**UNIVERSIDADE FEDERAL DO CEARÁ**

**CENTRO DE TECNOLOGIA**

**DEPARTAMENTO DE ENGENHARIA METALÚRGICA E DE MATERIAIS**

**PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA E CIÊNCIA DE MATERIAIS**

**MESTRADO ACADÊMICO EM ENGENHARIA E CIÊNCIA DE MATERIAIS**

**JOÃO VICTOR BARROSO XAVIER**

# PREVISÃO COMPUTACIONAL DO TEOR DE SILÍCIO NO FERRO-GUSA

**FORTALEZA**

**2026**

JOÃO VICTOR BARROSO XAVIER

# PREVISÃO COMPUTACIONAL DO TEOR DE SILÍCIO NO FERRO-GUSA

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia e Ciência de Materiais do Programa de Pós-Graduação em Engenharia e Ciência de materiais do Centro de tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia de materiais. Área de Concentração: Processos de transformação e degradação dos materiais

Orientador: Prof. Dr. Elineudo Pinho de Moura

Coorientador: Prof. Dr. Guilherme de Alencar Barreto

FORTALEZA

2026

JOÃO VICTOR BARROSO XAVIER

PREVISÃO COMPUTACIONAL DO TEOR DE SILÍCIO NO FERRO-GUSA

Dissertação apresentada ao Curso de Mestrado Acadêmico em Engenharia e Ciência de Materiais do Programa de Pós-Graduação em Engenharia e Ciência de materiais do Centro de tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de mestre em Engenharia de materiais. Área de Concentração: Processos de transformação e degradação dos materiais

Aprovada em: 23/02/2026

BANCA EXAMINADORA

---

Prof. Dr. Elineudo Pinho de Moura (Orientador)
Universidade Federal do Ceará (UFC)

---

Prof. Dr. Guilherme de Alencar Barreto (Coorientador)
Universidade Federal do Ceará (UFC)

---

Prof. Dr. Jeferson Leandro Klug
Universidade Federal do Ceará (UFC)

---

Dra. Raphaella Hermont Fonseca Murta
ArcelorMittal Pecém (AMP)

A Deus.

Aos meus pais, José Nazareno e Geovana Barroso.

## AGRADECIMENTOS

Agradeço à minha família, em especial aos meus pais, Geovana Barroso Xavier e José Nazareno Crispim Xavier por todo apoio e carinho. Me encorajam a persistir sempre.

Agradeço aos meus colegas do centro de ensaios não destrutivos (CENDE) pelas discussões e risadas durante todo o meu mestrado. Ambas foram de grande valia, tanto para gerar novas ideias quanto para tornar a jornada mais leve.

Agradeço ao Prof. Dr. Emanuel Seixas Campos pelas discussões no laboratório e pela oportunidade de fazer parte do programa PROPAG/EIDEIA. Ambos foram fundamentais para a minha formação.

Agradeço ao Prof. Dr. Guilherme de Alencar Barreto pela paciência e pelos grandes ensinamentos. Suas aulas me ensinaram a ter uma visão mais ampla sobre tópicos relacionados à área de reconhecimento de padrões e *machine learning*.

Agradeço ao Prof. Dr. Elineudo Pinho de Moura pela excelente orientação e conselhos. As reflexões e dúvidas que o senhor promoveu foram vitais para a execução do trabalho. Além disso, me incentivou a buscar sentido e ter cautela quanto aos resultados gerados pelos modelos computacionais.

Finalmente, agradeço à fundação cearense de apoio ao desenvolvimento científico e tecnológico (FUNCAP) pelo apoio financeiro prestado durante a realização do presente trabalho.

“O passo mais importante que um homem pode dar. Não é o primeiro, é? É o próximo.”

(Brandon Sanderson)

## RESUMO

O teor de silício do ferro-gusa é um dos principais indicadores da sua qualidade e do estado térmico do alto-forno. Além disso, um elevado teor de silício pode danificar os equipamentos industriais, levando à necessidade de realizar manutenções e consequentemente, à perda de eficiência do processo. Por essas razões, diversos estudos vêm sendo desenvolvidos ao longo de décadas para prever e monitorar o teor de silício no ferro-gusa, sugerindo o uso de modelos guiados por dados (*data-driven models*). Nesse contexto, o presente trabalho testou modelos como o *perceptron* logístico, redes neurais artificiais do tipo *perceptron* de múltiplas camadas com até duas camadas ocultas e diferentes versões de máquinas de vetores suporte adaptadas para regressão (SVR, TSVR, LSSVR) a fim de realizar a tarefa de previsão do teor de silício no ferro-gusa. Uma técnica para estimar o número de neurônios ocultos nas redes neurais baseada em decomposição de valores singulares (SVD) também foi investigada com o intuito de reduzir o tempo de ajuste e custo computacional. Entre os modelos baseados em neurônios, a rede neural com uma camada oculta apresentou o melhor balanço entre performance e custo computacional, enquanto a técnica baseada em SVD proporcionou uma janela de teste de hiperparâmetros menor. Vale ainda salientar que o LSSVR apresentou o menor erro de previsão entre todos os modelos testados, logo, ele foi utilizado em uma análise de sensibilidade para estudar a influência de cada variável de entrada sobre o teor de silício no ferro-gusa.



**Palavras-chave:** teor de silício; alto-forno; previsão; redes neurais artificiais; máquinas de vetores suporte.

## ABSTRACT

The silicon content of pig iron is one of the main indicators of its quality and the thermal state of the blast furnace. Furthermore, a high silicon content can damage industrial equipment, leading to the need for maintenance and, consequently, a loss of process efficiency. For these reasons, several studies have been developed over decades to predict and monitor the silicon content in pig iron, suggesting the use of data-driven models. In this context, this work tested models such as the logistic perceptron, multilayer perceptron artificial neural networks with up to two hidden layers, and different versions of support vector machines adapted for regression (SVR, TSVR, LSSVR) to predict the silicon content in pig iron. A technique for estimating the number of hidden neurons in neural networks based on singular value decomposition (SVD) was also investigated to reduce tuning time and computational cost. Among the neuron-based models, the neural network with one hidden layer presented the best balance between performance and computational cost, while the SVD-based technique provided a smaller hyperparameter testing window, therefore, it was used in a sensitivity analysis to study the influence of each input variable on the silicon content in pig iron.



**Keywords:** silicon content; blast furnace; prediction; artificial neural networks; Support vector machines

## LISTA DE FIGURAS

## LISTA DE TABELAS

**SUMÁRIO**

# 1 INTRODUÇÃO

O aço é um produto amplamente utilizado pela sociedade e possui uma grande participação na economia global. O aço é produzido em plantas siderúrgicas de diferentes países a partir do refino do ferro-gusa, o qual, por sua vez, é obtido a partir da redução do minério de ferro em um reator metalúrgico conhecido como alto-forno. Conforme Senesi *et al.* (2020), o principal objetivo dos profissionais responsáveis pela operação do alto-forno é produzir ferro-gusa com baixo custo e alta qualidade, a qual só pode ser monitorada pela teor de alguns elementos químicos, tais como manganês, fósforo, enxofre e silício.

A mudança no teor de silício reflete o estado térmico do alto-forno e desempenha um papel fundamental na estabilidade de operação e controle de reações no alto-forno, além de ser uma garantia necessaria para alcançar a conservação de energia e reduzir emissões (Song *et al.*, 2023). Adicionalmente, se o teor de silício do ferro-gusa durante a operação do conversor a oxigênio (BOF) for alto (maior do que 0.7%), a viscosidade da escória irá aumentar devido ao aumento da polimerização, o que aumenta a tendência de vazamentos (Barella *et al.*, 2016). Diante do exposto, ressalta-se a importância de prever o teor de silício durante a operação do alto-forno.

A previsão do teor de silício no ferro-gusa é uma tarefa que pode ser realizada a partir da construção de modelos guiados por dados (*data driven*) obtidos da matéria-prima e das condições operacionais do alto-forno. Entretanto, devido à natureza complexa do processo de redução, modelos tradicionais não conseguem obter resultados satisfatórios e frequentemente é necessário buscar técnicas mais sofisticadas. Estudos mais recentes sobre este tópico concentram seus esforços na utilização de modelos baseados em aprendizado de máquina e têm obtido relativo sucesso.

Em síntese, o presente trabalho buscou prever o teor de silício no ferro-gusa utilizando modelos computacionais, tais como regressão linear múltipla, *perceptron* logístico, redes neurais artificiais do tipo *perceptron* de múltiplas camadas com até duas camadas ocultas e diferentes tipos de máquinas de vetores suporte com o intuito de conseguir o menor erro de previsão possível. Além disso, uma técnica baseada em SVD para estimar o número de neurônios nas camadas ocultas das redes neurais foi investigada para reduzir o tempo de busca por hiperparâmetros ótimos. Adicionalmente, também buscou-se realizar uma análise de sensibilidade nas variáveis de entrada do modelo para verificar se as relações entre cada uma delas e o teor de silício condiz com o esperado teoricamente.

# 2 FUNDAMENTAÇÃO TEÓRICA

Esta seção possui o objetivo de fornecer para o leitor, uma base teórica sobre os conceitos utilizados no decorrer deste trabalho. Serão abordados temas ligados à redução do minério de ferro no alto-forno e a importância do teor de silício neste processo, algoritmos como a regressão linear múltipla, o *perceptron* logístico, redes neurais artificiais e regressão de vetores suporte. Outros temas relacionados ao trabalho desenvolvido, tais como decomposição de matrizes em valores singulares, multicolinearidade e parâmetros para a comparação de modelos regressores também estão presentes nesta seção.

## 2.1 A importância do teor de silício para a indústria siderúrgica

O silício é um elemento químico com ampla importância para a indústria siderúrgica, seja nas etapas de redução ou refino. No âmbito da operação do alto-forno, o silício é essencial para a avaliação do seu estado térmico e pode refletir na qualidade do aço produzido posteriormente (Cardoso *et al.*; Lakshmanan *et al.*, 2022, 2023). Além disso, uma operação com baixo teor de silício possui grandes vantagens como a redução do consumo de combustível e aumento da produção de ferro-gusa, pois para uma diminuição de 0.1% de Si, há aproximadamente uma redução de até 7kg de combustível por tonelada de gusa produzida e um aumento em até 0.6% na sua produção (Zhang *et al.*, 2026).

No conversor a oxigênio (BOF), se o teor de silício for maior do que 0.7%, o ferro-gusa líquido precisará passar por um processo duplo de remoção da escória, no qual a primeira remoção busca retirar escória com elevada viscosidade (Barella *et al.*, 2016). Uma escória com essa característica possui permeabilidade prejudicada e conforme Donayo, R. *et al.* (2010), isso pode levar a situações em que ocorrerá vazamento de material para fora do conversor a oxigênio. Quando um vazamento ocorre, a lança de oxigênio, o coletor de gases acima do conversor e o maquinário responsável pela coleta da escória podem ser danificados (Wang *et al.*, 2023).

Diante do exposto, é interessante controlar o teor de sílicio no processo siderúrgico desde o alto-forno. Modelos com o intuito de prever o teor de silício no ferro-gusa já foram propostos ao longo das décadas. Inicialmente, eles eram baseados apenas no conhecimento termodinâmico do processo, mas, com o avanço do poder de processamento computacional, a captação de sinais via sensores e o desenvolvimento de algoritmos de aprendizado de máquina, os modelos passaram a ser cada vez mais guiados pelos dados coletados durante a fase de

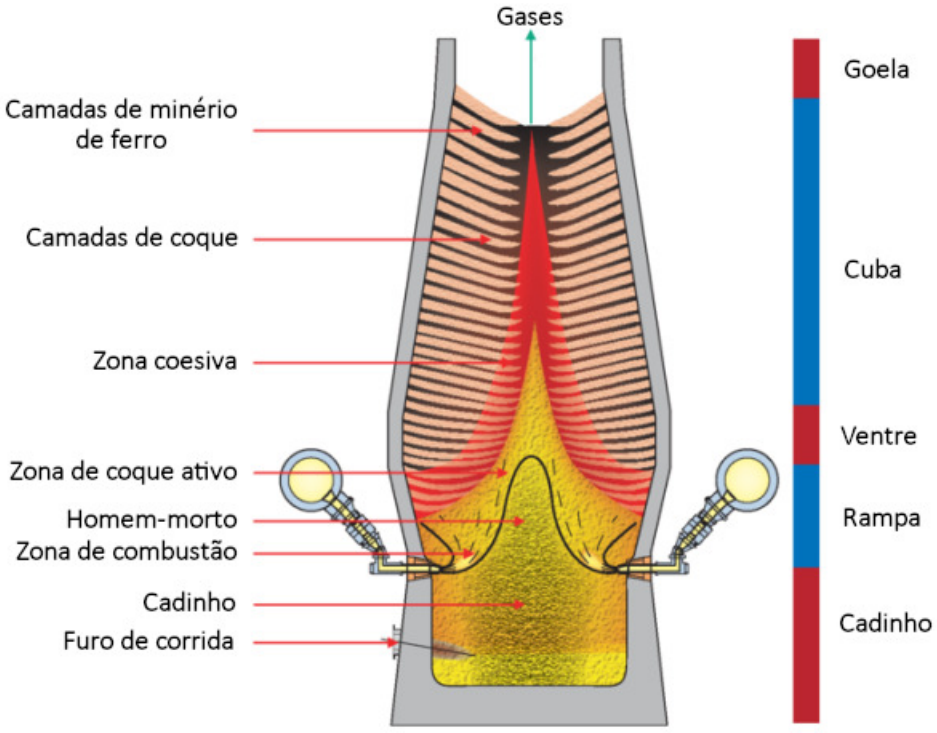
operação do alto-forno. Nesse contexto, o tema continua sendo relevante para o setor industrial e pesquisadores continuam propondo novos modelos.

O teor de silício no ferro-gusa depende das condições de operação do alto-forno, das suas dimensões, das matérias-primas com as quais ele é carregado e das necessidades imediatas da usina siderúrgica, logo, cada alto-forno constitui um caso independente (Mei *et al.*, 2020). Além disso, qualquer modelo direcionado para a previsão do teor de silício deve passar por revisões periódicas, uma vez que quaisquer intervenções por motivo de manutenção ou substituição de componentes podem alterar as condições internas do alto-forno.

## 2.2 Processo de redução em altos-fornos

Antes de detalhar como o silício é incorporado no ferro-gusa, é necessário entender o processo de redução do minério de ferro, pois é no decorrer dele que ocorrem os mecanismos de incorporação. Esse processo acontece no interior do alto-forno, o qual pode ser definido como um grande reator metalúrgico em contra-corrente por onde o minério de ferro e o coque são alimentados pelo topo e descem enquanto os gases redutores ascendem (Suopajärvi *et al.*, 2013). Além do ferro-gusa, também é produzida a escória, que pode ser utilizada como coproduto pela indústria cimenteira, e gases que precisam ser devidamente tratados.

Figura 1 – Representação esquemática das regiões internas do alto-forno.




Fonte: Adaptada de Geerdes *et al.* (2020).

A Figura 1 ilustra as regiões internas do alto-forno de forma simplificada. Inicialmente, o alto-forno é carregado em camadas alternadas de coque e minério de ferro, ambos no estado sólido, por meio de um sistema de distribuição localizado no topo. Uma vez no interior do alto-forno, a carga encontra-se na zona granular (*lumpy zone*), onde o padrão de camadas alternadas de minério de ferro (*burden layers*) e coque (*coke slits*) permanece até a eventual fusão da carga. Nesse ponto, é válido salientar que as camadas de coque são importantes, pois elas proporcionam caminhos pelos quais os gases redutores podem ascender (Cameron *et al.*, 2020). Além disso, a temperatura no interior do alto-forno aumenta conforme a carga se aproxima do nível do sistema de ventaneiras (*tuyere system*). Logo, no início da zona granular, já ocorre o pré-aquecimento da carga.

Conforme a carga desce e avança na zona granular, ocorre a redução do minério de ferro. Segundo Geerdes *et al.* (2020), o minério de ferro é reduzido por conta dos gases ascendentes, os quais, em contrapartida, perdem calor para a carga à medida que se aproximam do topo do alto-forno. As reações a seguir mostram que o minério de ferro carregado na forma de hematita ($Fe_2O_3$) passa por transformações intermediárias até a obtenção do ferro metálico.

$$3Fe_2O_{3(s)} + CO_{(g)} \rightarrow 2Fe_3O_{4(s)} + CO_{2(g)} \tag{2.1}$$

$$Fe_3O_{4(s)} + CO_{(g)} \rightarrow 3FeO_{(s)} + CO_{2(g)} \tag{2.2}$$

$$FeO_{(s)} + CO_{(g)} \rightarrow Fe_{(s)} + CO_{2(g)} \tag{2.3}$$

As transformações da hematita para magnetita ($Fe_3O_4$) e desta para a wustita ($FeO$) ocorrem por volta de 600 a 900ºC e a redução desta última para o ferro metálico ocorre entre 900 e 1100ºC. Em temperaturas superiores, o dióxido de carbono ($CO_2$) pode reagir diretamente com o coque, produzindo monóxido de carbono ($CO$) novamente, conforme a Equação 2.4, a qual é conhecida como reação de Boudoard. A reação total, representada pela Equação 2.5, pode ser obtida unindo a reação de Boudoard com a reação de redução indireta da wustita para a ferrita por meio da lei de Hess (Babich e Senk, 2015).

$$CO_{2(g)} + C_{(s)} \rightarrow 2CO_{(g)} \tag{2.4}$$

$$FeO_{(s)} + C_{(s)} \rightarrow Fe_{(s)} + CO_{(g)} \tag{2.5}$$

A Equação 2.5 corresponde à reação de redução direta da wustita. O balanço na proporção de material reduzido de forma direta ou indireta é fundamental para a operação do alto-forno. A reação de redução direta usa o carbono do coque e gera gás CO ao custo de gastar

muita energia, logo, os operadores experientes estão cientes de que assim que a taxa de redução direta ou a perda de solução (quantidade de coque usada para a solução) aumentam, a descida da carga irá acelerar e o interior do forno irá resfriar (Geerdes *et al.*, 2020).

Eventualmente, parte da carga amolece e funde, formando a zona coesiva (*cohesive zone*), localizada imediatamente abaixo da zona granular. Segundo Babich *et al.* (2008), a zona coesiva consiste em camadas de coque permeáveis intercaladas com camadas de material da carga semi-fundidos viscosos e ferro, que resistem à passagem de gás. Quando a carga funde totalmente na parte inferior da zona coesiva, ocorre a separação entre o metal e a escória, a qual consiste de minério de ferro não reduzido e outros materiais da ganga (Ghosh *et al.*, 2017).

Após a fusão, o metal e a escória líquidos começam a percolar pelos interstícios do coque no estado sólido na chamada zona de gotejamento (*dripping zone*) ou zona de coque ativo (*active coke zone*), a qual pode ser descrita como um leito de coque compactado. Boa parte do coque que chega até esta região acaba rolando em direção às ventaneiras. Além disso, é importante salientar que grande parte da dissolução do carbono, da redução da wustita (FeO) na escória, das reações heterogêneas de transferência de metalóides como Si e Mn e da transferência de calor para o líquido ocorrem na zona de gotejamento (Ghosh *et al.*, 2017).

Abaixo da zona de coque ativo há a zona de coque estagnado ou homem-morto (*deadman*). Essa zona consiste em uma coluna de coque presente no cadinho (*hearth*). Conforme Zong *et al.* (2024), o homem-morto possui o papel de promover sustentação da coluna de material, atuar como agente cementante e conduzir o fluxo de ferro e escória. O homem-morto é constantemente renovado com coque na sua região superior e consumido lentamente na sua porção inferior, levando a um tempo de residência relativamente longo.

Na zona de combustão (*raceway*), localizada na frente das ventaneiras, são injetados ar quente e combustíveis auxiliares no alto-forno. O ar quente, produzido nos regeneradores, reage com o carbono do coque produzindo dióxido de carbono, o qual reage com o carbono do coque novamente produzindo o gás redutor, conforme as reações abaixo

$$O_{2(g)} + C_{(s)} \rightarrow CO_{2(g)} \tag{2.6}$$

$$CO_{2(g)} + C_{(s)} \rightarrow 2CO_{(g)} \tag{2.7}$$

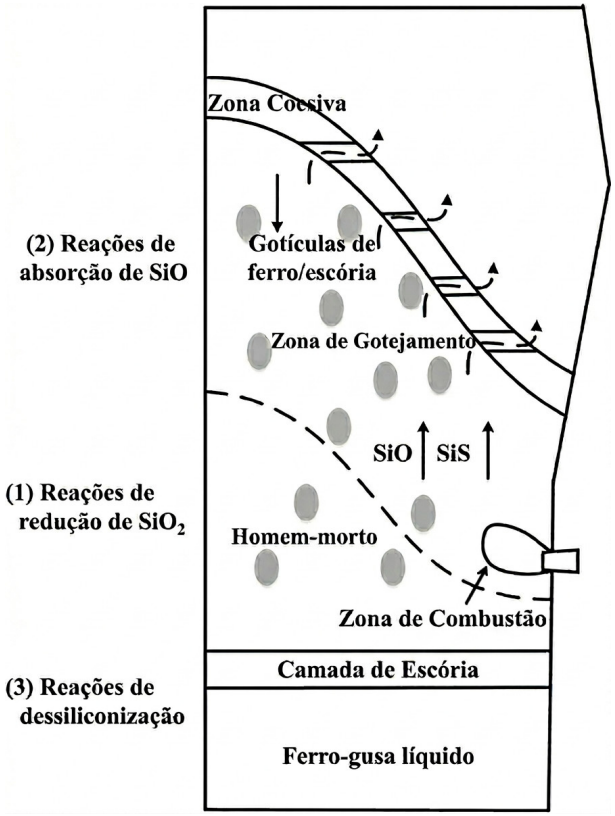
Devido à oxidação do coque, há a liberação de uma grande quantidade de calor, o que torna as temperaturas da zona de combustão as mais elevadas durante todo o processo. Os combustíveis auxiliares, tais como carvão pulverizado, gás natural e hidrogênio, suplementam o coque no alto-forno, diminuindo os custos da produção de ferro-gusa (Chatterjee *et al.*, 2026).

No cadinho (*hearth*), o ferro-gusa e a escória são vazados para fora do alto-forno. Devido à diferença de densidade, a escória permanece acima do metal, o que permite a separação de forma simples. O ferro-gusa, ainda no estado líquido é despejado em carros-torpedo, os quais são encarregados de realizar o transporte até a aciaria, onde o ferro-gusa será transformado em aço. A escória é resfriada e comumente torna-se coproduto para a indústria cimenteira.

## 2.3 Mecanismos de incorporação de silício no ferro-gusa

A literatura frequentemente reporta que as principais fontes de silício no alto-forno são provenientes, na forma de silica ($SiO_2$), da ganga do minério de ferro e das cinzas de ambos o coque e carvão pulverizado, sendo que 70% do silício encontrado no ferro-gusa é oriundo das cinzas do coque (Mei *et al.*, 2020). No interior do alto-forno, o silício liberado pelas fontes citadas passa por uma série de reações químicas responsáveis pela dinâmica de incorporação no ferro-gusa, a qual possui algumas etapas. A Figura 2 ilustra os mecanismos de incorporação do silício no alto-forno de forma simplificada.

Figura 2 – Representação esquemática dos mecanismos de incorporação de silício no ferro-gusa




Fonte: adaptada de Mei *et al.* (2020).

Segundo Ghosh e Chatterjee (2008), há reações envolvendo o silício nas proximida-

des da zona de combustão, na região da rampa (*bosh*) do alto-forno e no cadinho. Geralmente, o $SiO_{(g)}$ é citado como o principal meio gasoso para o transporte de silício. A dinâmica da incorporação envolve a produção de $SiO_{(g)}$ na zona de combustão, o transporte desse gás, a sua dissolução no ferro-gusa e a mistura de ferro-gusa no cadinho do alto-forno (Hage *et al.*, 2022).

Segundo Mei *et al.* (2020), o $SiO_{(g)}$ é produzido por meio da redução da sílica nas proximidades das ventaneiras, conforme as reações abaixo:

$$SiO_{2(s,l)} + C_{(s)} \rightarrow SiO_{(g)} + CO_{(g)} \tag{2.8}$$

$$SiO_{2(s,l)} + CO_{(g)} \rightarrow SiO_{(g)} + CO_{2(g)} \tag{2.9}$$

$$SiC_{(s)} + CO_{(g)} \rightarrow SiO_{(g)} + 2C_{(s)} \tag{2.10}$$

$$2SiO_{2(s)} + SiC_{(s)} \rightarrow 3SiO_{(g)} + CO_{(g)} \tag{2.11}$$

Observando o conjunto de reações acima, é possível perceber que, apesar de ser a principal fonte de silício no processo, a sílica não é a única a contribuir para a formação do $SiO_{(g)}$ , a exemplo do $SiC_{(s)}$. Uma vez gerado, o $SiO_{(g)}$ ascende até a zona de gotejamento e entra em contato com o metal líquido que está descendo. Esse contato ocorre segundo a reação abaixo:

$$SiO_{(g)} + [C] = [Si] + CO_{(g)} \tag{2.12}$$

onde os elementos químicos entre colchetes $[.]$ estão incorporados no ferro-gusa.

Existe ainda a possibilidade do $SiO_{(g)}$ ser incorporado na escória, o que pode acontecer segundo as reações abaixo:

$$SiO_{(g)} + CO_{2(g)} \rightarrow (SiO_2) + CO_{(g)} \tag{2.13}$$

$$SiO_{(g)} + CO_{(g)} \rightarrow (SiO_2) + C_{(s)} \tag{2.14}$$

onde as espécies químicas entre parênteses $(.)$ se encontram incorporadas na escória. Abaixo do nível das ventaneiras, Hage *et al.* (2022) afirmam que no momento em que as gotas de metal líquido atravessam a camada de escória, há uma reoxidação do silício segundo a reação abaixo:

$$[Si] + 2(FeO) \rightarrow (SiO_2) + 2[Fe] \tag{2.15}$$

Dessa forma, o teor de silício no ferro-gusa começa a aumentar a partir da zona de gotejamento, atinge o seu máximo na mesma altura das ventaneiras e em seguida, volta a diminuir. É possível perceber que o teor de silício do ferro-gusa depende de fatores que são alterados direta e indiretamente por parâmetros operacionais do alto-forno, os quais podem ser

utilizados para criar modelos de previsão computacionais. Diante do exposto, é interessante que tais modelos tenham um bom balanço entre precisão, simplicidade e explicabilidade. A seguir, serão apresentados os fundamentos teóricos dos modelos computacionais utilizados no presente trabalho.

## 2.4 Modelos de regressão

Em uma tarefa de regressão deseja-se prever uma variável de interesse, denotada por $\mathbf{y} \in \mathbb{R}^{N\times 1}$, utilizando-se uma ou mais variáveis de entrada ou regressoras. A coletânea de variáveis reunidas para a tarefa é denotada pela matriz $\mathbf{X} \in \mathbb{R}^{N\times p}$, onde $N$ e $p$ representam as quantidades de observações e de variáveis regressoras, respectivamente. Na área de reconhecimento de padrões e correlatas, cada variável regressora também é denominada atributo ou característica de $\mathbf{X}$, a qual é composta de $N$ vetores de atributos do tipo $\mathbf{x}_i = [x_1 \;\; x_2 \;\; \cdots \;\; x_p]$. Além disso, frequentemente será necessário incluir um vetor coluna com todos os elementos iguais a um à matriz $\mathbf{X}$ no desenvolvimento teórico dos modelos que serão discutidos a seguir. Essa matriz será chamada de $\tilde{\mathbf{X}}$. Essa nomenclatura será seguida adiante.

A relação entre a variável-alvo e os atributos pode ser descrita por meio da expressão abaixo:

$$\mathbf{y} = f(X) \tag{2.16}$$

onde $f(.)$ representa uma função geralmente desconhecida. O grau de complexidade de $f(.)$ pode variar deste uma simples função linear até funções que não podem ser encontradas facilmente, levando à necessidade de estimar diversos parâmetros. Dessa forma, os modelos de regressão utilizados no presente trabalho serão apresentados a seguir, em ordem crescente de complexidade.

### *2.4.1 Regressão linear múltipla (RLM)*

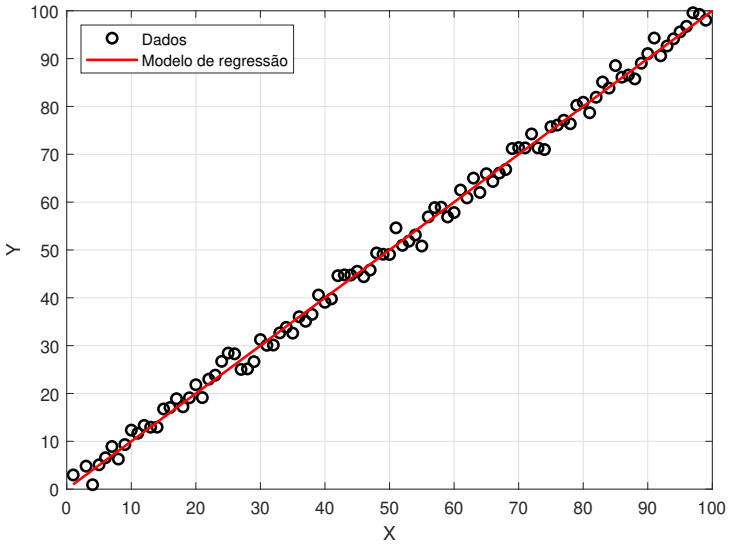
A regressão linear múltipla (RLM) é um dos modelos mais simples para uma tarefa de regressão. Esse modelo assume que a relação entre as variáveis de entrada e a saída de um sistema é função linear de seus parâmetros. A Figura 3 ilustra essa ideia.

Matematicamente, essa ideia é expressa por meio da equação abaixo:

$$\mathbf{y} = \tilde{\mathbf{X}}\mathbf{w} + \boldsymbol{\varepsilon} \tag{2.17}$$

onde $\mathbf{y} \in \mathbb{R}^{N\times 1}$, $\tilde{\mathbf{X}} \in \mathbb{R}^{N\times(p+1)}$, $\mathbf{w} \in \mathbb{R}^{(p+1)\times 1}$ e $\boldsymbol{\varepsilon} \in \mathbb{R}^{N\times 1}$ representam o vetor de observações da variável alvo, a matriz de vetores de atributos com um vetor coluna unitário de dimensões

Figura 3 – Exemplo de relação linear entre duas variaveis




Fonte: Autor (2026)

apropriadas adicionado, o vetor de coeficientes ou parâmetros da regressão e o ruído, respectivamente.

Nesse contexto, é possível realizar previsões da forma

$$\hat{\mathbf{y}} = \tilde{\mathbf{X}}\hat{\mathbf{w}} \tag{2.18}$$

onde:

$$\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{bmatrix}_{N\times 1} \qquad \tilde{\mathbf{X}} = \begin{bmatrix} 1 & \mathbf{x}_1 \\ 1 & \mathbf{x}_2 \\ \vdots & \vdots \\ 1 & \mathbf{x}_N \end{bmatrix}_{N\times(p+1)} \qquad \hat{\mathbf{w}} = \begin{bmatrix} \hat{w}_0 \\ \hat{w}_1 \\ \vdots \\ \hat{w}_p \end{bmatrix}_{(p+1)\times 1}$$

Os vetores $\hat{\mathbf{y}}$ e $\hat{\mathbf{w}}$ carregam os valores de predição de cada amostra e os coeficientes da regressão linear, respectivamente. A partir da equação 2.18, nota-se que é preciso determinar $\hat{\mathbf{w}}$ para obter $\hat{\mathbf{y}}$. É possível encontrar $\hat{\mathbf{w}}$ por meio do método dos mínimos quadrados ordinários, ou em inglês *ordinary least squares* (OLS). Nesse método, o objetivo é encontrar a melhor reta ou hiperplano que diminua o erro quadrado entre os valores previstos e observados da variável alvo. Matematicamente, isso equivale a minimizar a seguinte função-custo:

$$J(\varepsilon) = \|\varepsilon\|^2 = \varepsilon^T\varepsilon = (\mathbf{y}-\hat{\mathbf{y}})^T(\mathbf{y}-\hat{\mathbf{y}}) = (\mathbf{y}-\tilde{\mathbf{X}}\hat{\mathbf{w}})^T(\mathbf{y}-\tilde{\mathbf{X}}\hat{\mathbf{w}}) \tag{2.19}$$

desenvolvendo a função-custo acima, tem-se:

$$J(\tilde{\mathbf{X}},\hat{\mathbf{w}}) = \mathbf{y}^T\mathbf{y} - \mathbf{y}^T\tilde{\mathbf{X}}\hat{\mathbf{w}} - (\tilde{\mathbf{X}}\hat{\mathbf{w}})^T\mathbf{y} + (\tilde{\mathbf{X}}\hat{\mathbf{w}})^T(\tilde{\mathbf{X}}\hat{\mathbf{w}})$$

$$J(\tilde{\mathbf{X}},\hat{\mathbf{w}}) = \mathbf{y}^T\mathbf{y} - 2\mathbf{y}^T\tilde{\mathbf{X}}\hat{\mathbf{w}} + (\tilde{\mathbf{X}}\hat{\mathbf{w}})^T(\tilde{\mathbf{X}}\hat{\mathbf{w}}) \tag{2.20}$$

onde $[(\tilde{\mathbf{X}}\hat{\mathbf{w}})^T\mathbf{y}]^T = \mathbf{y}^T(\tilde{\mathbf{X}}\hat{\mathbf{w}})$. Derivando a Equação 2.20 em relação a $\hat{\mathbf{w}}$ e igualando-a a zero, o desenvolvimento continua como abaixo:

$$\begin{aligned}
&\frac{\partial J(\tilde{\mathbf{X}},\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{y}^T\tilde{\mathbf{X}} + 2\hat{\mathbf{w}}^T(\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}) = 0 \\
&\hat{\mathbf{w}}^T(\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}) = \mathbf{y}^T\tilde{\mathbf{X}} \\
&\hat{\mathbf{w}}^T = \mathbf{y}^T\tilde{\mathbf{X}}(\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1} \\
&\hat{\mathbf{w}} = (\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\mathbf{y}
\end{aligned} \tag{2.21}$$

Frequentemente em problemas reais, a matriz $\tilde{\mathbf{X}}$ é singular, o que significa que o seu determinante é igual a zero. Isso acontece quando um ou mais atributos são colineares, ou seja, linearmente dependentes. Para contornar essa situação, costuma-se utilizar o estimador de mínimos quadrados regularizado, o qual altera a função-custo da Equação 2.19 para a seguinte forma:

$$J(\varepsilon,\mathbf{w}) = \|\varepsilon\|^2 + \lambda\|\mathbf{w}\|^2 \tag{2.22}$$

onde foram introduzidas a norma do vetor de parâmetros $\mathbf{w}$ e uma constante de regularização $\lambda$ que assume valores pequenos. Esse método, conhecido como regularização de thikonov busca minimizar ambas as normas quadráticas do erro $\|\varepsilon\|^2$ e do vetor de parâmetros $\|\mathbf{w}\|^2$. Procedendo da mesma forma que anteriormente, é possível determinar um novo estimador para o vetor de parâmetros:

$$\hat{\mathbf{w}} = (\tilde{\mathbf{X}}^T\tilde{\mathbf{X}} + \mathbf{I}\lambda)^{-1}\tilde{\mathbf{X}}^T\mathbf{y} \tag{2.23}$$

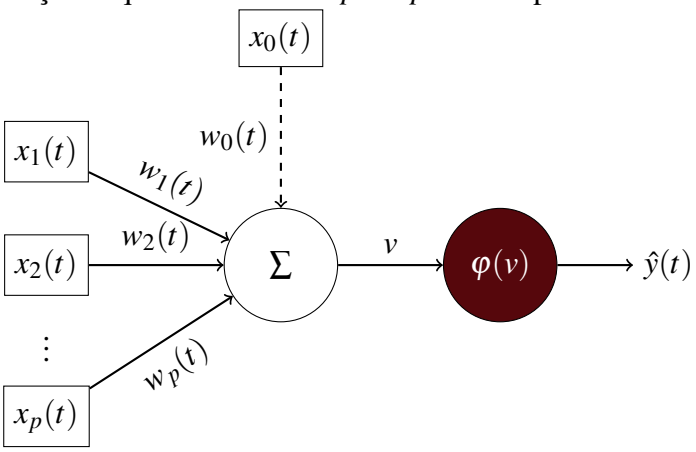
onde $\mathbf{I}$ é uma matriz identidade com as mesmas dimensões de $\tilde{\mathbf{X}}^T\tilde{\mathbf{X}}$. Quando utiliza-se a Equação 2.23 em conjunto com a Equação 2.18, costuma-se chamar o modelo de regressão de cumeeira ou *ridge regression*.

Segundo Montgomery *et al.* (2012), geralmente é difícil comparar os elementos do vetor de coeficientes $\mathbf{w}$ devido à diferença de magnitude entre eles. Dessa forma, é possível optar por normalizar o conjunto de dados, obtendo coeficientes de regressão adimensionais, os quais ajudam a melhorar a interpretabilidade da importância de cada atributo. Apesar disso, Montgomery *et al.* (2012) alertam que a importância dos atributos deve ser analisada com cuidado, visto que amostras diferentes podem levar a conclusões diferentes.

### *2.4.2 Perceptron logístico (PL)*

As origens desse modelo remontam à década de 50, quando Frank Rosenblatt propôs o *perceptron* simples. A Figura 4 abaixo apresenta uma representação esquemática do modelo proposto por ele.

Figura 4 – Representação esquemática de um *perceptron* simples



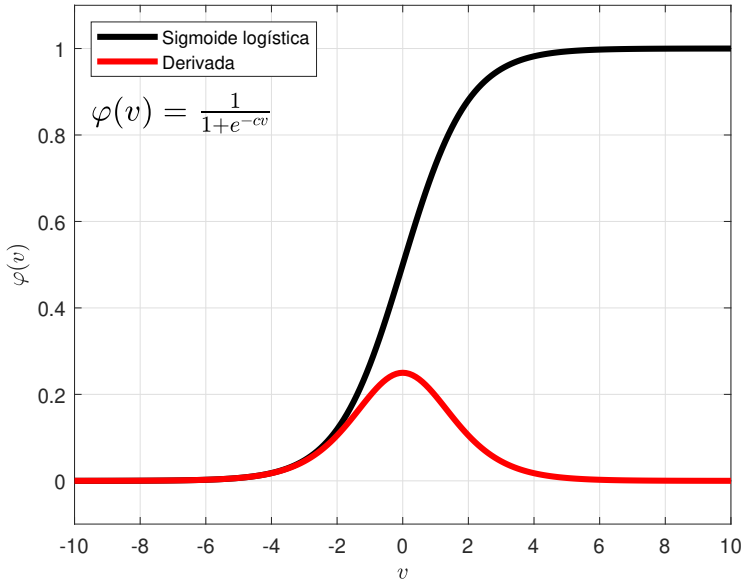

Fonte: Autor (2026)

onde $x_j(t)$ e $w_j(t)$ representam a entrada do $j$-ésimo atributo no *perceptron* simples no instante $t$ e os pesos sinápticos, isto é, os parâmetros livres que precisam ser aprendidos para obter o valor previsto $\hat{y}(t)$, respectivamente. O elemento agregador $\sum$ é um combinador linear cuja saída $v$, conhecida como potencial de ativação, é utilizada como a entrada da função de ativação $\varphi(.)$, que, no caso do *perceptron* simples, pode ser definida como uma função degrau ou degrau bipolar. O elemento $x_0$ atua como um limiar de ativação ou *bias*, usualmente igual a um. Matematicamente, é possível expressar o *perceptron* simples como:

$$\hat{y}(t) = \begin{cases} 1, \ \ se \ \ v(t) \geq 0 \\ -1, \ \ se \ \ v(t) < 0 \end{cases} \tag{2.24}$$

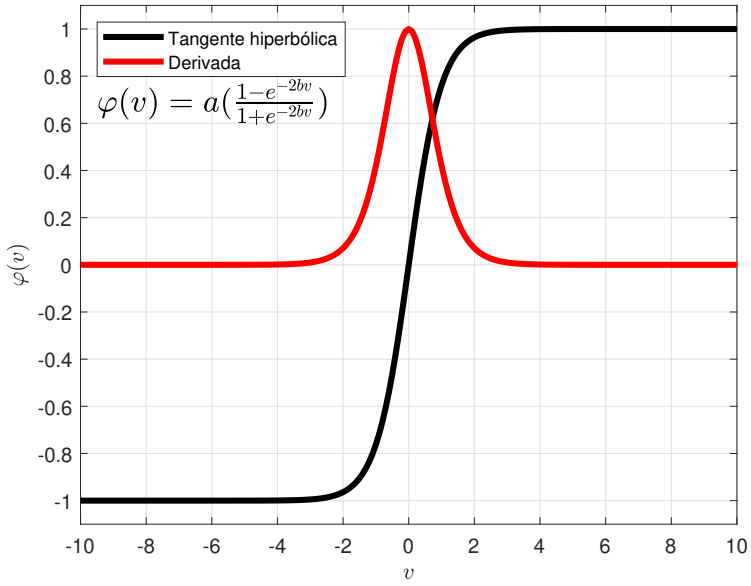
O *perceptron* simples pode ser utilizado em problemas de classificação onde as classes dos dados são linearmente separáveis. Ao substituir a função degrau por outras funções de ativação com inclinações mais suaves, tais como a tangente hiperbólica ou a sigmoide logística destacadas nas figuras 5 e 6. Com isso, o modelo passa a se chamar *perceptron* logístico. Segundo Haykin (2009) ambas as funções possuem constantes positivas que alteram as suas características. No presente trabalho, as constantes da tangente hiperbólica serão chamadas de $a$ e $b$, enquanto a constante presente na sigmoide logística será chamada de $c$. O efeito da variação delas pode ser visto nas figuras 7 e 8.

Figura 5 – Gráfico da função sigmoide logística




Fonte: Autor (2026)

Figura 6 – Gráfico da função tangente hiperbólica



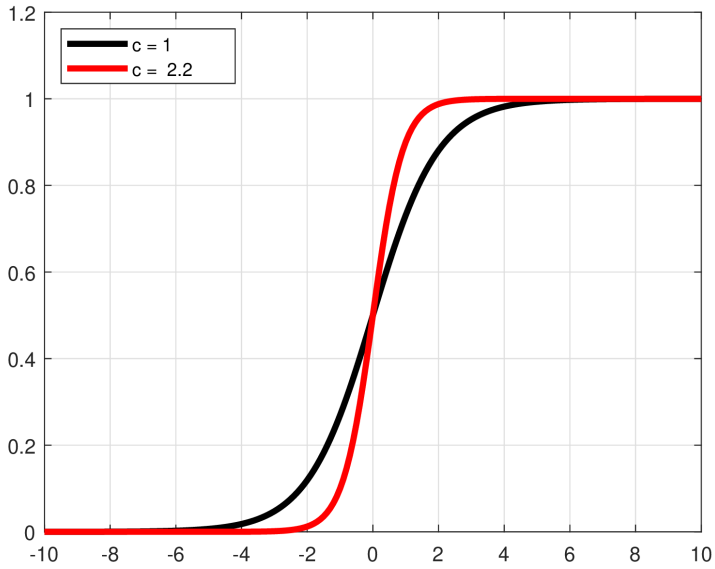

Fonte: Autor (2026)

Com a introdução dessas novas funções de ativação, a Equação 2.24 pode ser modificada para a seguinte forma:

$$\hat{y}(t) = \boldsymbol{\varphi}(\tilde{\mathbf{x}}(t)\mathbf{w}^T(t)) \tag{2.25}$$

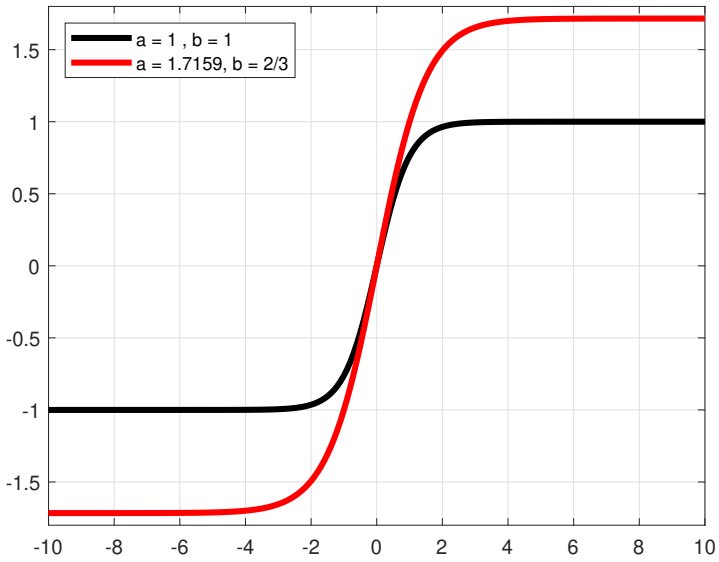
onde $\hat{y}(t)$ e $\mathbf{w}(t) \in \mathbb{R}^{1\times(p+1)}$ representam o valor predito de um neurônio na camada de saída e o seu respectivo vetor de pesos durante o instante $t$. É possível prever múltiplas saídas com o *perceptron* logístico, o que faz ele assumir a forma da Figura 9 .

Figura 7 – Efeito da variação da constante c na sigmoide logística




Fonte: Autor (2026)

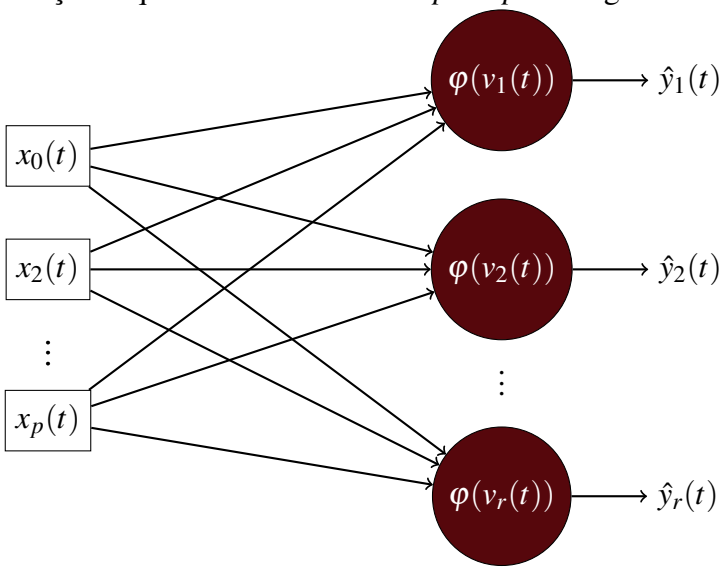
Figura 8 – Efeito da variação das constantes a e b na tangente hiperbólica




Fonte: Autor (2026)

O *perceptron* logístico é treinado iterativamente, isto é, os seus pesos são atualizados a cada instante com base na regra delta generalizada, a qual utiliza um método de otimização baseado no gradiente descendente. Primeiramente, gera-se a previsão do *perceptron* logístico

Figura 9 – Representação esquemática de uma rede *perceptron* logístico com múltiplas saídas




Fonte: Autor (2026)

conforme o desenvolvimento a seguir:

$$\tilde{\mathbf{x}}(t) = \begin{bmatrix} x_1(t) & \dots & x_p(t) & 1 \end{bmatrix}, \mathbf{W}(t) = \begin{bmatrix} w_{11}(t) & \dots & w_{1p}(t) & b_1(t) \\ \vdots & \ddots & \vdots & \vdots \\ w_{r1}(t) & \dots & w_{rp}(t) & b_r(t) \end{bmatrix}$$

$$\mathbf{v}_{(1\times r)}(t) = \tilde{\mathbf{x}}(t)\mathbf{W}^T(t) \tag{2.26}$$

$$\begin{bmatrix} v_1(t) & \dots & v_r(t) \end{bmatrix} = \begin{bmatrix} x_1(t) & \dots & x_p(t) & 1 \end{bmatrix} \begin{bmatrix} w_{11}(t) & \dots & w_{r1}(t) \\ \vdots & \ddots & \vdots \\ w_{1p}(t) & \dots & w_{rp}(t) \\ b_1(t) & \dots & b_r(t) \end{bmatrix}$$

$$\hat{\mathbf{y}}_{(1\times r)}(t) = \boldsymbol{\varphi}(\mathbf{v}(t)) \tag{2.27}$$

onde a matriz $\mathbf{W} \in \mathbb{R}^{(r\times(p+1))}$ é uma matriz que carrega os pesos de cada neurônio e $\mathbf{v} \in \mathbb{R}^{(1\times r)}$ representa um vetor com os seus respectivos potenciais de ativação. Os neurônios, representados pelos círculos vermelhos, são unidades básicas de processamento que desempenham o papel de calcular as predições para cada variável que se deseja prever.

Após adquirir a previsão do *perceptron* logístico em um dado instante, calcula-se a função-custo a seguir:

$$J(t) = \frac{1}{2}e^2(t) = \frac{1}{2}\sum_{n=1}^{r}(y_n(t) - \hat{y}_n(t))^2 \tag{2.28}$$

Em problemas de regressão, a função-custo geralmente será o erro quadrático médio. Deseja-se estimar um conjunto de pesos que minimize a função-custo, logo, é possível calcular o

seu gradiente na direção de cada um dos pesos $w_{nj}$ do *perceptron* logístico. O desenvolvimento a seguir demonstra como calculá-lo:

$$\frac{\partial J(t)}{\partial w_{nj}(t)} = \frac{\partial J(t)}{\partial \hat{y}_n(t)} \frac{\partial \hat{y}_n(t)}{\partial v_n(t)} \frac{\partial v_n(t)}{w_{nj}(t)} \tag{2.29}$$

onde:

$$\frac{\partial J(t)}{\partial \hat{y}_n(t)} = -(y_n(t) - \hat{y}_n(t)) \tag{2.30}$$

$$\frac{\partial \hat{y}_n(t)}{\partial v_n(t)} = \varphi_n'(v_n(t)) \tag{2.31}$$

$$\frac{\partial v_n(t)}{w_{nj}(t)} = x_j(t) \tag{2.32}$$

Substituindo as expressões acima na Equação 2.29, ela toma a forma abaixo:

$$\frac{\partial J(t)}{\partial w_{nj}(t)} = -(y_n(t) - \hat{y}_n(t))\varphi_n'(v_n(t))x_j(t)$$

$$\frac{\partial J(t)}{\partial w_{nj}(t)} = -\delta_n(t)x_j(t) \tag{2.33}$$

onde $\delta_n(t)$ representa o gradiente local instantâneo do n-ésimo neurônio. Analisando a Equação 2.33, nota-se que o cálculo do gradiente depende da derivada da função de ativação dos neurônios. Caso a função de ativação não tenha uma inclinação suave, como nas funções degrau ou degrau bipolar, o gradiente se torna nulo. Por esse motivo, as funções sigmoide logística e tangente hiperbólica foram introduzidas no *perceptron* logístico.

Após calcular o gradiente instantâneo, os pesos do *perceptron* logístico são atualizados com a seguinte regra:

$$w_{nj}(t+1) = w_{nj}(t) - \eta \frac{\partial J(t)}{\partial w_{nj}(t)}$$

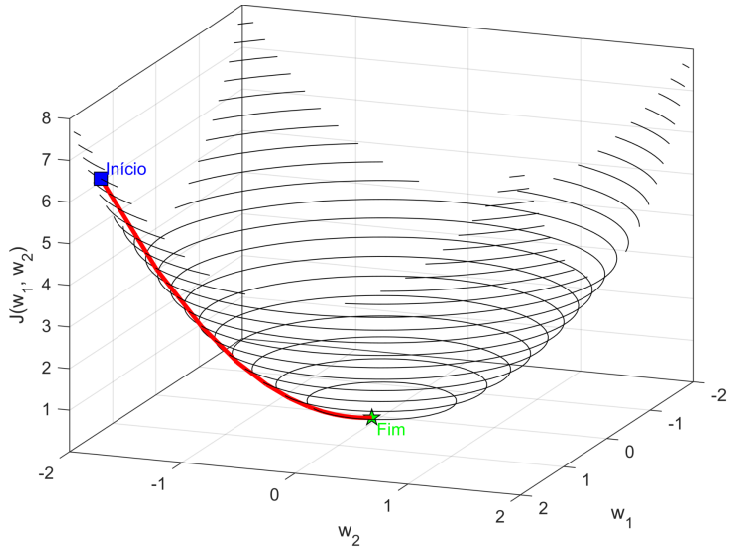
$$w_{nj}(t+1) = w_{nj}(t) + \eta \delta_n(t)x_j(t) \tag{2.34}$$

onde $\eta$ é uma taxa de aprendizado, a qual influencia na velocidade do treinamento. A Equação 2.34 é uma regra de ajuste de pesos recursiva que busca encontrar o ponto mínimo global da função-custo. Esse é o princípio do gradiente descendente, o qual se encontra ilustrado pela Figura 10.

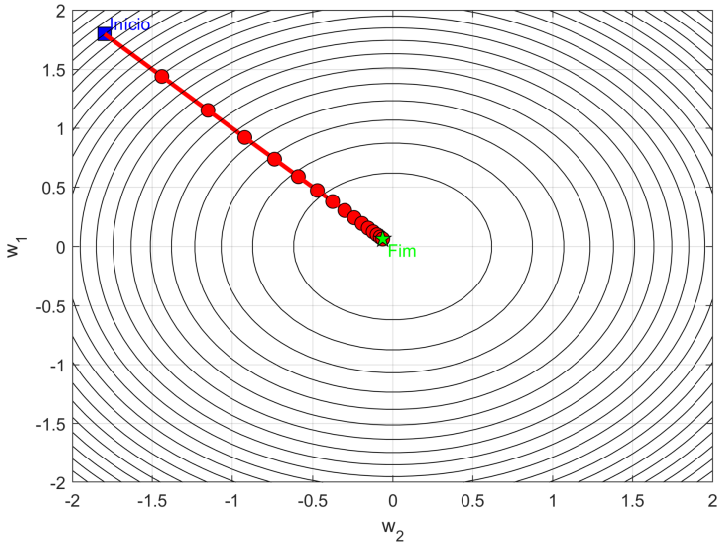
Quando todas as instâncias disponíveis para realizar o treinamento são apresentadas, diz-se que ocorreu uma época de treinamento. Cada época possui o seu próprio custo, calculado da seguinte maneira:

$$J_{epoca} = \frac{1}{N}\sum_{t=1}^{N} J(t) = \frac{1}{2N}\sum_{t=1}^{N}\sum_{n=1}^{r} (y_n(t) - \hat{y}_n(t))^2 \tag{2.35}$$

Figura 10 – Gradiente descendente




(a) 3D




(b) 2D

Fonte: Autor (2026)

Conforme os pesos são atualizados e o modelo aprende a relação entre as variáveis alvo e os atributos, espera-se que o custo diminua com o passar das épocas, até que ele fique estagnado ou atinja um valor desejável.

Existem diferentes formas de se atualizar os pesos. Na primeira, chamada de aprendizado *online*, os pesos são atualizados à medida que cada instância do conjunto de treinamento é apresentada. Na segunda, conhecida como aprendizado *offline*, a atualização só ocorre quando todos as instâncias são apresentadas, logo, é calculado o gradiente médio de todo o conjunto de treinamento a cada época.

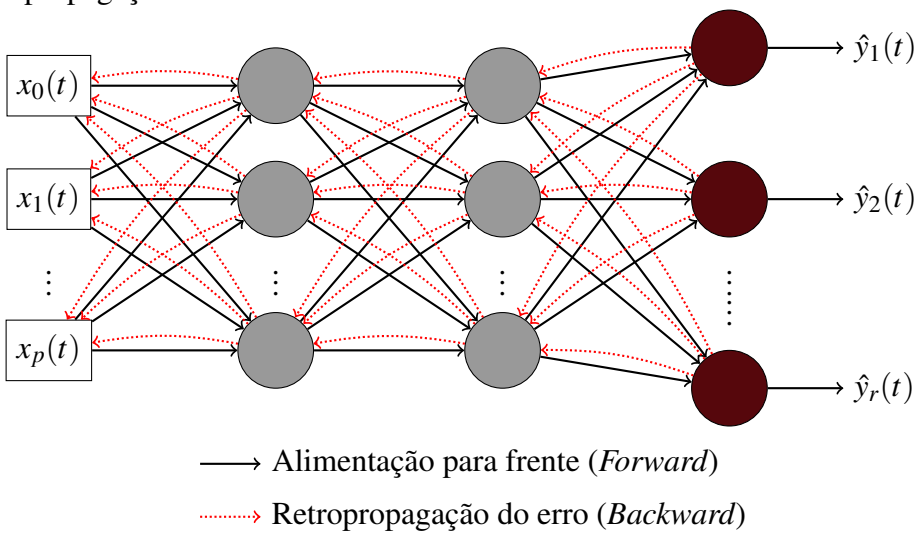
Segundo Bishop e Bishop (2024), o segundo método lida com a redundância no

conjunto de treinamento de forma mais eficiente. Apesar disso, o aprendizado *online* promove uma estimativa muito ruidosa do gradiente. Para contornar essa desvantagem, uma metodologia intermediária pode ser utilizada, como calcular o gradiente médio de pequenos mini-lotes dos dados.

### *2.4.3 Perceptron de múltiplas camadas (MLP)*

A rede *perceptron* de múltiplas camadas, do inglês *multilayer perceptron* (MLP), é composta por neurônios dispostos em camadas consecutivas, onde cada uma possui o seu próprio número de neurônios, funções de ativação e matrizes de pesos que precisam ser aprendidos para fornecer uma boa previsão. Em uma rede do tipo MLP, as saídas dos neurônios de uma camada são utilizadas como entradas de outros neurônios na camada subsequente, de modo que ela também é conhecida como uma rede totalmente conectada. A Figura 11 a seguir apresenta um exemplo de uma MLP com duas camadas ocultas.

Figura 11 – Representação esquemática de uma rede MLP com duas camadas ocultas e a retropropagação do erro.




Fonte: Autor (2026)

A última camada, conhecida como camada de saída, é responsável por realizar a previsão da rede. Entre a camada de saída e os sinais de entrada estão as camadas intermediárias ou ocultas, que são uma novidade em relação ao *perceptron* logístico, e são responsáveis por introduzir a não linearidade ao modelo de regressão, habilitando a rede a mapear relações mais complexas entre atributos e variáveis-alvo (Haykin, 1999).

A MLP também é treinada iterativamente, com o intuito de minimizar a função-custo

da Equação 2.28. Para demonstrar como ocorre o seu treinamento, será utilizada uma rede com duas camadas ocultas como exemplo, a qual pode ser expressa matematicamente por:

$$\hat{\mathbf{y}}(t) = \varphi_s(\mathbf{M}\varphi_2(\mathbf{L}\varphi_1(\tilde{\mathbf{x}}(t)\mathbf{W}^T(t)) \tag{2.36}$$

onde $\mathbf{W} \in \mathbb{R}^{w\times(p+1)}$, $\mathbf{L} \in \mathbb{R}^{l\times(w+1)}$ e $\mathbf{M} \in \mathbb{R}^{r\times(l+1)}$ denotam as matrizes de pesos entre as camadas da rede. $\varphi_s$ e $\varphi_h$ representam a função de ativação utilizada pelos neurônios na camada de saída e em uma camada oculta $h$ qualquer, respectivamente. Em um primeiro momento, a rede opera no sentido de alimentação para frente, ou *forward*, onde a informação flui dos sinais de entrada para a primeira camada oculta:

$$\tilde{\mathbf{x}}(t) = \begin{bmatrix} x_1(t) & \dots & x_p(t) & 1 \end{bmatrix}, \mathbf{W}(t) = \begin{bmatrix} w_{11}(t) & \dots & w_{1p}(t) & b_1(t) \\ \vdots & \ddots & \vdots & \vdots \\ w_{w1}(t) & \dots & w_{wp}(t) & b_w(t) \end{bmatrix}$$

$$\mathbf{v}^{(1)}_{(1\times w)}(t) = \tilde{\mathbf{x}}(t)\mathbf{W}^T(t) \tag{2.37}$$

$$\begin{bmatrix} v^{(1)}_1(t) & \dots & v^{(1)}_w(t) \end{bmatrix} = \begin{bmatrix} x_1(t) & \dots & x_p(t) & 1 \end{bmatrix} \begin{bmatrix} w_{11}(t) & \dots & w_{w1}(t) \\ \vdots & \ddots & \vdots \\ w_{1p}(t) & \dots & w_{wp}(t) \\ b_1(t) & \dots & b_w(t) \end{bmatrix}$$

$$\mathbf{h}^{(1)}_{(1\times w)}(t) = \varphi_1(\mathbf{v}^{(1)}(t)) \tag{2.38}$$

Dando continuidade à fase de *forward*, o vetor de saídas da primeira camada oculta é utilizado como entrada para a segunda camada oculta, conforme abaixo:

$$\tilde{\mathbf{h}}^{(1)}(t) = \begin{bmatrix} h^{(1)}_1(t) & \dots & h^{(1)}_w(t) & 1 \end{bmatrix}, \mathbf{L}(t) = \begin{bmatrix} l_{11}(t) & \dots & l_{1w}(t) & b'_1(t) \\ \vdots & \ddots & \vdots & \vdots \\ l_{l1}(t) & \dots & l_{lw}(t) & b'_l(t) \end{bmatrix}$$

$$\mathbf{v}^{(2)}_{(1\times l)}(t) = \tilde{\mathbf{h}}^{(1)}(t)\mathbf{L}^T(t) \tag{2.39}$$

$$\begin{bmatrix} v^{(2)}_1(t) & \dots & v^{(2)}_l(t) \end{bmatrix} = \begin{bmatrix} h^{(1)}_1(t) & \dots & h^{(1)}_w(t) & 1 \end{bmatrix} \begin{bmatrix} l_{11}(t) & \dots & l_{l1}(t) \\ \vdots & \ddots & \vdots \\ l_{1w}(t) & \dots & l_{lw}(t) \\ b'_1(t) & \dots & b'_l(t) \end{bmatrix}$$

$$\mathbf{h}^{(2)}_{(1\times l)}(t) = \varphi_2(\mathbf{v}^{(2)}(t)) \tag{2.40}$$

Repetindo o que foi feito na primeira camada oculta, o vetor de saídas da segunda camada oculta é utilizado como entrada para os neurônios da camada de saída, logo:

$$\tilde{\mathbf{h}}^{(2)}(t) = \begin{bmatrix} h_1^{(2)}(t) & \dots & h_l^{(2)}(t) & 1 \end{bmatrix}, \mathbf{M}(t) = \begin{bmatrix} m_{11}(t) & \dots & m_{1l}(t) & b_1''(t) \\ \vdots & \ddots & \vdots & \vdots \\ m_{r1}(t) & \dots & m_{rl}(t) & b_r''(t) \end{bmatrix}$$

$$\mathbf{v}_{(1\times r)}^{(s)}(t) = \tilde{\mathbf{h}}(t)^{(2)}\mathbf{M}^T(t) \tag{2.41}$$

$$\begin{bmatrix} v_1^{(s)}(t) & \dots & v_r^{(s)}(t) \end{bmatrix} = \begin{bmatrix} h_1^{(2)}(t) & \dots & h_l^{(2)}(t) & 1 \end{bmatrix} \begin{bmatrix} m_{11}(t) & \dots & m_{r1}(t) \\ \vdots & \ddots & \vdots \\ m_{1l}(t) & \dots & m_{rl}(t) \\ b_1''(t) & \dots & b_r''(t) \end{bmatrix}$$

$$\hat{\mathbf{y}}(t)_{(1\times r)} = \varphi_s(\mathbf{v}^{(s)}(t)) \tag{2.42}$$

Após produzir a previsão $\hat{\mathbf{y}}(t)$, calcula-se a função-custo instantânea $J(t)$ dada pelo erro quadrático médio de previsão, o qual deseja-se que seja o menor possível.

$$\min \quad J(t) = \frac{1}{2}e^2(t) = \frac{1}{2}\sum_{n=1}^{r}(y_n(t) - \hat{y}_n(t))^2 \tag{2.43}$$

Lembrando que o aprendizado da MLP é iterativo, para atualizar os pesos das camadas, a rede começa a operar no sentido inverso (*backward*) de forma que o erro de previsão é retropropagado da camada de saída até os sinais de entrada. Logo, é necessário calcular o gradiente instantâneo do erro na direção de cada peso da rede. Para um peso $m_{nj}$ qualquer da camada de saída, é adotado um procedimento semelhante ao *perceptron* logístico:

$$\frac{\partial J(t)}{\partial m_{nj}(t)} = \left(\frac{\partial J(t)}{\partial \hat{y}_n(t)}\right)\left(\frac{\partial \hat{y}_n(t)}{\partial v_n^{(s)}(t)}\right)\left(\frac{\partial v_n^{(s)}(t)}{m_{nj}(t)}\right) \tag{2.44}$$

onde:

$$\frac{\partial J(t)}{\partial \hat{y}_n(t)} = -(y_n(t) - \hat{y}_n(t)) \tag{2.45}$$

$$\frac{\partial \hat{y}_n(t)}{\partial v_n^{(s)}(t)} = \varphi_s'(v_n^{(s)}(t)) \tag{2.46}$$

$$\frac{\partial v_n(t)}{m_{nj}(t)} = h_j^{(2)}(t) \tag{2.47}$$

Com isso, a Equação 2.44 pode ser reescrita como:

$$\frac{\partial J(t)}{\partial m_{nj}(t)} = -(y_n(t) - \hat{y}_n(t))\varphi_s'(v_n^{(s)}(t))h_j^{(2)}(t)$$

$$\frac{\partial J(t)}{\partial m_{nj}(t)} = -\delta_n^{(s)}(t)h_j^{(2)}(t) \tag{2.48}$$

onde $\delta_n^{(s)}(t)$ denota o gradiente local do neurônio $n$ da camada de saída no instante t. Notavelmente, o gradiente do erro na direção dos pesos da matriz $\mathbf{M}$ depende das saídas dos neurônios da camada oculta anterior, isto é, a segunda camada oculta.

Prosseguindo com a retropropagação do erro, agora é necessário calcular o gradiente do erro na direção de cada peso $l_{ji}$ da segunda camada oculta. Utilizando a regra da cadeia, temos que:

$$\frac{\partial J(t)}{\partial l_{ji}(t)} = \left(\frac{\partial J(t)}{\partial h_j^{(2)}(t)}\right)\left(\frac{\partial h_j^{(2)}(t)}{\partial v_j^{(2)}(t)}\right)\left(\frac{\partial v_j^{(2)}(t)}{\partial l_{ji}(t)}\right) \tag{2.49}$$

onde:

$$\frac{\partial J(t)}{\partial h_j^{(2)}(t)} = -\sum_{n=1}^{r} \delta_n^{(s)}(t) m_{nj}(t) \tag{2.50}$$

$$\frac{\partial h_j^{(2)}(t)}{\partial v_j^{(2)}(t)} = \varphi_2'(v_j^{(2)}(t)) \tag{2.51}$$

$$\frac{\partial v_j^{(2)}(t)}{\partial l_{ji}(t)} = h_i^{(1)}(t) \tag{2.52}$$

O termo da equação 2.50 pode ser entendido como o erro do $j$-ésimo neurônio da segunda camada oculta e representa a "alma"do algoritmo de retropropagação do erro, uma vez que ele entrega aos neurônios da segunda camada oculta, uma combinação linear entre gradientes locais da camada de saída e os seus pesos. A Figura 12 ilustra exatamente esse conceito. Diante do exposto, é possível reescrever a Equação 2.49:

$$\frac{\partial J(t)}{\partial l_{ji}(t)} = \left(-\sum_{n=1}^{r} \delta_n^{(s)}(t) m_{nj}(t)\right) \varphi_2'(v_j^{(2)}(t)) h_i^{(1)}(t)$$

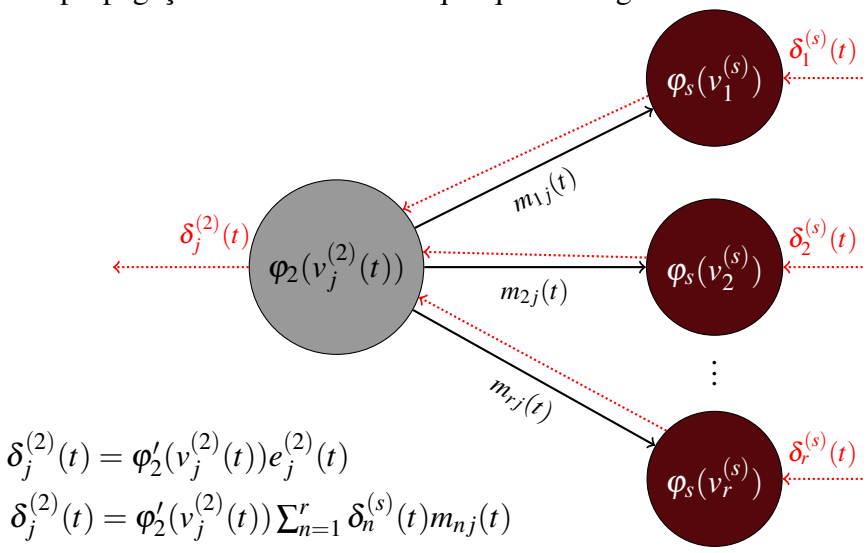
$$\frac{\partial J(t)}{\partial l_{ji}(t)} = -\delta_j^{(2)}(t) h_i^{(1)}(t) \tag{2.53}$$

onde $\delta_j^{(2)}(t)$ representa o gradiente local do $j$-ésimo neurônio da segunda camada oculta no instante $t$ . Novamente, é válido ressaltar que o gradiente do custo na direção dos pesos da segunda camada oculta depende da saída da camada anterior.

Seguindo para a primeira camada oculta, deseja-se calcular o gradiente da função-custo na direção de cada peso $w_{ik}$ presente nessa camada. Dessa forma, tem-se:

$$\frac{\partial J(t)}{\partial w_{ik}(t)} = \left(\frac{\partial J(t)}{\partial h_i^{(1)}(t)}\right)\left(\frac{\partial h_i^{(1)}(t)}{\partial v_i^{(1)}(t)}\right)\left(\frac{\partial v_i^{(1)}(t)}{\partial w_{ik}(t)}\right) \tag{2.54}$$

Figura 12 – Retropropagação em um neurônio qualquer da segunda camada oculta




Fonte: Autor (2026)

onde:

$$\frac{\partial J(t)}{\partial h_i^{(1)}(t)} = -\sum_{j=1}^{l}\delta_j^{(2)}(t)l_{ji}(t) \tag{2.55}$$

$$\frac{\partial h_i^{(1)}(t)}{\partial v_i^{(1)}(t)} = \varphi_1'(v_i^{(1)}(t)) \tag{2.56}$$

$$\frac{\partial v_i^{(1)}(t)}{\partial w_{ik}(t)} = x_k(t) \tag{2.57}$$

O termo $\frac{\partial J(t)}{\partial h_i^{(1)}(t)}$ pode ser entendido como o erro $e_i^{(1)}$ do $i$-ésimo neurônio da primeira camada oculta. Com isso, simplifica-se a Equação 2.54 para a expressão a seguir:

$$\frac{\partial J(t)}{\partial w_{ik}(t)} = \left(-\sum_{j=1}^{l}\delta_j^{(2)}(t)l_{ji}(t)\right)\varphi_1'(v_i^{(1)}(t))x_k(t)$$

$$\frac{\partial J(t)}{\partial w_{ik}(t)} = -\delta_i^{(1)}(t)x_k(t) \tag{2.58}$$

As equações dos gradientes locais instantâneos dos neurônios nas camadas da MLP formam um padrão, conforme a expressão abaixo revela

$$\delta_q^{(l)} = \begin{cases} (y_n(t) - \hat{y}_n(t))\varphi_s'(v_n^{(s)}(t)) & \text{,se l for a camada de saída} \\ \left(-\sum_{u=1}^{z}\delta_u^{(l+1)}(t)w_{uq}^{(l+1)}(t)\right)\varphi_{(l)}'(v_q^{(l)}(t)) & \text{,se l for uma camada oculta} \end{cases} \tag{2.59}$$

onde $w_{uq}^{(l+1)}(t)$ denota o peso entre o neurônio $u$ da camada $l+1$ e o neurônio $q$ da camada $l$. A Equação 2.59 é bastante útil para projetar redes neurais com muitas camadas ocultas. Uma vez que os gradientes instantâneos foram determinados, o gradiente descendente é utilizado para

para atualizar os pesos da rede:

$$m_{nj}(t+1) = m_{nj}(t) - \eta \frac{\partial J(t)}{\partial m_{nj}(t)} = m_{nj}(t+1) + \eta \delta_n^{(s)}(t) h_j^{(2)}(t) \tag{2.60}$$

$$l_{ji}(t+1) = l_{ji}(t) - \eta \frac{\partial J(t)}{\partial l_{ji}(t)} = l_{ji}(t+1) + \eta \delta_j^{(2)}(t) h_i^{(1)}(t) \tag{2.61}$$

$$w_{ik}(t+1) = w_{ik}(t) - \eta \frac{\partial J(t)}{\partial w_{ik}(t)} = w_{ik}(t+1) + \eta \delta_i^{(1)}(t) x_k(t) \tag{2.62}$$

onde $\eta$ representa a taxa de aprendizado novamente. Na literatura, frequentemente são encontradas técnicas para acelerar o aprendizado da rede neural. Uma das mais comuns envolve o cálculo da variação instantânea das matrizes de pesos, isto é, a adição dessa variação às regras de atualização dos pesos. Com isso, tem-se:

$$m_{nj}(t+1) = m_{nj}(t) + \Delta m_{nj}(t) \tag{2.63}$$

$$l_{ji}(t+1) = l_{ji}(t) + \Delta l_{ji}(t) \tag{2.64}$$

$$w_{ik}(t+1) = w_{ik}(t) + \Delta w_{ik}(t) \tag{2.65}$$

onde:

$$\Delta m_{nj}(t) = \alpha \Delta m_{nj}(t-1) - \eta \frac{\partial J(t)}{\partial m_{nj}(t)} \tag{2.66}$$

$$\Delta l_{ji}(t) = \alpha \Delta l_{ji}(t-1) - \eta \frac{\partial J(t)}{\partial l_{ji}(t)} \tag{2.67}$$

$$\Delta w_{ik}(t) = \alpha \Delta w_{ik}(t-1) - \eta \frac{\partial J(t)}{\partial w_{ik}(t)} \tag{2.68}$$

Com isso, as regras de atualização originais são alteradas para as expressões a seguir:

$$m_{nj}(t+1) = m_{nj}(t) + \alpha \Delta m_{nj}(t-1) + \eta \delta_n^{(s)}(t) h_j^{(2)}(t) \tag{2.69}$$

$$l_{ji}(t+1) = l_{ji}(t) + \alpha \Delta l_{ji}(t-1) + \eta \delta_j^{(2)}(t) h_i^{(1)}(t) \tag{2.70}$$

$$w_{ik}(t+1) = w_{ik}(t) + \alpha \Delta w_{ik}(t-1) + \eta \delta_i^{(1)}(t) x_k(t) \tag{2.71}$$

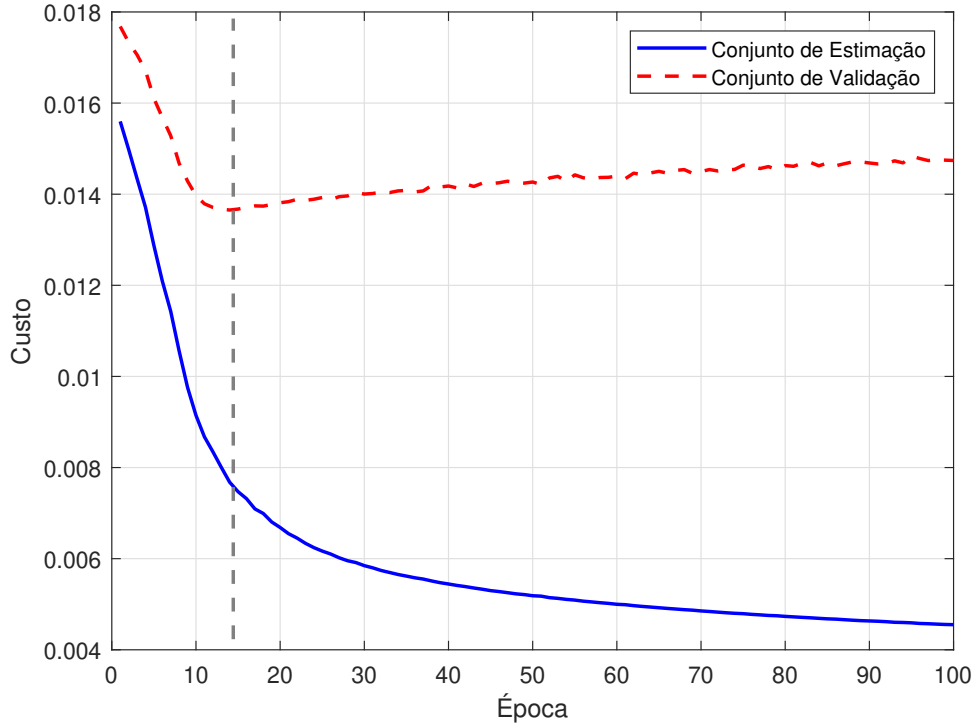
onde a constante $0 < \alpha < 1$ é chamada de constante de *momentum*, a qual pondera o efeito da variação instantânea, tornando a trajetória da descida do gradiente mais ou menos suave e acelerando a convergência.

Durante a fase de treinamento de uma rede neural, é comum preocupar-se com o poder de generalização do modelo, isto é, com a forma que o erro de predição será afetado quando novos dados forem apresentados à rede. Nesse contexto, a divisão do conjunto de dados em subconjuntos é uma prática essencial para evitar tanto o *underfitting* quanto o *overfitting*.

O primeiro ocorre quando a rede não aprende adequadamente a relação entre os atributos e a variável-alvo, geralmente por insuficiência de épocas de treinamento. O segundo manifesta-se quando o modelo se ajusta excessivamente aos dados de treinamento, chegando, em essência, a 'decorá-los'.

Os dados podem ser subdivididos entre os conjuntos de estimação, validação e teste. O subconjunto de estimação será aquele utilizado para efetivamente atualizar os pesos dos neurônios a cada época e o subconjunto de validação é utilizado para verificar como o modelo se comporta frente a novos dados após cada época. À medida que o treinamento ocorre, o valor da função-custo de ambos os subconjuntos diminui até que, eventualmente, ele começa a subir para o subconjunto de validação. Nesse momento, recomenda-se interromper o treinamento. A Figura 13 ilustra essa ideia.

Figura 13 – Curvas de aprendizado dos conjuntos de estimação e validação durante o treinamento




Fonte: Autor (2026)

Outro aspecto importante sobre a MLP diz respeito à sua quantidade de neurônios e camadas. Em uma primeira instância, uma rede MLP pode ter dezenas ou até mesmo centenas de camadas ocultas. No entanto, há limitações práticas em treinar redes neurais com muitas camadas ocultas. É amplamente conhecido na literatura que redes com poucos neurônios podem não ser capazes de modelar toda a complexa relação entre os atributos e a variável-alvo, levando a rede ao *underfitting*. No caso contrário, quando há uma quantidade excessiva de neurônios,

a rede se comporta de forma a superajustar o modelo aos dados de treinamento, entrando em *overfitting* e levando a uma baixa capacidade de generalização.

Diante do exposto, é notável que uma rede neural com camadas ocultas possui diversos hiperparâmetros. A escolha da combinação que leva ao melhor desempenho de predição pode ser desafiadora, pois geralmente, ela é feita com base em tentativa e erro. Dessa forma, o treinamento das redes neurais pode consumir muito tempo ou exigir *hardware* mais potente.

### *2.4.4 Regressão de vetores suporte (SVR)*

As máquinas de vetores suporte, do inglês *support vector machines* (SVM), foram construídas originalmente para problemas de classificação. Drucker *et al.* (1997) adaptaram-na para o contexto de tarefas de regressão, onde utiliza-se o termo regressão de vetores suporte, ou *support vector regression* (SVR). Segundo Zhang e O'Donnell (2020), modelos SVR são úteis porque possuem um bom balanço entre a complexidade do modelo e sua capacidade de previsão, além de prover uma ótima performance para dados com alta dimensão.

A SVR pode ser um modelo linear ou não linear. No primeiro caso, o objetivo da SVR é encontrar a função abaixo:

$$f(\mathbf{x}_n) = \mathbf{w}^T \mathbf{x}_n + b \tag{2.72}$$

onde $\mathbf{w} \in \mathbb{R}^p$, $\mathbf{x}_n \in \mathbb{R}^p$ e $b$, denotam o vetor de parâmetros que precisa ser aprendido, um vetor de atributos pertencente ao conjunto de treinamento e o viés. Deseja-se que a função $f$ seja tão plana quanto possível, tolerando até no máximo um erro $\varepsilon$. Conforme Basak *et al.* (2007), essa situação pode ser escrita como um problema de otimização convexa em que os parâmetros ótimos podem ser encontrados pela minimização da seguinte função-custo:

$$\min_{\mathbf{w}} J(\mathbf{w}) = \min_{\mathbf{w}} \left( \frac{1}{2} ||\mathbf{w}||_2^2 \right) \tag{2.73}$$

$$\text{restrita a} \begin{cases} y_n - \mathbf{w}^T \mathbf{x}_n + b \leq \varepsilon, \text{ para } n = 1,2, \ldots \text{ N} \\ \mathbf{w}^T \mathbf{x}_n + b - y_n \leq \varepsilon, \text{ para } n = 1,2, \ldots \text{ N} \end{cases} \tag{2.74}$$

Formalmente, tornar a função $f$ tão plana quanto possível equivale a minimizar a norma euclidiana $||.||_2$ do vetor de parâmetros $\mathbf{w}$ (Zhang e O'Donnell, 2020). O problema de otimização, como está configurado acima, é razoável para os casos onde a função $f$ existe e

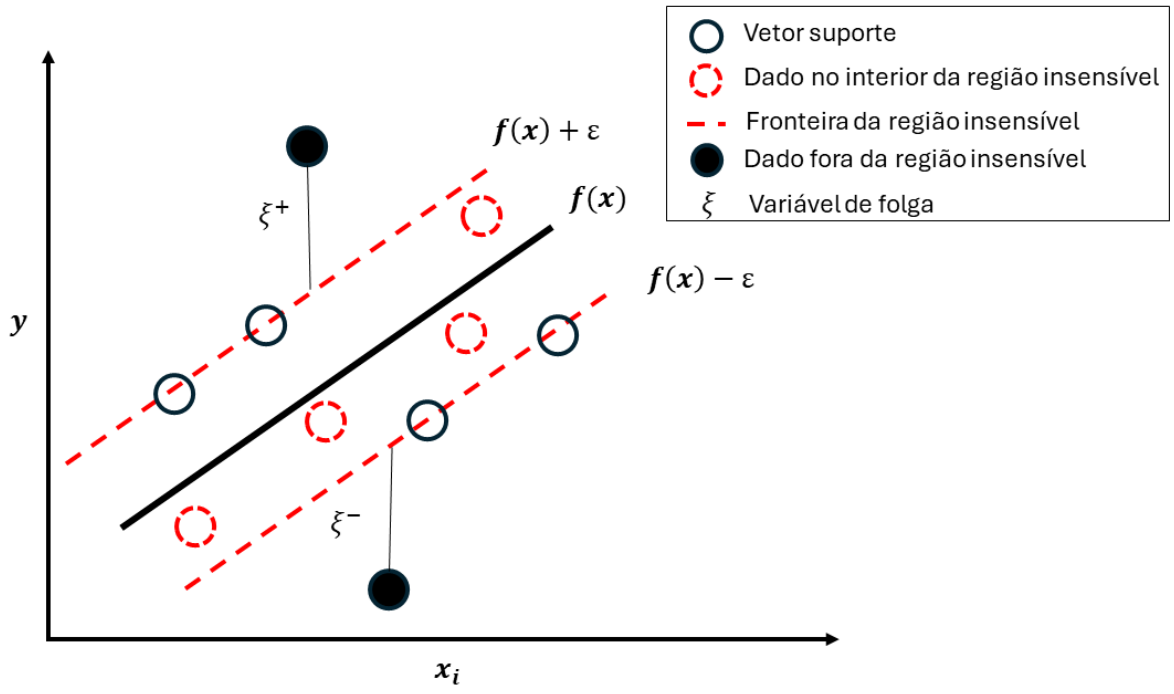
aproxima todas as instâncias da variável-alvo com uma precisão $\varepsilon$, no entanto, algumas vezes erros são admissíveis (Basak *et al.*, 2007). Diante do exposto, são as introduzidas variáveis de folga $\xi_n$ e $\xi_n^*$ no problema de otimização convexa, e a função-custo pode ser reformulada para:

$$\min_{\mathbf{w},\boldsymbol{\xi},\boldsymbol{\xi}^*} J(\mathbf{w},\boldsymbol{\xi},\boldsymbol{\xi}^*) = \min_{\mathbf{w},\boldsymbol{\xi},\boldsymbol{\xi}^*} \left( \frac{1}{2}||\mathbf{w}||_2^2 + c\sum_{n=1}^{N}(\xi_n + \xi_n^*) \right) \tag{2.75}$$

$$\text{restrita a} \begin{cases} y_n - \mathbf{w}^T\mathbf{x}_n - b \leq \varepsilon + \xi_n, \xi_n \geq 0 \\ \mathbf{w}^T\mathbf{x}_n + b - y_n \leq \varepsilon + \xi_n^*, \xi_n^* \geq 0 \end{cases} \tag{2.76}$$

Essa nova configuração está ilustrada na Figura 14, onde há uma região ou tubo insensível de largura $2\varepsilon$, na qual os erros de previsão são desconsiderados. O parâmetro $c$ na Equação 2.75 representa uma constante de regularização que promove um balanço entre a planicidade da função $f$ e o erro de predição, visto que $\xi_n$ e $\xi_n^*$ determinam quantos pontos podem ser tolerados fora do tubo insensível. Nesse contexto, os vetores suporte são o pequeno subconjunto de instâncias do conjunto de treinamentos que estão próximos ou além da região insensível (Awad e Khanna, 2015).

Figura 14 – Representação da região insensível no espaço de hiperparâmetros




Fonte: Autor (2026)

O conjunto de equações 2.75 e 2.76 é conhecido como a forma primordial ou *primal form* do problema de otimização. Segundo (Haykin, 2009), é possível solucionar o problema primordial por meio do método dos múltiplos de Lagrange. Diante do exposto, tem-se que:

$$\begin{aligned}\mathscr{L}(\mathbf{w},b,\xi,\xi^*,\boldsymbol{\alpha},\boldsymbol{\alpha}^*,\boldsymbol{\eta},\boldsymbol{\eta}^*) &= \frac{1}{2}\|\mathbf{w}\|_2^2 + c\sum_{n=1}^{N}(\xi_n+\xi_n^*)\\ &-\sum_{n=1}^{N}\alpha_n(\mathbf{w}^T\mathbf{x}_n+b+\varepsilon+\xi_n-y_n)\\ &-\sum_{n=1}^{N}\alpha_n^*(y_n-\mathbf{w}^T\mathbf{x}_n-b+\varepsilon+\xi_n^*)\\ &-\sum_{n=1}^{N}(\eta_n\xi_n+\eta_n^*\xi_n^*)\end{aligned} \tag{2.77}$$

onde $\boldsymbol{\alpha}$, $\boldsymbol{\alpha}^*$, $\boldsymbol{\eta}$ e $\boldsymbol{\eta}^*$ são os múltiplos positivos de Lagrange. É possível resolver o problema de otimização encontrando o ponto de sela de $\mathscr{L}$,ou seja, o ponto onde as raízes são iguais, mas de sinais opostos. Para encontrar o ponto de sela de acordo com as condições de Karush-Kuhn-Tucker (KKT) para otimalidade, basta calcular o gradiente nas direções de $\mathbf{w}$, $b$, $\boldsymbol{\xi}$ e $\boldsymbol{\xi}^*$ e torná-lo igual a zero. Com isso, tem-se que:

$$\frac{\partial\mathscr{L}}{\partial\mathbf{w}} = 0 \leftrightarrow \mathbf{w} = \sum_{n=1}^{N}(\alpha-\alpha^*)\mathbf{x}_n \tag{2.78}$$

$$\frac{\partial\mathscr{L}}{\partial b} = 0 \leftrightarrow \sum_{n=1}^{N}(\alpha_n-\alpha_n^*) = 0 \tag{2.79}$$

$$\frac{\partial\mathscr{L}}{\partial\xi_n} = 0 \leftrightarrow c-\alpha_n-\eta_n = 0 \tag{2.80}$$

$$\frac{\partial\mathscr{L}}{\partial\xi_n*} = 0 \leftrightarrow c-\alpha_n^*-\eta_n^* = 0 \tag{2.81}$$

Substituindo as expressões 2.78, 2.79, 2.80 e 2.81 na Equação 2.77, é possível reformular o problema de otimização convexa conforme abaixo:

$$\max_{\boldsymbol{\alpha},\boldsymbol{\alpha}^*}\mathscr{L}(\boldsymbol{\alpha},\boldsymbol{\alpha}^*) = -\frac{1}{2}\sum_{n,k=1}^{N}(\alpha_n+\alpha_n^*)(\alpha_k+\alpha_k^*)\mathbf{x}_n^T\mathbf{x}_k - \varepsilon\sum_{n=1}^{N}(\alpha_n+\alpha_n^*)+\sum_{n=1}^{N}y_n(\alpha_n-\alpha_n^*) \tag{2.82}$$

$$\text{restrita a}\begin{cases}\sum_{n=1}^{N}(\alpha_n-\alpha_n^*)=0\\ 0\le\alpha_n\le c, n=1,2,\ldots,N\\ 0\le\alpha_n^*\le c, n=1,2,\ldots,N\end{cases} \tag{2.83}$$

Ambas as equações 2.75 e 2.82, com as suas respectivas restrições, são conhecidas como problemas de programação quadrada ou *quadratic programming problems* (QPPs). O conjunto formado pela equação Equação 2.82 e suas restrições é conhecido como a forma dual do QPP, a qual possui os mesmos pontos ótimos da forma primordial, mas com a solução dependente apenas dos múltiplos de Lagrange (Haykin, 2009). Substituindo a Equação 2.78 na Equação 2.72, é possível encontrar a expressão abaixo:

$$f(\mathbf{x}_n) = \sum_{i=1}^{M} (\alpha - \alpha^*)\mathbf{x}_i\mathbf{x}_n + b \tag{2.84}$$

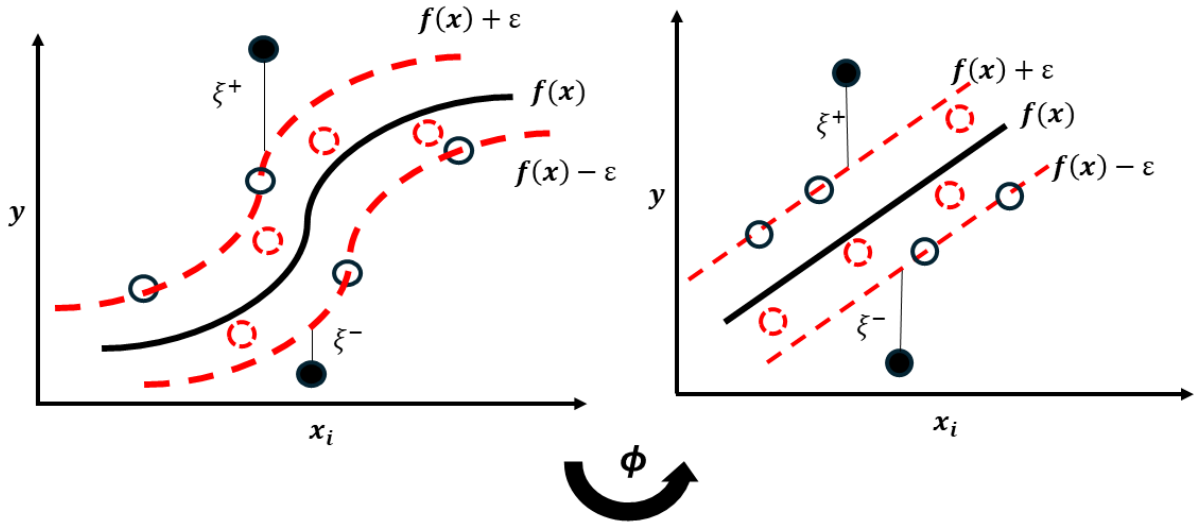
onde $M$ é a quantidade de múltiplos de Lagrange não nulos, os quais estão relacionados aos vetores suporte.

É importante lembrar que apenas uma pequena parte dos problemas de regressão são lineares. Nesse ponto, é importante adaptar a SVR para os caso não linear. Para realizar essa adaptação, modifica-se a Equação 2.72 para

$$f(\mathbf{x}_n) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b \tag{2.85}$$

A função $\boldsymbol{\phi}(.)$, desconhecida, mapeia $\mathbf{x}_n$ em um espaço de dimensão superior em que o problema de regressão se torna linear. A Figura 15 ilustra essa transformação promovida pela função $\boldsymbol{\phi}$. Diante do exposto, deve-se proceder da mesma forma que anteriormente, isto é, formular a forma primordial do SVR.

Figura 15 – Representação da região insensível do SVR em uma dimensão de ordem superior




Fonte: Autor (2026)

Lembrando que a SVR pode ser tratada como um problema de otimização convexa, para o caso não linear, é necessário reformular as restrições impostas pela Equação 2.76, de forma a incluir $\boldsymbol{\phi}(\mathbf{x}_n)$ conforme a seguir.

$$\text{restrita a} \begin{cases} y_n - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) - b \leq \varepsilon + \xi_n, \xi_n \geq 0 \\ \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b - y_n \leq \varepsilon + \xi_n^*, \xi_n^* \geq 0 \end{cases} \tag{2.86}$$

De forma similar ao caso linear, é possível introduzir a Lagrangiana $\mathscr{L}$ novamente para reformular o problema de otimização. Logo, tem-se:

$$\begin{aligned} \mathscr{L}(\mathbf{w}, b, \xi, \xi^*, \boldsymbol{\alpha}, \boldsymbol{\alpha}^*, \boldsymbol{\eta}, \boldsymbol{\eta}^*) &= \frac{1}{2}\|\mathbf{w}\|_2^2 + c\sum_{n=1}^{N}(\xi_n + \xi_n^*) \\ &- \sum_{n=1}^{N} \alpha_n(\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b + \varepsilon + \xi_n - y_n) \\ &- \sum_{n=1}^{N} \alpha_n^*(y_n - \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) - b + \varepsilon + \xi_n^*) \\ &- \sum_{n=1}^{N} (\eta_n \xi_n + \eta_n^* \xi_n^*) \end{aligned} \tag{2.87}$$

Para encontrar os parâmetros ótimos, é necessário calcular os gradientes de $\mathscr{L}$ na direção de $\mathbf{w}$, $b$, $\boldsymbol{\xi}$ e $\boldsymbol{\xi}^*$ e igualá-los a zero para obter a forma dual do QPP, encontrando a equação abaixo:

$$\max_{\boldsymbol{\alpha}, \boldsymbol{\alpha}^*} \mathscr{L}(\boldsymbol{\alpha}, \boldsymbol{\alpha}^*) = -\frac{1}{2}\sum_{n,k=1}^{N}(\alpha_n + \alpha_n^*)(\alpha_k + \alpha_k^*)k(\mathbf{x}_n, \mathbf{x}_k) - \varepsilon\sum_{n=1}^{N}(\alpha_n + \alpha_n^*) + \sum_{n=1}^{N} y_n(\alpha_n - \alpha_n^*) \tag{2.88}$$

$$\text{restrita a} \begin{cases} \sum_{n=1}^{N}(\alpha_n - \alpha_n^*) = 0 \\ 0 \leq \alpha_n \leq c, n = 1, 2, \ldots, N \\ 0 \leq \alpha_n^* \leq c, n = 1, 2, \ldots, N \end{cases} \tag{2.89}$$

Comparando as equações 2.82 e 2.88, é possível perceber que houve a introdução da função kernel $k$, a qual deve satisfazer as condições de Mercer (Awad e Khanna, 2015). A função kernel é utilizada para substituir o produto interno realizado no caso linear, evitando a necessidade de determinar a função $\boldsymbol{\phi}$. Na Tabela 1 as funções kernel mais comuns podem ser encontradas.

Tabela 1 – Funções Kernel

| Kernel | Fórmula | Hiperparâmetros |
|---|---|---|
| Linear | $\mathrm{k}(\mathbf{x}_n,\mathbf{x}_k) = \mathbf{x}_n^T\mathbf{x}_k$ | - |
| Polinomial | $\mathrm{k}(\mathbf{x}_n,\mathbf{x}_k) = \left(\mathbf{x}_n^T\mathbf{x}_k + \varsigma\right)^d$ | $\varsigma$, $d$ |
| Sigmoidal | $\mathrm{k}(\mathbf{x}_n,\mathbf{x}_k) = \tanh(\omega_1\mathbf{x}_n^T\mathbf{x}_k + \omega_2)$ | $\omega_1, \omega_2$ |
| Gaussiano | $\mathrm{k}(\mathbf{x}_n,\mathbf{x}_k) = exp(-\frac{\|x_n - x_k\|^2}{2\sigma^2})$ | $\sigma$ |

Fonte: Autor (2026)

Com isso, é possível obter uma versão modificada da Equação 2.84:

$$f(\mathbf{x}_n) = \sum_{i=1}^{M}(\alpha - \alpha^*)k(\mathbf{x}_i, \mathbf{x}_n) + b \tag{2.90}$$

A SVR produz uma solução global e é simples de calcular para problemas pequenos (Rivas-Perea *et al.*, 2013). Além disso, foram desenvolvidas variantes que são constantemente exploradas na tentativa de melhorar o desempenho da previsão. As próximas seções do presente trabalho irão abordar duas delas.

### *2.4.5 Regressão de vetores suporte dupla (TSVR)*

Esta seção irá abordar os fundamentos da regressão de vetores suporte dupla, conhecida como *twin support vector regression* (TSVR) na língua inglesa. A TSVR foi introduzida por Peng (2010) com o intuito de aumentar a velocidade de aprendizado em relação à SVR. O princípio da TSVR consiste em resolver dois QPP menores ao invés de um único com dimensão maior. Dessa forma, deseja-se encontrar duas funções regressoras de limite inferior e superior $f_1$ e $f_2$ tais que a função regressora final seja dada por:

$$f(\mathbf{x}_n) = \frac{1}{2}f_1(\mathbf{x}_n) + \frac{1}{2}f_2(\mathbf{x}_n) \tag{2.91}$$

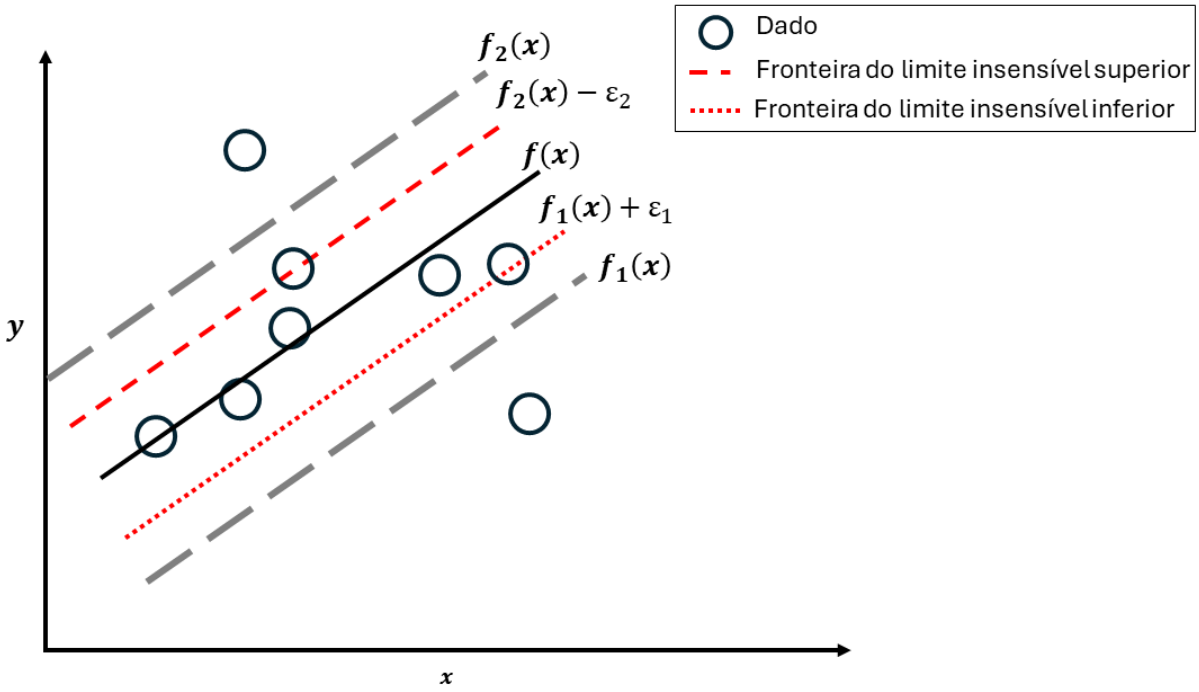
onde, no caso linear, as funções regressoras são dadas por:

$$f_1(\mathbf{x}_n) = \mathbf{w}_1^T\mathbf{x}_n + b_1 \tag{2.92}$$

$$f_2(\mathbf{x}_n) = \mathbf{w}_2^T\mathbf{x}_n + b_2 \tag{2.93}$$

A Figura 16 ilustra as funções regressoras da TSVR no espaço de atributos. Antes de abordar a solução dos QPPs da TSVR, é válido salientar que o seu desenvolvimento matemático geralmente é encontrado na forma matricial, isto é, utilizando toda a matriz de dados disponível para treinamento $\mathbf{X}$ e o vetor de instâncias da variável-resposta $\mathbf{y}$ como pode ser visto em

Figura 16 – Representação da região insensível do TSVR




Fonte: Autor (2026)

Peng (2010), Gu *et al.* (2020) e Huang *et al.* (2022). Diante do exposto, tem-se que as formas primordiais do TSVR são expressas como abaixo:

$$\min_{\mathbf{w_1},b_1,\boldsymbol{\xi}} J_1(\mathbf{w_1},b_1,\boldsymbol{\xi}) = \min_{\mathbf{w_1},b_1,\boldsymbol{\xi}} \left(\frac{1}{2}\|\mathbf{y}-\mathbf{e}\varepsilon_1-(\mathbf{X}\mathbf{w}_1+\mathbf{e}b_1)\|_2^2+c_1\mathbf{e}^T\boldsymbol{\xi}\right) \tag{2.94}$$

$$\text{restrita a} \quad \mathbf{y}-(\mathbf{X}\mathbf{w}_1+\mathbf{e}b_1) \geq \mathbf{e}\varepsilon_1-\boldsymbol{\xi}, \quad \boldsymbol{\xi} \geq \mathbf{0}$$

$$\min_{\mathbf{w_2},b_2,\boldsymbol{\eta}} J_2(\mathbf{w_2},b_2,\boldsymbol{\eta}) = \min_{\mathbf{w_2},b_2,\boldsymbol{\eta}} \left(\frac{1}{2}\|\mathbf{y}+\mathbf{e}\varepsilon_2-(\mathbf{X}\mathbf{w}_2+\mathbf{e}b_2)\|_2^2+c_2\mathbf{e}^T\boldsymbol{\eta}\right) \tag{2.95}$$

$$\text{restrita a} \quad (\mathbf{X}\mathbf{w}_2+\mathbf{e}b_2)-\mathbf{y} \geq \mathbf{e}\varepsilon_2-\boldsymbol{\eta}, \quad \boldsymbol{\eta} \geq \mathbf{0}$$

onde $c_1$ e $c_2$ são os parâmetros reguladores, $\varepsilon_1$ e $\varepsilon_2$ regulam a distância dos limites insensíveis até suas respectivas funções, $\boldsymbol{\xi}$ e $\boldsymbol{\eta}$ são os vetores de folga, $\mathbf{e}$ e $\mathbf{0}$ representam vetores de dimensões apropriadas com todos os elementos iguais a uns e zeros, respectivamente. De forma similar ao SVR, utiliza-se o método dos múltiplos de Lagrange para encontrar as funções Lagrangianas e eventualmente chegar nas seguintes formas duais dos QPPs:

$$\max_{\boldsymbol{\alpha}} \mathscr{L}_1(\boldsymbol{\alpha}) = \max_{\boldsymbol{\alpha}} \left( -\frac{1}{2}\boldsymbol{\alpha}^T \tilde{\mathbf{X}}(\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\boldsymbol{\alpha} + f^T\tilde{\mathbf{X}}(\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\boldsymbol{\alpha} - (\mathbf{y} - \mathbf{e}\varepsilon_1)^T\boldsymbol{\alpha} \right) \quad (2.96)$$

restrita a $\quad \mathbf{0} \leq \boldsymbol{\alpha} \leq c_1\mathbf{e}$

$$\max_{\boldsymbol{\beta}} \mathscr{L}_2(\boldsymbol{\beta}) = \max_{\boldsymbol{\beta}} \left( -\frac{1}{2}\boldsymbol{\beta}^T \tilde{\mathbf{X}}(\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\boldsymbol{\beta} - h^T\tilde{\mathbf{X}}(\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T\boldsymbol{\beta} + (\mathbf{y} + \mathbf{e}\varepsilon_2)^T\boldsymbol{\beta} \right) \quad (2.97)$$

restrita a $\quad \mathbf{0} \leq \boldsymbol{\beta} \leq c_2\mathbf{e}$

onde $\boldsymbol{\alpha}$ e $\boldsymbol{\beta}$ representam os vetores de múltiplos de Lagrange de cada QPP e lembrando que $\tilde{\mathbf{X}} = [\mathbf{X} \quad \mathbf{e}]$. Segundo Gu *et al.* (2020), ao resolver as equações 2.96 e 2.97, é possível obter as soluções das equações 2.92 e 2.93, as quais podem ser expressas como:

$$\begin{bmatrix} \mathbf{w}_1^T & b_1 \end{bmatrix}^T = (\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T(\mathbf{y} - \mathbf{e}\varepsilon_1 - \boldsymbol{\alpha}^*) \quad (2.98)$$

$$\begin{bmatrix} \mathbf{w}_2^T & b_2 \end{bmatrix}^T = (\tilde{\mathbf{X}}^T\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}^T(\mathbf{y} + \mathbf{e}\varepsilon_2 + \boldsymbol{\beta}^*). \quad (2.99)$$

onde $\boldsymbol{\alpha}^*$ e $\boldsymbol{\beta}^*$ representam os parâmetros ótimos. Estendendo o desenvolvimento da TSVR para o caso não linear, é possível introduzir funções Kernel na forma primordial dos QPPs de ambas as funções regressoras, obtendo as funções-custo abaixo:

$$\min_{\mathbf{w}_1,b_1,\boldsymbol{\xi}} J_1(\mathbf{w}_1,b_1,\boldsymbol{\xi}) = \min_{\mathbf{w}_1,b1,\boldsymbol{\xi}} \left( \frac{1}{2}\|\mathbf{y} - \mathbf{e}\varepsilon_1 - (\mathbf{K}(\mathbf{X},\mathbf{X}^T)\mathbf{w}_1 + \mathbf{e}b_1)\|_2^2 + c_1\mathbf{e}^T\boldsymbol{\xi} \right) \quad (2.100)$$

restrita a $\quad \mathbf{y} - (\mathbf{K}(\mathbf{X},\mathbf{X}^T)\mathbf{w}_1 + \mathbf{e}b_1) \geq \mathbf{e}\varepsilon_1 - \boldsymbol{\xi}, \quad \boldsymbol{\xi} \geq \mathbf{0}$

$$\min_{\mathbf{w}_2,b_2,\boldsymbol{\eta}} J_2(\mathbf{w}_2,b_2,\boldsymbol{\eta}) = \min_{\mathbf{w}_2,b_2,\boldsymbol{\eta}} \left( \frac{1}{2}\|\mathbf{y} + \mathbf{e}\varepsilon_2 - (\mathbf{K}(\mathbf{X},\mathbf{X}^T)\mathbf{w}_2 + \mathbf{e}b_2)\|_2^2 + c_2\mathbf{e}^T\boldsymbol{\eta} \right) \quad (2.101)$$

restrita a $\quad (\mathbf{K}(\mathbf{X},\mathbf{X}^T)\mathbf{w}_2 + \mathbf{e}b_2) - \mathbf{y} \geq \mathbf{e}\varepsilon_2 - \boldsymbol{\eta}, \quad \boldsymbol{\eta} \geq \mathbf{0}.$

Novamente, é possível utilizar o método dos múltiplos de Lagrange para encontrar as formas duais dos QPPs abaixo:

$$\max_{\boldsymbol{\alpha}} \mathscr{L}(\boldsymbol{\alpha}) = \max_{\boldsymbol{\alpha}} \left( -\frac{1}{2}\boldsymbol{\alpha}^T\mathbf{Q}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\boldsymbol{\alpha} + (\mathbf{y} - \mathbf{e}\varepsilon_1)^T\mathbf{Q}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\boldsymbol{\alpha} - (\mathbf{y} - \mathbf{e}\varepsilon_1)^T\boldsymbol{\alpha} \right) \quad (2.102)$$

restrita a $\quad \mathbf{0} \leq \boldsymbol{\alpha} \leq c_1\mathbf{e}$

$$\max_{\boldsymbol{\beta}} \mathscr{L}(\boldsymbol{\beta}) = \max_{\boldsymbol{\beta}} \left( -\frac{1}{2}\boldsymbol{\beta}^T\mathbf{Q}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\boldsymbol{\beta} - (\mathbf{y} + \mathbf{e}\varepsilon_2)^T\mathbf{Q}(\mathbf{Q}^T\mathbf{Q})^{-1}\mathbf{Q}^T\boldsymbol{\beta} + (\mathbf{y} + \mathbf{e}\varepsilon_2)^T\boldsymbol{\beta} \right) \quad (2.103)$$

restrita a $\quad \mathbf{0} \leq \boldsymbol{\beta} \leq c_2\mathbf{e}$

onde $\mathbf{Q} = [\mathbf{K}(\tilde{\mathbf{X}}, \tilde{\mathbf{X}}^T) \quad \mathbf{e}]$. Solucionando as equações 2.102, 2.103, é possível encontrar expressões semelhantes às equações 2.98 e 2.99, em que a matriz $\tilde{\mathbf{X}}$ é substituida pela matriz $\mathbf{Q}$. Diante do exposto, as funções regressoras podem ser reescritas conforme abaixo:

$$f_1(\mathbf{x}_n) = \mathbf{w}_1^T \mathbf{k}(\mathbf{x}_n, \mathbf{X}) + b_1 \tag{2.104}$$

$$f_2(\mathbf{x}_n) = \mathbf{w}_2^T \mathbf{k}(\mathbf{x}_n, \mathbf{X}) + b_2 \tag{2.105}$$

Durante as investigações realizadas por Peng (2010), a TSVR não foi apenas mais rápida, mas também mais precisa em relação à SVR. Nesse contexto, há uma expectativa de que a TSVR tenha uma performance mais elevada entre os dois modelos no presente trabalho.

### *2.4.6 Regressão de vetores suporte com mínimos quadrados (LSSVR)*

O último modelo baseado em regressão de vetores suporte a ser apresentado no presente trabalho trata-se da regressão de vetores suporte com mínimos quadrados ou *least squares support vector regression*. A LSSVR é consideravelmente diferente da SVR, pois a primeira almeja, no caso não linear, minimizar a seguinte função-custo:

$$\min_{\mathbf{w},\mathbf{e}} J(\mathbf{w}, \gamma) = \min_{\mathbf{w},\mathbf{e}} \left( \frac{1}{2} \|\mathbf{w}\|_2^2 + \frac{\gamma}{2} \sum_{n=1}^{N} e_n^2 \right) \tag{2.106}$$

$$\text{restrita a } y_n = (\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b) + e_n \tag{2.107}$$

onde $e_n \in \mathbb{R}$, $\mathbf{w} \in \mathbb{R}^p$, $\mathbf{x}_n \in \mathbb{R}^p$ e $b \in \mathbb{R}$ denotam o erro de previsão da n-ésima instância, o vetor de pesos, o vetor de atributos e o viés, respectivamente. No caso do LSSVR, $\gamma$ faz o papel da constante que realiza o balanço entre a planicidade da função regressora e os erros de previsão, os quais desempenham o papel das variáveis de folga.

Procedendo de forma semelhante à SVR, é possível transformar o conjunto de equações acima, o qual representa a forma primordial do problema de otimização restrita, para a sua forma dual por meio do método dos múltiplos de Lagrange (KARAL, 2024). Dessa forma, tem-se que a função Lagrangiana $\mathscr{L}$ é dada por:

$$\mathscr{L}(\mathbf{w}, \mathbf{e}, b, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|_2^2 + \frac{\gamma}{2} \sum_{n=1}^{N} e_n^2 - \sum_{n=1}^{N} \alpha_n (\mathbf{w}^T \boldsymbol{\phi}(\mathbf{x}_n) + b + e_n - y_n) \tag{2.108}$$

Diferenciando a função $\mathscr{L}$ em relação a $\mathbf{w}$, $b$, $e_n$ e $\alpha_n$ e igualando-a a zero, temos o seguinte conjunto de equações:

$$\frac{\partial \mathscr{L}}{\mathbf{w}} = 0 \leftrightarrow \mathbf{w} = \sum_{n=1}^{N} \alpha_n \phi(\mathbf{x}_n) \tag{2.109}$$

$$\frac{\partial \mathscr{L}}{\partial e_n} = 0 \leftrightarrow \alpha_n = \gamma e_n \tag{2.110}$$

$$\frac{\partial \mathscr{L}}{\partial b} = 0 \leftrightarrow \sum_{n=1}^{N} \alpha_n = 0 \tag{2.111}$$

$$\frac{\partial \mathscr{L}}{\alpha_n} = 0 \leftrightarrow y_n = \mathbf{w}^T \phi(\mathbf{x}_n) + b + e_n - y_n \tag{2.112}$$

Substituindo as expressões 2.109 e 2.110 na Equação 2.112, é possível reescrevê-la como abaixo:

$$y_n = \sum_{n=1}^{N} \alpha_n \phi(\mathbf{x}_n)^T \phi(\mathbf{x}_n) + b + \frac{\alpha_n}{\gamma} \tag{2.113}$$

A partir desse ponto do desenvolvimento, é comum encontrar na literatura um tratamento matricial. Reescrevendo a Equação 2.113 na sua forma matricial, tem-se:

$$\begin{aligned} \mathbf{y} &= \boldsymbol{\alpha}\mathbf{K} + \mathbf{b} + \boldsymbol{\alpha}\gamma^{-1} \\ \mathbf{y} &= \boldsymbol{\alpha}(\mathbf{K} + \mathbf{I}\gamma^{-1}) + \mathbf{b} \end{aligned} \tag{2.114}$$

onde $\mathbf{y} \in \mathbb{R}^{N\times 1}$, $\boldsymbol{\alpha} \in \mathbb{R}^{1\times N}$, $\mathbf{K} \in \mathbb{R}^{N\times N} =$, $\mathbf{b} \in \mathbb{R}^{N\times 1}$ são os vetores que denotam as saídas, os múltiplos de Lagrange, e o vetor de vieses. Além disso, a Equação 2.111 também pode ser escrita na sua forma matricial:

$$\boldsymbol{\alpha} = 0 \tag{2.115}$$

Juntas, as equações 2.114 e 2.115 formam um sistema de equações lineares dado por:

$$\begin{bmatrix} \mathbf{y} \\ 0 \end{bmatrix} = \begin{bmatrix} (\mathbf{K} + \mathbf{I}\gamma^{-1}) & \mathbf{e} \\ \mathbf{e} & 0 \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \mathbf{b} \end{bmatrix} \tag{2.116}$$

onde, novamente, **e** representa um vetor de dimensões apropriadas com todos os elementos iguais a um. Resolvendo o sistema, encontra-se a solução da LSSVR. Finalmente, é possível escrever a função regressora da LSSVR:

$$f(\mathbf{x}_n) = \sum_{k=1}^{N} \alpha_n k(x_n, x_k) + b \tag{2.117}$$

Observando a Equação 2.110, nota-se que ao contrário da SVR, a LSSVR utiliza todos as instâncias do conjunto de treinamento, sendo proporcional ao vetor de erros. Com isso, a LSSVR não produz soluções esparsas, apesar de ser mais fácil de calcular o seu QPP.

### *2.4.7 Avaliação de um modelo de regressão*

Em modelos de regressão, é importante avaliar não somente a capacidade preditiva, mas também os atributos que serão utilizados para prever a variável-alvo. Nesse âmbito, esta subseção será dedicada a tratar de ambos os temas, comentando sobre técnicas como análise de inflação de variância(FIV), regressão *stepwise* e parâmetros para quantificar a precisão dos modelos.

#### *2.4.7.1 Avaliação do desempenho*

Na literatura, frequentemente encontram-se diversos parâmetros para avaliar modelos de regressão. Um dos mais comuns é o coeficiente de determinação $R^2$, o qual pode ser calculado pela expressão abaixo:

$$R^2 = 1 - \frac{\sum_{n=1}^{N} (y_n - \hat{y}_n)^2}{\sum_{n=1}^{N} (y_n - \bar{y})^2} \tag{2.118}$$

onde $\bar{y}$ representa a média dos valores reais da variável-alvo. Conforme Montgomery (2013), o $R^2$ não é a medida de qualidade do modelo mais adequada, visto que ele aumenta conforme mais variáveis são adicionadas ao modelo de regressão. Dessa forma, é comum utilizar o $R^2_{ajustado}$, o qual pode ser calculado da seguinte maneira:

$$R^2_{ajustado} = 1 - \frac{\frac{\sum_{n=1}^{N} (y_n - \hat{y}_n)^2}{(N-p)}}{\frac{\sum_{n=1}^{N} (y_n - \bar{y})^2}{(N-1)}} = 1 - (1 - R^2)(\frac{N-1}{N-p-1}) \tag{2.119}$$

onde, uma vez que termo $\frac{\sum_{n=1}^{N} (y_n - \hat{y}_n)^2}{(N-p)}$ representa o erro quadrático médio e $\frac{\sum_{n=1}^{N} (y_n - \bar{y})^2}{(N-1)}$ é constante, $R^2_{ajustado}$ só irá aumentar quando a variável a ser adicionada conseguir diminuir o erro quadrático médio.

Além do $R^2$ e $R^2_{ajustado}$, é comum encontrar estudos apresentando medidas como o erro absoluto médio ( ou MAE, do inglês *mean absolute error*) ou a raiz do erro quadrático médio (ou RMSE, do inglês conhecido como *root mean squared error*). Eles podem ser calculados conforme abaixo:

$$MAE = \frac{1}{N} \sum_{n=1}^{N} |y_n - \hat{y}_n| \tag{2.120}$$

$$RMSE = \sqrt{\frac{1}{N} \sum_{n=1}^{N} (y_n - \hat{y}_n)^2} \tag{2.121}$$

Naturalmente, há outros tipos de parâmetros que podem ser utilizados. Nesse contexto, Naser e Alavi (2023) proporcionam uma grande coletânea de parâmetros. Vale ainda ressaltar que, em experimentos envolvendo algoritmos computacionais, costuma-se executar várias rodadas para obter estatísticas descritivas dos parâmetros de desempenho.

#### *2.4.7.2 Avaliação dos atributos*

É bastante comum que os atributos de um banco de dados possuam uma alta correlação entre si. Quando isso ocorre, diz-se que há uma alta multicolinearidade entre os atributos do conjunto de dados. Em um modelo de regressão linear, a multicolinearidade pode tanto aumentar ou diminuir os coeficientes da regressão, tornando-os falsamente significativos ou não (Tsagris e Pandis, 2021).

Diante dessa situação, foram desenvolvidas algumas técnicas para lidar com a multicolinearidade. A análise do fator de inflação de variância é uma ferramenta usada para medir e quantificar a inflação sofrida pelo coeficiente de um determinado atributo (Daoud, 2017). Com isso, tem -se:

$$FIV_j = \frac{1}{1+R_j^2} \tag{2.122}$$

onde $R_j^2$ representa o coeficiente de determinação de um modelo de regressão linear múltipla em que o $j$-ésimo atributo foi utilizado como variável-alvo. Daoud (2017) também reporta que um atributo com FIV maior do que 10 indica que ele possui uma elevada correlação.

A regressão *stepwise* pode ser utilizada para a eliminação regressiva de variáveis no modelo que não contribuem significativamente para o modelo. Nesse sentido, a regressão *stepwise* trata-se de uma forma sistemática de se utilizar a regressão linear múltipla para promover uma seleção de atributos.

Segundo Montgomery (2013), inicia-se com todos os regressores candidatos e em seguida, seleciona-se o regressor com a menor estatística F e a compara com um limiar pré-definido. Se a estatística F do regressor for menor do que o valor desse limiar, elimina-se o regressor correspondente do modelo. Em seguida, realiza-se uma nova regressão com os regressores restantes. Esse algoritmo é executado iterativamente, até que não seja mais possível excluir atributos do modelo. Darlington e Hayes (2016) relatam outras variantes, onde o modelo começa sem regressores e estes vão sendo adicionados ou removidos à medida que o desempenho aumenta ou diminui.

## 2.5 Estimativa do número de neurônios ocultos por decomposição em valores singulares

No presente trabalho, a decomposição em valores singulares (SVD) foi utilizada para estimar o número de neurônios ocultos das redes neurais. SVD é uma técnica de decomposição onde uma matriz qualquer $A_{m\times n}$ pode ser fatorada como abaixo (Strang, 2019):

$$\mathbf{A} = \mathbf{USV}^T = \sum_{i=1}^{r} \sigma_i \mathbf{u}_i \mathbf{v}_i^T \tag{2.123}$$

onde **S** e **A** possuem as mesmas dimensões, $\mathbf{U} \in \mathbb{R}^{m\times m}$ e $\mathbf{V} \in \mathbb{R}^{n\times n}$ são matrizes quadradas de dimensões $m \times m$ e $n \times n$, respectivamente. A diagonal principal de **S** é composta pelos valores singulares $\sigma_i$ de **A** e as colunas de **U** e **V** são os vetores singulares esquerdos $\mathbf{u}_i \in \mathbb{R}^{m\times 1}$ e direitos $\mathbf{v}_i \in \mathbb{R}^{n\times 1}$ da matriz A, respectivamente. O número de vetores singulares $r$ é igual ao *rank* de **A**.

De acordo com Bermeitinger *et al.* (2019), os produtos **US** e $\mathbf{V}^T$ podem ser vistos como dois mapeamentos lineares, onde o o primeiro mapeia o espaço de entradas com dimensão $n$ para um espaço intermediário de dimensão $\hat{r}$ e o segundo mapeia este para o espaço de saída com dimensão $m$. Nesse contexto, considerando que $m, n \gg r$, o espaço intermediário atua como um espaço latente, isto é, ele comprime os dados de entrada em uma representação de dimensão reduzida, onde as redundâncias inerentes são eliminadas e as informações mais importantes são preservadas.

Santos *et al.* (2010) propuseram que o SVD pode ser usado nas camadas ocultas de uma rede neural do tipo MLP para estimar o número de neurônios com a regra abaixo:

$$q^* = \arg\min_{q=1,\ldots,Q} \left\{ \frac{\sum_{i=1}^{q} \sigma_i^2}{\sum_{j=1}^{Q} \sigma_j^2} \geq \gamma \right\} \tag{2.124}$$

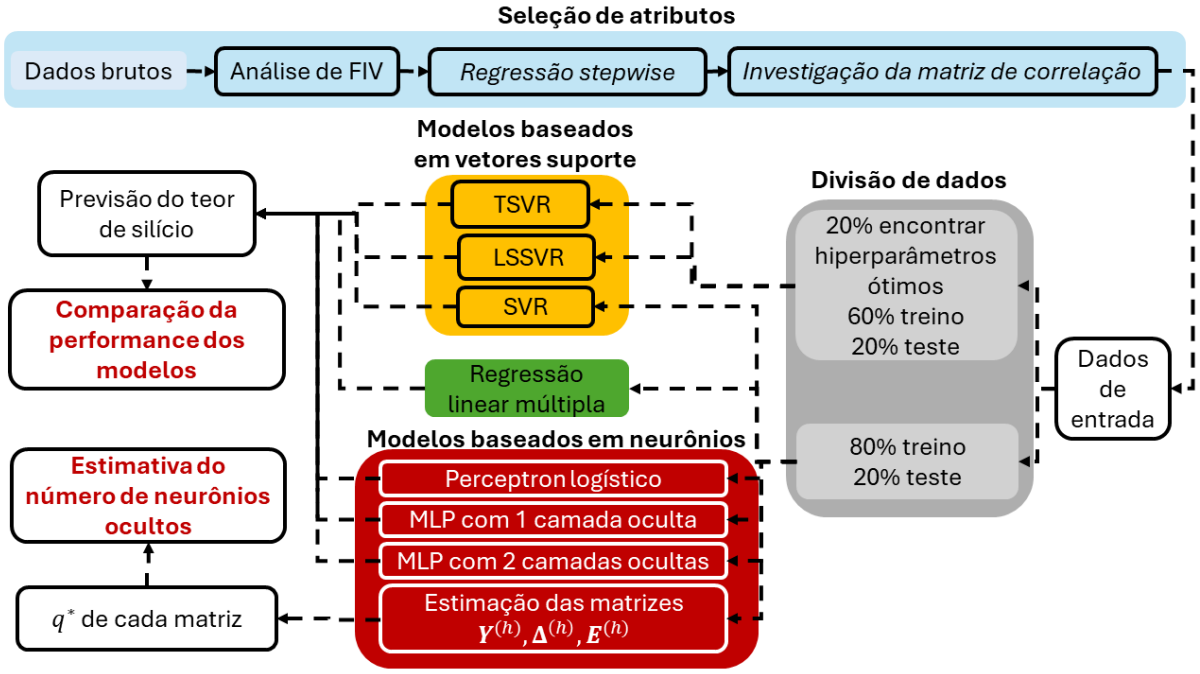
onde $q^* \leq Q$ é o número de neurônios estimado, $Q$ é o número de neurônios inicial e $0 < \gamma \leq 1$ representa um limiar de decisão.

A camada oculta de uma MLP pode ser interpretada como um espaço latente. A Equação 2.124 especifica que o número mínimo de neurônios ocultos, $q^*$, é determinado pelo valor de $q$ para o qual a razão entre a soma dos primeiros $q$ valores singulares ao quadrado e a soma de todos os $Q$ valores singulares ao quadrado é maior ou igual a um limiar pré-definido $\gamma$. Nesse contexto, supõe-se que a dimensão desse espaço oculto/latente é equivalente ao número de neurônios na camada oculta correspondente.

# 3 METODOLOGIA

Esta seção providencia uma explicação sobre como o presente trabalho foi realizado, desde a obtenção do conjunto de dados até o uso de algoritmos de aprendizado de máquinas para realizar a tarefa de previsão do teor de silício no ferro-gusa. A Figura 17 resume a metodologia utilizada no presente trabalho

Figura 17 – Representação esquemática da metodologia




Fonte: Autor (2026)

## 3.1 Aquisição e pré-processamento do conjunto de dados

O conjunto de dados utilizado neste trabalho é composto por 6970 vetores de atributos referentes às 17 variáveis operacionais do alto-forno de uma usina siderúrgica movida a coque. Os atributos foram escolhidos com base na literatura existente sobre a tarefa de previsão do teor de silício e na sugestão de profissionais da indústria. O registro das variáveis de entrada (instâncias) foi realizado a cada corrida do alto-forno, de forma que elas estão registradas em um intervalo médio de duas horas e cinquenta minutos. A Tabela 2 a seguir sumariza as variáveis coletadas.

Inicialmente, alguns vetores de atributos presentes no banco de dados estavam incompletos, isto é, o valor de um ou mais atributos não foram registrados. Isso pode ocorrer

Tabela 2 – Tabela sumário do conjunto de dados

| Acrônimo | Nome da variavel | Intervalo | Unidade |
|---|---|---|---|
| Si | Silício | 0,06 - 1,68 | % |
| Rendimento | Rendimento de gás | 40,52 - 58,31 | % |
| - | Sinter | 64,96 - 94,22 | % |
| B2 | Basicidade binária da escória | 1,00 - 1,32 | %/% |
| HMT | Temperatura do ferro-gusa | 1400,00 - 1556,00 | °C |
| TS | Temperatura do sopro | 18,27 - 1806,22 | °C |
| TTC | Temperatura teórica de chama | 1047,40 - 2700,27 | °C |
| T. *Liquidus* | Temperatura *Liquidus* da escória | 1272,26 - 1387,42 | °C |
| CR | *Coke rate* | 285,10 - 570,52 | Kg./ Ton. de gusa |
| FR | *Fuel rate* | 294,50 -2491,63 | Kg./ Ton. de gusa |
| SR | *Sinter rate* | 228,67 - 339,59 | Kg./ Ton. de gusa |
| CT | Carga térmica | 2367,00 - 3450,00 | MJ/min. |
| VS | Volume de sopro | 0,00 - 6096,98 | $Nm^3$/min. |
| VGV | Volume de gás nas ventaneiras | 1520,47 - 8721,10 | $Nm^3$/min. |
| Ritmo | Ritmo de produção | 0,00 - 9324,52 | Ton./dia |
| PS | Pressão de sopro | 0,44 - 4,04 | $Kg/cm^2$ |
| $\Delta P/V$ | $\Delta P$ /Vol. de sopro | 0,16 - 0,36 | $(Kg \cdot min)/(cm^2 \cdot Nm^3)$ |

Fonte: Autor (2026)

pelos mais variados motivos como parada de equipamentos, erros operacionais ou falha no registro das instâncias. Para completar o vetor de atributos $\mathbf{x}_i$ nesse estado, o valor do atributo $x_j$ com valor desconhecido foi estimado pela média desse $j$-ésimo atributo nos demais vetores de atributos em que o seu valor era conhecido. No total, havia 613 vetores de atributos incompletos, o que representa 8.79% do total. A Tabela 3 a seguir mostra a quantidade de instâncias incompletas descriminadas por atributo.

## 3.2 Seleção de atributos

Antes de alimentar os algoritmos computacionais com o conjunto de dados, este foi submetido a uma etapa de seleção de atributos composta por três fases. A primeira consistiu na análise do fator de inflação de variância (FIV) de cada preditor. Para garantir uma análise mais robusta, foram executadas 100 rodadas de cálculo do FIV, selecionando 80% dos dados aleatoriamente a cada rodada. Atributos com FIV maior do que 10 foram excluídos do conjunto de dados, pois eles indicam forte multicolinearidade (Senaviratna e Cooray, 2019).

Na segunda fase da seleção de atributos, foi realizada uma regressão *stepwise* no conjunto de dados resultante da fase anterior. Em uma primeira etapa, o conjunto de preditores da regressão linear, denotado por $\mathbf{X}^*$, possuia apenas um preditor constante. Após organizar o conjunto de preditores do banco de dados, denotado por $\mathbf{X}$, em ordem decrescente de correlação com o silício, foram realizadas sucessivas regressões lineares, adicionando-se temporariamente

Tabela 3 – Quantidade de instâncias incompletas por atributo

| Variável | Quantidade de vetores de atributos incompletos |
|---|---|
| Si | 37 (0,53 %) |
| Rendimento | 0 (0 %) |
| - | 0 (0 %) |
| B2 | 95 (1,36 %) |
| HMT | 38 (0,54 %) |
| TS | 489 (7,00 %) |
| TTC | 3 (0,04 %) |
| T. *Liquidus* | 95 (1,36 %) |
| CR | 0 (0 %) |
| FR | 0 (0 %) |
| SR | 0 (0 %) |
| CT | 0 (0 %) |
| VS | 0 (0 %) |
| VGV | 0 (0 %) |
| Ritmo | 0 (0 %) |
| PS | 489 (7,00 %) |
| $\Delta P$/V | 490 (7,01 %) |

Fonte: Autor (2026)

uma variável de $\mathbf{X}$ em $\mathbf{X}^*$ por vez e obtendo o vetor $\mathbf{f} = [f_1\ f_2\ ...\ f_p]$, onde o seu $j$-ésimo elemento $f_j$ representa o $F_{calculado}$ do preditor $x_j$ que ainda não estava em $\mathbf{X}^*$. O preditor com maior $F_{calculado}$ foi comparado com um $F_{critico}$ previamente escolhido. Se $F_{calculado} > F_{critico}$, a variável ingressava em $\mathbf{X}^*$.

Na segunda etapa da regressão *stepwise*, que ocorre logo após a entrada do preditor $x_j$ de $\mathbf{X}$ em $\mathbf{X}^*$, foram realizadas sucessivas regressões lineares novamente, removendo-se temporariamente um preditor por vez de $\mathbf{X}^*$ e obtendo-se $\mathbf{f}^* = [f_1^*\ f_2^*\ ...\ f_m^*]$, onde $f_i^*$ representa o $F'_{calculado}$ da regressão linear sem o preditor $x_i^*$ em $\mathbf{X}^*$. O preditor com o menor valor de $F'_{calculado}$ se torna um possível candidato a sair de $\mathbf{X}^*$, o que só ocorre efetivamente se $F'_{calculado} < F'_{critico}$, onde $F'_{critico}$ também foi previamente definido. As duas etapas da regressão *stepwise* foram executadas iterativamente até que não fosse mais possível remover e adicionar variáveis de $\mathbf{X}$ e $\mathbf{X}^*$. Ambos os limiares de decisão de entrada ou saída de variáveis em $\mathbf{X}^*$ foram configurados com base no valor de F correspondente a um nível de significância $\alpha$ usualmente utilizado igual a 0.05, conforme Galvão e Araújo (2009).

A última fase da seleção de atributos consistiu na análise da matriz de correlação do conjunto de preditores que não foram removidos na fase anterior. Com o intuito de remover preditores com grande colinearidade entre si, aqueles com menor correlação com o teor do silício nos pares com correlação maior do que 0,70 entre si foram removidos do conjunto de dados.

Após a seleção de atributos, os preditores restantes foram utilizados para alimentar algoritmos regressores para realizar a previsão do teor de silício no ferro-gusa.

No espírito de alcançar o melhor desempenho com o algoritmo mais simples possível, buscou-se experimentar os algoritmos em ordem crescente de complexidade. Dessa forma, foram testados modelos de regressão linear múltipla, *perceptron* logístico, redes neurais artificiais do tipo *perceptron* de múltiplas camadas com até duas camadas ocultas, regressão de vetores suporte, regressão de vetores suporte com mínimos quadrados, e regressão de vetores suporte dupla.

## 3.3 Regressão linear múltipla

Na regressão linear múltipla, foram executadas 100 rodadas de treino e teste para obter estatísticas descritivas sobre o desempenho desse modelo. A cada rodada, 80% do conjunto de dados era utilizado para estimar o vetor de coeficientes $\mathbf{w}$ da regressão linear e os 20% restantes foram utilizados para gerar o vetor de previsões $\hat{\mathbf{y}}_n$ da n-ésima rodada. Vale salientar que a Equação 2.23 foi utilizada para estimar $\mathbf{w}$ e que a cada rodada, os dados foram normalizados no intervalo [0,1], utilizando os valores máximos e mínimos do subconjunto de treinamento para normalizar o subconjunto de teste.

## 3.4 *Perceptron* logístico

No *perceptron* logístico, foi necessário experimentar diferentes combinações de hiperparâmetros. A Tabela 4 mostra quais foram as configurações experimentadas.

Tabela 4 – Configurações dos experimentos do *perceptron* logístico

| Experimento | $\varphi(v)$ | a | b | c | $\eta$ |
|---|---|---|---|---|---|
| 1 | tanh | 1 | 1 | - | $10^{-4}$ |
| 2 | tanh | 1,7159 | (2/3) | - | $10^{-4}$ |
| 3 | sig | - | - | 1 | $10^{-4}$ |
| 4 | sig | - | - | 2,2 | $10^{-4}$ |

Fonte: Autor (2026)

De forma semelhante à RLM, para cada configuração, foram executadas 100 rodadas de treino e teste, onde 80% dos dados foram usados para treinar o modelo, ou seja, estimar o vetor de pesos $\mathbf{w}$ do neurônio que liga os sinais de entrada à saída do *perceptron*, e os 20% restantes foram utilizados para produzir o vetor de previsões $\hat{\mathbf{y}}_n$ da n-ésima rodada. Em cada

rodada, os conjuntos de treino e teste foram normalizados conforme a função de ativação $\varphi(.)$ utilizada, se ela fosse tangente hiperbólica, os dados eram normalizados entre -1 e 1. Caso ela fosse a sigmoide logística, os dados eram normalizados entre 0 e 1. Além disso, vale salientar que os dados de teste foram normalizados utilizando os valores mínimos e máximos do conjunto de treinamento a cada rodada.

## 3.5 *Perceptron* de múltiplas camadas

As redes neurais artificiais utilizadas no presente trabalho foram do tipo *perceptron* de múltiplas camadas. Em comparação com o *perceptron* logístico, há uma quantidade maior de hiperparâmetros para variar. A Tabela 5 mostra quais foram os hiperparâmetros variados no presente trabalho. É possível perceber que somente a função de ativação tangente hiperbólica foi utilizada nas camadas ocultas. Segundo Goodfellow *et al.* (2016), quando se utiliza uma função de ativação do tipo sigmoidal, a tangente hiperbólica geralmente possui uma performance melhor do que a sigmoide logística. Dessa forma, optou-se por variar a função de ativação apenas na camada de saída.

Tabela 5 – Hiperparâmetros da rede neural

| Hiperparâmetro | Valores experimentados |
|---|---|
| Nº de camadas ocultas | [1; 2] |
| Nº de neurônios nas camadas ocultas | 1 até 40 (1 camada oculta)<br>7 até 13 (2 camadas ocultas) |
| Função de ativação das camadas ocultas | $\varphi(v) = a \cdot tanh(bv)$ |
| Funções de ativação da camada de saída | $\varphi(v) = a \cdot tanh(bv)$<br>$\varphi(v) = sig(cv)$<br>$\varphi(v) = v$ |
| Constantes {a,b} da função tanh | {1; 1} ou {1,7159; 2/3} |
| Constante c da função sigmoide | [1; 2,2] |
| $\eta$ | [0,01; 0,001] |
| $\alpha$ | [0,90; 0,95] |

Fonte: Autor (2026)

Repetindo a abordagem utilizada com o PL e a RLM, foram executadas 100 rodadas de treino e teste da rede neural. No entanto, o tamanho do conjunto de treinamento foi determinado com o auxílio da heurística abaixo presente em Haykin (2009):

$$N = O\left(\frac{W}{\varepsilon}\right) \tag{3.1}$$

onde N, W e $\varepsilon$ representam o número de instâncias necessárias para o treinamento da rede neural, a quantidade de parâmetros livres a serem determinados presentes na rede e o erro de previsão desejado, respectivamente. $O(.)$ representa a ordem de grandeza do seu argumento. Exemplificando, caso se deseje um erro de previsão de 10%, a Equação 3.1 diz que será necessário pelo menos 10 vezes mais instâncias de treinamento do que parâmetros livres na rede. O número máximo de neurônios das camadas ocultas exposto na Tabela 5 foi determinado como de acordo com essa heurística.

O subconjunto de dados de treinamento foi subdividido nos subconjuntos de estimação e validação. O subconjunto de estimação foi responsável por atualizar as matrizes de pesos sinápticos de cada camada da rede durante as épocas de treinamento e o subconjunto de validação foi utilizado para o critério de parada antecipada. A proporção entre o tamanho desses dois subconjuntos foi determinada pela regra abaixo presente em Haykin (1999):

$$r_{otimo} = 1 - \frac{\sqrt{2W-1}-1}{2(W-1)} \tag{3.2}$$

onde W e $r_{otimo}$ representam a quantidade de parâmetros livres a serem determinados na rede e a proporção de instâncias do subconjunto de treinamento destinadas ao subconjunto de estimação, respectivamente.

## 3.6 Estimação do número de neurônios ocultos por SVD

Testar diversas combinações de hiperparâmetros para encontrar a configuração que irá proporcionar o melhor desempenho de previsão é uma tarefa exaustiva. No presente trabalho, foram realizados 1280 e 1568 experimentos com as redes neurais artificias com uma e duas camadas ocultas, respectivamente. Existem, diversas heurísticas para prever a quantidade de neurônios nas camadas ocultas, tais como as presentes no trabalho de Sheela e Deepa (2013). Estas heurísticas são utilizadas para a finalidade de diminuir a quantidade de testes que precisam ser realizados.

Nesse contexto o trabalho de Santos *et al.* (2010) utilizou uma técnica baseada em SVD para estimar o número ótimo de neurônios na camada oculta de uma MLP aplicada em um problema de classificação. Recentemente, Braga e Moura (2025) adaptaram a técnica para uma rede MLP com duas camadas ocultas aplicada para um problema de regressão, ou seja, situação semelhante ao presente trabalho.

Dessa forma, cada combinação de hiperparâmetros, com exceção do número de

neurônios nas camadas ocultas, foi executada novamente. O número de neurônios das camadas ocultas foi definido para o limite superior presente na Tabela 5, isto é, 40 e 13 neurônios para as redes com uma e duas camadas ocultas, respectivamente.

Em seguida, cada combinação de hiperparâmetros passou por 100 rodadas de treino e teste, onde o conjunto de estimação era utilizado para aprender os pesos sinápticos e depois reapresentado para estimar as matrizes abaixo:

$$\mathbf{Y}^{(h)} = \begin{bmatrix} y_1^{(h)}(1) & y_1^{(h)}(2) & \cdots & y_1^{(h)}(N) \\ y_2^{(h)}(1) & y_2^{(h)}(2) & \cdots & y_2^{(h)}(N) \\ \vdots & \vdots & \ddots & \vdots \\ y_Q^{(h)}(1) & y_Q^{(h)}(2) & \cdots & y_Q^{(h)}(N) \end{bmatrix} \tag{3.3}$$

$$\mathbf{\Delta}^{(h)} = \begin{bmatrix} \delta_1^{(h)}(1) & \delta_1^{(h)}(2) & \cdots & \delta_1^{(h)}(N) \\ \delta_2^{(h)}(1) & \delta_2^{(h)}(2) & \cdots & \delta_2^{(h)}(N) \\ \vdots & \vdots & \ddots & \vdots \\ \delta_Q^{(h)}(1) & \delta_Q^{(h)}(2) & \cdots & \delta_Q^{(h)}(N) \end{bmatrix} \tag{3.4}$$

$$\mathbf{E}^{(h)} = \begin{bmatrix} e_0^{(h)}(1) & e_0^{(h)}(2) & \cdots & e_0^{(h)}(N) \\ e_1^{(h)}(1) & e_1^{(h)}(2) & \cdots & e_1^{(h)}(N) \\ \vdots & \vdots & \ddots & \vdots \\ e_Q^{(h)}(1) & e_Q^{(h)}(2) & \cdots & e_Q^{(h)}(N) \end{bmatrix} \tag{3.5}$$

onde $\mathbf{Y}^{(h)}$, $\mathbf{\Delta}^{(h)}$ e $\mathbf{E}^{(h)}$ as quais representam as saídas, os gradientes locais e os erros de retropropagação de uma camada oculta, respectivamente. Uma vez que as matrizes da $m$-ésima rodada eram determinadas, os seus respectivos conjuntos de valores singulares eram obtidos em ordem decrescente de valor por meio da técnica SVD. Para determinar o número ótimo de neurônios ocultos de cada matriz da $m$-ésima rodada, foi utilizada a regra exposta na Equação 2.124 com um limiar $\gamma$ igual a 0,98. Após todas as rodadas de uma dada combinação de hiperparâmetros, o número de neurônios sugerido por cada matriz foi considerado como o valor $q^*$ mais frequente entre as rodadas daquela combinação.

## 3.7 Modelos de regressão baseados em vetores suporte

No presente trabalho, foram experimentados três tipos de modelos de regressão baseados em vetores suporte: SVR, TSVR e LSSVR. Durante os experimentos, foram testadas as funções kernel do tipo linear e gaussiana. Para cada uma das funções kernel, foram executados

experimentos variando o intervalo de normalização entre [-1;1] e [0;1]. Para cada experimento, foram executadas 100 rodadas, nas quais apenas aquelas com valores de $R^2_{ajust.}$ positivos foram consideradas no cálculo das estatísticas descritivas dos parâmetros de desempenho.

No caso dos experimentos da SVR, o conjunto de dados foi dividido na proporção 80:20, de forma que a maior parte das instâncias foi destinada ao subconjunto de treino. Para os experimentos envolvendo a TSVR e LSSVR, a divisão do conjunto de dados foi realizada na proporção de 20:60:20 a cada rodada, onde 20% dos dados foram utilizados para encontrar os hiperparâmetros ótimos, 60% para treinar o modelo e o restante para testá-lo. Em ambas as normalizações, os subconjuntos foram normalizados utilizando os valores mínimos e máximos do subconjunto de treinamento.

Nos experimentos da SVR, a função *fitrsvm* disponível no matlab foi utilizada para calcular os vetores suporte e os hiperparâmetros da função kernel otimamente. Para os experimentos envolvendo a TSVR, assim como realizado por Peng (2010), a distância dos limites insensíveis até suas respectivas funções e as constantes reguladoras foram mantidas constantes para reduzir a complexidade computacional do modelo, isto é, $\varepsilon = \varepsilon_1 = \varepsilon_2$ e $c = c_1 = c_2$. Com isso, foi realizada uma busca organizada (*grid search*) nos hiperparâmetros, variando $\varepsilon$ no intervalo [0,01; 0,1; 0.5] e $c$ no intervalo [0;1, 1; 10]. Além disso, foi utilizada uma constante $\rho = \frac{1}{2\sigma^2}$ com um valor fixo igual a 0,001 para o kernel gaussiano.

No caso da LSSVR, as constantes reguladoras $\gamma$ e $\sigma$ foram otimamente determinadas utilizando a função *tunelssvm* e o modelo foi treinado utilizando a função *simlssvm*. Ambas as funções estão disponíveis na *toolbox* LSSVM.

## 3.8 Análise de sensibilidade do melhor modelo

Após a seleção dos atributos que serão utilizados para alimentar os modelos de regressão e a determinação do modelo mais preciso, é interessante que se busque verificar a influência de cada atributo na variável-alvo. Nesse intuito, é possível realizar uma análise de sensibilidade do melhor modelo. Segundo Sysoev (2023), essa análise visa estudar a relação entre as entradas e saídas de um modelo.

Há diversas formas de realizar a análise de sensibilidade. No presente trabalho, para observar o efeito da influência de um determinado atributo no teor de silício, os demais atributos foram mantidos em seus valores médios enquanto o referido atributo ficou livre para variar, conforme Braga e Moura (2025). Esse procedimento foi realizado em todos os atributos do

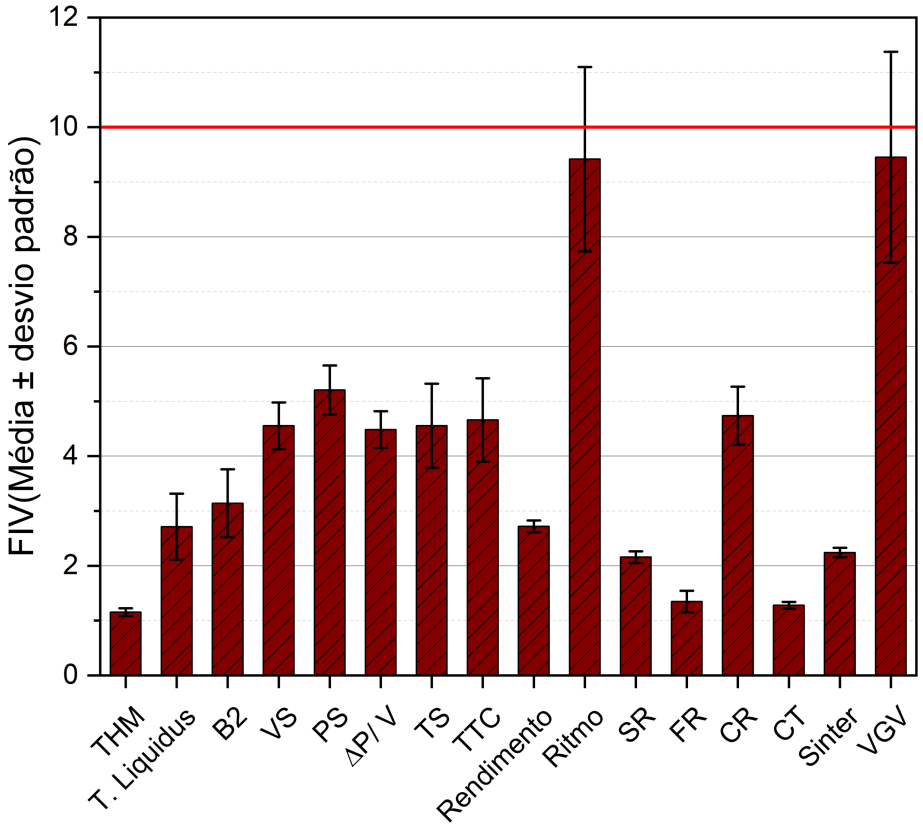
conjunto de dados.

# 4 RESULTADOS

## 4.1 Seleção de atributos

A Figura 18 abaixo revela os resultados da primeira fase da seleção de atributos, que consiste na análise do fator de inflação de variância (FIV) de cada atributo do conjunto de dados.

Figura 18 – Fator de inflação de variância das variáveis do alto-forno




Fonte: Autor (2026)

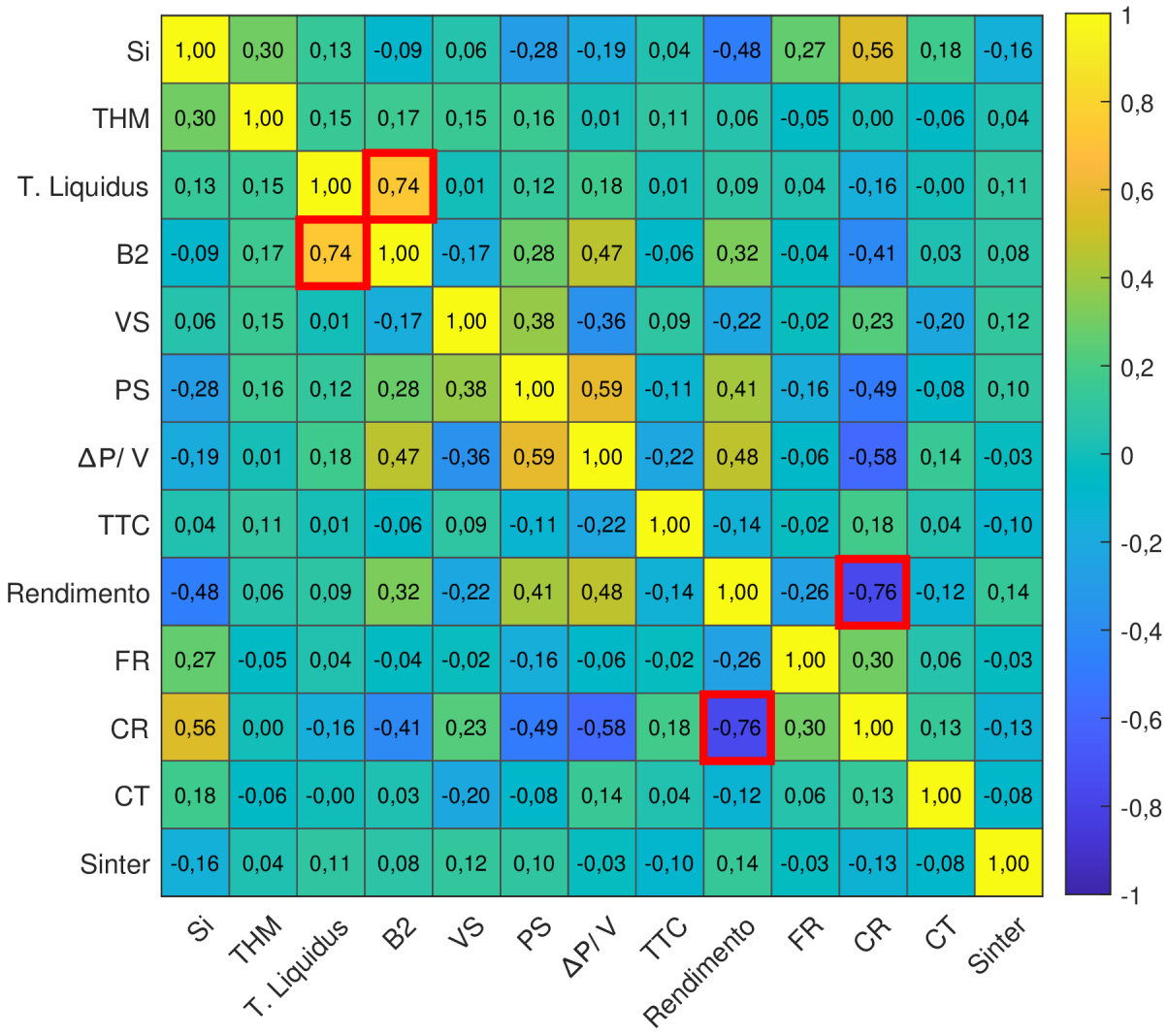
Entre os 16 atributos presentes originalmente no conjunto de dados, o volume de gás nas ventaneiras (VGV) e o ritmo de produção do alto-forno obtiveram os valores médios de FIV mais elevados. Além disso, o desvio padrão sugere que o FIV desses dois atributos foi maior do que 10 em algumas rodadas, indicando que eles são influenciados por outros atributos no conjunto de dados. O ritmo de produção do forno é influenciado por diversos fatores operacionais e pela qualidade da matéria-prima (Geerdes *et al.*, 2020). Além disso, o VGV está correlacionado com o ritmo do alto-forno, afetando a quantidade de gás redutor durante o processo. Diante do exposto, as duas variáveis foram removidas do conjunto de dados.

Na segunda fase da seleção de atributos, a regressão *stepwise* cujo objetivo é selecionar as melhores variáveis para a composição do conjunto de dados, indicou a remoção

da temperatura de sopro (TS) e do sinter *rate* (SR). Essa indicação pode ter ocorrido devido à presença de variáveis que carregam informações semelhantes, como a variável "sinter", que informa o percentual de sinter utilizado na carga que alimenta o alto-forno e a temperatura teórica de chama (TTC), a qual pode ser deduzida por uma relação matemática em que um dos seus termos é a própria, conforme exposto em (Geerdes *et al.*, 2020).

Na terceira e última fase da seleção de atributos, a matriz de correlação dos atributos remanescentes e do teor de silício na Figura 19 foi investigada.

Figura 19 – Matriz de correlação dos dados remanescentes da regressão *stepwise*




Fonte: Autor (2026)

Os pares de atributos [B2, T.*Liquidus*] e [Rendimento, CR] possuem uma correlação estatística maior do que 0,7, indicando que eles possuem uma forte correlação. Por apresentarem as menores correlações com o teor de silício, a basicidade binária e o rendimento gasoso

foram removidos do conjunto de dados. No primeiro par, sabe-se que a basicidade binária e a temperatura *liquidus* dependem da composição da escória. Além disso, (Wang *et al.*, 2022) reportam em seus estudos que um acréscimo na basicidade binária levou a um aumento na temperatura *liquidus* da escória.

No segundo par, o rendimento gasoso e o *coke rate* (CR) são inversamente proporcionais (Wu *et al.*, 2012). Enquanto o rendimento gasoso indica a efetividade dos gases redutores, principalmente $CO_2$, em remover oxigênio do minério de ferro, o *coke rate* (CR) se refere à quantidade de coque carregada no alto-forno por tonelada de ferro-gusa produzida, lembrando que o coque é fonte de gases redutores durante o processo.

## 4.2 Resultados da regressão linear múltipla

Após a seleção de atributos, o conjunto de dados passou a ser composto por dez atributos e o teor de silício. Em seguida, utilizou-se um modelo de regressão linear múltipla, cujo os resultados podem ser encontrados abaixo:

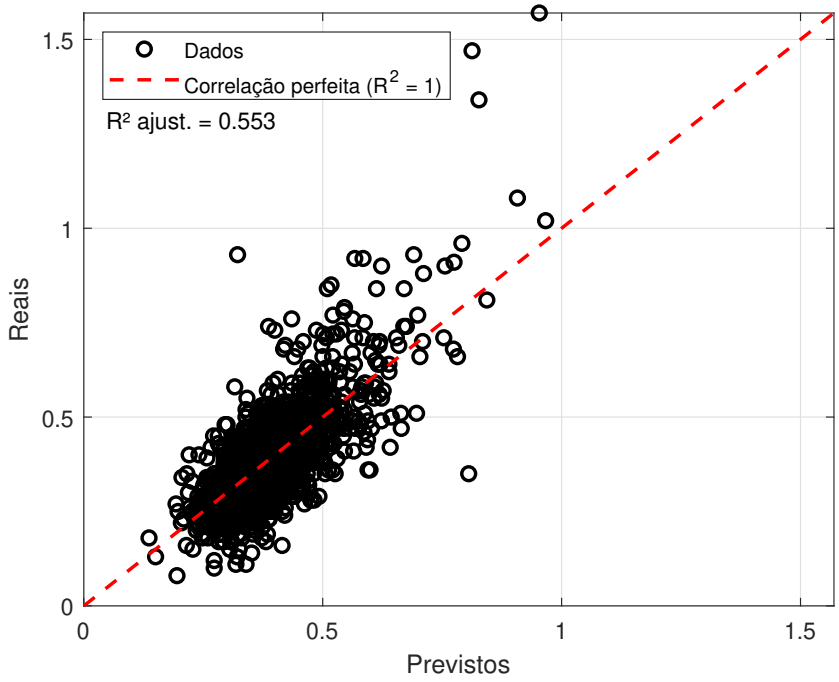
Tabela 6 – Resultados dos experimentos da RLM

| | $\mathbf{R^2}$ | $\mathbf{R^2_{ajust.}}$ | **MAE** | **MAPE** | **RMSE** |
|---|---|---|---|---|---|
| Média | 0,499 | 0,496 | 0,068 | 18,585 | 0,0930 |
| Máximo | **0,556** | **0,553** | 0,072 | 19,692 | 0,0103 |
| Mínimo | 0,405 | 0,401 | **0,062** | **17,192** | **0.0837** |
| Mediana | 0,502 | 0,498 | 0,068 | 18,574 | 0,0928 |
| Desvio Padrão | 0,0288 | 0,0290 | 0,0016 | 0,4836 | 0,0035 |

Fonte: Autor (2026)

Observando a Tabela 6, percebe-se a RLM atingiu, para 100 rodadas de treinamento e teste, um $R^2_{ajust.}$ máximo de 0,553, um MAE mínimo de 0,062, representando um erro de aproximadamente 17%, e um RMSE mínimo de 0,0837. A Figura 20 abaixo apresenta o gráfico de dispersão entre os valores reais e preditos da melhor rodada da RLM, na qual nota-se que algumas instâncias apresentaram um erro de previsão elevados. Isso pode acontecer devido à presença de *outliers* no conjunto de dados. Os resultados da RLM permitirão a comparação do desempenho de modelos complexos.

Figura 20 – Grafico de dispersão entre valores reais e preditos da melhor rodada da RLM




Fonte: Autor (2026)

### 4.3 Resultados do *perceptron* logístico

O *perceptron* logístico representa um pequeno avanço no grau de complexidade dos modelos, devido à introdução de funções de ativação com inclinação suave. Os resultados dos experimentos desse modelo podem ser encontrados a seguir:

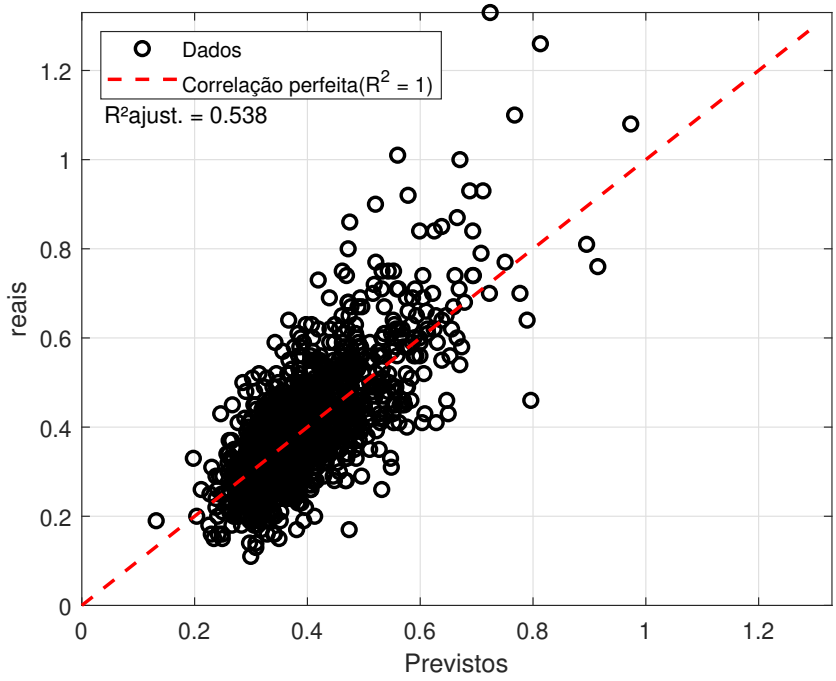
Tabela 7 – $R^2_{ajust.}$ dos experimentos do *perceptron logístico*, ordenados pela média.

| Exp. | $\varphi(.)$ | a | b | c | $\eta$ | $R^2_{ajust.}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Média | Mínimo | Máximo | Mediana | Desvio |
| 1 | tanh | 1 | 1 | - | $10^{-4}$ | 0,4307 | 0,2729 | 0,5279 | 0,4351 | 0,0491 |
| 2 | tanh | 1,7159 | (2/3) | - | $10^{-4}$ | 0,4263 | 0,2740 | 0,5386 | 0,4302 | 0,0555 |
| 3 | sig | - | - | 1 | $10^{-4}$ | 0,1655 | -0,2185 | 0,3709 | 0,1858 | 0,1207 |
| 4 | sig | - | - | 2,2 | $10^{-4}$ | 0,0921 | -0,3419 | 0,4966 | 0,0989 | 0,1664 |

Fonte: Autor (2026)

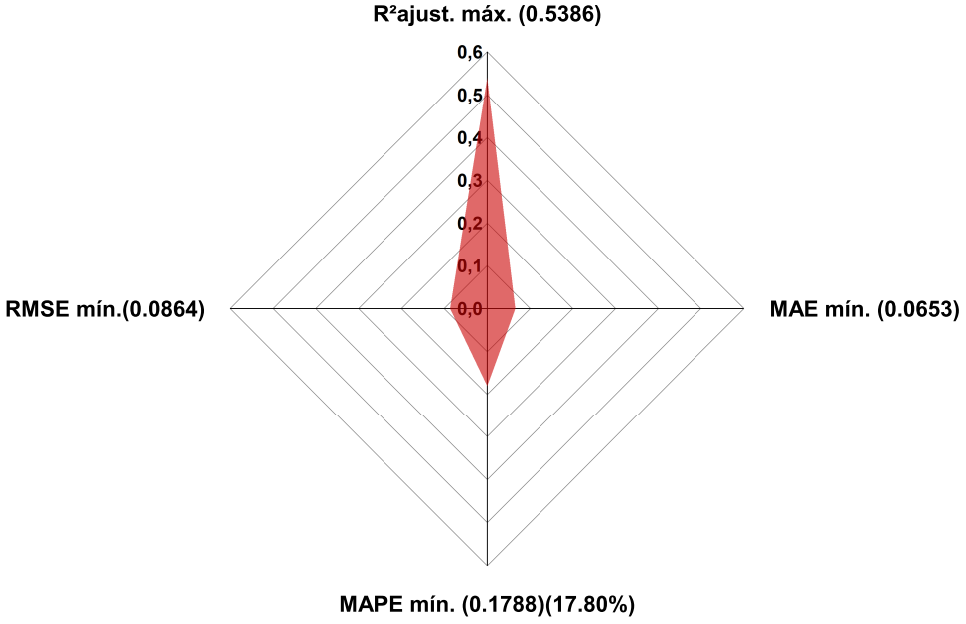
A Tabela 7, revela que os experimentos do *perceptron* logístico que utilizaram a tangente hiperbólica obtiveram $R^2_{ajust.}$ significativamente maiores do que aqueles que utilizaram a sigmoide logística, os quais inclusive obtiveram valores negativos em algumas rodadas. De forma geral, as tabelas 6 e 7 mostram que *perceptron* logístico obteve um desempenho consideravelmente inferior à RLM.

Figura 21 – Grafico de dispersão entre valores reais e preditos da melhor rodada do *perceptron* logístico




Fonte: Autor (2026)

Figura 22 – Gráfico radar da melhor rodada do experimento do *perceptron* logístico com maior $R^2_{ajust.}$




Fonte: Autor (2026)

Até mesmo no melhor caso, o *perceptron* logístico obteve um desempenho inferior, pois a Tabela 6 e a Figura 22 mostram que o coeficiente de determinação ajustado da RLM foi mais elevado, assim como as outras medidas de avaliação do *perceptron* logístico, o que não é positivo para o *perceptron*, pois essas outras medidas tratam de diferentes tipos de erro. Logo, o *perceptron* logístico foi menos preciso. Além disso, vale destacar que o diagrama de dispersão na Figura 21 também mostra que há instâncias com erro de previsão elevado, destacando a presença dos *outliers*.

## 4.4 Resultados das redes neurais

As tabelas 8 e 9 sumarizam os dez melhores experimentos da rede neural com uma única camada oculta baseando-se nos valores médios e máximos de $R^2_{ajust.}$ respectivamente.

Tabela 8 – $R^2_{ajust.}$ dos experimentos da MLP com 1 camada oculta, ordenados pela média.

| Exp. | Nº neurônios | $\varphi(v)$ | a:b:c | $\eta$ | $\alpha$ | $R^2_{\text{ajust.}}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Média | Mínimo | Máximo | Mediana | Desvio |
| 601 | 19-1 | tanh-lin | 1:1: – | 0,01 | 0,9 | 0,5644 | 0,4822 | 0,6410 | 0,5703 | 0,0324 |
| 828 | 26-1 | tanh-lin | 1:1: – | 0,001 | 0,95 | 0,5638 | 0,4972 | 0,6173 | 0,5696 | 0,0265 |
| 577 | 19-1 | tanh-tanh | 1:1: – | 0,01 | 0,9 | 0,5632 | 0,4406 | 0,6243 | 0,5656 | 0,0315 |
| 764 | 24-1 | tanh-lin | 1:1: – | 0,001 | 0,95 | 0,5626 | 0,5038 | 0,6252 | 0,5640 | 0,0248 |
| 865 | 28-1 | tanh-tanh | 1:1: – | 0,01 | 0,9 | 0,5622 | 0,4469 | 0,6281 | 0,5653 | 0,0351 |
| 633 | 20-1 | tanh-lin | 1:1: – | 0,01 | 0,9 | 0,5619 | 0,3995 | 0,6329 | 0,5642 | 0,0362 |
| 481 | 16-1 | tanh-tanh | 1:1: – | 0,01 | 0,9 | 0,5618 | 0,4448 | 0,6461 | 0,5609 | 0,0303 |
| 769 | 25-1 | tanh-tanh | 1:1: – | 0.01 | 0,9 | 0,5617 | 0,4772 | 0,6286 | 0,5635 | 0,0323 |
| 700 | 22-1 | tanh-lin | 1:1: – | 0,001 | 0,95 | 0,5607 | 0,3115 | 0,6150 | 0,5666 | 0,0394 |
| 1025 | 33-1 | tanh-tanh | 1:1: – | 0,01 | 0,9 | 0,5606 | 0,4540 | 0,6153 | 0,5580 | 0,0306 |

Fonte: Autor (2026)

Tabela 9 – $R^2_{ajust.}$ dos experimentos da MLP com 1 camada oculta, ordenados pelo valor máximo.

| Exp. | Nº neurônios | $\varphi(v)$ | a:b:c | $\eta$ | $\alpha$ | $R^2_{\text{ajust.}}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Média | Mínimo | Máximo | Mediana | Desvio |
| 644 | 21-1 | tanh-tanh | 1:1:– | 0,001 | 0.95 | 0,5563 | 0,4634 | 0,6522 | 0,5567 | 0,0316 |
| 353 | 12-1 | tanh-tanh | 1:1:– | 0,01 | 0,9 | 0,5572 | 0,4691 | 0,6514 | 0,5628 | 0,0315 |
| 226 | 8-1 | tanh-tanh | 1:1:– | 0,01 | 0,95 | 0,5546 | 0,4608 | 0,6471 | 0,5599 | 0,0334 |
| 481 | 16-1 | tanh-tanh | 1:1:– | 0,01 | 0,9 | 0,5618 | 0,4448 | 0,6461 | 0,5609 | 0,0303 |
| 513 | 17-1 | tanh-tanh | 1:1:– | 0,01 | 0,9 | 0,5557 | 0,4586 | 0,6453 | 0,5596 | 0,0339 |
| 1113 | 35-1 | tanh-lin | 1:1:– | 0,01 | 0,9 | 0,5547 | 0,4317 | 0,6453 | 0,5584 | 0,0388 |
| 1157 | 37-1 | tanh-tanh | 1,7159:(2/3) | 0,01 | 0,9 | 0,5456 | 0,3919 | 0,6442 | 0,5486 | 0,0364 |
| 225 | 8-1 | tanh-tanh | 1:1:– | 0,01 | 0,9 | 0,5594 | 0,4700 | 0,6439 | 0,5609 | 0,0315 |
| 289 | 10-1 | tanh-tanh | 1:1:– | 0,01 | 0,9 | 0,5595 | 0,4738 | 0,6424 | 0,5607 | 0,0300 |
| 409 | 13-1 | tanh-lin | 1:1:– | 0,01 | 0,9 | 0,5547 | 0,4532 | 0,6420 | 0,5593 | 0,0356 |

Fonte: Autor (2026)

Comparando os resultados da MLP com uma camada oculta aos obtidos pela RLM,

verifica-se uma melhoria expressiva na capacidade de previsão. As tabelas 7 e 8 mostram um aumento de até aproximadamente 6% nos valores médios de $R^2_{ajust.}$. No melhor cenário, as tabelas 7 e 9 e a Figura 24 mostram um aumento de aproximadamente 10% no $R^2_{ajust.}$ da MLP com uma camada oculta em comparação com a RLM. Comparando as demais medidas de desempenho da RLM e da MLP com uma camada oculta, percebe-se que estas possuem valores inferiores. Apesar dos parâmetros de desempenho apresentarem uma melhora significativa, vale salientar que o diagrama de dispersão da MLP com uma camada oculta na Figura 23 ainda apresenta instâncias com erros de previsão consideráveis.

Diante do exposto e considerando que o *perceptron* logístico é equivalente a uma rede neural sem camadas ocultas, o salto expressivo na capacidade de previsão justificou a adição de uma camada oculta. Isso cria uma certa expectativa para a MLP com duas camadas ocultas, na esperança de que ela seja ainda mais precisa em suas previsões.

As tabelas 10 e 11 expõem os resultados dos dez melhores experimentos para a rede neural com duas camadas ocultas baseados nos valores médios e máximos do $R^2_{ajust.}$, respectivamente.

Tabela 10 – $R^2_{ajust.}$ dos experimentos da MLP com duas camadas ocultas, ordenados pela média.

| Exp. | Nº neurônios | $\varphi(v)$ | a:b:c | $\eta$ | $\alpha$ | $R^2_{\text{ajust.}}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Média | Mínimo | Máximo | Mediana | Desvio |
| 396 | 8-12-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5589 | 0,4590 | 0,6256 | 0,5660 | 0,0361 |
| 1228 | 12-10-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5569 | 0,4387 | 0,6214 | 0,5578 | 0,0341 |
| 428 | 8-13-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5564 | 0,4068 | 0,6245 | 0,5646 | 0,0363 |
| 844 | 10-12-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5556 | 0,4445 | 0,6168 | 0,5604 | 0,0354 |
| 1196 | 12-9-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5555 | 0,4146 | 0,6134 | 0,5607 | 0,0352 |
| 108 | 7-10-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5553 | 0,4011 | 0,6484 | 0,5608 | 0,0437 |
| 1100 | 11-13-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5553 | 0,4433 | 0,6429 | 0,5553 | 0,0335 |
| 1516 | 13-12-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5549 | 0,4222 | 0,6408 | 0,5613 | 0,0401 |
| 140 | 7-11-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5549 | 0,4175 | 0,6248 | 0,5576 | 0,0324 |
| 76 | 7-9-1 | tanh-tanh-lin | 1:1:– | 0,001 | 0,95 | 0,5544 | 0,4628 | 0,6265 | 0,5554 | 0,0383 |

Fonte: Autor (2026)

Tabela 11 – $R^2_{ajust.}$ dos experimentos da MLP com duas camadas ocultas, ordenados pelo valor máximo.

| Exp. | Nº neurônios | $\varphi(v)$ | a:b:c | $\eta$ | $\alpha$ | $R^2_{ajust.}$ | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Média | Mínimo | Máximo | Mediana | Desvio |
| 770 | 10-10-1 | tanh-tanh-tanh | 1:1:- | 0,01 | 0,95 | 0,5465 | 0,4400 | 0,6616 | 0,5464 | 0,0395 |
| 97 | 7-10-1 | tanh-tanh-tanh | 1:1:- | 0,01 | 0,9 | 0,5432 | 0,4243 | 0,6579 | 0,5435 | 0,0337 |
| 1389 | 13-8-1 | tanh-tanh-lin | 1.7159:(2/3):- | 0,01 | 0,9 | 0,5305 | 0,2901 | 0,6521 | 0,5413 | 0,0562 |
| 1454 | 13-10-1 | tanh-tanh-lin | 1,7159:(2/3):- | 0,01 | 0,95 | 0,4834 | -0,1443 | 0.6507 | 0,5308 | 0,1405 |
| 1005 | 11-10-1 | tanh-tanh-lin | 1,7159:(2/3):- | 0,01 | 0,9 | 0,5326 | -0,0472 | 0,6506 | 0,5438 | 0,0778 |
| 1410 | 13-9-1 | tanh-tanh-tanh | 1:1:- | 0,01 | 0,95 | 0,5434 | 0,3343 | 0,6489 | 0,5454 | 0,0453 |
| 972 | 11-9-1 | tanh-tanh-lin | 1:1:- | 0,001 | 0,95 | 0,5487 | 0,4468 | 0,6485 | 0,5513 | 0,0380 |
| 108 | 7-10-1 | tanh-tanh-lin | 1:1:- | 0,001 | 0.95 | 0,5553 | 0,4011 | 0,6484 | 0,5608 | 0,0437 |
| 1421 | 13-9-1 | tanh-tanh-lin | 1.7159:(2/3):- | 0,01 | 0,9 | 0,5243 | 0,2621 | 0,6479 | 0,5332 | 0,0601 |
| 1387 | 13-8-1 | tanh-tanh-lin | 1:1:- | 0,001 | 0,9 | 0,5395 | 0,4380 | 0,6474 | 0,5438 | 0,0421 |

Fonte: Autor (2026)

As tabelas 10 e 11, revelam que uma única combinação de hiperparâmetros obteve os maiores $R^2_{ajust.}$ médios e que nos experimentos expostos, a função sigmoide logística também não estava presente, assim como na MLP com uma camada oculta.

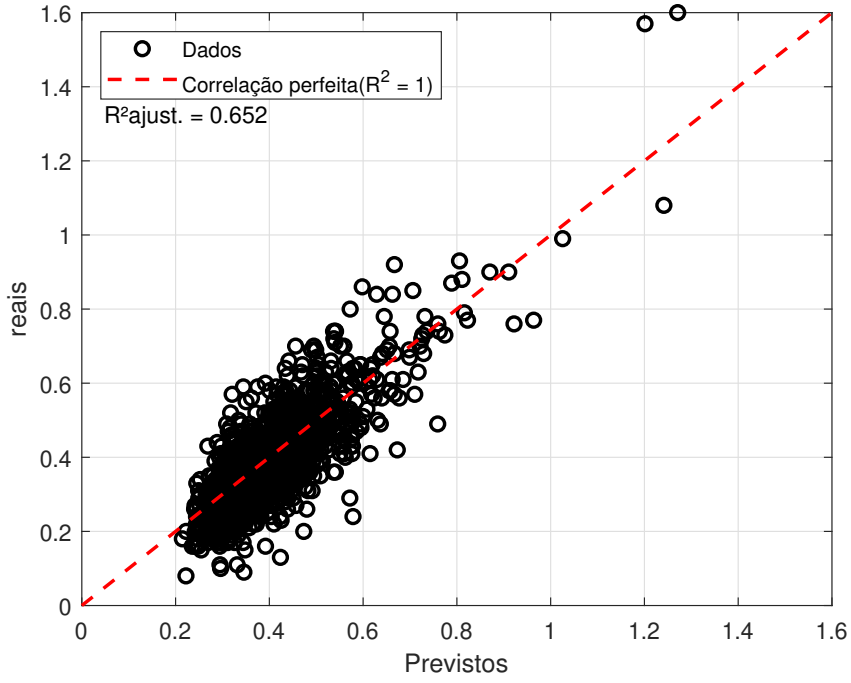
Apesar da rede neural com duas camadas ocultas também possuir uma precisão mais elevada do que a RLM, as tabelas 8, 9, 10 e 11 mostram que o ganho no desempenho entre as MLPs com uma e duas camadas ocultas é muito baixo, pois em termos de valores médios de $R^2_{ajust.}$, os valores obtidos pela MLP com uma camada oculta foram ligeiramente superiores e em termos de valores máximos, a MLP com duas camadas ocultas obteve valores até 1% maiores, aproximadamente.

As outras medidas de avaliação de desempenho da MLP com duas camadas ocultas presentes na Figura 26 também indicam desempenho similar, pois são ligeiramente inferiores. Além disso, vale salientar que o diagrama de dispersão na Figura 25 também indica a presença de *outliers*. Diante do exposto, a adição da segunda camada oculta não se traduziu em um aumento satisfatório na capacidade de previsão do modelo, o que desmotivou a execução de experimentos de redes neurais com três camadas ocultas, as quais também elevariam exageradamente a quantidade de testes que seriam executados.

Além disso, analisando as tabelas 8, 9, 10 e 11, nota-se a ausência da sigmoide logística como função de ativação da camada de saída. Vale ainda lembrar que para os experimentos do *perceptron* logístico na Tabela 7, aqueles que utilizaram a sigmoide logística levaram aos piores resultados. Esse padrão também foi encontrado por Murta *et al.* (2021) ao prever propriedades mecânicas de vergalhões de aço utilizados na construção civil. De acordo com Mahima *et al.* (2023), esse desempenho está associado ao fato de que a tangente hiperbólica possui um gradiente mais acentuado, favorecendo uma convergência mais rápida, e à sua simetria

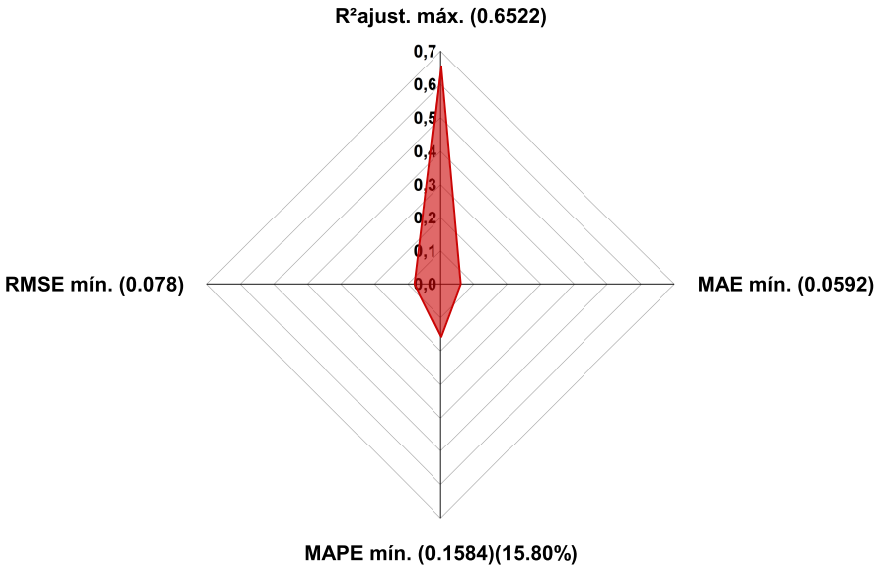
em torno de zero que reduz o problema do gradiente evanescente.

Figura 23 – Diagrama de dispersão da melhor rodada da MLP com uma camada oculta



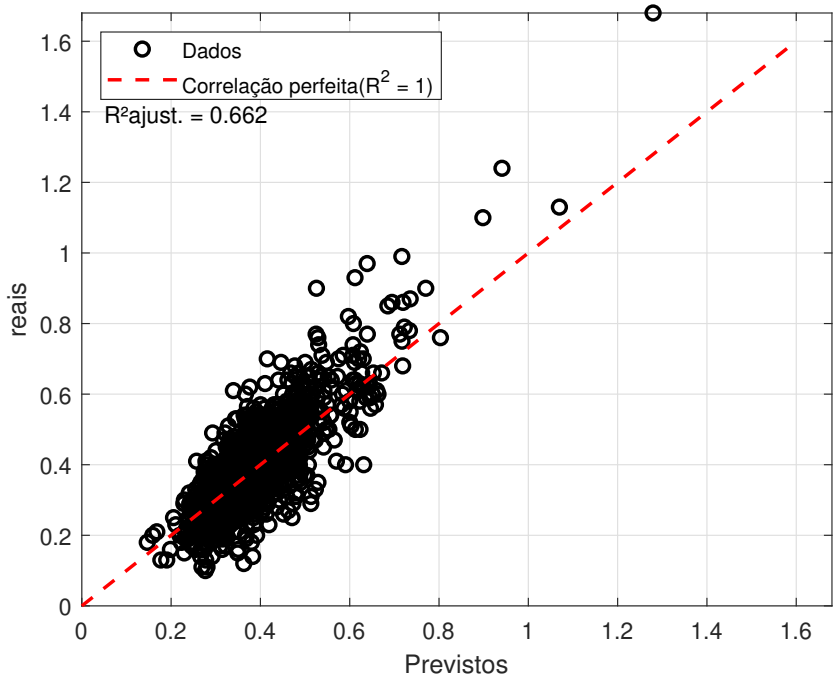

Fonte: Autor (2026)

Figura 24 – Gráfico radar da melhor rodada do experimento da MLP com uma camada oculta com maior $R^2_{ajust.}$



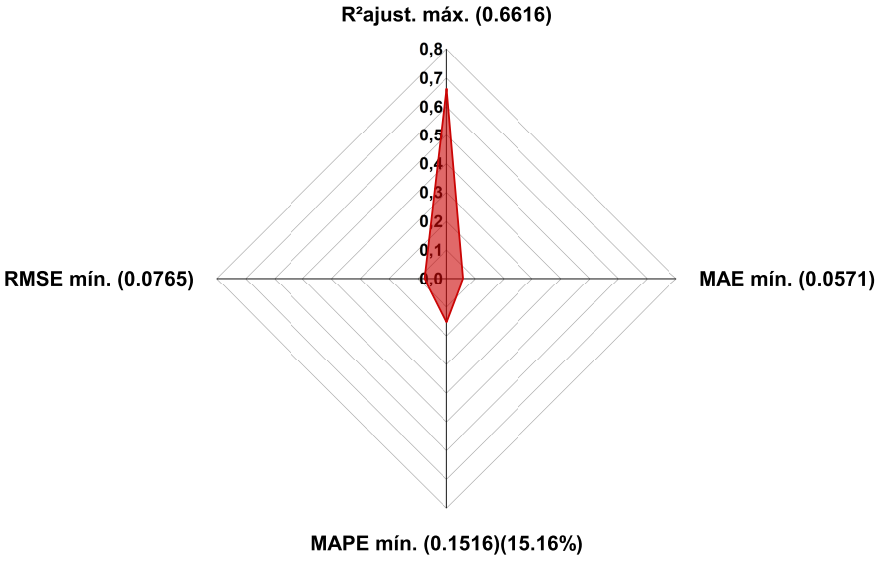

Fonte: Autor (2026)

Figura 25 – Diagrama de dispersão da melhor rodada da MLP com duas camadas ocultas




Fonte: Autor (2026)

Figura 26 – Gráfico radar da melhor rodada do experimento da MLP com duas camadas ocultas com maior $R^2_{ajust.}$




Fonte: Autor (2026)

## 4.5 Resultados da estimação do número de neurônios ocultos pela SVD

A Tabela 12 a seguir ilustra o resultado da estimativa do número de neurônios ocultos para as redes neurais com uma e duas camadas ocultas

Tabela 12 – Resultado da estimação do número de neurônios ocultos por SVD

| Rede neural | $\mathbf{Y}^{(h)}$ | $\mathbf{\Delta}^{(h)}$ | $\mathbf{E}^{(h)}$ |
|---|---|---|---|
| MLP com uma camada oculta | 7 | 1 | 3 |
| MLP com duas camadas ocultas - 1ª camada oculta | 10 | 5 | 10 |
| MLP com duas camadas ocultas - 2ª camada oculta | 10 | 5 | 10 |

Fonte: Autor (2026)

Para a MLP com uma camada oculta, as matrizes $\mathbf{Y}^{(h)}$, $\mathbf{\Delta}^{(h)}$ e $\mathbf{E}^{(h)}$ sugeriram 7, 1 e 3 neurônios ocultos, respectivamente. Comparando essas sugestões com a Tabela 9, observa-se uma diferença de um neurônio entre o valor indicado pela matriz $\mathbf{Y}^{(h)}$ e o número de neurônios ocultos no experimento 226, o qual obteve apenas o terceiro maior valor de $R^2_{ajust.}$. No caso da MLP com duas camadas ocultas, as matrizes sugeriram os mesmos valores para ambas as camadas e além disso, as matrizes $\mathbf{Y}^{(h)}$ e $\mathbf{E}^{(h)}$ sugeriram o mesmo número de neurônios ocultos, os quais coincidem exatamente com o experimento 770 na Tabela 11, que obteve o maior valor de $R^2_{ajust}$ de todas as combinações de hiperparâmetros testadas.

Em seu trabalho, Santos *et al.* (2010) afirmam que uma estimativa ruim da matriz $\mathbf{E}^{(h)}$ em um dos conjuntos de dados testados foi causada pela presença de colunas linearmente dependentes. Diante do exposto e dos resultados encontrados no presente trabalho, o uso da decomposição em valores singulares para estimar o número de neurônios é encorajado, mas com cautela. Sugere-se o uso de uma janela de testes com até dois ou três neurônios de diferença dos valores sugeridos pelas matrizes do SVD.

## 4.6 Resultados da SVR

A Tabela 13 revela o resultado dos experimentos da SVR. A diferença de $R^2_{ajust.}$ entre os experimentos que utilizaram as funções kernel gaussiana e linear chegou até aproximadamente 8%, alcançando a diferença máxima quando a normalização no intervalo [-1;1] ocorreu. Comparando o desempenho por intervalo de normalização, verifica-se que o $R^2_{ajust.}$ sofreu variações de até 1%. O kernel gaussiano foi notavelmente superior, ressaltando a não linearidade da tarefa de regressão. No melhor cenário, ou seja, utilizando o kernel gaussiano e normalizando os dados

no intervalo [-1;1], foram utilizados 5543 vetores suporte, o que representa cerca de 99% do subconjunto de treinamento.

Tabela 13 – Resultados da SVR

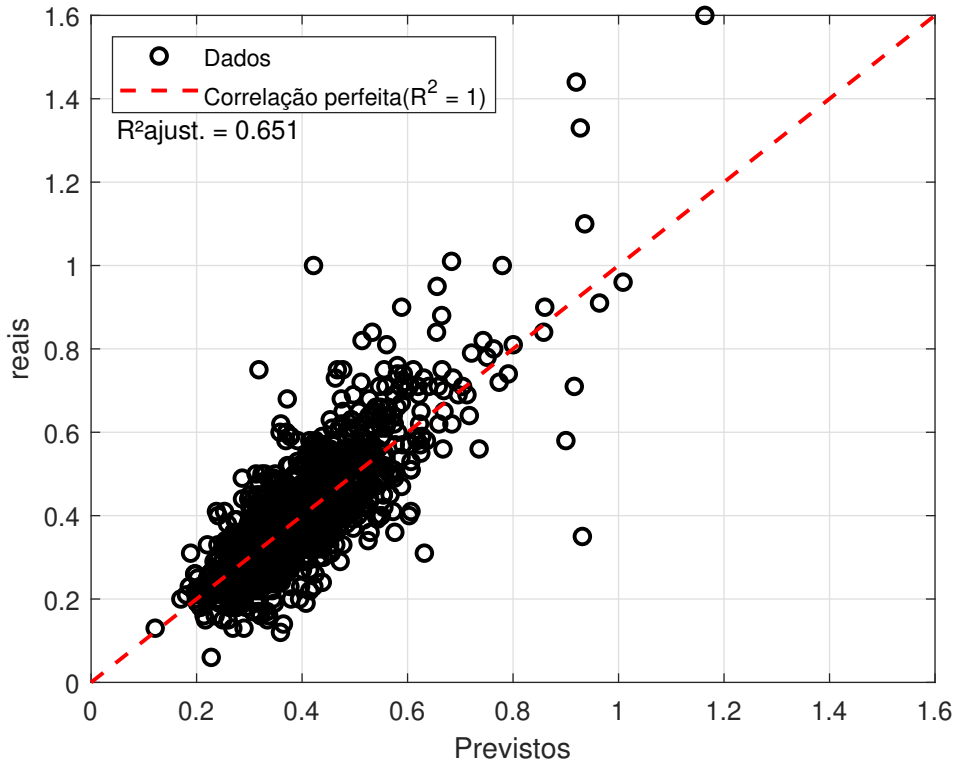
| Kernel | Normalização | $R^2_{\text{adj.}}$ Média | $R^2_{\text{adj.}}$ Mínimo | $R^2_{\text{adj.}}$ Máximo | $R^2_{\text{adj.}}$ Mediana | $R^2_{\text{adj.}}$ Desvio |
|---|---|---|---|---|---|---|
| Gaussiano | min-max(-1,1) | 0,5223 | 0,0203 | 0,6506 | 0,5545 | 0,1158 |
| Linear | min-max(-1,1) | 0,4477 | 0,0468 | 0,5539 | 0,4773 | 0,0864 |
| Gaussiano | min-max(0,1) | 0,5239 | 0,0831 | 0,6378 | 0,5466 | 0,0985 |
| Linear | min-max(0,1) | 0,4514 | 0,0121 | 0,5474 | 0,4739 | 0,0937 |

Fonte: Autor (2026)

A SVR obteve desempenho significativamente superior à RLM. Ao analisar as tabelas 6 e 13, observa-se que o maior $R^2_{ajust.}$ médio da SVR, obtido com o kernel gaussiano e normalização entre [0;1], sendo superior em até 2% em relação ao $R^2_{ajust.}$ médio da RLM. Além disso, houve uma diferença de aproximadamente 10% no $R^2_{ajust.}$ máximo, considerando o kernel gaussiano com normalização entre [-1;1]. Analisando as outras medidas de desempenho das melhores rodadas de cada modelo, conforme a Tabela 6 e a Figura 27, também é possível notar um desempenho superior da SVR em relação à RLM.

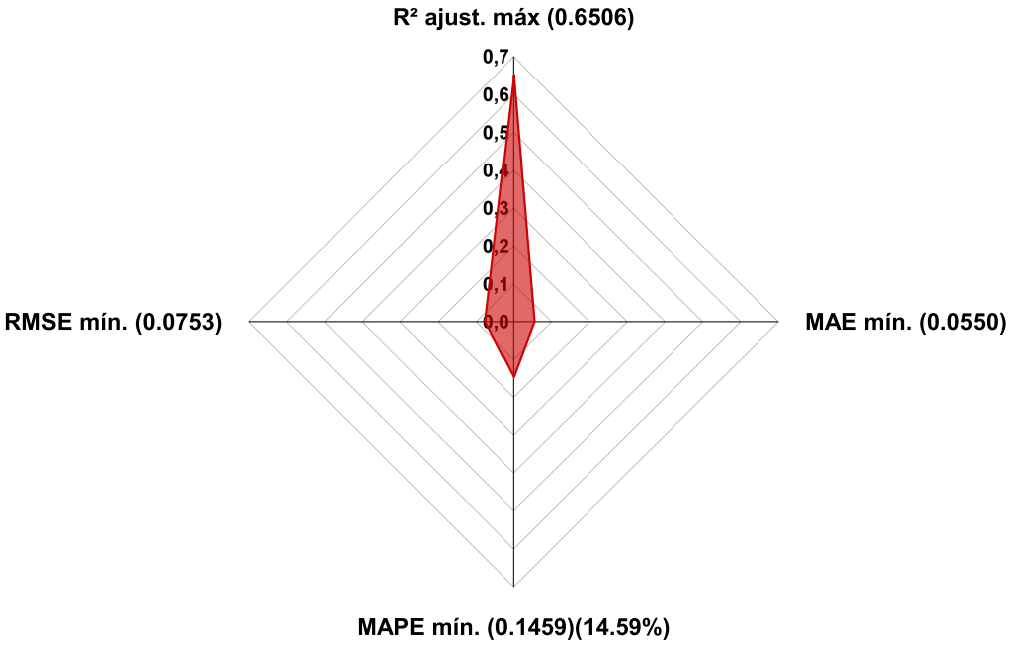
Os experimentos na Tabela 13 revelam que a SVR alcançou um desempenho inferior à rede neural com uma única camada oculta, pois comparando-os com os experimentos na Tabela 8, percebe-se que os experimentos da rede neural com uma camada oculta possuem valores maiores de $R^2_{ajust.}$ médio. As tabelas 9 e 13 revelam uma diferença pequena entre os valores de $R^2_{ajust.}$ máximo do experimento 644 e da SVR com os dados normalizados no intervalo [-1;1]. Essa pequena diferença nos melhores cenários de ambos os modelos também pode ser vista nos gráficos radar das figuras 24 e 28. Além disso, os gráficos de dispersão entre os valores reais e preditos da MLP com uma camada oculta e da SVR possuem comportamento semelhante.

Figura 27 – Diagrama de dispersão da melhor rodada da SVR




Fonte: Autor (2026)

Figura 28 – Gráfico radar da melhor rodada da SVR




Fonte: Autor (2026)

## 4.7 Resultados da TSVR

A Tabela 14 apresenta os resultados do segundo modelo baseado em vetores suporte, a TSVR.

Tabela 14 – Resultados da TSVR.

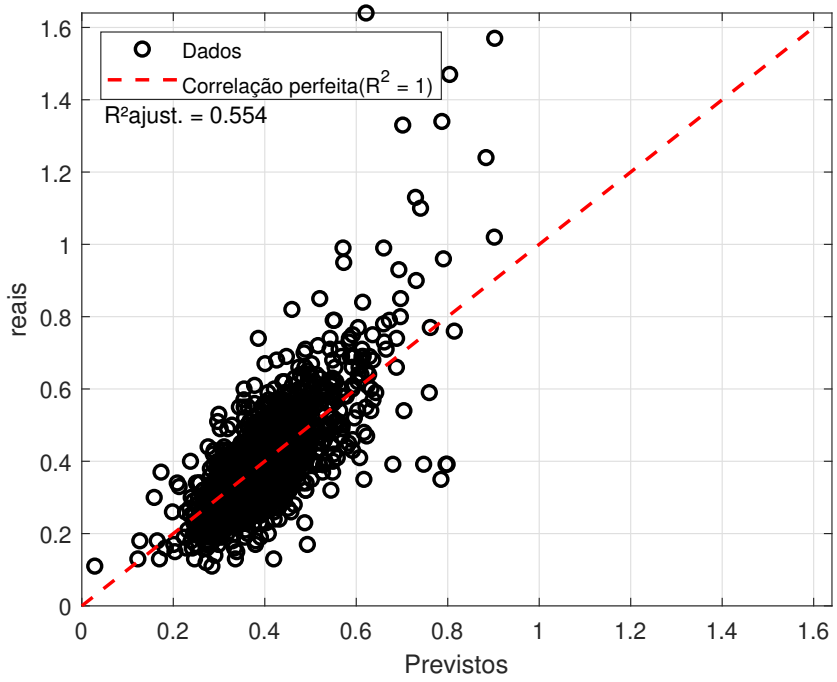
| Kernel | Normalização | $R^2_{\text{adj.}}$ Média | $R^2_{\text{adj.}}$ Mínimo | $R^2_{\text{adj.}}$ Máximo | $R^2_{\text{adj.}}$ Mediana | $R^2_{\text{adj.}}$ Desvio |
|---|---|---|---|---|---|---|
| Gaussiano | min-max(-1,1) | 0,4892 | 0,2248 | 0,5542 | 0,4948 | 0,0476 |
| Linear | min-max(-1,1) | 0,4685 | 0,0258 | 0,5521 | 0,4885 | 0,0821 |
| Gaussiano | min-max(0,1) | 0,4876 | 0,2352 | 0,5542 | 0,4897 | 0,0404 |
| Linear | min-max(0,1) | 0,4826 | 0,1676 | 0,5530 | 0,4947 | 0,0543 |

Fonte: Autor (2026)

Novamente, os valores entre os diferentes intervalos de normalização apresentaram apenas uma pequena diferença no $R^2_{ajust.}$ de forma geral, destacando-se que o melhor desempenho foi atingido quando a função kernel gaussiana foi utilizada, independentemente do tipo de normalização. Diante dessa situação, o $R^2_{ajust.}$ médio foi eleito como critério de desempate, favorecendo a normalização no intervalo [-1;1].

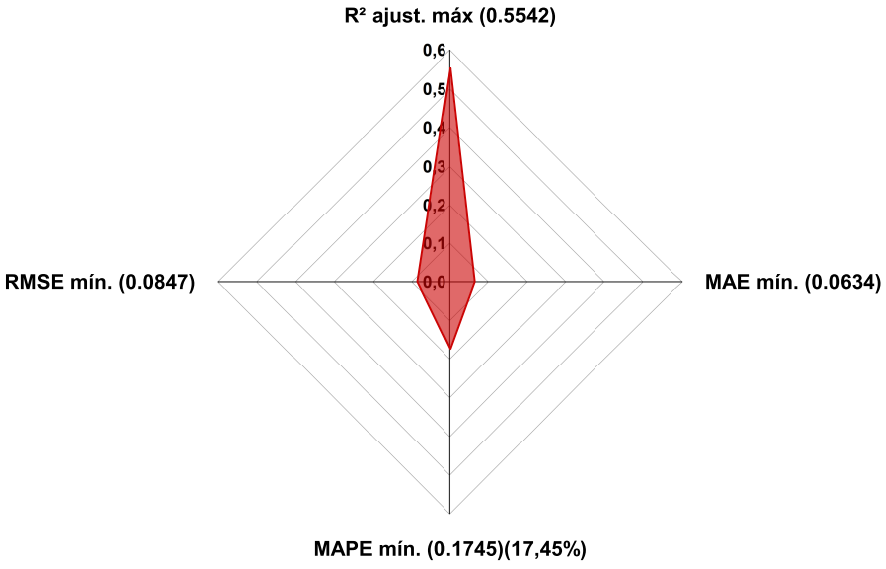
Comparando as performances da TSVR e da RLM, verifica-se que a TSVR obteve um desempenho ligeiramente inferior, pois as tabelas 6 e 14 mostram valores menores de $R^2_{ajust.}$ médio para a TSVR, apesar desta ter obtido valores similares de $R^2_{ajust.}$ máximo, o que também reflete na semelhança entre as medidas de desempenho no gráfico radar da Figura 30 e da Tabela 6. Uma vez que a TSVR obteve um desempenho no máximo comparável à RLM, a primeira também obteve medidas inferiores à SVR e às redes neurais. Em contraste com a SVR, a TSVR utilizou apenas 2391 vetores suporte, aproximadamente 57% do subconjunto de treinamento.

Figura 29 – Diagrama de dispersão da melhor rodada da TSVR




Fonte: Autor (2026)

Figura 30 – Gráfico radar da melhor rodada da TSVR




Fonte: Autor (2026)

## 4.8 Resultados da LSSVR

A Tabela 15 revela os resultados do terceiro modelo baseado em vetores suporte e o último experimentado no presente trabalho.

Tabela 15 – Resultados da LSSVR.

| Kernel | Normalização | $R^2_{\text{adj.}}$ Média | $R^2_{\text{adj.}}$ Mínimo | $R^2_{\text{adj.}}$ Máxima | $R^2_{\text{adj.}}$ Mediana | $R^2_{\text{adj.}}$ Desvio |
|---|---|---|---|---|---|---|
| Gaussiano | min-max(0,1) | 0,5914 | 0,4870 | 0,6537 | 0,5926 | 0,0295 |
| Linear | min-max(0,1) | 0,4857 | 0,3851 | 0,5454 | 0,4934 | 0,0336 |
| Gaussiano | min-max(-1,1) | 0,5830 | 0,2166 | 0,6450 | 0,5899 | 0,0524 |
| Linear | min-max(-1,1) | 0,4849 | 0,1963 | 0,5475 | 0,4960 | 0,0494 |

Fonte: Autor (2026)

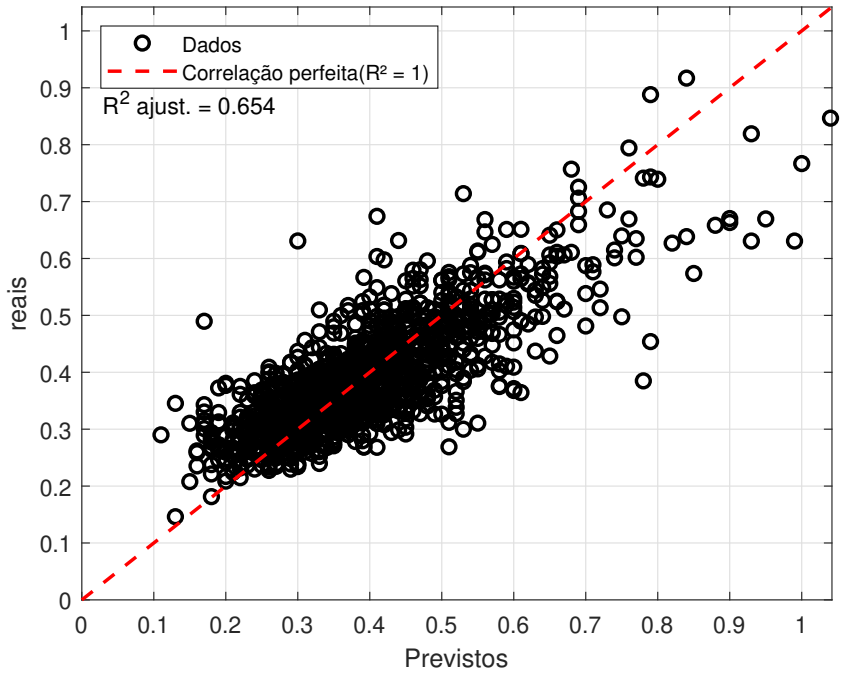
Assim como nos modelos baseados em vetores suporte anteriores, a função kernel gaussiana se destacou em relação à função linear, com uma diferença mais expressiva do que a apresentada na TSVR e similar à SVR, com valores chegando até aproximadamente 11% no $R^2_{ajust.}$ médio. Em relação aos intervalos de normalização utilizados, a diferença no impacto na performance da LSSVR foi pequena.

Comparando o desempenho da LSSVR com os outros modelos baseados em vetores suporte, mais especificamente, a SVR devido ao seu maior sucesso em relação à TSVR, é possível perceber que a LSSVR obteve a melhor performance. As figuras 28 e 32 mostram que a SVR e a LSSVR possuem performances similares no melhor cenário de ambos os modelos. Apesar disso, as tabelas 13 e 15 indicam diferenças claras nos valores de $R^2_{ajust.}$ médio, os quais são mais elevados na LSSVR. Diante do exposto, vale lembrar que a LSSVR não produz soluções esparsas, logo, foram utilizados 4182 vetores suportes, correspondendo à totalidade do subconjunto de treinamento.

As performances da LSSVR e das redes neurais com camadas ocultas, foram bastante similares. As tabelas 8 e 15 indicam um valor de $R^2_{ajust.}$ médio para a LSSVR com a função kernel gaussiana superior ao dos experimentos da MLP com uma camada oculta. Retomando a Tabela 9, esta indica que o $R^2_{ajust.}$ máximo da LSSVR com kernel gaussiano foi ligeiramente superior ao da MLP com uma camada oculta, com uma diferença aproximada de 0.5%.

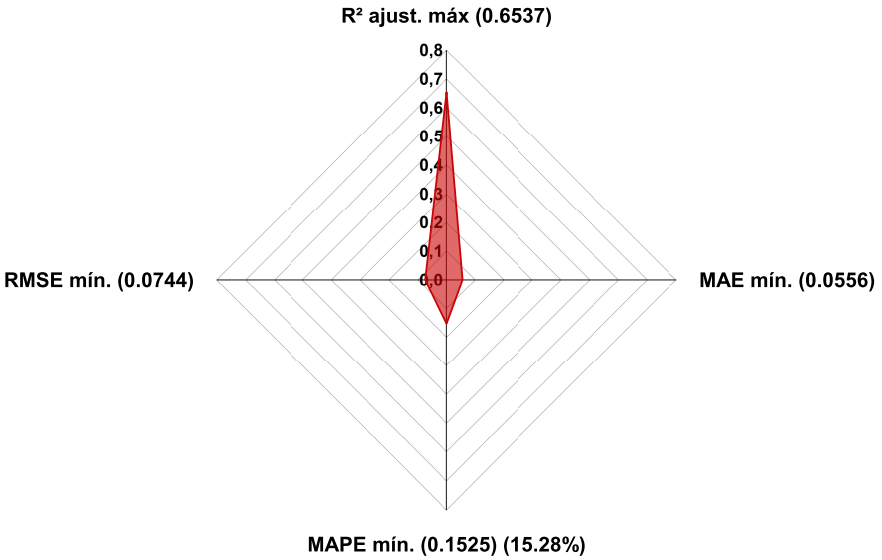
Em relação à MLP com duas camadas ocultas, a LSSVR com função kernel gaussiana atingiu um desempenho muito parecido com o experimento 720 da referida MLP, conforme as tabelas 11 e 15. As figuras 26 e 32 reforçam essa similaridade. Apesar disso, as tabelas 10 e 15 revelam que a LSSVR com função kernel gaussiana obteve um $R^2_{ajust.}$ médio superior aos

Figura 31 – Resultados da LSSVR




Fonte: Autor (2026)

Figura 32 – Gráfico radar da melhor rodada da LSSVR




Fonte: Autor (2026)

experimentos da referida MLP com duas camadas ocultas.

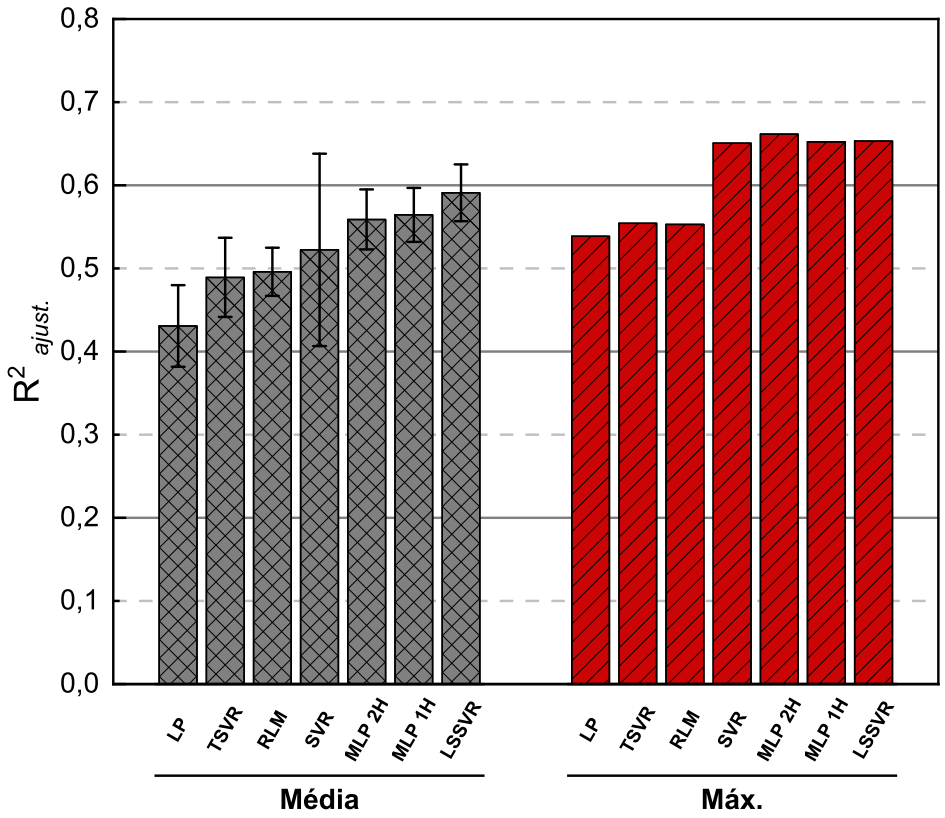
Em relação ao diagrama de dispersão na Figura 31, é notável que a LSSVR também realizou algumas previsões pontuais com erros de previsão consideravelmente maiores. Nesse

contexto, ressalta-se que esse fenômeno foi observado em todos os modelos experimentados no presente trabalho.

## 4.9 Análise de sensibilidade do melhor modelo

A Figura 33 compila o desempenho de cada modelo utilizado no presente trabalho. Foram considerados os experimentos que produziram os maiores $R^2_{ajust.}$ médios e as rodadas com os maiores $R^2_{ajust.}$ máximos de cada modelo. Uma vez que a LSSVR possui o maior $R^2_{ajust.}$ médio e um $R^2_{ajust.}$ máximo muito próximo do maior valor obtido, ela foi eleita como o melhor modelo. Dessa forma, a LSSVR será utilizada para realizar a análise de sensibilidade.

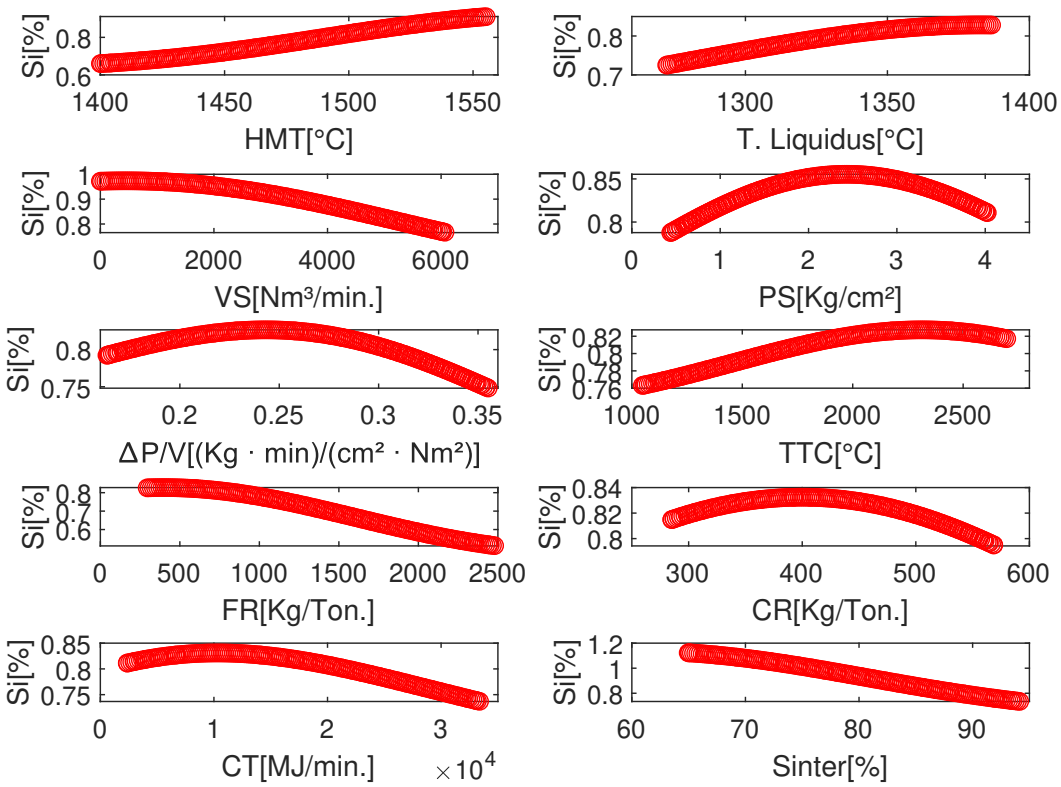
Figura 33 – Performance dos modelos experimentados




Fonte: Autor (2026)

A Figura 34 revela o efeito da variação individual de cada atributo sobre o teor de silício, enquanto os demais atributos foram mantidos em seus respectivos valores médios. A seguir, as relações obtidas no presente trabalho serão comparadas com a literatura sobre o tema.

Figura 34 – Resultado da análise de sensibilidade




Fonte: Autor (2026)

Em seus estudos, Elmomen (2017) e Cardoso *et al.* (2022) afirmam que o teor de silício é diretamente dependente da temperatura do ferro-gusa. Dessa forma, a relação encontrada pela análise de sensibilidade para o teor de silício e a temperatura do ferro-gusa está de acordo com a literatura sobre o tema.

A temperatura *liquidus* da escória precisa ser suficientemente baixa para prevenir a solidificação da escória durante o vazamento, mas não tão baixa a ponto de impedir a separação física entre a escória e o ferro-gusa (Muller e Erwee, 2011). Recentemente, Shiau *et al.* (2018) apresentaram resultados que mostram uma relação direta entre a basicidade e a temperatura *liquidus* da escória. Adicionalmente, Hage *et al.* (2022) afirmam que baixos teores de sílica ($SiO_2$) na escória levam a baixos teores de silício no ferro-gusa. Diante do exposto, um aumento no teor de sílica da escória diminui a basicidade desta e, consequentemente, deveria diminuir a sua temperatura *liquidus*. Paralelamente, também haverá um aumento no teor de silício do ferro-gusa. Isso significa que a temperatura *liquidus* da escória e o teor de silício no ferro-gusa estão inversamente correlacionados. Apesar disso, a análise de sensibilidade indica que a relação

entre ambos é direta.

Essa discrepância entre a análise de sensibilidade e a relação descrita na literatura para as variáveis temperatura *liquidus* da escória e o teor de silício no ferro-gusa pode ser devido à influência de outras variáveis do alto-forno com um impacto mais significativo, pois, conforme Cardoso *et al.* (2022), a quantidade de sílica na escória possui pouca influência na variação do teor de silício do ferro-gusa. Com isso, para trabalhos futuros, seria interessante considerar diretamente a composição química da escória ao invés de sua temperatura *liquidus*.

Para os operadores experientes, é bem sabido que um aumento no volume de sopro acarreta em uma diminuição do teor de silício no ferro-gusa. Essa relação ocorre porque uma diminuição no sopro resulta em uma diminuição do ritmo do alto-forno, levando a um aumento do tempo de contato do ferro-gusa com o gás que contem $SiO$ (Geerdes *et al.*, 2020). Com isso, a relação descrita pela análise de sensibilidade está de acordo com a literatura existente.

Em seus estudos, GUSTAVSSON *et al.* (2005) afirmam que o teor de silício no ferro-gusa em equilíbrio calculado é maior em pressões de sopro mais baixas. Apesar disso, a análise de sensibilidade realizada por David *et al.* (2016) demonstra que a relação entre a pressão de sopro e o teor de silício no ferro-gusa é positiva. Logo, há uma divergência por parte dos pesquisadores sobre o tema. Da mesma forma, a relação encontrada pela análise de sensibilidade no presente trabalho foi inconclusiva e justifica maiores investigações.

A variável $\Delta P/V$ trata-se de uma medida da permeabilidade gasosa no interior do alto-forno. Segundo Pan *et al.* (2019), uma permeabilidade baixa leva a uma menor quantidade de gás redutor atravessando o leito de coque, o que aumenta a dependência da redução do minério de ferro em relação à redução direta e aumenta o consumo de combustível. Além disso, uma permeabilidade baixa aumenta a diferença de pressão no sopro, o que dificulta a realização de uma operação suave do alto-forno. Nesse contexto, não foi possível encontrar uma relação direta entre o teor de silício no ferro-gusa e a permeabilidade descrita na literatura, apesar de esta influenciar o fluxo gasoso do alto-forno. Adicionalmente, a análise de sensibilidade foi inconclusiva.

Segundo David *et al.* (2016), elevar a temperatura teórica de chama acelera a taxa de produção de $SiO_{(g)}$, que é o principal meio gasoso responsável pela incorporação do silício no ferro-gusa. Geerdes *et al.* (2020) também relata esta mesma relação. Dessa forma, a relação encontrada pela análise de sensibilidade no presente trabalho está de acordo com o previsto pela literatura.

No caso do *fuel rate*, Mei *et al.* (2020) e Gasparini *et al.* (2017) destacam que uma redução do teor de silício implica na redução do consumo de combustível, o que torna a sua relação positiva. Logo, a relação encontrada pela análise de sensibilidade vai de encontro ao que é relatado pelos estudos citados anteriormente. A técnica de análise de sensibilidade utilizada pertence à classe de métodos locais, sendo conhecida como o método de um fator por vez. Segundo Tian (2013), essa abordagem explora apenas um espaço reduzido ao redor de um caso base e ignora as interações entre as variáveis. Portanto, e também levando em consideração que o desempenho máximo da LSSVR, medido pelo coeficiente $R^2_{ajust.}$, foi de apenas 0.65, essa pode ser a razão pela qual as relações de alguns atributos com o teor de silício não foram modeladas conforme descrito na literatura existente.

Conforme discutido na seção 2.3, as cinzas do coque carregado no alto-forno constituem a principal fonte de silício nesse reator, logo, é de se esperar que o teor de silício e o *coke rate* tenham uma relação positiva. Apesar disso, Raza *et al.* (2025) encontraram que a relação entre *coke rate* e o teor de silício do ferro-gusa só é positiva quando este é alto. Do contrário, baixos teores de *coke rate* não levaram a um baixo teor de silício. Diante do exposto e visto que a análise de sensibilidade sobre essa relação no presente trabalho foi inconclusiva, novos estudos devem ser realizados para um melhor entendimento dessa correlação.

A carga térmica do alto-forno depende da quantidade de calor em seu interior. Lembrando que o $SiO_{(g)}$ é o principal meio gasoso responsável pelo transporte de silício e que a relação deste com o teor de silício é direta, Mei *et al.* (2020) destaca que a reação predominante para gerar $SiO_{(g)}$ (Equação 2.8) é altamente endotérmica. Logo, a produção de $SiO_{(g)}$ consome calor, diminuindo a carga térmica do alto-forno, o que implica em uma relação inversa, conforme a análise de sensibilidade do presente trabalho.

No caso da quantidade de sinter na carga, Liu *et al.* (2025) afirmam que o teor silício depende da quantidade de sílica no sinter, mas que essa influência depende de múltiplos fatores. Além disso, após discutir profundamente sobre os mecanismos de incorporação de silício no ferro-gusa, Hage *et al.* (2022) afirmam que a sílica do sinter que irá compor a escória não faz parte do processo de gaseificação. Diante do exposto, um acréscimo da quantidade de sinter na carga pode levar à uma redução do teor de silício no ferro-gusa, pois haverá um aumento de uma fonte de silício que não contribui para a formação de $SiO_{(g)}$, o que está em concordância com a relação encontrada na análise de sensibilidade.

# 5 CONCLUSÕES

O presente trabalho buscou realizar a predição do teor de silício utilizando modelos computacionais com o intuito de obter os menores erros de previsão possíveis. Para cumprir esse propósito, foram testados modelos como o *perceptron* logístico, redes neurais artificiais do tipo MLP e três tipos de regressão de vetores suporte. Adicionalmente, também foi investigada uma técnica para reduzir o tempo de busca do número de neurônios ocultos das redes neurais e uma análise de sensibilidade foi executada para estudar o efeito individual de cada variável de entrada no teor de silício no ferro-gusa.

No âmbito dos modelos baseados em neurônios, o *perceptron* logístico foi o modelo com o pior desempenho entre todos os testados, obtendo parâmetros de desempenho inferiores à regressão linear múltipla. A respeito das redes MLP com uma e duas camadas ocultas, as quais ultrapassaram consideravelmente a regressão linear múltipla, a primeira obteve medidas de desempenho similares à segunda, o que indica a necessidade de apenas uma camada oculta para a previsão do teor de silício do ferro-gusa. Adicionalmente, A função sigmoide logística não figurou como a função de ativação que produziu os melhores resultados medidos pelo $R^2_{ajust.}$ médios ou $R^2_{ajust.}$ máximos, tanto para a MLP com uma camada oculta quanto para a MLP com duas camadas ocultas (ver tabelas 8, 9, 10 e 11).

O uso da técnica com SVD para estimar o número de neurônios ocultos provou-se uma ferramenta poderosa. A diferença entre a sugestão da matriz $\mathbf{Y}^{(h)}$ e o experimento da MLP com uma camada oculta com o terceiro maior valor de $R^2_{ajust.}$, o qual possui valor similar ao experimento com o primeiro maior valor, foi de apenas um neurônio. No caso da MLP com duas camadas ocultas, as matrizes $\mathbf{Y}^{(h)}$ e $\mathbf{E}^{(h)}$ sugeriram exatamente o mesmo número de neurônios ocultos que o experimento com o maior $R^2_{ajust.}$ possui. Diante do exposto, recomenda-se o uso de uma janela de testes de até dois ou três neurônios ao redor do valor sugerido.

Levando em consideração os valores $R^2_{ajust.}$ médio, entre os modelos de regressão de vetores suporte, a TSVR obteve o pior desempenho, ligeiramente inferior ao da RLM. A SVR foi inferior às redes com camadas ocultas e a LSSVR foi o modelo mais bem sucedido, obtendo um $R^2_{ajust.}$ médio maior do que à rede neural com uma camada oculta.

Em relação à análise de sensibilidade, as relações encontradas entre o teor de silício e a temperatura do ferro-gusa, a temperatura teórica de chama, o volume de sopro, a carga térmica e a quantidade de sinter na carga condizem com as informações encontradas na literatura. As relações encontradas para o teor de silício do ferro-gusa com a temperatura *liquidus* da escória e o

*fuel rate* divergem da literatura, apesar das justificativas encontradas. Paralelamente, as relações entre o teor de silício e a pressão de sopro e o *coke rate* foram inconclusivas, reforçando as divergências entre pesquisadores. Finalmente, a relação entre o teor de silício e a permeabilidade foi inconclusiva e não pôde se encontrada na literatura, merecendo investigações futuras

Em síntese, os resultados apresentados no presente trabalho mostram que os modelos utilizados são soluções promissoras para a tarefa de previsão do teor de silício no ferro-gusa, mas que ainda precisam ser mais refinados antes da aplicação em ambientes industriais. Para alcançar resultados melhores, sugere-se diferentes abordagens. Inicialmente, é possivel tentar melhorar a qualidade dos dados de entrada dos modelos, buscando períodos de operação sem grandes alterações na operação do alto-forno e novas abordagens para capturar as variáveis de entrada. Além disso, selecionar mais variáveis para compor os modelos preditivos também pode melhorar o sucesso da predição. Finalmente, é interessante testar novos modelos e técnicas de pré-processamento dos dados, tais como a análise de componentes principais (PCA) para descorrelacionar os atributos de entrada antes de alimentar os modelos preditivos ou usar algoritmos de agrupamento para detectar diferentes regimes de operação do alto-forno e criar modelos preditivos para cada um deles.

# REFERÊNCIAS

AWAD, M.; KHANNA, R. **Efficient learning machines**: theories, concepts, and applications for engineers and system designers. Berkeley: Apress, 2015. Disponível em: https://link.springer.com/book/10.1007/978-1-4302-5990-9. Acesso em 14 out. 2025

BABICH, A.; SENK, D. Recent developments in blast furnace iron-making technology. *In*: LU, L. (ed.). **Iron ore**. Sawston: Woodhead Publishing, 2015. p. 505–547. Disponível em: https://www.sciencedirect.com/science/article/pii/B9781782421566000174. Acesso em: 27 set. 2025.

BABICH, A.; SENK, D.; GUDENAU, H. W.; MAVROMMATIS, K. **Ironmaking**: textbook. Aachen: Mainz, 2008.

BARELLA, S.; CECCA, C. D.; MAPELLI, C.; GRUTTADAURIA, A.; BONDI, E.; MARINARI, A. Study of a new operating practice for refining high silicon hot metal in a bof converter. **steel research international**, Weinheim, v. 87, n. 7, p. 941–946, 2016. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/srin.201500283. Acesso em: 5 out. 2025.

BASAK, D.; PAL, S.; PATRANABIS, D. Support vector regression. **Neural Information Processing – Letters and Reviews**, New York v. 11, n. 10, p. 203-224, 2007. Disponível em: https://www.researchgate.net/publication/228537532_Support_Vector_Regression. Acesso em: 14 dez. 2025.

BERMEITINGER, B.; HRYCEJ, T.; HANDSCHUH, S. Singular value decomposition and neural networks. *In*: INTERNATIONAL CONFERENCE ON ARTIFICIAL NEURAL NETWORKS, 28., 2018, Munich. Cham: Springer International Publishing, 2019. p. 153–164. Disponível em: https://www.researchgate.net/publication/334081882_Singular_Value_Decomposition_and_Neural_Networks. Acesso em 12 out. 2025.

BISHOP, C. M.; BISHOP, H. **Deep learning**: foundations and concepts. Cham: Springer, 2024.

BRAGA, F. D.; MOURA, E. P. Mathematical modelling of sinter quality and analysis of variables in a sinter plant. **Ironmaking & Steelmaking.** Londres, v. 0, n. 0, p. 03019233251386906, 2025. Disponível em: https://doi.org/10.1177/03019233251386906. Acesso em: 04 set. 2025.

CAMERON, I.; SUKHRAM, M.; LEFEBVRE, K.; DAVENPORT, W. **Blast furnace ironmaking**: analysis, control, and optimization. Kidlington: Elsevier, 2020. p. 19–30. Disponível em: https://www.sciencedirect.com/science/article/pii/B9780128142271000026. Acesso em: 16 nov. 2025.

CARDOSO, W.; FELICE, R. di; BAPTISTA, R. C. Artificial neural network for predicting silicon content in the hot metal produced in a blast furnace fueled by metallurgical coke. **Materials Research**, São Carlos, v. 25, p. e20210439, 2022. Disponível em: https://doi.org/10.1590/1980-5373-MR-2021-0439. Acesso em: 22 dez. 2025.

CHATTERJEE, R.; KRISHNA, S.; VARUNKUMAR, S.; NAG, S. The impact of coal ash fusion on blast furnace decarbonization: a study without and with natural gas co-injection. **Fuel.** Kidlington, v. 406, p. 136886, 2026. Disponível em: https://www.sciencedirect.com/science/article/pii/S0016236125026110. Acesso em: 07 dez. 2025.

DAOUD, J. I. Multicollinearity and regression analysis. **Journal of Physics**, Bristol, v. 949, n. 1, p. 012009, dec 2017. Disponível em: https://doi.org/10.1088/1742-6596/949/1/012009. Acesso em: 01 out. 2025.

DARLINGTON, R.; HAYES, A. **Regression analysis and linear models**: concepts, applications, and implementation. New York: Guilford Publications, 2016.

DAVID, S. F.; DAVID, F. F.; MACHADO, M. Artificial neural network model for predict of silicon content in hot metal blast furnace. **Materials Science Forum**, Cuiabá, v. 869, p. 572–577, 2014. Trabalho apresentado no Brazilian Conference on Materials Science and Engineering, 21., 2016. Disponível em: https://www.scientific.net/MSF.869.572. Acesso em: 16 nov. 2025.

DONAYO, R.; DATA, A.; GÓMEZ, A.; BALANTE, W.; PÉREZ, J. Refining high-silicon hot metal in an oxygen converter. new process to decrease slopping and fume emissions. **Rev. Metall.** Les Ulis, v. 107, n. 7, p. 319–328, 2010. Disponível em: https://doi.org/10.1051/metal/2010112. Acesso em: 12 dez. 2025.

DRUCKER, H.; BURGES, C. J.; KAUFMAN, L.; SMOLA, A. J.; VAPNIK, V. Support vector regression machines. *In*: MIT PRESS. **Advances in neural information processing systems**. [S.l.], 1997. v. 9, p. 155–161. Disponível em: https://www.researchgate.net/publication/309185766_Support_vector_regression_machines. Acesso em: 20 out. 2025.

ELMOMEN, S. S. A. Influence of slag composition and temperature on silicon distribution between slag and hot metal in the egyptian blast furnace no.iii. **Journal of Petroleum and Mining Engineering**, v. 19, n. 1, p. 26–32, 2017. Disponível em: https://www.researchgate.net/publication/334297481_Influence_of_Slag_Composition_and_Temperature_on_Silicon_Distribution_between_Slag_and_Hot_Metal_in_the_Egyptian_Blast_Furnace_NoIII. Acesso em: 11 out. 2025.

GALVãO, R.; ARAúJO, M. Variable selection. *In*: BROWN, S. D.; TAU-LER, R.; WALCZAK, B. (ed.). **Comprehensive Chemometrics**. Oxford: Elsevier, 2009. p. 233–283. Disponível em: https://www.sciencedirect.com/science/article/pii/B9780444527011000752. Acesso em: 05 nov. 2025.

GASPARINI, V. M.; CASTRO, L. F. A. d.; MOREIRA, V. E. d. S.; QUINTAS, A. C. B.; VIANA, A. O.; ANDRADE, D. H. B. Impact of operational parameters on fuel consumption of a blast furnace. **REM, International Engineering Journal (Revista Escola de Minas)**, Ouro Preto, v. 70, n. 4, p. 465–470, 2017. Disponível em: https://www.scielo.br/j/remi/a/xYbCpNhxghhXkpvTRnSPZRG/?lang=en. Acesso em: 5 set. 2025.

GEERDES, M.; CHAIGNEAU, R.; LINGIARDI, O. **Modern blast furnace ironmaking:** an introduction. 4 ed. Amsterdam: IOS Press, 2020.

GHOSH, A.; CHATTERJEE, A. **Ironmaking and steelmaking**: theory and practice. New Delhi: PHI Learning Pvt. Ltd., 2008.

GHOSH, S.; VISWANATHAN, N. N.; BALLAL, N. B. Flow phenomena in the dripping zone of blast furnace a review. **steel research international.** Weinheim, v. 88, n. 9, p. 1600440, 2017. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/srin.201600440. Acesso em: 13 set. 2025.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. **Deep learning**. MIT Press, 2016. Disponível em: http://www.deeplearningbook.org. Acesso em: 22 set. 2025.

GU, B.; FANG, J.; PAN, F.; BAI, Z. Fast clustering-based weighted twin support vector regression. **Soft Computing.** Heidelberg, v. 24, n. 8, p. 6101–6117, 2020. Disponível em: https://doi.org/10.1007/s00500-020-04746-6. Acesso em: 20 nov. 2025.

GUSTAVSSON, J.; ANDERSSON, A. M. T.; JOUML; ENSSON, P. G. A thermodynamic study of silicon containing gas around a blast furnace raceway. **ISIJ International**. Tóquio, v. 45, n. 5, p. 662–668, 2005. Disponível em: https://www.jstage.jst.go.jp/article/isijinternational/45/5/45_5_662/_article. Acesso em: 10 out. 2025.

HAGE, J. L. T.; STEL, J. van der; YANG, Y. Silicon in hot metal from a blast furnace, the role of feo. **Ironmaking & Steelmaking.** Londres, v. 49, n. 6, p. 581–587, 2022. Disponível em: https://journals.sagepub.com/doi/abs/10.1080/03019233.2022. Acesso em: 17 nov. 2025.

HAYKIN, S. **Neural networks:** a comprehensive foundation. Upper Saddle River: Prentice Hall, 1999.

HAYKIN, S. **Neural networks and learning machines**. Hamilton: Prentice Hall, 2009.

HUANG, H.; WEI, X.; ZHOU, Y. An overview on twin support vector regression. **Neurocomputing.** Amsterdam, v. 490, p. 80–92, 2022. Disponível em: https://www.sciencedirect.com/science/article/pii/S0925231222003101. Acesso em: 26 nov. 2025.

KARAL, O. Comparative performance analysis of epsilon-insensitive and pruning based algorithms for sparse least squares support vector regression. **Sigma Journal of Engineering and Natural Sciences.** Istanbul, v. 42, p. 578–589, 2024. Disponível em: https://www.semanticscholar.org/paper/Comparative-performance-analysis-of-and-algorithms-Karal/e1e713cdcbaba45ca3a1208b4655779865eb789b. Acesso em: 28 nov. 2025.

LAKSHMANAN, K.; KARIMI, A.; CARR, A.; WAUTERS, P.; AUINGER, M.; PLEYDELL-PEARCE, C.; GIANNETTI, C. A hybrid modelling approach based on deep learning for the prediction of the silicon content in the blast furnace. **Procedia Computer Science**, Atenas, v. 225, p. 2204–2213, 2023. Trabalho apresentado no International Conference on

Knowledge Based and Intelligent Information and Engineering Sytems, 27., 2023, Atenas. Disponível em: https://www.sciencedirect.com/science/article/pii/S1877050923013698. Acesso em: 28 nov. 2025.

LIU, Q.; CHEN, Q.; ZHANG, N.; YAN, R.; CHU, J.; SUN, H.; DAI, B. The method of reducing energy consumption in large blast furnace smelting by low-silicon smelting. **Ironmaking & Steelmaking.** Londres, v. 0, n. 0, p. 03019233241266129, 2025. Disponível em: https://journals.sagepub.com/doi/10.1177/03019233241266129. Acesso em: 24 out. 2025.

MAHIMA, R.; MAHESWARI, M.; ROSHANA, S.; PRIYANKA, E.; MOHANAN, N.; NANDHINI, N. A comparative analysis of the most commonly used activation functions in deep neural network. *In*: INTERNATIONAL CONFERENCE ON ELECTRONICS AND SUSTAINABLE COMMUNICATION SYSTEMS (ICESC),4., Coimbatore. **Proceedings** [...]. Coimbatore: IEEE,2023. p. 1334–1339. Disponível em: https://ieeexplore.ieee.org/document/10193390. Acesso em: 30 nov. 2025.

MEI, Y.; CHENG, S.; NIU, Q.; XU, W.; GE, J. A review on transfer mechanism and influence factors of silicon in blast furnace. **Ironmaking & Steelmaking.** Londres, v. 47, n. 3, p. 246–262, 2020. Disponível em: https://journals.sagepub.com/doi/abs/10.1080/03019233.2019.1690836. Acesso em: 02 nov. 2025.

MONTGOMERY, D**. Applied statistics and probability for engineers**. 6 ed. Hoboken: John Wiley & Sons, 2013.

MONTGOMERY, D.; PECK, E.; VINING, G.; SAFARI, A. O. M. C. **Introduction to linear regression analysis**. 5 ed. Hoboken: John Wiley & Sons, 2012.

MULLER, J.; ERWEE, M. Blast furnace control using slag viscosities and liquidus temperatures with phase equilibria calculations. *In*: JONES, R.; HOED, P. den (ed.). **Southern African Pyrometallurgy 2011**. Johannesburg, South Africa, 2011. p. 137–152. Acesso em: 21 out. 2025.

MURTA, R. H. F.; BRAGA, F. D.; MAIA, P. P. N.; DIÓGENES, O. B. F.; MOURA, E. P. de. Mathematical modelling for predicting mechanical properties in rebar manufacturing. **Ironmaking & Steelmaking**, Londres, v. 48, n. 2, p. 161–169, 2021. Disponível em: https://doi.org/10.1080/03019233.2020.1749357. Acesso em: 08 out. 2025.

NASER, M. Z.; ALAVI, A. H. Error metrics and performance fitness indicators for artificial intelligence and machine learning in engineering and sciences. **Architecture, Structures and Construction.** Cham, v. 3, n. 4, p. 499–517, 2023. Disponível em: https://doi.org/10.1007/s44150-021-00015-8. Acesso em: 01 dez. 2025.

PAN, Y. zhu; ZUO, H. bin; WANG, J. song; XUE, Q. guo; WANG, G.; SHE, X. feng. Review on improving gas permeability of blast furnace. **Journal of Iron Steel Research International.** Beijing, v. 27, p. 121–131, 2019. Disponível em: https://link.springer.com/article/10.1007/s42243-019-00321-y. Acesso em: 11 out. 2025.

PENG, X. Tsvr: An efficient twin support vector machine for regression. **Neural**

**Networks.** Kidlington, v. 23, n. 3, p. 365–372, 2010. Disponível em: https://www.sciencedirect.com/science/article/pii/S0893608009001567. Acesso em: 15 set. 2025.

RAZA, O.; WALLA, N.; OKOSUN, T.; LEONTARAS, K.; ENTWISTLE, J.; ZHOU, C. Prediction of silicon content in a blast furnace via machine learning: a comprehensive processing and modeling pipeline. **Materials.** Basel, v. 18, n. 3, 2025. Disponível em: https://www.mdpi.com/1996-1944/18/3/632. Acesso em: 10 dez. 2025.

RIVAS-PEREA, P.; COTA-RUIZ, J.; CHAPARRO, D. G.; VENZOR, J. A. P.; CARREÓN, A. Q.; ROSILES, J. G. Support vector machines for regression: a succinct review of large-scale and linear programming formulations. **International Journal of Intelligence Science.** Irvine, v. 3, n. 1, p. 5–14, 2013. Disponível em: https://www.scirp.org/journal/paperinformation?paperid=27408. Acesso em: 19 out. 2025.

SANTOS, J. D. A.; BARRETO, G. A.; MEDEIROS, C. M. S. Estimating the number of hidden neurons of the mlp using singular value decomposition and principal components analysis: a novel approach. *In*: BRAZILIAN SYMPOSIUM ON NEURAL NETWORKS, 11., 2010, São Paulo. **Proceedings** […]. São Paulo: IEEE, 2010. p. 19–24. Disponível em: https://ieeexplore.ieee.org/document/5715207. Acesso em: 21 nov. 2025.

SENAVIRATNA, N. A. M. R.; COORAY, T. M. J. A. Diagnosing multicollinearity of logistic regression model. **Asian Journal of Probability and Statistics**, Hooghly, v. 5, n. 2, p. 1–9, 2019. Disponível em: https://journalajpas.com/index.php/AJPAS/article/view/96. Acesso em: 25 nov. 2025.

SENESI, G. S.; PASCALE, O. D.; BOVE, A.; MARANGONI, B. S. Quantitative analysis of pig iron from steel industry by handheld laser-induced breakdown spectroscopy and partial least square (pls) algorithm. **Applied Sciences**, Basel, v. 10, n. 23, 2020. Disponível em: https://www.mdpi.com/2076-3417/10/23/8461. Acesso em: 10 out. 2025.

SHEELA, K. G.; DEEPA, S. N. Review on methods to fix number of hidden neurons in neural networks. **Mathematical Problems in Engineering**, Cairo, v. 2013, n. 1, p. 425740, 2013. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1155/2013/425740. Acesso em: 29 aug. 2025.

SHIAU, J.-S.; LIU, S.-H.; HO, C.-K. Development of slag flowability prediction formula for blast furnace operation and its application. **ISIJ International**, Tóquio, v. 58, n. 1, p. 52–59, 2018. Disponível em: https://www.jstage.jst.go.jp/article/isijinternational/58/1/58_ISIJINT-2017-394/_article/-char/ja. Acesso em: 16 out. 2025.

SONG, J.; XING, X.; PANG, Z.; LV, M. Prediction of silicon content in the hot metal of a blast furnace based on fpa-bp model. **Metals**, Basel, v. 13, n. 5, 2023. Disponível em: https://www.mdpi.com/2075-4701/13/5/918. Acesso em: 10 out. 2025.

STRANG, G. **Linear algebra and learning from data**. Wellesley: Cambridge Press, 2019.

SUOPAJäRVI, H.; PONGRáCZ, E.; FABRITIUS, T. The potential of using biomass-based

reducing agents in the blast furnace: a review of thermochemical conversion technologies and assessments related to sustainability. **Renewable and Sustainable Energy Reviews**, Kidlington, v. 25, p. 511–528, 2013. Disponível em: https://www.sciencedirect.com/science/article/pii/S1364032113002918. Acesso em: 22 set. 2025.

SYSOEV, A. Sensitivity analysis of mathematical models. **Computation**, Basel, v. 11, n. 8, 2023. Disponível em: https://www.mdpi.com/2079-3197/11/8/159. Acesso em: 19 nov. 2025.

TIAN, W. A review of sensitivity analysis methods in building energy analysis. **Renewable and Sustainable Energy Reviews**, Kidlington, v. 20, p. 411–419, 2013. Disponível em: https://www.sciencedirect.com/science/article/pii/S1364032112007101. Acesso em: 19 nov. 2025.

TSAGRIS, M.; PANDIS, N. Multicollinearity. **American Journal of Orthodontics and Dentofacial Orthopedics**, Saint Louis, v. 159, n. 5, p. 695–696, 2021. Disponível em: https://doi.org/10.1016/j.ajodo.2021.02.005. Acesso em: 08 nov. 2025.

WANG, R.-f.; ZHANG, B.; LIU, C.-j.; JIANG, M.-f. Review on monitoring and prevention technologies of splashing induced by inappropriate slag foaming in BOF. **Journal of Iron and Steel Research International**, Beijing, v. 30, n. 9, p. 1661–1674, 2023. Disponível em: https://doi.org/10.1007/s42243-023-00954-0. Acesso em: 4 out. 2025.

WANG, S.; JIANG, Y.; GUO, Y.; YANG, Z.; CHEN, F.; YANG, L.; LI, G. Effects of basicity and al2o3 content on viscosity and crystallization behavior of super-high-alumina slag. **Crystals**, Basel, v. 12, n. 6, 2022. Disponível em: https://www.mdpi.com/2073-4352/12/6/851. Acesso em: 12 dez. 2025.

WU, S.-L.; ZHANG, L.-H.; WU, J.; XU, J.; KOU, M.-Y.; SHEN, W. Real-time estimate of blast furnace theoretical combustion temperature based on the variation of gas utilization rate and coke ratio. *In*: INTERNATIONAL CONGRESS ON THE SCIENCE AND TECHNOLOGY OF IRONMAKING, 6., 2012, Rio de Janeiro. **Proceedings** […]. Rio de Janeiro: Blucher, 2012, p. 366–372. Disponível em: https://abmproceedings.com.br/en/article/real-time-estimate-of-blast-furnacetheoretical-combustion-temperature-based-onthe-variation-of-gas-utilization-rate-and-cokeratio-3. Acesso em: 03 dez. 2025

ZHANG, F.; O'DONNELL, L. J. Support vector regression. *In*: MECHELLI, A.; VIEIRA, S. (ed.). **Machine Learning**. Academic Press, 2020. p. 123–140. Disponível em: https://www.sciencedirect.com/science/article/pii/B9780128157398000079. Acesso em: 12 nov. 2025

ZHANG, Y.; FAN, X.; ZHANG, J.; LIU, Y.; YU, Y.; YU, C.; LIU, L.; WANG, Y.; SI, L.; XIA, L. Energy-saving synergy in blast furnace ironmaking: multi-region regulation of silicon migration for fuel consumption and co2 emissions reduction. **Fuel**, Amsterdam, v. 404, p. 136321, 2026. Disponível em: https://www.sciencedirect.com/science/article/pii/S0016236125020460. Acesso em: 4 out. 2025.

ZONG, Y.; GUO, Z.; ZHANG, J.; LIU, Y.; MENG, S.; NING, X.; JIAO, K. Deadman behavior and slag–iron–coke interaction of low carbon and safety blast furnace: A review. **steel research international**, Weinheim, v. 95, n. 10, p. 2400366, 2024. Disponível em: https://onlinelibrary.wiley.com/doi/abs/10.1002/srin.202400366. Acesso em: 21 nov. 2025.