UNIVERSIDADE DE SÃO PAULO
Instituto de Ciências Matemáticas e de Computação

Aprendizado de máquina para detecção de desvios de qualidade em pontos de solda na indústria automotiva

Caio Quaglierini Ribeiro

Monografia - MBA em Inteligência Artificial e Big Data

ICMC USP
SÃO CARLOS

Caio Quaglierini Ribeiro

Aprendizado de máquina para detecção de desvios de qualidade em pontos de solda na indústria automotiva

Monografia apresentada ao Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, como parte dos requisitos para obtenção do título de Especialista em Inteligência Artificial e Big Data.

Área de concentração: Inteligência Artificial

Orientador: Prof. Dr. Valdir Grassi Junior

Versão original

São Carlos

2024

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

Ribeiro, Caio Quaglierini
R484a Aprendizado de máquina para detecção de desvios de qualidade em pontos de solda na indústria automotiva / Caio Quaglierini Ribeiro; orientador Valdir Grassi Junior. -- São Carlos, 2024.
77 p.

Trabalho de conclusão de curso (MBA em Inteligência Artificial e Big Data) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2024.

1. Aprendizado de Máquina. 2. Ponto de solda. 3. Qualidade. 4. Classificação. 5. Desvios. I. Grassi Junior, Valdir, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Caio Quaglierini Ribeiro

Machine Learning for Detecting Quality Deviations of Weld Spots in the Automotive Industry

Monograph presented to the Departamento de Ciências de Computação do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo - ICMC/USP, as part of the requirements for obtaining the title of Specialist in Artificial Intelligence and Big Data.

Concentration area: Artificial Intelligence

Advisor: Prof. Dr. Valdir Grassi Junior

Original version

São Carlos

2024

Esta monografia é dedicada aos meus amigos, familiares e docentes que estiveram comigo durante minha jornada acadêmica e profissional.

AGRADECIMENTOS

Meus sinceros agradecimentos à coordenação do curso MBA em Inteligência Artificial e Big Data, professoras Dra. Roseli Aparecida Francelin Romero e Dra. Solange Oliveira Rezende, pela elaboração de um excelente curso e trilhar meu aprendizado na área. Entre os demais docentes que participaram de minha jornada, sou extremamente grato especialmente a meu orientador Prof. Dr. Valdir Grassi Junior por me auxiliar em momentos de questionamentos. Por disponibilizar seu tempo e habilidade para me auxiliar com sugestões e suporte durante o desenvolvimento do projeto, demonstrou-se um excelente professor.

Gostaria de expressar minha profunda gratidão a Thiago Tomageski, Anderson Valentim Menezes e Fernando Coutinho Irigoyen Moyano, por sempre disponibilizarem recursos e suporte para o desenvolvimento de todo o projeto, bem como oportunidades de desenvolvimento pessoal. Igualmente relevante, meus agradecimentos aos técnicos Midiã Azevedo Faustino e Rodrigo Batista das Neves, por contribuírem principalmente com extrema competência técnica sobre o assunto, tempo e na catalogação de dados.

Por fim, agradeço aos meus familiares, amigos e colegas de trabalho, pelo apoio incondicional durante a elaboração do projeto, promovendo meu sucesso e desenvolvimento pessoal e profissional.

"O desejo profundo da humanidade pelo conhecimento é justificativa suficiente para nossa busca contínua."

Stephen Hawking

RESUMO

RIBEIRO, C. Q. **Aprendizado de máquina para detecção de desvios de qualidade em pontos de solda na indústria automotiva**. 2024. 77p. Monografia (MBA em Inteligência Artificial e Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2024.

Uma empresa automotiva alvo de estudo encontra 16.259 falhas em pontos de solda por ano – sejam elas rebarba, solda fria, pontos soltos ou retorcidos –, causando retrabalho de 4,5% do *takt time*. Visando contribuir para a modernização do setor, este trabalho propõe otimizar as inspeções manuais de qualidade de solda, com digitalização do processo e detecção automática de anomalias através dos dados coletados. Para isso, foi criado um ecossistema de Big Data, conectando o controlador de solda Matuschek a um servidor SQL. Em 6 meses, foram coletados dados quase 50 milhões de pontos de solda (15GB), com parâmetros como corrente elétrica, energia e espessura de chapas. Neste trabalho, foram utilizados métodos de aprendizado indutivo para detecção das falhas. Escolheu-se os modelos de SVM, MLP e CATBoost para aprendizado supervisionado e K-Means e clusterização hierárquica para não supervisionado. Contudo, as referências na literatura indicavam que SVM, MLP e K-Means demonstrariam melhor desempenho. Devido a natureza das anomalias de solda e a alta demanda de recursos especializados para catalogação dos dados, foi inviável catalogar toda a base de dados de 6 meses, bem como treinar modelos com alto nível de especificidade do processo. O conceito foi testado na linha principal de produção com 15 controladores de solda. Então, após rotulação dos dados por técnicos de qualidade e aplicadas técnicas de transformação, a amostra resultante consistiu em 26 parâmetros e 13.110 pontos, sendo 20% de desvios. Por fim, os experimentos demonstraram que clusterização hierárquica e K-Means não foram eficientes: este último, pode ser justificado devido à baixa quantidade de dados e por ser sensível à escolha inicial do centróide e *outliers*, não garantindo assim a convergência para o mínimo global. Já os modelos de aprendizado supervisionado apresentaram acurácias semelhantes: 89,60% (SVM), 90,02% (MLP) e 91,06% (CATBoost). A diferença maior encontrada foi no valor de *F1-score* e tempo de treinamento. Logo, o modelo de *Multi-Layer Perceptron* apresentou melhor desempenho entre os três modelos principais inicialmente avaliados, mas o CATBoost foi superior também em processamento. Os resultados obtidos foram considerados aceitáveis para um projeto piloto, provando encontrar duas soluções para classificação de pontos de solda. Todavia, futuramente deseja-se otimizar a busca de indicadores e parâmetros para o modelo, incluindo dados de afiação de eletrôdos e, conforme o algoritmo é implementado na produção, utilizar novos dados e suas classificações para reavaliar os modelos.

Palavras-chave: Aprendizado de Máquina; ponto de solda; desvios; qualidade; classificação

ABSTRACT

RIBEIRO, C. Q. **Machine Learning for Detecting Quality Deviations of Weld Spots in the Automotive Industry**. 2024. 77p. Monograph (MBA in Artificial Intelligence and Big Data) - Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos, 2024.

An automotive company subject of study detects 16.259 deviations on weld spots per year – regarding burr, twisted or loose spots –, causing a 4,5% takt time rework. In order to contribute to the modernization of this sector, the purpose of this work is to optimize manual welding quality inspections with process digitization and automatic anomaly detection using collected data. As such, it has been built a Big Data environment by connecting the Matuschek welding controller to a SQL server. In 6 months, almost 50 million data (15GB) from weld spots were collected, regarding parameters such as electric current, energy and sheet thickness. In this work, inductive learning methods were used for defects detection. SVM, MLP and CATBoost were chosen as supervised learning models and K-Means and hierarchy clustering for unsupervised learning. However, literature references pointed out that SVM, MLP and K-Means would show better performance. Due to welding anomalies nature and the high demand for data labeling specialized resources, it was not viable to label the entire 6 months database, as well as to train high specificity process models. The concept was tested at the main production line, which holded 15 welding controllers. So, after quality technicians labeled the data and data transformation methods were applied, the resulting sample had 26 variables and 13.110 weld spots, of which 20% were deviations. Finally, the experiments showed that both the hierarchical and particional clustering were not efficient: K-means can be justified due to the low data quantity and for being sensitive to the initial centroid choosing and outliers, not ensuring a lowest global value convergence. On the other hand, supervised learning models presented similar accuracies: 89,60% (SVM), 90,02% (MLP) e 91,06% (CATBoost). The biggest difference was the F1-score and spent trainning time. Therefore, the Multilayer Perceptron model showed better performance between the three main algorithms initially proposed, but CATBoost was also superior in processing. The given results were considered acceptable for a pilot project, proving to find two possible solutions for welding spots classification. However, in the future the model features search could be optimized, including tip dressing data and, as the algorithm is implemented on production system, to use new data and its classification in order to reevaluate the models.

Keywords: Machine Learning; weld spot; deviations; quality; classification

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ACO Ant Colony Optimization

ADALINE Adaptive Linear Element

ADAM Adaptive Moment Estimation

ANFIS Adaptive Network-Based Fuzzy Inference System

ANN Artificial Neural Network

BDK

BiW Body in White

BPN Backpropagation Network

CATBoost Categorical Boosting

DL Deep Learning

DT Decision Tree

DWT Discrete Wavelet Transform

GA Genetic Algorithms

GMAW Gas Metal Arc Welding

KNN k-Nearest Neighbour

LR Linear Regression

MAG Metal Active Gas

MDCR Mean Dynamic Contact Resistance

MIG Metal Inert Gas

MLP Multilayer Perceptron

MSE Mean Square Error

NI Nugget Index

PLC Programmable Logic Controller

PNN Probabilistic Neural Network

PSO Particle Swarm Optimization

RBF Radial Basis Function

ReLU Rectified Linear Unit

RF Random Forest

RMS Root Mean Square

ROC Receiver Operating Characteristic

SGD Stochastic Gradient Descent

SNP Single-Nucleotide Polymorphism

SOM Self-Organizing Feature Map

SVC Support Vector Classification

SVM Support Vector Machine

SVR Support Vector Regression

WQI Water Quality Index

XGBoost Ectreme Gradient Boosting

LISTA DE SÍMBOLOS

a(i)	Distância média de i até outros pontos do mesmo *cluster*
a_j	Atividade de um neruônio j
b	Coeficiente linear
b(i)	Distância média de i até outros pontos do *cluster* diferente mais próximo
C_j	Centróide de um *cluster* j
d_i	ponto amostral
E	Função de erro entre a saída y_l e os valores reais t_l
f_j	Função de transferência da camada oculta j
f_k	Função de transferência da camada oculta k
H	Função de árvore de decisão
h_j	Saída de a_j
I	Corrente elétrica
k	Neurônios na camada de saída
m	Neurônios na camada de saída
n	Parâmetros
P	Distribuição probabilística
Q	Calor
q	Cálculo da distância
R	Resistência do metal
R_j	Região de folhas da árvore de decisão
S(i)	Coeficiente de silhueta
t	Tempo de soldagem
V	Função de perda
w	vetor peso

W_0	*bias*
W_{ij}	peso entre o i-ésimo e j-ésimo neurônio
W_{jk}	peso entre o j-ésimo e k-ésimo neurônio
x	ponto vetorial
δ	sinal de erro
ζ	função de custo
η	eficiência térmica de soldagem

SUMÁRIO

1 INTRODUÇÃO

1.1 Contextualização

A Terceira Revolução Industrial, marcada pela automação da manufatura e uso extensivo da informática no início da década de 1970, impulsionou a integração da robótica na produção industrial. Além de tecnologia, nesse momento também foram estudados novos conceitos de produção adotados especialmente pela indústria automotiva, visando melhorar a eficiência de toda a cadeia produtiva. Um exemplo desses conceitos foi o Toyotismo, o qual trouxe a descentralização dos polos industriais para outros locais, a produção sob demanda para manter um estoque mínimo (*just in time*) e controle de qualidade ao longo do processo, não mais no final da linha de montagem (VARGAS; BUNDE, 2021).

No Brasil, a relevância da indústria automotiva desde os anos 1970 foi notável, influenciada pela abertura de montadoras estrangeiras e políticas de incentivo governamental. Esse cenário foi crucial para a consolidação do país como um dos principais polos automotivos do mundo, impulsionando o desenvolvimento tecnológico, a geração de empregos e o crescimento econômico nacional. O setor saiu de um faturamento de U$ 12,2 bilhões ao final da década de 1980 para U$ 37,3 bilhões no início do novo milênio (DAUDT; WILLCOX, 2018). Na última década, chegou a representar 5% do PIB nacional (ANFAVEA, 2018).

A modernização de diversos processos da indústria automotiva nacional durante as últimas décadas evidencia os resultados econômicos descritos. A robotização da soldagem de carrocerias, por exemplo, transformou as tarefas repetitivas e de grande volume, agora com tempo de ciclo e fadiga muito inferiores, visto que a soldagem – seja *MIG (Metal Inert Gas) brazing/ MAG (Metal Active Gas) brazing* , a laser ou a ponto – é responsável por 90% das junções de componentes nas carrocerias (CASTILLO, 2016). É um processo de manufatura complexo que busca impactar minimamente as propriedades do material (chapas metálicas) e maximizar a resistência estrutural. Todavia, com a implementação de novos controles servo-elétricos e servo-pneumáticos, a solda agora tem mais opções de parametrização e monitoramento de dados para materiais de alta resistência. Assim, a digitalização e análise de dados do processo de solda é um passo necessário para sua compreensão, melhorias e controle de qualidade Toyotista.

1.2 Justificativa e Motivação

O fato do processo de solda estar profundamente inserido no setor secundário da economia brasileira justifica o investimento em pesquisas na área. Em outros países, como Espanha, a soldagem pode representar cerca de 40% da produção industrial local

(GONZáLEZ-GONZáLEZ *et al.*, 2023).

Nesse contexto, uma empresa automotiva, alvo deste estudo, utiliza robôs para aplicar pontos de solda na estrutura da carroceria de seus produtos: são 73 robôs produzindo cerca de 100 produtos por dia, com 3000 a 3500 pontos de solda cada um. Seus parâmetros – dentre eles tensão, corrente e resistência aplicadas – são ajustados por um controlador de solda em tempo real durante o processo, de acordo com espessura de chapa e especificações do produto. Ainda assim, podem ser gerados pontos de solda defeituosos, como rebarbas e pontos soltos, ocasionando descartes de componentes e retrabalho. Hoje, esses pontos de solda são inspecionados mais detalhadamente apenas ao final do processo e a olho nu por um colaborador. Além de não ser um método 100% eficiente para detecção de falhas, também afeta o *takt time* da linha de produção, ou seja, o tempo determinado que se deve fabricar um produto baseado em capacidade produtiva, tempo de entrega, demanda, entre outros fatores. Estima-se que em 1 ano, operadores da empresa alvo de estudo encontram 16.259 falhas de solda, causando um retrabalho de 17 segundos por produto e equivalente a 4,5% do *takt time*.

Logo, baseando-se no conceito do Toyotismo de inspeções de qualidade durante o processo, o objetivo inicial desse trabalho é desenvolver um algoritmo capaz de identificar pontos de solda defeituosos utilizando dados do controlador. Uma inteligência artificial com aprendizado supervisionado pode melhorar a produtividade do processo de solda com decisões baseadas em dados, uma vez que busca aprender os padrões e relações relevantes a partir de exemplos, para então criar regras e predições para o negócio (JANIESCH; ZSCHECH; HEINRICH, 2021). No caso de estudo, o modelo teve um grande volume de dados de produção automobilística disponíveis para treinamento – cerca de 50 milhões de pontos de solda, equivalentes a 15 GB de armazenamento e 6 meses de histórico –, classificados previamente entre defeitos ou pontos normais por colaboradores envolvidos no processo. Com um modelo treinado e avaliado, este poderá ser implementado na borda de linha para detectar novos pontos de solda com defeitos em tempo real e então, alarmar um operador no posto de inspeção de qualidade. Nesse sentido, a eficiência e acurácia de investigação do produto foram elevadas, reduzindo o tempo de ciclo de trabalho do ser humano, bem como o risco de não detectar anomalias apropriadamente e suas perdas financeiras.

Esse trabalho visa continuar a modernizar a indústria automotiva – a exemplo do avanço da robótica nas últimas décadas (MAKEDON; MYKHAILENKO; VAZOV, 2021) – visto sua relevância econômica no Brasil, melhorando e otimizando as inspeções de qualidade do processo de soldagem com digitalização e predição de dados.

1.3 Questões de Pesquisa e Objetivos

Este trabalho busca desenvolver um modelo de predição e classificação de pontos de solda com aprendizado supervisionado em grande volume de dados. O modelo deve ser viável para aplicação industrial, mantendo um ecossistema de digitalização de Big Data sustentável para armazenamento e tratamento dos dados da empresa. Algumas questões de pesquisa foram formuladas imaginando as adversidades dessa missão:

Q1 *"É possível desenvolver um algoritmo capaz de identificar pontos de solda defeituosos, treinando uma rede neural a partir de um histórico de dados gerados pelo controlador de solda da empresa?*

Q2 *"É possível criar um ecossistema de Big Data sustentável para a empresa armazenar e tratar esses dados?*

Q3 *"Quanto será possível otimizar e melhorar as inspeções de qualidade de solda?*

Observando essas questões de pesquisa, são definidos alguns objetivos específicos para o desenvolvimento deste trabalho:

a) Classificar automaticamente cada ponto de solda a partir dos dados gerados pelo controlador. Esse objetivo está relacionado à questão de pesquisa Q1;

b) Desenvolver algoritmo para predição em tempo real na borda de linha. Esse objetivo está relacionado à questão de pesquisa Q1;

c) Desenvolver um ecossistema de dados sustentável para longo prazo. Esse objetivo está relacionado à questão de pesquisa Q2.

d) Padronizar um desvio de solda e tendências. Esse objetivo está relacionado à questão de pesquisa Q3;

e) Suportar o time de qualidade, reduzindo o tempo de ciclo das inspeções de qualidade e melhorando sua acurácia. Esse objetivo está relacionado à questão de pesquisa Q3;

1.4 Organização do Trabalho

Este trabalho está dividido em 4 capítulos, sendo o Capítulo 1 uma breve introdução ao assunto. O Capítulo 2 apresenta a revisão bibliográfica utilizada para compreensão do tema, o Capítulo 3 aborda o tratamento dos dados *Big Data* de pontos de solda, bem como a relevância de cada parâmetro de solda e desenvolvimento do modelo de classificação propriamente dito, comparando algoritmos diferentes. No Capítulo 4 são apresentados os resultados, e finalmente no Capítulo 5 a conclusão do trabalho.

2 REVISÃO BIBLIOGRÁFICA

No processo de solda a ponto – ou solda por resistência elétrica –, uma chapa metálica é posicionada entre dois eletrôdos de cobre (material de alta condutividade), os quais exercem força sobre o objeto. O calor gerado pela passagem de corrente entre os eletrôdos funde o metal, resultando em um ponto de solda (ou *nugget*) e unindo as peças do ensaio. Após a aplicação de corrente, geralmente o *nugget* é resfriado pela passagem de água dentro dos eletrôdos, ainda pressionando a chapa metálica (JOHNSON *et al.*, 2023). A energia do processo de solda pode ser equacionada como:

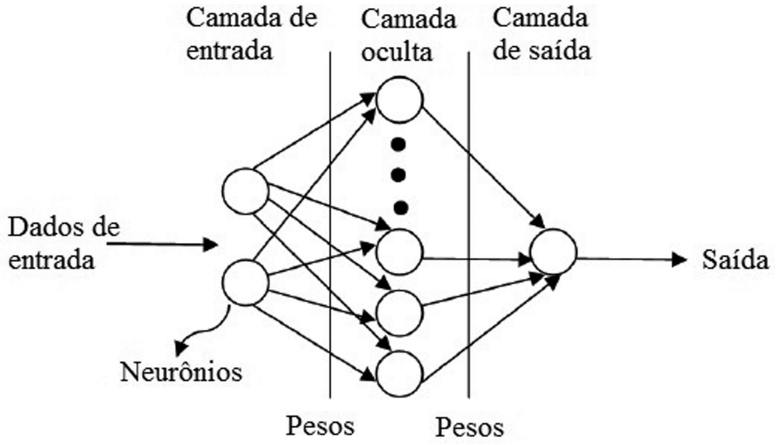
$$Q = \eta \int_0^t I^2(t)R(t)dt, \tag{2.1}$$

sendo Q o calor (Joules), η a eficiência térmica de soldagem, I a corrente aplicada (A), R a resistência do metal (Ω) e t o tempo de soldagem (s). Embora a expressão já seja um indicativo dos principais parâmetros para serem considerados na construção de uma rede neural, também é preciso estudar arquiteturas e metodologias mais adequadas que direcionem para uma otimização de parâmetros e modelo para classificação dos pontos de solda.

2.1 Redes Neurais

Com estudos datados desde a década de 1940, hoje Redes Neurais Artificiais (ANN, *Artificial Neural Network*) já são definidas – conforme sugere o nome – como uma tentativa de simulação do processamento do cérebro humano, associando dados da memória a novas informações. Cada rede é construída por inúmeras camadas de nós (ou neurônios): cada nó funciona como saída da camada anterior e entrada da próxima, conectados por sinais representados por pesos para cada escolha ou cálculo da estrutura, como representado na Figura 1. O resultado de cada rede neural representa uma lógica desenvolvida por uma memória de dados e varia de acordo com os dados de entrada, funções e pesos utilizados (WU; FENG, 2017). Essa capacidade de adaptação e imitação do pensamento humano é o que torna as Redes Neurais Artificiais cada vez mais utilizadas para a solução de problemas matemáticos complexos, como classificação de falhas na indústria. McCulloch and Pitts (1943) foram um dos primeiros a pesquisar redes neurais, propondo o modelo computacional M-P, o primeiro imitando um neurônio biológico. Em seguida, o psicólogo Hebb (1949) sugeriu que a intensidade das conexões das sinapses são variáveis, indicando o caminho para a compreensão do aprendizado e memória biológica. Quase 10 anos depois, Rosenblatt (1958) propôs uma melhoria para o modelo M-P com o conceito de perceptron. Nesse momento, a arquitetura de uma ANN já estava mais próxima do que se conhece hoje, conseguindo até mesmo reconhecer e classificar padrões.

Figura 1 – Representação básica de uma arquitetura de Redes Neurais Artificiais.



Fonte: Extraído de (BARROS *et al.*, 2020).

Os engenheiros americanos Widrow and Hoff (1960) propuseram a primeira rede neural para soluções práticas em 1960, com o método de treinamento de Elemento Adaptativo Linear (ADALINE, *Adaptive Linear Element*). Já Minsky and Papert (1969) publicaram o livro "Perceptrons", no qual indicaram as primeiras limitações das redes neurais artificiais com perceptrons por não serem capazes de solucionar a categorização de dois tipos de amostras não separáveis linearmente. Já o finlandês Kohonen (1982) apresentou pouco tempo depois o modelo Mapa de Recursos Auto-Organizável (SOM, *Self-Organizing Feature Map*) ou simplesmente Mapas de Kohonen, um tipo de aprendizado não-supervisionado utilizado para reconhecimento de padrões, texto e classificação (WU; FENG, 2017).

Ackley, Hinton and Sejnowski (1985) contribuem com os estudos em ANN com o primeiro algoritmo de rede multi-camada, conhecido como modelo estocástico Máquina de Boltzmann, também um aprendizado não-supervisionado. Chauvin and Rumelhart (1995) então complementou a pesquisa com o conceito de algoritmo de Rede de Retropropagação (BPN, *Backpropagation Network*), ajudando a solucionar o erro de autocorreção dos pesos das redes neurais de multi-camada. Finalmente, Hinton, Osindero and Teh (2006) usaram o termo Aprendizado Profundo (DL, *Deep Learning*) como um novo campo de estudos para desenvolver modelos com arquiteturas mais complexas, explorando múltiplas camadas ocultas, funções e treinamento de dados em larga escala, permitindo ao usuário modelar o algoritmo de acordo com a aplicação em questão (WU; FENG, 2017). Tais evoluções descritas nessa seção contribuiram para o desenvolvimento das Redes Neurais Artificiais e suas aplicações práticas.

2.2 Formas de Aprendizado Indutivo

Segundo Wu and Feng (2017), o reconhecimento de padrão é um processo de descrição, identificação, classificação e interpretação das coisas ou fenômenos. É necessário processar e analisar diversas formas de informação que caracterizam coisas ou fenômenos.

Dessa forma, ao se tratar de classificação de dados de solda com inteligência artificial, é imprescindível a compreensão da natureza do problema que se deseja solucionar, bem como a escolha das variáveis relacionadas e seu armazenamento a longo prazo. O volume de dados requeridos para uma predição dependerá, por exemplo, da proposta do estudo e do algoritmo utilizado. Ainda, a catalogação prévia dos dados de treinamento é necessária para modelos triviais de classificação, mas podem demandar muita carga horária de trabalho para devidamente rotular os dados antes do treinamento, além de dependerem de uma mão de obra qualificada, especialista na área de solda para compreender a natureza dos dados e classificá-los em massa. Logo, é válido um estudo sobre tipos de aprendizados indutivos.

A respeito de inteligência artificial, a literatura divide o aprendizado indutivo em dois tipos: supervisionado e não-supervisionado. O aprendizado supervisionado parte do pressuposto que todas as classes do problema já são conhecidas, assim como os dados para treinamento do modelo já estão rotulados de acordo. Sendo assim, o modelo classificador irá rotular novas informações baseado no padrão de dados encontrado no histórico.

Em contrapartida, no método de aprendizado não-supervisionado o objetivo é agrupar a amostra baseado em um padrão. Neste caso, os dados não precisam estar previamente classificados por um especialista, porém espera-se que o usuário conheça a natureza dos dados e saiba o número esperado de grupos que devem ser corretamente separados. Também denominados de algoritmos de clusterização, alguns exemplos de modelos de aprendizado não-supervisionado são K-Means, Ligação (*Linkage*), detecção de anomalias, entre outros.

Dito isso, neste trabalho é feita então uma comparação de desempenho e metodologias entre aprendizado supervisionado e de aprendizado não-supervisionado, bem como seu comportamento como modelos de classificação de desvios.

2.3 Principais Algoritmos para Classificação de Dados

Na literatura são encontrados diversas pesquisas de *Machine Learning* e *Deep Learning* para classificação e, antes de entrar no estado da arte, primeiramente aqui destacam-se alguns dos algoritmos: Máquina de Vetores de Suporte (SVM, *Support Vector Machine*), Perceptron de Multicamadas (MLP, *Multilayer Perceptron*), BPN e clusterização particional. Em segundo plano, apresentam-se como algoritmos alternativos o CATBoost e clusterização hierárquica, visto as poucas menções no estado da arte.

Conforme mencionado na Seção 2.1, Rosenblatt foi responsável por introduzir o conceito de *perceptron* em Redes Neurais Artificiais. Além da combinação de uma camada de entrada com pesos para as variáveis, Rosenblatt adicionou um limite – *threshold* – à somatória dos produtos de entradas e pesos, controlando a entrada e a saída da ANN:

com uma função de ativação – habitualmente sigmoide ou Unidade Linear de Retificada (ReLU, *Rectified Linear Unit*) –, um *perceptron* é capaz de realizar uma classificação binária (BENTO, 2021). Contudo, também foi dito que Minsky e Papert demonstraram que o modelo de *perceptron* não era capaz de solucionar problemas com dados não-lineares, levando ao desenvolvimento do MLP. O modelo de Perceptron de Multicamadas apresenta inúmeras camadas ocultas com neurônios alinhados (Figura 1), os quais por sua vez podem utilizar qualquer função de ativação, não mais restritos a sigmoide ou ReLU. A ideia do modelo MLP em propagar o resultado de uma camada para a seguinte caracteriza-o como uma Rede Neural de Propagação Direta (*Feedforward Neural Network*). Nasir *et al.* (2022) a resume da seguinte maneira:

$$Y_k = f_k(\Sigma_{i=1}^{m} W_{kj} \times f_j(\Sigma_{i=1}^{n} X_i W_{ij})) + W_0, \tag{2.2}$$

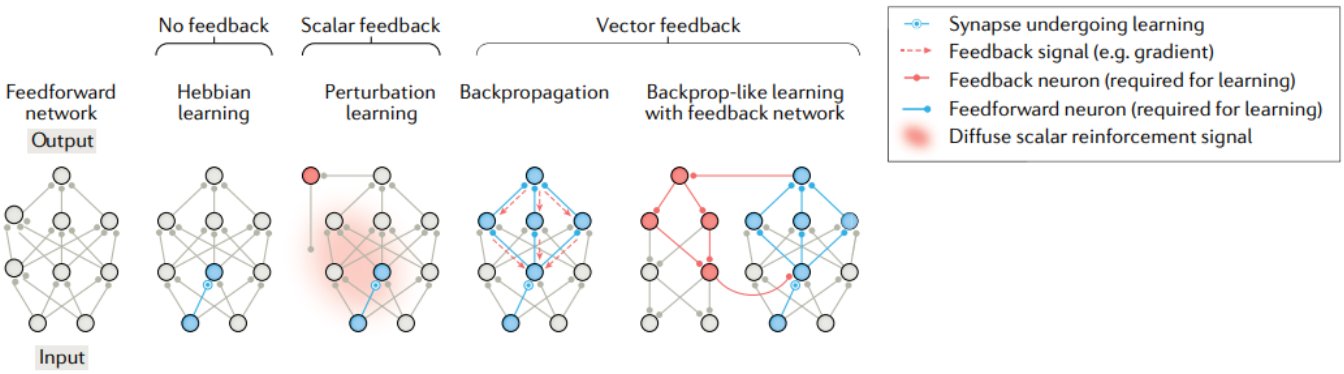
onde n é o número de parâmetros, m a quantidade de neurônios na camada oculta, k neurônios da camada de saída, W_0 o *bias*. W_{jk} e W_{ij} são os pesos entre o i-ésimo, j-ésimo e k-ésimo neurônios, enquanto f_k e f_j são as funções de transferência dos neurônios da camada de saída k e camada oculta j, respectivamente.

Já a BPN, apresentada por David Rumelhart (Seção 2.1), traz uma evolução em relação a um MLP comum de propagação direta: o algoritmo tem como princípio o "aprendizado" com os erros durante o treinamento de dados. Lillicrap *et al.* (2020) diz que uma Retropropagação utiliza a regra da cadeia de cálculo para processar como a mudança em cada sinapse mudaria o erro final da rede neural. O nome Retropropagação se dá devido ao sinal de erro de uma camada ser calculado dos sinais de erro da camada anterior, levando a ideia de propagação "contrária" do erro através da rede. O erro final da BPN é reduzido depois que os sinais de todos os neurônios forem processados, atualizando seus pesos em direção ao valor retornado. Vale mencionar que graças ao uso da regra da cadeia, mesmo computando os sinais de erro para todas as sinapses da arquitetura, o cálculo da BPN ainda é mais veloz do que de uma ANN de propagação direta de multicamadas (LILLICRAP *et al.*, 2020). A Equação 2.3 expressa a atualização de um peso W_{ij} dado W_{jk} entre os neurônios i, j e k. Já a_j é a atividade de um neurônio j e $h_j = f(a_j)$ sua saída, enquanto o termo E é uma função de erro – como erro quadrático – entre a saída da rede y_l e os valores reais t_l.

$$\begin{aligned}
\Delta W_{ij} &= -\eta \frac{\partial E}{\partial W_{ij}} \\
\Delta W_{ij} &= -\eta h_i \delta_j \\
\Delta W_{ij} &= -\eta h_i e_j f\,'(a_j) \\
\Delta W_{ij} &= -\eta h_i (\Sigma_k \delta_k W_{jk}) f\,'(a_j)
\end{aligned} \tag{2.3}$$

O sinal de erro δ_j é calculado na camada seguinte por δ_k, até que na camada de saída da BPN sabe-se que $\delta_l = y_l - t_l$, provando matematicamente a retropropagação

Figura 2 – Perspectiva de algoritmos de Redes Neurais Artificiais.

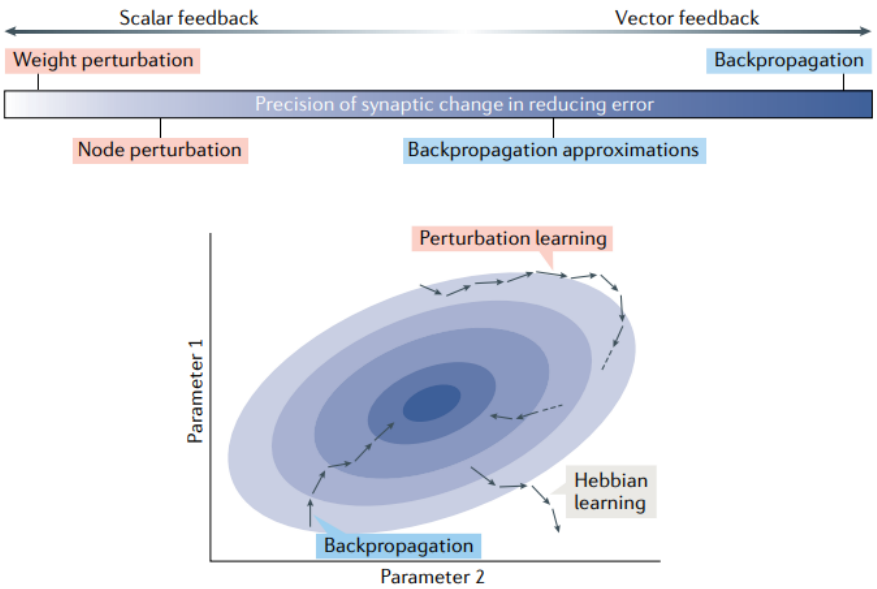


Fonte: Adaptado de (LILLICRAP *et al.*, 2020).

da arquitetura. Embora, algoritmos BPN sejam conhecidos por atuarem em aprendizado supervisionado, observa-se que o cálculo do erro não depende de uma amostra de dados rotulado. Portanto, também é eficiente para aprendizados por reforço ou aprendizados não supervisionados. A Figura 2 é um espectro comparativo entre algumas arquiteturas de redes neurais e sua influência na camada de saída. Começando à esquerda por uma rede de propagação direta – como uma MLP simples –, evoluindo para um Aprendizado Hebbiano, no qual uma única atualização na sinapse azul não causa uma mudança significativa na saída do sistema. Então, observa-se o aprendizado por perturbações, um método randômico de retroalimentação escalar para controlar a atualização dos pesos, seguido pela simbolização de uma BPN com atualização dos pesos. Além disso, o espectro da Figura 3 apresenta a diferença entre métodos de retroalimentações escalares e vetoriais – como a BPN –, implicando na velocidade de aprendizado de cada um. O primeiro causa perturbações aleatórias em todas as conexões simultaneamente, criando ruídos e levando várias tentativas até determinar qual alteração deve ser aprovada como novo peso, enquanto a retropropagação possui um vetor que conduz a um aprendizado mais efetivo e rápido.

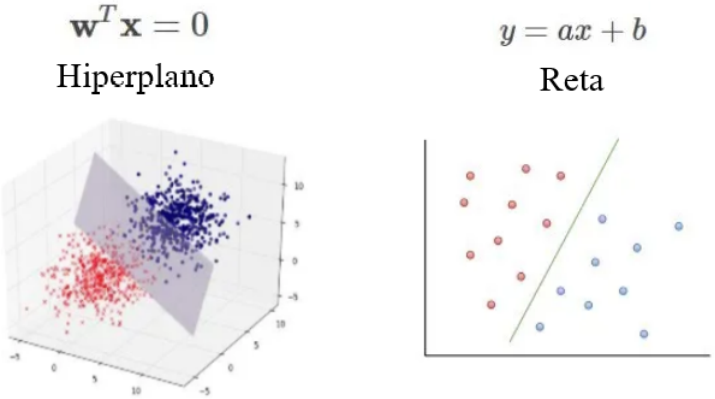
O modelo MLP é constituído por uma arquitetura complexa, possibilitando múltiplas entradas e saídas. Todavia, também pode apresentar diversos mínimos locais, o que dificulta a otimização dos pesos. Não obstante, um modelo mais simples de *Machine Learning* SVM foi introduzido por Boser, Guyon and Vapnik (1992) em sua pesquisa *"A Training Algorithm For Optimal Margin Classifiers"*. Nela, SVM é definido como um método de aprendizado supervisionado para classificação e regressão, utilizando um hiperplano de funções lineares em um espaço hiperdimensional e estatística para minimizar os erros (JAKKULA, 2006), contornando a dificuldade com os mínimos locais de um MLP. A Figura 4 demonstra a divisão de um *dataset* por meio de um hiperplano, bem como sua fórmula: um SVM permite encontrar diversos hiperplanos como esse, o desafio é definir qual se enquadra melhor para a classificação dos dados, uma vez que apenas um alcança a máxima separação.

Figura 3 – Espectro de algoritmos de Redes Neurais Artificiais.



Fonte: Adaptado de (LILLICRAP *et al.*, 2020).

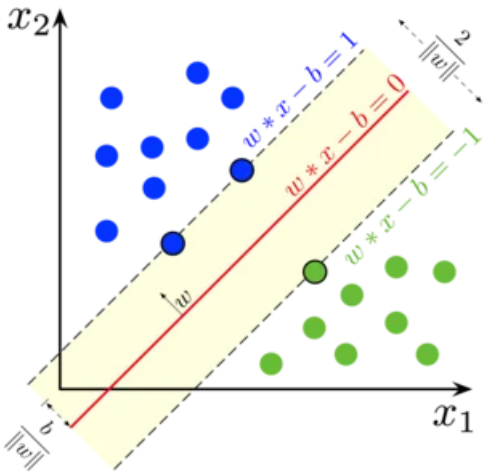
Figura 4 – Comparativo entre as fórmulas de um hiperplano e uma reta classificadores.



Fonte: Adaptado de (COUTINHO, 2019).

Para isso, é usado um Classificador de Margem Máxima: o algoritmo calcula as distâncias perpendiculares entre os pontos de treinamento até os hiperplanos encontrados, sendo a maior distância – ou margem – pertencente ao melhor hiperplano de separação. Tais pontos de cada classe de maiores margens e que "empurram" o hiperplano para longe são chamados vetores de suporte, os quais dão nome ao SVM (JAKKULA, 2006). Dado o exemplo mais simples de um SVM em duas dimensões na Figura 5, observa-se que os vetores de suporte das classes azul e verde determinam as margens $wx - b = 1$ e $wx - b = -1$, respectivamente, sendo x um ponto vetorial, w um vetor peso e b o coeficiente linear. A classe azul portanto pode ser determinada com a expressão $wx + b \geq 1$, enquanto a verde por $wx + b \leq -1$ (COUTINHO, 2019). Logo, o hiperplano ideal do SVM atende a

Figura 5 – Hiperplano de classificação e seus vetores de suporte que determinam as margens.



Fonte: Extraído de (COUTINHO, 2019).

inequação

$$y_i(wx_i + b) \geq 1, \tag{2.4}$$

para todos os pontos i, valores reais y e que w seja o menor de acordo com a equação da margem máxima:

$$f = \underset{x \in D}{argmin} d(x) = \underset{x \in D}{argmin} \frac{|xw + b|}{\sqrt{\Sigma_{i=1}^{d} w_i^2}}, \tag{2.5}$$

Ainda segundo Jakkula (2006), a margem máxima traz melhor desempenho, evita mínimos locais e melhora a classificação dos dados. A busca do SVM em reduzir os erros de predição é demonstrada por:

$$f = \int V(y, f(x))P(x, y)dxdy, \tag{2.6}$$

onde $(x_1, y_1)...(x_n, y_n)$ no domínio $\mathbb{R}^n \times \mathbb{R}$ é uma amostra de dados de treinamento de acordo com uma distribuição probabilística $P(x, y)$, função de perda $V(y, f(x))$ para um dado x, e valor $f(x)$ previsto em relação ao valor real y. Dessa forma, de acordo com Nasir *et al.* (2022) um algoritmo SVM é eficiente para dados de múltiplas dimensões e poucos exemplos, mas treinar um SVM para amostras grandes torna-se um problema do ponto de vista de processamento.

CATBoost é um algoritmo proposto em 2016 por Anna Dorogush na Yandex o qual, segundo Jabeur *et al.* (2021), performa bem para *datasets* pequenos com parâmetros categóricos com pouca perda de informação, utilizando impulso ordenado (do inglês, *ordered boosting*) para superar problemas recorrentes com vazamento de alvo. *Target leakage*, como é conhecido, acontece quando variáveis disponíveis no treinamento não se aplicam na predição, resultando em predições supervalorizadas ao serem implementadas no ambiente de produção (LARSEN; BECKER, 2019). Sendo muito aplicado em modelos financeiros e

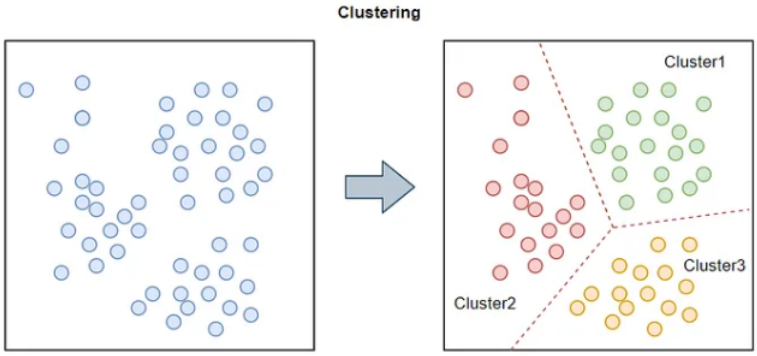
séries temporais, ainda segundo Jabeur *et al.* (2021) outra característica do CATBoost são suas permutações randômicas, a fim de estimar os valores das folhas de sua árvore de decisão e evitar *overfitting*. O modelo então retorna árvores de decisão binárias de acordo com a equação 2.7:

$$Z = H(x_i) = \sum_{j=1}^{J} C_j^1, \tag{2.7}$$

sendo $x \in R_j$, $H(x_i)$ uma função de árvore de decisão das variáveis explicativas x_i e R_j a região correspondente às folhas da árvore de decisão.

Esta seção também estuda a clusterização, um método de aprendizado não supervisionado que pode ser do tipo hierárquico ou particional. Em seu trabalho, Ezugwu *et al.* (2022) diz que ambos dividem um *dataset* em classes (ou *clusters*) no espaço amostral, com o viés de depender do usuário inserir a quantidade de classes que o algoritmo deve procurar. Dessa forma, a clusterização não requer rotulação de dados para classificação, mas sim de um pouco de informação a respeito da natureza dos dados e o objetivo do algoritmo. Os autores ainda mencionam o uso do método para análises de *Big Data*.

Figura 6 – Exemplo de clusterização.



Fonte: Extraído de (AZANK, 2022).

A clusterização hierárquica formata os dados em níveis, com abordagens aglomerativas de baixo para cima e de cima para baixo na hierarquia como abordagem divisiva. Os principais exemplos de clusterização hierárquica são os algoritmos de Ligação Simples (*Single Linkage*), Ligação Média (*Average Linkage*) e Ligação Completa (*Complete Linkage*), os quais basicamente diferenciam-se no modo de calcular a distância entre pontos e subconjuntos de pontos para formar os *clusters* (EZUGWU *et al.*, 2022). Apesar de permitir visualizar diversos níveis de estrutura de classes em um dendograma – ou diagramas de árvores –, sempre é necessário indicar previamente a quantidade *clusters* na amostra (LOPEZ *et al.*, 2018).

Por outro lado, a clusterização particional não apresenta uma estrutura de dendrograma aos dados de treinamento. Ezugwu *et al.* (2022) argumenta então que o método é útil quando o processamento computacional é limitado, há um grande volume de dados ou

reconhecimento de padrões. Esse algoritmo portanto recebe esse nome pois particiona o conjunto de dados em diversos subconjuntos, conforme pré-definido por uma função de otimização, como critério de erro quadrático. A clusterização particional visa encontrar a partição que minimiza tal erro, representada pelo valor mínimo da função de custo alvo ζ na equação:

$$\zeta = \Sigma_{i=1}^{n} ||d_i - C_j||^q, \tag{2.8}$$

na qual C_j é o centróide de um cluster j, assim como o centróide mais próximo de um ponto amostral d_i. Já o elemento n indica a quantidade de variáveis na clusterização e q o método para cálculo da distância. Um dos algoritmos mais famosos de clusterização particional é o *K-Means*, o qual escolhe arbitrariamente k pontos para representarem os centróides de k *clusters*, aplicando a distância Euclidiana da Equação 2.8 nos pontos restantes e retornando o valor para o *cluster* de menor distância. O processo é repetido para cada ponto novo inserido da amostra de treinamento, recalculando a distância média e reavaliando a posição das classes até obter um valor estável. Segundo Ezugwu *et al.* (2022), uma desvantagem do *K-Means* é o fato de ser sensível à escolha inicial do centróide e *outliers*. Além disso, não garante a convergência para o mínimo global e só pode ser aplicado em dados numéricos por usar a média como métrica.

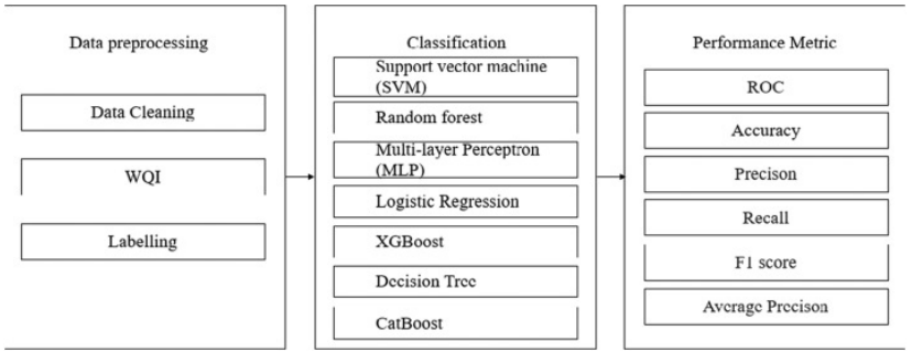
Entre diversos algoritmos conhecidos para classificação de dados, o MLP, BPN, SVM e clusterização foram estudados a fundo a fim de compreender como diferenciam-se em relação ao aprendizado indutivo, consumo de dados para treinamento, cálculo matemático e processamento. Então, a seguir é tratado o estado da arte, com o propósito de estudar quais algoritmos são apropriados para este trabalho, baseando-se também em seus desempenhos.

2.4 Trabalhos Relacionados

Nessa seção são apresentados trabalhos relacionados ao tema como embasamento teórico para a metodologia desenvolvida a seguir. Antes de procurar classificar falhas especificamente em pontos de solda, foram estudados alguns algoritmos comuns na literatura para classificação de falhas no geral.

Embora menos utilizados na literatura, os algoritmos de aprendizado não supervisionado permitem afastar-se dos problemas de conhecimento *a priori* sobre os dados em questão. No aprendizado supervisionado, por exemplo, há necessidade de classificação prévia do histórico de dados de anomalias, uma atividade trabalhosa em processos industriais ou problemas de *Big Data*. Portanto, Lopez *et al.* (2018) propõe o aprendizado não supervisionado como técnica para mapear doenças genéticas em pessoas, utilizando amostras sem rótulos de DNA para esclerose múltipla, leucemia, entre outras doenças. O trabalho baseou-se no conceito de Polimorfismo de Nucleotídeo Único (SNP, *Single-Nucleotide Polymorphism*), isto é, o conceito de alterações genéticas em apenas uma sequência do genoma

Figura 7 – Métodos utilizados para pré-processamento, algoritmos e métricas de avaliação para estudo de qualidade da água.



Fonte: Extraído de (NASIR *et al.*, 2022)

para evitar redundâncias nas amostras de dados. Com um *linkage desequilibrium* como etapa de redução das propriedades, foram aplicados diversos algoritmos de clusterização hierárquica, tal qual Ligação Simples, Ligação Completa, Ligação Média, Ligação de Ward e Ligação de McQuitt. No passo seguinte foi implementada uma validação métrica de método de silhueta para otimizar a quantidade de *clusters*, a serem identificados em cada amostra e método *linkage* diferente. Dessa forma, uma validação cruzada em 10 pastas provou com sucesso uma acurácia de 96,33%. Um índice Rand de 0,969 também identifica que o modelo não sofreu *overfitting*.

A seguir, são apresentados apenas algoritmos de aprendizado supervisionado.

Nasir *et al.* (2022) preocupa-se em relacionar atributos da água – oxigênio dissolvido, pH, coliformes fecais, entre outros – à sua qualidade utilizando diversos métodos de *Machine Learning*. Com esse propósito, foram extraídos dados de diversas amostras de água, pré-processados e rotulados de "muito poluída" a "limpa" utilizando o Índice de Qualidade da Água (WQI, *Water Quality Index*). O artigo detalha as arquiteturas utilizadas para os modelos. Em ordem crescente de acurácia obtida, são eles: Regressão Logística, SVM, Árvore de Decisão (DT, *Decision Tree*), XGBoost (*Extreme Gradiente Boosting*), MLP, Random Forest e CATBoost (*Categorical Boosting*). Os quatro últimos destacaram-se pelo desempenho, em especial o CATBoost com acurácia de 94,51%, precisão de 94,58%, *recall* de 94,51% e 94,49% de F1-score. Contudo, os autores também revelam algumas particularidades de cada algoritmo, como a capacidade de resolver problemas não-lineares, complexos e de grande volume de dados com o MLP, ou mesmo a tendência do CATBoost sofrer *overfitting*. Consulte a Tabela 1 para mais detalhes.

Ainda sobre problemas gerais de classificação, Khokhar *et al.* (2015) estudam a qualidade de energia fornecida para a indústria, a fim de monitorar distúrbios na rede elétrica que podem ocasionar falhas de equipamentos. Nesse trabalho, diversas metodologias da literatura são comparadas. Após a extração dos dados de energia, as variáveis são selecionadas por técnicas de processamento de sinais no domínio do tempo ou frequência

como Transformada de Fourier, Transformada de Wavelet e Transformada de Stockwell. Então é aplicada inteligência artificial para classificar padrões de distúrbios na energia fornecida como redes neurais BPN, Rede Neural Probabilística (PNN, *Probabilistic Neural Network*), SVM, sistema de inferência *neuro-fuzzy* adaptativo (ANFIS, *Artificial Neural Fuzzy Inference System*) e KNN (*k-Nearest Neighbour*). A terceira e última etapa proposta é a implementação de algoritmos para otimização de indicadores e parâmetros do modelo, frutos da Ciência da Biologia: algoritmo genético (GA, *Genetic Algorithm*), Otimização de Colônia de Formigas (ACO, *Ant Colony Optimization*) e Otimização por Enxame de Partículas (PSO, *Particle Swarm Optimization*). Alguns dos modelos com melhor desempenho foram ilustrados na Tabela 1, destacando-se a Transformada de Stockwell para processamento de sinais, classificadores DT e SVM, e otimizador GA. Todos com resultados para dados sem ruído e com ruídos próximos a 99% e 97%, respectivamente.

Convergindo para os estudos sobre solda após classificação de dados mais abrangentes, também foram encontrados alguns trabalhos na literatura relacionados a classificação de solda por processo MIG/MAG.

Florence, Samsingh and Babureddy (2018), por exemplo, relatam os problemas que a microestrutura e propriedades de uma chapa metálica sofre com cordões de solda – não pontos –, uma vez que torna-se uma zona de alto estresse residual e propensa a falhas da junta. O estudo foca especialmente em falta de fusão da parede lateral, falta de penetração da solda, porosidade da solda e inclusão de escória, tendo como base ensaios de ultrassom para classificação. Dois tipos de redes neurais artificiais são levados em consideração para classificação: BPN – a qual usa ambos *forward* e *back propagation* – e PNN. A rede neural é então alimentada por amostras dos dados de teste de ultrassom para ser treinada e identificar os quatro tipos de defeitos das juntas de solda. A arquitetura BPN atingiu uma eficiência de 91,7%, um número entendido como suficientemente alto para implementar o algoritmo em situações reais do processo. Dessa forma, mesmo análises precisas de testes de ultrassom podem ser substituídas pelo modelo, uma vez que não requer um operador altamente qualificado ou elevado tempo de atividade.

Além de Florence, Ho *et al.* (2022) estuda um processo de solda GMAW (*Gas Metal Arc Welding*) – também um processo de solda MAG –, mas dessa vez parametrizando o gás de proteção do arco de solda. O foco do trabalho foi a classificação em cordões de solda sem defeitos visualmente aparentes, com pouco preenchimento, com respingos ou com superfície porosa. As amostras foram classificadas com um ultrassom de feixe angular, um *scanner* a laser e observações a olho nu, em seções de 10 mm e um total de 22 imagens, incluindo capturas pelos lados direito e esquerdo do cordão de solda. Assim, cada seção corresponde a um vetor de dimensão $[1 \times 13]$, sendo 11 valores de corrente elétrica medida a cada 0,1s, 1 de velocidade do cordão e 1 de fluxo do gás, enquanto a saída é uma classificação binária para os defeitos mencionados. A rede neural artificial

desenvolvida consiste em uma camada de entrada, uma camada oculta e uma de saída. O tamanho do *batch* e épocas foram declarados como 64 e 1200, respectivamente, atingindo no máximo 750 épocas no treinamento quando não demonstrou melhora e preveniu um *overfitting*. Contudo, para o modelo não ser enviesado pela corrente elétrica apenas, sua arquitetura apresenta 2 estágios. O primeiro normaliza os 3 parâmetros para valores entre 0 e 1 e tornam-se 2 entradas separadas da camada oculta: a corrente é o "lado direito", enquanto o fluxo do gás e velocidade da solda são a entrada do "lado esquerdo" do modelo. Já o segundo estágio concatena os resultados ao final da camada oculta. Nessa camada, os valores passam por uma função ReLU, função de ativação não-linear para suportar muitos neurônios de entrada com processamento e acurácia elevados. Então, há uma função tangente hiperbólica para aprender rapidamente uma faixa considerável de valores, seguida por uma função sigmoide para um método do gradiente estocástico (SGD, *Stochastic Gradient Descent*) para encontrar os valores mínimos locais, enquanto a função de perda escolhida foi a entropia-cruzada binária para classificação de 0 (solda com defeito) ou 1 (sem defeito). Finalmente, o modelo apresentou acurácia de 85,589%, especificidade de 77,193% e sensitividade (ou *recall*) de 89,326%. O autor também desenvolveu uma interface gráfica para um operador saber a melhor configuração das variáveis a partir de apenas uma delas inserida no *software*.

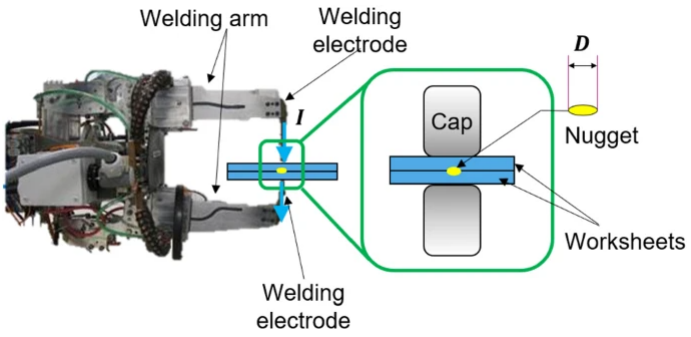
Figura 8 – Exemplos de cordão de solda GMAW: a) sem defeitos, b) sem preenchimento, c) solda com respingos, d) solda com porosidade.

Fonte: Adaptado de (HO *et al.*, 2022).

Já em relação a pontos de solda, Zhou *et al.* (2022) apresenta uma proposta diferente, enxergando a soldagem com o passar do tempo e a relação com fatores externos ao invés de cada ponto de solda como um evento individual. Como fatores externos inclui-se ciclos de afiação dos eletrodos – vide Figura 9 –, espessuras de chapa, modelos de carrocerias do produto, programas de solda pré-determinados, entre outros. Consequentemente, os autores analisam o conhecimento de engenharia e ciclos de manutenção dos controladores de solda Bosch, além das variáveis do processo de solda propriamente dito. Por essa razão o objetivo do projeto não é classificar os defeitos de solda, mas sim prevê-los antes que aconteçam. O estudo aborda diversas possibilidades de modelos, utilizando 2 *datasets* de ferramentas de solda distintas, 3 modelos de *Machine Learning* (LR, MLP e SVR) e 4 configurações de entrada para o modelo, totalizando 24 treinamentos de modelos diferentes.

Concluiu-se que o modelo de regressão linear, mesmo que considerado mais simples, foi o que apresentou melhores resultados ao abordar os fatores de engenharia, manutenção e séries temporais de solda.

Figura 9 – Representação de um processo de solda.



A ferramenta possui dois eletrodos de cobre, que aplicam determinada força na chapa metálica, pressionando-a. Em seguida, uma corrente elétrica passa pelos eletrodos, fechando o circuito e formando o ponto de solda (*nugget*) de diâmetro D.
Fonte: Extraído de Zhou *et al.* (2022).

Johnson *et al.* (2023), por sua vez, associa a soldagem por ponto a 3 parâmetros em especial, sejam eles corrente elétrica, tempo de solda e a força aplicada no eletrôdo de cobre. Diferente deste trabalho que visa uma classificação binária dos pontos de solda – com defeito ou sem – independente dos possíveis tipos de defeitos encontrados, Johnson *et al.* (2023) busca prever os indicadores de qualidade: diâmetro do ponto (ou *nugget*), força superficial do ponto, resistência de cisalhamento e a resistência de contato dinâmico médio (MDCR, *Mean Dynamic Contact Resistance*). Foi utilizado uma análise de regressão com GA para otimizar os parâmetros de solda de acordo com sua influência na soldagem e assim, atribuir os pesos para os modelos. Segundo o autor, tratando o problema de qualidade de solda como um estudo de regressão, a natureza de dados contínuos é preservada e permite assim uma variedade de informações a respeito do processo. Desenvolvendo simultaneamente um modelo de rede neural artificial e outro de ANFIS, a acurácia de predição dos índices de qualidade chegou a 99,36% e 99,98%, respectivamente, demonstrando resultados muito próximos.

Por fim, You and Kim (2021) apresentam outro foco para os pontos de solda, desta vez também levantando aspectos mecânicos do *nugget*. Os autores preocuparam-se em prever indicadores de qualidade – como diâmetro do ponto de solda e comportamento nos testes de cisalhamento de tração – a partir de materiais de base e informações do processo. Foram desenvolvidos modelos de regressão para determinação do diâmetro e carga de ruptura, apresentando acurácia de 90% e 95%, respectivamente. Já para classificar a localização da falha, 2 arquiteturas foram modeladas: um modelo de 1 etapa que utiliza as informações do material e do processo, e outro modelo de 2 etapas que também utiliza o diâmetro do ponto de solda previamente calculado. No modelo de regressão foi utilizada a

função de ativação ReLU na camada oculta e a função sigmoide na camada de saída. Como funções de perda, You and Kim (2021) utilizaram erro quadrático médio (MSE, *Mean Square Error*), erro cruzado e por fim, função de otimização ADAM (*Adaptive Moment Estimation*) do modelo. Por outro lado, ambos os modelos de classificação do modo de falha possuem função de ativação sigmóide. Todos os hiperparâmetros estão descritos na Tabela 1.

2.5 Considerações finais

Esta revisão bibliográfica tem como intuito pesquisar o estado da arte para classificação com inteligência artificial, a fim de compreender os métodos mais indicados já utilizados na literatura para categorizar pontos de solda com defeitos na indústria. Foi narrado a evolução histórica das Redes Neurais Artificiais, bem como a diferença entre aprendizado supervisionado e não supervisionado: o primeiro tipo foi encontrado mais facilmente em trabalhos sobre classificação, sendo a única exceção aqui a menção ao trabalho de Lopez *et al.* (2018), mostrando grandes resultados com aprendizado não supervisionado de clusterização. Então, estudou-se a fundo alguns dos principais algoritmos, sendo eles as ANN de *Multilayer Perceptron*, *Backpropagation Network* e os algoritmos de *Machine Learning* como *Support Vector Machine* e clusterização hierárquica.

Um resumo dos resultados obtidos dos trabalhos relacionados pode ser encontrado na Tabela 1. Foram analisados desde trabalhos sobre classificação de dados em temas mais abrangentes – como contaminação da água (NASIR *et al.*, 2022), perturbações na rede elétrica de (KHOKHAR *et al.*, 2015) e doenças genéticas (LOPEZ *et al.*, 2018) –, até artigos sobre cordões MIG / MAG (FLORENCE; SAMSINGH; BABUREDDY, 2018; HO *et al.*, 2022), e pontos de solda (ZHOU *et al.*, 2022; JOHNSON *et al.*, 2023; YOU; KIM, 2021). Entre eles, destaca-se a clusterização hierárquica pelo método distinto (LOPEZ *et al.*, 2018). Nasir *et al.* (2022) também traz uma análise de diversos algoritmos de *Machine Learning* intermediários como MLP, apontado pelo autor como eficiente para dados não-lineares todavia de processamento demorado, enquanto o SVM tem características opostas. Além disso, Khokhar *et al.* (2015) revisou diversos modelos para um tema diferente, atingindo excelentes resultados ao inserir cálculos para extratores de características e otimizadores em dados sem e com ruídos, a exemplo de SVM com Transformada Discreta de Wavelet ou Transformada-S Hiperbólica. Florence, Samsingh and Babureddy (2018) consegue bons resultados com BPN, e Johnson *et al.* (2023) pode ter atingindo a melhor acurácia, porém não descreve a rede neural utilizada para classificar os indicadores de qualidade de um ponto de solda. Entre tantos métodos do estado da arte, é preciso escolher os mais apropriados para seguir com a análise deste trabalho.

Finalmente, deve ser levado em consideração o perfil dos dados deste trabalho. Há um grande volume de dados disponíveis para análise, referentes aos últimos 6 meses de solda

de produtos na indústria. Contudo, a rotulação dos mesmos é um grande obstáculo e requer um especialista em qualidade de solda para o trabalho de inspeção, conforme mencionado na Seção 2.2. Por esse motivo, o primeiro algoritmo escolhido foi de clusterização particional *K-Means* – diferente do hierárquico utilizado por Lopez *et al.* (2018) – pois é um volume grande de dados não-lineares e não é preciso um dendrograma. Além disso, também foram escolhidos SVM e o MLP com BPN, pois foram dos mais citados pelos autores – 3 e 2 vezes, respectivamente – e apresentam comportamentos distintos para a quantidade, natureza dos dados e ruídos. Denominados neste trabalho como modelos alternativos por serem pouco utilizados nos trabalhos relacionados, também foram incluídos na análise os modelos de clusterização hierárquica e CATBoost. Logo, a Metodologia (Capítulo 3) trata do comparativo entre 3 principais modelos – *K-Means*, SVM e MLP com BPN – e mais 2 ditos como alternativos – clusterização hierárquica e CATBoost –, totalizando 5 modelos.

Tabela 1 – Arquiteturas de trabalhos relacionados

Referência	Estudo	Modelos	Acurácia	Comentários
(LOPEZ *et al.*, 2018)	Doenças genéticas	Clusterização hierárquica	96,33%	Aprendizado Não Supervisionado
(NASIR *et al.*, 2022)	Qualidade da água	Regressão Logística	72,91%	
		SVM	80,68%	
		DT	81,62%	
		XGBoost	88,07%	
		MLP	88,63%	
		Random Forest	93,93%	
		CATBoost	94,51%	
(KHOKHAR *et al.*, 2015)	Qualidade de energia	PNN / SVM	95,97%	Dados sem ruídos
		SVM	99,71% - 96,86%	Dados sem / com ruídos
		LMT	99,11% - 97,79%	Dados sem / com ruídos
		PNN	96,3% - 96,1%	Dados sem / com ruídos
		FCM	95,97%	Dados sem ruídos
		SVM / DT	99,5% - 96,1%	Dados sem / com ruídos
(FLORENCE; SAMSINGH; BABU-REDDY, 2018)	Cordão de solda (classificação)	BPN	91,7%	Uso de ultrassom
		PNN	...	Uso de ultrassom
(HO *et al.*, 2022)	Cordão de solda (classificação)	ANN de 2 etapas	85,589%	Parametrização do gás de proteção
(YOU; KIM, 2021)	Ponto de solda (diâmetro)	Regressão	90%	
	Ponto de solda (classificação da falha)	ANN de 1 etapa	91,2%	
	Ponto de solda (classificação da falha)	ANN de 2 etapas	88,2%	

Fonte: Elaborada pelo autor.

Tabela 1 - Arquiteturas de trabalhos relacionados (continuação)

Referência	Estudo	Modelos	Acurácia	Comentários
(JOHNSON *et al.*, 2023)	Ponto de solda (indicadores de qualidade)	ANN	99,36%	
		ANN	99,98%	
(ZHOU *et al.*, 2022)	Ponto de solda (previsão temporal)	LR	1,92% - 2,48% (MAPE)	4 configurações de entrada, 2 *data-sets*
		MLP	1,92% - 2,42% (MAPE)	4 configurações de entrada, 2 *data-sets*
		SVR	1,87% - 2,25% (MAPE)	4 configurações de entrada, 2 *data-sets*

Fonte: Elaborada pelo autor.

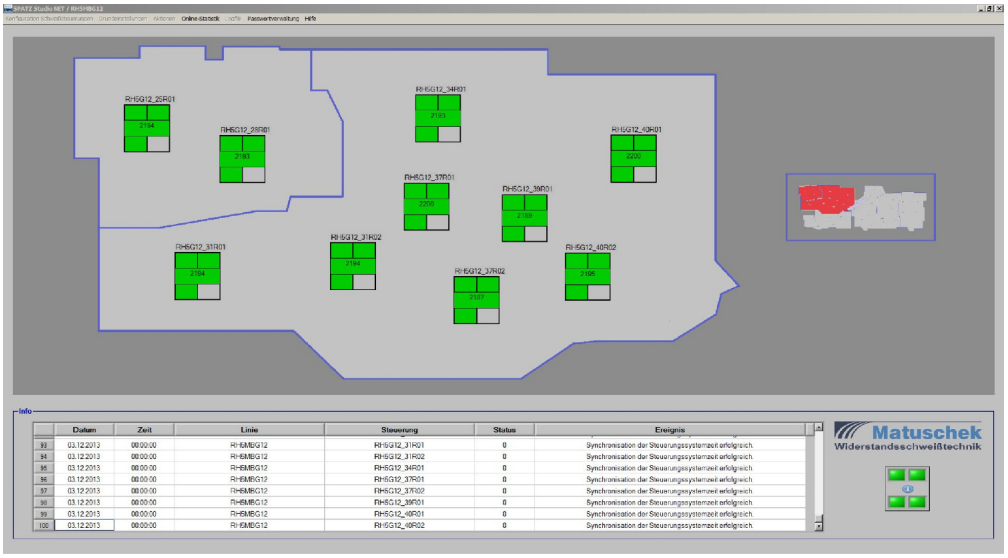
3 MATERIAIS E MÉTODOS

Nesse capítulo é abordada a metodologia utilizada para o desenvolvimento do trabalho, passando pela origem e armazenamento dos dados e seguindo com as particularidades encontradas no processo de solda relevantes para o estudo. Além disso, é dado continuidade ao Capítulo 2 a respeito dos métodos de aprendizado supervisionado e não-supervisionado, descrevendo a técnica e avaliação dos modelos além dos conceitos teóricos.

3.1 Banco de Dados

Os dados de solda utilizados neste trabalho foram cedidos por uma montadora e são originados de controladores de solda da marca Matuschek, empresa alemã especializada no processo de solda por resistência elétrica. É necessário um controlador para cada robô que atua com a arma de solda, porém é possível interagir com todos por meio do software fornecido, Spatz Studio NET, ilustrado na Figura 10. Nele, um operador pode visualizar em tempo real as curvas de solda, parâmetros elétricos, estatísticas e carregar novos programas ou realizar *backups* de antigos, permitindo a regulagem do processo. A Figura 11 representa a interface inicial para um controlador.

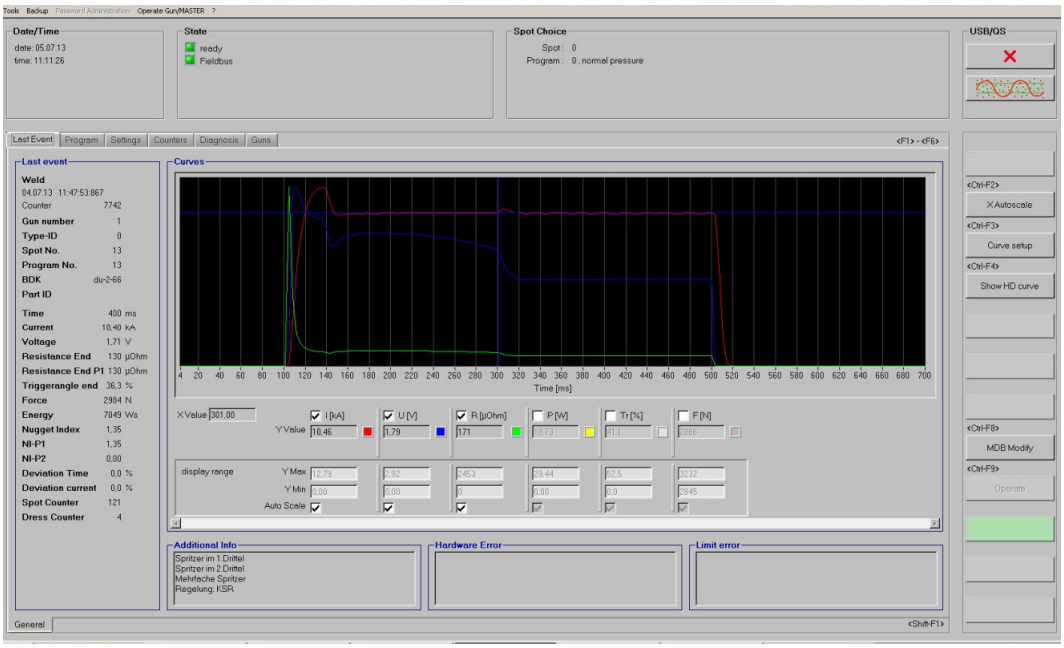
Figura 10 – Menu inicial do software Spatz Studio NET para controladores de solda Matuschek.



Fonte: Extraído de Matuschek.

Ainda, o Spatz Studio NET possui um recurso nativo para conexão do software com servidores SQL, permitindo que os dados gerados para cada ponto de solda por controlador sejam armazenados em tempo real. Logo, a coleta de dados foi facilitada, uma vez que dados locais de cada controlador de solda foram concentrados em um único banco de dados relacional Microsoft SQL Server e com acesso remoto, ou seja, sem necessidade

Figura 11 – Exemplo de curvas de solda no software Spatz Studio referente a um controlador Matuschek.



Fonte: Extraído de Matuschek.

Tabela 2 – Volume de dados coletados a longo prazo.

Característica	Quantidade
Linha de produção	1
Células	10
Robôs	73
Controladores de solda	73
Pontos de solda	3.000
Produtos / dia	130
Total de pontos / dia	390.000
Total de pontos / 6 meses	49.000.000

Fonte: Elaborada pelo autor.

de extraí-los *in loco*. A fim de calcular o volume de dados armazenados, imagina-se o seguinte cenário: ao longo de sua linha de produção BiW (*Body in White*, "carroceria crua"), a empresa automobilística em estudo realiza cerca de 3000 pontos de solda em seu produto. Para tanto, são utilizados 73 robôs ABB espalhados em 10 componentes, isto é, 73 controladores de solda em 10 células produtivas diferentes. Considerando uma produção média de 130 produtos ao dia, em 6 meses é possível coletar aproximadamente 49 milhões de eventos – vide a Tabela 2 –, permitindo caracterizar este trabalho como *Big Data*. Na Seção 3.2 é tratado como essa granularidade do processo e volume de dados influenciaram na amostragem dos dados.

A partir da conexão dos controladores ao servidor SQL Server, milhões de dados foram disponibilizados durante 6 meses para iniciar o desenvolvimento prático. Contudo,

é necessário relacionar os dados de solda com as informações do produto. Enquanto a identificação das peças – aqui denominada "PopID" – é recebida apenas pelo Controlador Lógico Programável (PLC, *Programmable Logic Controller*), o controlador de solda por sua vez não "enxerga" o que está soldando. Antes mesmo de compreender a natureza dos parâmetros historiados, portanto, foi iniciado o tratamento de dados com um código cíclico, de forma a atualizar frequentemente o banco de dados com a identificação da peça e modelo do produto soldado. Assim, a Figura 12 representa uma amostra de dados das principais variáveis já com desvios rotulados por uma variável binária "Anomaly", bem como a definição do tipo de desvio em "AnomalyType". Sobre os parâmetros, há alguns sobre o processo – como horário do evento de solda, produto, modelo e robô que efetuou a solda – e outros parâmetros elétricos – a exemplo da corrente RMS, tensão média no eletrôdo, resistência, entre outros –. O Spatz Studio armazena 32 parâmetros de solda que, complementados por outros 5 com informações do PLC de área, totalizam 37 variáveis disponíveis para o desenvolvimento do trabalho. A Tabela 3 detalha todas as variáveis encontradas, inclusive com unidade de medida e seu tipo.

Figura 12 – Amostra de dados extraídos do Matuschek.

Results | Messages

	TimeStamp	PopID	ManufCode	SpotNo	SpotName	TimerName	GunNo	ProgNo	IRMS	Umean	Rend	Energy	NI	NI_P2	Splashes	TriggerAngle	Anomaly	AnomalyType
99	2024-04-30 08:51:30.019	714020	P14L600001	417007	BW_010-A7-007-R	MF100R04TIMER	1	1018	7792,051	1,615113	0,000178	12220,41	1,36139	1,312342	0	38,13874	1	Ponto Retorcido
100	2024-04-30 08:51:33.015	714020	P14L600001	417008	BW_010-A7-008-R	MF100R04TIMER	1	1018	7124,722	1,504119	0,000196	10088,39	1,194588	1,164773	0	36,53377	1	Ponto Retorcido
101	2024-04-30 08:51:46.007	714020	P14L600001	418002	BW_010-A8-002-R	MF100R04TIMER	1	1018	7603,875	1,621329	0,000183	12760,33	1,347421	1,319511	0	38,01874	1	Ponto Retorcido
102	2024-04-30 08:39:52.015	714020	P14L600001	418013	BW_010-A8-013-R	MF080R04TIMER	1	1018	7261,282	1,661634	0,000183	14063,07	1,507099	1,40676	1	32,85585	1	Rebarba
103	2024-04-30 08:38:36.016	714020	P14L600001	118014	BW_010-A8-014-L	MF080R03TIMER	1	1018	7118,231	1,741477	0,000208	13591,93	1,406378	1,348934	1	36,25378	1	Rebarba
104	2024-04-30 08:37:01.015	714020	P14L600001	521002	BW_010-L1-002-R	MF080R05TIMER	3	1018	7086,451	1,744953	0,000198	9640,489	1,570752	1,447194	0	34,8778	1	Rebarba
105	2024-04-30 08:44:24.017	714020	P14L600001	521002	BW_010-L1-002-R	MF080R05TIMER	3	1018	7169,311	1,7058	0,000191	9959,484	1,54204	1,440591	1	33,99432	1	Rebarba
106	2024-05-02 23:50:48.019	714356	R20H600001	173013	BW_010-G3-013-L	MF100R01TIMER	2	1000	7991,72	1,106677	0,000142	6691,143	1,00811	1,027739	0	35,63029	1	Respingo
107	2024-04-30 01:18:20.009	714726	R20H600001	410005	BW_010-A0-005-R	MF080R04TIMER	1	1008	7244,389	1,384998	0,000161	11041,09	1,32815	1,283563	1	33,01234	1	Rebarba
108	2024-04-30 01:18:17.005	714726	R20H600001	410006	BW_010-A0-006-R	MF080R04TIMER	1	1008	7256,119	1,402937	0,000159	11454,67	1,376853	1,286103	1	32,30386	1	Rebarba
109	2024-04-30 01:30:12.021	714726	R20H600001	417008	BW_010-A7-008-R	MF100R04TIMER	1	1018	7066,259	1,5192	0,000186	10749,32	1,328948	1,306527	0	36,27378	1	Rebarba
110	2024-04-30 01:28:38.013	714726	R20H600001	118002	BW_010-A8-002-L	MF100R03TIMER	1	1018	7752,782	1,635017	0,000175	12142,39	1,398446	1,326368	1	37,37875	1	Rebarba
111	2024-04-30 01:18:51.009	714726	R20H600001	118014	BW_010-A8-014-L	MF080R03TIMER	1	1018	7399,548	1,739738	0,000195	14006,37	1,445039	1,379664	0	37,64625	1	Rebarba
112	2024-04-30 01:22:15.022	714726	R20H600001	221002	BW_010-L1-002-L	MF080R05TIMER	4	1018	6853,143	1,567356	0,000203	9629,752	1,321064	1,200187	1	34,61331	1	Rebarba
113	2024-04-30 01:26:00.020	714726	R20H600001	521002	BW_010-L1-002-R	MF080R05TIMER	3	1018	7125,313	1,705529	0,000195	9638,622	1,522031	1,419383	0	34,8148	1	Rebarba
114	2024-04-26 02:03:01.015	717711	R20N600001	110017	BW_010-A0-017-L	MF080R03TIMER	1	1008	7121,245	1,470663	0,000166	9748,276	1,43385	1,386293	0	35,49479	1	Rebarba
115	2024-04-26 02:02:33.641	717711	R20N600001	119001	BW_010-A9-001-L	MF080R03TIMER	1	1000	8751,042	1,428167	0,000136	5253,05	1,307939	1,332608	0	37,11976	1	Ponto Solto
116	2024-04-26 02:02:31.017	717711	R20N600001	119002	BW_010-A9-002-L	MF080R03TIMER	1	1000	8208,661	1,411736	0,000163	5837,804	1,114213	1,150967	0	40,42469	1	Ponto Solto
117	2024-05-02 02:05:10.023	717728	R20N600001	110039	BW_010-A0-039-L	MF080R03TIMER	2	1008	6897,822	1,277761	0,000161	10031,68	1,27203	1,232236	0	30,67839	1	Rebarba
118	2024-05-02 02:21:15.017	717728	R20N600001	118002	BW_010-A8-002-L	MF100R03TIMER	1	1018	7562,372	1,665355	0,000186	10640,77	1,408045	1,316413	0	37,60025	1	Rebarba
119	2024-05-02 02:07:03.011	717728	R20N600001	221002	BW_010-L1-002-L	MF080R05TIMER	4	1018	6994,184	1,712453	0,000208	10494,79	1,441014	1,357593	0	34,57231	1	Rebarba
120	2024-05-02 02:10:42.017	717728	R20N600001	521002	BW_010-L1-002-R	MF080R05TIMER	3	1018	7057,048	1,671964	0,000192	8949,042	1,537846	1,423182	1	34,24232	1	Rebarba
121	2024-05-02 02:03:01.017	717728	R20N600001	521002	BW_010-L1-002-R	MF080R05TIMER	3	1018	7188,026	1,557897	0,000172	9154,541	1,509908	1,424255	0	33,60433	1	Rebarba
122	2024-05-02 01:48:17.025	718342	G17N600001	221002	BW_010-L1-002-L	MF080R05TIMER	4	1018	7030,402	1,636666	0,000203	10098,69	1,356005	1,275235	0	36,22328	1	Rebarba
123	2024-04-26 18:01:02.005	719162	G20N600001	410011	BW_010-A0-011-R	MF080R04TIMER	2	1008	7040,18	1,510812	0,000177	11274,77	1,425087	1,366517	0	34,30281	1	Retorcido

Fonte: Elaborada pelo autor.

3.2 Particularidades do Processo e Catalogação de Dados

Conforme comentado na seção anterior, o processo base de solda pode ser dividido em diversas camadas. A fábrica em questão é dividida em 10 células produtivas, produzindo componentes diferentes do produto e para tanto, utilizando 72 robôs os quais manipulam 144 ferramentas de solda ao todo. Ao comparar peças diversas, também é natural uma variedade de espessuras de chapas metálicas, programas de solda e a própria localização de cada ponto, os quais somam mais de 8.000. A Figura 13 ilustra a dimensão de cada um desses níveis. Conclui-se portanto que a escolha das variáveis "TimerName", "BDK", "GunNo", "ProgNo" ou "SpotName" como parte dos modelos de *Machine Learning* são determinantes para o nível de especificação desejado. Imagina-se, por exemplo, que um

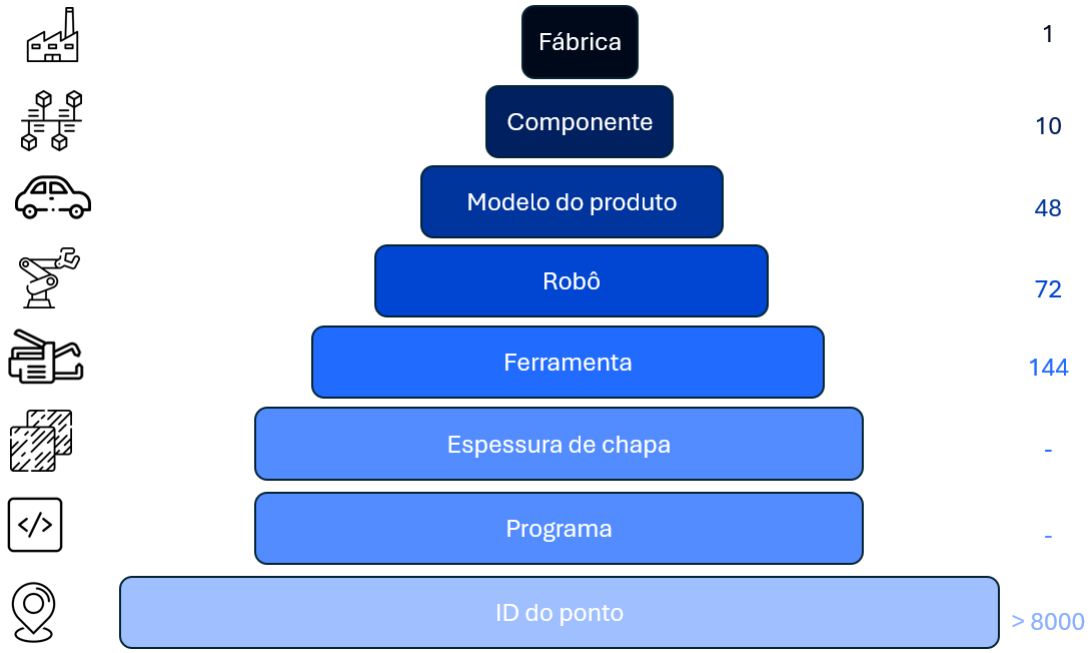
Tabela 3 – Variáveis monitoradas pelo controlador de solda Matuschek.

N°	Variável	Descrição	Unidade	Tipo
1	TimeStamp	Data e hora do ponto	yyyy-mm-dd hh:mm:ss.fff	datetime
2	PopID	ID do produto	-	string
3	SpotNo	ID do ponto, identifica a posição	-	int
4	SpotName	ID do ponto, identifica a posição	-	string
5	TimerName	Robô	-	string
6	TypeID	diferenciação de modelos	-	int
7	BDK	Espessura de chapa	-	string
8	GunNo	Ferramenta de solda	-	int
9	ProgNo	Programa de solda	-	int
10	PartID	-	-	string
11	Time	Tempo	s	float
12	IRMS	Corrente RMS	mA	float
13	Umean	Tensão Média no eletrôdo	kV	float
14	Rend	Resistência ao final da sequência de solda	µ	float
15	RendP1	Resistência ao final do primeiro pulso de corrente	µ	float
16	TriggerAngle	Fase; valor final de controle	%	float
17	Force	Força aplicada	N	float
18	Energy	Energia total	W	float
19	NI	Nugget Index; qualidade do ponto	-	float
20	NI_P1	Nugget Index P1; 1° pulso	-	float
21	NI_P2	Nugget Index P2; 2° pulso	-	float
22	DeviationT	Tempo prolongado da corrente operacional	%	float
23	DeviationI	Corrente de desvio operacional	%	float
24	SpotCounter	Contador de pontos	-	int
25	DressCounter	Contador de afiações	-	int
26	Splashes	Respingo	-	bit
27	WithoutCurrent	-	-	bit
28	ProcessError	Erro do processo	-	bit
29	HardError	Erro de hardware	-	bit
30	LimitErrors	Limites	-	int
31	NiReweld	-	-	bit
32	WeldPos	-	-	float
33	WeldPosTipwear	-	-	float
34	SheetThickness	-	-	float
35	Anomaly	Anomalia	-	bit
36	ManufCode	Modelo do produto	-	string
37	AnomalyType	Tipo de anomalia	-	string

Fonte: Elaborada pelo autor.

único algoritmo responsável por identificar desvios no processo inteiro pode tornar-se genérico e apresentar baixo desempenho, ao passo que seria inviável desenvolver e aplicar um algoritmo para cada endereço ou programa de solda. Quanto maior a granularidade do modelo, menor é a quantidade de dados disponíveis para treinamento e maior a dificuldade de implementação. Em vista disso, em um primeiro momento foi escolhido apenas a linha de solda principal entre as 10, esta contendo 15 robôs.

Figura 13 – Granularidade de um modelo baseado no processo.



Fonte: Elaborada pelo autor.

Filtrado o nível de granularidade do estudo, o próximo passo para o aprendizado supervisionado é a rotulação dos dados com desvios. Com esse objetivo, foi mobilizado uma equipe de inspeção de qualidade para verificar diariamente uma pequena parcela dos produtos produzidos. O método consistiu em alocar os produtos em uma área para controle de qualidade ao final do processo e observar, a olho nu, cerca de 7 peças soldadas na linha principal por dia. Identificados os endereços exatos dos pontos e seus tipos de desvios – seja rebarba, ponto solto ou retorcido –, estes eram abastecidos no banco de dados, utilizando a identificação do produto “PopID” e endereço “SpotName” para cruzar as informações. Além da captação de desvios pelo operador a olho nu, também foi realizada por meio de testes com aparelho ultrassom em cabinas aleatoriamente selecionadas na cadeia produtiva. Mesmo analisando poucos produtos, a coleta de dados levou cerca de duas horas por dia devido ao trabalho manual. Essa atividade foi mantida por cerca de dois meses, gerando no total 2.570 pontos de solda com desvios para iniciar o desenvolvimento de aprendizado supervisionado.

3.3 Pré-Processamento e Análise de Dados

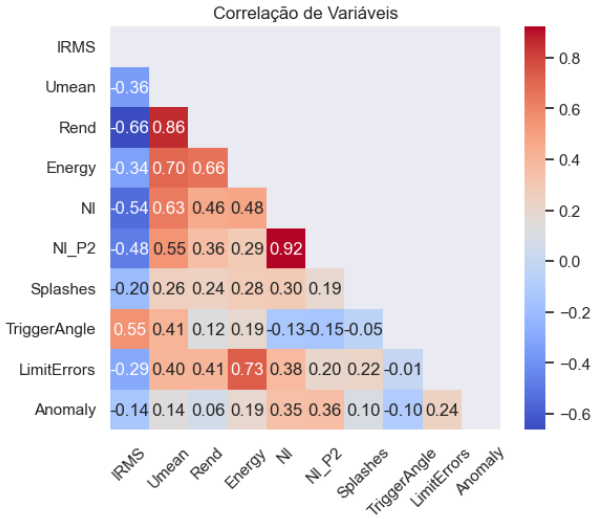
As etapas de extração e catalogação dos dados geraram uma grande amostra de dados de solda armazenados no banco relacional, referentes apenas a linha de produção escolhida na etapa de granularidade. A amostra continha 37 colunas de variáveis – entre elas aspectos de solda ou do processo, conforme a Tabela 3 – e cerca de um milhão de pontos de solda, fossem eles pontos inspecionados ou não. Foi necessário um pré-processamento e análise dos dados antes de seguir com a modelagem do aprendizado de máquina, de forma a tratá-los corretamente. Estudou-se o comportamento dos parâmetros elétricos dos pontos de solda com e sem desvio e, por fim, selecionadas previamente as variáveis mais relevantes para o estudo. Algumas análises separam os tipos de desvio, seja ele rebarba, respingo, ponto retorcido ou solto. Porém, vale recordar que o intuito do trabalho é apenas identificar um desvio, sem necessariamente categorizá-lo. Portanto, a saída do SVM e MLP deve ser binária, a fim de não restringir mais os dados disponíveis para treinamento. Com esses objetivos, utilizou-se o ambiente de desenvolvimento integrado Visual Studio Code com linguagem de programação Python e diversas bibliotecas *open source*, a exemplo do *pandas*, *numpy*, *matplotlib*, *seaborn* e *scikit learn*.

Embora inicialmente houvessem cerca de um milhão pontos disponíveis para análise, apenas 34 produtos foram de fato inspecionados na produção, resultando em 2.570 pontos com desvio ("Anomaly" = 1) e 58.558 restantes considerados "normais" ("Anomaly" = 0). A fim de evitar um *overfitting* dos algoritmos, os dados foram melhor balanceados em relação à coluna binária "Anomaly", de uma proporção de 3,32% para 20% com uma técnica de *undersampling*, isto é, utilizando uma subamostragem de 10.540 pontos normais e 2.570 desvios. Além disso, de 37 variáveis, 10 apresentavam dados nulos em sua maior parte ou totalidade, uma vez que não foram monitoradas pelo software Spatz Studio. Outras, como "TypeID", "PartID", "ProgNo" e "SpotNo" foram compreendidas como redundâncias de outras variáveis, portanto também foram descartadas de início. Neste momento, restaram 20 colunas para seguir as análises.

Foi aplicada uma técnica de anonimização das variáveis "ManufCode" e "TimerName" – modelo de produto e controlador de solda, respectivamente – a fim de garantir o anonimato da empresa e suas informações. Cada item foi substituído por uma *string* no formato ("coluna" + "_" + "letra"), sendo "coluna" referente à variável (programa, modelo ou timer) e "letra" um elemento randômico do alfabeto, constituindo a chamada técnica de generalização dos dados. Os *bits* de limites atingidos ("LimitErrors") foram transformados em números ordinais com a técnica *label encoding*, de forma a normalizar os dados mantendo a compreensão de quanto maior o número, maior a quantidade de erros encontrados. Já as variáveis restantes foram consideradas inofensivas para a privacidade da empresa e portanto, não foi preciso aplicar outras técnicas como generalização ou supressão.

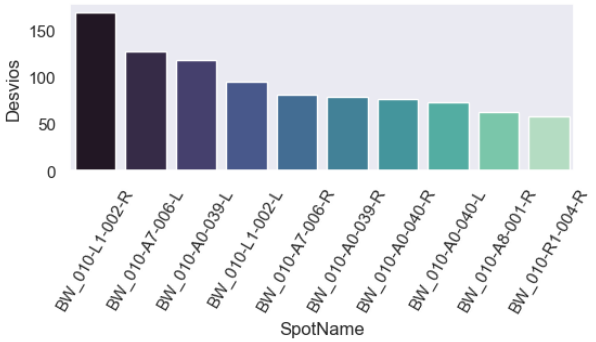
A seguir, foi feita uma análise gráfica detalhada utilizando as bibliotecas *Seaborn* e *Matplotlib*, a fim de reduzir as 20 variáveis. O mapa de calor na Figura 14, por exemplo, ilustra o nível de relação entre variáveis numéricas: o gradiente vermelho ($0 < x \leq 1$) simboliza variáveis diretamente proporcionais, enquanto o azul ($-1 \leq x < 0$) simboliza as inversamente proporcionais. Dito isso, observa-se que o mapa de calor já é um primeiro indicativo de que resistência elétrica "Rend", fase "TriggerAngle" e "Splashes" podem influenciar menos os desvios. A Figura 15 demonstra que, à exceção do endereço BW_-010-L1-002-R, os desvios são bem distribuídos e portanto a localização do ponto de solda não influencia os desvios. Por outro lado, na Figura 16 o modelos Z demonstra maior probabilidade de apresentar desvios futuramente em relação aos outros.

Figura 14 – Mapa de calor de variáveis numéricas.



Fonte: Elaborada pelo autor.

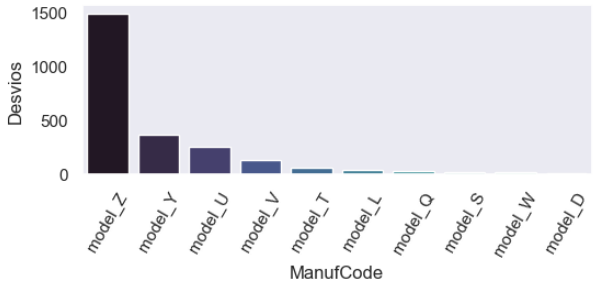
Figura 15 – Endereços com mais desvios.



Fonte: Elaborada pelo autor.

O Índice do Ponto de Solda (NI, *Nugget Index*) é uma variável importante para qualidade do ponto, calculado com outros parâmetros elétricos como resistência final, corrente RMS e tensão média, como pode ser observado pela alta relação no mapa de calor da Figura 14. Logo, foi analisado o índice para dados anômalos e normais por meio

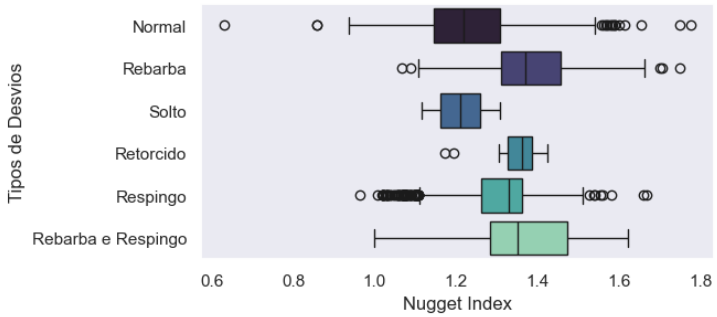
Figura 16 – Modelos com mais desvios.



Fonte: Elaborada pelo autor.

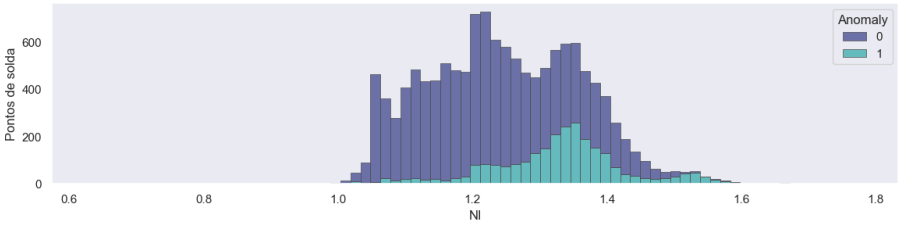
de um boxplot (Figura 17) e histograma (Figura 18), comprovando a diferença nas duas condições e relevância de *Nugget Index* para o modelo. Explorando mais a variável, *Nugget Index* também foi utilizado para analisar os 13 controladores de solda simultaneamente: os robôs "C", "D", "G", "J" e "M" não apresentam distribuições normais de NI na Figura 19. Considerando que todos os robôs compartilharam chapas metálicas e programas de solda, entende-se que não deveria haver uma diferença relevante de parâmetros elétricos, portanto "TimerName" também foi levado em consideração para a amostragem final.

Figura 17 – Boxplot de Nugget Index por tipos de desvios.



Fonte: Elaborada pelo autor.

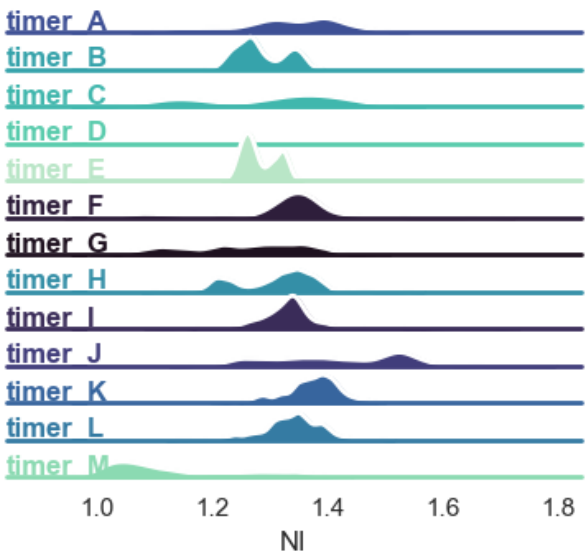
Figura 18 – Histograma de Nugget Index por desvios.



Fonte: Elaborada pelo autor.

O chamado ângulo de gatilho representa a fase da corrente elétrica aplicada ou valor final de controle em porcentagem. O boxplot na Figura 20 demonstra a paridade entre pontos normais e anômalos e que portanto, "TriggerAngle" não é determinante para o modelo. Além disso, o gráfico de barras empilhadas na Figura 21 calcula a quantidade de

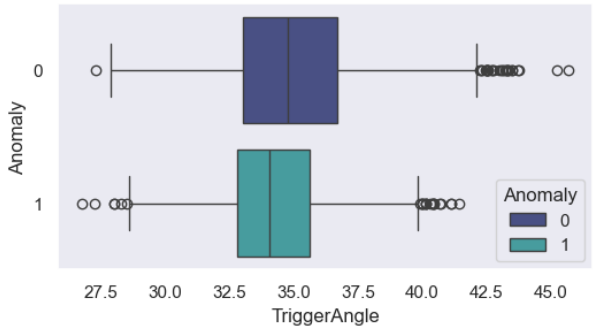
Figura 19 – Histogramas de Nugget Index por controladores de solda.



Fonte: Elaborada pelo autor.

desvios quando houve respingo ("Splashes" = 1) ou não ("Splashes" = 0) e não demonstra relação considerável, pois há ligeiramente menos desvios com respingos ("Anomaly" = 1 e "Splashes" = 1). Esta análise valida o valor encontrado anteriormente no mapa de calor. O mesmo vale para as ferramentas de solda, enumeradas de 1 a 5: a Figura 22 apresenta a ferramenta "1" com mais desvios do que as outras, porém analisando proporcionalmente à quantidade de pontos de solda realizados pela mesma ferramenta, faz sentido gerarem mais desvios. Logo, "TriggerAngle", "Splashes" e "GunNo" foram removidas do *dataset*.

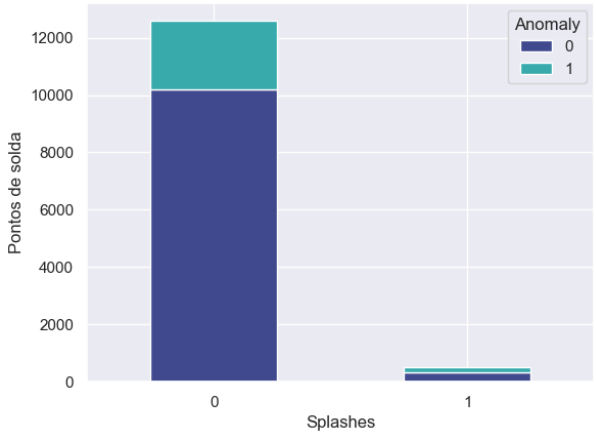
Figura 20 – Boxplot de fase por desvio.



Fonte: Elaborada pelo autor.

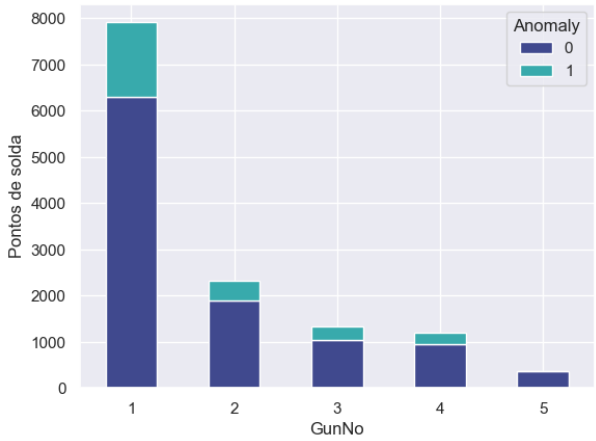
Outro fator relevante para a construção dos modelos de aprendizado de máquina foi a composição e espessura das chapas metálicas, denominado "BDK". No gráfico de "BDK" por desvios (Figura 23), observa-se que materiais como "HCC6 UHSS, t > 3mm" sequer apresentaram defeitos, enquanto "HCC6, BOR < 5 mm_V1" teve mais de 50%. Por outro lado, uma variável de processo relevante foi os limites atingidos na programação de solda. O histograma na Figura 24 demonstra que todos os tipos de desvios apresentam

Figura 21 – Gráfico de desvios por respingos.



Fonte: Elaborada pelo autor.

Figura 22 – Gráfico de desvios por ferramentas.



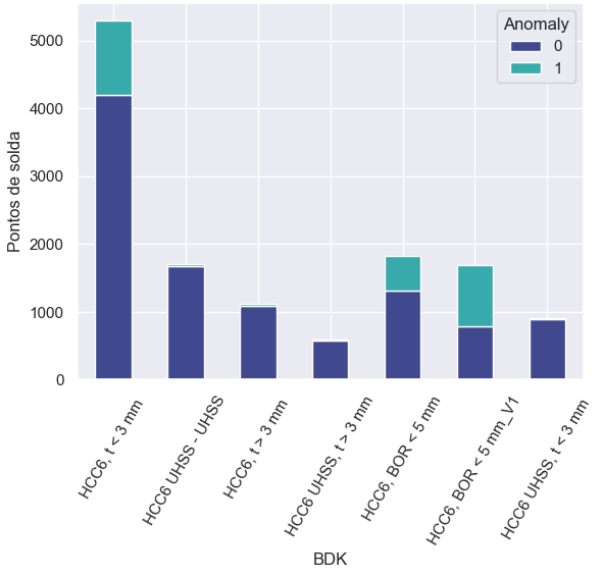
Fonte: Elaborada pelo autor.

uma distribuição similar em relação aos limites encontrados, com maior densidade nos *bits* 5, 7 e 10. Sendo assim, "LimitErrors" foi outra variável a ser considerada.

Finalizando a seleção de variáveis, foi inevitável analisar corrente elétrica e resistência: estando elas extremamente relacionadas, foi feita uma análise conjunta para compreender a diferença de distribuição entre os pontos com e sem anomalias. A Figura 25 mostra uma distribuição platicúrtica dos dois parâmetros com desvios em relação à distribuição normal de pontos de solda comuns, a qual leva a entender que corrente elétrica e resistência são um fator importante para a composição do modelo, como era esperado.

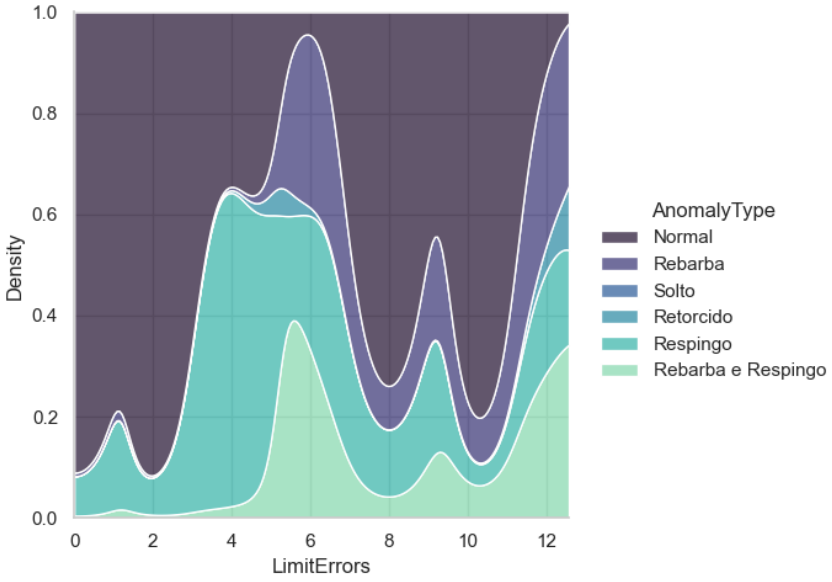
Após o tratamento de dados e filtro de variáveis com uma longa análise gráfica, obteve-se o *dataset* no formato da Tabela 4. Todavia, ainda foi preciso transformar algumas colunas, uma vez que os algoritmos SVM, MLP e K-Means trabalham apenas com variáveis numéricas. Então, aplicou-se a técnica de *one hot encoding* com a biblioteca *Scikit Learn* para decodificar os modelos, controladores de solda e materiais de chapa em colunas booleanas. Uma vez que as variáveis numéricas estavam em escalas distintas, também

Figura 23 – Gráfico de desvios por espessura de chapa (BDK).



Fonte: Elaborada pelo autor.

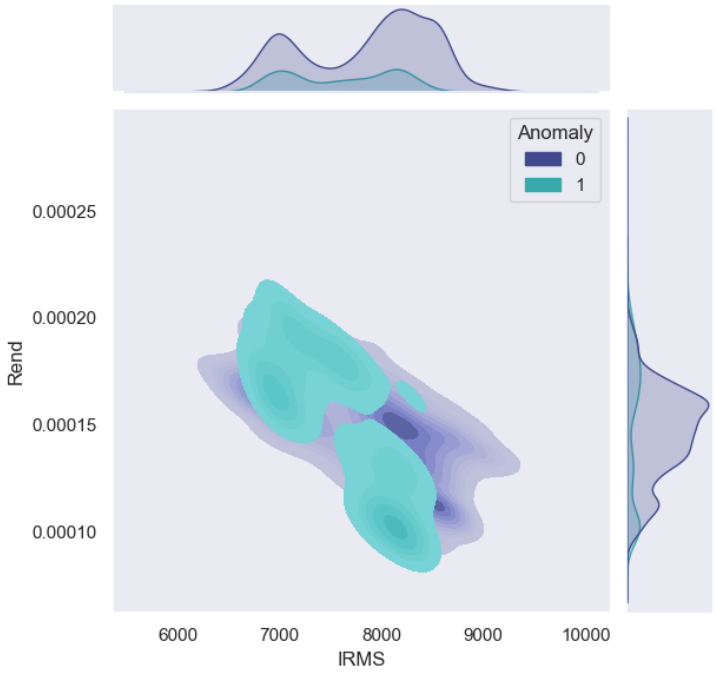
Figura 24 – Gráfico de distribuição de tipo de desvios por erros de limites (LimitErrors).



Fonte: Elaborada pelo autor.

foi preciso normalizá-las: segundo Brownlee (2020b), algoritmos como SVM ou K-Means que utilizam medidas de distância são diretamente afetados pelo peso de variáveis com escalas maiores. Contudo, é necessário conhecer a natureza dos parâmetros antes de normalizá-los, pois corrente elétrica e energia não atendem uma distribuição gaussiana por apresentarem desvios padrões de 640,57 e 2740,93, respectivamente. Logo, ao invés do método tradicional de *standardization* (do inglês, "padronização" ou também chamado de "escala central"), foi aplicado a técnica de normalização e função "MinMaxScaler" do *Scikit Learn*, transformando as novas escalas baseando-se nos valores mínimos e máximos

Figura 25 – Gráfico de distribuição de resistência e corrente elétrica por desvios.



Fonte: Elaborada pelo autor.

conhecidos. A Tabela 4 exemplifica a amostra após o método de seleção de variáveis, enquanto a Tabela 5 ilustra sua formatação após a transformação e normalização dos dados, finalizando assim a etapa de pré-processamento e análise de dados. Os aprendizados indutivos a seguir também foram formulados utilizando a linguagem de programação *Python* e sua biblioteca *Scikit Learn*, principalmente.

Tabela 4 – Exemplo de *dataset* após selecionar as variáveis.

Colunas	Exemplo 1	Exemplo 2	Exemplo 3
ManufCode	model_Z	model_Y	model_M
SpotName	BW_010-R5-030-L	BW_010-C9-011-R	BW_010-A2-005-L
TimerName	timer_A	timer_B	timer_C
ProgNo	prog_A	prog_B	prog_B
BDK	HCC6 UHSS, t > 3 mm	HCC6, t > 3 mm	HCC6, t > 3 mm
IRMS	8.083024	8.787907	8.223250
Umean	1.268148	1.416155	1.362710
Rend	0.000140	0.000136	0.000148
Energy	7.096311	6.18.445	5.833529
NI	1.204353	1.296011	1.230750
LimitErrors	6	0	0
Anomaly	0	1	1
AnomalyType	Normal	Rebarba	Respingo

Fonte: Elaborada pelo autor.

Tabela 5 – Exemplo de *dataset* após transformação e normalização.

Colunas	Exemplo 1	Exemplo 2	Exemplo 3
IRMS	0.592356287105	0.336144544814	0.1981008518216
Umean	0.0970100792208	0.3427210328168	0.505883708199
Rend	0.0655737704918	0.508196721311	0.590163934426
Energy	0.0496716592440	0.2132402472698	0.323803826914
NI	0.3494521176290	0.2794126474468	0.583005736484
LimitErrors	0.0	0.0	0.0
TimerName_timer_A	1.0	0.0	0.0
TimerName_timer_B	0.0	1.0	0.0
BDK_HCC6 UHSS, t > 3 mm	1.0	1.0	0.0
BDK_HCC6 UHSS, t < 3 mm	0.0	0.0	0.0

Fonte: Elaborada pelo autor.

3.4 Aprendizado Supervisionado

Conforme descrito na Seção 2.2 deste trabalho, o aprendizado supervisionado é uma das formas de aprendizado indutivo, aplicado quando as classes do estudo já são previamente conhecidas. Com os dados devidamente coletados, rotulados e processados, esses foram divididos em amostras de treinamento (75%) e teste (25%), possibilitando então praticar os modelos SVM, MLP e CATBoost. O primeiro, sintetizando a Seção 2.3, utiliza um hiperplano de funções lineares no espaço e calcula suas distâncias euclidianas - por isso a normalização prévia dos dados - até os pontos de treinamento. O nome *Support Vector Machine* é devido aos chamados vetores de suporte os quais definem as maiores margens do hiperplano, objetivo final do algoritmo. A Figura 5 ilustra a condição.

Entre os trabalhos relacionados (Seção 2.4), as principais referências para a construção do modelo SVM foram os trabalhos de Nasir *et al.* (2022) e Khokhar *et al.* (2015). O primeiro apresenta um modelo simples, o qual atingiu 80,68% de acurácia, enquanto Khokhar *et al.* (2015) complementa o modelo com um extrator de características de Transformada Discreta de Wavelet (DWT, *Discrete Wavelet Transform*). Embora Khokhar *et al.* (2015) tenha atingido uma acurácia superior de 96,86% ao classificar perturbações na rede elétrica, entende-se que um DWT não se aplica a este estudo. O trabalho não tem como objetivo analisar uma série temporal de pontos de solda e, de acordo com Percival and Mondal (2012), seria prejudicado se o fizesse, uma vez que os dados de ponto de solda não foram catalogados em períodos rigidamente determinados. Em contrapartida, o pré-estudo de caractéristicas realizado por meio de análises gráficas foi uma técnica mais simples para extrair os melhores parâmetros a serem utilizados no SVM.

O classificador utilizado no *Scikit Learn* foi o Classificação de Vetores de Suporte (SVC, *Support Vector Classification*), sendo o mais comum para o algoritmo com poucos dados de entrada. O SVC foi configurado com *gamma* igual a 2, probabilidade e *verbose*

ativadas, tolerância para encerramento de 0,001, pesos balanceados e estado randômico 32.

O segundo algoritmo de aprendizado supervisionado desenvolvido foi o MLP, uma rede neural com perceptrons de múltiplas camadas e aprendizado por retropropagação. Isso significa que inúmeros neurônios alinhados propagam os resultados entre camadas ocultas, porém a entrada é retroalimentada com os pesos dos parâmetros a cada iteração do treinamento, isto é, os valores são constantemente atualizados e o erro da MLP tende a cair durante o treinamento. A Seção 2.3 descreve matematicamente o algoritmo e sua eficiência na resolução de problemas não-lineares, como a análise elétrica de pontos de solda.

Novamente, Nasir *et al.* (2022) é uma das principais referências para o desenvolvimento do MLP: a título de comparação, o autor desenvolveu a rede neural para classificar a contaminação da água, utilizando a função de ativação ReLU, otimizador ADAM, 100 camadas ocultas, taxa de aprendizagem de 0,001 e 200 épocas para o treinamento. Seu resultado final foi de 88,63% de acurácia, superior ao seu próprio SVM. Zhou *et al.* (2022), por outro lado, detalhou menos os atributos de sua rede. Ele utilizou uma função de ativação de tangente hiperbólica e de 5 a 50 nós para atingir erro percentual médio absoluto de 1,92% a 2,42%. Por outro lado, Zhou *et al.* (2022) também utilizou um extrator de características de regressão linear, que por sua vez não será utilizado neste estudo pelo mesmo motivo citado para o extrator de Transformada Discreta de Wavelet de Khokhar *et al.* (2015).

O classificador de MLP foi treinado com 100 camadas de ocultas, 300 épocas e função de ativação ReLU, a qual traz a vantagem de ser uma função não-linear e, portanto, com grande poder computacional para dados complexos (BROWNLEE, 2020a). Além disso, adotou-se o taxa de aprendizagem de 0,1, estado randômico 32 e o parâmetro *alpha* igual a 0,01 para evitar *overfitting* por penalizar pesos com grandes magnitudes. Por fim, a função de otimização SGD a fim de encontrar os valores mínimos locais. O progresso do erro durante o treinamento da rede neural foi acompanhado ativando a *verbose*.

O terceiro algoritmo desenvolvido foi de CATBoost, sendo Nasir *et al.* (2022) o único trabalho mencionado no Capitúlo 2 que utiliza este método. Embora o autor não apresente a parametrização do modelo em seu trabalho, foi relatado 94,51% de acurácia e conclui-se que é um algoritmo eficiente com CPU. Contudo, Nasir *et al.* (2022) também alerta para a tendência do CATBoost sofrer *overfitting*, não lidar bem com ruídos, necessitar de muitos parâmetros e apresentar um treinamento demorado.

Com o suporte de um método de busca de hiperparâmetros, o CATBoost foi modelado com amostragem randômica de 0,0581, divisões de dados numéricos para processamento em 88 – a fim de equilibrar tempo de processamento e qualidade dos resultados – e profundidade das árvores de decisão definida em 8. Foram necessárias 109 épocas com taxa de aprendizagem 0,2224 e regularizador de nós (ou “folhas”) definida

em 8 para não deixar *outliers* influenciarem demais nos resultados e causar um *overfitting* indesejado.

3.5 Aprendizado Não-Supervisionado

O aprendizado não-supervisionado, por sua vez, não recebe a saída catalogada como pontos de solda com desvio pois seu objetivo é clusterizar os dados, isto é, agrupá-los em classes baseando-se apenas nos dados de entrada. Assim, também não foi necessário separar os dados em amostras de treinamento e teste para esse algoritmo. O único modelo obtido como referência no estado da arte foi o de clusterização hierárquica de Lopez *et al.* (2018) para doenças genéticas, o qual atingiu 96,60% de acurácia. Todavia, na Seção 2.3 é justificado a escolha da clusterização particional como alternativa: o recurso computacional limitado, grande volume de dados de entrada e a simplicidade em reconhecimento de padrões definiram o *K-Means* como o principal algoritmo comparativo em relação ao aprendizado supervisionado, seguido pela clusterização hierárquica de ligação simples.

Recapitulando, *K-Means* calcula a distância euclidiana entre os pontos até o centróide dos *clusters*, atualizando sua nova posição a cada dado inserido no treinamento até obter a separação ideal das classes. Por esse motivo, o algoritmo requer que o usuário saiba a quantidade de *clusters* que devem ser encontrados. Há técnicas para estimar esse número, como o "método do cotovelo" ou da "silhueta" (SAPUTRA; SAPUTRA; OSWARI, 2019). Contudo, este trabalho já apresenta como objetivo identificar apenas 2 classes: pontos de solda normais e anômalos, independente do tipo de desvio. Logo, foi escolhido o classificador KMeans do *Scikit Learn* com os dados de entrada processados.

3.6 Raciocínio Prático

Neste capítulo, foi descrito toda a metodologia utilizada para o desenvolvimento prático dos algoritmos de *Machine Learning*. Mostrou-se a complexidade da extração dos dados por meio de um *software* antes desconhecido da empresa Matuschek e, principalmente, o caminho para manter um ambiente de armazenamento de dados saudável em Microsoft SQL para condições de *Big Data*, respeitando o volume e velocidade de inserção de novos dados. Além disso, as camadas de automação da fábrica foram associadas ao nível de granularidade mais apropriado para o modelo, escolhendo um modelo por linha produtiva. Dessa forma, equilibrou-se quantidade de modelos diferentes e seu nível de especificação para garantir bom volume de dados com desvio para treinamento e praticidade de implementação no processo.

A seguir, os pontos de solda foram segmentados e tratados com técnicas de balanceamento, decodificação e até mesmo anonimização. Foi necessário uma análise gráfica para filtrar os parâmetros mais relevantes para criação dos modelos e, tão importante quanto, a transformação dos dados a partir de seus valores mínimos e máximos garantiu a

normalização da amostra. Ao final da fase de processamento, restaram 26 colunas, entre dados numéricos normalizados e categóricos transformados em valores booleanos.

Por fim, os dados processados foram divididos para o treinamento de cinco modelos, sendo SVM, MLP e CATBoost modelos de aprendizado supervisionado, *K-Means* e clusterização hierárquica os modelos de aprendizado não-supervisionado. É possível verificar os detalhes do modelamento nas Seções 3.4 e 3.5 antes de seguir para os resultados no Capítulo 4.

4 RESULTADOS E DISCUSSÕES

Os modelos principais avaliados foram o SVM, MLP e K-Means, seguidos pelos modelos CATBoost e de clusterização hierárquica como comparativos. Como base de treinamento, foram utilizados 2.570 pontos de solda catalogados como anômalos e, em contrapartida, 10.540 pontos considerados normais pelos técnicos de qualidade dentro da mesma amostragem de produtos. Dessa forma, temos uma proporção de aproximadamente 20% de desvios em relação ao total da amostra, a qual foi dividida em 75% para treinamento e 25% para teste.

As variáveis de entrada escolhidas para o desenvolvimento dos modelos foram: modelo do produto, nome do ponto de solda, nome do controlador de solda, número do programa de solda, espessura de chapa, corrente elétrica média quadrática (RMS, *Root Mean Square*), tensão elétrica média, resistência elétrica, energia, *nugget index* e *bits* de limites. Já a saída dos modelos atendem apenas como classificação binária: desvio (1) ou normal (0). Todo o cuidado com a base de dados foi essencial para alcançar os resultados dos experimentos para cada modelo.

4.1 Desempenho de Support Vector Machine

O primeiro experimento foi a modelagem de um SVM com SVC. Entre outros hiperparâmetros escolhidos, tem-se a função de base radial (RBF, *Radial Basis Function*) como *kernel*, responsável por calcular a similaridade entre dois pontos do treinamento. Seu valor *gamma* é inversamente proporcional à influência dos vetores de suporte, isto é, à distância entre os pontos (SREENIVASA, 2020). A fim de evitar um *overfitting* dos dados com uma sensibilidade do modelo muito alta, definiu-se *gamma* e *random state* como 2 e 32, respectivamente. A Tabela 6 resume os hiperparâmetros do modelo SVM.

Tabela 6 – Parametrização do modelo SVM.

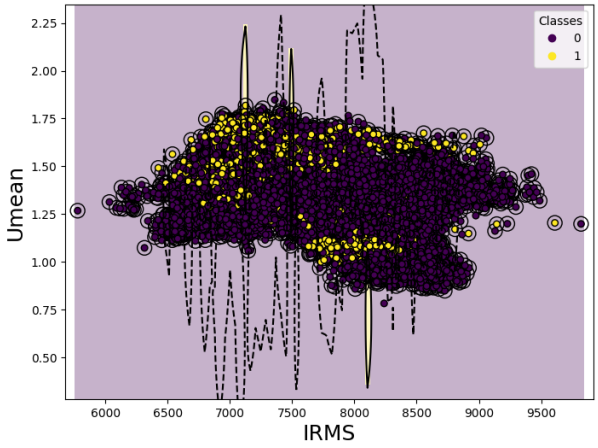
Parâmetros	Valor
Classificador	SVC
Kernel	RBF
Gamma	2
Random state	32

Fonte: Elaborada pelo autor.

Uma forma convencional de ilustrar o desempenho do classificador é por meio dos limites de decisão, como na Figura 26. Nela, observa-se 2 das 26 variáveis de entrada em um plano bidimensional para traçar o hiperplano que define as saídas com desvio (pontos amarelos) ou normais (pontos roxos). Não foi possível identificar um padrão separando as

duas classes com o *kernel* RBF, restando detalhar a matriz de confusão do modelo (Tabela 7) para calcular acurácia (89,60%), precisão (78,59%), *recall* (64,22%) e *F1-score* (70,68%) para melhor avaliação do desempenho. As métricas de todos os modelos foram descritas na Tabela 13 na Seção 4.5.

Figura 26 – Limites de decisão de SVM.



Fonte: Elaborada pelo autor.

Tabela 7 – Matriz de confusão resultante do modelo de SVM.

	Valor	Predito
Valor Real	Anomalia	Normal
Anomalia	411	229
Normal	112	2526

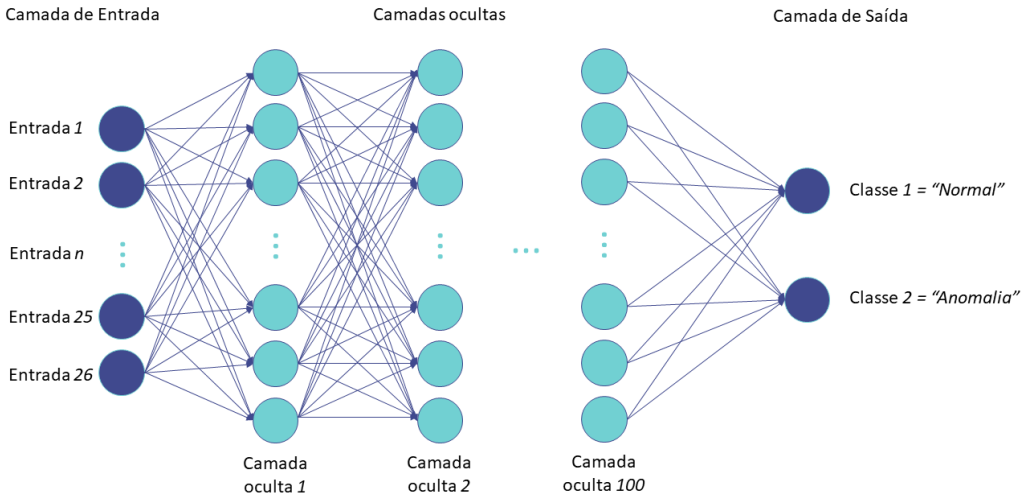
Fonte: Elaborada pelo autor.

4.2 Desempenho de Multi-Layer Perceptron

O modelo MLP, por sua vez, foi desenvolvido como uma rede neural de retropropagação de 26 dados de entrada - entre colunas normalizadas e transformadas -, 100 camadas ocultas e 2 saídas, conforme a arquitetura na Figura 27. Conforme também justificado na metodologia da Seção 3.4, utilizou-se a função de ativação ReLU para a classificação binária. Embora o treinamento tenha sido configurado para executar por 300 épocas, o modelo atingiu a perda mínima de 0,2217 com 155 iterações apenas. O restante dos hiperparâmetros podem ser conferidos na Tabela 8.

Tal configuração de hiperparâmetros gerou a matriz de confusão da Tabela 9. Nela, nota-se uma semelhança alta com os valores do modelo anterior SVM, sendo a maior diferença nos menores falsos negativos (208 contra 229) e maiores verdadeiros positivos (432 contra 411). A matriz gerou os seguintes índices: acurácia de 90,02%, precisão de 78,40%, *recall* de 67,50% e, por fim, *F1-score* de 72,54%.

Figura 27 – Arquitetura da Rede Neural Artificial de MLP com retropropagação.



Fonte: Elaborada pelo autor.

Tabela 8 – Parametrização do modelo MLP.

Parâmetros	Valor
Função de ativação	ReLU
Função de otimização	SGD
Camadas ocultas	100
Épocas	300
Taxa de aprendizagem	0,1
Random state	32
Alpha	0,01

Fonte: Elaborada pelo autor.

Tabela 9 – Matriz de confusão resultante do modelo MLP.

	Valor	Predito
Valor Real	Anomalia	Normal
Anomalia	432	208
Normal	119	2519

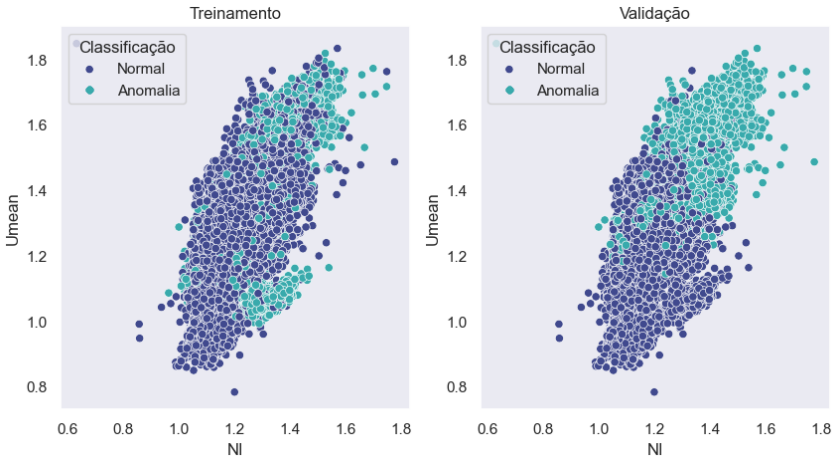
Fonte: Elaborada pelo autor.

4.3 Desempenho de Clusterização K-Means

Já como principal modelo de aprendizado não-supervisionado, o K-Means utilizou toda a amostra de dados original – 13.110 pontos de solda e mesma dimensão –, a qual antes fora dividida entre bases de treinamento e teste. Neste modelo é possível escolher o número de *clusters* a serem encontrados por meio do "método do cotovelo", como é comumente chamado. Porém, como a natureza do problema já é conhecido, o número de classes foi definido previamente como 2, sendo este o único parâmetro necessário configurar. Como resultado, foi obtida a matriz de confusão da Tabela 10, a qual indica valores altos para falsos positivos e negativos e, consequentemente, afetando as métricas tradicionais

de avaliação. Acurácia resultante foi de 60,67%, precisão de 80,17%, *recall* de 67,87% e *F1-score* com uma taxa de 73,51% apenas. A Figura 28 compara graficamente as predições por K-Means com o *dataset* de treinamento, indicando como os centróides do modelo induziram falsos negativos e falsos positivos.

Figura 28 – Gráfico de dispersão de pontos preditos pelo modelo classificador K-Means.



Fonte: Elaborada pelo autor.

Tabela 10 – Matriz de confusão resultante do modelo classificador K-Means.

	Valor	Predito
Valor Real	Anomalia	Normal
Anomalia	7153	3387
Normal	1769	801

Fonte: Elaborada pelo autor.

Especificamente para modelos de clusterização, é possível calcular outras métricas para entender seu comportamento, como por exemplo o método de silhueta (*silhouette score*). O coeficiente indica pela Equação 4.1 quão bem definidas são as classes preditas, sendo valores próximos de 1 uma clusterização condizente com a realidade e -1, não:

$$S(i) = \frac{(b(i) - a(i))}{max(a(i), b(i))}, \tag{4.1}$$

onde *a(i)* a distância média de i até outros pontos do mesmo *cluster* e, consequentemente, *b(i)* a distância média de i até outros pontos do *cluster* diferente mais próximo. Dito isso, foi calculado um valor de 0,65, próximo do valor ideal 1,0 porém ainda de baixo desempenho.

4.4 Desempenho de Algoritmos Alternativos

Nessa seção são descritos dois modelos alternativos aos apresentados no Capítulo 2, que trata dos trabalhos relacionados, e Capítulo 3, que apresenta as metodologias: CAT-Boost para aprendizado supervisionado - inspirado em Nasir *et al.* (2022) - e clusterização

hierárquica, um método menos convencional para aprendizado não supervisionado em comparação ao K-Means, porém estudado por Lopez *et al.* (2018).

O CATBoost é essencialmente um algoritmo de árvores de decisão binárias e simétricas. Logo, com uma procura randômica de hiperparâmetros, foi definida a configuração da Tabela 11 para o modelo final.

Tabela 11 – Parametrização do modelo CATBoost.

Parâmetros	Valor
Baggin temperature	0,0581
Border count	88
Profundidade	8
Épocas	109
Taxa de aprendizagem	0,2224
L2 leaf reg	8
random strength	0,0205

Fonte: Elaborada pelo autor.

Utilizando a mesma base de dados dos algoritmos de aprendizado supervisionado anteriores - 3.278 pontos, equivalentes a 25% da amostra total -, resultou uma matriz de confusão ligeiramente superior na Tabela 12, com destaque para os verdadeiros negativos obtidos. A matriz gerou os seguintes índices para o CATBoost: acurácia de 91,06%, precisão de 78,03%, *recall* de 75,47% e, por fim, *F1-score* de 76,73%.

Tabela 12 – Matriz de confusão resultante do modelo CATBoost.

	Valor	Predito
Valor Real	Anomalia	Normal
Anomalia	483	157
Normal	136	2502

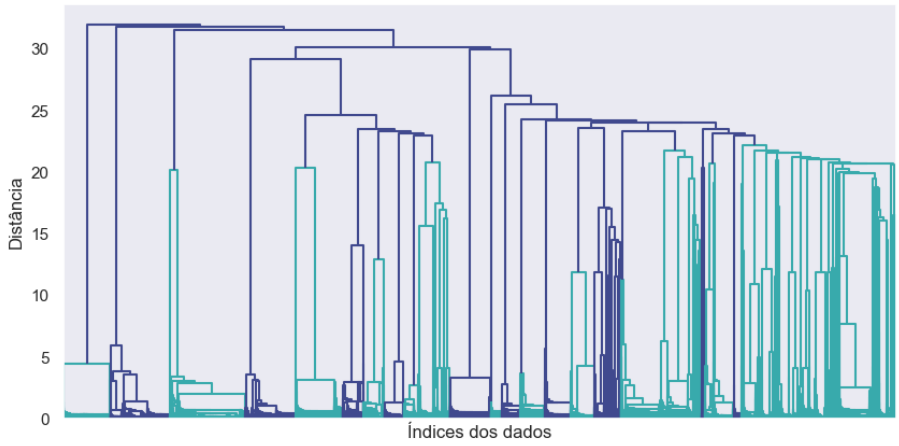
Fonte: Elaborada pelo autor.

Finalmente, a clusterização hierárquica foi desenvolvida com o método de ligação única (*single linkage*). Além de ser o método que tomou mais tempo para treinar (4 minutos e 12 segundos) em relação aos anteriores, analisando o dendograma da Figura 29 é possível concluir que apresentou um desempenho baixo: há diversos ramos não aglomerados para um único *cluster* e com distâncias elevadas, indicando que o modelo tende a encontrar mais classes. Portanto, no futuro a clusterização hierárquica pode ser mais efetiva para classificar tipos de desvios separadamente (rebarba, ponto retorcido, entre outros), porém não demonstrou-se adequada para o objetivo em questão.

4.5 Comparativo Geral

Analisando os resultados descritos neste capítulo, é válido concluir que os algoritmos de aprendizado não-supervisionado não apresentaram um bom desempenho. Enquanto o

Figura 29 – Dendograma de Clusterização Hierárquica com métrica euclidiana.



Fonte: Elaborada pelo autor.

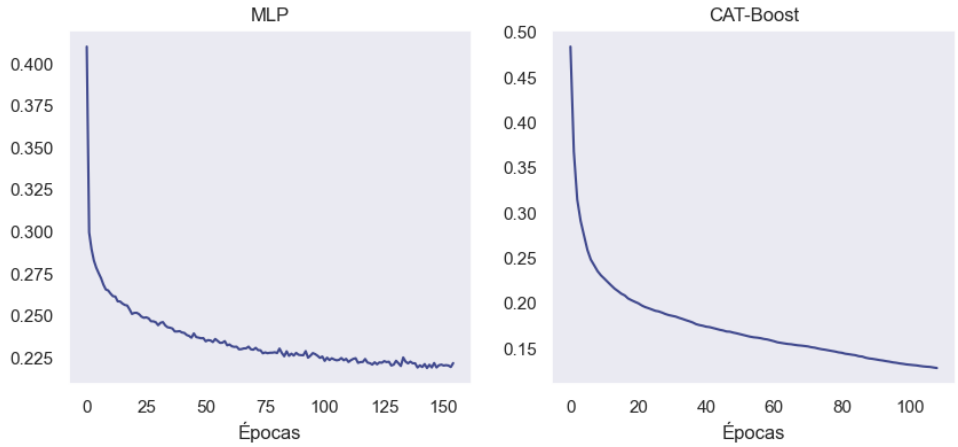
modelo K-Means prejudicou-se com a grande quantidade de falsos negativos e positivos, com uma acurácia de 60,67% e *F1-score* de 73,51%, a clusterização hierárquica por ligação única resultou em um *overfitting* do modelo, possivelmente demonstrando-se melhor para classificar mais classes.

Todavia, os modelos de aprendizado supervisionado desempenharam um papel melhor, com acurácias mínimas e máximas de 89,60% e 91,06%, respectivamente. Quanto aos valores de *F1-score*, foi calculado mínimo de 70,68% para o SVM e máximo de 76,73% na avaliação do CATBoost. Entre os três modelos, é possível afirmar que o *Multi-Layer Perceptron* apresentou desempenho intermediário, destacando-se para o coeficiente de acurácia 90,02% porém *recall* de 67,50%, ou seja, uma boa identificação de pontos normais porém pode ser melhorada a identificação de anomalias.

Por outra perspectiva, a Figura 30 ilustra a taxa de perda dos algoritmos ao longo das iterações durante o treinamento, destacando para um valor menor ao final do aprendizado do CATBoost (0,1282 em relação a 0,2187) e em menos épocas (108 e 151). Além disso, as curvas de Característica de Operação do Receptor (ROC, *Receiver Operating Characteristic*) na Figura 31 comparam as taxas de valores positivos – isto é, identificação de anomalias – entre os modelos SVM, MLP e CATBoost. Com as curvas puxadas mais para o canto superior esquerdo, observa-se um bom desempenho dos classificadores, próximo do valor ideal 1,0: o algoritmo SVM ficou atrás com 0,92, seguido pelo MLP com 0,95 e CATBoost muito próximo com área 0,96. Contudo, vale ressaltar que o modelo SVM não apresentou limites de decisão claros, isto é, um hiperplano que separe bem os dados com desvios ou não.

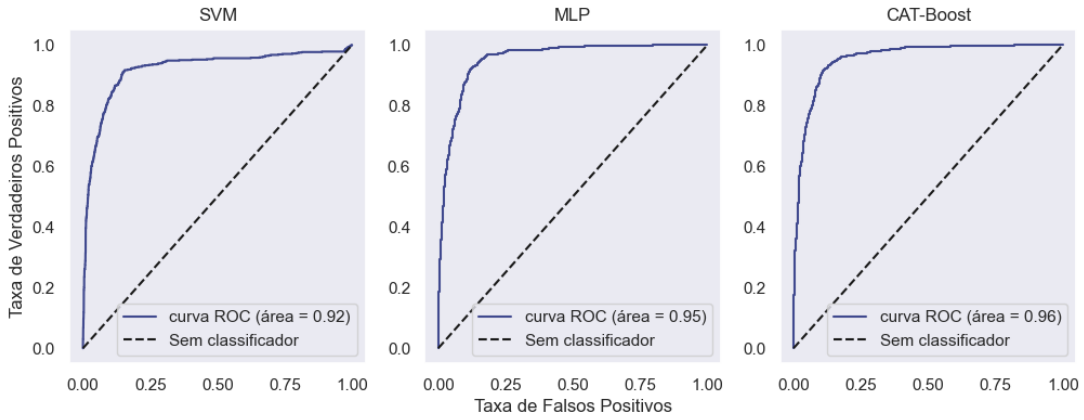
Enfim, o modelo MLP provou-se superior aos algoritmos SVM e K-Means inicialmente propostos, porém o CATBoost tornou-se uma alternativa eficiente também para o trabalho. A Tabela 13 apresenta os coeficientes calculados durante o treinamento de cada algoritmo, bem como seu tempo total de treinamento.

Figura 30 – Curvas de perda de validação em MLP e CATBoost.



Fonte: Elaborada pelo autor.

Figura 31 – Curvas ROC para classificação.



Fonte: Elaborada pelo autor.

Tabela 13 – Comparativo dos resultados.

Modelo	Iterações	Perda	Acurácia	Precisão	Recall	F1-Score	Tempo de Treinamento (s)
SVM	-		89,60%	78,59%	64,22%	70,68%	10,700
MLP	151	0,2187	90,02%	78,40%	67,50%	72,54%	8,800
K-Means	-		60,67%	80,17%	67,87%	73,51%	0,584
CATBoost	108	0,1282	91,06%	78,03%	75,47%	76,73%	0,203
Clusterização Hierárquica	-	-	-	-	-	-	252

Fonte: Elaborada pelo autor.

5 CONCLUSÃO

Durante a elaboração deste trabalho, foram estudados diversas metodologias descritas na literatura a fim de desenvolver um algoritmo de classificação de desvios de qualidade em pontos de solda na indústria automotiva. Com esse propósito, foi estruturada um banco de dados SQL com diversos parâmetros de cada ponto, bem como pontos anômalos previamente rotulados com o suporte de um grupo de técnicos de qualidade especializados, atividade esta que tornou-se a primeira dificuldade do projeto.

Dado o volume e velocidade de armazenamento dos dados, com aproximadamente 390.000 novos eventos inseridos no banco de dados diariamente, inicialmente o projeto foi caracterizado como *Big Data*. Nesse cenário, a catalogação *in loco* de desvios já existentes demandaria muitas horas de trabalho de uma mão-de-obra especializada. Dessa forma, foi escolhida uma granularidade do modelo a nível de componente do produto para viabilizar o aprendizado supervisionado, resultando em 2.570 pontos com desvios coletados em 4 meses. Ao todo, a amostra de treinamento consistiu de 13.110 dados, sendo 10.540 (80%) pontos normais. Embora neste trabalho os modelos de aprendizado de máquina tenham computado uma amostragem simplificada, estes foram considerados como uma espécie de piloto para melhorias futuras de um problema que persiste como *Big Data*. Portanto, considera-se que a infraestrutura criada para armazenamento dos dados tornou-se um ecossistema sustentável a longo prazo, periodicamente relacionando informações de processo (Matuschek) com produto (PLC). Vale dizer que também está sendo avaliada a possibilidade de migração dos dados para um servidor de *Data Lake*, mantendo apenas uma cópia mais recente no banco de dados relacional.

Foram desenvolvidos 3 modelos de classificação com aprendizado supervisionado (SVM, MLP e CATBoost) e 2 de aprendizado não-supervisionado (K-Means e clusterização hierárquica) para comparação de resultados. Ambos algoritmos de K-Means e clusterização hierárquica não apresentaram resultados aceitáveis, com acurácia de 60,67% e um dendograma não conclusivo, respectivamente. Uma hipótese para o baixo desempenho é que a amostra disponibilizada foi pequena para um aprendizado não-supervisionado com muitas variáveis, embora K-Means seja útil quando o processamento computacional é limitado, corroborando com Ezugwu *et al.* (2022). Neste caso, seria mais benéfico utilizar os algoritmos de aprendizado não-supervisionado como métodos exploratórios de variáveis.

Em contrapartida, os modelos SVM, MLP e CATBoost apresentaram acurácias de validação de 89,60%, 90,02% e 91,06%, respectivamente. Embora os três modelos apresentarem resultados muito próximos, o *Support Vector Machine* apresentou as menores métricas de validação, além de seu tempo de treinamento superior a 10 segundos e limites de decisão não definidos. Segundo, Nasir *et al.* (2022), um algoritmo SVM é eficiente

para dados de múltiplas dimensões e poucos exemplos, mas pode ter baixo processamento para grande volume de dados. O *Multilayer Perceptron*, por sua vez, mostrou resultados intermediários. O modelo MLP é constituído por uma arquitetura complexa, possibilitando múltiplas entradas e saídas, por isso tem capacidade de resolver problemas não-lineares complexos e de grande volume de dados. Em contrapartida, também pode apresentar diversos mínimos locais, dificultando a otimização dos pesos (NASIR *et al.*, 2022). Já o CATBoost, antes apresentado apenas como alternativa, foi superior em todas as métricas exceto precisão, apresentando uma predição mais balanceada entre anomalias e pontos normais. Além disso, o modelo CATBoost reduz as transformações dos dados uma vez que recebe dados categóricos e apresentou o menor tempo de treinamento, com 203 milissegundos apenas.

Logo, os experimentos demonstraram que é possível classificar um ponto de solda defeituoso, bem como analisar tendências no seu comportamento: uma atividade essencial para a indústria que hoje é manual, especializada e sensível a experiência do colaborador pode ser padronizada e até otimizada em 37 minutos por dia com predições precisas de modelos MLP e CATBoost. Isso equivale a 4,5% do tempo de ciclo atual ou 6 produtos por dia. Contudo, algumas melhorias são bem-vindas para este trabalho, como por exemplo inclusão de dados de reafiação de eletrodos das ferramentas de solda, relacionando assim a qualidade do ponto de solda à qualidade da ferramenta. Além disso, a implementação do modelo na borda de linha com realimentação das predições – e supervisão constante – geraria uma amostragem maior para treinamento e, portanto, melhora no desempenho, em especial as taxas de *recall*. Visto que há o risco de deixar desvios passarem sem apontamentos pelo controle de qualidade, é preferível que na indústria automotiva o modelo preveja mais falsos positivos (prever pontos normais reais como desvios) ao invés de falsos negativos (prever pontos com desvios reais como normais). Vale dizer que, visando um impacto maior, futuramente um modelo aperfeiçoado poderia ser treinado para ajustar os parâmetros elétricos do controlador em tempo real, evitando até mesmo 50 minutos por dia em retrabalhos de pontos de solda.

REFERÊNCIAS

ACKLEY, D. H.; HINTON, G. E.; SEJNOWSKI, T. J. A learning algorithm for boltzmann machines. **Cognitive Science**, v. 9, n. 1, p. 147–169, 1985.

ANFAVEA. **Anuário da Indústria Automobilística Brasileira 2018**. 2018. 152 p.

AZANK, F. **Clustering — Conceitos básicos, principais algoritmos e aplicações**. 2022. Available at: https://medium.com/turing-talks/clustering-conceitos-b%C3%A1sicos-principais-algoritmos-e-aplica%C3%A7%C3%A3o-ace572a062a9#:~:text=Clustering%2C%20ou%20agrupamento%2C%20consiste%20na,com%20base%20em%20suas%20semelhan%C3%A7as.

BARROS, G. V. P. de *et al.* Eficiência de redes neurais artificiais na classificação de uso e do solo da bacia hidrográfica do rio japaratuba - se. **Revista Brasileira de Meteorologia**, v. 35, 2020.

BENTO, C. **Multilayer Perceptron Explained with a Real-Life Example and Python Code: Sentiment Analysis**. 2021. Available at: https://towardsdatascience.com/multilayer-perceptron-explained-with-a-real-life-example-and-python-code-sentiment-analysis-cb408ee

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. *In*: **Proceedings of the Fifth Annual Workshop on Computational Learning Theory**. New York, NY, USA: Association for Computing Machinery, 1992. (COLT '92), p. 144–152.

BROWNLEE, J. **A Gentle Introduction to the Rectified Linear Unit (ReLU)**. 2020. Available at: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/.

BROWNLEE, J. **How to Use StandardScaler and MinMaxScaler Transforms in Python**. 2020. Available at: https://machinelearningmastery.com/standardscaler-and-minmaxscaler-transforms-in-python/.

CASTILLO, J. **A Soldagem nos Novos Tempos da Indústria Automobilística**. 2016. Available at: https://www.linkedin.com/pulse/soldagem-nos-novos-tempos-da-industria-jose-castillo-1/?originalSubdomain=pt.

CHAUVIN, Y.; RUMELHART, D. E. **Backpropagation**. [*S.l.: s.n.*]: Psychology Press, 1995. 576 p.

COUTINHO, B. **Modelos de Predição | SVM**. 2019. Available at: https://medium.com/turing-talks/turing-talks-12-classifica%C3%A7%C3%A3o-por-svm-f4598094a3f1.

DAUDT, G.; WILLCOX, L. D. **Indústria Automotiva**. [*S.l.*], 2018. 26 p.

EZUGWU, A. E. *et al.* A comprehensive survey of clustering algorithms: State-of-the-art machine learning applications, taxonomy, challenges, and future research prospects. **Engineering Applications of Artificial Intelligence**, v. 110, 2022.

FLORENCE, S. E.; SAMSINGH, R. V.; BABUREDDY, V. Artificial intelligence based defect classification for weld joints. **IOP Publishing Ltd**, v. 402, p. 1–14, 2018.

GONZáLEZ-GONZáLEZ, C. *et al.* Environmental and economic analyses of tig, mig, mag and smaw welding processes. **Metals 2023**, v. 1094, n. 13, p. 21, 2023.

HEBB, D. O. **The Organization of Behavior: A Neuropsychological Theory**. [*S.l.: s.n.*]: Psychology Press, 1949. 378 p.

HINTON, G. E.; OSINDERO, S.; TEH, Y.-W. A fast learning algorithm for deep belief nets. **Neural Comput**, v. 18, n. 7, p. 1527–1554, 2006.

HO, M. *et al.* An artificial neural network approach for parametric study on welding defect classification. **The International Journal of Advanced Manufacturing Technology**, v. 120, p. 527–535, 2022.

JABEUR, S. B. *et al.* Catboost model and artificial intelligence techniques for corporate failure prediction. **Technological Forecasting and Social Change**, v. 166, p. 120658, 2021. ISSN 0040-1625. Available at: https://www.sciencedirect.com/science/article/pii/S0040162521000901.

JAKKULA, V. Tutorial on support vector machine (svm). **School of EECS, Washington State University**, v. 37, p. 3, 2006. Available at: https://course.khoury.northeastern.edu/cs5100f11/resources/jakkula.pdf.

JANIESCH, C.; ZSCHECH, P.; HEINRICH, K. Machine learning and deep learning. **Electronic Markets**, v. 31, n. 3, p. 685–695, 2021.

JOHNSON, N. N. *et al.* Implementation of machine learning algorithms for weld quality prediction and optimization in resistance spot welding. **Journal of Materials Engineering and Performance**, 2023.

KHOKHAR, S. *et al.* A comprehensive overview on signal processing and artificial intelligence techniques applications in classification of power quality disturbances. **Renewable and Sustainable Energy Reviews**, v. 51, p. 1650 – 1663, 2015.

KOHONEN, T. **Self-Organizing Maps**. [*S.l.: s.n.*]: Springer Berlin, Heidelberg, 1982. 502 p.

LARSEN, K. R.; BECKER, D. Automated machine learning for business. **Oxford University Press**, 2019. Available at: https://www.researchgate.net/profile/Kai-Larsen/publication/327477467_Chapter_24_Seven_Types_of_Target_Leakage_in_Machine_Learning_and_an_Exercise/links/5b91889ea6fdccfd541f78ef/Chapter-24-Seven-Types-of-Target-Leakage-in-Machine-Learning-and-an-Exercise.pdf.

LILLICRAP, T. P. *et al.* Backpropagation and the brain. **Nature Reviews Neuroscience**, v. 21, p. 335–346, 2020.

LOPEZ, C. *et al.* An unsupervised machine learning method for discovering patient clusters based on genetic signatures. **Journal of Biomedical Informatics**, v. 85, p. 30–39, 2018.

MAKEDON, V.; MYKHAILENKO, O.; VAZOV, R. Dominants and features of growth of the world market of robotics. **European Journal of Management Issues**, v. 29, n. 3, p. 133–141, 2021.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, v. 5, p. 115 – 133, 1943.

MINSKY, M.; PAPERT, S. Perceptron: an introduction to computational geometry. **The MIT Press, Cambridge, expanded edition**, v. 19, n. 88, p. 2, 1969.

NASIR, N. *et al.* Water quality classification using machine learning algorithms. **Journal of Water Process Engineering**, v. 48, 2022.

PERCIVAL, D. B.; MONDAL, D. 22 - a wavelet variance primer. *In*: Subba Rao, T.; Subba Rao, S.; RAO, C. (ed.). **Time Series Analysis: Methods and Applications**. Elsevier, 2012, (Handbook of Statistics, v. 30). p. 623–657. Available at: https://www.sciencedirect.com/science/article/pii/B9780444538581000223.

ROSENBLATT, F. The perceptron: A probabilistic model for information storage and organization in the brain. **The bulletin of mathematical biophysics**, v. 65, n. 6, p. 386 – 408, 1958.

SAPUTRA, D. M.; SAPUTRA, D.; OSWARI, L. D. Effect of distance metrics in determining k-value in k-means clustering using elbow and silhouette method. **Advances in Intelligent Systems Research**, v. 172, p. 6, 2019.

SREENIVASA, S. **Radial Basis Function (RBF) Kernel: The Go-To Kernel**. 2020. Available at: https://towardsdatascience.com/radial-basis-function-rbf-kernel-the-go-to-kernel-acf0d22c798a.

VARGAS, P. G.; BUNDE, A. Indústria automobilística brasileira: uma análise das principais transformações tecnológicas no sistema produtivo e seu impacto sobre o emprego. **Revista Pegada - A Revista da Geografia do Trabalho**, v. 22, n. 2, p. 49–84, 2021.

WIDROW, B.; HOFF, T. An adaptive "adaline"neuron using chemical "memistors". **Stanford Eletronics Laboratories**, v. 1553, n. 2, 1960.

WU, Y. chen; FENG, J. wen. Development and application of artificial neural network. **Wireless Personal Communications**, v. 102, p. 1645–1656, 2017.

YOU, H.; KIM, C. Machine learning modeling and hyper-parameter optimization for weld nugget formation and failure behavior of resistance spot welds. **Journal of Welding and Joining**, v. 39, n. 6, p. 658–665, 2021.

ZHOU, B. *et al.* Machine learning with domain knowledge for predictive quality monitoring in resistance spot welding. **Journal of Intelligent Manufacturing**, v. 33, p. 1139–1163, 2022.