem serem explicitamente programados. Esses algoritmos identificam padrões em conjuntos de dados e usam esse conhecimento para generalizar e aplicar o aprendizado em situações novas. A abordagem baseia-se em modelos matemáticos e estatísticos que, quanto mais expostos a dados, melhoram seu desempenho ao longo do tempo. Esse processo de aprendizado ocorre de forma iterativa, permitindo que os modelos se adaptem a contextos variados e complexos. (Sun et al. 2024; Smiti, 2020; Yüksel, et al. 2023)

Existem três categorias principais de aprendizado**:** supervisionado (*supervised*), não supervisionado (*unsupervised)*, por reforço (*reinforcement*) e o semi-supervisionado (*Semi-supervised*). No aprendizado supervisionado, o modelo aprende com dados rotulados, ou seja, entradas com as respostas corretas já conhecidas, como prever preços ou classificar imagens. No aprendizado não supervisionado, o objetivo é encontrar padrões ou estruturas ocultas em dados não rotulados, como segmentação de clientes. Já o aprendizado por reforço ocorre por tentativa e erro, onde o modelo é recompensado ou penalizado por suas ações. E o aprendizado semi-supervisionado é um tipo de aprendizado de máquina que utiliza uma combinação de dados rotulados (com respostas conhecidas) e dados não rotulados (sem respostas) para treinar um modelo. (Sarker, 2021a, 2021b; Smiti, 2020)

Dentro das três principais categorias de *Machine Learning*, existe uma ampla variedade de algoritmos sendo desenvolvidos continuamente, cada um com características específicas e aplicações voltadas para diferentes desafios computacionais. De modo geral, esses algoritmos podem ser agrupados em grandes categorias, cada uma delas englobando técnicas distintas que visam solucionar problemas específicos. No entanto o mesmo algoritmo pode ser adaptado, para

desempenhar funções distintas como é o caso do SVM, que é utilizado comumente como classificador e seu conceito adaptado para a regressão temos o SVR. A imagem da Figura *1* apresenta uma classificação resumida das principais abordagens de ML, destacando suas categorias, técnicas e algoritmos aplicados em diferentes contextos.

Figura 1 - Tipos de algoritmos que envolvem *Machine Learning*.



Fonte: (Autor, 2025; El-Gawad et. al., 2021 – adaptado)

O Aprendizado Supervisionado é caracterizado pela utilização de dados rotulados para treinar modelos preditivos, sendo dividido em classificação e regressão. Na classificação, os algoritmos mais utilizados incluem *Support Vector Machines* (SVM), *Decision Tree*, *Random Forest* (RF) e *Naive Bayes*, que são amplamente empregados para problemas que envolvem a predição de categorias discretas, como diagnóstico de doenças ou reconhecimento de padrões. Já na regressão, os principais métodos incluem *Linear Regression*, *Ridge Regression*, LASSO, *Support Vector Regression* (SVR) e Redes Neurais, voltados para a predição de valores contínuos, como previsão de preços ou tendências de mercado. (El-Gawad, 2021; Shah, 2021; Sarker, 2021b); da mesma maneira que este trabalho que tem por objetivo prever continuamente a produção de carboidratos.

No Aprendizado Não Supervisionado, os dados utilizados não possuem rótulos, e o objetivo é identificar padrões ou estruturas ocultas nos dados. Essa abordagem é dividida em redução de dimensionalidade e *clustering*. Na redução de dimensionalidade, técnicas como *Principal Component Analysis* (PCA) e *Linear Discriminant Analysis* (LDA) são utilizadas para simplificar conjuntos de dados complexos, preservando as informações mais relevantes, o que facilita a análise e visualização. Já no *clustering*, ou seja, grupos de dados formados com base em características ou similaridades compartilhadas. Entre eles algoritmos como *K-Means* e *Mean Shift* são aplicados para agrupar os dados em clusters com base em similaridades, sendo amplamente usados em segmentação de clientes e agrupamento de imagens. (El-Gawad et. al., 2021; Vaish et. al., 2021; Zhou, 2022)

O Aprendizado por Reforço é uma abordagem baseada em tomada de decisão em que um agente aprende interagindo com um ambiente dinâmico, recebendo recompensas ou penalidades conforme suas ações. Os algoritmos mais comuns nessa categoria são *Q-Learning* e *R-Learning*, que são utilizados em aplicações como controle de robôs, jogos, otimização de processos e navegação autônoma. (Vaish et. al., 2021; Nayeri et. al., 2021; Zhang et al., 2023; Schwartz, 1993; Watkins et al., 1992.)

Por fim, o Aprendizado Semi-Supervisionado combina dados rotulados e não rotulados, sendo uma solução intermediária que aproveita o potencial dos dados não rotulados para melhorar o desempenho dos modelos. Essa abordagem é subdividida em classificação e *clustering*. Na classificação, destacam-se os algoritmos *Semi-Supervised* SVM (S3VM) e *Gaussian Mixture Models* (GMM), que utilizam técnicas avançadas para explorar a estrutura dos dados não rotulados e aprimorar a predição em cenários com poucos dados rotulados disponíveis. Na área de *clustering*, métodos como COP-K*Means*, PCK-*Means* e DBSCAN incorporam restrições e informações parciais para formar agrupamentos mais consistentes e precisos. (Piccialli et. al., 2024; Zhao et al., 2025; Rao et al. 2023; Ikotun, 2023; Quinones-Grueiro et al. 2019; Hajihosseinlou et al., 2024)

3.3. Vantagens e desvantagens do uso de *machine learning*

A Tabela 1 apresenta uma visão equilibrada das vantagens e desvantagens do uso de *machine learning* (ML), destacando tanto o potencial transformador dessa

tecnologia quanto os desafios que precisam ser superados. Entre as vantagens, destaca-se a capacidade do ML de lidar com tarefas complexas e grandes volumes de dados, oferecendo alta precisão e automação de processos repetitivos. Além disso, a adaptabilidade dos modelos permite que eles sejam continuamente aprimorados com novos dados, o que é crucial em cenários dinâmicos. A tomada de decisão baseada em dados também é um ponto forte, pois reduz vieses humanos e aumenta a eficiência do resultado (DAHIYA et al., 2022). Esses benefícios são respaldados por estudos como os de Goodfellow et al. (2016) e LeCun et al. (2015), que destacam o poder do ML e do *deep learning* em resolver problemas antes considerados intratáveis.

Tabela 1 - Desempenho e Características de Modelos de Regressão e Machine Learning.

Algoritmo	Característica	Vantagem	Desvantagem	Referência
Linear multivariada	30 amostras, 4 variáveis	Simplicidade e facilidade de uso na previsão de toxicidade,	Limitação em capturar a complexidade das interações, resultando em previsões imprecisas	PRICE, et al. 2023
Ridge	42 amostras, 2 variáveis	Precisão em predições com dados colineares; lida bem com pequenos conjuntos de dados	Sensível à escolha de parâmetros e exige dados consistentes	CHING, et al. 2022
Lasso	129 amostras, 12 variáveis	Seleção automática de variáveis, reduzindo redundâncias	O uso do LASSO não só melhorou a precisão da estimativa, mas também simplificou o modelo, tornando-o mais prático e eficiente para aplicações em tempo real	NGUYEN, L. et al. 2023
RNA	1 milhão de amostras, 5 variáveis	Capacidade de modelar relações não-lineares complexas	Requer maior número de dados para evitar overfitting	IGOU et al. 2023
SVR	149 amostras, 3 variáveis	Boa generalização em dados não-lineares	Sensível à escolha de parâmetros como kernel	CHEN, J. et al. 2022
Random Forest	25 amostras, 6 variáveis	Robusto contra overfitting e útil para dados complexos	Alta demanda computacional e menos interpretabilidade	SONACHALAM, et al. 2024

Fonte: Autor, 2025

Por outro lado, as desvantagens revelam desafios significativos que precisam ser abordados para garantir o uso ético e eficaz do ML. A dependência de dados de qualidade é um dos maiores obstáculos, pois modelos treinados com dados incompletos ou enviesados podem gerar resultados imprecisos ou até prejudiciais. Além disso, o custo computacional elevado e a falta de interpretabilidade de modelos

complexos, como redes neurais, limitam sua aplicação em cenários onde a transparência é essencial (Ahmad Sobri et al., 2023; Kenge, 2020). Outro ponto crítico é o risco de viés e discriminação, já que modelos podem perpetuar desigualdades presentes nos dados de treinamento, como discutido por Mehrabi et al. (2021). Esses desafios exigem atenção contínua da comunidade científica e de desenvolvedores para garantir que o ML seja utilizado de forma responsável e justa.

3.4. Aplicação de *Machine learning* no cultivo de microalgas

A aplicação de aprendizado de máquina na fase de crescimento de microalgas tem se mostrado promissora para melhorar o rendimento e facilitar sua colheita, fatores que influenciam diretamente o custo de produção. O crescimento e a morfologia das microalgas são afetados por diversos fatores, e, embora muitos estudos tenham sido realizados para reduzir esses custos, abordagens tradicionais podem ser trabalhosas ou pouco precisas, dificultando a oferta de soluções eficientes para a produção real (Ning et al., 2022). Nesse contexto, técnicas de aprendizado de máquina têm sido cada vez mais exploradas para prever o crescimento e o rendimento final das microalgas.

Os estudos analisam diferentes abordagens para modelagem do crescimento e produção de biomassa em microalgas. He et al. (2016), Wang et al. (2019) e Figueroa-Torres et al. (2017) desenvolveram modelos cinéticos para estimar a formação de biomassa e lipídios sob diferentes condições nutricionais, enquanto Kaplan et al. (2020) avaliaram os efeitos do NaCl e fontes de carbono no crescimento de *Chlorella vulgaris*. Murwanashyaka et al. (2020) e Gojkovic et al. (2020) empregaram modelos baseados em Monod e Droop para analisar a geração de biomassa e armazenamento de moléculas em culturas heterotróficas, e Packer et al. (2011) propuseram um modelo matemático para síntese de lipídios neutros.

O aprendizado de máquina tem sido amplamente explorado. Supriyanto et al. (2019), Rodríguez-Rangel et al. (2022) e Hossain et al. (2022) aplicaram redes neurais artificiais (ANN) e algoritmos híbridos para prever a produtividade de microalgas. Lopez-Exposito et al. (2019), Yew et al. (2020) e Coşgun et al. (2021) utilizaram técnicas de visão computacional para prever propriedades fotossintéticas e biomassa. Liyanaarachchi et al. (2021) e Mohamed et al. (2013) otimizaram condições de cultivo com ANN e RSM.

Outros estudos abordaram modelagem estatística e aprendizado profundo. Ambat et al. (2019) aplicaram regressão linear para avaliar a produtividade de biomassa em águas residuais, enquanto Noguchi et al. (2019) e Susanna et al. (2019) usaram ANN para estimar o crescimento de culturas mistas e a produtividade de *Spirulina platenses*, respectivamente. Liu et al. (2020) desenvolveram uma rede neural para prever concentrações celulares a partir de espectros de fluorescência, e Garcia-Camacho et al. (2016) aplicaram redes neurais *feed-forward* para prever a concentração celular de *Karlodinium veneficum.*

A regressão linear multivariada foi utilizada por Price et al. (2023) para prever a toxicidade crônica do zinco para *Chlorella sp.,* enquanto Ching et al. (2022) aplicaram regressão Ridge para estimar o rendimento final da biomassa de *Spirulina platensis*. Nguyen et al. (2023) utilizaram o modelo LASSO para prever a densidade de *Chlorella vulgaris* a partir de imagens. Elmalky e Araji (2024) e Igou et al. (2023) aplicaram redes neurais e *Random forest* para monitoramento de produtividade, enquanto Chen et al. (2022) usaram aprendizado de máquina para prever o poder calorífico de biochars. Sonachalam et al. (2024) demonstraram alto desempenho na predição de emissões e eficiência de motores dual-fuel.

Além disso, Rogers et al. (2022) exploraram *transfer learning* para modelar bioprocessos, Del Rio-Chanona et al. (2019) revisaram modelos físicos e baseados em dados para simulação dinâmica de processos biológicos, e Bradford et al. (2018) usaram processos Gaussianos para otimizar a produção de algas sob incerteza.

Por fim, este estudo utilizou abordagens de aprendizado de máquina, como regressão multivariada, *Ridge*, LASSO, SVR e *Random Forest*, para modelar a produtividade de biomassa e carboidratos e concentração de *Chlorella vulgaris*, considerando tempo normalizado, intensidade luminosa e concentração de nutrientes. Um resumo das previsões é possível ser observado a seguir, na Tabela 2.

Tabela 2 - Resumo das previsões que envolvem microalgas usando abordagens baseadas em ML e não baseadas em ML.

Modelo de cultura	Modelo de Entrada	Tipos de Modelos	Saídas	Ref.
Isochrysis galbana	Biomassa, Lipídios, $NaNO_3$	Equações de Baranyi-Roberts e Luedeking-Piret	Produção de lipídios	He et al. (2016)
Dunaliella viridis	Biomassa funcional, carboidratos, lipídios, clorofila a, nitrogênio extracelular e intracelular	Modelo cinético	Produção de lipídios, carboidratos e biomassa	Wang et al. (2019)
Chlamydomonas reinhardtii	Nitrogênio, Acetato na biomassa, formação de amido e lipídios	Modelo cinético	Crescimento de biomassa, acúmulo de amido e lipídios	Figueroa-Torres et al. (2017)
Chlorella sorokiniana.	Velocidade de agitação, concentração de biomassa, floculante	Random Forest, CCA (*Cluster-Cluster Aggregation*); CLD (*Chord Length Distribution*); CLSM,	Concetração de biomassa	Lopez-Exposito; Negro; Blanco, (2019).
Chlorella kessleri	temperatura, ciclo luz-escuro (LD) e razão N/P	SVR, RSM (*Response Surface methodology*), *Generalized Linear model, Crow Search*, modelo de remoção	Eficiência de remoçao de N/P	Hossain et al. (2022).
Cultura mista	luz, temperatura, pH, oxigênio dissolvido (DO) e sólidos totais dissolvidos (TDS).	CNN (Densenet121), AVM	Produtividade de microalgas	Igou et al. (2023)
Chlorella sp.	Médias da cores (RGB), intervalo das médias de valores	LASSO, Processo Gaussiano, GS2	Densidade de microalgas	Nguyen et al. (2023)

	de pixels, frequências espaciais, entropias			
Spirulina platensis	pH, densidade óptica (OD)	Ridge, Linear multivariada	Rendimento de biomassa	Ching et al. (2022)
Chlorella sp.	pH, dureza ($CaCO_3$) e (demanda química de oxigênio DQO)	Linear multivariado	Toxicidade de Zinco	Price et al. (2023)
Chlorella vulgaris SAG 211–12	NaCl, glicose, glicerol	Modelo polinomial de baixa ordem	Crescimento, lipídios e amido	Kaplan et al. (2020)
Chlorella sorokiniana FACHB-275	Glicose, nitrogênio, fósforo	Modelo cinético baseado em Monod e Luedeking-Piret	Biomassa, carboidratos e lipídios	Murwanashyaka et al. (2020)
Coelastrella sp. 3-4, Scenedesmus sp. B2-2 e Scenedesmus obliquus RISE (UTEX 417)	Lipídios, biomassa, nitrogênio, carboidratos	Modelo cinético baseado no modelo matemático de Droop	Crescimento de biomassa	Gojkovic et al. (2020)
Pseudochlorococcum sp.	Concentração de biomassa excluindo lipídios neutros, Concentração de lipídios neutros, clorofila, concentração de nitrogênio extracelular	Modelo cinético baseado no modelo matemático de Droop	Crescimento de microalgas e lipídios neutros	Packer et al. (2011)
Cultura mista	Concentração inicial de microalgas (base seca), Período de colheita, Tempo de retenção hidráulica, Adição de acetato de sódio, Irradiância solar média, temperatura média da água, pH médio, Concentração de nitrato	Rede Neural Artificial (ANN)	Concentração de microalgas	Supriyanto et al. (2019)

Cultura mista	Licor misto, produção de biomassa, carboidratos, população de cianobactérias, população de diatomáceas, população de algas verdes, população de protozoários, TIC, TOC, TIN	ANNs, CNN, LSTMs, KNN, RF	Conteúdo de carboidratos	Rodríguez-Rangel et al. (2022)
C. vulgaris	Biomassa inicial, concentração de nitrogênio e valor de pH.	KNN	Imagens de microalgas para estimativa de propriedades fotossintéticas.	Yew et al. (2020)
Várias espécies de microalgas	Temperatura, intensidade luminosa, fotoperíodo, conteúdo de CO_2, PO_4 e N.	AR, DT	Produtividade de biomassa e conteúdo lipídico.	Coşgun et al. (2021)
C. vulgaris	pH e tempo de cultivo.	MLP, RSM	Biomassa, concentração de lipídios totais, lipídios insaturados e ácido oleico.	Liyanaarachchi et al. (2021)
Tetraselmis sp.	Concentração de glicose, extrato de levedura e nitrato.	MLP, RSM	Concentração de biomassa e rendimento lipídico.	Mohamed et al. (2013)
Várias espécies de microalgas	Concentração de nutrientes, produção de biomassa e produtividade lipídica.	LR	TN, TP, COD e produção de biomassa.	Ambat et al. (2019)

Desmodesmus sp., Scenedesmus sp., Dictyosphaerium sp., Klebsormidium sp., Microctinium sp.	Concentração inicial de microalgas, período de colheita, tempo de retenção hidráulica, adição de acetato de sódio, irradiância solar média, temperatura média da água, pH médio, concentração de nitrato.	ANN	Concentração de microalgas no período de cultivo.	Noguchi et al. (2019)
Spirulina platensis	Temperatura da cultura, intensidade luminosa, pH, oxigênio dissolvido, taxa de produção de oxigênio, tempo após colheita, bicarbonato, fosfato, nitrato e biomassa inicial.	MLP	Concentração de tricomas, tamanho dos tricomas e densidade óptica.	Susanna et al. (2019)
C. reinhardtii	Espectros de emissão de fluorescência.	ANN, GA	Concentração celular.	Liu et al. (2020)
Karlodinium veneficum	Concentração inicial de células e nutrientes, e duração da cultura.	FFBN	Concentração celular.	García-Camacho et al. (2016)
Ácido succínico e salicílico	Intensidade luminosa, taxa de influxo de nitrato, biomassa, nitrato e concentração de luteína, concentração de ácido salicílico.	*Transfer Learning*: Comparado com modelos cinéticos e ANN	Mudança na biomassa, nitrato e concentração de luteína.	Rogers et al. (2022)
C. vulgaris	Concentração inicial de biomassa, fosfato, glicose e	ANN	Mudança na biomassa, fosfato, glicose e concentração de nitrato.	Del Rio-Chanona et al. (2019)

	nitrato, coeficientes de rendimento.			
Desmodesmus sp. F51	Biomassa, concentração de nitrato e luteína, taxa de influxo, nitrato influente, intensidade luminosa.	LR, SGD	Biomassa, concentração de nitrato e produção de luteína.	Bradford et al. (2018)
Chlorella vulgaris	Dados experimentais de pH, temperatura, COD de entrada e taxa de fluxo de ar	ANN, correlações empíricas	Eficiência de remoção de DQO e DQO residual	Jerry et al. (2023)
Chlorella vulgaris	Tempo normalizado, concentração de fósforo, concentração de nitrogênio, tempo de residência, intensidade luminosa, densidade óptica	Linear multivariada, Ridge, LASSO, SVR, RF	Porcentagem de carboidratos em biomassa, Produtividade de carboidratos, Produtividade de biomassa	Este estudo

3.5. Chlorella vulgaris

Chlorella vulgaris é uma microalga verde unicelular amplamente estudada devido à sua versatilidade metabólica e à rica composição nutricional, o que a torna uma matéria-prima promissora para diversas aplicações industriais. Pertencente ao filo *Chlorophyta*, essa microalga destaca-se por sua elevada taxa de crescimento, alta eficiência fotossintética e resistência a condições ambientais adversas. Sua composição é rica em proteínas (cerca de 61,6%), lipídios (12,5%) e carboidratos (13,7%), além de vitaminas, minerais, pigmentos, polissacarídeos, fatores de crescimento e outros compostos bioativos. Isso confere à *C. vulgaris* um grande potencial para a produção de biocombustíveis, alimentos, ração animal e nutracêuticos, além de suas aplicações em biotecnologia e bioeconomia sustentável (Coronado-Reyes et al., 2022; Safi et al., 2014).

O cultivo de *C. vulgaris* pode ser realizado de forma sustentável, utilizando fontes alternativas de nutrientes, como resíduos industriais e agroindustriais, o que não só reduz os custos de produção, mas também contribui para a mitigação dos impactos ambientais causados pelo descarte inadequado de resíduos. De fato, a capacidade de adaptação dessa microalga e seu alto teor de biomoléculas de interesse tornam-na uma excelente candidata para aplicações em larga escala, tanto na biotecnologia quanto na bioenergia (Peter, et al., 2022; Chu et al., 2022).

Em resposta a condições de estresse, como a deficiência de nitrogênio e excesso de carbono como provimento de ar enriquecido com CO_2 continuamente, *C. vulgaris* acumula amido, e se o estresse for prolongado, aumenta sua concentração de lipídios intracelular, principalmente triacilgliceróis – TAGs, como reservas energéticas. Esses compostos são armazenados nos plastídios, como os cloroplastos, e em gotículas lipídicas no citoplasma, que servem como fontes de energia quando as condições ambientais se tornam desfavoráveis (Chokshi et al., 2017; Sun, et al. 2018; de Farias Silva e Sforza, 2016).

O estresse salino (estresse osmótico) e mudanças de temperatura também podem induzir esse comportamento metabólico, promovendo a síntese de reservas energéticas. A combinação de salinidade e luz intensa aumenta a biomassa, lipídios e antioxidantes em *C. vulgaris*, otimizando a produção de biodiesel e compostos valiosos (El-Fayoumy et al., 2024; Mountourakis, et al. 2023).

Deficiências de fósforo ou enxofre interferem em processos essenciais, como a produção de ATP e proteínas, forçando a célula a acumular carboidratos e lipídios como mecanismos adaptativos. A limitação de fósforo tem um impacto significativo na composição bioquímica da microalga *Chlorella vulgaris*, aumentando o teor de lipídios e reduzindo a quantidade de proteínas (Javed et al., 2022; Fu, et al., 2019; Xing et al., 2021).

No entanto, provocar condições de estresse não quer dizer que necessariamente haverá aumento na produtividade de forma significativa como Marino (2018) demonstrou ao inibir a luz, é importante verificar a combinação de todos os fatores, ambientais. Nutricionais e operacionais. Ainda assim, estratégias como o cultivo mixotrófico, que combina fotossíntese e consumo de carbono orgânico, podem melhorar o crescimento celular e a produção de biomassa e lipídios. Ao utilizar a vinhaça e biochar (Ferreira, et al. 2021), glicose (Yun, et al., 2021), águas residuais (Chu et al. 2022), dejetos de frango (Tan et al., 2021), junto do fornecimento de luz, houve a produção de biomassa com teores de carboidratos e lipídios aumentados.

Outro fator importante é que o comprimento de onda da luz também pode estimular o acúmulo de amido e lipídios, uma vez que o carbono fixado pela fotossíntese é redirecionado para o armazenamento desses compostos, quando as vias metabólicas de crescimento celular estão limitadas. Six et al. (2024a) revelou que a indução de amido pode ser realizada através de métodos abióticos, como a privação de nitrogênio ou exposição à luz vermelha, que promove uma produção estável e constante de amido, adequada para sistemas de cultivo contínuos.

É destacada a potencialidade da *C. vulgaris* para a produção de bioplásticos, oriunda da extração do amido do seu crescimento para produção de amido termoplástico, que pode ser processado em escala industrial com altos rendimentos e purezas (Arora et al., 2023; Six et al. 2024b). A produção de Microalgas tem sido utilizada amplamente para diversos fins, desde a produção de ração para alimentação, suplementação de nutrientes, produtos farmacêuticos. (Su et al. 2023). As microalgas possuem um grande potencial na aquicultura por seus compostos bioativos, como antioxidantes, ácidos graxos, proteínas e pigmentos, que melhoram a saúde dos organismos aquáticos, aumentando a resistência ao estresse e a imunidade (Abdel-Latif *et al*. 2022).

No campo da saúde humana, *C. vulgaris* pode ser eficaz na redução do colesterol total e LDL, com benefícios comprovados na prevenção de doenças

cardiovasculares. Uma meta-análise de ensaios clínicos randomizados revelou uma redução média de 7,47 mg/dL no colesterol total e 7,71 mg/dL no colesterol LDL, destacando seu potencial como suplemento alimentar para o controle lipídico (Sherafati, 2022). Sua composição rica em fibras, carotenoides e ácidos graxos insaturados também pode contribuir para a saúde cardiovascular, além de melhorar a absorção de lipídios no intestino (Barghchi et al., 2023; Bito et al., 2020)

Além de suas aplicações industriais, *C. vulgaris* também tem demonstrado grande eficiência na adsorção e bioacumulação de metais pesados, como cádmio, chumbo, níquel, cromo, cobre e zinco, com eficiências de remoção superiores a 90% em alguns casos. A microalga utiliza mecanismos de detoxificação, como a produção de fitochelatinas e metalotioneínas, para mitigar os efeitos tóxicos desses metais, tornando-a uma excelente candidata para o tratamento de águas contaminadas (Faruque et al., 2024; Yadav, et al., 2022). No entanto, a exposição de *C. vulgaris* a nanopartículas (NP) de ZnO e Fe_2O_3 reduz a taxa de crescimento, o conteúdo de clorofila e a integridade celular. As NP induzem estresse oxidativo, aumentando ROS, peroxidação lipídica e a atividade de enzimas antioxidantes. Microscopia revelou danos estruturais, como lise celular e desintegração da membrana. Em comparação com suas formas particuladas maiores, as NP mostraram maior toxicidade devido à maior área superficial e interação celular. (Saxena et al., 2021)

Quando associada a tratamento anaeróbico *C. vulgaris* também pode biodegradar poluentes, e a combinação do cultivo de microalgas com o tratamento anaeróbico melhora a qualidade do efluente, reduzindo a DQO de 16.000 mg/L para 1.000 mg/L (Sidabutar, et al., 2024; Zhang et al., 2024). A produção de energia através da biomassa de *C. vulgaris* por carbonização hidrotérmica catalítica com ácido acético, vem destacando seu potencial como fonte renovável para energia de baixo carbono. O hidrocarvão resultante apresentou maior densidade energética, estabilidade e desempenho de combustão em relação à biomassa bruta. Além disso, *C. vulgaris* mostrou alto potencial de fixação de CO_2, reduzindo emissões em até -1,54 kg CO_2,eq/kWh na combustão conjunta com carvão, reforçando sua viabilidade como recurso sustentável para energia limpa (Sztancs et al., 2021).

Por fim, a produção de etanol a partir de *C. vulgaris* tem se mostrado promissora devido ao seu alto teor de carboidratos. Em estudos recentes, foi possível produzir etanol a partir da biomassa da microalga cultivada em águas residuais, com uma produtividade que sugere viabilidade econômica para grandes escalas de

produção. A otimização de fatores como iluminação, aeração enriquecida com CO_2 e teor de nutrientes pode ainda melhorar a eficiência desse processo, consolidando a *C. vulgaris* como uma opção viável para a produção de biocombustíveis e outros produtos sustentáveis (Honório et al., 2024; de Farias Silva e Bertucco, 2019).

3.6. Cultivo contínuo com limitação de nutrientes

O estudo de Farias Silva e Sforza (2016) que é a base para este trabalho, investigou a produtividade de carboidratos em *Chlorella vulgaris* cultivada em um fotobiorreator contínuo de 300 mL, sob diferentes intensidades luminosas (150, 300 e 450 µmol fótons m^{-2} s^{-1}), variações na concentração de nitrogênio e fósforo no cultivo, além do tempo de residência. O sistema operou continuamente após atingir uma concentração celular significativa, com o tempo de residência controlado por uma bomba peristáltica. A biomassa foi analisada diariamente para medir a concentração celular, o conteúdo de carboidratos e o consumo de nutrientes.

Os resultados indicaram que a limitação de nitrogênio combinada com alta intensidade luminosa aumentou o acúmulo de carboidratos, atingindo 52% do peso seco em 450 µmol fótons $m^{-2}s^{-1}$, embora a produtividade de biomassa tenha diminuído. A fotossaturação reduziu a eficiência fotossintética em intensidades mais altas, e o tempo de residência maior favoreceu o acúmulo de carboidratos. O fósforo teve um comportamento influenciado pela luz, acumulando-se mais em intensidades elevadas (de Farias Silva e Sforza, 2016).

O estudo identificou condições ideais para maximizar a produção de carboidratos, equilibrando produtividade de biomassa e acúmulo de reservas energéticas, através da combinação das variáveis supracitadas. Os resultados demonstram a viabilidade do cultivo contínuo de *Chlorella vulgaris* em relação a batelada no aumento de sua produtividade e explicita que o motivo principal do trabalho para aplicações industriais visou a produção de bioetanol a partir dessa biomassa.

3.7. Regressão linear multivariada

A regressão linear multivariada é uma técnica estatística amplamente utilizada para modelar a relação entre uma variável dependente e múltiplas variáveis

independentes. Diferente da regressão linear simples, que lida com uma única variável preditora, a regressão multivariada permite que múltiplos fatores influenciem o comportamento da variável resposta. Essa abordagem é fundamental para entender relações complexas, onde muitos fatores podem contribuir simultaneamente para um determinado resultado. O objetivo principal é identificar como cada variável independente afeta a variável dependente, proporcionando uma previsão precisa e insights valiosos sobre as interações entre as variáveis. (Rencher; Christensen, 2012; Meyers, 2003)

No entanto, para que os resultados da regressão linear multivariada sejam válidos e úteis, algumas suposições precisam ser atendidas, como linearidade, homoscedasticidade e ausência de multicolinearidade entre as variáveis independentes. Quando essas condições são cumpridas, o modelo pode fornecer estimativas robustas e interpretações confiáveis. Apesar de ser uma ferramenta poderosa, a regressão linear multivariada também apresenta limitações, como a sensibilidade a *outliers* e a complexidade no tratamento de multicolinearidade, o que pode comprometer a precisão dos coeficientes estimados. Portanto, é fundamental realizar diagnósticos adequados após o ajuste do modelo para garantir que os resultados sejam interpretáveis e relevantes para o contexto de análise. (CHEIN, 2019; Rencher; Christensen, 2012)

A regressão linear considera a seguinte relação entre um número de variáveis independentes de entrada (preditores) $x_1, \ldots, x_{k-1}$e uma variável dependente na saída, y (resposta) (Melkumova et al., 2017). Assim, o y é dado por:

$$y = b_0 + b_1 x_1 + \ldots + b_{k-1} x_{k-1} + \varepsilon \quad (1)$$

As medições são realizadas n vezes para que se tenha n valores de y para n conjuntos de x_j

$$y_i = b_0 + b_1 x_{i1} + \ldots + b_{k-1} x_{ik-1} + \varepsilon_i \quad | \quad i = 1, n \quad (2)$$

Onde x_{ij} é a i-ésima observação de x_j. Os ε_i não são observados diretamente. As equações (2) podem ser expressas na forma matricial após adicionar os parâmetros:

$$x_{10} = x_{20} = \ldots = x_{n0} = 1 \quad (3)$$

Então:

$$\boldsymbol{Y} = \boldsymbol{XB} + \boldsymbol{\xi} \tag{4}$$

Onde:

$$\boldsymbol{Y} = [\boldsymbol{y_i}]_{\boldsymbol{n}}\,;\ \boldsymbol{X} = [\boldsymbol{x_{ij}}]_{\boldsymbol{n\times k}}\,;\ \ \boldsymbol{B} = [\boldsymbol{b_j}]_{\boldsymbol{k}}\,;\ \ \boldsymbol{\xi} = [\boldsymbol{\varepsilon_i}]_{\boldsymbol{n}} \tag{5}$$

As coordenadas $b_0 + b_1 + \ldots + b_{k-1}$ do vetor B são desconhecidas. O objetivo da análise de regressão é estimar o vetor B com base nas observações multivariadas.

$$[\boldsymbol{X}, \boldsymbol{Y}] = \begin{bmatrix} x_{10} & x_{11} & \ldots & x_{1k-1} & y_1 \\ x_{20} & x_{21} & \ldots & x_{2k-1} & y_2 \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ x_{n0} & x_{n1} & \ldots & x_{nk-1} & y_n \end{bmatrix} \tag{6}$$

Uma abordagem tradicional para este problema é usar o estimador de mínimos quadrados (MQ) onde:

$$\sum_{i=1}^{n} \left(y_i - \sum_{i=1}^{k-1} \left(b_i x_{ij} \right) \right)^2 \rightarrow min \tag{7}$$

As estimativas de MQ dos coeficientes desconhecidos: $b_0 + b_1 + \ldots + b_{k-1}$ minimizam (7):

$$\widehat{\boldsymbol{B}} = [\widehat{\boldsymbol{b_j}}]_{\boldsymbol{k}} \tag{8}$$

Como: det $\boldsymbol{X^t X} > \boldsymbol{0}$. O MQ pode ser calculado utilizando a seguinte equação:

$$\widehat{\boldsymbol{B}} = (\boldsymbol{X^t X})^{-1}\, \boldsymbol{X^t Y} \tag{9}$$

Sendo $\widehat{\boldsymbol{Y}} = \boldsymbol{X}\widehat{\boldsymbol{B}}$. Pode-se reescrever esta equação na forma coordenada da seguinte forma:

$$\widehat{y_i} = \widehat{b_0} + \widehat{b_1} x_{i1} + \ldots + \widehat{b}_{k-1} x_{ik-1} \quad | \quad i = 1, n \tag{10}$$

Onde $\widehat{y_i}$ é o valor de resposta previsto que corresponde aos valores preditores $x_1, \ldots, x_{k-1}$. A soma residual dos quadrados (SRQ) mede a discrepância entre os dados e o modelo de estimativa, assim:

$$SRQ = \sum_{i=1}^{n} (\hat{y}_i - y_i)^2 \tag{11}$$

O coeficiente de determinação R²:

$$R^2 = 1 - \frac{\sum_{i=1}^{n} (\hat{y}_i - y_i)^2}{\sum_{i=1}^{n} (y_i - \bar{y})^2} \tag{12}$$

O valor de R² (12) também mede a qualidade do modelo de regressão: quanto mais próximo de 1, melhor o modelo de regressão (10) se ajusta aos dados (6). A padronização dos dados pela normalização é frequentemente usada na análise de regressão linear (Melkumova et al., 2017). Ou seja, denotando:

$$\bar{x}_j = \frac{1}{n}\sum_{i=1}^{n} x_{ij}\,;\ \ \bar{y} = \frac{1}{n}\sum_{i=1}^{n} y_i\,;\ S_y^2 = \frac{1}{n}\sum_{i=1}^{n} (y_i - \bar{y})^2\,;\ \ S_j^2 = \sum_{i=1}^{n} \left(x_{ij} - \bar{x}_j\right)^2\,;\ j = \overline{1, k-1} \tag{13}$$

Obtemos variáveis centralizadas e normalizadas para a amostra inicial (6):

$$v_i = \frac{y_i - \bar{y}}{S_y};\ \ w_{ij} = \frac{x_{ij} - \bar{x}_j}{S_j};\ i = \overline{1, n};\ \ j = \overline{1, k-1} \tag{14}$$

Assim podemos denotar que:

$$\boldsymbol{V} = [\boldsymbol{v_i}]_{\boldsymbol{n}}\,;\ \ \boldsymbol{W} = [w_{ij}]_{n \times k-1} \tag{15}$$

Caso det $\boldsymbol{W^t W} > \mathbf{0}.$, as estimativas de MQ para o modelo padronizado podem ser calculadas usando a fórmula:

$$\widehat{\boldsymbol{B}} = (\boldsymbol{W^t W})^{-1}\, \boldsymbol{W^t V} \tag{16}$$

Existem várias vantagens em usar dados padronizados para regressão linear. Primeiro, com dados padronizados a solução não depende da escala de medição. Os preditores x_j podem ser medidos em escalas diferentes, enquanto os preditores padronizados w_j são reduzidos à mesma escala “neutra”. Segundo, a entrada do

preditor tem mais probabilidade de depender do valor relativo w_j do que do valor absoluto x_j. (Melkumova et al., 2017)

3.8. Regressão LASSO

A regressão LASSO (*Least Absolute Shrinkage and Selection Operator*) foi introduzida por Tibshirani (1996) como uma abordagem inovadora que combina regularização e seleção de variáveis em modelos de regressão linear, destacando suas vantagens em cenários de alta dimensionalidade e multicolinearidade. O método rapidamente ganhou popularidade por sua eficácia e simplicidade, tornando-se uma ferramenta essencial em estatística e aprendizado de máquina (Varshini; Kumari, 2020).

A regressão LASSO. É um operador de mínima redução absoluta e seleção. Se destacando os dois aspectos principais do método: a redução (ou "encolhimento") dos coeficientes e a seleção automática de variáveis. É uma técnica amplamente utilizada na modelagem estatística e aprendizado de máquina para realizar seleção de variáveis e regularização de modelos lineares. Diferente da regressão linear comum, que minimiza o erro quadrático, o LASSO adiciona um termo de penalidade baseado na soma dos valores absolutos dos coeficientes. Essa penalidade força alguns coeficientes a se tornarem exatamente zero, permitindo a exclusão automática de variáveis irrelevantes ou redundantes. Como resultado, o LASSO é especialmente útil em cenários onde há muitas variáveis preditoras e o objetivo é identificar um subconjunto de variáveis que mais contribuem para a previsão do modelo. (Ranstam; Cook, 2018).

Uma das principais vantagens do LASSO é sua capacidade de evitar o overfitting, proporcionando modelos mais simples e interpretáveis. No entanto, essa técnica também apresenta limitações, como a tendência de selecionar apenas uma variável entre aquelas altamente correlacionadas, o que pode levar à perda de informações importantes. Além disso, o desempenho do LASSO depende do ajuste cuidadoso de seu parâmetro de regularização, geralmente determinado por validação cruzada. (Saini, et al*.,* 2023)

A estimativa Lasso B_λ é a solução dos seguintes problemas de minimização equivalentes para observações padronizadas {***W, V***}.

$$\|\boldsymbol{V} - \boldsymbol{W}B\|^2 + \lambda \|B\|_1 \rightarrow min \quad (23)$$

Para todo $\lambda > 0$, existe um $t(\lambda) > 0$ tal que:

$$\|\boldsymbol{V} - \boldsymbol{W}B\|^2 \rightarrow min \quad (24)$$

Tal que:

$$\lambda \|B\|_1 \leq t(\lambda) \quad (25)$$

Onde:

$$\|B\|_1 \leq \sum_{j=1}^{k=1} |\beta_j| \quad (26)$$

A penalidade no vetor de coeficientes β_j, j = $\overline{1, \mathrm{k} - 1}$ imposta pelo LASSO é ligeiramente diferente de Ridge. No caso do LASSO, o parâmetro λ é multiplicado pela norma ℓ_1 do vetor (β_1, ..., β_{k-1}) enquanto Ridge usa a norma ℓ_2.

Um dos efeitos positivos desta mudança em termos de interpretação do modelo de Lasso, ao contrário da regressão de *Ridge* resulta em um modelo onde algumas estimativas de coeficiente são exatamente iguais a zero quando λ é grande. Em outras palavras, a regularização LASSO realiza adicionalmente a seleção de variáveis, o que facilita a interpretação do modelo. Como no caso de *Ridge*, diferentes valores de λ produzem diferentes vetores β_λ. É por isso que é importante selecionar um valor adequado de λ (Melkumova, et al. 2017).

3.9. Regressão *Ridge*

A Regressão *Ridge* é uma técnica amplamente utilizada em problemas de regressão para lidar com a multicolinearidade, que ocorre quando as variáveis independentes estão altamente correlacionadas entre si. Proposta inicialmente por Arthur Hoerl e Robert Kennard em 1970, essa abordagem é uma extensão da regressão linear tradicional, com a adição de um mecanismo de regularização que reduz a complexidade do modelo (Hoerl, 2020; Gruber, 1998; Hoerl, 1970).

A regressão *Ridge* é uma variante da regressão linear que introduz uma penalização L_2 aos coeficientes do modelo para reduzir a complexidade e evitar o *overfitting*. Ela funciona adicionando um termo de regularização à função de custo,

que é proporcional ao quadrado da magnitude dos coeficientes. O objetivo é minimizar o erro quadrático médio, enquanto restringe o tamanho dos coeficientes. A regularização ajuda a lidar com problemas de multicolinearidade, ou seja, quando as variáveis independentes são altamente correlacionadas e melhora a generalização do modelo (Theodoridis; Koutroumbas, 2009).

Um aspecto importante da regressão *Ridge* é que, ao contrário da regressão LASSO, ela não força os coeficientes a zero, mas os reduz de forma suave, mantendo todos os preditores no modelo. Isso a torna útil quando se deseja manter todas as variáveis explicativas, mas sem permitir que algumas delas dominem as demais. O parâmetro de regularização, λ, controla o equilíbrio entre ajuste aos dados e complexidade do modelo: quanto maior o valor de λ, mais forte a penalização e, consequentemente, menores os coeficientes. (Shah, et al., 2021)

A estimativa *Ridge* de um vetor desconhecido B, para observações padronizadas {$\boldsymbol{W, V}$} é dada por:

$$\widehat{\boldsymbol{B}}_{\lambda} = (\boldsymbol{W^t W} + \lambda I_n)^{-1} \boldsymbol{W^t V} \tag{17}$$

Onde I_n é a matriz identidade e $\lambda > 0$ é chamado de parâmetro de regularização. Denotaremos a forma coordenada do estimador de Ridge por:

$$\widehat{\boldsymbol{B}}_{\lambda} = [\widehat{\boldsymbol{\beta}}_j(\lambda)]_{k-1} \tag{18}$$

Ao adicionar parâmetro λ (*Ridge*) aos elementos diagonais da matriz $\boldsymbol{W^t V}$, é possível transformar a matriz em $(\boldsymbol{W^t W} + \lambda I_n)$ e não interferir significativamente no formato da matriz. Desta forma evitamos problemas habituais com a inversão de matriz mal condicionada. Contudo, vale a pena notar que, ao contrário do MQ, a estimativa de *Ridge*: $\widehat{\boldsymbol{B}}_{\lambda}$ é tendenciosa (Melkumova, et al. 2017). Pode ser mostrado que a estimativa de *Ridge:* $\widehat{\boldsymbol{B}}_{\lambda}$ é a solução dos seguintes problemas de minimização equivalentes:

$$\|\boldsymbol{V} - \boldsymbol{W}B\|^2 + \lambda\|B\|^2 \rightarrow min \tag{19}$$

Para todo $\lambda > 0$, existe um $t(\lambda) > 0$ tal que:

$$\|\boldsymbol{V} - \boldsymbol{W}B\| \rightarrow min \ \|\boldsymbol{V} - \boldsymbol{W}B\|^2 + \lambda\|B\|^2 \rightarrow min \tag{20}$$

Tal que:

$$\lambda \, \|B\|_2 \, \leq \, t(\lambda) \tag{21}$$

Onde:

$$\|B\|_1 \, \leq \, \sum_{j=1}^{k=1} |\, \beta_j| \tag{22}$$

Portanto, a estimativa de *Ridge* pode ser vista como uma estimativa SRQ com uma penalidade adicional imposta ao vetor de coeficientes.

3.10. Redes Neurais para Regressão

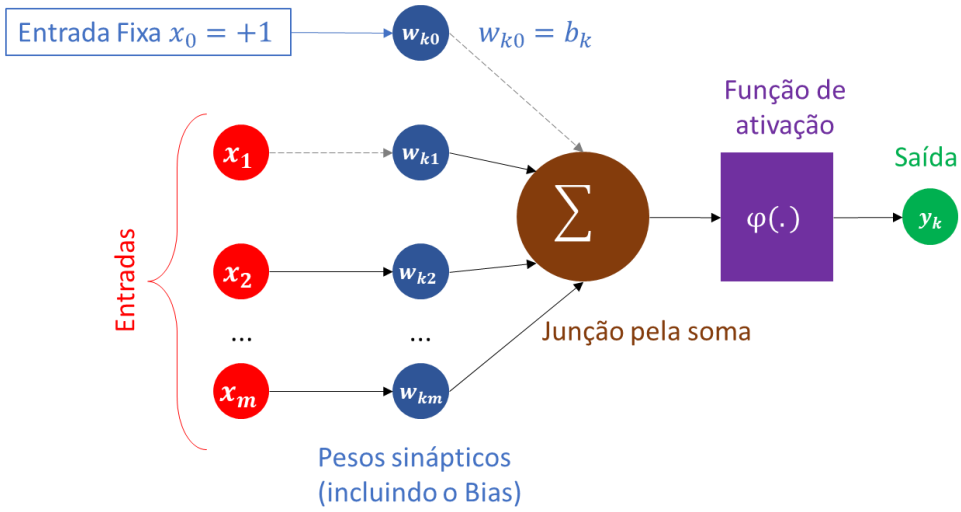
Uma rede neural é um sistema computacional que opera de maneira distribuída e massivamente paralela, composto por unidades de processamento simples. Ela possui a capacidade de aprender com a experiência, armazenando esse conhecimento e utilizando-o conforme necessário. Assim como o cérebro humano, a rede neural adquire informações por meio de interações com o ambiente. O aprendizado ocorre ajustando as conexões entre as unidades de processamento, chamadas de pesos sinápticos, onde o conhecimento é efetivamente armazenado (Géron, 2019).

Essas redes têm uma natureza de ‘‘caixa preta’’ e possuem a capacidade comum de construir modelos empíricos dos sistemas para os quais as dependências teóricas entre a entrada e a saída são extremamente complicadas ou mesmo desconhecidas. Esse processo permite à rede reconhecer padrões e realizar tarefas complexas de forma semelhante ao funcionamento dos neurônios biológicos. (Elyashberg; Williams; Martin, 2008; Haykin, 2001).

Tais Tarefas de aprendizado de máquina geralmente são descritas em termos de como o sistema deve processar um exemplo. Um exemplo é uma coleção de características que foram medidas quantitativamente a partir de algum objeto ou evento que queremos que o sistema de aprendizado de máquina processe. Normalmente, representamos um exemplo como um vetor $x \in R^n$, onde cada entrada x_i do vetor corresponde a uma característica. (Goodfellow et al., 2016).

O modelo neural ilustrado na *Figura 2*, possui um viés externo, representado por b_k, que altera a entrada líquida da função de ativação. Esse viés b_k pode aumentar ou reduzir a entrada, dependendo de ser positivo ou negativo, respectivamente.

Figura 2 - Modelo rede neural não linear.



Fonte: HAYKIN, 2009. (adaptado)

Onde $x_1, x_2, \ldots, x_m$ são os sinais de entrada; $w_{k1}, w_{k2}, \ldots, w_{km}$ são os respectivos pesos “sinápticos” do neurônio k; u_k (não mostrado na Fig. 2) é a combinação linear devido aos sinais de entrada; b_k é o viés; $\varphi(.)$ é a função de ativação; e y_k é o sinal de saída do neurônio. Matematicamente, o neurônio k na Fig. 5 pode ser descrito pelas equações a seguir:

$$u_k = \sum_{=1}^{m} w_{kj}\, x_j \tag{27}$$

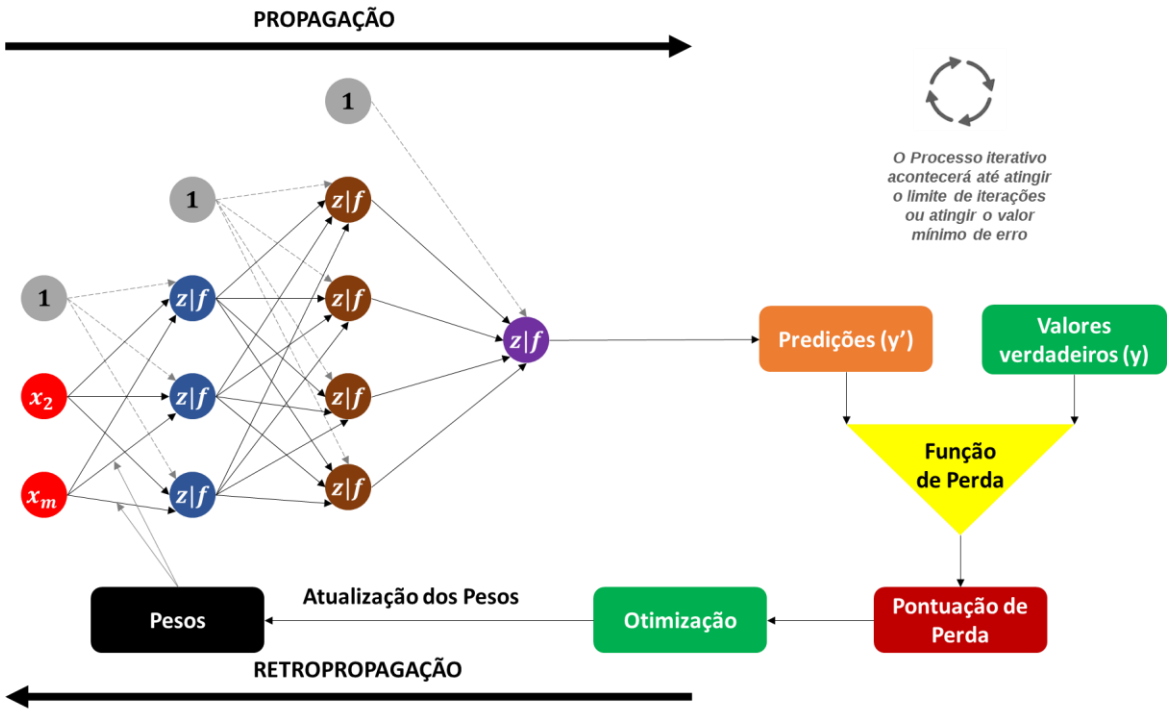
E

$$y_k = \varphi(u_k + b_k) \tag{28}$$

Existem diversos tipos de funções de ativação φ amplamente utilizadas, as funções disponíveis no Scikit-learn são a: *‘identity’*, é uma função de ativação *no-op,* uma abreviação de *"no operation"*, que em português significa "sem operação", ou seja, não há transformação aplicada à entrada, retorna $\mathrm{f(x)} = \mathrm{x}$; ‘*logistic’*, a função sigmoide logística, retorna $\mathrm{f(x)} = \frac{1}{1+\mathrm{e}^{-\mathrm{x}}}$; ‘tanh’, a função tangente hiperbólica, retorna $\mathrm{f(x)} = \tanh(\mathrm{x})$; ‘*relu’*, a função unitária linear retificada, retorna $f(x) = max(0, x)$. (Haykin, 2009)

O processo se repete para cada camada até a camada de saída, com cada camada fornecendo entradas ponderadas para a próxima como é possível observar na Figura *3* a seguir:

Figura 3 - Fluxograma de funcionamento de uma Rede Neural MLP.



Fonte: Autor, PRAMODITHA (2022), adaptado.

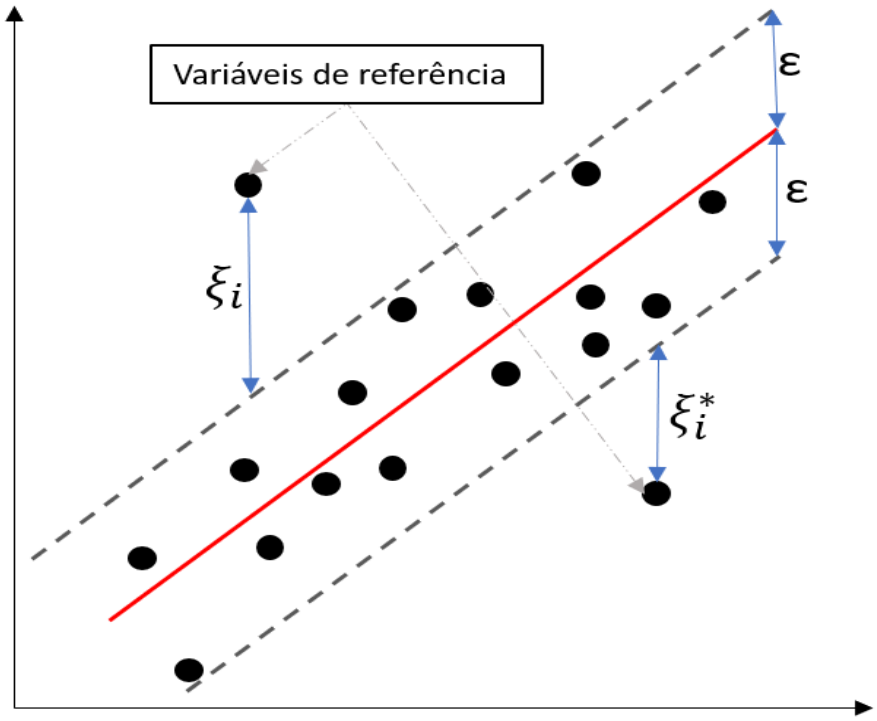
3.11. Support Vector Regression

O SVM (*Support Vector Machine*) foi inicialmente desenvolvido por Vapnik, 1998. O SVM destaca-se em problemas de aproximação de funções com alta dimensionalidade, graças à técnica de kernel, que transforma os vetores de características em um espaço de maior dimensão. Este modelo é um dos mais versáteis e populares no aprendizado de máquina, sendo adequado tanto para tarefas de classificação quanto de regressão, especialmente em pequenos conjuntos de dados complexos. (Sui et al. 2021; Suykens; Vandewalle, 1998) A Suporte Vetorial para Regressão (SVR) é uma extensão do conceito de Máquinas de Vetores de Suporte (SVM) para tarefas de regressão.

A Figura 4 ilustra o funcionamento SVR o modelo busca ajustar um hiperplano (representado pela linha vermelha) que descreve a relação entre as variáveis de

entrada e a variável de saída, mantendo os desvios dentro de uma margem de tolerância definida pelo parâmetro ϵ. As linhas tracejadas paralelas delimitam essa margem de tolerância ϵ, dentro da qual os desvios dos valores reais (pontos pretos) em relação ao hiperplano são considerados aceitáveis e não contribuem para a penalização na função de custo (Vapnik, 2000; Awad; Khanna, 2015; Arshad et al., 2021).

Figura 4 - Regressão vetorial de suporte para modelagem de resposta.



Fonte: Autor, 2025; Arshad et al., 2021 (adaptado)

Contudo, pontos que ultrapassam essa margem geram desvios residuais, representados pelas variáveis ξ (quando o desvio é positivo) e ξ_i^* (quando o desvio é negativo). Esses desvios são penalizados no processo de otimização, forçando o modelo a buscar um equilíbrio entre minimizar os erros e evitar um ajuste excessivo (*overfitting*). O SVR é, portanto, uma técnica robusta que permite flexibilidade ao lidar com ruídos nos dados, mantendo a capacidade de generalização. (COMITO, C.; PIZZUTI, C.; 2022)

Dado um conjunto de dados com n exemplos de treinamento $\{(x_i, y_i)\}_{i=1}^{n}$ onde $x_i \in \mathbb{R}^d$ são as características de entrada e onde $y_i \in \mathbb{R}$. O objetivo do SVR é encontrar uma função $f(x)$ que seja o mais "plana" possível e que se desvie dos

valores reais y_i por no máximo uma certa margem ϵ Awad e Khanna (2015). No SVR, a função $f(x)$ é representada como:

$$f(x) = \hat{y} = w^T \psi(x) + b + \epsilon \tag{29}$$

Na regressão por vetores de suporte (SVR), a função de perda desempenha um papel crucial, e o uso de ψ para representar essa função de perda ajuda a explicar como o erro é tratado no modelo. onde ψ é um mapeamento que cria um espaço de características lineares com dimensão $\mathbb{R}^d$ para os dados de entrada. Usando uma função de perda insensível a ϵ, o modelo SVM difere de outros modelos de regressão linear. De acordo com Awad e Khanna (2015), um erro maior que ε é considerado indesejável. Ou seja, o objetivo do SVM básico é identificar os coeficientes ótimos w e b tais que a função, f, não contenha erros superiores a ε. Por esse motivo, esse modelo também é conhecido como SVM de margem rígida. Assim podemos definir o modelo como:

$$min_{w,b} \ \frac{1}{2} w^T w \tag{30}$$

$$\begin{cases} y_i - w^T \psi(x_i) - b \leq \epsilon \\ w^T \psi(x_i) + b - y_i \geq \epsilon \end{cases} \qquad \forall i \in \{1, 2, \dots, N\} \tag{31}$$

No entanto, nem sempre é viável encontrar um mínimo sob estas restrições. Portanto, a seguinte função de perda é introduzida:

$$\xi_\epsilon(\hat{y}_i, y_i) = \begin{cases} 0, \ |y_i - \hat{y}_i| < \epsilon \\ |y_i - \hat{y}_i| - \epsilon, \ \ senão \end{cases} \qquad \forall i \in \{1, 2, \dots, N\} \tag{32}$$

Assim a solução primal deste problema torna-se:

$$min_{w,b,\xi_i,\xi_i^*} \ \frac{1}{2} \left|\left|w\right|\right|^2 + C \sum_{i=1}^{n} \xi_i + \xi_i^* \tag{33}$$

Sujeitada a:

$$\begin{aligned} y_i - w^T x_i &\leq \epsilon + \xi_i^* \quad | \quad i = 1 \dots N \\ w^T x_i - y_i - &\leq \epsilon + \xi_i \quad | \quad i = 1 \dots N \\ \xi_i, \xi_i^* &\geq 0 \quad | \quad i = 1 \dots N \end{aligned} \tag{34}$$

O erro de regressão aumentará para amostras fora da banda ε, mas amostras aceitáveis são aquelas com erro previsto menor que ε, conforme. Para tornar a otimização possível com limitações que de outra forma seriam impraticáveis, as variáveis da margem limite ξ_i e ξ_i^* são introduzidas para fornecer uma margem suave e permitir erros de medição. (Awad; Khanna, 2015)

O objetivo do SVR é minimizar a norma do vetor de pesos w (que controla a "planicidade" da função) enquanto permite algum desvio ϵ dos valores alvo y_i. Para lidar com erros que excedem essa margem, são introduzidas variáveis de folga ξ_i e ξ_i^*, que representam os erros de superestimação e subestimação, respectivamente.

Assim, o problema de otimização primal (33) quando colocada na forma Lagrangiana, é introduzido os multiplicadores de Lagrange não-negativos para cada restrição (34). α_i e α_i^* para as duas primeiras restrições de erro, e η_i e η_i^* para a terceira. Assim a Lagrangiana do problema se torna:

$$L(w,b,\xi,\xi^*,\alpha,\alpha^*,\eta,\eta^*) = \frac{1}{2}\left||w|\right|^2 + C\sum_{i=1}^{n}(\xi+\xi^*) + \alpha_i\sum_{i=1}^{n}(\epsilon+\xi_i - y_i + w.x_i + b) - \alpha_i^*\sum_{i=1}^{n}(\epsilon+\xi_i^* + y_i - w.x_i - b) - \sum_{i=1}^{n}(\eta_i\xi_i) - \sum_{i=1}^{n}(\eta_i^*\xi_i^*) \quad (35)$$

Para encontrar o valor otimizado, ou seja, com o mínimo de erro basta derivar a equação anterior (35) em relação as variáveis primais w, b, ξ, ξ^* para obter o valor mínimo. Isso requer definir as derivadas parciais de L em relação a essas variáveis como zero. A minimização em relação a w.

$$\frac{\partial L}{\partial w} = w - \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\,x_i = 0 \quad (36)$$

Resultando em:

$$w = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*)\,x_i \quad (37)$$

A minimização em relação a b.

$$\frac{\partial L}{\partial b} = \sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \quad (38)$$

Assim:

$$\sum_{i=1}^{n}(\alpha_i - \alpha_i^*) = 0 \quad (39)$$

Minimização em relação a $\xi \; e \; \xi^*$:

$$\frac{\partial L}{\partial \xi_i} = C - \alpha_i - \eta_i = 0 \tag{40}$$

$$\frac{\partial L}{\partial \xi_i^*} = C - \alpha_i^* - \eta_i^* = 0 \tag{41}$$

Como $\eta_i, \eta_i^* \geq 0$, é possível chegar nas seguintes conclusões:

$$0 \leq \alpha_i \leq C \tag{42}$$

$$0 \leq \alpha_i^* \leq C \tag{43}$$

Substituindo:

$$min_{\alpha,\alpha^*} \frac{1}{2} \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_i - \alpha_i^*) \langle x_i, x_j \rangle + \epsilon \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) - \sum_{i=1}^{n} y_i (\alpha_i - \alpha_i^*) \tag{44}$$

No caso de ser necessário uma fronteira de decisão não-linear, podemos usar o truque do kernel para substituir o produto interno $\langle x_i, x_j \rangle$ por uma função de kernel $K(x_i, \; x_j)$ que mapeia os dados para um espaço de características de dimensão mais alta. Isso transforma o problema dual em:

$$min_{\alpha,\alpha^*} \frac{1}{2} \sum_{i=1}^{n} (\alpha_i - \alpha_i^*)(\alpha_i - \alpha_i^*) K(x_i, x_j) + \epsilon \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) - \sum_{i=1}^{n} y_i (\alpha_i - \alpha_i^*) \tag{43}$$

Sujeita às mesmas restrições anteriores obtidas pelas derivadas. Depois de resolvermos os valores ótimos de α_i e α_i^*, a função de predição para uma nova entrada x é:

$$f(x) = \sum_{i=1}^{n} (\alpha_i - \alpha_i^*) \, K\left(x_i, x_j\right) + b \tag{45}$$

As funções kernel mais utilizadas são: Linear, polinomial, *Radial Basis Function* (RBF), Sigmoide. Segundo Schölkopf (2001), a função kernel linear que é definida como: $K\left(x_i, x_j\right) = x_i . x_j$, este tipo de função é adequado quando os dados são aproximadamente linearmente separáveis, possui como vantagem ser simples e eficiente em problemas de alta dimensionalidade. A função kernel polinomial é

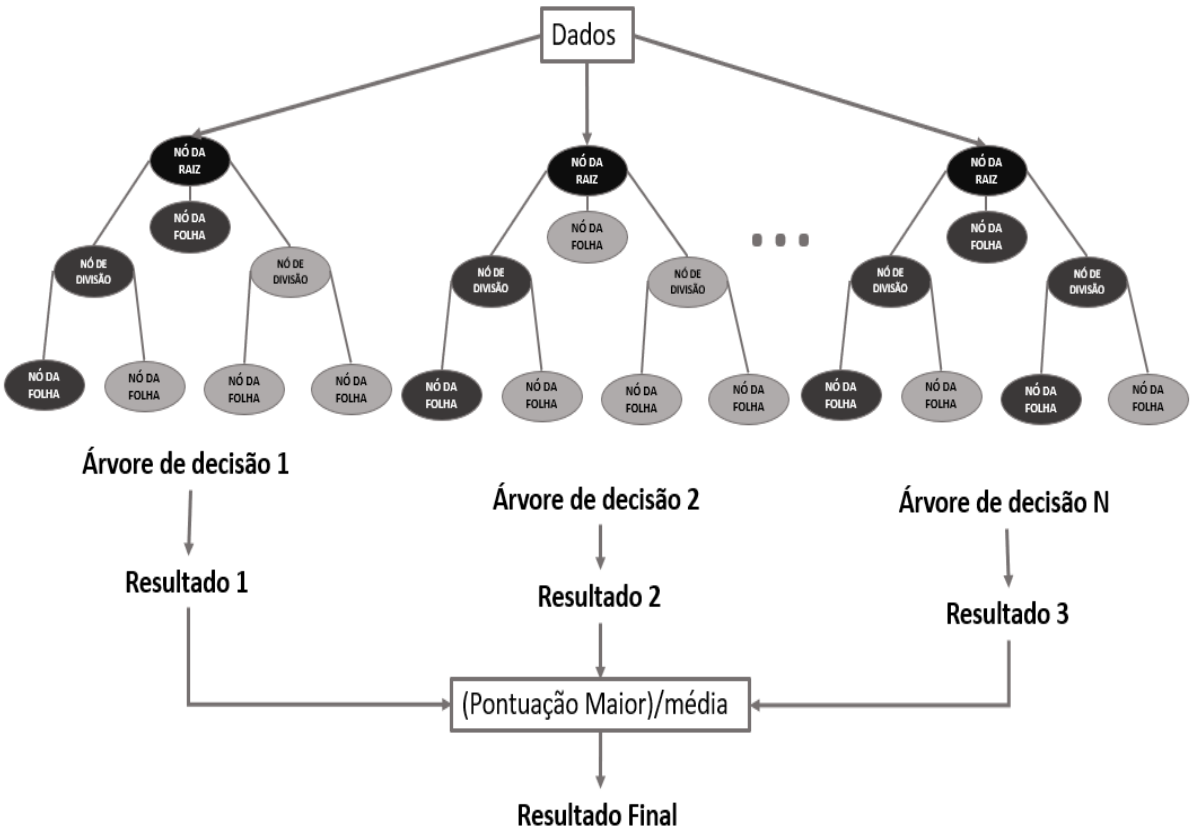
definida por: $K(x_i, x_j) = (\gamma . x_i . x_j + c)^d$. C, d e γ são parâmetros ajustáveis, d representa o grau do polinômio, γ controla a largura do kernel (geralmente ajustado entre 0 e 1, c representa o deslocamento do kernel. É bastante útil quando existe uma relação polinomial entre as características. Por outro lado, a função kernel RBF é definida por: $K(x_i, x_j) = e^{\left(-\gamma ||x_i - x_j||^2\right)}$. γ é um parâmetro ajustável e assim como no kernel polinomial ele tem a função de controlar a largura do kernel. É adequado para dados complexos e não linearmente separáveis. Por fim a função kernel sigmoide: $K(x_i, x_j) = \tan(\gamma . x_i . x_j + c)$, os parâmetros variáveis seguem o mesmo raciocínio do kernel polinomial. E pode funcionar bem em alguns tipos de dados não lineares.

3.12. Random Forest

O *Random Forest* é um método de aprendizado de máquina supervisionado amplamente utilizado para tarefas de classificação e regressão. Introduzido por Leo Breiman em 2001, o *Random Forest* pertence à categoria de algoritmos baseados em conjuntos (*ensemble methods*), que combinam múltiplos modelos fracos para formar um modelo robusto e preciso. A base do *Random Forest* está nas árvores de decisão individuais, cuja combinação permite melhorar a generalização do modelo e reduzir o risco de *overfitting*.

O *Random Forest* constrói uma coleção de árvores de decisão a partir de subconjuntos aleatórios dos dados de treinamento e utiliza amostras aleatórias das características (variáveis preditoras) em cada divisão dos nós das árvores, como é possível observar na Figura 5. Esse processo de amostragem, conhecido como *Bagging* (ou *Bootstrap Aggregating*), promove a independência entre as árvores, aumentando a robustez e a precisão do modelo final. (Sarker et al., 2021)

Figura 5 - Árvore de decisão do modelo *Random Forest*.



Fonte: Autor, 2025; Sarker 2021a (adaptado)

Uma das principais vantagens do *Random Forest* é sua capacidade de lidar com grandes volumes de dados e com um número elevado de variáveis, mesmo na presença de ruído ou dados faltantes. Além disso, o algoritmo fornece métricas importantes, como a importância das variáveis (*feature importance*), facilitando a interpretação dos resultados (BREIMAN, 2001).

Seja um conjunto de N árvores de decisão, denotadas por $h_n(x)$, onde $n = 1, \ldots, N$. A previsão final do modelo *Random Forest* para um dado ponto x é dada por:

$$\hat{y}(x) = \frac{1}{N}\sum_{n=1}^{N} h_n(x) \tag{46}$$

Onde:

- $h_n(x)$ representa a previsão da n-ésima árvore para x .
- N é o número total de árvores na floresta.

Cada árvore do Random Forest é construída usando *Bootstrap Aggregating (Bagging)* e seleção aleatória de variáveis. Para cada árvore h_n, amostramos um subconjunto D_n do conjunto de treinamento N com reposição (técnica de *bootstrap*) (Hastie; Tibshirani; Friedman, 2009). Em cada nó da árvore, em vez de considerar todas as variáveis, escolhemos aleatoriamente um subconjunto de m variáveis**:**

$$D^n = \{(X_i, y_i)\}_{\boldsymbol{i=1}}^{\boldsymbol{m_n}} \tag{46}$$

Onde cada D^n tem o mesmo tamanho de D (ou seja, m_n), mas pode incluir amostras duplicadas devido à reposição. Cada árvore de decisão f_n na floresta é treinada em um conjunto de dados *bootstrap:* D^n. Em cada nó na árvore, seleciona-se aleatoriamente um subconjunto de atributos $F \subset \{1,2,\dots,n\} \mid F \mid < n)$. O objetivo em cada nó é encontrar o atributo $j \in F$ e o limiar t_j que minimiza o erro, geralmente o Erro Quadrático Médio (EQM). (Hastie; Tibshirani; Friedman, 2009). Calculado como:

$$EQM\left(j, t_j\right) = \frac{1}{|L|}\sum_{i \in L}(y_i - \bar{y}_L)^2 + \frac{1}{|L|}\sum_{i \in L}(y_i - \bar{y}_R)^2 \tag{47}$$

Onde:

- $L\ e\ R$ são os conjuntos de amostras divididos pelo limiar t_j no atributo j,
- $\bar{y}_R\ e\ \bar{y}_L$ são as médias dos valores-alvo y para as amostras em $L\ e\ R$.

Para entender a precisão do Random Forest, é útil analisar a decomposição de viés e variância do erro. Para uma única árvore, o erro pode ser decomposto como:

$$\mathbb{E}[(y - \hat{y}_i)^2] = Viés^2 + Variância + Erro\ Irredutível \tag{51}$$

Ao fazer a média de várias árvores, o *Random Forest* reduz a variância porque a média de N árvores reduz a variância por um fator de *N*. conforme mostrado:

$$Variância_{Floresta} = \frac{Variância_{Árvore}}{N} \tag{52}$$

Essa redução na variância, combinada com o mínimo viés de árvores profundas, torna o *Random Forest* um modelo estável e preciso para tarefas de regressão. (Hastie; Tibshirani; Friedman, 2009)

3.13. Grid Search

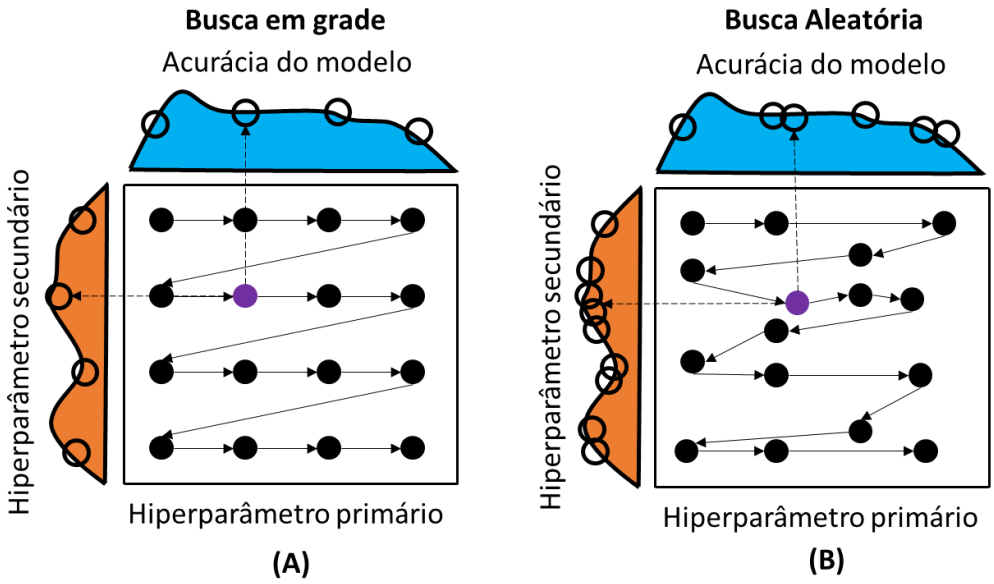
O *Grid Search* é uma técnica sistemática de otimização de hiperparâmetros amplamente utilizada em aprendizado de máquina. Seu principal objetivo é encontrar a combinação de parâmetro de ajuste que maximize o desempenho de um modelo, com base em uma métrica de avaliação previamente definida. Essa abordagem consiste em testar exaustivamente todas as combinações possíveis de valores para os parâmetros de controle especificados dentro de um espaço definido, garantindo que o melhor conjunto seja identificado. (Ghate; Hemalatha, 2023)

O *Grid Search* apresenta diversas vantagens. Por ser uma abordagem sistemática, ele garante que todas as combinações sejam testadas dentro do espaço definido. Além disso, sua facilidade de implementação e ampla adoção em bibliotecas como o S*cikit-learn* tornam-no acessível para muitos usuários. Por fim, os resultados confiáveis ajudam a identificar a melhor combinação de hiperparâmetros para o modelo. (Pedregosa et al., 2011)

Entretanto, o custo computacional é uma limitação significativa. O número de combinações cresce exponencialmente com o número de hiperparâmetros e seus valores, tornando-o inviável para grades muito amplas. Além disso, pode ser ineficiente em espaços de busca grandes, especialmente quando muitas combinações geram desempenhos semelhantes. (Attar et al., 2024)

O funcionamento do *Grid Search* envolve algumas etapas essenciais. Primeiro, o usuário deve definir o espaço de busca, especificando os parâmetros de ajuste do modelo a serem otimizados e os valores possíveis para cada um. Em seguida, todas as combinações de parâmetros de controle são geradas, formando uma "grade" de opções (Figura 6). Por exemplo, se dois parâmetros de controle possuem três valores cada, o total de combinações serão nove. (Pilario et al., 2021; Belete; Huchaiah, 2022)

Figura 6 - (a) busca em grade; e (b) busca aleatória.



Fonte: (Pilario et al., 2021; Shim et al., 2023) adaptado.

Na etapa seguinte, o modelo é treinado e avaliado para cada combinação de hiperparâmetros, utilizando validação cruzada ou um conjunto de validação separado. Por fim, a combinação de parâmetro de controle que obtiver o melhor desempenho em uma métrica especificada é selecionada como a melhor configuração. (Pilario et al., 2021)

Grid Search e *Random Search* são métodos usados para otimizar hiperparâmetros em modelos de machine learning, mas diferem na abordagem e eficiência. O *Grid Search* testa todas as combinações possíveis de valores dentro de um espaço definido, garantindo uma busca exaustiva. Apesar disso, é computacionalmente caro e ineficiente para espaços de busca grandes, já que avalia combinações redundantes em regiões de baixa relevância. Por outro lado, o *Random Search* seleciona combinações de maneira aleatória, cobrindo o espaço de busca de forma mais ampla e eficiente (Figura 6), especialmente em cenários com muitos parâmetros ou valores possíveis (Sukamto; Hadiyanto; Kurnianingsih, 2023; Belete; Huchaiah, 2022).

Enquanto o *Grid Search* é ideal para espaços pequenos e bem definidos, o Random Search se destaca em contextos em que poucos hiperparâmetros têm impacto significativo no desempenho do modelo. Shim et al. (2023) observaram que o *Random Search* frequentemente encontra boas soluções mais rapidamente, ao

passo que o *Grid Search* pode gastar tempo avaliando configurações menos relevantes. Assim, a escolha entre os dois métodos depende da dimensão do espaço de busca, do orçamento computacional disponível e da importância de explorar sistematicamente todas as combinações possíveis.

A escolha da métrica de avaliação (ou *scoring*) é essencial para determinar a qualidade de um modelo. O *Scikit-learn*, uma das bibliotecas mais populares de aprendizado de máquina em Python, oferece uma ampla gama de opções de *scoring,* que variam de acordo com o tipo de problema. (Pedregosa et al., 2011)

Para problemas de regressão, as métricas que avaliam a diferença entre os valores previstos e os valores reais:

Coeficiente de determinação ('r2'), mede o quão bem o modelo explica a variância dos dados. A variância explicada é matematicamente similar ao R^2, pois ambos calculam a proporção da variação explicada. A principal diferença é que a variância explicada pode ser usada em outros contextos, incluindo modelos de classificação, PCA etc; enquanto o R^2 é comumente usado em regressão (Pedregosa et al., 2011; Figueiredo; Silva; Rocha, 2011). Ambos são calculados pela mesma equação a seguir:

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \overline{y})^2} \tag{53}$$

O Erro absoluto médio negativo (*'neg_mean_absolute_error'*): Diferença média absoluta entre as previsões e os valores reais. (Pedregosa et al., 2011). Também conhecido como MAE_{neg} (*Negative Mean Absolute Error*) é dado pela equação:

$$MAE_{neg} = -\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i\,| \tag{54}$$

Erro quadrático médio negativo (*'neg_mean_squared_error'*): Penaliza erros maiores de forma mais severa. O *Scikit-learn* usa o valor negativo do MSE para que a métrica funcione em algoritmos que maximizam a pontuação (como em validações cruzadas). Isso significa que para valores menos negativos (mais próximos de 0) indicam melhor desempenho do modelo, e valores mais negativos indicam pior desempenho. (Pedregosa et al., 2011) É dada pela equação:

$$-MSE = -\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad (55)$$

Raiz quadrada do erro quadrático médio ('neg_root_mean_squared_error') é a versão da raiz do MSE raiz do erro quadrático médio, também é conhecido como (RMSE – *Root Mean Squared Error*). O RMSE é expresso na mesma unidade da variável de saída, o que facilita a interpretação (Kiraga et al., 2024; Pedregosa et al., 2011;). Por exemplo, se você está prevendo a produção de biomassa em gramas por litro (g/L), o RMSE também será expresso em g/L. É dada pela equação:

$$-\sqrt{MSE} = -\sqrt{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \quad (56)$$

Métricas gerais, que podem ser aplicadas em diferentes contextos, incluem o Erro máximo (*'max_error'*), que mede o maior erro absoluto em uma predição, e o *Log-Loss* (*'log_loss'*), que avalia a incerteza das previsões probabilísticas. Também há desvios específicos, como Poisson ou Gamma, para distribuições particulares (Pedregosa et al., 2011).

4. METODOLOGIA

Este trabalho utilizou como referência os dados fornecidos por de Farias Silva e Sforza (2016), no total de 145 conjunto de dados foram inseridos, onde foi estudada a produtividade de biomassa e carboidratos em cultivo contínuo sob limitação de nitrogênio, efeito da intensidade luminosa e do tempo de residência na absorção de nutrientes em Chlorella vulgaris. A espécie de microalga foi mantida e cultivada em meio BG11 modificado (tamponado com 10 mM HEPES pH 8), esterilizado em autoclave por 20 min a 121◦C. Os teores de P e N do meio, na forma de K2HPO4 e NaNO3, foram otimizados para estudar a limitação de nutrientes.

Experimentos contínuos foram realizados em fotobioreator de policarbonato de placa plana vertical operando como um CSTR (reator de tanque agitado continuamente), com um volume de trabalho de 300 mL, uma profundidade de 1,2 cm e uma superfície exposta à luz de 250 cm^2. O excesso de CO_2 foi fornecido por uma mistura de CO_2-ar (5% v/v) borbulhando no fundo do reator (1 L h^{-1} da vazão total de gás), que também proporcionou mistura. Um agitador magnético também foi utilizado para evitar qualquer deposição de biomassa e assim garantir uma boa mistura do reator. O meio fresco foi alimentado a uma taxa constante por uma bomba peristáltica (Watson-Marlow sci400, faixa de vazão: 25–250 mL d^{-1}, diretamente relacionada ao tempo de residência). A luz foi fornecida por uma lâmpada LED (*Photon System Instruments*, SN-SL 3500-22) para experimentos contínuos. A intensidade luminosa foi medida nos painéis frontal e traseiro do reator usando um fotorradiômetro (HD 2101.1 da Delta OHM), que quantifica a radiação fotossinteticamente ativa (PAR).

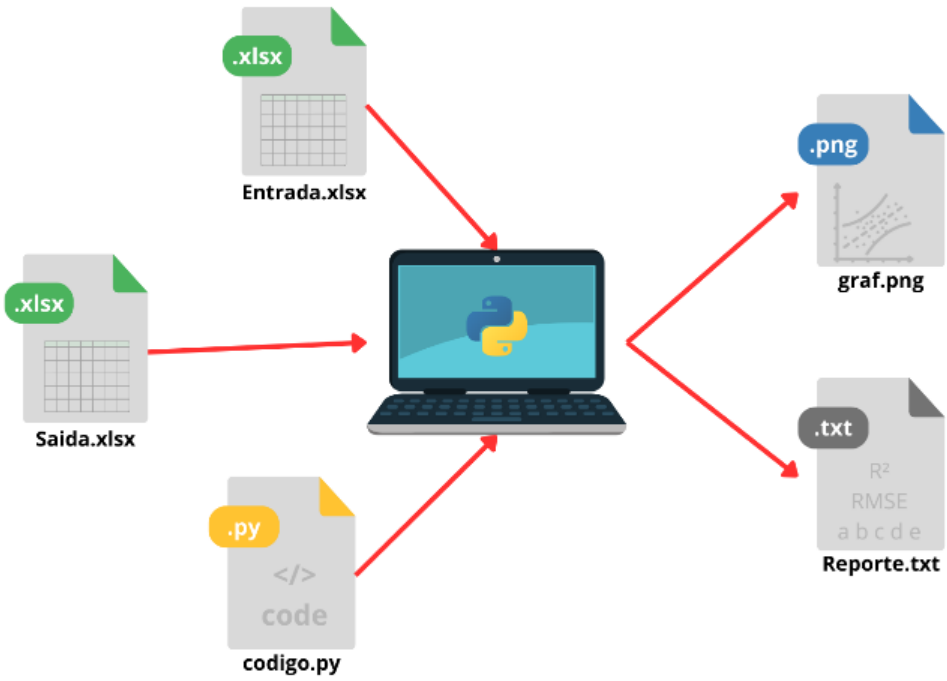
As variáveis estudadas foram na faixa de 150-450 $\mu mol.m^{-2}\ s^{-1}$ de intensidade luminosa, tempos de residência entre 2,9-4,4 dias e concentrações na entrada para N e P entre 500-75 e 200-50 mg L^{-1}, respectivamente. Importante mencionar que para considerar o estado estacionário, medias diárias durante pelo menos 3 dias, foram realizadas para mostrar a constância da concentração celular e porcentagem de carboidrato em biomassa.

A concentração de biomassa foi monitorada diariamente por análise espectrofotométrica da densidade óptica (DO - 750nm) usando um espectrofotômetro UV-vis (UV 500, Spectronic Unicam, Reino Unido). A concentração de biomassa também foi medida gravimetricamente como peso seco (PS) em termos de g L^{-1} em células previamente coletadas com um filtro de 0,22 m e, em seguida, secas por 4

horas a 80◦C em um forno de laboratório. Os nutrientes analisados foram nitrato (N-NO_3) e fosfato (P-PO_4), avaliados em pelo menos três momentos diferentes para cada estado estacionário. As amostras de cultura foram filtradas para medir apenas nutrientes dissolvidos (0,2 m): a concentração de N foi medida por um kit de teste analítico fornecido pela St. Carlo Erba Reagenti, Itália (código 0800.05482) e P foram medidos pelo método do ácido ascórbico descrito em APHA-AWWA-WEF, 1992.

Para realizar as simulações, foi utilizado um notebook *Acer® Aspire 5,* 8gb de ram, processador 12º Geração *Intel® Core™* i5-12450H 2.00 GHz, utilizando Windows® 11. Os dados obtidos foram então organizados e colocados numa planilha no formato (.xlxs) esses dados foram separados em duas partes: as variáveis de entrada e variáveis de saída. Conforme é possível observar no esquema da Figura 7.

Figura 7 – Esquema do processamento de arquivos.



Fonte: Autor (2025)

Para cada teste foram utilizadas uma das variáveis de saída. Na primeira bateria de testes foram utilizados os dados de acúmulo de carboidratos (%), na segunda bateria de testes foram utilizados os dados de produtividade de biomassa (g L^{-1} d^{-1}), e na terceira bateria de testes foi utilizado produtividade de carboidratos (g L^{-1} d^{-1}). Esses dados foram analisados através de um algoritmo escrito em *Python* utilizando uma regressão predefinida. Após serem calculados o algoritmo retornou

duas saídas: A primeira em formato .txt que retornavam os valores de R^2 e do RMSE, e a segunda contendo o gráfico dos dados originais em relação ao que foi estimado pela regressão. A escolha de utilizar R^2 e RMSE para fazer a avaliação se dá pelo fato de serem métricas bem difundidas e utilizadas para esse tipo de tarefa, como é observado por (Kuhn; Johnson, 2013)

No arquivo de entrada existem as variáveis independentes que foram utilizadas para estimar a variável de saída contida no arquivo de saída. As variáveis utilizadas foram o tempo normalizado, densidade óptica, concentração de fósforo, concentração de nitrogênio e intensidade luminosa. O principal critério para considerar o uso dessas variáveis de processo foi o fato de que elas pouco estão correlacionadas entre si, suportando a hipótese de que sejam variáveis independentes. O tempo normalizado se refere ao tempo que se passou desde o início da operação do reator, relacionado com o tempo máximo que ficou em funcionamento, definido pela equação:

$$t_n = \frac{t - t_{min}}{t_{max} - t_{min}} \tag{57}$$

Para utilizar o tempo normalizado em análises de regressão, primeiramente, foram organizados os dados temporais de cada série, que podem ter magnitudes muito variáveis, o que afeta a precisão da regressão. A normalização transformou o tempo em uma escala de 0 a 1 ao dividir cada valor pelo valor máximo do conjunto, criando uma nova variável que permite comparar pontos ao longo do tempo de forma consistente. O restante das variáveis fora obtido integralmente do trabalho realizado por de Farias Silva e Sforza (2016). E estão descritas a seguir na Tabela 3.

Tabela 3 - Variáveis de entrada

Nome	Sigla	Unidade
Tempo normalizado	t_n	adimensional
Fósforo	P	$mg\ L^{-1}$
Nitrogênio	N	$mg\ L^{-1}$
Tempo de residência	τ	dias
Intensidade luminosa	I	$\mu E/m^2 s$
Densidade óptica	OD	adimensional

Os parâmetros analisados neste estudo são essenciais para o controle e a otimização do cultivo de microalgas. O tempo normalizado (t_n) é uma variável

adimensional que permite a comparação direta entre diferentes escalas temporais de experimentos, facilitando a análise de processos com durações distintas. O fósforo (P) e o nitrogênio (N), ambos medidos em mg L^{-1}, são nutrientes cruciais para o metabolismo microalgal. O fósforo é indispensável para a formação de moléculas como ATP e ácidos nucleicos, enquanto o nitrogênio está associado à síntese de proteínas e outros compostos celulares. Concentrações insuficientes desses nutrientes podem limitar o crescimento das microalgas, ao passo que valores excessivos podem causar impactos ambientais, como a eutrofização de corpos d'água (Li et al., 2019).

O tempo de residência (τ), em dias, indica o período médio em que a biomassa ou o fluido permanece no sistema, sendo um parâmetro que afeta diretamente a produtividade e a eficiência do processo. A intensidade luminosa (I), expressa em $\mu E \cdot m^{-2} \cdot s^{-1}$, representa a quantidade de luz disponível para a fotossíntese, fator determinante para a taxa de crescimento das microalgas. Já a densidade óptica (OD ou DO) é uma medida adimensional usada para monitorar a turbidez do meio, funcionando como um indicador indireto da concentração celular (de Farias Silva e Sforza, 2016).

Do mesmo modo as variáveis utilizadas isoladamente nas regressões para a saída (tabela 4) em cada teste, também foram retiradas integralmente do trabalho de Farias Silva e Sforza (2016). A tabela 4 apresentada resume parâmetros essenciais na análise de bioprocessos, especificamente aqueles relacionados à produção de biomassa e carboidratos.

Tabela 4 - Variáveis de saída utilizadas nas regressões.

Nome	Sigla	Unidade
Acúmulo de Carboidratos em Biomassa	%	adimensional
Produtividade da Biomassa	P_X	mg L^{-1} d^{-1}
Produtividade de Carboidratos	P_C	mg L^{-1} d^{-1}

A conversão em carboidratos (%), adimensional, reflete a eficiência do processo na transformação de substratos em carboidratos, fornecendo uma medida importante para a otimização de condições operacionais, como a taxa de consumo de nutrientes. Esse parâmetro é crucial para entender a conversão de recursos e avaliar a eficácia do processo em termos de utilização de substratos. Já a produtividade da biomassa (P_X), expressa em mg L^{-1} d^{-1}, quantifica o crescimento celular, sendo

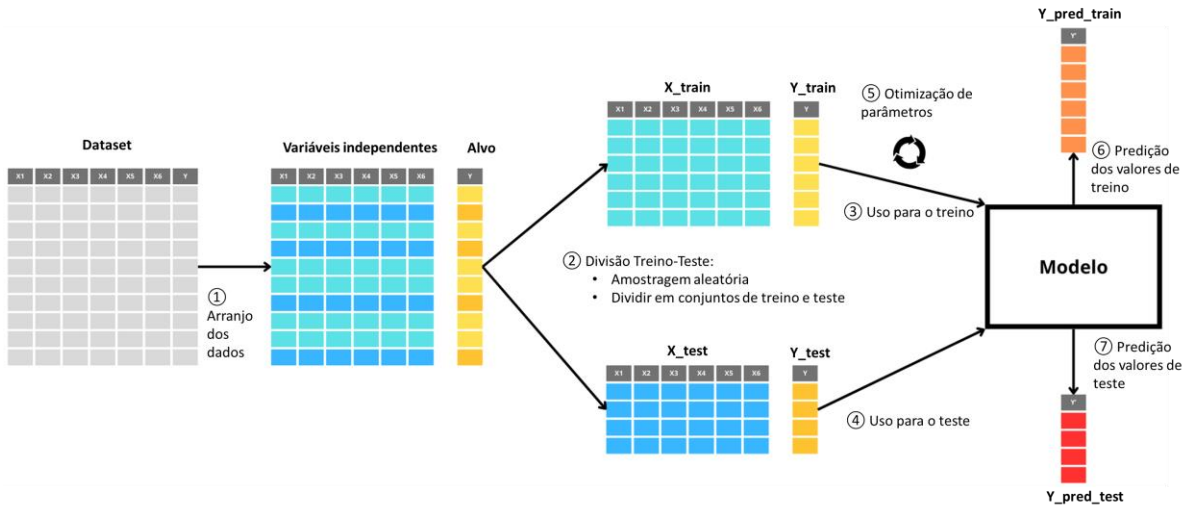
fundamental para o monitoramento da eficiência do cultivo e para garantir que a produção de biomassa esteja em conformidade com os objetivos do processo biotecnológico.

Por sua vez, a produtividade de carboidratos (P_C), também medida em mg L^{-1} d^{-1}, indica a taxa de produção de carboidratos desejada no sistema. Esse parâmetro é de particular importância em processos biotecnológicos que visam à produção de carboidratos como bioprodutos, como na fabricação de biocombustíveis ou alimentos. Juntos, esses parâmetros fornecem uma visão abrangente da eficiência de um bioprocesso, permitindo ajustes finos nas variáveis de operação para maximizar a produção de biomassa ou carboidratos.

A conexão entre os parâmetros de bioprocessos, como produtividade de biomassa e carboidratos, e técnicas de aprendizado de máquina, como o *train-test split*, se dá pela necessidade de análise preditiva e otimização de processos. Modelos preditivos podem ser usados para antecipar o comportamento de sistemas biotecnológicos, baseando-se em dados históricos de produtividade. Para garantir a precisão dessas previsões e evitar ajustes excessivos aos dados de treinamento, a técnica de divisão do conjunto de dados em treinamento e teste se torna essencial. Isso permite que o modelo seja validado de maneira robusta e sua capacidade de generalização seja testada, possibilitando uma análise mais eficaz e controlada do desempenho dos bioprocessos.

O train-test split é uma técnica usada em aprendizado de máquina para dividir um conjunto de dados (*data set)* em duas partes: uma para treinar o modelo (*training set*) e outra para testá-lo (*test set*), como é possível observar na Figura 8. A divisão é feita aleatoriamente, em proporção 80/20, para garantir que o modelo não seja tendencioso. É o padrão recomendado pela própria biblioteca do *Sk-learn (*Pedregosa et al., 2011). O conjunto de treinamento é usado para ajustar o modelo, enquanto o conjunto de teste avalia sua capacidade de generalização em dados não vistos, ajudando a evitar o *overfitting* e fornecendo métricas de desempenho para validar o modelo.

Figura 8 - Processo de regressão dos dados.



Os testes foram conduzidos foram conduzidos utilizando a série de variação de parâmetros contidos na tabela 4 para cada tipo de modelo utilizado, e considerando 50.000 de quantidade máxima de iterações para cada combinação e tolerância mínima de convergência de 10^{-8}.

Utilizou-se a técnica de *grid search* como abordagem principal para a otimização dos parâmetros dos modelos de aprendizado de máquina aplicados no estudo. Essa metodologia consiste em uma busca exaustiva no espaço de hiperparâmetros, avaliando sistematicamente diversas combinações predefinidas para identificar aquelas que maximizam o desempenho dos modelos. o critério de avaliação utilizado foi o *negative mean squared error* (*neg MSE*), que mede a qualidade do ajuste do modelo por meio do erro médio quadrático. Essa métrica, quando maximizada durante o processo de otimização, busca minimizar os erros de predição do modelo.

A Tabela 5 apresenta uma lista de algoritmos de aprendizado de máquina (LASSO, *Ridge*, Redes Neurais Artificiais, SVR e *Random Forest*) juntamente com os parâmetros que podem ser ajustados para melhorar o desempenho de cada modelo. Esses parâmetros possuem valores variados, abrangendo tanto opções contínuas (como o coeficiente de regularização 'alpha' em LASSO e *Ridge*, que varia na escala logarítmica entre 10^{-4} a 10^{4} quanto categorias discretas (como os kernels no SVR ou funções de ativação em Redes Neurais).

Tabela 5 – Relação de parâmetros que foram otimizados e faixa de variação.

Modelo	Parâmetros	Variação
Lasso	'alpha'	np.logspace(-4,4,100)
	'selection'	'Ciclic', 'Random'
	'fit_intercept'	'True', 'False'
	'positive'	'True', 'False'
	'precompute'	'True', 'False'
Ridge	'alpha'	np.logspace(-4,4,100)
	'solver'	'svd', 'cholesky', 'sparse_cg', 'lsqr', 'sag', 'lbfgs'
	'fit_intercept'	'True', 'False'
	'positive'	'True', 'False'
RNA	'hidden_layer_sizes'	(3,), (5,), (6,),(10,), (11,), (12,), (13,),(14,), (15,), (16),(3, 3,), (5, 5,), (7, 7,), (3, 3, 3), (5, 5, 5), (6, 6, 6)
	'activation'	'tanh', 'relu', 'logistic', 'identity'
	'solver'	sgd', 'adam', 'lbfgs'
	'alpha'	0.0001, 0.05, 0.1, 0.5
	'learning_rate'	constant', 'adaptive', 'invscaling'
SVR	'kernel'	'linear', 'poly', 'rbf', 'sigmoid'
	'degree'	1, 2, 3, 4, 5
	'gamma'	'scale', 'auto'
	'coef0'	0, 0.25, 0.75, 0.5, 1, 2
	'C'	0.5, 0.75, 1, 1.5, 2
	'epsilon'	0.1,0.01, 0.001, 0.5, 1
	'shrinking'	'True', 'False'
Random Forest	'n_estimators'	20, 50, 100, 150, 200
	'max_depth'	None, 5, 10, 20, 30
	'min_samples_split'	2, 3, 4, 5, 6, 7, 10
	'min_samples_leaf'	1, 2, 3, 4
	'bootstrap'	'True', 'False'

A Tabela 5 apresentada sintetiza os hiperparâmetros considerados durante o ajuste fino de modelos utilizados no estudo. Para os modelos de regressão LASSO e *Ridge*, o principal parâmetro de controle é alpha, que regula a intensidade da penalização imposta aos coeficientes, variando em uma escala logarítmica para explorar uma ampla faixa de valores. No caso do LASSO, os parâmetros adicionais "*selection"* e "*positive"* permitem customizar a estratégia de atualização dos coeficientes (como seleção cíclica ou aleatória) e impor restrições como coeficientes exclusivamente positivos. Essas configurações são importantes para equilibrar a capacidade de generalização e a interpretabilidade dos modelos.

No modelo de redes neurais (*Multi-Layer Perceptron – MLP*), foi explorada uma ampla diversidade de configurações estruturais por meio do hiperparâmetro "*hidden_layer_sizes"*, que define o número de neurônios e camadas ocultas. Combinações de tamanhos de camadas simples e múltiplas foram testadas, permitindo avaliar o impacto da complexidade estrutural sobre o desempenho. Além disso, funções de ativação, como "*tanh"* e "*relu"*, foram incluídas para capturar não linearidades dos dados. Diferentes otimizadores ("*sgd", "adam", "lbfgs"*) foram utilizados para comparar a eficiência na convergência, enquanto ajustes na taxa de aprendizado ("*learning_rate"*) permitiram testar o impacto de velocidades constantes, adaptativas e inversamente escalonadas no treinamento.

Para o *Support Vector Regression* (SVR), os hiperparâmetros controlam aspectos centrais do modelo. O tipo de kernel ("*linear", "poly", "rbf", "sigmoid*") define a natureza da função de decisão, enquanto o grau (*"degree")* é ajustado especificamente para kernels polinomiais, permitindo capturar diferentes níveis de complexidade. O parâmetro "*gamma"* regula a influência de cada ponto de suporte, e "épsilon" define a margem de tolerância ao erro na predição.

No caso do modelo Random Forest, foram testados diferentes valores para o número de estimadores ("*n_estimators"*), que controla a quantidade de árvores na floresta. Também foram ajustadas a profundidade máxima das árvores ("*max_depth*"), o número mínimo de amostras para dividir um nó *("min_samples_split*") e o número mínimo de amostras em cada folha terminal ("*min_samples_leaf*"). Esses parâmetros influenciam diretamente a capacidade do modelo de capturar padrões complexos sem superajustar. Por fim, o uso ou não de amostragem com reposição ("*bootstrap*") foi avaliado, permitindo analisar o impacto da variação dos dados de treinamento na robustez final do modelo.

Por fim, neste trabalho também está sendo proposto o índice de generalização para regressões (*Regression Generalization – REGE*) ele foi calculado a relação de treino e teste, dado para a equação a seguir:

$$Índice = \left(\frac{f_{teste} - f_{treino}}{f_{treino}}\right) \times 100 \quad (57)$$

Para o R^2, pela equação 57 a seguir:

$$Índice = \left(\frac{R^2_{teste} - R^2_{treino}}{R^2_{treino}}\right) \times 100 \quad (57)$$

Um modelo com boa generalização em relação ao treino apresentará valores de R^2_{treino} e R^2_{teste} próximos. Isso indica que ele não apenas ajusta bem os dados de treinamento, mas também é capaz de prever corretamente em um conjunto de dados não visto. Por outro lado, grandes discrepâncias entre esses valores, como um R^2_{treino} muito maior que o R^2_{teste} , são um sinal claro de *overfitting*. Nesse caso, o modelo aprende não só os padrões reais dos dados, mas também os ruídos ou variações específicas do conjunto de treinamento, comprometendo sua capacidade de generalização.

Seguindo o mesmo raciocínio o índice REGE foi calculado a relação de treino e teste para o RMSE, pela equação 58 a seguir:

$$Índice = \left(\frac{RMSE^2_{teste} - RMSE^2_{treino}}{RMSE^2_{treino}}\right) \times 100 \quad (58)$$

O índice vai mostrar a variação relativa entre o erro de treino e o erro de teste. Se o índice for 0%, isso indica que o erro de teste e o de treino são iguais. Valores positivos indicam que o erro de teste é maior do que o erro de treino, enquanto valores negativos indicam o contrário. Esse tipo de índice é útil para entender a generalização de um modelo, ajudando a identificar sobre ajuste (*overfitting*) se o erro de treino for muito menor que o de teste.

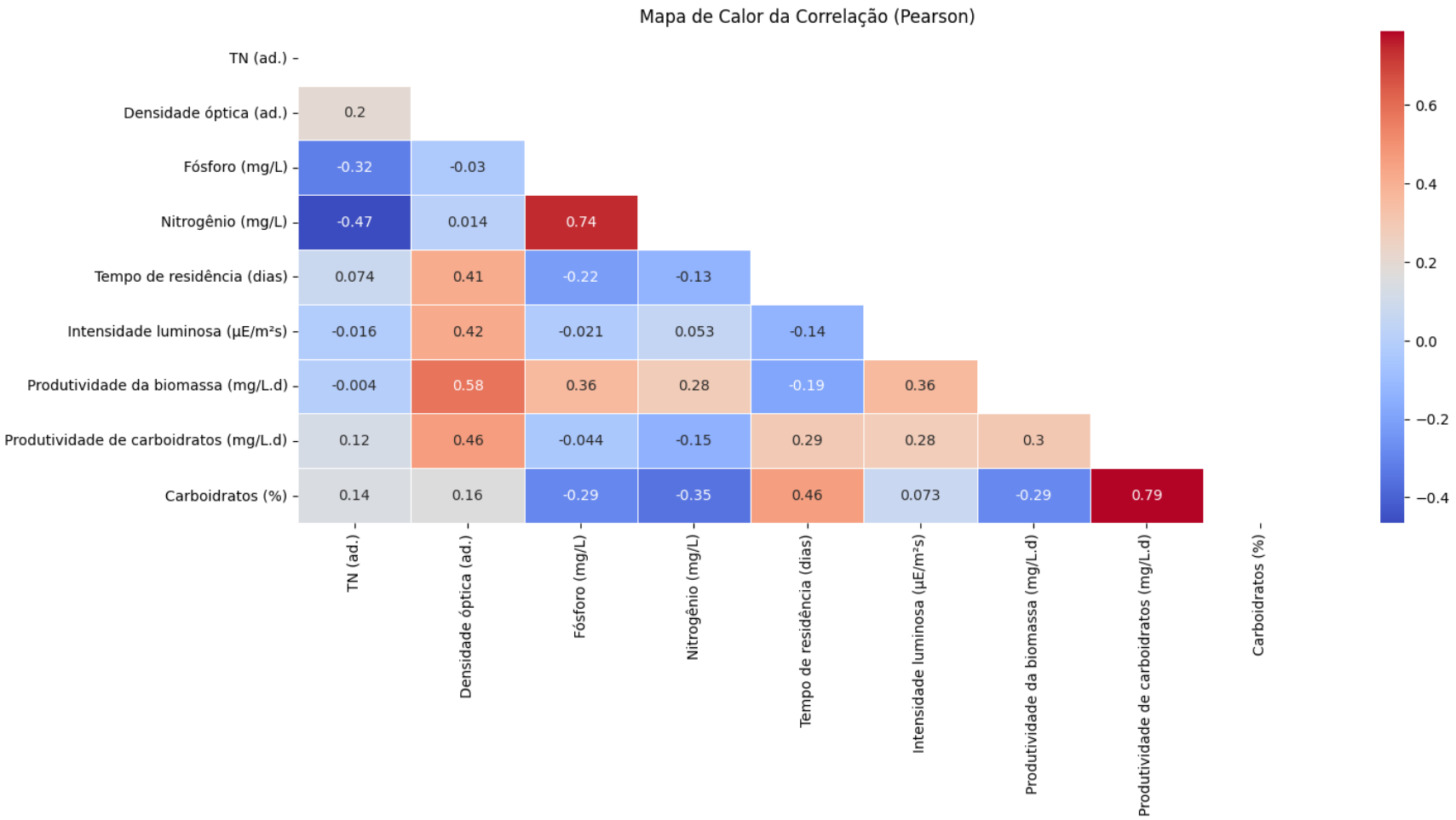
5. RESULTADOS E DISCUSSÃO

Uma das primeiras condições validadas na construção de todas a análise é a observação da correlação de Pearson. Esta correlação é muito importante para que seja possível selecionar as variáveis independentes corretamente. É possível observar na Figura 9 que não existe correlação significativa entre maioria das variáveis de entrada contidas na Tabela 3 entre si, com exceção da concentração de fósforo e nitrogênio que possuem correlação de 0,74.

Isso acontece devido a necessidade de o microrganismo sintetizar proteínas essenciais a sua sobrevivência, onde o nitrogênio é intrinsecamente envolvido nas reações metabólicas. Ao passo que o Fósforo é consumido para que essas reações aconteçam, como é evidenciado nos trabalhos por de Farias Silva e Sforza (2016), de Farias Silva e Bertucco (2019) e Fu et al. (2019). Assim, como o fósforo também está relacionado com síntese de carboidratos e lipídios a correlação com o nitrogênio naturalmente não é maior. Essa correlação alta entre o nitrogênio e o fósforo é um indicativo de que possa existir sobreajuste nos modelos utilizados. Como alternativa, poderia ao invés de considerar as duas variáveis Nitrogênio e fósforo separadas considerar a razão N/P como foi utilizado por Hossain et al. (2022).

Com relação aos valores das variáveis de saída (Tabela 4), as variáveis independentes, ou seja, as variáveis de entrada possuem baixa correlação, com valores inferiores a 0,5 às variáveis de entrada. Com exceção densidade óptica (OD) que possuiu correlações maiores que 0.5 com as variáveis de saída. Existem casos que nas variáveis de entrada, uma ou outra correlação possui um certo grau de correlação, mas nenhuma obteve $R^2 > 0,5$. Velásquez-Orta et al. (2024) ao correlacionar todas as variáveis estudadas do seu sistema de cultivo de microalgas também obteve baixa correlação entre as variáveis na maioria das vezes, ressaltando a complexidade do sistema que envolve cultivo de microrganismos. Onde apenas a concentração e iluminação possuíam correlação positiva.

Figura 9 - Correlação de Pearson entre todas as variáveis utilizadas.



As cores mais quentes (vermelho e laranja) indicam uma correlação positiva entre as variáveis. Isso significa que quando uma variável aumenta, a outra tende a aumentar também. Por exemplo, a produtividade de carboidratos apresenta uma forte correlação positiva com a produtividade da biomassa, sugerindo que um aumento na biomassa está associado a um aumento na produção de carboidratos.

As cores mais frias (azul) indicam uma correlação negativa. Neste caso, quando uma variável aumenta, a outra tende a diminuir. Por exemplo, o nitrogênio apresenta uma correlação negativa com o tempo de residência, indicando que um aumento no tempo de residência pode estar associado a uma diminuição nos níveis de nitrogênio. As cores próximas ao branco indicam uma correlação fraca ou nula entre as variáveis.

Os nutrientes (nitrogênio e fósforo) apresentam correlações positivas com a produtividade da biomassa e dos carboidratos, indicando que a disponibilidade de nutrientes é um fator importante para a produção biológica. O tempo de residência apresenta correlações mistas. Enquanto há uma correlação negativa com o nitrogênio, há uma correlação positiva com a produtividade da biomassa, sugerindo que o tempo de retenção da água pode influenciar a dinâmica dos nutrientes e a produção biológica de maneiras complexas. A intensidade luminosa apresenta correlações positivas com a produtividade da biomassa e dos carboidratos, o que é esperado, pois a luz é essencial para a fotossíntese como é observado por Marino (2018) e Six (2024b).

O mapa de calor revela um intrincado conjunto de relações entre as variáveis analisadas. Destaca-se a forte correlação positiva entre a produtividade da fração de Carboidratos e a produtividade de carboidratos, indicando que um aumento na porcentagem de carboidratos está diretamente ligado a um aumento na produção de carboidratos, isso se deve a dependência entre si. Também é possível observar essa forte correlação entre o nitrogênio e o fósforo, isso se deve ao fato de serem nutrientes indispensáveis para o crescimento celular e ao fato de seguirem uma proporção entre si. Esse fenômeno também foi observado por Huang et al. (2021) e Xing et al. (2021), quando cultivaram a *Chlorella vulgaris.*

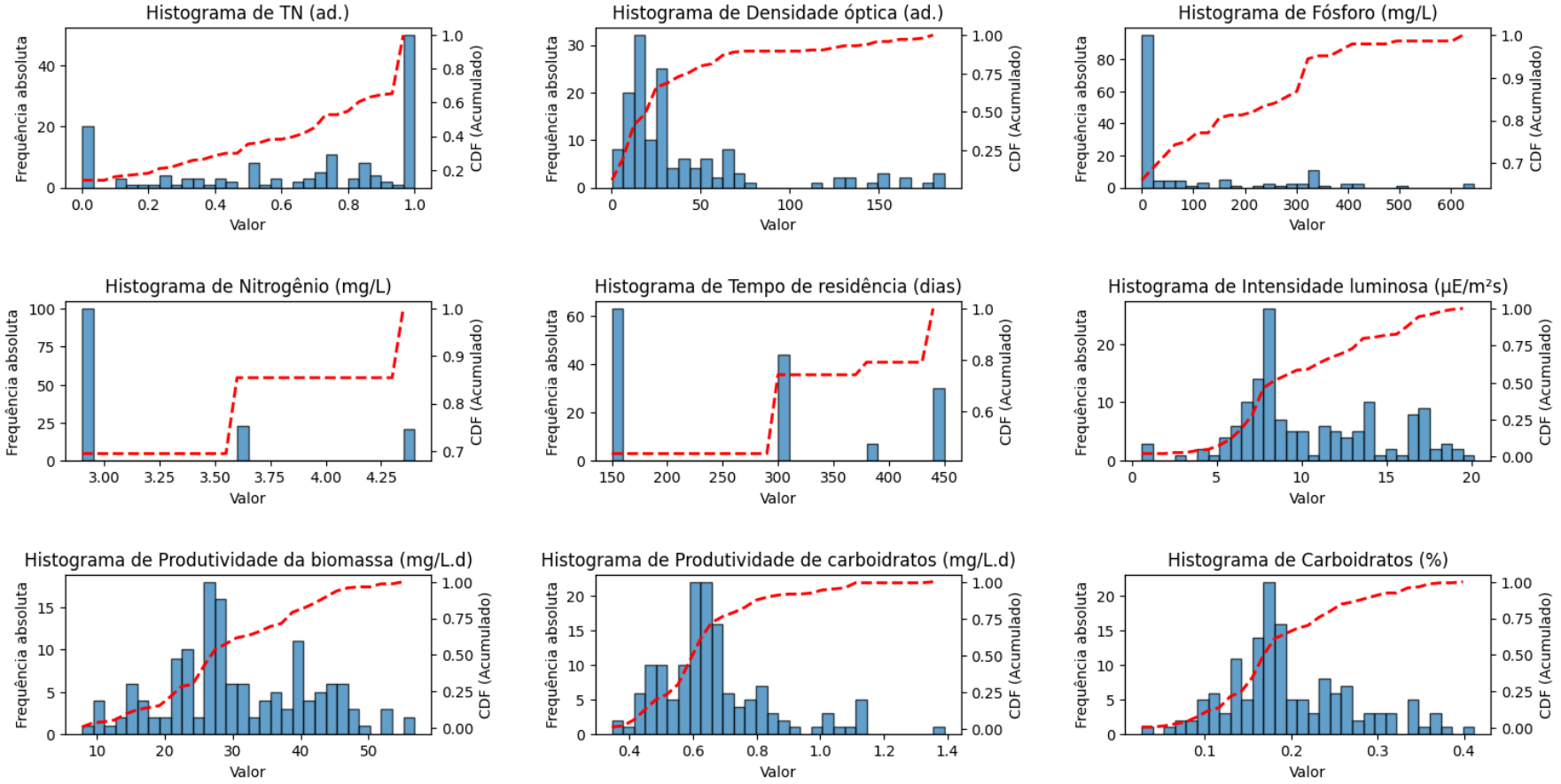
Em seguida, foi feito uma análise com relação à distribuição dos dados, ou seja, como ele se comportava com relação à sua própria grandeza. A construção de histogramas se mostrou uma ferramenta muito eficaz na percepção de como certos conjuntos de dados possuem certa assimetria.

Na Figura 10 é possível observar como se dá organização das grandezas de cada variável. É notável que nas condições de entrada existe (A – F) os dados tendem a possuir acúmulos em uma determinada margem do histograma, dessa forma é possível inferir que não há uma distribuição normal neles, em alguns casos como o tempo de residência e intensidade luminosa, ou seja, os histogramas D e E existem espaços vazios nos conjuntos de dados em determinados limites. Assim como em B e C para o fósforo e nitrogênio respectivamente, possuem picos muito altos nos limites do histograma. Essa assimetria não é vista significativamente nos dados de saída, que apesar de possuir a média mais centrada ao limite esquerdo do histograma, tende a possuir uma distribuição parcialmente normal.

Chong et al. (2024) ao analisarem os dados de entrada para realizar as regressões relataram que existem comportamentos diversos na distribuição dos histogramas dependendo da espécie analisada. A *Chlorella vulgaris* tende a normalidade, enquanto a *Spirulina platensis* e *Chlamydomonas reinhardtii* tende a se concentrar nos extremos. Esse comportamento também é observado Hajinajaf et al. (2022) onde os valores preditos para a fixação de CO_2 também apresentaram comportamento próximo a distribuição normal. Logo, é possível observar que os dados obtidos e analisados são da *Chlorella vulgaris* e que condizem com o esperado com a literatura.

A ausência de normalidade na distribuição dos dados pode ser um fator determinante para a discrepância entre os resultados de treino e teste, uma vez que muitos modelos de aprendizado de máquina, especialmente os lineares como Linear, *Ridge* e LASSO, assumem implicitamente que as variáveis seguem uma distribuição aproximadamente normal para maximizar sua eficiência. Quando essa premissa não é atendida, os modelos podem ajustar-se bem aos dados de treino, mas falhar em capturar padrões generalizáveis, levando a desempenhos inferiores no teste. Além disso, modelos mais complexos, como *Random Forest* e RNA, que são menos sensíveis à distribuição dos dados, podem superajustar *outliers* e ruídos no conjunto de treino, resultando em *overfitting* e, consequentemente, em uma queda acentuada no desempenho no teste.

Figura 10 - Histograma de TN, Histograma de Fósforo, Histograma de Nitrogênio, Histograma de TR, Histograma de IL, Histograma de DO, Histograma de Acúmulo de carboidratos, Histograma de produtividade da biomassa, Histograma de produtividade de carboidratos.



TN – tempo normalizado, TR – tempo de residência, DO – densidade óptica, Ad. – adimensional.

Fonte: Autor (2025)

Outra análise feita previamente antes da própria estimação através dos métodos de regressão, foi cálculo dos valores máximos e mínimos dos dados, assim como a média e desvio padrão. Diferentemente do que é esperado normalmente, não é importante que esses valores sejam curtos, ou estreitamente espaçados. Assim, como é possível observar na Tabela 6 existe uma margem ampla de dados que podem ser previstos utilizando os modelos de regressão. Vale ressaltar, no entanto que esses métodos tendem a não calcular muito bem dados que estão muito distantes da margem utilizada na regressão.

Tabela 6 – Análise dos dados de entrada.

Nome	Unidade	Máximo	Mínimo	média	Desvio padrão
Tempo normalizado	$adimensional$	1.00	0.00	0.65	0.36
Fósforo	$mg\ L^{-1}$	186.71	0.10	39.89	43.49
Nitrogênio	$mg\ L^{-1}$	644.35	0.00	84.07	142.11
Tempo de residência	$dias$	4.40	2.90	3.24	0.55
Intensidade Luminosa	$\mu E/m^2 s$	450.00	150.00	269.51	118.36
Densidade óptica	$adimensional$	20.16	0.58	10.47	4.29

A análise das variáveis mostra que algumas apresentam um espectro amplo, enquanto outras têm variações mais controladas. O fósforo e o nitrogênio, ambos medidos em mg L^{-1}, destacam-se pela alta amplitude: o fósforo varia de 0,10 a 186,71, enquanto o nitrogênio vai de 0,00 a 644,35. Esses valores máximos muito superiores às médias (39,89 para fósforo e 84,07 para nitrogênio) indicam uma grande dispersão, evidenciada também pelos altos desvios padrão (43,49 e 142,11, respectivamente). A intensidade luminosa, com intervalo de 150 a 450 µE m^{-2} s^{-1} e desvio padrão elevado (118,36), também apresenta um espectro amplo, sugerindo que diferentes condições experimentais ou ambientais influenciaram fortemente os dados. Já a densidade óptica, adimensional, varia de 0,58 a 20,16 e apresenta uma dispersão moderada (desvio padrão de 4,29), cobrindo uma faixa relativamente ampla dentro de sua escala.

Por outro lado, variáveis como o tempo normalizado e o tempo de residência mostram espectros mais limitados. O tempo normalizado, por definição, varia de 0 a 1, mas os dados estão concentrados principalmente em torno da média de 0,65, com desvio padrão de 0,36. Isso indica que a variação não é tão distribuída ao longo da escala completa. O tempo de residência, medido em dias, apresenta o intervalo mais estreito (2,90 a 4.40), com uma média de 3,24 e desvio padrão de 0,55; sugerindo

maior controle experimental nessa variável. Assim, enquanto algumas variáveis apresentam ampla variabilidade e cobertura de suas escalas, outras são bem mais restritas, provavelmente devido ao tipo de controle aplicado ou à natureza do experimento.

A Tabela 7 apresenta os melhores parâmetros encontrados para cada modelo avaliado. Para o modelo Linear, nenhum parâmetro foi ajustado, apenas o tempo de execução foi registrado.

Tabela 7 – Melhores parâmetros encontrados para cada modelo.

Modelo	Parâmetros	Melhor parâmetro
Linear	Tempo (s)	0.0039
Lasso	'alpha'	0.298364724
	'selection'	"random"
	'fit_intercept'	"False"
	'positive'	"False"
	'precompute'	"False"
	Tempo (s)	9.3982
Ridge	'alpha'	0.388292423
	'solver'	'sparse_cg'
	'fit_intercept'	"False"
	'positive'	"False"
	Tempo (s)	6053.0817
RNA	'hidden_layer_sizes'	"(13,)"
	'activation'	"relu"
	'solver'	"lbfgs"
	'alpha'	0.5
	'learning_rate'	"constant"
	Tempo (s)	73.8455
SVR	'kernel'	"rbf"
	'degree'	1
	'gamma'	"auto"
	'coef0'	0
	'C'	2
	'epsilon'	0.001
	'shrinking'	"False"
	Tempo (s)	25.0407
Random Forest	'n_estimators'	100
	'max_depth'	5
	'min_samples_split'	2
	'min_samples_leaf'	2
	'bootstrap'	"True"
	Tempo (s)	357.9394

Os resultados obtidos com as variações dos parâmetros indicam diferenças significativas no desempenho e no tempo de execução entre os modelos. O modelo Linear destacou-se pelo tempo extremamente baixo (0,0039 s), mas não oferece flexibilidade para capturar padrões mais complexos devido à sua simplicidade. Ele serve como referência inicial, mas é insuficiente para problemas que exigem modelagem mais elaborada. Price et al. (2023) relatou dificuldade semelhante. No entanto a quantidade de dados utilizada (30 amostras) em seu trabalho deve ter maior influência, pois, os resultados dele obtiveram R^2 em geral inferior a este.

O LASSO obteve o máximo desempenho ao ajustar o parâmetro 'alpha' para 0,298364724, com um tempo de execução de 9,3982 s. O teste da seleção "*Random*" pode ter contribuído para resultados mais robustos, reduzindo o risco de *overfitting*. Entretanto, o tempo foi consideravelmente maior do que o do modelo linear, evidenciando o custo adicional da regularização e da otimização em busca esparsada. Melkumova et al. (2017), Ching et al. (2022), Nguyen et al. (2023) e também encontraram valores para alpha inferiores a 1. O que é esperado ao atingir a convergência segundo Pedregosa et al. (2011).

No caso do Ridge, seu maior desempenho foi alcançado com 'alpha' ajustado em 0,388292423 e o solver *'sparse_cg'*, resultando no maior tempo de execução, 6053,0817 s. Este tempo elevado está provavelmente relacionado ao uso de um solver mais intensivo computacionalmente. Embora o *Ridge* seja uma extensão robusta da regressão linear, sua eficiência diminui para conjuntos de dados grandes ou ajustes muito finos. Esse resultado condiz com outros trabalhos que foram conduzidos Melkumova et al. (2017) e Ching et al. (2022) também encontraram valores para alpha inferiores a 1.

Para a RNA, a configuração ótima utilizou uma camada oculta de 13 neurônios e ativação 'relu', com solver 'lbfgs', gerando um tempo de 73,8455 s. Apesar do tempo ser maior do que os modelos lineares, a RNA apresentou boa flexibilidade para capturar padrões mais complexos com uma arquitetura simples. A escolha do solver 'lbfgs' garantiu um ajuste eficiente, mas aumentou o tempo de execução.

O SVR destacou-se pela utilização do kernel 'rbf', com 'C = 2' e 'epsilon = 0.001', atingindo um tempo de execução de 25,0407 s. Este modelo mostrou bom equilíbrio entre capacidade preditiva e tempo computacional. A escolha de '*epsilon*' pequeno indica alta sensibilidade a desvios, enquanto o kernel RBF lidou bem com não-linearidades.

Por fim, o *Random Forest* alcançou resultados satisfatórios com ‘n_estimators’ = 100' e 'max_depth = 5', registrando 357,9394 s. A limitação da profundidade das árvores favoreceu o controle da complexidade do modelo, garantindo robustez e interpretabilidade. O uso de *bootstrap* como padrão ajudou na generalização, mas o custo computacional foi relativamente alto em comparação aos outros modelos não lineares, como SVR.

A comparação entre os métodos de regressão revelou diferenças significativas no desempenho preditivo, especialmente entre as técnicas lineares e não lineares, como é possível observar na Tabela 8. A regressão linear multivariada, embora direta e computacionalmente eficiente, mostrou limitações na captura de relações complexas entre as variáveis preditoras e a concentração de substrato. Isso se deve ao fato de que ela assume uma relação linear que não reflete adequadamente a dinâmica não linear da biomassa da *Chlorella vulgaris*.

Tabela 8 – Resultado das regressões para a amostragem de Treino.

	Acúmulo de Carboidratos (%)		Produtividade da Biomassa (mg L^{-1} d^{-1})		Produtividade de Carboidratos (mg L^{-1} d^{-1})	
	R^2	RMSE	R^2	RMSE	R^2	RMSE
Linear	0.0147	10.5994	0.4785	0.1228	-0.0310	0.0721
Ridge	0.3197	8.8077	0.6486	0.1008	0.1927	0.0638
Lasso	0.3185	8.8154	0.6469	0.1010	0.1954	0.0637
RNA	0.8717	0.3582	0.8962	0.3222	0.7053	0.5429
SVR	0.7454	0.5046	0.7850	0.4637	0.6180	0.6311
Random Forest	0.8770	0.3507	0.9220	0.2792	0.9347	0.2556

As técnicas de regularização, como *Ridge* e Lasso, melhoraram a estabilidade da predição ao controlar o efeito de variáveis correlacionadas e ajustar modelos mais parcimoniosos. No entanto, a precisão desses métodos também foi limitada em comparação com técnicas mais robustas.

Por outro lado, métodos baseados em aprendizado de máquina, como redes neurais e *Random Forest*, apresentaram desempenho superior, com menor erro quadrático médio e valores de R^2 mais elevados, especialmente em dados complexos e de alta dimensionalidade. As redes neurais, por exemplo, mostraram grande capacidade de ajuste, mas com o risco de *overfitting*, ressaltando a importância de otimizar parâmetros e utilizar validação cruzada. O *Random Forest* destacou-se pela combinação de robustez e capacidade de generalização, o que o torna uma opção promissora para modelar sistemas biológicos complexos. Esses resultados indicam

como também Igou et al. (2023) e Sonachalam et al (2024) observaram, que o uso de técnicas não lineares é essencial para a predição em sistemas biológicos, especialmente onde há variáveis altamente correlacionadas e interações não lineares.

Nos resultados de regressão para os dados de teste contidos na tabela 9 a observa-se uma queda generalizada no desempenho dos modelos em comparação ao treino, o que é esperado, mas indica possíveis problemas de generalização para alguns casos. O *Random Forest*, embora tenha apresentado redução nos R^2, ainda manteve o melhor desempenho relativo, com R^2 acima de 0.39 em todas as métricas e erros mais baixos em relação a outros modelos. A RNA, por sua vez, sofreu uma redução mais acentuada na precisão, especialmente na produtividade de carboidratos, onde o RMSE subiu significativamente, sugerindo *overfitting*. Modelos como *Ridge* e LASSO apresentaram resultados mais consistentes entre treino e teste, mas seus desempenhos absolutos continuam inferiores aos modelos mais complexos, como o *Random Forest*.

Tabela 9 – Resultado das regressões para a amostragem de Teste.

	Acúmulo de Carboidratos (%)		Produtividade da Biomassa (mg L^{-1} d^{-1})		Produtividade de Carboidratos (mg L^{-1} d^{-1})	
	R^2	RMSE	R^2	RMSE	R^2	RMSE
Linear	0.5825	7.0761	0.7052	0.1032	0.4963	0.0583
Ridge	0.2633	9.4000	0.54892	0.1276	0.2194	0.0725
Lasso	0.2651	9.3887	0.5682	0.1248	0.3261	0.0673
RNA	0.3788	0.8083	0.698	0.6141	0.3412	0.9389
SVR	0.2610	0.8816	0.6583	0.6533	0.1702	1.0537
Random Forest	0.3951	0.7976	0.7782	0.5263	0.4053	0.8920

Os fatos evidenciados anteriormente ficam mais claros quando são calculados os índices através das equações 57 e 58 contidos na Tabela 10 a seguir. Onde é possível observar a relação de treino e teste de cada algoritmo para cada tipo de variável de saída estimada.

Tabela 10 – Relação entre os Resultados de treino e teste.

	Acúmulo de Carboidratos (%)		Produtividade da Biomassa (mg L^{-1} d^{-1})		Produtividade de Carboidratos (mg L^{-1} d^{-1})	
	R^2	RMSE	R^2	RMSE	R^2	RMSE
Linear	3862.585	-33.2406	47.3772	-15.9609	-1700.9677	-19.1401
Ridge	-17.6415	6.7248	-15.3685	26.5873	13.8557	13.6364
Lasso	-16.7661	6.5034	-12.1657	23.5644	66.8884	5.6515
RNA	-56.5447	125.6561	-22.1156	90.5959	-51.6234	-100.0000
SVR	-64.9852	74.7126	-16.1401	40.8885	-72.4595	48.7720
Random Forest	-54.9487	127.4309	-15.5965	88.5029	-56.6385	312.2457

O modelo linear apresenta os valores dos índices bem divergentes entre cada tipo de variável estimada bastante negativo, especialmente na Conversão em Carboidratos, onde o índice foi de: 3862.5850%; -33.2406%. Uma anomalia incomum, onde o treino teve pior desempenho que o teste. Essa discrepância pode estar associada a limitações inerentes do modelo linear em capturar relações não-lineares ou interações mais complexas presentes nos dados. A anomalia observada na conversão em carboidratos, sugere que o modelo linear falhou em encontrar um ajuste mínimo aos dados, potencialmente devido à presença de outliers, variáveis com alta multicolinearidade ou falta de transformação adequada das variáveis independentes.

O índice calculado para o *Ridge* em relação ao R^2 também é negativo na maior parte das métricas, indicando que o desempenho no conjunto de teste é pior do que no conjunto de treino. Por exemplo, na Produtividade da Biomassa, o índice para o RMSE foi de 26.5873%, o que sugere que o modelo, apesar de estar regularizado, não conseguiu capturar as relações entre as variáveis de forma eficaz. Isso pode ser um sinal de subajuste, onde o modelo não consegue modelar adequadamente os dados, mesmo com a regularização.

O Lasso tem índices negativos também, mas com um desempenho ligeiramente melhor do que o Ridge, especialmente na Produtividade de Carboidratos (índice de 5.6515%). Isso indica que, embora o modelo LASSO seja mais simples, ele consegue fazer uma melhor generalização do que o Ridge, mas ainda assim apresenta uma diferença considerável entre treino e teste, o que sugere que ajustes adicionais nos parâmetros ou na escolha das variáveis podem ser necessários. Tanto no modelo utilizado do *Ridge*, quanto no LASSO, houve uma anomalia onde a produtividade de carboidratos obteve o índice para o R^2 positivo.

O modelo de Redes Neurais apresenta índices elevados e negativos (por exemplo, -56.5447% de R^2 para Conversão em Carboidratos), o que indica uma grande diferença no desempenho entre treino e teste. Esse comportamento sugere que a arquitetura ou os parâmetros de treinamento do modelo precisam de ajustes significativos, pois o modelo está falhando tanto no treino quanto no teste, refletindo um possível sub-ajuste. Pois, Jerry et al. (2023) utilizaram mais de 100 neurônios na camada oculta e alcançaram $R^2 > 0,98$ que são considerados ótimos, mas ao observar o RMSE > 1 o que sugere *overfitting.* Logo, deveria ter usado mais neurônios na camada oculta para obter um desempenho melhor. Supriyanto et al. (2019) ao montar Redes Neurais constatou que ao retirar algumas variáveis de entrada, houve melhor desempenho das predições.

SVR (*Support Vector Regression*) também mostra índices negativos, com um valor muito alto de índice na Produtividade de Carboidratos (-72.4595%), o que sugere uma grande discrepância no desempenho entre treino e teste. Isso pode indicar que o modelo não está capturando as complexidades dos dados, resultando em uma performance insatisfatória nos dados de teste. Hossain et al. (2022) ao relacionar nitrogênio e fósforo em uma única variável de entrada diminui a correlação entre as variáveis de entrada. O que pode ser uma peça central para melhorar este desempenho.

O *Random Forest* também apresenta índices negativos em todas as métricas de R^2, com valores elevados, como -56.6385% na Produtividade de Carboidratos. Isso sugere que o modelo está tendo dificuldades em generalizar para o conjunto de teste. No entanto, em relação a produção de Biomassa obteve o melhor desempenho de R^2 de todos os ensaios. Primeiramente, a colinearidade entre variáveis de entrada, como a forte correlação entre nitrogênio e fósforo, pode ter impactado negativamente o modelo. A colinearidade de nitrogênio e fósforo pode influenciar nas previsões deste algoritmo. Embora a razão N/P tenha sido considerada para mitigar esse problema, sua influência nas predições ainda pode ser relevante. Além disso, desbalanceamento ou ruído nos dados pode ter afetado a modelagem da produtividade de carboidratos, especialmente se houver grande variabilidade ou a presença de outliers.

Ao analisar os resultados de Igou et al. (2023) que utilizou uma base de dados muito maior, os resultados dos índices foram bem semelhantes aos obtidos neste trabalho R^2 variando de (-49,14%, -20,41%) e o RMSE variando de (123%, 208%), enquanto que neste trabalho (sem considerar o modelo linear), para o índice

de R^2 variando de (-72, 4595%, 66,884%) e o índice do RMSE variando de (5,65%, 312,2467%). Salvo os casos extremos, a predição utilizando *Random Forest* conseguiu desempenhar até mais, mesmo ao possuir bem menos dados

O regime de operação tem impacto significativo na predição. A produção de carboidratos por *Chlorella vulgaris* em regime semi-contínuo tende a apresentar maior erro em comparação ao regime em batelada. Isso ocorre porque, no processo contínuo, há maior variabilidade nas condições operacionais, como taxa de diluição, disponibilidade de nutrientes e variações ambientais (como luz e CO_2). Pequenas flutuações nesses parâmetros podem impactar significativamente a composição bioquímica das células, levando a variações na produção de carboidratos como é possível observar nos trabalhos de Khoo, et al. (2016). Já no regime batelada, as condições do meio são mais estáveis ao longo do tempo, resultando em menor variabilidade na síntese de carboidratos e, consequentemente, menor erro na predição e controle da produção como foi relatado por He et al. (2016), Wang et al. (2019) e Figueroa-Torres et al. (2017).

Os gráficos de dispersão para os carboidratos na Figura 11 apresentados oferecem uma visão geral da performance de diferentes modelos de regressão em um conjunto de dados específico. Ao comparar a distribuição dos pontos em relação à linha de regressão (diagonal), podemos inferir algumas características sobre cada modelo.

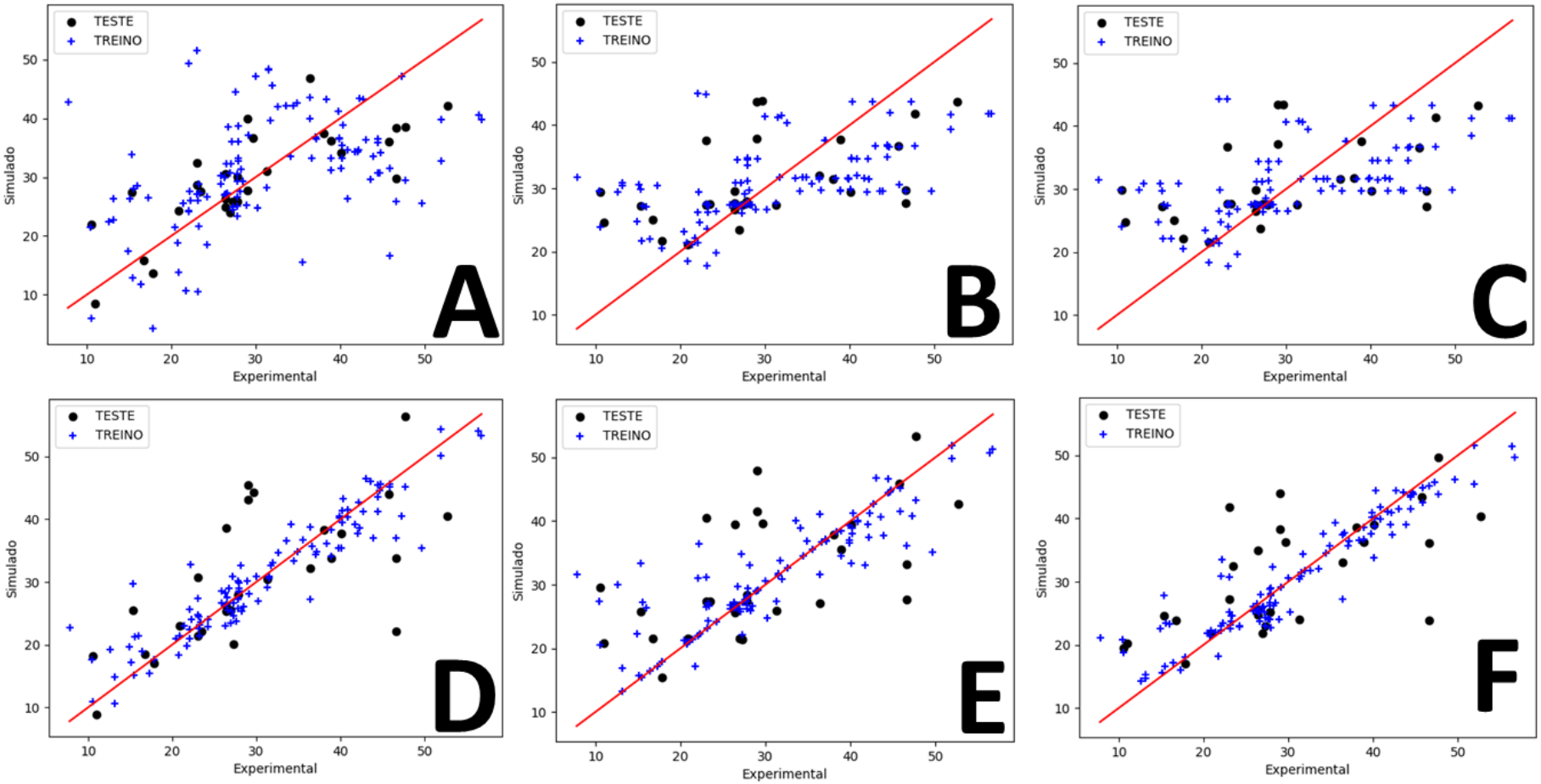
Os gráficos de A a C demonstram um ajuste ruim (Figura 11), com os pontos dispersos de forma aleatória ao redor da linha de regressão. Isso sugere que os modelos correspondentes, possivelmente regressão linear simples ou múltipla, estão capturando a relação entre as variáveis de forma insatisfatória, que é condizente aos valores de R^2 e RMSE obtidos. Já os gráficos de D a F (Figura 11), por sua vez, apresentam uma dispersão menor dos pontos em torno da linha, indicando um ajuste mais preciso. Esses modelos, Redes neurais e *Random Forest*, podem estar capturando relações mais complexas entre as variáveis.

Os gráficos de resíduos na

Figura 12 fornecem visualmente *insights* valiosos sobre a qualidade do ajuste dos modelos de regressão. Idealmente, os resíduos devem estar distribuídos aleatoriamente em torno da linha zero, indicando que o modelo captura a relação entre as variáveis de forma adequada. Ao analisar os gráficos apresentados, observamos que, em geral, os resíduos estão distribuídos de forma aleatória, sugerindo um bom

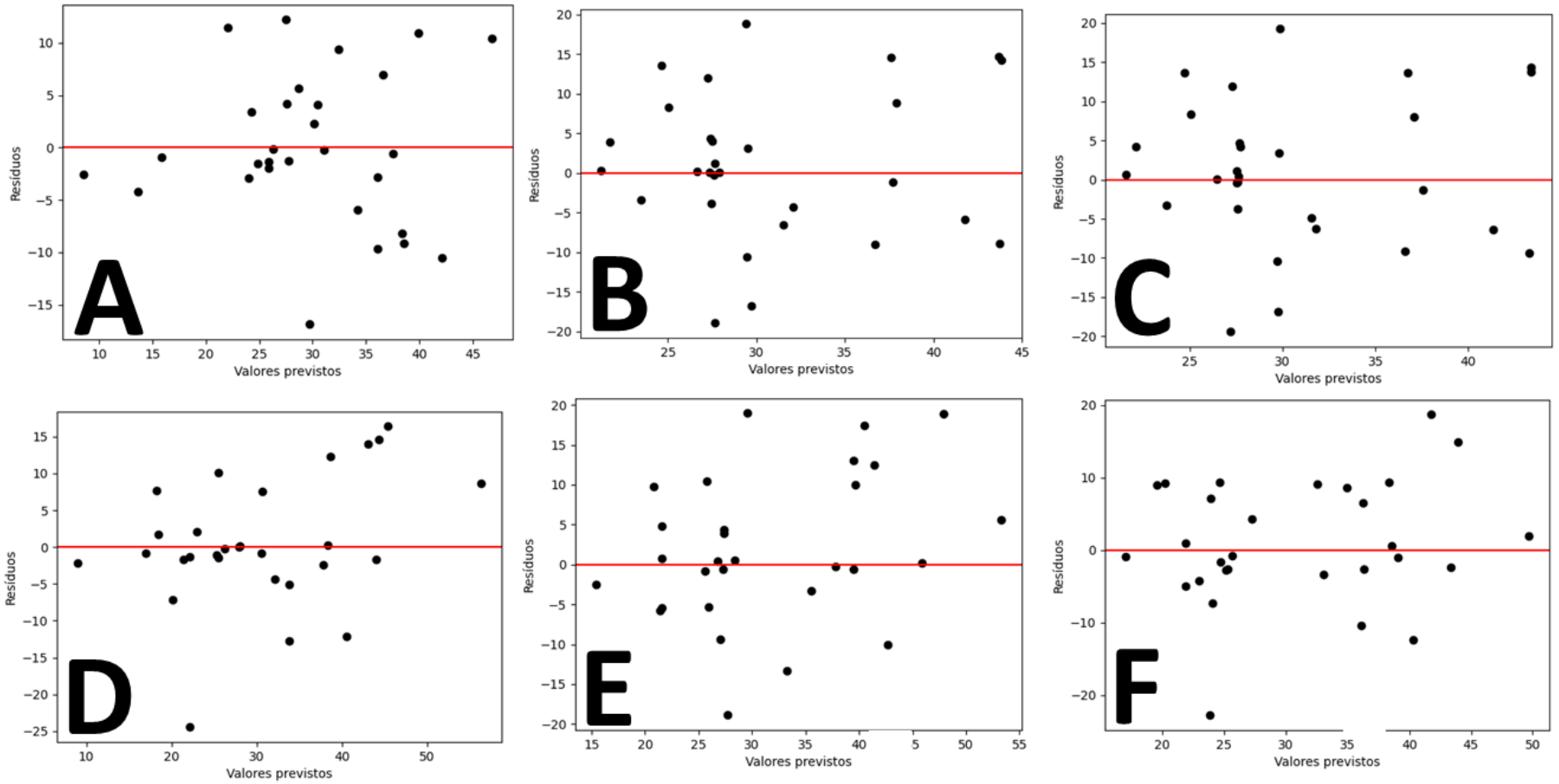
ajuste dos modelos. No entanto, alguns gráficos apresentam nuances que merecem atenção, como a possível presença de alguns *outliers* ou leves padrões nos resíduos, principalmente nas Regressões A, B e C. Especialmente na regressão B e C que são *Ridge* e LASSO respectivamente existe uma grande semelhança entre si nos resíduos.

Figura 11 - Regressão para a determinação do acúmulo de carboidratos.



A –Linear multivariada; B – Ridge; C – Lasso; D – RNA; E – SVR; F – Random Forest.

Figura 12 - Gráficos dos resíduos para o acúmulo de carboidratos.



A –Linear multivariada; B – Ridge; C – Lasso; D – RNA; E – SVR; F – Random Forest.

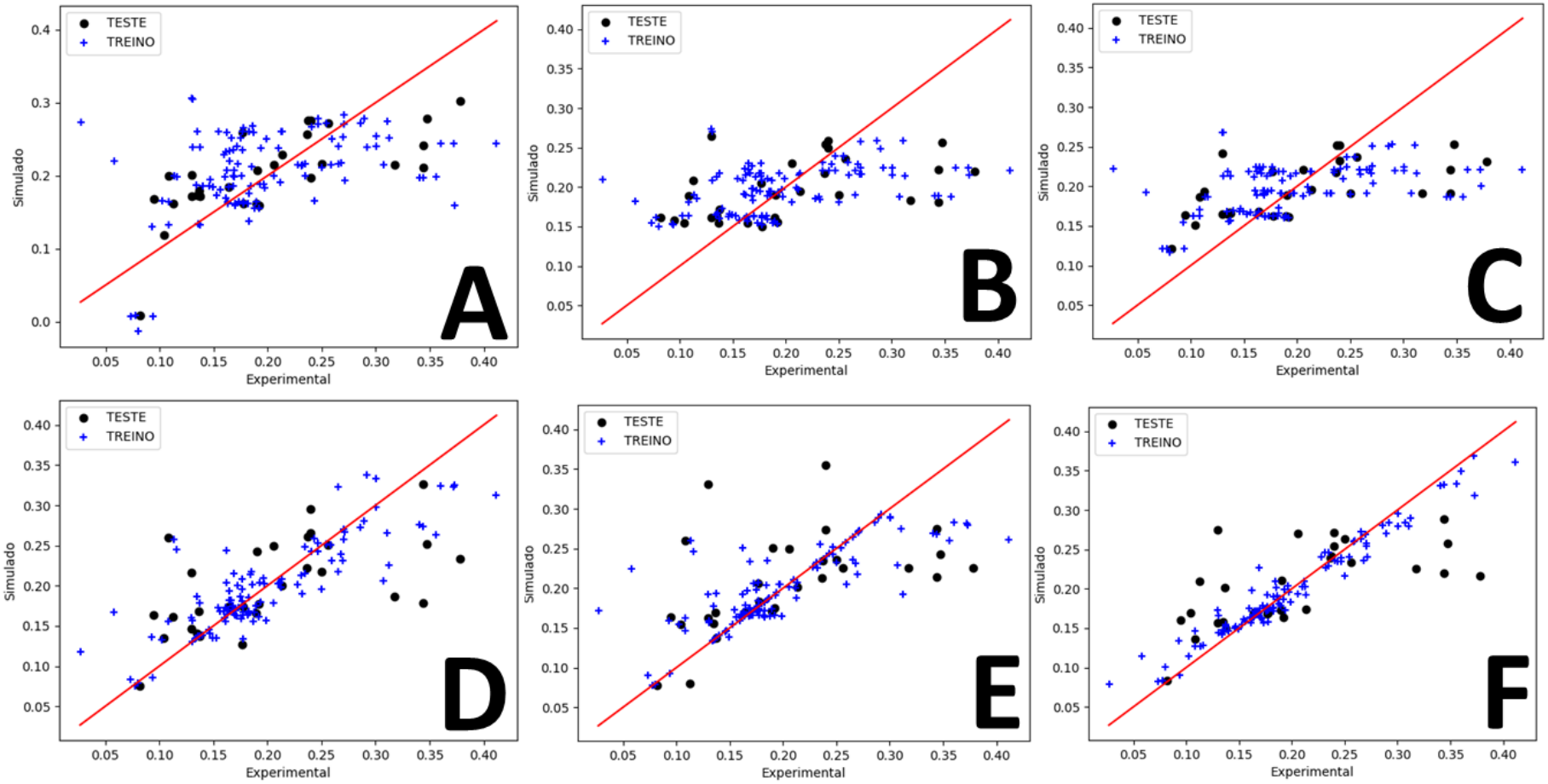
De modo semelhante aconteceu quando utilizada a produtividade de Carboidratos como a variável de saída (Figura 13). Os resultados sugerem que os modelos de aprendizado de máquina (RNA, SVR e *Random Forest*) superam os modelos lineares tradicionais (linear multivariada, *Ridge* e LASSO) na tarefa de previsão utilizando as variáveis de entrada deste trabalho. Esses modelos mais complexos conseguem capturar padrões mais complexos nos dados, resultando em previsões mais precisas.

Os gráficos de resíduos apresentados na Figura 14 mostram a relação entre os resíduos e os valores previstos para os seis modelos distintos (A, B, C, D, E, F). De maneira geral, os pontos estão distribuídos aleatoriamente ao redor da linha vermelha, indicando que os modelos não apresentam erros sistemáticos evidentes. Além disso, a ausência de padrões claros, como tendências em U ou inclinações, sugere que as previsões capturam bem a relação entre as variáveis. No entanto, alguns gráficos, como B e F, apresentam variações na dispersão dos resíduos, o que pode indicar a presença de heterocedasticidade, ou seja, uma mudança na variabilidade dos erros ao longo dos valores previstos.

Ao comparar os gráficos sobre os resíduos nas predições, não existem um consenso sobre o comportamento da *C. vulgaris* perante qualquer tipo de regressão. O efeito do meio de cultura também tem um grande impacto, Ma et al. (2023) relataram alta dispersão nos resíduos e alta dispersão ao utilizarem resíduos da suinocultura. No entanto, Lam et al. (2017) não obtiveram os mesmos resultados e o desempenho da regressão obteve menor margem, sofrendo com as margens viciadas. De modo que todos os modelos testados possuem o mesmo comportamento, o que não é observado neste trabalho, onde todos os resíduos são aleatórios e cada gráfico diverge entre si.

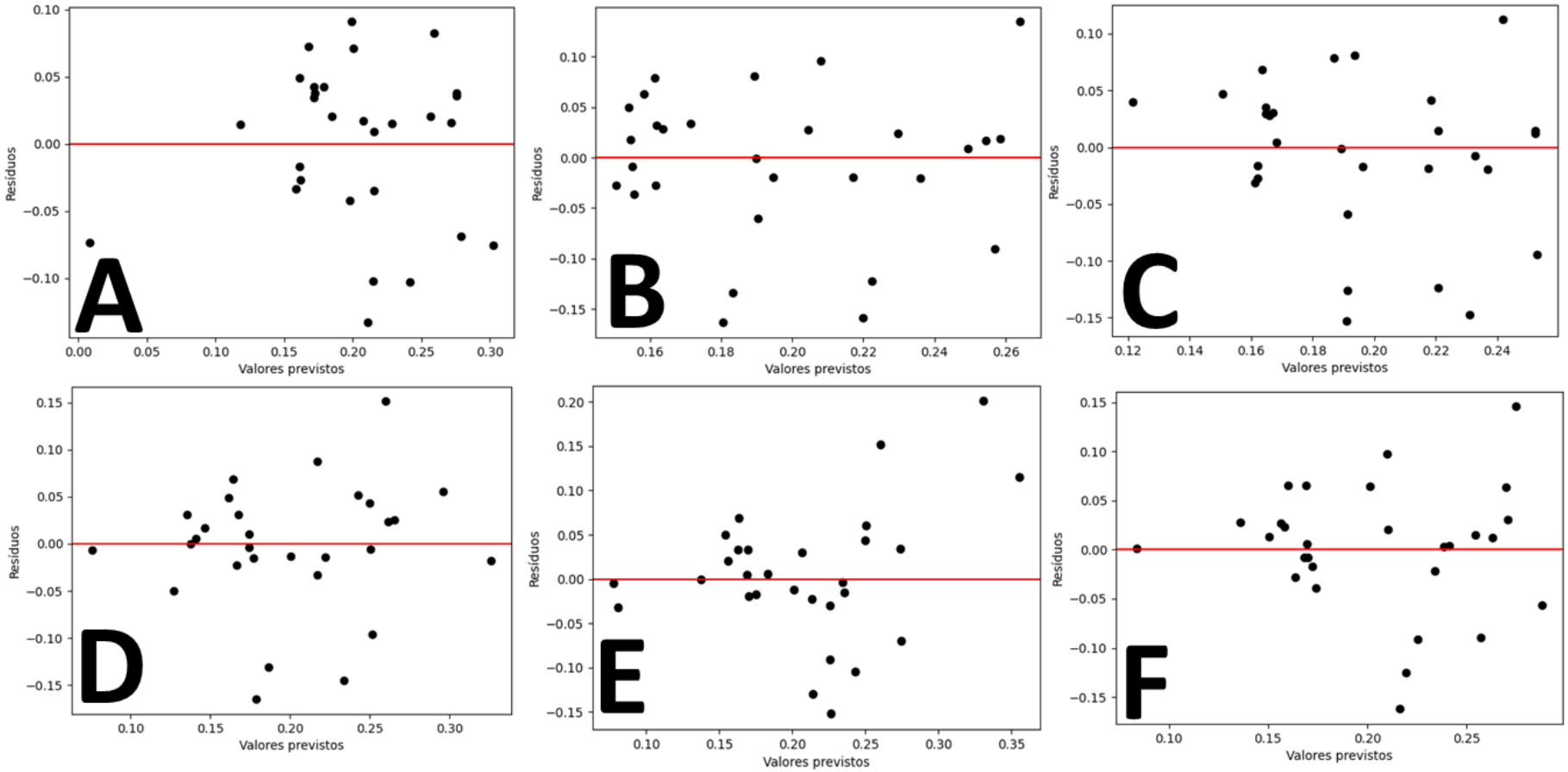
A divergência entre os estudos reforça a complexidade da modelagem preditiva da produtividade de carboidratos em *C. vulgaris*, evidenciando que a escolha do modelo ideal pode depender fortemente das condições experimentais e das variáveis de entrada utilizadas, como é evidenciado na Tabela 2. Além disso, a influência do meio de cultura sobre os resíduos e a variabilidade dos erros sugere que ajustes nos modelos podem ser necessários para diferentes cenários produtivos. Assim, futuras pesquisas podem explorar a combinação de técnicas de aprendizado de máquina com abordagens híbridas ou o uso de variáveis adicionais para aprimorar a robustez das previsões e reduzir incertezas associadas às variações ambientais e operacionais.

Figura 13 - Regressão para a determinação da produtividade de carboidratos.



A –Linear multivariada; B – Ridge; C – Lasso; D – RNA; E – SVR; F – Random Forest.

Figura 14 - Gráficos dos resíduos para a produtividade de carboidratos.



A –Linear multivariada; B – Ridge; C – Lasso; D – RNA; E – SVR; F – Random Forest.

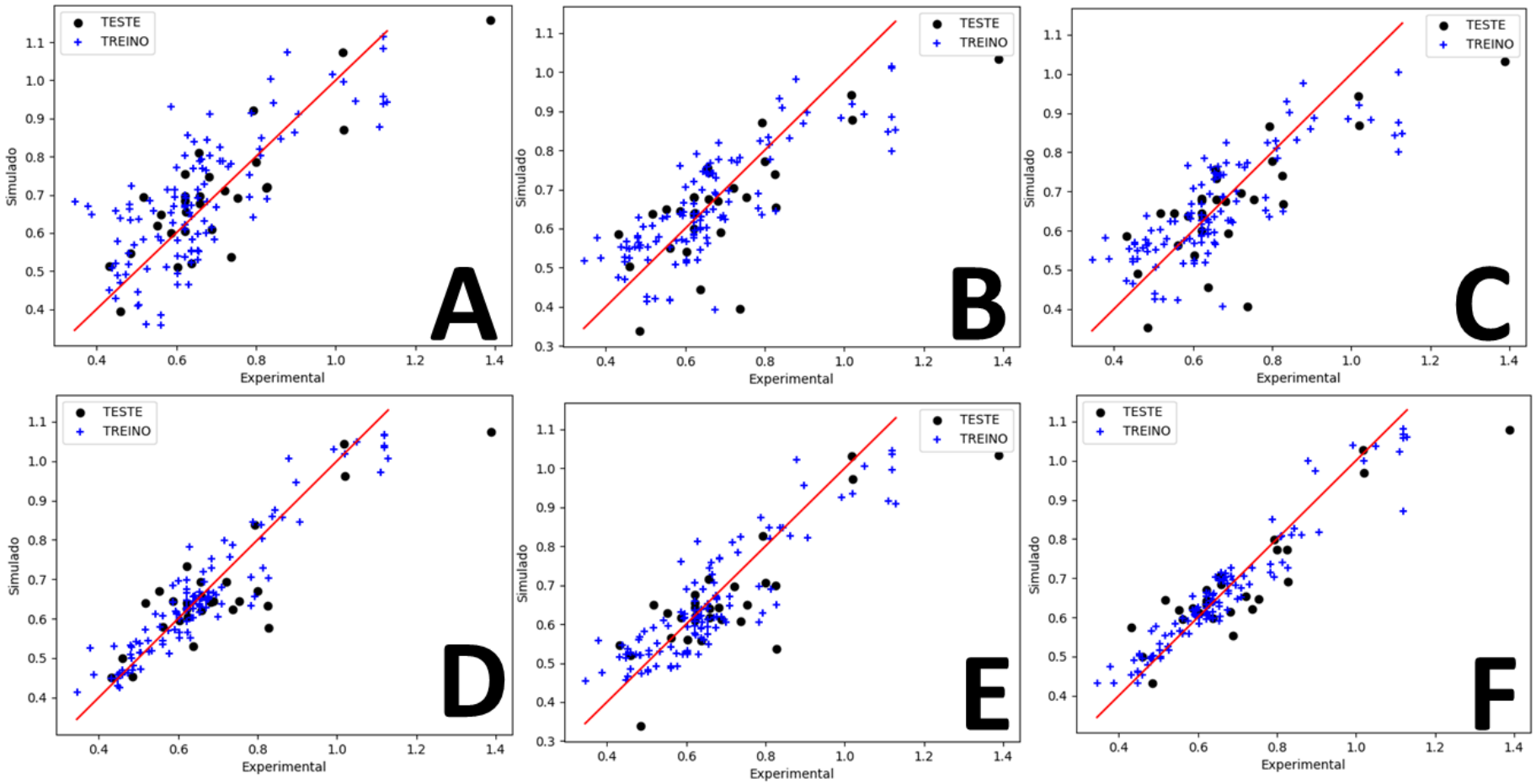
O melhor desempenho dos algoritmos se deu quando foi analisado a produtividade da biomassa como variável de saída, esse resultado que foi observado numericamente com o R^2 e RMSE também foi possível ser observado graficamente na Figura 15. O modelo linear teve o pior desempenho, onde se observa que a dispersão dos pontos está mais distante da linha de convergência. Ridge e Lasso tiveram desempenhos muito semelhantes e melhores ao algoritmo linear. Os algoritmos RNA e SVR tiveram desempenhos semelhantes entre si. E por fim o algoritmo Random Forest teve o melhor desempenho de todos os testes de saída.

Tanto nos resíduos com relação a fração de carboidratos e a produtividade de carboidratos existe um comportamento aleatório na distribuição dos pontos ao longo da linha, o mesmo comportamento ocorreu com Velásquez-Orta et al (2024) com uma escala maior (-80,+80) e 69 amostras. Indicando que este trabalho conseguiu melhorar a performance, principalmente por usar mais variáveis de entrada independentes e mais dados.

Os gráficos de resíduos para a produtividade de biomassa na figura 16 apresentaram um comportamento distinto das demais variáveis analisadas. Observou-se um melhor desempenho, com os pontos mais próximos da linha da idealidade. No entanto, a análise dos resíduos revelou a ausência da aleatoriedade esperada, evidenciada pelo acúmulo dos pontos predominantemente na região central do gráfico. Esse comportamento sugere a presença de um viés nas regressões realizadas, comprometendo a qualidade das estimativas obtidas. Assim, pode-se afirmar que a hipótese levantada anteriormente é válida: os dados utilizados nas regressões possuem uma abrangência limitada, dificultando o estabelecimento de uma relação precisa entre as variáveis de entrada e a produtividade de biomassa.

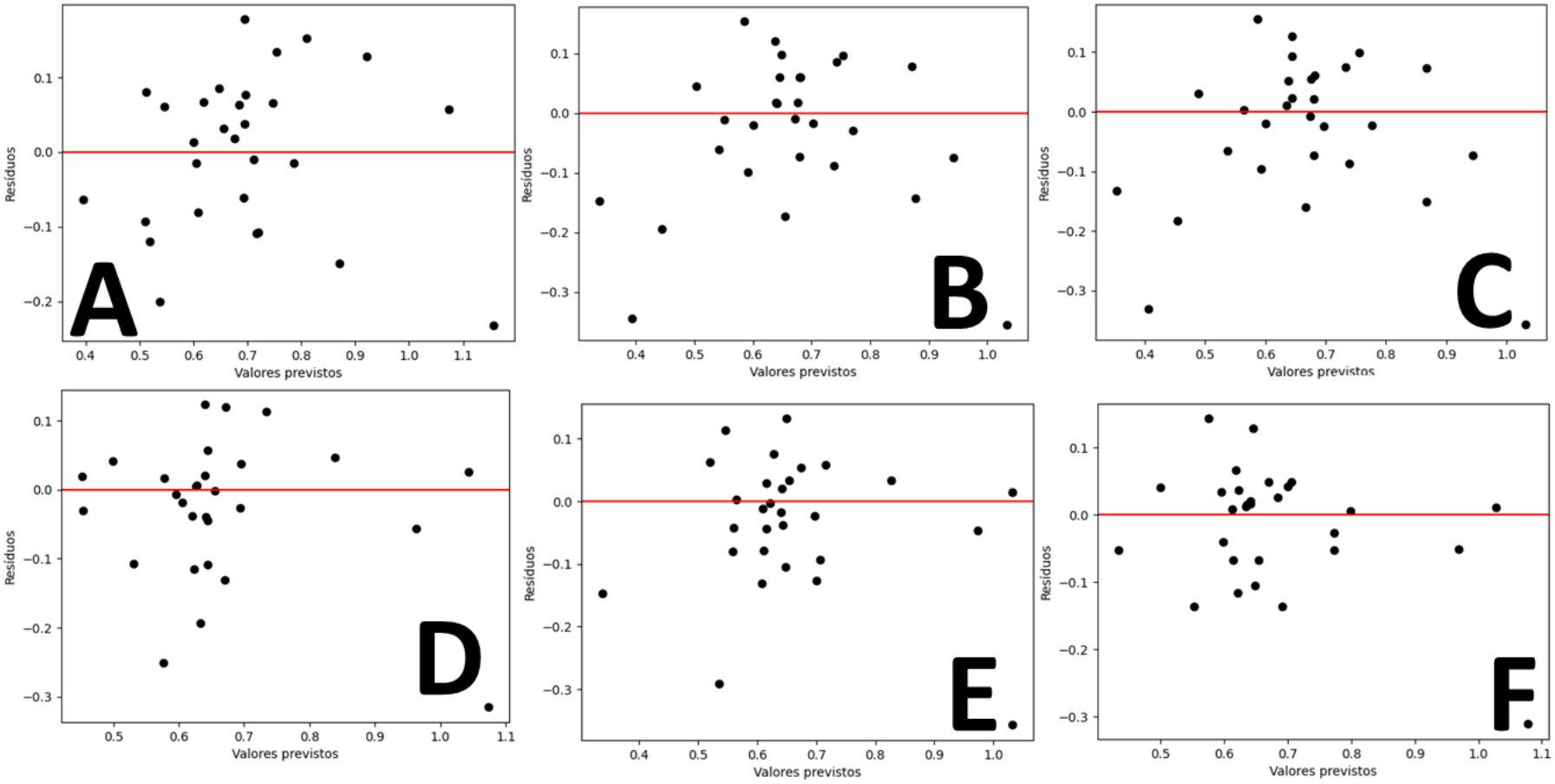
A comparação entre os algoritmos mostra que os métodos de aprendizado de máquina, como SVR e *Random Forest*, capturam melhor as relações não lineares entre as variáveis, evidenciando a limitação dos modelos lineares, que simplificam padrões complexos e podem perder informações relevantes. Ainda assim, a análise dos resíduos indica a presença de padrões nos erros, possivelmente relacionados à falta de variáveis explicativas ou à necessidade de um pré-processamento mais refinado. Além disso, a variação no desempenho entre os algoritmos sugere que a complexidade do modelo deve ser equilibrada com sua interpretabilidade e capacidade de generalização. O *Random Forest* teve o melhor desempenho, mas pode exigir um volume maior de dados para evitar sobreajuste.

Figura 15 - Regressão para a determinação da produtividade de biomassa.



A –Linear multivariada; B – Ridge; C – Lasso; D – RNA; E – SVR; F – Random Forest.

Figura 16 - Gráficos dos resíduos para a produtividade de biomassa.



A –Linear multivariada; B – Ridge; C – Lasso; D – RNA; E – SVR; F – Random Forest.

Desse modo é possível salientar que a questão em si não é a qualidade dos dados, e sim como eles estão distribuídos em seu domínio. Quanto maior o intervalo de abrangência dos dados de entrada para a variável de saída, existe a tendência de que a regressão desempenhe melhor. Isso não anula o fato de que este trabalho foi capaz de determinar as concentrações de carboidratos além da produtividade de biomassa e carboidratos. Além de que essa metodologia desenvolvida nesse trabalho é capaz de averiguar a qualidade dos dados antes de fazer a regressão. Sendo assim, foi previsto que a regressão não poderia desempenhar devido a questão da proporcionalidade e da quantidade de dados utilizados para treino em detrimento dos dados separados para o teste.

Durante a avaliação do modelo, foi identificada uma discrepância significativa entre os resultados obtidos no teste e no treino, com valores consideravelmente inferiores no desempenho do modelo nos dados de teste. Essa diferença sugere uma possível falha relacionada à representatividade dos dados utilizados no treinamento, que correspondem a 80% do total disponível. Ao que tudo indica, os dados de treino não foram capazes de abranger de forma satisfatória a variabilidade e as características presentes nos 20% dos dados reservados para teste, comprometendo a generalização do modelo. Pozzobon et al. (2021) utilizou a mesma proporção de dados, utilizando 261 dados e conseguiu estimar bem os parâmetros de saída com 10% de erro. Ressaltando desta forma a importância de um conjunto maior de dados para fazer as previsões. Já Pääkkönen et al. (2024) utilizaram apenas 50 ensaios e conseguiu R^2 >0.9. Em ambos os casos são sistemas que não variam com o tempo e tendem a linearidade. O que favoreceu o desempenho com uma quantidade menor de dados.

Essa limitação também pode ter ocorrido devido a uma distribuição não homogênea dos dados, resultando em um conjunto de treino que não reflete adequadamente os padrões e outliers encontrados no conjunto de teste. Como consequência, o modelo apresentou um desempenho otimizado apenas para o subconjunto utilizado no treinamento, mas mostrou dificuldade em extrapolar para novos dados. Essa situação reforça a importância de realizar uma análise criteriosa prévia de amostragem, garantindo que tanto os dados de treino quanto os de teste representem de maneira equilibrada o domínio completo das variáveis em estudo.

6. CONSIDERAÇÕES FINAIS

Neste estudo, foi aplicada a técnica de *Machine Learning* para prever a produção de carboidratos pela microalga *Chlorella vulgaris* em cultivo contínuo, utilizando variáveis nutricionais, ambientais e operacionais como entradas para os modelos preditivos. A análise do coeficiente de Pearson permitiu identificar correlações significativas entre as variáveis independentes e dependentes, destacando a influência de fatores como a concentração de nutrientes e a intensidade luminosa na produção de carboidratos. Foi observada alta correlação entre nitrogênio e fósforo, utilizar essas duas variáveis de entrada pode prejudicar regressões por conta da colinearidade e alternativamente a razão de N/P foi utilizada para contornar este problema. O histograma das variáveis demonstrou a distribuição dos dados, evidenciando padrões e tendências que podem afetar as predições.

Diferentes técnicas de regressão foram implementadas e comparadas, abrangendo modelos lineares (Regressão Linear Multivariada, Ridge e LASSO) e não lineares (*Random Forest*, Redes Neurais e *Support Vector Regression - SVR*). Observou-se que os modelos não lineares apresentaram melhor desempenho na previsão de todas as variáveis de saída, especialmente o *Random Forest* e as Redes Neurais, devido à sua capacidade de capturar relações complexas entre as variáveis, e obtiveram o melhor R^2 de 0,9347; 0,8962 e RMSE 0,2556; 0,3222, na previsão da produtividade carboidratos e produtividade de biomassa, respectivamente.

A otimização dos modelos foi realizada através da definição dos melhores hiperparâmetros para cada técnica, utilizando a busca em grade, permitindo uma melhoria significativa na acurácia das previsões. Além disso, a validação cruzada foi empregada para evitar *overfitting* e garantir a generalização dos resultados, tornando os modelos mais robustos e confiáveis. Os resultados obtidos destacam o potencial do *machine learning* na predição e utilização futura para sistemas de controles para processos biotecnológicos.

7. REFERÊNCIAS

ABDEL-LATIF, H. M. R. et al. The effectiveness of *Arthrospira platensis* and microalgae in relieving stressful conditions affecting finfish and shellfish species: An overview. **Aquaculture Reports**, v. 24, p. 101135, 1 jun. 2022.

AHMAD SOBRI, M. Z. et al. A Review Unveiling Various Machine Learning Algorithms Adopted for Biohydrogen Productions from Microalgae. **Fermentation**, v. 9, n. 3, p. 243, mar. 2023.

AMBAT, Indu; TANG, Walter Z.; SILLANPÄÄ, Mika. Statistical analysis of sustainable production of algal biomass from wastewater treatment process. **Biomass and Bioenergy**, v. 120, p. 471-478, 2019. Disponível em: https://doi.org/10.1016/j.biombioe.2018.12.003.

APHA. (American Public Health Association). 1992. Greenberg, A. E.; Clesceri, L. S.; Eaton, A. D. **Standards methods for the examination of water and wastewater**. 19.a ed. AWWA, WES. Baltimore, Maryland.

ARORA, Y.; SHARMA, S.; SHARMA, V. Microalgae in Bioplastic Production: A Comprehensive Review. **Arabian Journal for Science and Engineering**, v. 48, n. 6, p. 7225–7241, 1 jun. 2023.

ARSHAD, U. et al. SVM, ANN, and PSF modelling approaches for prediction of iron dust minimum ignition temperature (MIT) based on the synergistic effect of dispersion pressure and concentration. **Process Safety and Environmental Protection**, v. 152, p. 375–390, ago. 2021.

ATTAR, H. et al. A modified grid search-based optimization for possibly repetitive global extremum with an application to edge intelligence in IIoT towards time-domain signals. **Wireless Networks**, v. 30, n. 8, p. 7015–7027, 1 nov. 2024.

AWAD, M.; KHANNA, R. Support Vector Regression. Em: AWAD, M.; KHANNA, R. (Eds.). **Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers**. Berkeley, CA: Apress, 2015. p. 67–80.

BARGHCHI, H. et al. The effects of Chlorella vulgaris on cardiovascular risk factors: A comprehensive review on putative molecular mechanisms. **Biomedicine & Pharmacotherapy**, v. 162, p. 114624, 1 jun. 2023.

BITO, T. et al. Potential of Chlorella as a Dietary Supplement to Promote Human Health. **Nutrients**, v. 12, n. 9, p. 2524, 20 ago. 2020.

BRADFORD, Eric et al**.** Dynamic modeling and optimization of sustainable algal production with uncertainty using multivariate Gaussian processes. ***Computers & Chemical Engineering*****,** v. 118, p. 143-158, 2018. Disponível em: https://doi.org/10.1016/j.compchemeng.2018.07.015.

BREIMAN, L. Random Forests. **Machine Learning**, v. 45, n. 1, p. 5–32, 1 out. 2001.

CHEIN, Flávia. *Introdução aos modelos de regressão linear: um passo inicial para compreensão da econometria como uma ferramenta de avaliação de políticas públicas*. Brasília: Enap, 2019. 76 p. ISBN 978-85-256-0115-5.

CHEN, J. et al. The Estimation of the Higher Heating Value of Biochar by Data-Driven Modeling. **Journal of Renewable Materials**, v. 10, n. 6, p. 1555–1574, 2022.

CHING, P. M. L. et al. Early prediction of Spirulina platensis biomass yield for biofuel production using machine learning. **Clean Technologies and Environmental Policy**, v. 24, n. 7, p. 2283–2293, 1 set. 2022.

CHOKSHI, K. et al. Nitrogen starvation-induced cellular crosstalk of ROS-scavenging antioxidants and phytohormone enhanced the biofuel potential of green microalga Acutodesmus dimorphus. **Biotechnology for Biofuels**, v. 10, n. 1, p. 60, 9 mar. 2017.

CHONG, J. W. R. et al. Artificial intelligence-driven microalgae autotrophic batch cultivation: A comparative study of machine and deep learning-based image classification models. **Algal Research**, v. 79, p. 103400, 1 abr. 2024.

CHU, R. et al. Recycling spent water from microalgae harvesting by fungal pellets to re-cultivate *Chlorella vulgaris* under different nutrient loads for biodiesel production. **Bioresource Technology**, v. 344, p. 126227, 1 jan. 2022.

COMITO, C.; PIZZUTI, C. Artificial intelligence for forecasting and diagnosing COVID-19 pandemic: A focused review. **Artificial Intelligence in Medicine**, v. 128, p. 102286, jun. 2022.

CORONADO-REYES, J. A. et al. Chlorella vulgaris, a microalgae important to be used in Biotechnology: a review. **Food Science and Technology**, v. 42, p. e37320, 2022.

COŞGUN, Ahmet et al**.** Exploring the critical factors of algal biomass and lipid production for renewable fuel production by machine learning**. Renewable Energy**, v. 163, p. 1299-1317, 2021. Disponível em: https://doi.org/10.1016/j.renene.2020.09.100.

DAHIYA, N.; GUPTA, S.; SINGH, S. A Review Paper on Machine Learning Applications, Advantages, and Techniques. **ECS Transactions**, v. 107, n. 1, p. 6137–6150, 24 abr. 2022.

DAVID, A. N. et al. Harnessing Artificial Neural Networks and large language models for bioprocess optimization: Predicting sugar output from Kraft waste-based lignocellulosic pretreatments. **Industrial Crops and Products**, v. 206, p. 117686, 15 dez. 2023.

DE FARIAS SILVA, C. E.; SFORZA, E. Carbohydrate productivity in continuous reactor under nitrogen limitation: Effect of light and residence time on nutrient uptake in Chlorella vulgaris. **Process Biochemistry**, v. 51, n. 12, p. 2112–2118, dez. 2016.

DE FARIAS SILVA, C. E; BERTUCCO, A. Bioethanol from Microalgal Biomass: A Promising Approach in Biorefinery. **Brazilian Archives of Biology and Technology**, v. 62, p. e19160816, 1 ago. 2019.

DEL RIO-CHANONA, Ehecatl Antonio et al**.** Review of advanced physical and data-driven models for dynamic bioprocess simulation: Case study of algae–bacteria consortium wastewater treatment. **Biotechnology and Bioengineering**, v. 116, n. 2, p. 342-353, 2019. Disponível em: https://doi.org/10.1002/bit.26845.

EL-FAYOUMY, E. A. et al. Co-production of high-density biomass and high-value compounds via two-stage cultivation of Chlorella vulgaris using light intensity and a

combination of salt stressors. **Biomass Conversion and Biorefinery**, v. 14, n. 18, p. 22673–22686, 1 set. 2024.

EL-GAWAD, A. F. A.; ZAKI, S.; KAMAL, E. A Survey on Machine Learning Techniques for Supply Chain Management. **American Journal of Business and Operations Research**, n. Issue 1, p. 24–38, 1 jan. 2021.

ELMALKY, A. M.; ARAJI, M. T. Neural Networks for Monitoring Microalgae Biomass in Building Façades. **Technology|Architecture + Design**, v. 8, n. 1, p. 60–69, 2 jan. 2024.

ELYASHBERG, M. E.; WILLIAMS, A. J.; MARTIN, G. E. Computer-assisted structure verification and elucidation tools in NMR-based structure elucidation. **Progress in Nuclear Magnetic Resonance Spectroscopy**, v. 53, n. 1, p. 1–104, 8 jul. 2008.

FARUQUE, M. O. et al. A comprehensive review on microalgae-driven heavy metals removal from industrial wastewater using living and nonliving microalgae. **Journal of Hazardous Materials Advances**, v. 16, p. 100492, 1 nov. 2024.

FERREIRA, G. F. et al. Effects of cultivation conditions on *Chlorella vulgaris* and *Desmodesmus* sp. grown in sugarcane agro-industry residues. **Bioresource Technology**, v. 342, p. 125949, 1 dez. 2021.

FIGUEIREDO, D. B. F.; SILVA, J. A. J.; ROCHA, E. C. What is R2 all about? **Leviathan (São Paulo)**, n. 3, p. 60, 16 nov. 2011.

FIGUEROA-TORRES, G. M.; PITTMAN, J. K.; THEODOROPOULOS, C. Kinetic modelling of starch and lipid formation during mixotrophic, nutrient-limited microalgal growth. *Bioresource Technology*, v. 241, p. 868–878, 2017. DOI: 10.1016/j.biortech.2017.05.177.

FISHER, O. J. et al. Data-driven modelling for resource recovery: Data volume, variability, and visualisation for an industrial bioprocess. **Biochemical Engineering Journal**, v. 185, p. 108499, 1 jul. 2022.

FU, L. et al. Hormesis effects of phosphorus on the viability of Chlorella regularis cells under nitrogen limitation. **Biotechnology for Biofuels**, v. 12, n. 1, p. 121, 13 maio 2019.

GARCÍA-CAMACHO, F. et al. Artificial neural network modeling for predicting the growth of the microalga *Karlodinium veneficum*. **Algal Research**, v. 14, p. 58-64, 2016. Disponível em: https://doi.org/10.1016/j.algal.2015.12.012.

GÉRON, A. **Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems**. [s.l.] O'Reilly Media, Incorporated, 2019.

GHATE, V.; HEMALATHA C, S. A comprehensive comparison of machine learning approaches with hyper-parameter tuning for smartphone sensor-based human activity recognition. **Measurement: Sensors**, v. 30, p. 100925, 1 dez. 2023.

GOJKOVIC, Z. et al. Modeling biomass production during progressive nitrogen starvation by North Swedish green microalgae. *Algal Research*, v. 47, p. 101835, 2020. DOI: 10.1016/j.algal.2020.101835.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep Learning**. [s.l.] MIT Press, 2016.

GRUBER, M. **Improving Efficiency by Shrinkage: The James--Stein and Ridge Regression Estimators**. [s.l.] CRC Press, p. 8–10 1998.

HAJIHOSSEINLOU, M.; MAGHSOUDI, A.; GHEZELBASH, R. Intelligent mapping of geochemical anomalies: Adaptation of DBSCAN and mean-shift clustering approaches. **Journal of Geochemical Exploration**, v. 258, p. 107393, 1 mar. 2024.

HAJINAJAF, N. et al. Integrated CO2 Capture and Nutrient Removal by Microalgae Chlorella vulgaris and Optimization Using Neural Network and Support Vector Regression. **Waste and Biomass Valorization**, v. 13, n. 12, p. 4749–4770, 1 dez. 2022.

HASTIE, Trevor; TIBSHIRANI, Robert; FRIEDMAN, Jerome. ***The elements of statistical learning: data mining, inference, and prediction***. 2. ed. New York: Springer, 2009.

HAYKIN, S. S. **Neural Networks and Learning Machines**. [s.l.] Pearson, 2009.

HAYKIN, Simon. *Princípios e práticas*. Porto Alegre: Bookman, 2001.

HE, Y. et al. Analysis and model delineation of marine microalgae growth and lipid accumulation in flat-plate photobioreactor. *Biochemical Engineering Journal*, v. 111, p. 108–116, 2016. DOI: 10.1016/j.bej.2016.03.014.

HOERL, A. E.; KENNARD, R. W. Ridge Regression: Biased Estimation for Nonorthogonal Problems. **Technometrics**, v. 12, n. 1, p. 55–67, fev. 1970.

HOERL, R. W. Ridge Regression: A Historical Context. **Technometrics**, v. 62, n. 4, p. 420–425, 1 out. 2020.

HONÓRIO, G. et al. Microalgas e a produção de energia de baixo carbono: uma alternativa sustentável aos combustíveis fósseis. **Latin American Journal of Energy Research**, v. 11, n. 2, p. 176–191, 26 dez. 2024.

HOSSAIN, S. M. Z. et al. Hybrid support vector regression and crow search algorithm for modeling and multiobjective optimization of microalgae-based wastewater treatment. **Journal of Environmental Management**, v. 301, p. 113783, 1 jan. 2022.

HUANG, Y. et al. Insight into nitrogen and phosphorus coupling effects on mixotrophic *Chlorella vulgaris* growth under stably controlled nutrient conditions. **Science of The Total Environment**, v. 752, p. 141747, 15 jan. 2021.

IGOU, T. et al. Real-Time Sensor Data Profile-Based Deep Learning Method Applied to Open Raceway Pond Microalgal Productivity Prediction. **Environmental Science & Technology**, v. 57, n. 46, p. 17981–17989, 21 nov. 2023.

IKOTUN, A. M. et al. K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data. **Information Sciences**, v. 622, p. 178–210, 1 abr. 2023.

JAVED, S. et al. Limited Phosphorous Supply Improved Lipid Content of Chlorella vulgaris That Increased Phenol and 2-Chlorophenol Adsorption from Contaminated Water with Acid Treatment. **Processes**, v. 10, n. 11, p. 2435, nov. 2022.

JERY, A. E. et al. A novel experimental and machine learning model to remove COD in a batch reactor equipped with microalgae. **Applied Water Science**, v. 13, n. 7, p. 153, jul. 2023.

LAM, M. K. et al. Cultivation of *Chlorella vulgaris* using nutrients source from domestic wastewater for biodiesel production: Growth condition and kinetic studies. **Renewable Energy**, v. 103, p. 197–207, 1 abr. 2017.

KAPLAN, E. et al. Assessment of different carbon and salinity levels on growth kinetics, lipid, and starch composition of *Chlorella vulgaris* SAG 211-12. *International Journal of Green Energy*, v. 17, p. 290–300, 2020. DOI: 10.1080/15435075.2020.1727480.

KENGE, R. Machine Learning, Its Limitations, and Solutions Over IT. **International Journal of Applied Research on Information Technology and Computing**, v. 11, n. 2, p. 73, 2020.

KHOO, C. G.; LAM, M. K.; LEE, K. T. Pilot-scale semi-continuous cultivation of microalgae *Chlorella vulgaris* in bubble column photobioreactor (BC-PBR): Hydrodynamics and gas–liquid mass transfer study. **Algal Research**, v. 15, p. 65–76, 1 abr. 2016.

KIRAGA, S. et al. Reference Evapotranspiration Estimation Using Genetic Algorithm-Optimized Machine Learning Models and Standardized Penman–Monteith Equation in a Highly Advective Environment. **Water**, v. 16, n. 1, p. 12, jan. 2024.

KUHN, M.; JOHNSON, K. **Applied Predictive Modeling**. New York, NY: Springer, p. 95-97, 2013.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. **Nature**, v. 521, n. 7553, p. 436–444, maio 2015.

LI, K. et al. Microalgae-based wastewater treatment for nutrients recovery: A review. **Bioresource Technology**, v. 291, p. 121934, 1 nov. 2019.

LIU, Jing-Yan et al**.** Rapid in situ measurements of algal cell concentrations using an artificial neural network and single-excitation fluorescence spectrometry. **Algal Research**, v. 45, p. 101739, 2020. Disponível em: https://doi.org/10.1016/j.algal.2019.101739.

LIYANAARACHCHI, Vinoj Chamilka et al. Artificial neural network (ANN) approach to optimize cultivation conditions of microalga *Chlorella vulgaris* in view of biodiesel production**. Biochemical Engineering Journal**, v. 168, p. 107951, 2021. Disponível em: https://doi.org/10.1016/j.bej.2021.107951.

LOPEZ-EXPOSITO, P.; NEGRO, C.; BLANCO, A. Direct estimation of microalgal flocs fractal dimension through laser reflectance and machine learning. **Algal Research**, v. 37, p. 240–247, 1 jan. 2019.

LOPEZ-EXPOSITO, Patricio; NEGRO, Carlos; BLANCO, Angeles. Direct estimation of microalgal flocs fractal dimension through laser reflectance and machine learning. **Algal Research**, v. 37, p. 240-247, 2019. Disponível em: https://doi.org/10.1016/j.algal.2018.12.007.

MA, X.; JIAN, W. Growth Conditions and Growth Kinetics of Chlorella Vulgaris Cultured in Domestic Sewage. **Sustainability**, v. 15, n. 3, p. 2162, jan. 2023.

MARINO, V. F. **Análises dos parâmetros de cultivo da microalga *Chlorella vulgaris***. Mestrado em Desenvolvimento de Produtos e Processos—Lorena: Universidade de São Paulo, 4 dez. 2018.

MCKINNEY, Wes. *Pandas: a library for data analysis and manipulation*. 2010. Disponível em: https://pandas.pydata.org/. Acesso em: 16 fev. 2025.

MEHRABI, N. et al. A Survey on Bias and Fairness in Machine Learning. **ACM Comput. Surv.**, v. 54, n. 6, p. 115:1-115:35, 13 jul. 2021.

MELKUMOVA, L. E.; SHATSKIKH, S. YA. Comparing Ridge and LASSO estimators for data analysis. **Procedia Engineering**, 3rd International Conference “Information Technology and Nanotechnology", ITNT-2017, 25-27 April 2017, Samara, Russia. v. 201, p. 746–755, 1 jan. 2017.

MEYERS, R. A. (Ed.). **Encyclopedia of Physical Science and Technology (Third Edition)**. New York: Academic Press, 2003. p. 55–73.

MOHAMED, Mohd Shamzi et al. Comparative analyses of response surface methodology and artificial neural network on medium optimization for *Tetraselmis* sp. FTC209 grown under mixotrophic condition. **The Scientific World Journal**, v. 2013, p. 1-10, 2013. Disponível em: https://doi.org/10.1155/2013/948940.

MONDAL, P. P. et al. Review on machine learning-based bioprocess optimization, monitoring, and control systems. **Bioresource Technology**, v. 370, p. 128523, 1 fev. 2023.

MOUNTOURAKIS, F. et al. Evidence of physiological adaptation of *Chlorella vulgaris* under extreme salinity – new insights into a potential halotolerance strategy. **Environmental and Experimental Botany**, v. 216, p. 105543, 1 dez. 2023.

MURWANASHYAKA, T. et al. Kinetic modelling of heterotrophic microalgae culture in wastewater: Storage molecule generation and pollutants mitigation. *Biochemical Engineering Journal*, v. 157, p. 107523, 2020. DOI: 10.1016/j.bej.2020.107523.

NAYERI, Z. M.; GHAFARIAN, T.; JAVADI, B. Application placement in Fog computing with AI approach: Taxonomy and a state of the art survey. **Journal of Network and Computer Applications**, v. 185, p. 103078, 1 jul. 2021.

NGUYEN, L. et al. Analysis of Microalgal Density Estimation by Using LASSO and Image Texture Features. **Sensors**, v. 23, n. 5, p. 2543, 24 fev. 2023.

NING, H.; LI, R.; ZHOU, T. Machine learning for microalgae detection and utilization. **Frontiers in Marine Science**, v. 9, 26 jul. 2022.

NOGUCHI, Ryozo et al. Artificial neural networks model for estimating growth of polyculture microalgae in an open raceway pond. **Biosystems Engineering**, v. 177, p. 122-129, 2019. Disponível em: https://doi.org/10.1016/j.biosystemseng.2018.09.007.

PÄÄKKÖNEN, S. et al. Non-invasive monitoring of microalgae cultivations using hyperspectral imager. **Journal of Applied Phycology**, v. 36, n. 4, p. 1653–1665, 1 ago. 2024.

PACKER, A. et al. Growth and neutral lipid synthesis in green microalgae: A mathematical model. *Bioresource Technology*, v. 102, p. 111–117, 2011. DOI: 10.1016/j.biortech.2010.06.029.

PEDREGOSA, F. et al. Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, v. 12, n. 85, p. 2825–2830, 2011.

PETER, A. P. et al. Environmental analysis of *Chlorella vulgaris* cultivation in large scale closed system under waste nutrient source. **Chemical Engineering Journal**, v. 433, p. 134254, 1 abr. 2022.

PICCIALLI, V.; SCHWIDDESSEN, J.; SUDOSO, A. M. Optimization meets machine learning: an exact algorithm for semi-supervised support vector machines. **Mathematical Programming**, 19 dez. 2024.

PILARIO, K. E. S.; CAO, Y.; SHAFIEE, M. A Kernel Design Approach to Improve Kernel Subspace Identification. **IEEE Transactions on Industrial Electronics**, v. 68, n. 7, p. 6171–6180, jul. 2021.

POZZOBON, V. et al. Nitrate and nitrite as mixed source of nitrogen for Chlorella vulgaris: fast nitrogen quantification using spectrophotometer and machine learning. **Journal of Applied Phycology**, v. 33, n. 3, p. 1389–1397, 1 jun. 2021.

PRAMODITHA, R. **Overview of a Neural Network's Learning Process**. **Data Science 365**, 1 fev. 2022. Disponível em: <https://medium.com/data-science-365/overview-of-a-neural-networks-learning-process-61690a502fa>. Acesso em: 15 set. 2024

PRICE, G. A. V. et al. Development and Validation of Multiple Linear Regression Models for Predicting Chronic Zinc Toxicity to Freshwater Microalgae. **Environmental Toxicology and Chemistry**, v. 42, n. 12, p. 2630–2641, dez. 2023.

QUIÑONES-GRUEIRO, M. et al. Data-driven monitoring of multimode continuous processes: A review. **Chemometrics and Intelligent Laboratory Systems**, v. 189, p. 56–71, 15 jun. 2019.

RAMANDANI, A. A. et al. Artificial intelligence-driven prediction models for the cultivation of *Chlorella vulgaris* FSP-E in food waste culture medium: A comparative analysis and validation of models. **Algal Research**, p. 103935, 25 jan. 2025.

RANGANATHAN, S. et al. (Eds.). **Encyclopedia of Bioinformatics and Computational Biology**. Oxford: Academic Press, 2019. p. 195–198.

RANSTAM, J.; COOK, J. A. LASSO regression. **British Journal of Surgery**, v. 105, n. 10, p. 1348–1348, 7 ago. 2018.

RAO, M. et al. A secure routing protocol using hybrid deep regression-based trust evaluation and clustering for mobile ad-hoc network. **Peer-to-Peer Networking and Applications**, v. 16, n. 6, p. 2794–2810, 1 nov. 2023.

RENCHER, A. C.; CHRISTENSEN, W. F. **Methods of Multivariate Analysis**. [s.l.] John Wiley & Sons, p. 323-358, 2012.

RODRÍGUEZ-RANGEL, H. et al. Machine Learning Methods Modeling Carbohydrate-Enriched Cyanobacteria Biomass Production in Wastewater Treatment Systems. *Energies*, v. 15, n. 2500, 2022. DOI: 10.3390/en15072500.

ROGERS, Alexander W. et al**.** A transfer learning approach for predictive modeling of bioprocesses using small data. **Biotechnology and Bioengineering**, v. 119, n. 1, p. 182-192, 2022. Disponível em: https://doi.org/10.1002/bit.27964.

SAFI, C. et al. Morphology, composition, production, processing and applications of *Chlorella vulgaris*: A review. **Renewable and Sustainable Energy Reviews**, v. 35, p. 265–278, 1 jul. 2014.

SAINI, V. K. et al. Learning based short term wind speed forecasting models for smart grid applications: An extensive review and case study. **Electric Power Systems Research**, v. 222, p. 109502, 1 set. 2023.

SARKER, I. H. et al. Mobile Data Science and Intelligent Apps: Concepts, AI-Based Modeling and Research Directions. **Mobile Networks and Applications**, v. 26, n. 1, p. 285–303, fev. 2021.b

SARKER, I. H. Machine Learning: Algorithms, Real-World Applications and Research Directions. **SN Computer Science**, v. 2, n. 3, p. 160, 22 mar. 2021.a

SAXENA, P. et al. Mechanism of nanotoxicity in *Chlorella vulgaris* exposed to zinc and iron oxide. **Toxicology Reports**, v. 8, p. 724–731, 1 jan. 2021.

Schölkopf, B., & Smola, A. J. ***Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond***. MIT Press. (2001).

SCHWARTZ, A. A Reinforcement Learning Method for Maximizing Undiscounted Rewards. Em: **Machine Learning Proceedings 1993**. San Francisco (CA): Morgan Kaufmann, 1993. p. 298–305.

SHAH, I. et al. On the Performance of Jackknife Based Estimators for Ridge Regression. **IEEE Access**, v. 9, p. 68044–68053, 2021.

SHAH, Y. D.; SONI, S. M.; PATEL, M. P. Artificial intelligence in healthcare. **Indian Journal of Pharmacy and Pharmacology**, v. 8, n. 2, p. 102–115, 2021.

SHEIK, A. G. et al. Reinvigorating algal cultivation for biomass production with digital twin technology - a smart sustainable infrastructure. **Algal Research**, v. 84, p. 103779, 1 dez. 2024.

SHERAFATI, N. et al. Effect of supplementation with Chlorella vulgaris on lipid profile in adults: A systematic review and dose-response meta-analysis of randomized controlled trials. **Complementary Therapies in Medicine**, v. 66, p. 102822, 1 jun. 2022.

SHIM, S. W. et al. A Machine Learning-Based Algorithm for Short-Term SMP Forecasting Using 2-Step Method. **Journal of Electrical Engineering & Technology**, v. 18, n. 3, p. 1493–1501, 1 maio 2023.

SIDABUTAR, R. et al. Treatment of effluent from the upflow anaerobic sludge blanket-hollow centered packed bed fermentor by utilizing *Chlorella vulgaris* in a fed-batch system. **Case Studies in Chemical and Environmental Engineering**, v. 9, p. 100756, 1 jun. 2024.

SIX, A. et al. From raw microalgae to bioplastics: Conversion of *Chlorella vulgaris* starch granules into thermoplastic starch. **Carbohydrate Polymers**, v. 342, p. 122342, 15 out. 2024b.

SIX, A. et al. Red light induces starch accumulation in *Chlorella vulgaris* without affecting photosynthesis efficiency, unlike abiotic stress. **Algal Research**, v. 80, p. 103515, 1 jun. 2024a.

SMITI, A. When machine learning meets medical world: Current status and future challenges. **Computer Science Review**, v. 37, p. 100280, 1 ago. 2020.

SONACHALAM, M. et al. Performance analysis of dual-fuel engines using acetylene and microalgae biodiesel: The role of fuel injection timing. **Case Studies in Thermal Engineering**, v. 64, p. 105370, 1 dez. 2024.

SU, M. et al. Applications of Microalgae in Foods, Pharma and Feeds and Their Use as Fertilizers and Biostimulants: Legislation and Regulatory Aspects for Consideration. **Foods**, v. 12, n. 20, p. 3878, jan. 2023.

SUI, X. et al. A review of non-probabilistic machine learning-based state of health estimation techniques for Lithium-ion battery. **Applied Energy**, v. 300, p. 117346, out. 2021.

SUKAMTO, S.; HADIYANTO, H.; KURNIANINGSIH, K. KNN Optimization Using Grid Search Algorithm for Preeclampsia Imbalance Class. **E3S Web of Conferences**, v. 448, p. 02057, 2023.

SUN, X.-M. et al. Microalgae for the production of lipid and carotenoids: a review with focus on stress regulation and adaptation. **Biotechnology for Biofuels**, v. 11, n. 1, p. 272, 4 out. 2018.

SUN, K.; ROY, A.; TOBIN, J. M. Artificial intelligence and machine learning: Definition of terms and current concepts in critical care research. **Journal of Critical Care**, v. 82, p. 154792, 1 ago. 2024.

SUPRIYANTO, S. et al. Artificial neural networks model for estimating growth of polyculture microalgae in an open raceway pond. *Biosystems Engineering*, v. 177, p. 122–129, 2019. DOI: 10.1016/j.biosystemseng.2018.10.002.

SUSANNA, Deepti et al. Increasing productivity of *Spirulina platensis* in photobioreactors using artificial neural network modeling. **Biotechnology and Bioengineering**, v. 116, n. 11, p. 2960-2970, 2019. Disponível em: https://doi.org/10.1002/bit.27123.

SUYKENS, J. A. K.; VANDEWALLE, J. (Eds.). **Nonlinear Modeling: Advanced Black-Box Techniques**. Boston, MA: Springer US, 1998. p. 55–85.

SZTANCS, G. et al. Catalytic hydrothermal carbonization of microalgae biomass for low-carbon emission power generation: the environmental impacts of hydrochar co-firing. **Fuel**, v. 300, p. 120927, 15 set. 2021.

TAN, H. L. et al. Heterotrophic and Mixotrophic Cultivation of Chlorella vulgaris using Chicken Waste Compost as Nutrients Source for Lipid Production. **IOP Conference Series: Earth and Environmental Science**, v. 721, n. 1, p. 012011, 1 abr. 2021.

THEODORIDIS, S.; KOUTROUMBAS, K. (Eds.). **Pattern Recognition (Fourth Edition)**. Boston: Academic Press, 2009. p. 151–260.

TIBSHIRANI, R. Regression Shrinkage and Selection via the Lasso. **Journal of the Royal Statistical Society. Series B (Methodological)**, v. 58, n. 1, p. 267–288, 1996.

VAISH, R. et al. Machine learning applications in power system fault diagnosis: Research advancements and perspectives. **Engineering Applications of Artificial Intelligence**, v. 106, p. 104504, 1 nov. 2021.

VAPNIK, V. N. **The Nature of Statistical Learning Theory**. New York, NY: Springer, 2000. p. 183 -188.

VARSHINI, A. G. P.; KUMARI, A. K. Predictive analytics approaches for software effort estimation: A review. **Indian Journal of Science and Technology**, v. 13, n. 21, p. 2094–2103, 5 jun. 2020.

VELÁSQUEZ-ORTA, S. B. et al. Pilot-scale microalgae cultivation and wastewater treatment using high-rate ponds: a meta-analysis. **Environmental Science and Pollution Research**, v. 31, n. 34, p. 46994–47021, 10 jul. 2024.

WANG, D. et al. Dynamic Modeling of Microalgae Growth and Lipid Production under Transient Light and Nitrogen Conditions. *Environmental Science & Technology*, v. 53, p. 11560–11568, 2019. DOI: 10.1021/acs.est.9b02908.

WASKOM, Michael. *Seaborn: statistical data visualization*. 2020. Disponível em: https://seaborn.pydata.org/. Acesso em: 16 fev. 2025.

WATKINS, C. J. C. H.; DAYAN, P. Q-learning. **Machine Learning**, v. 8, n. 3, p. 279–292, 1 maio 1992.

XING, Y. et al. An insight into the phosphorus distribution in extracellular and intracellular cell of *Chlorella vulgaris* under mixotrophic cultivation. **Algal Research**, v. 60, p. 102482, 1 dez. 2021.

YADAV, M. et al. Quantitative evaluation of Chlorella vulgaris for removal of toxic metals from body. **Journal of Applied Phycology**, v. 34, n. 6, p. 2743–2754, 1 dez. 2022.

YEW, Guo Yong et al. The *Chlorella vulgaris* FSP-E cultivation in waste molasses: Photo-to-property estimation by artificial intelligence. **Chemical Engineering Journal**, v. 402, p. 126230, 2020. Disponível em: https://doi.org/10.1016/j.cej.2020.126230.

YING, X. An Overview of Overfitting and its Solutions. **Journal of Physics: Conference Series**, v. 1168, n. 2, p. 022022, fev. 2019.

YÜKSEL, N. et al. Review of artificial intelligence applications in engineering design perspective. **Engineering Applications of Artificial Intelligence**, v. 118, p. 105697, fev. 2023.

YUN, H.-S.; KIM, Y.-S.; YOON, H.-S. Effect of Different Cultivation Modes (Photoautotrophic, Mixotrophic, and Heterotrophic) on the Growth of Chlorella sp. and Biocompositions. **Frontiers in Bioengineering and Biotechnology**, v. 9, 17 dez. 2021.

ZHANG, M. et al. Top ten intelligent algorithms towards smart manufacturing. **Journal of Manufacturing Systems**, v. 71, p. 158–171, 1 dez. 2023.

ZHANG, Z. et al. Municipal solid waste management challenges in developing regions: A comprehensive review and future perspectives for Asia and Africa. **Science of The Total Environment**, v. 930, p. 172794, 20 jun. 2024.

ZHAO, B.; WEN, X.; HAN, K. **Learning Semi-supervised Gaussian Mixture Models for Generalized Category Discovery**. arXiv, 2023. Disponível em: <https://arxiv.org/abs/2305.06144>. Acesso em: 13 fev. 2025

ZHOU, Y. Advances of machine learning in multi-energy district communities–mechanisms, applications and perspectives. **Energy and AI**, v. 10, p. 100187, 1 nov. 2022.

APÊNDICE A – ALGORITMO LINEAR MULTIVARIADO

```
from pathlib import Path
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.linear_model import LinearRegression
from sklearn.metrics import root_mean_squared_error
from sklearn.metrics import r2_score
import time

#Display data options
pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)

#Path of the data
# Enter
path_ent =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Entrada.xlsx")
# out
path_out =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Saida.xlsx")
# First graph regression result
path_reg =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Regressao\linear.png")
# Second graph residual result
path_res =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Residuos\linear.png")
# Report .txt file path
```

```
path_rep                                                                    =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Report\linear.txt")

# Read the data
X   = pd.read_excel(path_ent)
y   = pd.read_excel(path_out)

# Time measure
inicio  = time.time()
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create and train the linear regression model
model   = LinearRegression()
model.fit(X_test, y_test)

# Make predictions on the train set
y_pred_train    = model.predict(X_train)

# Make predictions on the test set
y_pred          = model.predict(X_test)

# Time measure
fim         = time.time()
duration    = fim-inicio

# Calculate the mean square error and R²
# Test Data
MQE_test    = root_mean_squared_error(y_pred, y_test)
r_2_test    = r2_score(y_test, y_pred)
# Train Data
MQE_train   = root_mean_squared_error(y_train, y_pred_train)
r_2_train   = r2_score(y_train, y_pred_train)
```

```
# Organization of data:
y_test  = pd.DataFrame(y_test)
y_train = pd.DataFrame(y_train)
y_test  = y_test.values
y_train = y_train.values

# Residual calculation:
residuos=  y_pred.reshape(len(y_pred),1)-y_test

# Adicionando a linha y = x para referência
min_val = min(y_train)  # Valor mínimo para definir o início da linha
max_val = max(y_train)  # Valor máximo para definir o fim da linha

# Format text content in file:
def format_array(array):
    return "\n".join(", ".join(f"{num:.4f}" for num in sublist) for sublist in array)

# Report informations about all the regression process
report_text = (
    "Relatório de Resultados de Regressão Linear\n\n"
    "Parte específica do modelo:\n"
    f"Coeficientes: {', '.join(f'{coef:.4f}' for coef in model.coef_[0])}\n"
    f"Intercepto: {model.intercept_[0]:.4f}\n\n"

    "Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.4f} s\n\n"

    "Dados do Conjunto de Treinamento:\n"
    f"Y_train:\n{format_array(y_train)}\n\n"
```

```
    f"Y_pred_train:\n{format_array(y_pred_train)}\n\n"

    "Dados do Conjunto de Teste:\n"
    f"Y_test:\n{format_array(y_test)}\n\n"
    f"Y_pred_test:\n{format_array(y_pred)}\n"
)
path_rep.write_text(report_text)

# Print the report:
print("Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.6f} s\n\n"
    )

# Generate image of regression:
plt.scatter(y_test, y_pred, c='black', label='TESTE')
plt.scatter(y_train, y_pred_train, marker = '+', c='blue', label ='TREINO')
plt.plot([min_val, max_val],[min_val, max_val], color="red", linestyle="-")
#plt.title (f"")
plt.legend()
plt.xlabel('Experimental')
plt.ylabel('Simulado')
plt.savefig(path_reg, format='png')
plt.show()

# Previsões e cálculo dos resíduos
# Gráfico de resíduos
plt.scatter(y_pred, residuos, color="black")
plt.axhline(y=0, color="red", linestyle="-")
plt.xlabel("Valores previstos")
plt.ylabel("Resíduos")
```

```
#plt.title("Gráfico de Resíduos")
plt.savefig(path_res, format='png')
plt.show()
```

APÊNDICE B – ALGORITMO LASSO

```
from pathlib import Path
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.linear_model import Lasso
from sklearn.metrics import root_mean_squared_error
from sklearn.metrics import r2_score
from sklearn.model_selection import GridSearchCV, train_test_split
import time

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)

#Path of the data
# Enter
path_ent =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida1\Entrada.
xlsx")
# out
path_out =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Saida.xlsx")
# First graph regression result
path_reg =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Regressao\Lasso.png")
# Second graph residual result
path_res =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Residuos\Lasso.png")
# Report .txt file path
```

```
path_rep                                                                    =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Report\Lasso.txt")

# Read the data
X   = pd.read_excel(path_ent)
y   = pd.read_excel(path_out)

# Time measure
inicio  = time.time()
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Define the model:
model = Lasso(max_iter=10**7, tol=10**-8)

# Definindo os parâmetros para busca
param_grid = {
            'alpha': np.logspace(-4, 4, 100), # Testa valores de alpha de 0.0001 a 10000
             'selection':['ciclic', 'random'],
             'fit_intercept':[True, False],
             'positive':[True, False],
             'precompute': [True, False]
}

# GridSearchCV para Ridge
# GridSearchCV para o modelo
best_model          =          GridSearchCV(model,          param_grid,          cv=5,
scoring='neg_mean_squared_error')
best_model.fit(X_train, y_train.values.ravel())
model_estimator = best_model.best_estimator_

# Make predictions on the train set
y_pred_train    = model_estimator.predict(X_train)
```

```
# Make predictions on the test set
y_pred          = model_estimator.predict(X_test)

# Time measure
fim         = time.time()
duration    = fim-inicio

# Calculate the mean square error and R²
# Test Data
MQE_test    = root_mean_squared_error(y_pred, y_test)
r_2_test    = r2_score(y_test, y_pred)
# Train Data
MQE_train   = root_mean_squared_error(y_train, y_pred_train)
r_2_train   = r2_score(y_train, y_pred_train)

# Organization of data:
y_test  = pd.DataFrame(y_test)
y_train = pd.DataFrame(y_train)
y_test  = y_test.values
y_train = y_train.values

# Residual calculation:
residuos=  y_pred.reshape(len(y_pred),1)-y_test

# Adicionando a linha y = x para referência
min_val = min(y_train)  # Valor mínimo para definir o início da linha
max_val = max(y_train)  # Valor máximo para definir o fim da linha

# Report informations about all the regression process
report_text = (
    "Relatório de Resultados de Regressão Linear\n\n"
    "Parte específica do modelo:\n"
    f"Coeficientes: {best_model.best_params_}\n"
```

```
    "Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.4f} s\n\n"

    "Dados do Conjunto de Treinamento:\n"
    f"Y_train:\n{y_train}\n\n"
    f"Y_pred_train:\n{y_pred_train}\n\n"

    "Dados do Conjunto de Teste:\n"
    f"Y_test:\n{y_test}\n\n"
    f"Y_pred_test:\n{y_pred}\n"
)
path_rep.write_text(report_text)

# Print the report:
print("Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.6f} s\n\n"
    )

# Generate image of regression:
plt.scatter(y_test, y_pred, c='black', label='TESTE')
plt.scatter(y_train, y_pred_train, marker = '+', c='blue', label ='TREINO')
plt.plot([min_val, max_val],[min_val, max_val], color="red", linestyle="-")
#plt.title (f"")
plt.legend()
plt.xlabel('Experimental')
```

```
plt.ylabel('Simulado')
plt.savefig(path_reg, format='png')
plt.show()

# Previsões e cálculo dos resíduos
# Gráfico de resíduos
plt.scatter(y_pred, residuos, color="black")
plt.axhline(y=0, color="red", linestyle="-")
plt.xlabel("Valores previstos")
plt.ylabel("Resíduos")
#plt.title("Gráfico de Resíduos")
plt.savefig(path_res, format='png')
plt.show()
```

APÊNDICE C – ALGORITMO RIDGE

```
from pathlib import Path
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.linear_model import Ridge
from sklearn.metrics import root_mean_squared_error
from sklearn.metrics import r2_score
from sklearn.model_selection import GridSearchCV, train_test_split
import time

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)

#Path of the data
# Enter
path_ent =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Entrada.xlsx")
# out
path_out =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Saida.xlsx")
# First graph regression result
path_reg =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Regressao\Ridge.png")
# Second graph residual result
path_res =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Residuos\Ridge.png")
# Report .txt file path
```

```
path_rep                                                                    =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Report\Ridge.txt")

# Read the data
X   = pd.read_excel(path_ent)
y   = pd.read_excel(path_out)

# Time measure
inicio  = time.time()
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Define the model:
model = Ridge(max_iter=10**7, tol=10**-8)

# Definindo os parâmetros para busca
param_grid = {
              'alpha': np.logspace(-4, 4, 10**4), # Testa valores de alpha de 0.0001 a
10000
               'solver': ['svd', 'cholesky', 'sparse_cg', 'lsqr', 'sag', 'lbfgs'], # Testa com
diferentes tipos de solver
            'positive':[True, False]
}

# GridSearchCV para Ridge
# GridSearchCV para o modelo
best_model          =          GridSearchCV(model,          param_grid,          cv=5,
scoring='neg_mean_squared_error')
best_model.fit(X_train, y_train.values.ravel())
model_estimator = best_model.best_estimator_

# Make predictions on the train set
y_pred_train    = model_estimator.predict(X_train)
```

```
# Make predictions on the test set
y_pred          = model_estimator.predict(X_test)

# Time measure
fim         = time.time()
duration    = fim-inicio

# Calculate the mean square error and R²
# Test Data
MQE_test    = root_mean_squared_error(y_pred, y_test)
r_2_test    = r2_score(y_test, y_pred)
# Train Data
MQE_train   = root_mean_squared_error(y_train, y_pred_train)
r_2_train   = r2_score(y_train, y_pred_train)

# Organization of data:
y_test  = pd.DataFrame(y_test)
y_train = pd.DataFrame(y_train)
y_test  = y_test.values
y_train = y_train.values

# Residual calculation:
residuos=  y_pred.reshape(len(y_pred),1)-y_test

# Adicionando a linha y = x para referência
min_val = min(y_train)  # Valor mínimo para definir o início da linha
max_val = max(y_train)  # Valor máximo para definir o fim da linha

# Report informations about all the regression process
report_text = (
    "Relatório de Resultados de Regressão Linear\n\n"
    "Parte específica do modelo:\n"
    f"Coeficientes: {best_model.best_params_}\n"
```

```
    "Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.4f} s\n\n"

    "Dados do Conjunto de Treinamento:\n"
    f"Y_train:\n{y_train}\n\n"
    f"Y_pred_train:\n{y_pred_train}\n\n"

    "Dados do Conjunto de Teste:\n"
    f"Y_test:\n{y_test}\n\n"
    f"Y_pred_test:\n{y_pred}\n"
)
path_rep.write_text(report_text)

# Print the report:
print("Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.6f} s\n\n"
    )

# Generate image of regression:
plt.scatter(y_test, y_pred, c='black', label='TESTE')
plt.scatter(y_train, y_pred_train, marker = '+', c='blue', label ='TREINO')
plt.plot([min_val, max_val],[min_val, max_val], color="red", linestyle="-")
#plt.title (f"")
plt.legend()
plt.xlabel('Experimental')
```

```
plt.ylabel('Simulado')
plt.savefig(path_reg, format='png')
plt.show()

# Previsões e cálculo dos resíduos
# Gráfico de resíduos
plt.scatter(y_pred, residuos, color="black")
plt.axhline(y=0, color="red", linestyle="-")
plt.xlabel("Valores previstos")
plt.ylabel("Resíduos")
#plt.title("Gráfico de Resíduos")
plt.savefig(path_res, format='png')
plt.show()
```

APÊNDICE D – ALGORITMO RNA

```
from pathlib import Path
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
import time
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from pathlib import Path
from sklearn.metrics import root_mean_squared_error
from sklearn.metrics import r2_score
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.neural_network import MLPRegressor
from sklearn.preprocessing import StandardScaler

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)

#Path of the data
# Enter
path_ent =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Entrada.xlsx")
# out
path_out =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Saida.xlsx")
# First graph regression result
path_reg =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Regressao\RNA.png")
# Second graph residual result
```

```
path_res =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Residuos\RNA.png")
# Report .txt file path
path_rep =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Report\RNA.txt")

# Read the data
X   = pd.read_excel(path_ent)
y   = pd.read_excel(path_out)

# Time measure
inicio  = time.time()
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Normalização
# Dados de entrada
scalerInput = StandardScaler()
scalerInput.fit(X_train)

X_train = scalerInput.transform(X_train)
X_test = scalerInput.transform(X_test)

# Dados de saída
scalerOutput = StandardScaler()
scalerOutput.fit(y_train)

y_train = scalerOutput.transform(y_train)
y_test = scalerOutput.transform(y_test)

# Definindo o modelo
model = MLPRegressor(random_state=42)
```

```
# Definindo os parâmetros para busca
parameter_space = {
    'hidden_layer_sizes': [(3,), (5,), (6,),(10,), (11,), (12,), (13,),(14,), (15,), (16),
                           (3, 3,), (5, 5,), (7, 7,),
                           (3, 3, 3), (5, 5, 5), (6, 6, 6)],
    'activation': ['tanh', 'relu', 'logistic', 'identity'],
    'solver': ['sgd', 'adam', 'lbfgs'],
    'alpha': [0.0001, 0.05, 0.1, 0.5],
    'learning_rate': ['constant', 'adaptive', 'invscaling'],
}

# GridSearchCV para o modelo
best_model = GridSearchCV(model, parameter_space, n_jobs=-1,
scoring='neg_mean_squared_error')
best_model.fit(X_train, y_train)
model_estimator = best_model.best_estimator_

# Make predictions on the train set
y_pred_train    = model_estimator.predict(X_train)

# Make predictions on the test set
y_pred          = model_estimator.predict(X_test)

# Time measure
fim         = time.time()
duration    = fim-inicio

# Calculate the mean square error and R²
# Test Data
MQE_test    = root_mean_squared_error(y_pred, y_test)
r_2_test    = r2_score(y_test, y_pred)
# Train Data
MQE_train   = root_mean_squared_error(y_train, y_pred_train)
r_2_train   = r2_score(y_train, y_pred_train)
```

```
# Organization of data:
y_test  = pd.DataFrame(y_test)
y_train = pd.DataFrame(y_train)
y_test  = y_test.values
y_train = y_train.values

# ajuste das matrizes
y_test       =  y_test.reshape(-1,1)
y_pred       =  y_pred.reshape(-1,1)
y_train      =  y_train.reshape(-1,1)
y_pred_train =  y_pred_train.reshape(len(y_pred_train),1)

#mudanca para variaveis originais
y_test         =   scalerOutput.inverse_transform(y_test)
y_pred         =   scalerOutput.inverse_transform(y_pred)
y_train        =   scalerOutput.inverse_transform(y_train)
y_pred_train   =   scalerOutput.inverse_transform(y_pred_train)
#fim da mudanca

# Residual calculation:
residuos=  y_pred.reshape(len(y_pred),1)-y_test

# Adicionando a linha y = x para referência
min_val = min(y_train)  # Valor mínimo para definir o início da linha
max_val = max(y_train)  # Valor máximo para definir o fim da linha

# Report informations about all the regression process
report_text = (
    "Relatório de Resultados de Regressão MLP\n\n"
    "Parte específica do modelo:\n"
    f"Coeficientes: {best_model.best_params_}\n"
    f"Best parameters found:\n, {best_model.best_estimator_.coefs_}\n"
    f"Best intercept found:\n, {best_model.best_estimator_.intercepts_}\n"
```

```
    "Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.4f} s\n\n"

    "Dados do Conjunto de Treinamento:\n"
    f"Y_train:\n{y_train}\n\n"
    f"Y_pred_train:\n{y_pred_train}\n\n"

    "Dados do Conjunto de Teste:\n"
    f"Y_test:\n{y_test}\n\n"
    f"Y_pred_test:\n{y_pred}\n"
)
path_rep.write_text(report_text)

# Print the report:
print("Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.6f} s\n\n"
    )

# Generate image of regression:
plt.scatter(y_test, y_pred, c='black', label='TESTE')
plt.scatter(y_train, y_pred_train, marker = '+', c='blue', label ='TREINO')
plt.plot([min_val, max_val],[min_val, max_val], color="red", linestyle="-")
#plt.title (f"")
plt.legend()
```

```
plt.xlabel('Experimental')
plt.ylabel('Simulado')
plt.savefig(path_reg, format='png')
plt.show()

# Previsões e cálculo dos resíduos
# Gráfico de resíduos
plt.scatter(y_pred, residuos, color="black")
plt.axhline(y=0, color="red", linestyle="-")
plt.xlabel("Valores previstos")
plt.ylabel("Resíduos")
#plt.title("Gráfico de Resíduos")
plt.savefig(path_res, format='png')
plt.show()
```

APÊNDICE E – ALGORITMO SVR

```
from pathlib import Path
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.model_selection import GridSearchCV, train_test_split
import time
from sklearn.metrics import root_mean_squared_error
from sklearn.metrics import r2_score
from sklearn.preprocessing import StandardScaler
from sklearn.svm import SVR

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)

#Path of the data
# Enter
path_ent =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Entrada.xlsx")
# out
path_out =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Saida.xlsx")
# First graph regression result
path_reg =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Regressao\SVR.png")
# Second graph residual result
path_res =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Residuos\SVR.png")
# Report .txt file path
```

```
path_rep                                                                =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Report\SVR.txt")

# Read the data
X   = pd.read_excel(path_ent)
y   = pd.read_excel(path_out)

# Time measure
inicio  = time.time()
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Normalização
# Dados de entrada
scalerInput = StandardScaler()
scalerInput.fit(X_train)

X_train = scalerInput.transform(X_train)
X_test = scalerInput.transform(X_test)

# Dados de saída
scalerOutput = StandardScaler()
scalerOutput.fit(y_train)

y_train = scalerOutput.transform(y_train)
y_test = scalerOutput.transform(y_test)

# Definindo o modelo
model = SVR(max_iter=10**6, tol=10**-8)
# Definindo os parâmetros para busca
parameter_space = {
    'kernel': ['linear', 'poly', 'rbf', 'sigmoid'],
    'degree': [1, 2, 3, 4, 5],
```

```
    'gamma': ['scale', 'auto'],
    'coef0': [0, 0.25, 0.75, 0.5, 1, 2],
    'C': [.5,.75, 1, 1.5, 2],
    'epsilon': [0.1,0.01, 0.001, 0.5, 1],
    'shrinking': [True, False]
}

# GridSearchCV para o modelo
best_model = GridSearchCV(model, parameter_space, n_jobs=-1,
scoring='neg_mean_squared_error')
best_model.fit(X_train, y_train.ravel())
model_estimator = best_model.best_estimator_

# Make predictions on the train set
y_pred_train    = model_estimator.predict(X_train)

# Make predictions on the test set
y_pred          = model_estimator.predict(X_test)

# Time measure
fim         = time.time()
duration    = fim-inicio

# Calculate the mean square error and R²
# Test Data
MQE_test    = root_mean_squared_error(y_pred, y_test)
r_2_test    = r2_score(y_test, y_pred)
# Train Data
MQE_train   = root_mean_squared_error(y_train, y_pred_train)
r_2_train   = r2_score(y_train, y_pred_train)

# Organization of data:
y_test  = pd.DataFrame(y_test)
y_train = pd.DataFrame(y_train)
```

```
y_test  = y_test.values
y_train = y_train.values

# ajuste das matrizes
y_test       =  y_test.reshape(-1,1)
y_pred       =  y_pred.reshape(-1,1)
y_train      =  y_train.reshape(-1,1)
y_pred_train =  y_pred_train.reshape(len(y_pred_train),1)

#mudanca para variaveis originais
y_test         =   scalerOutput.inverse_transform(y_test)
y_pred         =   scalerOutput.inverse_transform(y_pred)
y_train        =   scalerOutput.inverse_transform(y_train)
y_pred_train   =   scalerOutput.inverse_transform(y_pred_train)
#fim da mudanca

# Residual calculation:
residuos=  y_pred.reshape(len(y_pred),1)-y_test

# Adicionando a linha y = x para referência
min_val = min(y_train)  # Valor mínimo para definir o início da linha
max_val = max(y_train)  # Valor máximo para definir o fim da linha

# Report informations about all the regression process
report_text = (
    "Relatório de Resultados de Regressão MLP\n\n"
    "Parte específica do modelo:\n"
    f"Coeficientes: {best_model.best_params_}\n"
    f"Best parameters found:\n, {best_model.best_estimator_.support_vectors_}\n"
    f"Best intercept found:\n, {best_model.best_estimator_.intercept_}\n"

    "Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
```

```
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.4f} s\n\n"

    "Dados do Conjunto de Treinamento:\n"
    f"Y_train:\n{y_train}\n\n"
    f"Y_pred_train:\n{y_pred_train}\n\n"

    "Dados do Conjunto de Teste:\n"
    f"Y_test:\n{y_test}\n\n"
    f"Y_pred_test:\n{y_pred}\n"
)
path_rep.write_text(report_text)

# Print the report:
print("Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.6f} s\n\n"
    )

# Generate image of regression:
plt.scatter(y_test, y_pred, c='black', label='TESTE')
plt.scatter(y_train, y_pred_train, marker = '+', c='blue', label ='TREINO')
plt.plot([min_val, max_val],[min_val, max_val], color="red", linestyle="-")
#plt.title (f"")
plt.legend()
plt.xlabel('Experimental')
plt.ylabel('Simulado')
plt.savefig(path_reg, format='png')
plt.show()
```

```
# Previsões e cálculo dos resíduos
# Gráfico de resíduos
plt.scatter(y_pred, residuos, color="black")
plt.axhline(y=0, color="red", linestyle="-")
plt.xlabel("Valores previstos")
plt.ylabel("Resíduos")
#plt.title("Gráfico de Resíduos")
plt.savefig(path_res, format='png')
plt.show()
```

APÊNDICE F – ALGORITMO RANDOM FOREST

```
from pathlib import Path
import matplotlib.pyplot as plt
import pandas as pd
import numpy as np
from sklearn.metrics import root_mean_squared_error
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
from sklearn.model_selection import GridSearchCV, train_test_split
from sklearn.preprocessing import StandardScaler
import time

pd.set_option('display.max_rows', None)
pd.set_option('display.max_columns', None)
pd.set_option('display.width', None)

#Path of the data
# Enter
path_ent=
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida1\Entrada.
xlsx")
# out
path_out                                                    =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Saida.xlsx")
# First graph regression result
path_reg                                                    =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Regressao\RF.png")
# Second graph residual result
path_res                                                    =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Residuos\RF.png")
```

```
# Report .txt file path
path_rep                                                                    =
Path(r"C:\Users\axiaj\OneDrive\Documentos\mestrado\IA\Simulation\Saida
1\Report\RF.txt")

# Read the data
X   = pd.read_excel(path_ent)
y   = pd.read_excel(path_out)

# Time measure
inicio  = time.time()
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Normalização
# Dados de entrada
scalerInput = StandardScaler()
scalerInput.fit(X_train)

X_train = scalerInput.transform(X_train)
X_test = scalerInput.transform(X_test)

# Dados de saída
scalerOutput = StandardScaler()
scalerOutput.fit(y_train)

y_train = scalerOutput.transform(y_train)
y_test = scalerOutput.transform(y_test)

# Definindo o modelo
model = RandomForestRegressor(random_state=42)
# Definindo os parâmetros para busca
param_grid = {
    'n_estimators': [20, 50, 100, 150, 200],       # Número de árvores
```

```
    'max_depth': [None, 5, 10, 20, 30],      # Profundidade máxima da árvore
    'min_samples_split': [2, 3, 4, 5, 6, 7, 10],      # Mínimo de amostras para dividir um nó
    'min_samples_leaf': [1, 2, 3, 4],        # Mínimo de amostras em cada folha
    'bootstrap': [True, False]           # Se deve usar amostragem com ou sem reposição
}

# GridSearchCV para o modelo
best_model = GridSearchCV(model, param_grid, cv=5, scoring='neg_mean_squared_error')
best_model.fit(X_train, y_train.ravel())
model_estimator = best_model.best_estimator_

# Make predictions on the train set
y_pred_train    = model_estimator.predict(X_train)

# Make predictions on the test set
y_pred          = model_estimator.predict(X_test)

# Time measure
fim         = time.time()
duration    = fim-inicio

# Calculate the mean square error and R²
# Test Data
MQE_test    = root_mean_squared_error(y_pred, y_test)
r_2_test    = r2_score(y_test, y_pred)
# Train Data
MQE_train   = root_mean_squared_error(y_train, y_pred_train)
r_2_train   = r2_score(y_train, y_pred_train)

# Organization of data:
y_test  = pd.DataFrame(y_test)
y_train = pd.DataFrame(y_train)
```

```
y_test  = y_test.values
y_train = y_train.values

# ajuste das matrizes
y_test       =  y_test.reshape(-1,1)
y_pred       =  y_pred.reshape(-1,1)
y_train      =  y_train.reshape(-1,1)
y_pred_train =  y_pred_train.reshape(len(y_pred_train),1)

#mudanca para variaveis originais
y_test          =   scalerOutput.inverse_transform(y_test)
y_pred          =   scalerOutput.inverse_transform(y_pred)
y_train         =   scalerOutput.inverse_transform(y_train)
y_pred_train    =   scalerOutput.inverse_transform(y_pred_train)
#fim da mudanca

# Residual calculation:
residuos=  y_pred.reshape(len(y_pred),1)-y_test

# Adicionando a linha y = x para referência
min_val = min(y_train)  # Valor mínimo para definir o início da linha
max_val = max(y_train)  # Valor máximo para definir o fim da linha

# Report informations about all the regression process
report_text = (
    "Relatório de Resultados de Regressão Random Forest\n\n"
    "Parte específica do modelo:\n"
    f"Coeficientes: {best_model.best_params_}\n"

    "Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
```

```
    f"Tempo de execução: {duration:.4f} s\n\n"

    "Dados do Conjunto de Treinamento:\n"
    f"Y_train:\n{y_train}\n\n"
    f"Y_pred_train:\n{y_pred_train}\n\n"

    "Dados do Conjunto de Teste:\n"
    f"Y_test:\n{y_test}\n\n"
    f"Y_pred_test:\n{y_pred}\n"
)
path_rep.write_text(report_text)

# Print the report:
print("Métricas de Avaliação:\n"
    f"R² - Treino: {r_2_train:.4f}\n"
    f"R² - Teste: {r_2_test:.4f}\n"
    f"RMSE - Treino: {MQE_train:.4f}\n"
    f"RMSE - Teste: {MQE_test:.4f}\n\n"
    f"Tempo de execução: {duration:.6f} s\n\n"
    )

# Generate image of regression:
plt.scatter(y_test, y_pred, c='black', label='TESTE')
plt.scatter(y_train, y_pred_train, marker = '+', c='blue', label ='TREINO')
plt.plot([min_val, max_val],[min_val, max_val], color="red", linestyle="-")
#plt.title (f"")
plt.legend()
plt.xlabel('Experimental')
plt.ylabel('Simulado')
plt.savefig(path_reg, format='png')
plt.show()

# Previsões e cálculo dos resíduos
# Gráfico de resíduos
```

```
plt.scatter(y_pred, residuos, color="black")
plt.axhline(y=0, color="red", linestyle="-")
plt.xlabel("Valores previstos")
plt.ylabel("Resíduos")
#plt.title("Gráfico de Resíduos")
plt.savefig(path_res, format='png')
plt.show()
```