MINISTÉRIO DA EDUCAÇÃO
Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Especialização em Ciência de Dados

Predição do teor de hidrogênio nos refinos secundários de uma aciaria usando redes neurais artificiais

Gilberto Henrique dos Reis Marçal
Glaucia Amanda Perucci
Leonardo Henrique Costa

João Monlevade, MG
2023

Gilberto Henrique dos Reis Marçal
Glaucia Amanda Perucci
Leonardo Henrique Costa

Predição do teor de hidrogênio nos refinos secundários de uma aciaria usando redes neurais artificiais

Trabalho de conclusão de curso apresentado ao curso de Ciência de Dados do Instituto de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto, como parte dos requisitos necessários para a obtenção do título de Especialista em Ciência de Dados.

Orientador: Prof. Dr. George Henrique Godim da Fonseca

João Monlevade, MG

2023

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

P471p Perucci, Glaucia Amanda.
Predição do teor de hidrogênio nos refinos secundários de uma aciaria usando redes neurais artificiais. [manuscrito] / Glaucia Amanda Perucci. Gilberto Henrique dos Reis Marçal. Leonardo Henrique Costa. - 2023.
61 f.: il.: color., gráf., tab.. + Código.

Orientador: Prof. Dr. George Henrique Godim da Fonseca.
Produção Científica (Especialização). Universidade Federal de Ouro Preto. Departamento de Engenharia de Produção.

1. Aço - Teor de hidrogênio. 2. Previsão. 3. Redes neurais (Computação). 4. Usinas siderúrgicas. I. Costa, Leonardo Henrique. II. Marçal, Gilberto Henrique dos Reis. III. Fonseca, George Henrique Godim da. IV. Universidade Federal de Ouro Preto. V. Título.

CDU 519.2:004.8

Bibliotecário(a) Responsável: Flavia Reis - CRB6-2431

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE COMPUTACAO E SISTEMAS

FOLHA DE APROVAÇÃO

Gilberto Henrique dos Reis Marçal
Glaucia Amanda Perucci
Leonardo Henrique Costa

Predição do teor de hidrogênio nos refinos secundários de uma aciaria através de redes neurais artificiais

Monografia apresentada ao Curso de Especialização em Ciência dos Dados da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Especialista em Ciência dos Dados

Aprovada em 31 de março de 2023

Membros da banca

Dr. George Henrique Godim da Fonseca - Orientador(a) (Universidade Federal de Ouro Preto)
Dra. Sarah Negreiros de Carvalho Leite - Universidade Federal de Ouro Preto
Me. Marlon José dos Anjos Silva - Usiminas

George Henrique Godim da Fonseca, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 13/06/2023

Documento assinado eletronicamente por **George Henrique Godim da Fonseca**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 13/06/2023, às 13:49, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0540176** e o código CRC **C128DF75**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: (31)3808-0819 - www.ufop.br

Agradecimentos

A Deus, por mais esta conquista.

Às nossas famílias, pelo incentivo nessa caminhada.

À Usiminas e à UFOP pela organização do curso e pela oportunidade que nos foi dada de cursá-lo.

Ao professor George Fonseca pelo apoio e ajuda na orientação desse trabalho.

À equipe da Aciaria pela ajuda, críticas e sugestões apresentadas.

A todos que nos ajudaram ao logo do curso: professores, colegas de curso e colegas de trabalho.

Resumo

Dentre os problemas críticos na produção do aço está a incorporação de hidrogênio durante o processo produtivo. A presença desse elemento pode ocasionar defeitos, fragilização e, em casos extremos, rompimento da pele de aço solidificada. A redução do teor de hidrogênio no aço líquido é feita apenas pelo equipamento RH, cujo custo e tempo de tratamento não torna viável o processamento de todo material produzido. Portanto, o objetivo deste trabalho é predizer o teor de hidrogênio das corridas antes do processo de lingotamento com o uso de redes neurais artificiais do tipo *Multi-Layer Perceptron* (MLP), baseado nos dados dos processos anteriores. Foram utilizados dados históricos de produção da aciaria da Usiminas, planta de Ipatinga/MG. Para atingir o objetivo, foi estruturada uma rede neural artificial cujos valores preditos foram comparados com os valores reais medidos no processo através do sistema Hydris. O modelo gerado respondeu satisfatoriamente, com um erro absoluto médio de $0,76$ ppm, dentro de faixa de tolerância de + ou - 2 ppm em relação ao valor aferido pelo sistema Hydris, atendendo às expectativas do público técnico e operacional da aciaria.

Palavras-chaves: Hidrogênio no aço. Aciaria. Redes neurais artificiais. Predição.

Abstract

One of the several problems faced by steel production is the incorporation of hydrogen during the production process. The presence of this element can cause defects, embrittlement and, in extreme cases, rupture of the solidified steel skin. The reduction of hydrogen content in liquid steel is done only by RH equipment, whose cost and processing time do not make it feasible to process all the material produced. Therefore, the objective of this work is to predict the hydrogen content of the heat before the continuous casting process using artificial neural networks Multi-Layer Perceptron (MLP), based on data from previous processes. Historical data from the production process of the Usiminas melt shop, plant in Ipatinga/MG, were used. To achieve the objective, an artificial neural network was structured whose predicted values were compared with the actual values measured in the process through the Hydris system. The created network responded satisfactorily, with an average absolute error of $0,76$ points, within a tolerance range of + or - 2 ppm in relation to the value measured by the Hydris system, meeting the expectations of the technical and operational public of the melt shop.

Keywords: Hydrogen in steel. Steel works. Artificial neural networks. Prediction.

Lista de ilustrações

Lista de tabelas

Sumário

1 Introdução

O aço é um produto usado em diversos setores da indústria. Como exemplos é possível citar o setor de transportes (fabricação de rodas, motores e lataria na indústria automotiva, trens, navios, bicicletas) utilidades domésticas (geladeiras, fogões, máquinas de lavar), ferramentas, construção civil, embalagens, entre outros.

O processo de produção do aço passa por várias etapas, desde o tratamento da matéria prima, transformando-a em gusa através dos altos fornos e, posteriormente, a transformação em aço nas linhas de aciaria. Durante a produção do aço podem ocorrer problemas que ocasionam o aumento de custo ou na diminuição da qualidade do produto. Dentre os problemas mais críticos, está a incorporação do hidrogênio no aço, principalmente em sua forma líquida (FUJII et al., 2003). A presença desse elemento no aço é considerada sempre prejudicial e é causa comum de defeitos, tais como trincas, pontos concentrados de gás e fragilização do aço (HURST e VERGAUWENS *apud* (ZORZATO, 2013)). Quando o nível de hidrogênio no aço está elevado durante o processo de lingotamento contínuo, em casos extremos, pode ocorrer o rompimento da pele de aço solidificada, também conhecida como rompimento de veio ou *break out*.

Dentre os gases existentes na natureza, o hidrogênio é o elemento químico mais leve e de menor massa atômica entre todos os elementos conhecidos, possuindo grande afinidade química com diversos compostos. Conforme citado, o hidrogênio também é reconhecido como sendo um elemento residual prejudicial ao aço, interferindo nas suas propriedades mecânicas do aço. O controle do teor desse gás no aço é fundamental para que o produto alcance o nível de qualidade desejado.

É importante reconhecer os pontos de incorporação do hidrogênio no aço e ter o controle das variáveis de processo para ajudar a minimizar os efeitos indesejáveis desse gás no produto nas etapas posteriores à de fabricação.

A redução do teor de hidrogênio nas aciarias da Usiminas é feita apenas pelo equipamento conhecido como desgaseificador a vácuo RH, cujo processo tem alto custo, representando aproximadamente 14% dos custos de transformação da tonelada de placa lingotada, conforme orçamento de custos da empresa. Outro fator é o alto tempo demandado por esse equipamento, que pode dobrar o tempo de processamento da corrida, tornando inviável que todas as corridas passem pelo mesmo.

Atualmente, existe um sistema denominado Hydris, desenvolvido pela empresa Electro-Nite, que é reconhecido como padrão mundial para análises de hidrogênio contido no aço líquido, com uma medição rápida e direta (HENRIQUES, 2010). A medição do hidrogênio no aço líquido na Usiminas segue critérios definidos nos padrões operacionais e não é feita em todas as corridas. Quando atende aos critérios, a medição é feita no momento de lingotamento, no distribuidor de aço líquido para o equipamento. Por esse motivo não se conhece o teor de hidrogênio de todas as corridas antes do início do processo de lingotamento, não sendo possível ajustar previamente os parâmetros operacionais. Conforme informado anteriormente, parâmetros operacionais não ajustados para o nível correto de hidrogênio aumentam o risco de rompimento de pele (*break out*).

Nesse contexto, tão importante quanto conhecer os pontos de incorporação do hidrogênio, é conhecer o valor do hidrogênio nas corridas em todas as etapas de fabricação. Com a identificação do teor de hidrogênio das corridas é possível alterar parâmetros de produção no processo de fabricação das placas de aço, bem como redirecionar o material para outros clientes ou redirecionar a corrida para o processo do desgaseificador a vácuo RH quando necessário.

1.1 Objetivo geral

O objetivo geral deste trabalho é predizer o teor de hidrogênio das corridas antes do processo de lingotamento com o uso de redes neurais artificiais *Multi-Layer Perceptron* (MLP), com base nos dados dos processos anteriores.

1.1.1 Objetivos específicos

Como objetivos específicos do presente trabalho pode-se destacar:

- identificar as principais fontes de incorporação de hidrogênio no aço;
- desenvolver uma rede neural treinada para identificar o teor de hidrogênio nas corridas da aciaria antes do processo de lingotamento;
- amparar o operador no processo de decisão sobre o direcionamento (ou não) das corridas para o desgaseificador a vácuo RH;
- promover maior segurança ao processo produtivo.

1.2 Organização do trabalho

Esse trabalho está organizado da seguinte maneira: no Capítulo 2 é apresentada a revisão bibliográfica com os principais conceitos aplicados e os trabalhos relacionados; o Capítulo 3 apresenta a metodologia utilizada; o Capítulo 4 apresenta a rede neural criada e seus resultados. Por fim, o Capítulo 5 apresenta as conclusões e trabalhos futuros.

2 Conceitos fundamentais

Nas seções a seguir são apresentados maiores detalhes sobre o processo de produção de aço e os principais conceitos e técnicas aplicados no presente trabalho.

2.1 Processos da aciaria

O processo de aciaria em uma siderúrgica consiste em transformar o gusa líquido, que é um material obtido através do processo de transformação do minério de ferro e carvão mineral em aço de diversos tipos.

Existem diversas fontes de incorporação de hidrogênio no aço líquido. Conforme (HENRIQUES, 2010), a água é a principal delas, seja na forma de vapor presente no ar, na forma de impurezas nos ferro-ligas e recarburantes ou associada com materiais formadores de escória à base de cal. Ainda de acordo com Henriques (2010), algumas práticas e condições operacionais contribuem para a incorporação de hidrogênio no aço. A partir do processo dos convertedores, em todas as etapas em que ocorre inclusão de materiais é possível ocorrer a inclusão de hidrogênio no aço líquido.

Nesse trabalho são descritos os processos da aciaria que processam o aço líquido e tem relação direta com a inclusão ou redução do hidrogênio, que são o refino primário (carro torpedo, panela) e o refino secundário (Forno Panela, CASOB, Desgaseificação a Vácuo). Também é descrito o lingotamento contínuo, que é a etapa em que o aço líquido é solidificado e transformado em placa. A Figura 1 apresenta um fluxo básico do processo da aciaria.

2.1.1 Refino primário

O gusa líquido produzido nos altos fornos apresenta em sua composição química teores de enxofre em patamar elevado para atender a maioria das especificações dos produtos fornecidos pela Usiminas. Os processos de refino consistem na adequação da composição química para valores compatíveis com as propriedades requeridas. É a transformação do gusa em aço. O refino primário (pré-tratamento de gusa, sopro de oxigênio no convertedor) tem como objetivo básico o ajuste dos teores de carbono, enxofre e fósforo.

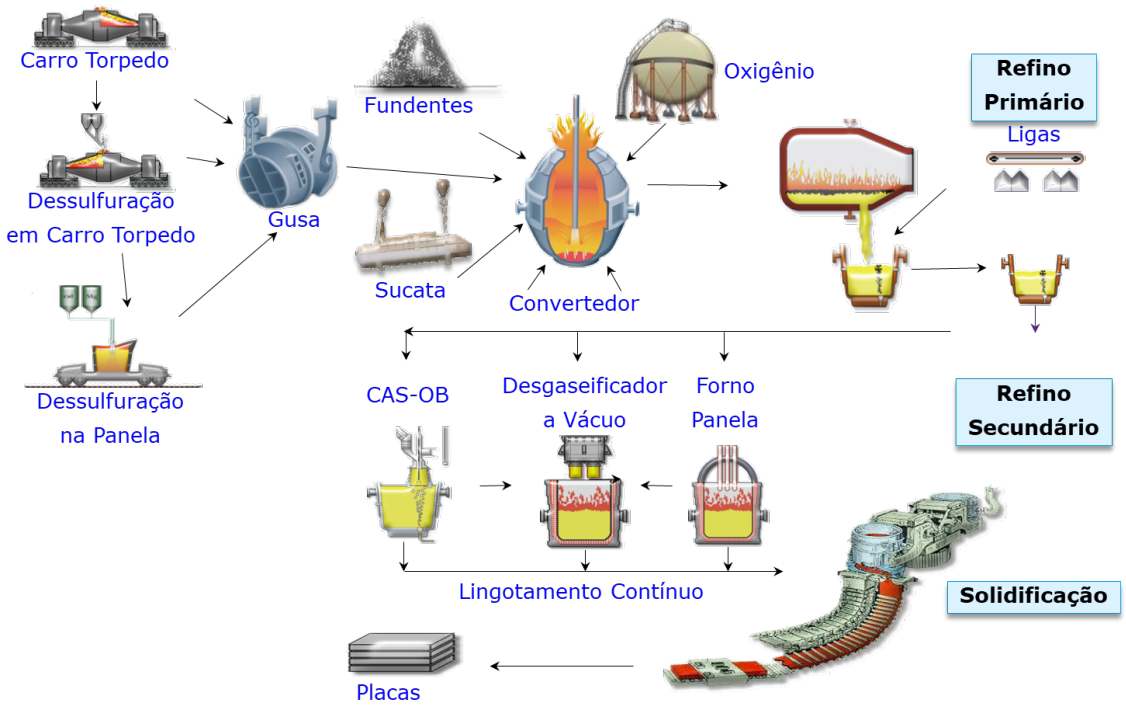
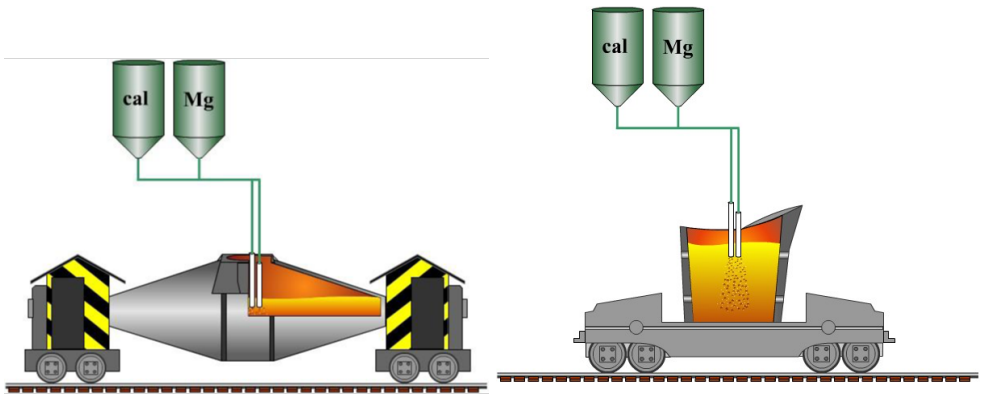


Figura 1 – Resumo do fluxo de produção da aciaria.

Fonte: Usiminas (material de treinamento).

2.1.1.1 Pré-tratamento de gusa

Na Usiminas, praticam-se dois processos de pré-tratamento de gusa, conforme mostra a Figura 2. O primeiro, dessulfuração em carro-torpedo, consiste na injeção de finos de óxido de cálcio e alumínio em pó no banho metálico, através de uma lança refratária usando nitrogênio com o gás de arraste. O segundo, dessulfuração em panela de transferência (panela pelicano), consiste de um sistema de co-injeção de finos de óxido de cálcio e magnésio metálico. Ambos processos permitem o tratamento de 100% do gusa e obtenção de teor de enxofre menor ou igual a 0,002%.



(a) Dessulfuração em carro torpedo. (b) Dessulfuração em panela.

Figura 2 – Processos de pré-tratamento de gusa

Fonte: Usiminas (material de treinamento).

2.1.1.2 Sopragem

O processo de sopro de oxigênio caracteriza-se pelas reações de oxidação parcial do carbono, manganês, silício, fósforo e outros elementos contidos no gusa líquido. Permite também a redução do teor de fósforo, estabilizado em uma escória básica, formada durante o refino. Após o carregamento, o convertedor é basculado para a posição vertical, e uma lança refrigerada a água é inserida no seu interior, a uma altura preestabelecida, soprando oxigênio sob pressão na superfície do banho, conforme mostra a Figura 3.

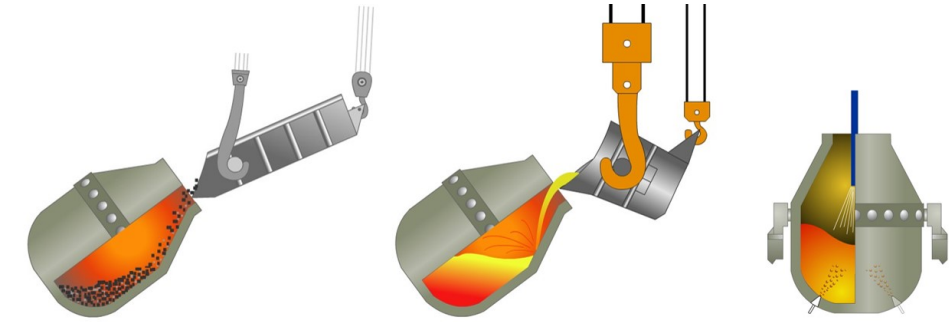
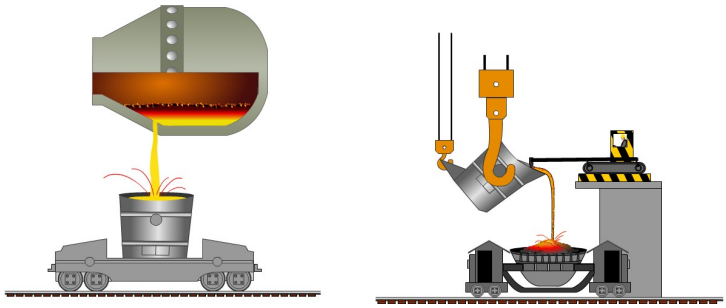


Figura 3 – Processos do refino primário da aciaria da Usiminas.

Fonte: Usiminas (material de treinamento).

Durante o sopro são adicionados fundentes que formam, juntamente com os óxidos obtidos a partir de reações do silício, manganês e ferro, uma escória que tem por finalidade fixar as substancias indesejáveis. O volume de oxigênio soprado é definido em função das matérias primas utilizadas, além do carbono e temperatura previstos no fim de sopro. No final do sopro, mede-se a temperatura do banho, retiram-se amostras de aço e escória. Então, o convertedor é basculado e o aço líquido é vazado para uma panela, onde são adicionados os ferro-ligas (desoxidação), que conferem ao aço as propriedades mecânicas especificadas para o produto. A escória remanescente no convertedor é vazada em um pote de escória e transportada para uma usina de beneficiamento. Os processos de vazamento do aço e escória são ilustrados na Figura 4.



(a) Vazamento do aço. (b) Vazamento de escória.

Figura 4 – Processos de vazamento de aço e escória.

Fonte: Usiminas (material de treinamento).

2.1.2 Refino secundário

No processo de refino secundário, o produto recebe tratamentos que conferem ao material características nobres, como homogeneidade química, homogeneidade térmica e limpidez. Os tratamentos se divergem de acordo com a aplicação do produto, podendo ser processados nas plantas de CAS-OB, Forno Panela e desgaseificador à vácuo (RH). A Usiminas possui como unidades de Refino secundário os equipamentos: forno panela, desgaseificador a vácuo RH, CAS-OB e estação de borbulhamento de argônio.

2.1.2.1 Forno panela

O forno panela é um equipamento que possui sistema de aquecimento elétrico através de arco-voltaico com 3 eletrodos de grafite, abóbada para reduzir o contato do aço com a atmosfera, plug poroso, lanças para injeção de gás argônio, sistema de adição de ferro-ligas e fundentes e equipamentos para tomada de amostra e medição de temperatura.

No forno panela é possível dessulfurar os aços pela injeção de cálcio em pó pela lança refratária ou pela agitação provocada sopro de argônio com alta vazão, agregando o aço com escória sintética. O tratamento no forno panela é aplicado nos nos casos de aquecimento, ajuste de composição química, dessulfuração, globulização e limpidez. A Figura 5 apresenta um ilustração do equipamento.

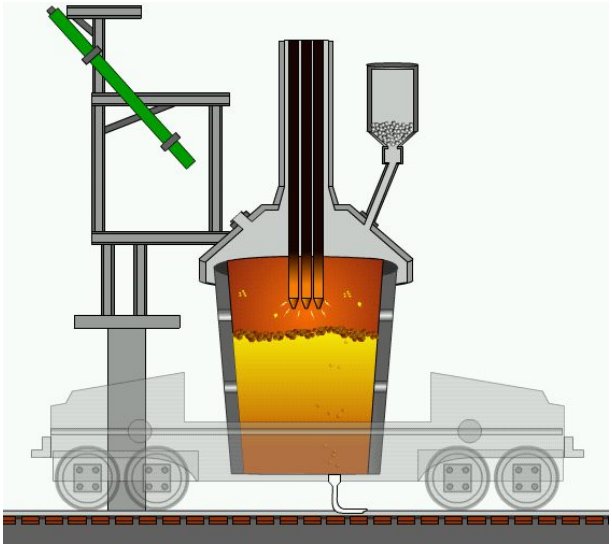


Figura 5 – Forno panela.

Fonte: Usiminas (material de treinamento).

2.1.2.2 Desgaiseficação à vácuo – RH

O desgaseificador à vácuo RH (Ruhrstahl-Heraeus) é um equipamento destinado à retirada de gases, desoxidação e limpidez dos aços. O processo de desgaseificação consiste da circulação do aço líquido no interior de um vaso, onde se faz vácuo. Na parte superior do vaso existe uma saída de gás acoplada ao sistema de produção de vácuo que é constituído basicamente de quatro a seis ejetores de vapor. Uma antecâmara para adição sob vácuo permite a introdução das ligas no interior da câmara de desgaseificação. Quando a câmara é evacuada pelo sistema de produção de vácuo, uma coluna de aço líquido com uma altura de aproximadamente 1,4 metros é atingida devido a diferença de pressão criada entre o interior da câmara (cerca de 0,5 torr) e a panela (760 torr).

Para promover a circulação de aço injeta-se um gás inerte - geralmente argônio - pela perna de subida, através de orifícios de cerca de 0,3 cm de diâmetro. Esse gás sofre aumento de volume devido ao aumento de temperatura e diminuição de pressão, fazendo com que o aço líquido adquira altura diferencial na câmara de vácuo, que é função da vazão de argônio. O aço misturado por bolhas de gás chega ao interior da câmara com velocidade e forma de gotículas de aço, liberando os gases dissolvidos no banho.

A diferença de altura fornece a energia potencial para que o aço possa retornar à panela pela perna de descida acelerado pelo seu próprio peso. Desta forma é efetuada a recirculação do aço e, consequentemente, uma desgaseificação deste através da câmara de vácuo. O desgaseificador a vácuo RH tem como funções básicas: remoção de gases dissolvidos (principalmente hidrogênio), descarburação, tratamento de limpidez, ajuste de composição química e aquecimento. O equipamento está ilustrado na Figura 6.

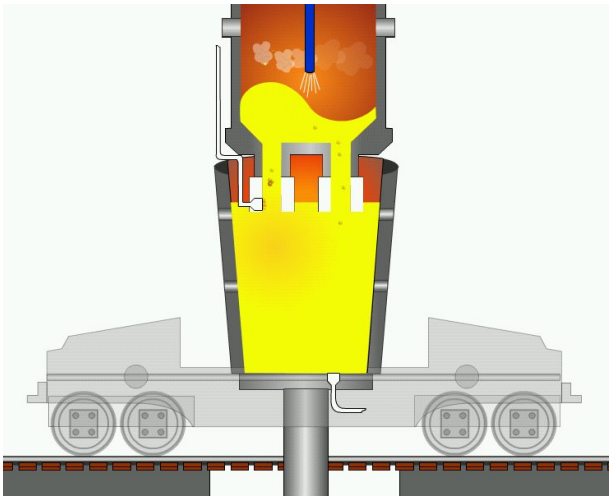


Figura 6 – Equipamento RH.

Fonte: Usiminas (material de treinamento).

2.1.2.3 CAS-OB

O CAS-OB (*Composition Adjustment by sealed Argon Bubbling, with Oxigen Blowing*), é um equipamento para ajuste de composição química e temperatura em ambiente inerte (*snorkel*) sem a presença de escória, atingindo deste modo maior rendimento de adição de ferro-ligas e alumínio pela redução da reoxidação do aço pelo ar e escória.

Similar ao forno panela, na homogeneização da composição química, limpidez e temperatura utiliza-se a injeção de argônio pelo plug poroso para afastamento da escória na superfície do aço abaixo do *snorkel*, possibilitando a adição de elementos de liga diretamente na superfície do aço em uma condição livre de escória e do oxigênio da atmosfera.

Quando necessário, o aquecimento é realizado pelo sopro do oxigênio e adição conjunto de alumínio ao material, esse processo recebe o nome de aluminotermia, pois o calor é liberado e em decorrência disso o aço é aquecido. No CASOB também é possível dessulfurar os aços pela injeção de cálcio em pó pela lança refratária, ou pela agitação provocada pelo sopro de argônio com alta vazão, agregando o aço com escória sintética. O equipamento está ilustrado na Figura 7.

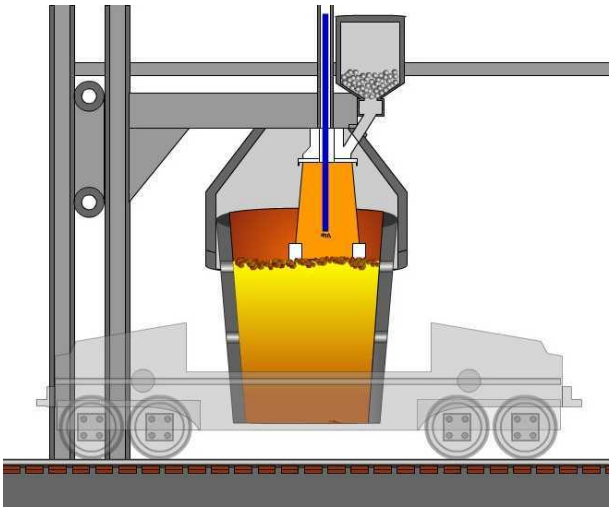


Figura 7 – Equipamento CAS-OB.

Fonte: Usiminas (material de treinamento).

2.1.3 Lingotamento contínuo

O lingotamento contínuo é o processo responsável por solidificar o aço em forma de placas após o tratamento de refino secundário. O equipamento que realiza esse processo está ilustrado na Figura 8.

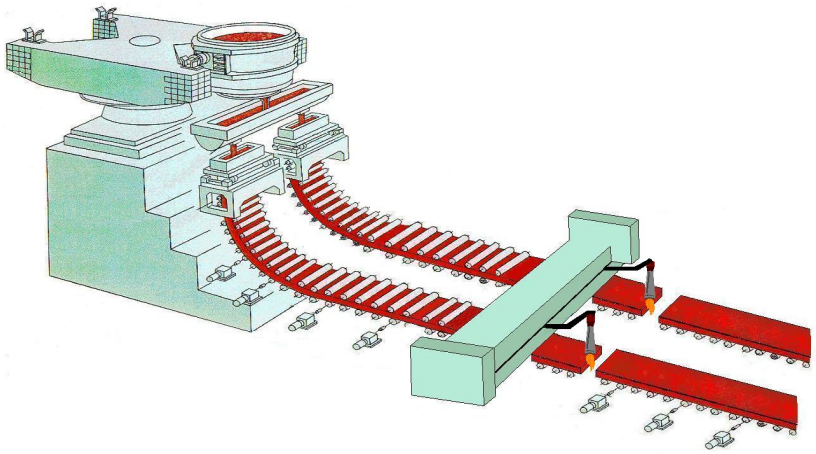


Figura 8 – Máquina de linguotamento contínuo.

Fonte: Usiminas (material de treinamento).

O processo de lingotamento contínuo é sensível às variações de temperatura e composição química, sendo necessário um processo intermediário entre os processos anteriores e a máquina de lingotamento contínuo para garantir a homogeneização do aço e temperatura. Na sequência do refino secundário ou primário, a panela de aço é transportada para a torre giratória acima do distribuidor na plataforma de lingotamento. Após abrir a válvula na extremidade inferior da panela, o aço é vazado para o distribuidor, regulando o fluxo de aço líquido para o molde retangular de cobre refrigerado a água, através das válvulas submersas.

O aço líquido, em contato com o molde, solidifica-se rapidamente dando forma à placa em formação, iniciando o processo de solidificação, formando inicialmente uma fina pele capaz de suportar o aço ainda líquido em seu interior. Essa pele solidificada permite a extração da placa e aumenta sua espessura ao longo do veio, que é constituído por rolos guias montados em segmentos. Entre os rolos estão os bicos de jatos de água e ar que promovem a extração de calor da placa, completando a solidificação, ilustrada na Figura 9.

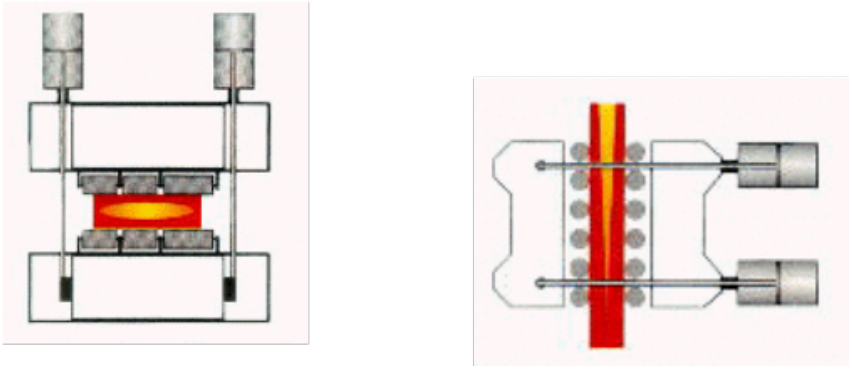


Figura 9 – Processo de solidificação do aço.

Fonte: Usiminas (material de treinamento).

A placa é desencurvada e guiada até a estação de corte - onde é subdividida em partes previamente programadas (Figura 10) - e são despachadas para a inspeção e acondicionamento de placas ou para as laminações.

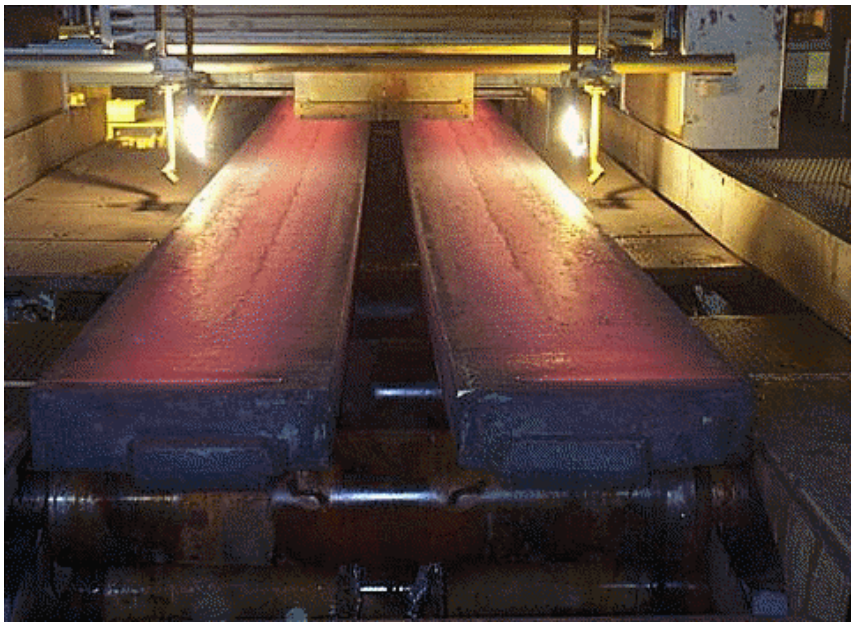


Figura 10 – Corte de placas de aço.

Fonte: Usiminas (material de treinamento).

2.2 Ciência dos dados

As próximas subseções apresentam os principais conceitos ligados à ciência dos dados que foram aplicados para o desenvolvimento do presente trabalho.

2.2.1 Dados e informação

Segundo Costa (2016), dado é um elemento, uma representação que sozinho não transmite significado. A informação é o arranjo dos dados de forma que transmitam significado, e o conhecimento se dá através do estudo/análise das informações existentes. A informação é um elemento fundamental para a existência das organizações. Está presente em todos os processos organizacionais, desde os mais simples aos mais complexos.

Dados e informações são requeridos, processados e analisados a todo o momento no contexto organizacional. Segundo Costa (2016), tão importante quanto ter informações, é saber analisá-las de forma a transformar as informações em ferramentas que auxiliam nos processos e/ou na resolução de problemas. Neste contexto, é de suma importância entender os estágios pelos quais os dados se transformam em informação, a informação em conhecimento e conhecimento em ação.

2.2.2 Extração dos dados

A extração de dados consiste em identificar, organizar e exportar dados que serão analisados a fim de obter informações referentes algum assunto ou problema. Esta tarefa requer uma análise cuidadosa sobre o problema para minimizar ambiguidades e erros nos dados. Além disso, os dados coletados devem ser significativos e cobrir amplamente o domínio do problema (FURTADO, 2019). Essa tarefa pode ser realizada através de fontes estruturadas, não estruturadas e/ou semiestruturadas, incluindo arquivos de todos os tipos, imagens, e-mails, dados de redes sociais, entre outros.

Atualmente existem diversas tecnologias que podem ser utilizadas na extração de dados, como o banco de dados relacional, muito utilizado para guardar dados de qualquer tipo. Um banco de dados relacional é composto por tabelas de nomes únicos que se relacionam entre si. Neste modelo de banco de dados uma tabela pode se relacionar a várias tabelas ao mesmo tempo, desse modo fica fácil a coleta e extração dos dados posteriormente (BOSCARIOLI et al., 2006).

2.2.3 Tratamento dos dados

No processo de transformação de dados em informação e informação em conhecimento, uma etapa de extrema importância é a de tratamento e análise dos dados (COSTA, 2016). Muitas vezes, nas bases de dados a serem trabalhadas, existem dados faltantes ou inconsistentes, assim é necessário que seja realizado um tratamento na base de dados selecionada, afim de expurgar dados errados e/ou *outliers*, identificar dados faltantes e definir o que fazer com eles ou com a falta deles. Os dados faltantes tornaram-se um dilema desafiador no processo de descoberta de conhecimento a partir de bases de dados (BROWN; KROS, 2003).

Existem técnicas já consolidadas para tratamento de dados ruidosos e faltantes. As mais utilizadas são a deleção das observações ou a inserção delas. Retirar as observações problemáticas da base de dados é uma solução aceitável, afinal, assim não se contamina o resultado das análises, no entanto, nem sempre essa é a melhor opção, pois podem existir outros dados que serão deletados juntamente com as observações faltantes que podem ser úteis para as análises.

Nestes casos, o mais indicado é fazer a inserção dos dados na base, em função de média, estimativa, classificação, entre outros métodos plausíveis, que se apliquem à variável em questão. O método de imputação consiste, basicamente, em substituir os dados faltantes por valores factíveis (VERONEZE, 2011).

2.2.4 Análise exploratória

Durante o pré-processamento dos dados, a transformação se faz presente em quase todo o processo e neste contexto, surgem termos relevantes a serem discutidos, como: categorização, padronização e normalização.

A categorização de textos por meio de algoritmos, envolve a determinação de alguma categoria pré-estabelecida a partir de uma base de referência previamente rotulada (JOHNSON *et al. apud* (BRANDÃO et al., 2022)).

Já a normalização e a padronização têm objetivo de ajustar todos os dados contidos na base a uma escala comum. O fato de existir diferenças muito grande na escala numérica das variáveis, pode gerar problemas durante a modelagem dos dados.

A normalização e a padronização reduzem a probabilidade de ocorrência de problemas dessa origem e mantêm a integridade dos dados coletados originalmente, sem distorção e sem perda de informação. Ambas são requisitos comuns para muitos estimadores de aprendizado de máquina, pois podem apresentar mau funcionamento se as variáveis individuais não se parecerem com dados normalmente distribuídos presentes na base de dados (PEDREGOSA et al., 2011).

A diferença entre a normalização e a padronização é que a normalização redimensiona os dados usando a fórmula min-max e ajusta a variável com um valor de distribuição entre 0 e 1. Já a padronização é feita pela z-score, onde se ajusta para que a média seja igual a 0 e o desvio padrão seja igual a 1, $z = (x - u)/s$, onde u é a média das amostras e s é o desvio padrão das amostras (PEDREGOSA et al., 2011).

Não existe uma indicação de qual é a melhor técnica a ser utilizada. Sua utilização varia em ressonância com o tipo de dados e o problema a ser solucionado. Uma informação relevante encontrada na literatura é aplicar ambas as técnicas, normalização/padronização, e verificar qual gera melhor performance ao modelo aplicado, ou até mesmo combinar as duas.

Ainda na análise exploratória dos dados é necessário lançar um olhar atencioso ao tamanho da base dados a ser trabalhada. A tecnologia atual permite coletar dados e armazená-los em bancos de dados de maneira fácil, o que torna os conjuntos de dados cada vez maiores. Quanto maior for o conjuntos de dados, maior será o tempo e o poder computacional utilizado para processamento das informações.

Neste sentido, a seleção dos atributos mais relevantes se tornou uma das principais tarefas da fase de pré-processamento. O ideal é que todos os atributos em uma base de dados sejam relevantes, porém, tal situação nem sempre acontece (DANTAS, 2017).

A seleção de atributos tem objetivo de indicar as variáveis mais relevantes para a resolução do problema em questão sem afetar o desempenho do modelo utilizado (DANTAS, 2017). Tal técnica permite também a remoção de dados contendo ruídos e/ou redundância e ajuda a reduzir a dimensão da base de dados, o que influi diretamente no tempo e poder computacional a ser utilizado para o processamento das informações. A maioria dos métodos existentes para a seleção de atributos tratam tanto relevância quanto redundância de atributos ao realizarem a avaliação (LEE, 2005).

Existem dois métodos de seleção de atributos que são bem difundidos na literatura: método de seleção *forward* (FSS) e método de seleção *backward* (BSS). O método de seleção *forward* inicia-se com um conjunto vazio e vai incorporando ao modelo as variáveis que seu algoritmo julga mais significativas. O método de seleção *backward* faz exatamente o contrário: inicia o modelo com todas as variáveis e vai removendo uma a uma, começando com as menos significativas (LEE, 2005). Ambos os métodos são utilizados em associação com algum algoritmo que faça a seleção das variáveis. O objetivo final desta etapa é reduzir a dimensão dos dados sem que exista perda de informação relevante para o modelo a ser aplicado.

2.2.5 Treinamento e validação

Os passos iniciais para desenvolvimento de redes neurais artificiais (RNAs) são a coleta de dados relativos ao problema e a sua separação em um conjunto de treinamento e outro de testes. Os dados de treinamento são utilizados para treinar a rede e dados de teste são utilizados para verificar a performance sob condições reais do problema.

Depois de determinados estes conjuntos, eles são colocados em ordem aleatória a fim de evitar tendências associadas à ordem de apresentação dos dados (FURTADO, 2019). Para aplicar redes neurais artificiais à resolução de um problema, são necessárias 3 etapas: o treinamento que gera o aprendizado da rede, o teste que valida as saídas geradas pela rede e a aplicação onde a rede será utilizada.

A divisão da base em treino e teste é parte fundamental na preparação de modelos, pois ajuda a escolher o modelo certo (ou um *ensemble* de modelos) e os melhores parâmetros para o mesmo. Um ponto de atenção no processo de treinamento, é que um treinamento muito longo conduz a especialização da rede naquele conjunto de dados, sobretudo se existir poucos dados disponíveis para o treinamento. O modelo fica sobre-ajustado para um determinado conjunto de dados mas fica com baixa capacidade de generalização para outros dados ainda não vistos (*overfitting*).

Tal situação piora a performance da rede quando novos dados são apresentados a ela (FURTADO, 2019). Ao conjunto de teste são aplicadas métricas de desempenho para que os resultados possam ser comparados (SILVA, 2022). Dessa forma é possível verificar a eficiência da rede.

2.3 Aprendizagem de máquina

A aprendizagem de máquina é um termo utilizado para definir uma área da inteligência artificial que trata da construção de sistemas capazes de adquirir conhecimento de forma automática, por métodos e algoritmos, tomando decisões baseadas em experiências de problemas anteriores. A partir de dados de treinamento, os algoritmos procuram por padrões de entrada e saída para realizar predições (MONARD; BARANAUSKAS, 2003).

O desenvolvimento de um algoritmo de aprendizagem de máquina é dividido em três fases (Figura 11), que são a de pré-processamento, quando é feita a organização da base de dados, definida a pergunta da pesquisa e é feita a divisão da base em treinamento e teste; treinamento e avaliação do modelo (PAIXÃO et al., 2022).

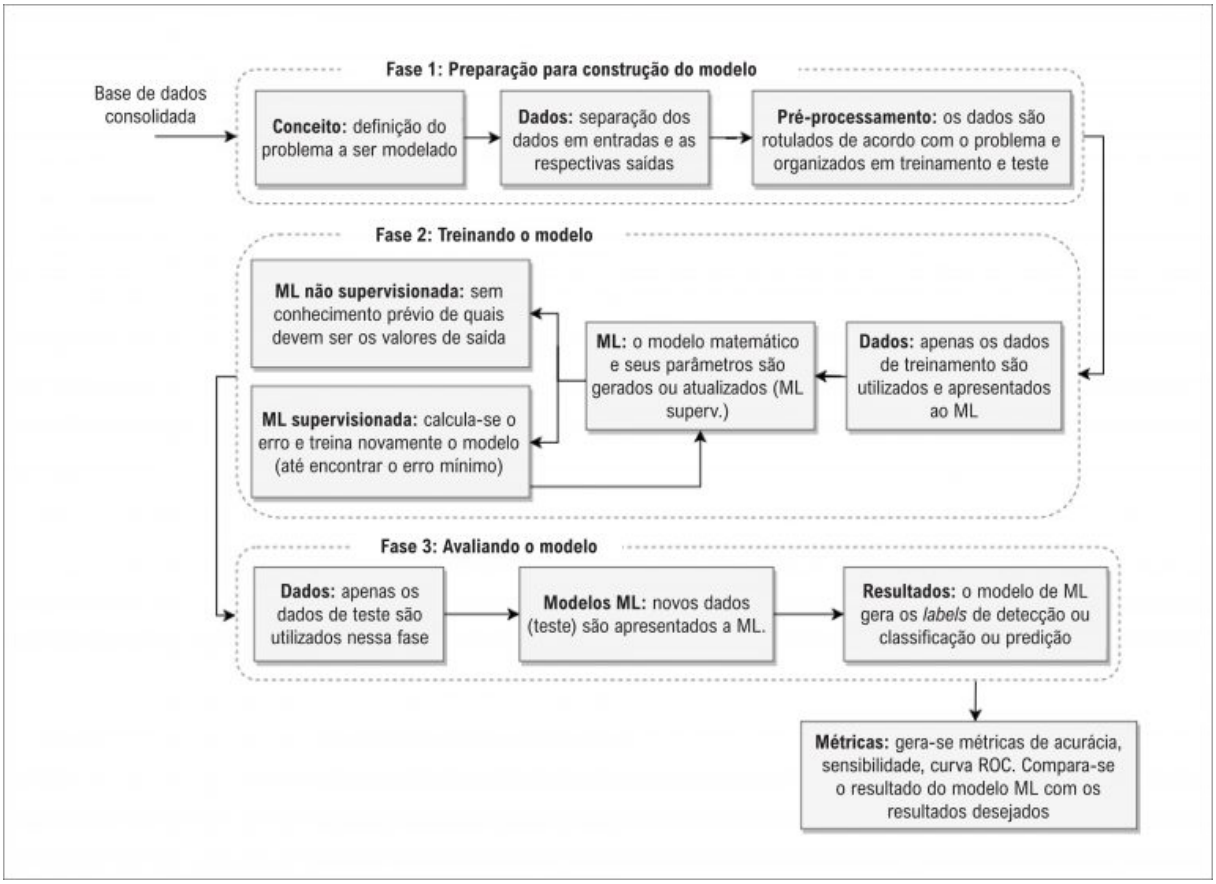


Figura 11 – Fases para o desenvolvimento de algoritmos de aprendizagem de máquina.

Fonte: Paixão et al. (2022).

De acordo com Fontana (2020), o treinamento dos algoritmos de aprendizagem de máquina podem ocorrer de forma supervisionada ou não-supervisionada:

- Aprendizagem supervisionada: relaciona saída com entrada baseada em dados rotulados, sendo o rótulo, um valor numérico ou classe. O algoritmo é treinado, com base em dados de entrada e saída rotulados, a predizer uma saída dada a condição de entrada. Os algoritmos onde a saída pode assumir somente um conjunto de rótulos pré-definidos, são chamados de algoritmos de classificação, e os algoritmos onde os rótulos de saída podem ser valores reais baseados em uma condição de entrada são chamados de algoritmos de regressão.

- Aprendizagem não-supervisionada: não tem dados de saída com rótulo atribuído, sendo essa a principal diferença com o modelo de aprendizagem supervisionado. O algoritmo busca padrões entre os dados, utilizando uma base com grande número de dados, identificando grupos similares ou similaridade de itens em grupos definidos. Os algoritmos podem ser divididos em algoritmos de transformação e algoritmos de agrupamento.

2.4 Redes neurais artificiais

As redes neurais artificiais (RNAs) são uma das técnicas mais adotadas de aprendizagem de máquina na literatura recente. As RNAs têm inspiração no cérebro humano que é um computador complexo, paralelo e não-linear (AZEVEDO, 2022). Os neurônios artificiais são estruturas lógicas/matemáticas e da mesma maneira que os neurônios humanos recebem estímulos, processam informações e repassam os resultados através de suas conexões de saídas, conseguindo assim, resolver problemas complexos através do conhecimento adquirido por meio de treinamento e aprendizado.

Existem vários métodos para se modelar uma RNA, porém, o método mais usual é o método empírico, que consiste em gerar um padrão de interconexão a fim de resolver problemas por meio de um processo de treinamento, podendo o criador da rede modificá-lo gradualmente, adaptando o modelo à deliberação do problema (FURTADO, 2019).

Os sinais de entrada em neurônio podem ser oriundos da entrada da rede ou de outros neurônios. Cada sinal recebido possui um peso sináptico associado $(w_1, w_2, w_3, \ldots, w_n)$ assim é realizada a soma dos produtos entre os sinais de entrada e os pesos sinápticos. O resultado do somatório é aplicado a uma função que gera o resultado final (AZEVEDO, 2022). O fim do processamento pelo neurônio se dá com a aplicação do resultado do somatório dos produtos das entradas multiplicadas pelos pesos a uma função de ativação previamente definida que gera o resultado final.

Segundo Furtado (2019), a capacidade de resolver um problema é definida pela arquitetura da rede, isto é, definida pelo número e modo que os neurônios processadores estão interconectados nos pesos e no número de camadas aplicadas. Os mecanismos de aprendizado permitem a modificação do padrão de interconexão. Para treinamento podem ser utilizados métodos de aprendizagem supervisionada, não-supervisionada, por reforço ou conforme a arquitetura neural fixada na rede.

2.4.1 Funções de ativação

As redes neurais artificiais possuem funções de ativação, que são responsáveis por alimentar o neurônio atual e enviar a saída para a próxima camada. Gharat (2019) relata que sem as funções de ativação a rede neural seria um modelo de regressão linear, limitada na capacidade de resolver problemas complexos. As redes geralmente usam funções de ativação não lineares para ajudar a calcular e aprender com dados complexos, dando respostas mais precisas. Caso a rede usasse apenas função de ativação linear, independente da quantidade de camadas, a rede se comportaria como uma rede de camada única, porque a soma das camadas daria outra função linear (GHARAT, 2019).

Conforme Gharat (2019), as funções de ativação são divididas basicamente em dois tipos: função de ativação linear ou de identidade e funções de ativação não lineares. Ambas estão descritas abaixo.

A função de ativação linear ou de identidade pega as entradas, multiplicadas pelos pesos de cada neurônio e cria um sinal de saída proporcional à entrada. Pode ser representada graficamente na Figura 12.

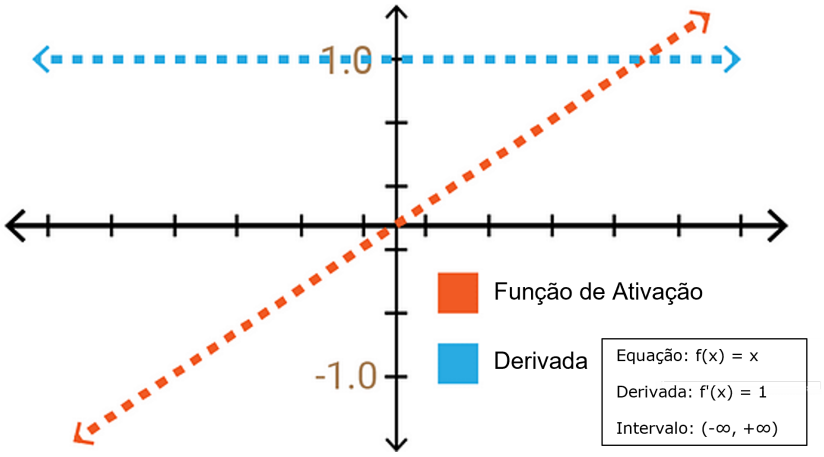


Figura 12 – Representação gráfica da função linear.

Fonte: Adaptado de Gharat (2019).

As funções não lineares não têm as limitações da função linear. Elas permitem o uso do *back propagation*, por possuírem uma derivada relacionada com as entradas e permitem o uso de múltiplas camadas de neurônios, para criar redes neurais profundas, com várias camadas, que são usadas para resolver problemas complexos. Entre as mais comuns, são descritas: ReLU (*Rectified Linear Unit* - Unidade Linear Retificada), Tanh (Tangente hiperbólica) e Sigmoide.

A função ReLU retorna um valor positivo ou 0 para os valores negativos. De acordo com (GUPTA, 2022), a principal vantagem de usar a função ReLU sobre outras funções de ativação é que ela não ativa todos os neurônios ao mesmo tempo, sendo considerada a mais eficiente para aprendizado profundo. Como desvantagem no uso dessa dessa função, está o fato de que para as entradas se aproximam de zero ou são negativas, não haverá aprendizado na rede, uma vez que não existe derivada para este domínio da função (BRAGA, 2022). A função ReLU pode ser representada no gráfico da Figura 13.

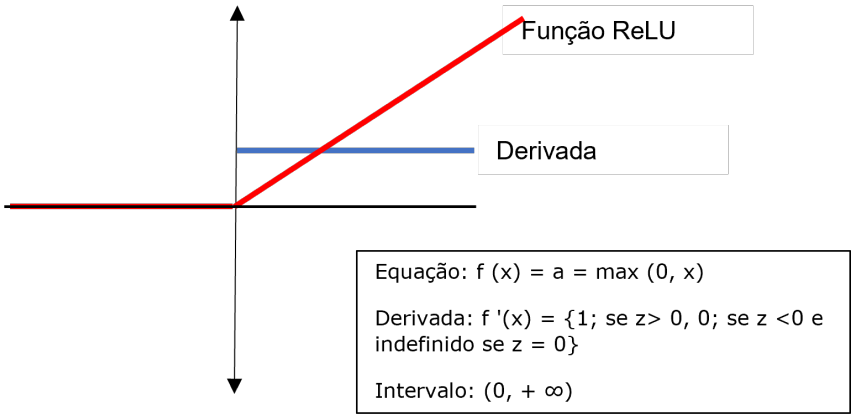


Figura 13 – Representação gráfica da função ReLU.

Fonte: Adaptado de Braga (2022).

A função Sigmoide, ou logística sigmoidal, é caracterizada como uma curva em 'S' na representação gráfica (GHARAT, 2019). A função mapeia qualquer valor real em outro valor entre 0 e 1. Como vantagem dessa função, Gharat (2019) cita que se os valores forem próximos de zero a função que consegue os resultados em menos tempo e com a menor margem de erro quando comparada a outras funções. Como desvantagem, nos casos em que os valores muito altos ou muito baixos, distantes de zero, a derivada nessas regiões se aproxima de zero, o que pode causar um problema de gradiente de fuga, com convergência lenta. A função Sigmoide pode ser representada no gráfico da Figura 14.

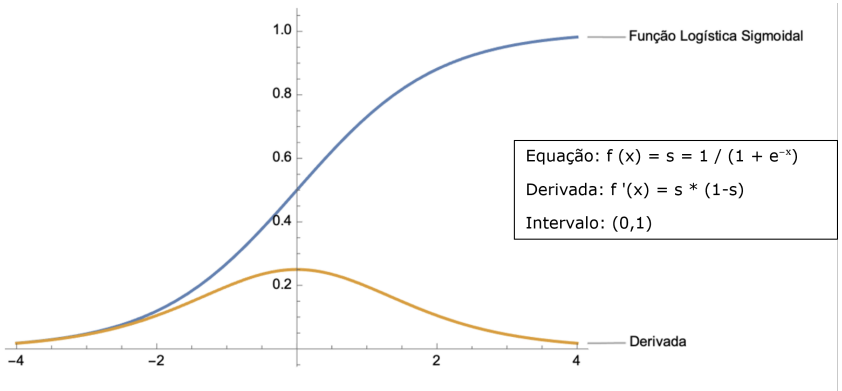


Figura 14 – Representação gráfica da função Sigmoide.

Fonte: Braga (2022).

Braga (2022) caracteriza a função TanH, ou tangente hiperbólica, como "bem similar à função sigmoidal", sendo a diferença básica assumir valores positivos e negativos. A função Tanh pode ser representada no gráfico da Figura 15.

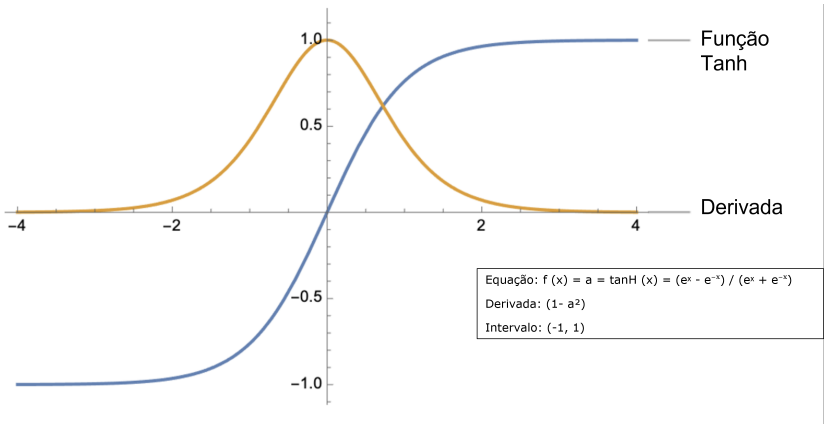


Figura 15 – Representação gráfica da função Tanh.

Fonte: Braga (2022).

São diversas as áreas para aplicação de redes neurais tais como, engenharia, economia, agronomia, medicina etc., resolvendo problemas que envolvam extração de características, classificação, categorização/clusterização; estimativa e previsão; otimização; aproximação de funções; dentre outras (FURTADO, 2019).

2.4.2 Tipos de RNAs

As redes neurais se distinguem em três tipos quanto a sua arquitetura: *feed forward* de uma camada, *feed forward* multicamadas e redes realimentadas ou recorrentes.

2.4.2.1 *Feed forward* de uma camada

Nas redes neurais do tipo *feed forward* de uma camada, os neurônios de entrada se comunicam diretamente com a camada de saída e não existem ligações entre os neurônios de mesma camada ou camadas anteriores. Neste tipo de rede os nós da entrada não processam informação, apenas repassam as informações à camada de saída e as informações se propagam sempre no sentido da entrada para saída. (FURTADO, 2019). A Figura 16 apresenta a estrutura de uma rede neural do tipo *feed forward* de uma camada.

2.4.2.2 *Feed forward* multi-camadas

Nas redes do tipo *feed forward* multi-camadas, a informação também flui sempre no sentido da entrada para saída e os nós da camada anterior comunicam-se com todos os nós da camada posterior. Assim a rede é totalmente conectada (FURTADO, 2019). A Figura 17 apresenta a estrutura de uma rede neural do tipo *feed forward* multi-camadas.

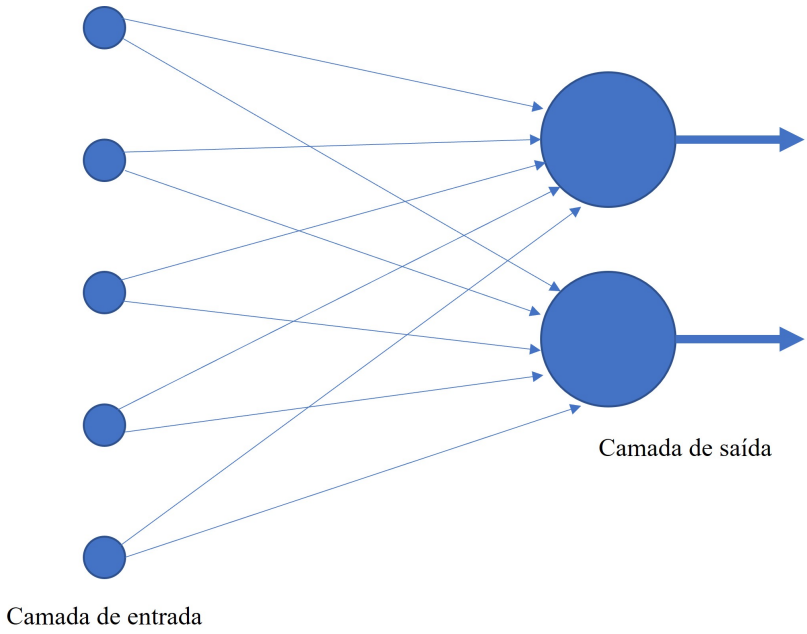


Figura 16 – RNA *feed forward* de uma camada.

Fonte: Adaptado de Furtado (2019).

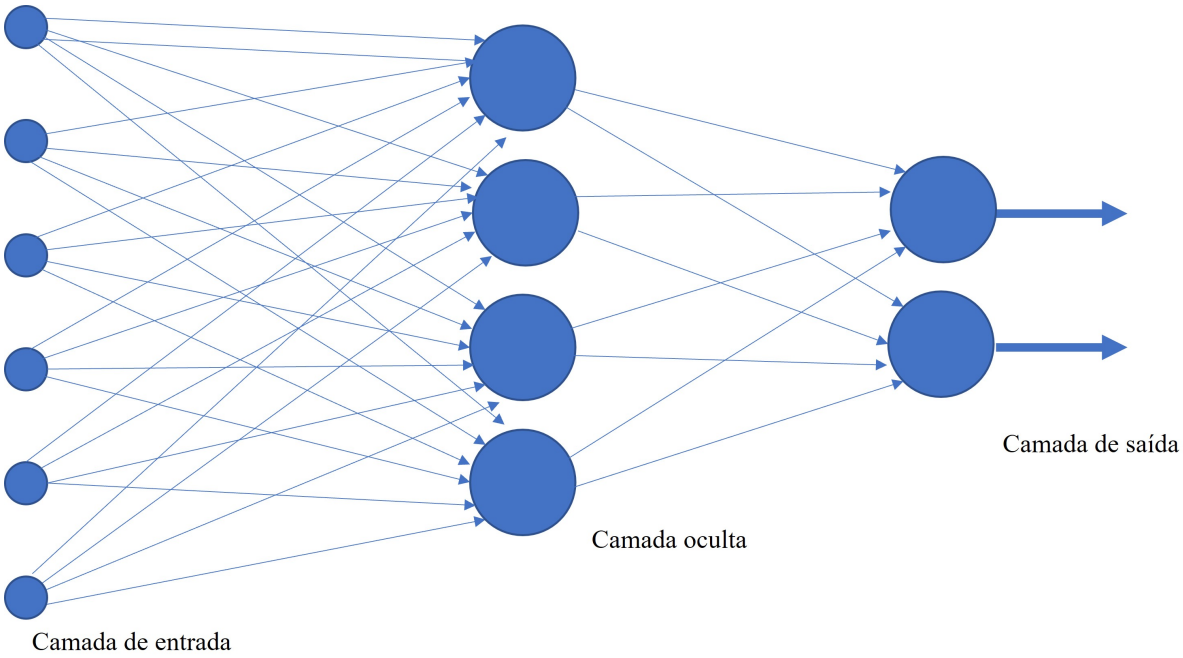


Figura 17 – RNA *feed forward* multi-camadas

Fonte: Adaptado de Furtado (2019)

2.4.2.3 Retroalimentadas

As redes realimentadas ou recorrentes permitem a realimentação de uma camada anterior com informações de saída de uma camada posterior. É possível também realimentar um neurônio com a informação gerada por sua própria saída. A Figura 18 apresenta a estrutura de uma rede neural do tipo recorrente.

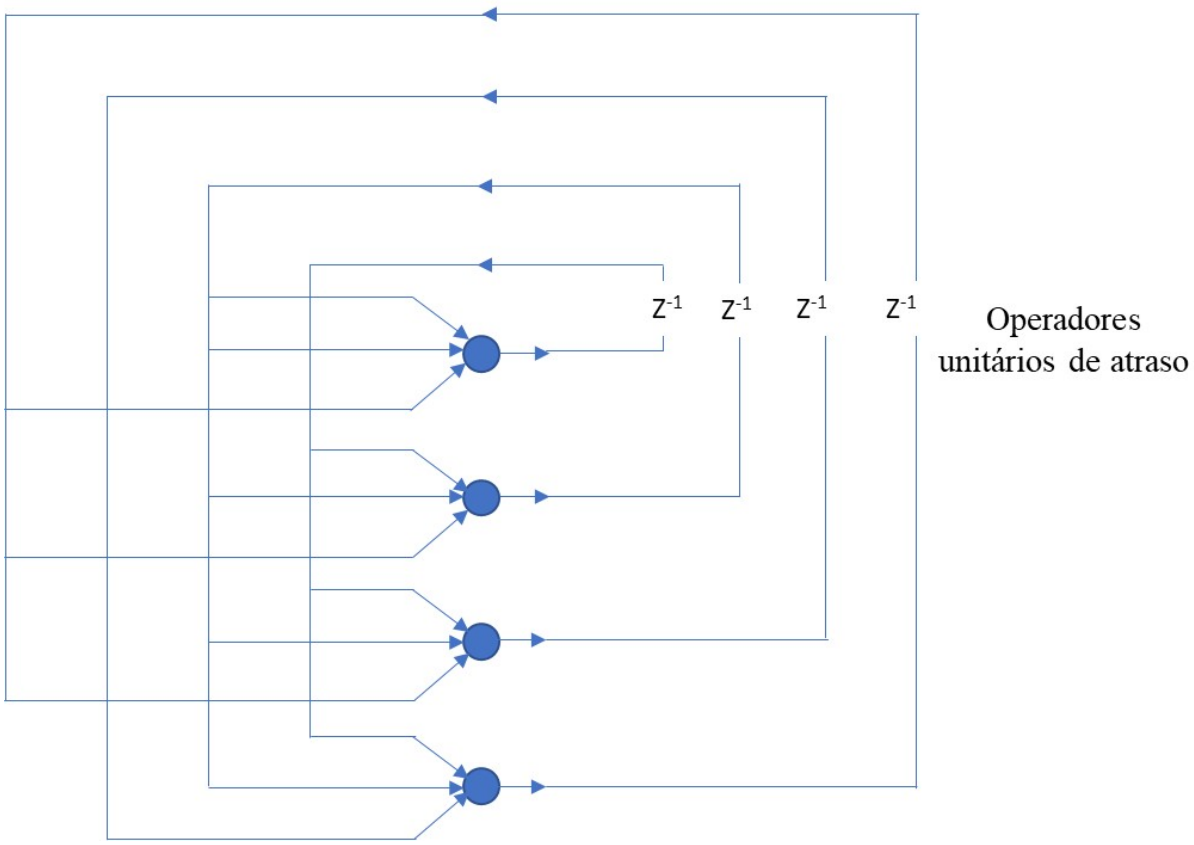


Figura 18 – RNA retroalimentada.

Fonte: Adaptado de Furtado (2019).

2.4.3 Treinamento e aprendizado

As redes neurais artificiais têm a capacidade de aprender a executar uma determinada tarefa, uma vez submetida a treinamento em relação ao problema que se deseja solucionar.

Na etapa de treinamento de uma rede é realizada a adaptação dos pesos iniciais e a aprendizagem do seu comportamento é especificada pelas regras do treinamento. Os algoritmos de treinamento/aprendizado ajustam de forma iterativa os pesos das conexões entre os neurônios até que os pares de entradas e saídas esperados sejam obtidos e as relações matemáticas de causa e efeito possam ser estabelecidas (GONÇALVES; COELHO; KRUEGER, 2010).

Se a configuração do problema em questão muda, é possível submeter a rede a mais treinamento apresentando novas condições de entrada e saída, a fim de melhorar o desempenho.

Em síntese, o processo de aprendizado da rede neural é estimulado pelo meio, sofre mudanças em seus parâmetros e, como resposta ao estímulo, devolve ao meio uma nova condição obtida através da mudança ocorrida na sua estrutura interna (FURTADO, 2019).

2.4.4 Redes neurais MLP

Nessa subseção é detalhada a rede neural MLP que foi aplicada nesse trabalho.

O conceito de redes do tipo perceptron foi apresentado em 1958 por Frank Rosenblatt. Consiste em uma rede neural construída com neurônios MCP (McCulloch-Pitts), onde as entradas são multiplicadas por pesos e os resultados são somados e comparados a um linear e arranjada de forma composta com duas camadas (NIED, 2007). Esta configuração mostrou-se eficiente para classificar padrões, porém não apresentou bons resultados na resolução de problemas mais complexos e não lineares (FURTADO, 2019). Alguns anos depois, com a criação de algoritmos que possibilitaram o treinamento de redes com múltiplas camadas, desenvolveu-se a estrutura de redes MLP (*Multilayer Perceptron*), que é a generalização do perceptron de única camada, acrescentando camadas ocultas ou intermediárias a rede. Esta nova arquitetura de rede proporcionou maior poder computacional às RNAs e dessa forma, o perceptron multi-camadas consegue tratar dados não linearmente separáveis e assim atuar na resolução de problemas complexos se sobressaindo em relação às redes configuradas sem camada intermediária.

A estrutura básica de uma rede MLP apresentada na Figura 19, consiste em uma camada de entrada, uma ou mais camadas intermediárias e uma camada de saída.

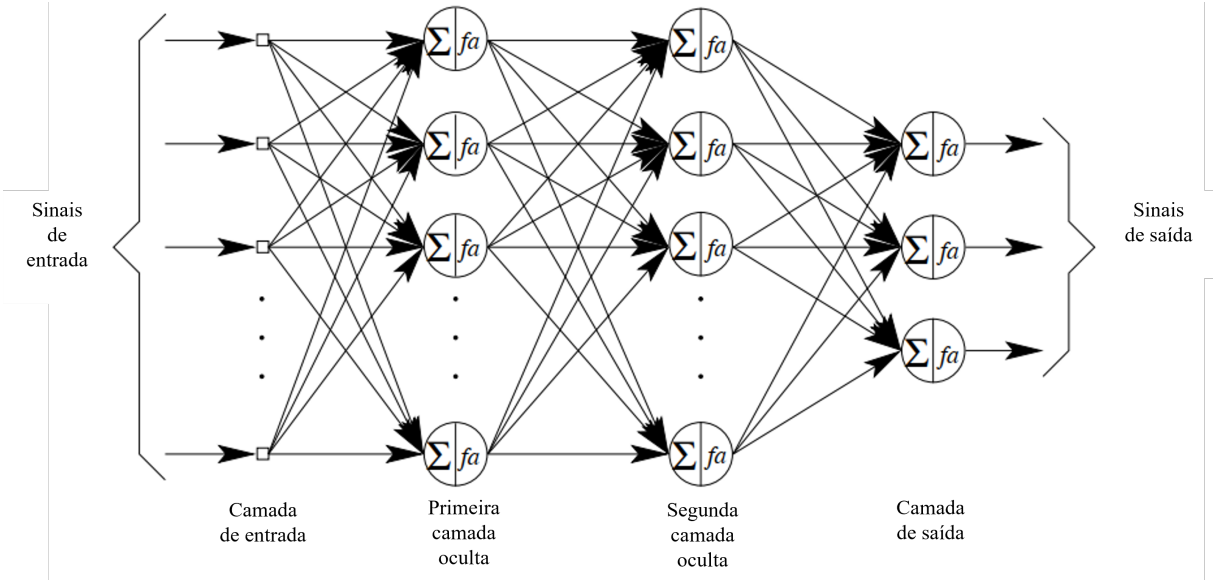


Figura 19 – Arquitetura de uma rede MLP com duas camadas ocultas.

Fonte: Adaptado de Nied (2007).

Nesta arquitetura, tipicamente, as entradas dos neurônios de cada camada, são as saídas dos neurônios das camadas antecessoras. Na camada de saída é concentrado o resultado global da rede, constituindo assim, o modelo de funcionamento do tipo Feedforward, também conhecido como alimentado adiante, onde o sinal se propaga da entrada para a saída.

Em uma rede MLP os neurônios são dispostos em várias camadas. Na camada de entrada, os padrões são apresentados; as camadas ocultas ou intermediárias são responsáveis pela maior parte do processamento e a camada de saída apresenta a conclusão do processamento (FURTADO, 2019).

Para as redes do tipo MLP não existe uma restrição para a escolha da função de ativação a ser usada, mas para que a rede consiga resolver problemas não lineares, é necessário que pelo menos uma camada oculta ou intermediária tenha funções de ativação não lineares (AZEVEDO, 2022). Quanto às funções de ativação em uma rede MLP, é importante ressaltar que não é obrigatório que seja utilizada a mesma função em todos os neurônios da rede.

A arquitetura de uma rede e a tarefa que a rede neural deve realizar estão intimamente relacionadas com o algoritmo de aprendizagem a ser usado para treiná-la e o tipo de treinamento a que a rede será submetida (FURTADO, 2019). Para treinamento das redes tipo MPL, o algoritmo mais utilizado é o algoritmo de retropropagação (*Backpropagation*) com aprendizado supervisionado.

Este algoritmo utiliza o processo de tentativa e erro até chegar a um resultado satisfatório orientado pelo agente supervisor. Um ponto negativo do *backpropagation*, é seu tempo de execução que dependendo do problema pode ser alto, justamente devido a tentativa e erro em cada iteração, onde o tempo é influenciado pela taxa de atualização dos pesos.

Se a taxa de atualização for muito baixa, a rede consome muito tempo para o treinamento, do contrário, se a taxa de atualização é alta, a rede alcança rapidamente um resultado satisfatório, porém se outra entrada for apresentada, a rede se torna instável ocasionado outro problema que é a confiabilidade dos resultados (BARCA; SILVEIRA, 2005). Ainda segundo Barca e Silveira (2005), para reduzir o tempo de convergência da rede treinada por este algoritmo existem técnicas de otimização numérica, como a técnica do gradiente descendente, regra delta, técnica de Levenberg Marquardt e Filtragem de Kalman. Mais detalhes sobre essas técnicas podem ser encontrados em Gardenghi e Santos (2011), Benatti (2017), Schwertner (2020) e Teixeira, Torres e Aguirre (2010).

De modo geral, uma rede do tipo MLP possui grande poder de processamento e consegue extrair conhecimento a partir de dados de treinamento e generalizá-lo, a dados desconhecidos a fim de realizar tarefas complexas. É importante mencionar que o aprendizado da rede se dá através dos constantes ajustes nos pesos sinápticos de entrada, que inicialmente são aleatórios, à passagem de cada iteração do treino (SILVA, 2022).

2.4.5 Métricas de avaliação de desempenho de RNAs

A avaliação do modelo criado é importante para entender e explicar seu desempenho. Existem muitas métricas de avaliação. Para os problemas de regressão, de acordo com Wu (2020), as principais são o *R Square* - (R^2), *Mean Square Error* - (MSE) / *Root Mean Square Error* - (RMSE) e *Mean Absolute Error* - (MAE).

O R^2 mede quanto da variável dependente pode ser explicado pela variável independente. É o quadrado do coeficiente de correlação (R). O valor de R^2 está entre 0 e 1, e um valor maior indica um melhor ajuste entre a previsão e o valor real. O R^2 não leva em consideração o problema de *overfitting*. Um modelo de regressão com uma base de dados complexa pode se ajustar muito bem no treinamento, mas ter um desempenho ruim no teste.

O MSE é uma medida absoluta da qualidade do ajuste. Ele fornece um número absoluto de quantos resultados previstos no modelo se desviam do valor real. O RMSE é a raiz quadrada do MSE, e é usado com mais frequência do que o MSE porque, de acordo com (WU, 2020), o valor do MSE pode ser algumas vezes muito grande para ser comparado facilmente e também, devido ao MSE ser calculado pelo quadrado do erro, a raiz quadrada o traz de volta ao mesmo nível de erro de previsão e facilita a interpretação.

O MAE é semelhante ao MSE. Avalia a distância absoluta das observações (as entradas do conjunto de dados) às previsões de uma regressão, tirando a média de todas as observações (TREVISAN, 2022). O valor absoluto das distâncias é utilizado para que os erros negativos sejam contabilizados adequadamente.

Wu (2020) diz que o R^2 é melhor usado para explicar o modelo como uma porcentagem da variabilidade de saída, e o MSE, RMSE e MAE são melhores para comparar o desempenho entre diferentes modelos de regressão. As métricas citadas podem ser calculadas em `Phyton` utilizado o pacote `Sklearn`.

2.5 Revisão de literatura

A aplicação de métodos de aprendizagem de máquina no contexto dos processos de produção de aço vem sendo abordada recentemente na literatura. Como exemplo, Fucun et al. (2018) aplicaram aprendizagem de máquina baseada em conjunto (*ensemble learning*) para a predição da qualidade do aço ao final do processo produtivo de acordo com dados históricos. O modelo proposto foi integrado ao processo produtivo e auxiliou a prever a qualidade do aço com erro de $2,86\%$.

Kholief, Darwish e Fors (2017) utilizaram aprendizagem de máquina baseada em redes neurais artificiais para detectar defeitos superficiais em aços laminados a quente através de imagens capturadas no processo. Foram propostos dois modelos, *feed forward* e *deep auto-encoder*, na classificação de 6 tipos de defeitos. Para o modelo de RNAs *feed forward*, a base de dados foi dividida em treinamento (90%) e teste (10%). Através de histograma apresentado o modelo *feed forward* apresentou acurácia de 100% no treinamento e entre $13,3\%$ e 73% no teste, utilizando validação cruzada. Para o modelo *deep auto-encoder*, a base de dados foi dividida em treinamento (50%) e teste (50%). Os dados foram tabulados em matriz de confusão e apresentaram acurácia geral de 100%.

Zhang, Zhang e Fan (2016) propuseram um modelo de previsão de *breakout* utilizando redes neurais *back propagation* (BP). O modelo proposto utiliza séries temporais para reconhecer a mudança de temperatura em ondas pelos termopares embutidos no molde. Os resultados mostraram que a acurácia do modelo para identificar as mudanças de temperatura foi de $96,43\%$.

Com o objetivo de criar um modelo matemático para prever o teor de hidrogênio no aço, Correa et al. (2019) criaram uma rede neural MLP por entenderem que era o recurso mais adequado para trabalhar inúmeros dados de entrada e obter um dado de saída. A base de dados foi dividida em 15% para testes, 15% para validação e 70% para treinamento. Os valores preditos foram comparados com os valores reais medidos no processo. Os resultados do modelo criado foram de 71,05% no treinamento, 63,04% no teste e 69,58% na validação.

Assim como o presente trabalho, os trabalhos citados aplicam aprendizagem de máquina no contexto industrial. Semelhante ao trabalho de Correa et al. (2019) obter o valor de hidrogênio no aço líquido é o objetivo desse trabalho. Os parâmetros utilizados nos modelos criados no presente trabalho foram diferentes do modelo proposto no trabalho de Correa et al. (2019), bem como os resultados obtidos. Outros fatores diferentes entre os dois trabalhos são as bases de dados, referentes a aciarias diferentes, equipamentos e processos.

3 Metodologia

O presente trabalho utilizou uma base de dados referente ao processamento de materiais durante a fabricação de aço, coletados nas linhas de produção da aciaria da Usinas Siderúrgicas de Minas Gerais, Usiminas, na planta de Ipatinga.

3.1 Extração dos dados

Os dados foram extraídos da base de produção dos processos de refino secundário da aciaria 2, processos anteriores ao lingotamento, onde os materiais foram processados. Cada processo produtivo possui modelos computadorizados que monitoram em tempo real cada etapa de refinamento, medindo e ajustando a composição química e a temperatura do aço em função das necessidades dos clientes. Todas as informações são coletadas e armazenadas em banco de dados relacional, possibilitando a rastreabilidade e controle das informações relativas aos materiais produzidos.

A base de dados utilizada possui 256 variáveis e 51.952 observações, referentes ao período de janeiro de 2019 a novembro de 2022, contendo informações das corridas, tais como: composição química; materiais adicionados para correção de características; dados ambientais (umidade do ar); tempo de tratamento nos equipamentos, entre outras.

O banco de dados foi acessado através de comandos de linguagem trans-SQL. Os dados selecionados foram extraídos para arquivo no formato `.xls`. Dentre os formatos disponíveis das variáveis coletadas, a base de dados possui informações do tipo data, inteiro, *float* e *string*. A base de dados foi carregada na plataforma Google Colab, acessada e tratada com o uso da linguagem de programação Python.

Através da função de leitura de arquivos `.xls` do Python (`pd.read_excel`), a base de dados foi carregada em um *dataframe* e a variável resposta `H_y`, foi remanejada para a última posição do *dataframe*, conforme mostra o Código 3.1.

As 256 variáveis da base de dados estão descritas no Apêndice A.

Código 3.1 – Carregamento da base de dados com o Python no Google Colab.

```
#Coleta dos documentos (pelo Drive)
dados = pd.read_excel('/content/drive/MyDrive/Colab Notebooks/
DadosH_Completa_Final_v10.xlsx')
#Organizar a coluna H, a colocando na última posição
dados = dados[[col for col in dados.columns if col != 'H_y' ] + ['H_y']]
dados.shape
```

3.2 Tratamento dos dados

Em uma verificação inicial na base de dados utilizada neste estudo, foram identificados alguns problemas comuns em base de dados, tais como: dados faltantes em várias observações, dados errados e *outliers*. No modelo aplicado, para verificar se o resultado encontrado pela rede neural artificial é satisfatório, é necessário saber o teor de hidrogênio medido no processo de cada corrida. Esse valor é apresentado na variável `H_y` e varia entre 0 a 14 unidades. Especialistas do processo indicaram que 0 de hidrogênio no aço é impossível e 14 é o maior valor histórico já medido pelo equipamento físico.

3.2.1 Ajustes preliminares

Conforme orientação dos especialistas do processo de aciaria, foram retiradas da base dados 11 variáveis, já conhecidas pela equipe do processo, que não contribuem para a inclusão de hidrogênio no aço, ou seja, não agregam valor ao modelo proposto. São elas: número da corrida (`Corrida`), data (`Data`), tipo de tratamento (`Trat`), justificativa para vácuo maior que 2torr (`Just_Vacuo`), volume de oxigênio no convertedor (`OL_OX_LD`), alumínio em barra (`ALBA`), coque calculado no RH (`COQ_CALC_RH`), silício metálico (`SIMT`), alumínio em fio (`ALFI`), antimônio (`ANTI`) e ferro titânio a granel (`FETR`). Foram removidas também as variáveis com a composição química visada e real das corridas, pois já existe na base de dados outras variáveis com a diferença entre elas, que são utilizadas no modelo, conforme apresentado no Código 3.2.

Código 3.2 – Exclusão de variáveis da base de dados que não agregam valor ao modelo.

```
#Removendo variáveis que não agregam valor ao modelo
Col_drop = ['Corrida', 'Data', 'Trat', 'Just_Vacuo', 'VOL_OX_LD', 'ALBA',
'COQ_CALC_RH', 'SIMT', 'ALFI', 'ANTI', 'FETR']
df = df.drop(Col_drop, axis=1)

#Remover as composições químicas visadas por já existir o cálculo da
#diferença (delta) entre visado e atual
Col_drop = ['C_VIS', 'SI_VIS', 'MN_VIS', 'P_VIS', 'S_VIS', 'AL_VIS',
'CU_VIS', 'NB_VIS', 'V_VIS', 'TI_VIS', 'CR_VIS', 'NI_VIS', 'MO_VIS',
'SN_VIS', 'N_VIS', 'H2_VIS', 'CO_VIS', 'AS_VIS', 'O2_VIS', 'B_VIS',
'CA_VIS', 'AT_VIS', 'SB_VIS', 'ZR_VIS', 'BI_VIS', 'CE_VIS', 'W_VIS',
'PB_VIS', 'CD_VIS', 'BE_VIS', 'ZN_VIS', 'BA_VIS', 'HG_VIS', 'CH_VIS',
'BSOL_VIS', 'AO_VIS', 'AI_VIS', 'TS_VIS']
df = df.drop(Col_drop, axis=1)

#Remover as composições químicas por já existir o cálculo da diferença
#(delta) entre visado e atual
Col_drop = ['AL', 'AS', 'AT', 'B', 'C', 'CA', 'CO', 'CR', 'CU', 'H',
'MN', 'MO', 'N', 'NB', 'NI', 'O', 'P', 'S', 'SB', 'SI', 'SN', 'TI',
'V', 'ZR', 'PB', 'BI', 'W', 'CD', 'BE', 'ZN', 'BA', 'HG', 'CH',
'BSOL', 'AO', 'AI', 'TS']
df = df.drop(Col_drop, axis=1)
```

Durante a validação do modelo, foram identificadas 10 corridas que apresentavam alta variação entre o valor real medido e o valor predito pela rede, com variação acima de 6ppm. Afim de refinar os dados utilizados, a entrada de dados para tais corridas foram revisadas na base de dados e o valor de medição real foi corrigido de acordo medições feitas em laboratório, que são uma contraprova para o hydris. A correção está apresentada no Código 3.3.

Código 3.3 – Correção de dados de corrida conforme valores de medição em laboratório.

```
#Ajuste de hidrogenio devido a erros de medição identificados na amostra
#Erro na medição no lingotamento
dados.loc[dados['Corrida'] == 548150, 'H_y'] = 4.82
dados.loc[dados['Corrida'] == 433488, 'H_y'] = 6.50
dados.loc[dados['Corrida'] == 426117, 'H_y'] = 7.12
dados.loc[dados['Corrida'] == 544495, 'H_y'] = 6.33
dados.loc[dados['Corrida'] == 427297, 'H_y'] = 6.33
dados.loc[dados['Corrida'] == 448538, 'H_y'] = 7.00
dados.loc[dados['Corrida'] == 444786, 'H_y'] = 5.00
dados.loc[dados['Corrida'] == 426832, 'H_y'] = 6.88
dados.loc[dados['Corrida'] == 544495, 'H_y'] = 6.33
dados.loc[dados['Corrida'] == 427512, 'H_y'] = 6.70
dados.loc[dados['Corrida'] == 548835, 'H_y'] = 7.90
```

3.2.2 Dados faltantes

Em sequência, foi realizada uma verificação na base de dados a fim de analisar os dados faltantes. A função `.isnull().sum().sum()` do Python percorre a base de dados a procura de dados nulos e retorna um somatório das ocorrências encontradas para cada variável de entrada, conforme mostra o Código 3.4.

Código 3.4 – Verificação de dados faltantes.

```
#Verificar a existência de valores nulos
df.isnull().sum().sum()
df.isnull().sum().sort_values(ascending=False)

> DIF_CD      51952
> DIF_AO      51952
> DIF_TS      51952
> DIF_AI      51952
> DIF_BE      51952
>             ...
> BRIQ_OB         0
> CSAL_OB         0
> CASV_OB         0
> FECV_OB         0
> H_y             0
> Length: 170, dtype: int64
```

A consulta mostrou que 70 variáveis possuíam todas as observações igual a zero. Essas variáveis foram removidas dos dados de entrada conforme mostra o Código 3.5.

Código 3.5 – Remoção de variáveis com valor total zero.

```
#Remover as colunas onde os valores de todas as linhas sao zero
print('Total de variáveis da base antes: ', df.shape[1])

df = df.loc[:, (df != 0).any(axis=0)]
dados_mod = dados.loc[df.index]

print('Total de variáveis da base depois: ', df.shape[1])
```

Após a remoção das variáveis com todas as observações igual a zero, foi gerado um *box plot* da base de dados para verificar se existiam distorções nos dados. As variáveis que apresentaram distorções são mostradas individualmente nos *box plot* a seguir conforme figuras 20, 21, 22 e 23 .

O *box plot* é uma ferramenta gráfica muito utilizada durante a execução da análise exploratória de dados. É a representação gráfica da distribuição dos dados de uma variável em função de seus parâmetros. É importante para identificar a posição, dispersão e assimetria da distribuição dos dados. No *box plot* são considerados os quartis e os limites da distribuição, permitindo uma visualização do posicionamento da distribuição na escala da variável (MEDRI, 2011).

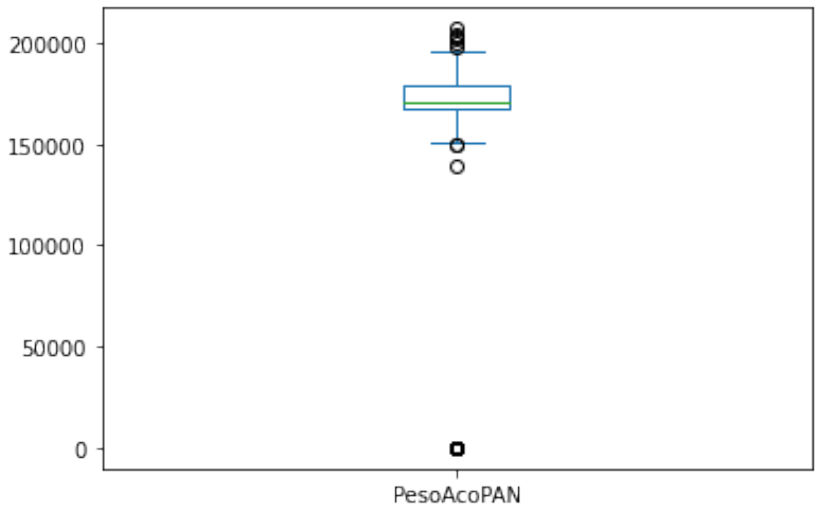


Figura 20 – *Box plot* peso da panela.

Foram encontradas informações faltantes na variável `PesoAcoPAN`, que é uma grandeza que deve ser considerada para análise de quaisquer informações relacionadas ao tratamento do aço líquido, pois identifica o volume de material a ser considerado para o modelo. Em conjunto com os especialistas do processo, concluiu-se que a melhor solução seria inserir as informações faltantes na base de dados utilizando a mediana das demais corridas existentes, conforme mostrado no Código 3.6.

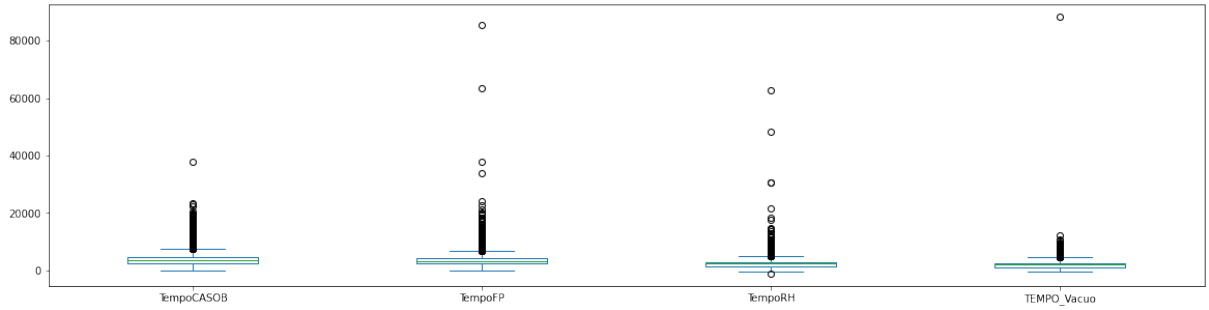


Figura 21 – *Box plot* tempos de processo.

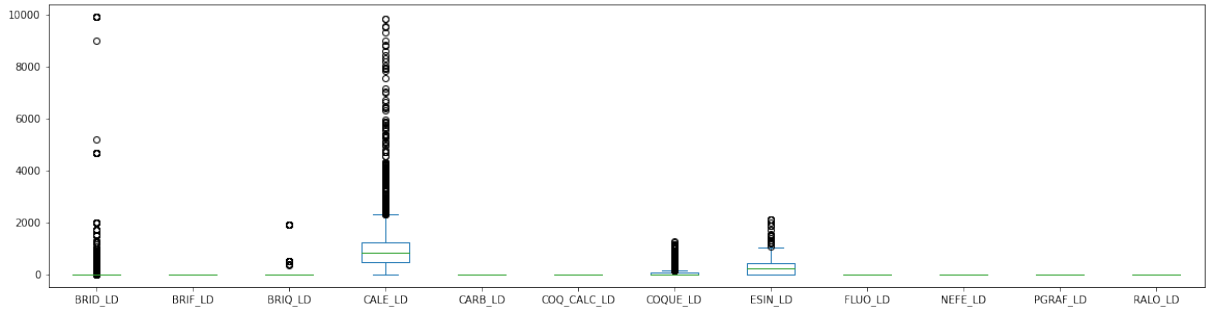


Figura 22 – *Box plot* consumo ligas no convertedor.

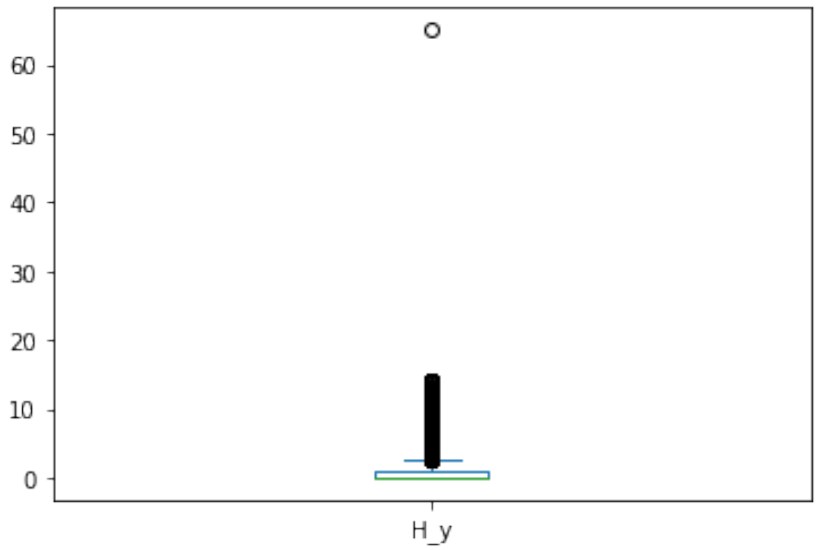


Figura 23 – *Box plot* hidrogênio medido.

Código 3.6 – Inclusão de valor na variável PesoAcoPAN cujo valor esteja zero.

```
#Dados de processo
#Incluir a mediana dos pesos de aço na panela
#por se tratar de um valor referência
SemPesoAcoPAN_mediana = df['PesoAcoPAN'].median()
df['PesoAcoPAN'].fillna(SemPesoAcoPAN_mediana, inplace=True)

#Substituir dados nulos por zero
df = df.fillna(0)
```

Algumas observações não possuíam medição da umidade relativa do ar (`UMIDADE_R`). Optou-se por remover essas corridas, conforme Código 3.7 e selecionar as corridas com medição de hidrogênio dentro da faixa >=2 e <=12, pois após todas as tratativas apresentadas, medições fora dessa faixa podem ser consideradas erros de medição.

Código 3.7 – Seleção de corridas que tiveram medição de umidade do ar e medição dentro da faixa normal do processo.

```
#Somente analisar corridas que tiveram medição de umidade relativa do ar
df = df[df['UMIDADE_R'] > 0]

#Somente analisar corridas que tiveram Hidrogênio medido dentro de uma
#faixa aceitável. Fora dessa faixa será erro de medição.
df = df[df['H_y'] <= 12]
df = df[df['H_y'] >= 2]
```

3.2.3 Remoção de *outliers*

As variáveis identificadas com valores distorcidos, que caracterizavam *outliers*, foram analisadas uma a uma juntamente com os especialistas do processo de aciaria. As mesmas mostraram valores discrepantes da realidade do processo, como por exemplo, o carbureto medido no forno panela, cujo limite máximo já identificado no processo foi de 150, e na base de dados constava uma medição com valor 2.400. Foram identificados tempos de processo registrados acima do máximo realizado e com valores negativos. As observações ruidosas foram ajustadas de acordo com a característica de cada variável, conforme mostram os Códigos 3.8 e 3.9 contem parte dos códigos utilizados para as correções.

Código 3.8 – Atualização dos limites para variáveis de composição química.

```
1
2  #CARB no Forno Panela
3  #Máximo 150
4  df.loc[(df['CARB_FP'] > 150), 'CARB_FP'] = 150
5  #AGRB no Forno Panela
6  #Máximo 2000
7  df.loc[(df['AGRB_FP'] > 2000), 'AGRB_FP'] = 2000
8
9  #FCAL no CASOB
10 #Máximo 300
11 df.loc[(df['FCAL_OB'] > 300), 'FCAL_OB'] = 300
12 #C_FIO no CASOB
13 #Máximo 1000
14 df.loc[(df['C_FIO_OB'] > 1000), 'C_FIO_OB'] = 1000
15
16 #AGRB
17 #Máximo 2500
18 df.loc[(df['AGRB_RH'] > 2500), 'AGRB_RH'] = 2500
19 #COQUE
20 #Máximo 500
21 df.loc[(df['COQUE_RH'] > 500), 'COQUE_RH'] = 500
22
23 #FECA no RH
24 #Máximo 40
25 df.loc[(df['FECA_RH'] > 40), 'FECA_RH'] = 40
26 #CASI
27 #Máximo 150
28 df.loc[(df['CASI_RH'] > 150), 'CASI_RH'] = 150
29 #FCAL no RH
30 #Máximo 150
31 df.loc[(df['FCAL_RH'] > 150), 'FCAL_RH'] = 150
```

Código 3.9 – Definição de limites aceitáveis para variáveis de tempo com outliers identificados.

```
1  #Limites mínimos para tempos de tratamento
2  df.loc[df['TempoCASOB'] < 0, 'TempoCASOB'] = 0
3  df.loc[df['TempoFP'] < 0, 'TempoFP'] = 0
4  df.loc[df['TempoRH'] < 0, 'TempoRH'] = 0
5  df.loc[df['TEMPO_Vacuo'] < 0, 'TEMPO_Vacuo'] = 0
6
7  #Limites máximos para tempos de tratamento
8  df.loc[df['TempoCASOB'] > 18000, 'TempoCASOB'] = 18000
9  df.loc[df['TempoFP'] > 18000, 'TempoFP'] = 18000
10 df.loc[df['TempoRH'] > 18000, 'TempoRH'] = 18000
11 df.loc[df['TEMPO_Vacuo'] > 18000, 'TEMPO_Vacuo'] = 18000
```

Os limites mínimo e máximo das variáveis peso da panela (`PesoAcoPAN`) foram atualizados de acordo com a capacidade real dos convertedores, conforme o Código 3.10. Os valores abaixo de 165.000 foram corrigidos para 165.000 e os valores acima de 175.000 foram corrigidos para 175.000.

Código 3.10 – Atualizando limites de processo.

```
#Limites peso panela
df.loc[df['PesoAcoPAN'] < 165000, 'PesoAcoPAN'] = 165000
df.loc[df['PesoAcoPAN'] > 175000, 'PesoAcoPAN'] = 175000
```

A fim de reduzir a dimensão da base de dados, as ligas de mesma composição química utilizadas no processo foram somadas e as variáveis de cada liga individual foram removidas da base de dados deixando somente os somatórios das ligas, como as variáveis `CALE_LD`, `CALE_FP` e `CALE_OB` que tratam da quantidade de cal utilizada no processo de convertedor, forno panela e CASOB, respectivamente. Estas variáveis foram somadas na variável `CALE` e depois foram removidas da base conforme apresenta o Código 3.11. Algumas variáveis cujas ligas são adicionadas em apenas um processo tiveram alteração de nome.

Código 3.11 – Agregação de variáveis ligadas a materiais similares adicionados nos processos.

```
#Somando as ligas após tratamento
df['CALE'] = df['CALE_LD'] + df['CALE_FP'] + df['CALE_OB']
df['ESIN'] = df['ESIN_LD']
df['BRID'] = df['BRID_LD']
df['BRIQ'] = df['BRIQ_LD'] + df['BRIQ_FP'] + df['BRIQ_OB'] + df['BRIQ_RH']
df['CARB'] = df['CARB_FP'] + df['CARB_OB']
df['C_FIO'] = df['C_FIO_FP'] + df['C_FIO_OB']
df['FECA'] = df['FECA_FP'] + df['FECA_OB'] + df['FECA_RH']
df['FLUO'] = df['FLUO_FP']
df['FCAL'] = df['FCAL_FP'] + df['FCAL_OB'] + df['FCAL_RH']
df['NEFE'] = df['NEFE_FP']
df['COQUE'] = df['COQUE_LD'] + df['COQUE_RH']
df['CASV'] = df['CASV_FP'] + df['CASV_OB'] + df['CASV_RH']
df['AGRB'] = df['AGRB_FP'] + df['AGRB_OB'] + df['AGRB_RH']
df['CASI'] = df['CASI_RH']
df['NBRQ'] = df['NBRQ_RH']
df['CAL_INJ_OB'] = df['CAL_INJ_CAS'] + df['CASI_INJ_CAS']

Col_drop = ['CALE_LD','CALE_FP','CALE_OB','ESIN_LD','BRID_LD','BRIQ_LD',
'BRIQ_FP','BRIQ_OB','BRIQ_RH','CARB_FP','CARB_OB','C_FIO_FP','C_FIO_OB',
'COQ_CALC_OB', 'FECA_FP', 'FECA_OB', 'FECA_RH', 'FLUO_FP', 'FCAL_FP',
'FCAL_OB','FCAL_RH','NEFE_FP','COQUE_RH','COQUE_LD','CASV_FP','CASV_OB',
'CASV_RH','AGRB_FP','AGRB_OB','AGRB_RH','CSAL_OB','CASI_RH','NBRQ_RH',
'ALGR', 'BLOG', 'COQC', 'AGRB', 'CAL_INJ_CAS' ,'CASI_INJ_CAS']

df = df.drop(Col_drop, axis=1)
```

A medida que as correções foram realizadas, a base de dados foi reavaliada junto aos especialistas do processo, com o intuito de melhorar os dados para a entrada no modelo. Além das correções já apresentadas, foram removidas todas as corridas que passaram no RH e tiveram tempo de vácuo menor ou igual a zero e maior que 10.000. Também foram removidas as corridas que passaram no RH, cujo tempo de vácuo foi maior que 420, (`TEMPO_Vacuo` > 420) e apresentavam teor de hidrogênio maior que 4 ppm. Segundo os especialistas, essa condição não condiz com a realidade do processo, pois em vácuo os gases são removidos do aço líquido.

Da mesma forma foram removidas todas as corridas que passaram no RH, cujo tempo de vácuo foi menor que 420 (`TEMPO_Vacuo` < 420) e apresentavam teor de hidrogênio maior que 8 ppm. Todas as situações mostradas se tratavam de erros de medição. Os ajustes citados foram realizados no Código 3.12.

Código 3.12 – Remoção de corridas com base no RH, tempo de vácuo e teor de hidrogênio medido.

```
#Substituir valores negativos por 0 para "TEMPO_Vacuo" (erro de leitura)
df.loc[df['TEMPO_Vacuo'] < 0, 'TEMPO_Vacuo'] = 0
#df[df['TEMPO_Vacuo'] < 0] = 0

#Remover todas as corrias que passaram no RH e tiveram tempo de Vácuo
#maior que 10000 (erro de medição)
df = df.drop(df[(df['RH'] == 1) & (df['TEMPO_Vacuo'] > 10000)].index)

#Remover todas as corrias que passaram no RH e não tiveram tempo de
#Vacuo (erro de medição)
df = df.drop(df[(df['RH'] == 1) & (df['TEMPO_Vacuo'] <= 0)].index)

#Remover todas as corrias que passaram no RH e tiveram tempo mínimo de
#vacuo e possuem teor de Hidroigênio altos (erro de medição)
df = df.drop(df[(df['RH'] == 1) & (df['TEMPO_Vacuo'] > 420) &
(df['H_y'] > 4)].index)

#Remover todas as corrias que passaram no RH e tiveram tempo mínimo de
#vacuo e possuem teor de Hidroigênio altos (erro de medição)
df = df.drop(df[(df['RH'] == 1) & (df['TEMPO_Vacuo'] < 420) &
(df['H_y'] > 8)].index)
```

Ainda em alinhamento com os especialistas do processo, foram removidas as corridas que passaram no RH, seguiram para outro processo de refino antes do lingotamento e possuíam teor de hidrogênio maior que 8 ppm, pois uma vez que processado no RH, o valor de hidrogênio no aço tende a ser menor que 4 ppm. Também foram removidas as corridas que não passaram no RH e tiveram alguma adição de ligas nos refinos secundários e possuem teor de hidrogênio baixo (menor que 3 ppm), pois esse teor de hidrogênio é alcançado quando o aço líquido é processado pelo RH, como mostra o Código 3.13.

Código 3.13 – Correção de erros de registro das variáveis relacionadas ao RH.

```
#Remover corridas que passaram no RH, nos refinos,
#tiveram alguma adição nos refinos secundários e
#possuem teor de Hidroigênio
#muito altos (erro de medição)
df = df.drop(df[(df['RH'] == 3) & (df['H_y'] > 8)].index)

#Remover todas as corrias que não passaram no RH e tiveram alguma
#adição nos refinos secundários e possuem teor de Hidroigênio
#baixos (erro de medição)
df = df.drop(df[(df['RH'] == 2) & (df['H_y'] < 3)].index)
```

As estatísticas descritivas neste estágio mostraram uma base de dados mais coerente com a realidade do processo e com menor incidência de *outliers* relacionados a erros de medição e lançamento dos dados, o que pode ser observado nas Figuras 24, 25, 26 e 27.

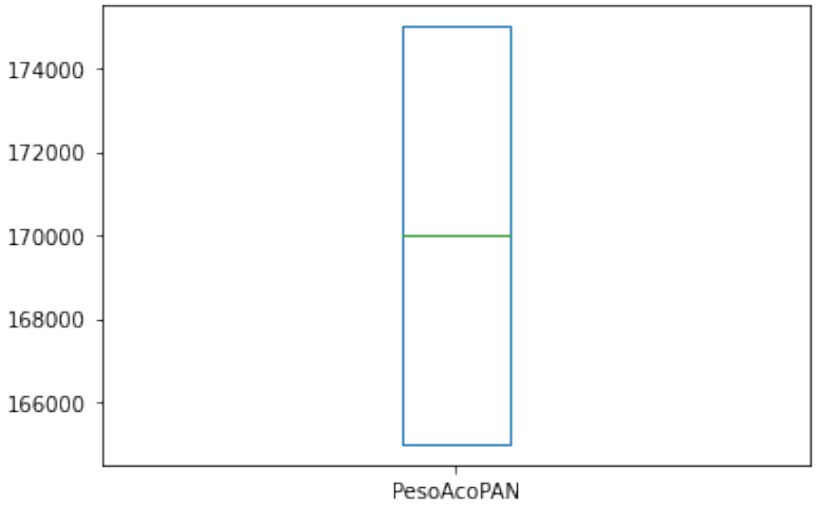


Figura 24 – *Box plot* peso da panela após tratamento dos dados.

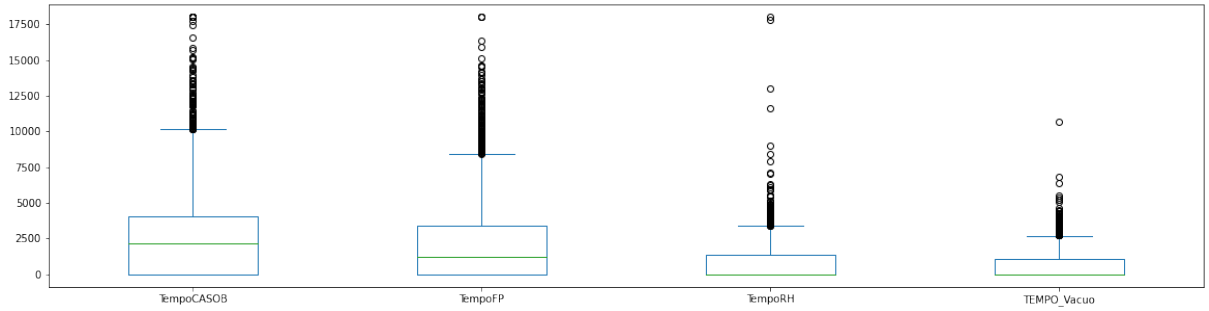


Figura 25 – *Box plot* tempos de processo após tratamento dos dados.

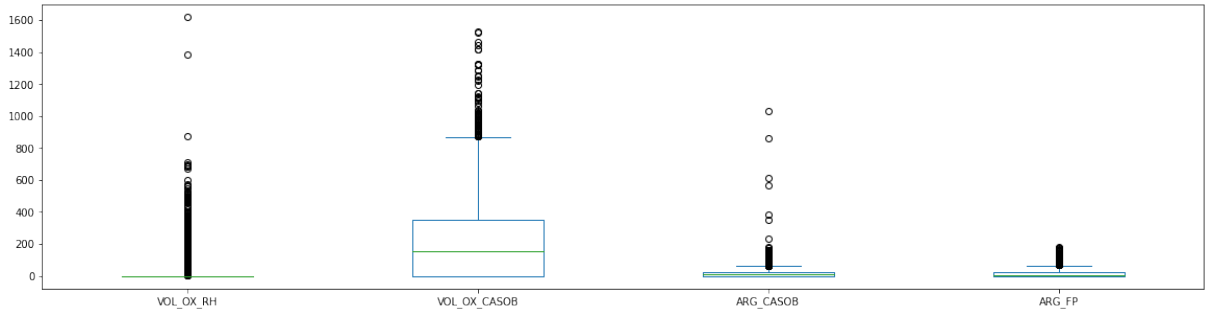


Figura 26 – *Box plot* consumo ligas no convertedor após tratamento dos dados.

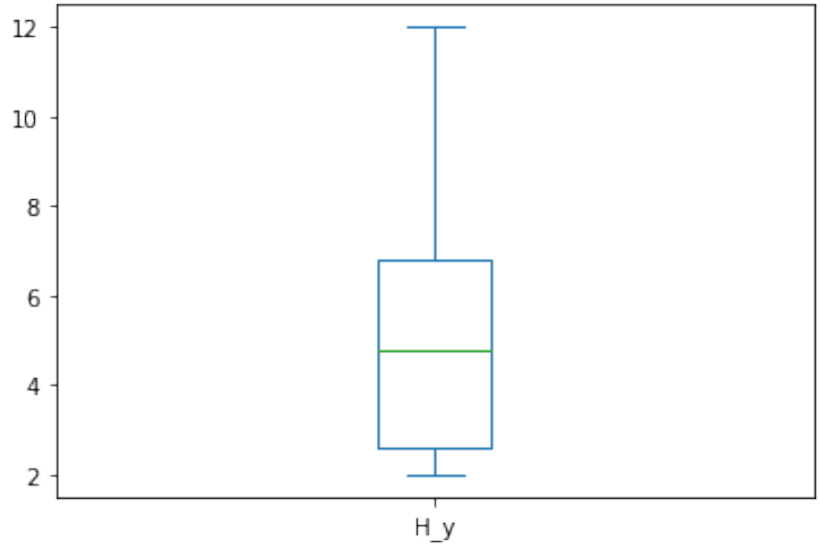


Figura 27 – *Box plot* hidrogênio medido após tratamento dos dados.

3.2.4 Categorização e padronização dos dados

Converter as diversas categorias em números é uma estratégia chamada de *label encoding* (BRANDÃO et al., 2022), técnica utilizada neste trabalho para converter as variáveis referentes ao tipo do aço (`Familia`) e rota da corrida nos processos de refino na aciaria (`ROTA_CORRIDA`). Essas variáveis foram categorizadas através de uma função do Python exibida no Código 3.14, pois se tratavam de informações relevantes para a avaliação do modelo.

Código 3.14 – Categorização de dados (*label encoding*).

```
#Tratar dados categóricos
df['Familia']= df['Familia'].astype('category').cat.codes
df['ROTA_CORRIDA']= df['ROTA_CORRIDA'].astype('category').cat.codes
```

Também foi utilizada a técnica de padronização dos valores da base de dados, a fim de ajustar os dados a uma escala uniforme. Esta escolha foi influenciada pela documentação da biblioteca `sklearn`, pois menciona que normalizar entradas é operação comum para classificação de texto ou agrupamento. Foi utilizado o método `StandardScaler` da biblioteca `sklearn`, conforme apresentado no Código 3.15.

Código 3.15 – Padronização dos dados.

```
from sklearn.preprocessing import StandardScaler

sc = StandardScaler() # cria objeto da classe
X_padr = sc.fit_transform(df1.iloc[:,0:(Col-1)]) # cria matriz com valores
padronizados
df1_padr = pd.DataFrame(X_padr, columns= df1.iloc[:,0:(Col-1)].columns)
# cria novo df
df1_padr.head()

df1_padr.plot(kind = 'box',figsize=(50,8)) #boxplot
```

No código acima também foi gerado um *box plot* com todas as variáveis após a padronização. Como exemplo, algumas variáveis estão nas Figuras 28 e 29.

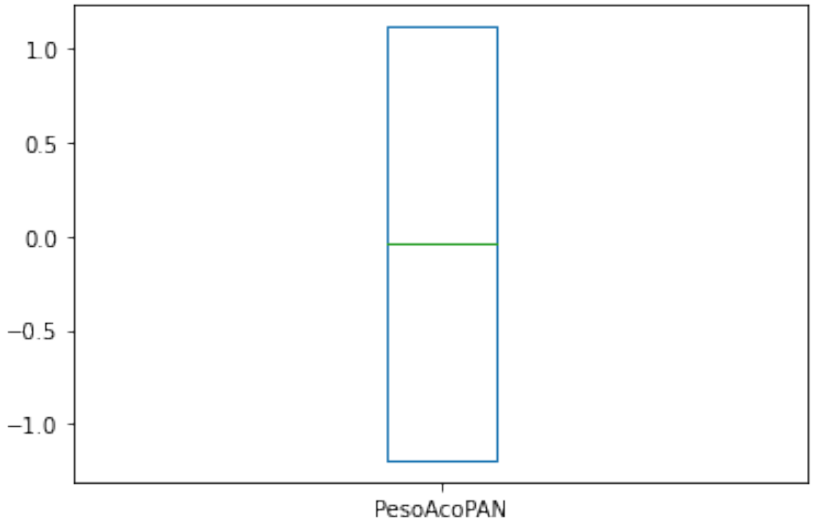


Figura 28 – *Box plot* peso da panela após padronização.

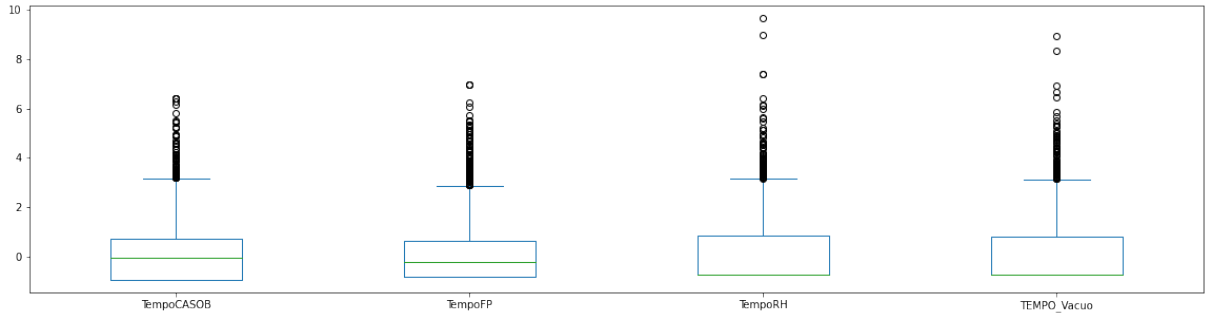


Figura 29 – *Box plot* tempos de processo após padronização.

Foi ajustado um modelo de regressão linear da biblioteca Sklearn, o `Linear Regression`, a fim de verificar o coeficiente de correlação R^2. O modelo de regressão retornou R^2 de 75,3% entre as variáveis da base de dados e a variável resposta.

Código 3.16 – Modelo de regressão linear com todas as variáveis.

```
#Ajustar um modelo de regressão com todas as variáveis.

from sklearn.feature_selection import SequentialFeatureSelector
from sklearn.linear_model import LinearRegression

y = df1.iloc[:,(Col-1)].to_numpy() # mudar a estrutura de dados para
#um array do numpy
X = df1_padr.iloc[:,0:(Col-1)].to_numpy()
reg = LinearRegression().fit(X, y)
r2 = reg.score(X, y)

print(f'Correlação da base para treinamento: {r2:.3f}')

> Correlação da base para treinamento:  0.753
```

3.3 Seleção de atributos e redução de dimensionalidade

Com o objetivo de reduzir a dimensão da base de dados, foi utilizada uma técnica computacional neste estudo, conhecida como método *forward feature selection* (FFS) em conjunto a um algoritmo de regressão linear da biblioteca `sklearn`. O algoritmo `LinearRegression` ajusta um modelo linear com coeficientes $w = (w_1, w_2, \ldots, w_n)$ para cada variável e minimiza a soma do erros quadrados entre os alvos observados no conjunto de dados e os alvos previstos pela aproximação linear (PEDREGOSA et al., 2011).

O método utilizado selecionou 26 variáveis para composição do modelo utilizando os parâmetros padrão. O Código 3.17 apresenta a aplicação do método FFS à base de dados do presente trabalho.

Código 3.17 – Aplicação do método *forward feature selection*.

```
#Sera utilizado um df com os dados padronizados
# usando scikit-learn
y = df1.iloc[:,(Col-1)].to_numpy()
X = df1_padr.iloc[:,0:(Col-1)].to_numpy()
reg = LinearRegression().fit(X, y)

#Metodo Forward:
sfs_forward = SequentialFeatureSelector(reg, direction='forward').fit(X, y)
sfs_fwd_nomes = [df1.columns[i]  for i, x in enumerate(sfs_forward.
get_support()) if x]

print(">>> Saida:")
print(f">>> Variaveis selecionadas FSS: {sfs_fwd_nomes} R2:
{reg.score(X,y)}")

print(">>> Equações:")
mdvars = ''
for vi in range(len(sfs_fwd_nomes)):
  mdvars += f'{reg.coef_[vi]:.2f}{sfs_fwd_nomes[vi]}'
  if vi < len(sfs_fwd_nomes)-1:
    if reg.coef_[vi+1]>0:
        mdvars += ' + '

print(f'>>> Equação da reta FSS : y ~ {mdvars}, R^2 = {reg.
score(X,y):.3f}')
```

Ao fim de todo o processo de tratamento, a base de dados foi reduzida de 256 variáveis e 51.684 observações para 52 variáveis e 8.423 observações. Após a execução da etapa de seleção de atributos, a base de dados foi reduzida para 26 variáveis, descritas na Tabela 1, e 8.423 observações, uma redução significativa nos dados, porém, extremamente importante para alcance de resultados mais precisos no modelo de rede neural aplicado.

3.3.1 *Heat map*

O grau de relação entre duas variáveis é medido através dos coeficientes de associação ou correlação. Tais medidas demonstram, numericamente, a dependência entre duas variáveis. A representação é feita no intervalo de -1 a 1, sendo que quanto mais próximo de 1, maior é a correlação positiva existente entre as variáveis e quanto mais próximo de -1, maior é a correlação negativa entre as variáveis.

Uma forma muito usual de representar visualmente a relação de dependência entre as variáveis é o gráfico *heat map*. No presente trabalho foi realizada a construção do gráfico *heat map* a fim de avaliar a correlação entre as variáveis, conforme Código 3.18 e Figura 30. O resultado mostrou alta correlação entre as variáveis de entrada e a variável resposta, o que comprova o alto score retornado pelo modelo de regressão.

Tabela 1 – Variáveis selecionadas para o modelo.

Nome da variável	**Descrição**
UMIDADE_R	Umidade do ar
Familia	Família do aço
RH	Tratamento RH
TempoCASOB	Tempo de CASOB
TempoFP	Tempo de FP
TEMPO_Vacuo	Tempo de Vácuo
VOL_OX_RH	Volume oxigênio RH
VOL_OX_CASOB	Volume oxigênio CASOB
ARG_FP	Volume argônio FP
CAL_INJ_FP	Cal Micro no FP
ALGO	Alumínio gotão
FEV	Ferro vanadio
MNAC	Ferro manganes alto carbono
MNBP	Fe manganes baixo P
SCCO	Sucata de cobre
SIBR	Fe silicio baixo C 10-30 mm
SIGL	Fe silicio no gusa l¡quido
CALE	Cal
ESIN	Escoria sintetica
CARB	Carbureto SI
C_FIO	Carbono fio
FCAL	FeCaAl em fio (vertical)
COQUE	Coque fino
CASV	Ca silicio fio vertical
CAL_INJ_OB	Cal Micro no CASOB
H_y	Hidrogênio medido Hydris

Código 3.18 – Geração de *heat map*.

```
from matplotlib import pyplot
import seaborn as sns
dims = (15,15) # tamanho da figura
pyplot.figure(figsize=dims)
corr = df1_padr.corr() #Matriz de valores com a correlacao
sns.heatmap(corr, xticklabels=corr.columns.values, yticklabels=
corr.columns.values, vmin=-1, vmax=1, center= 0, cmap= 'vlag')
```

É possível observar no gráfico de heatmap que as variáveis `RH` e `TempoCASOB` tem uma correlação positiva com a variável resposta `H_y` e as variáveis `TEMPO_Vacuo` e `SIBR` tem correlação negativa com a variável resposta `H_y`. O gráfico de *heatmap* é bom direcionador para indicar quais variáveis devem ser exploradas pelo modelo de rede neural.

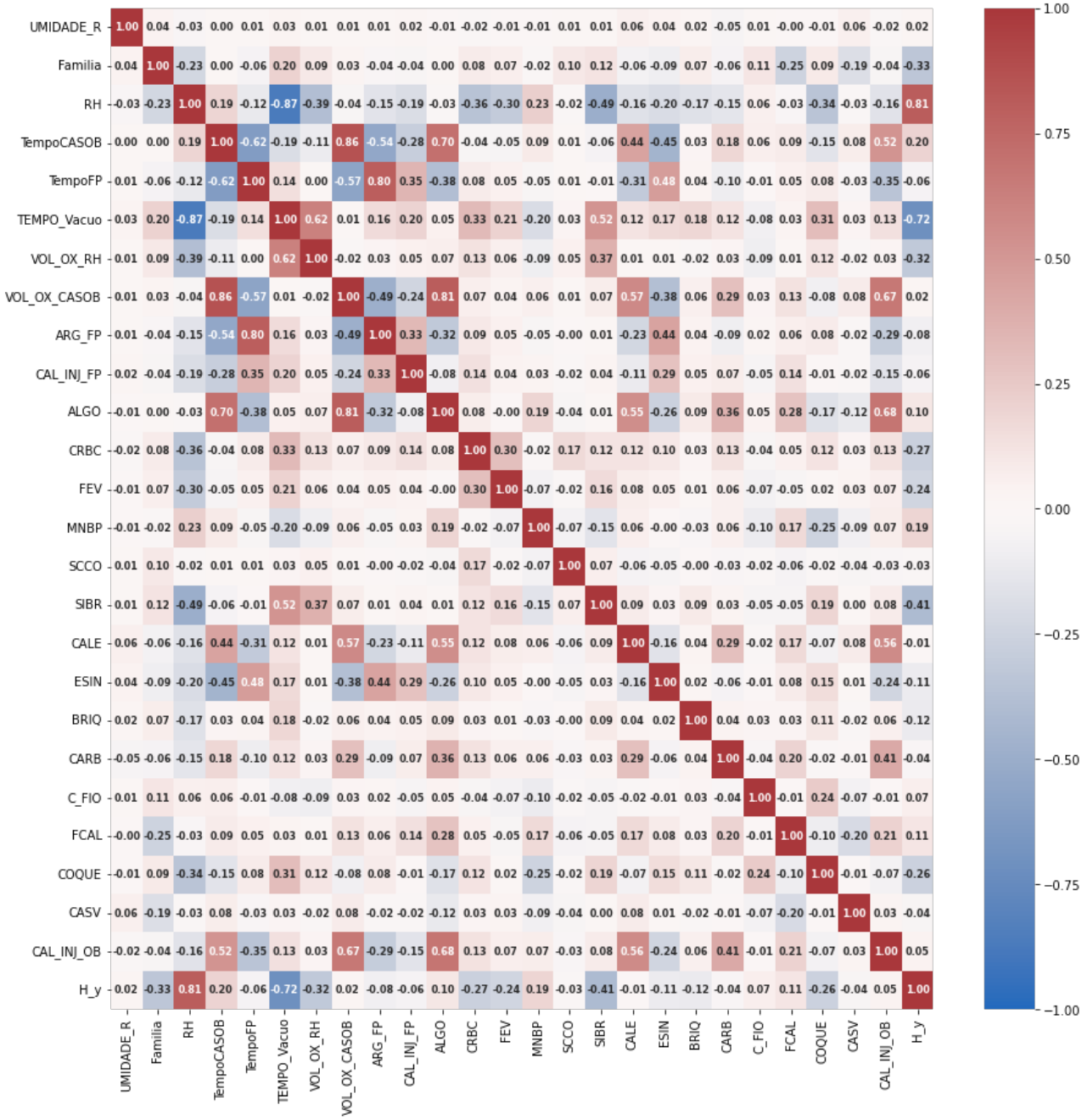


Figura 30 – *Heat map* da base de dados.

4 Resultados

4.1 Separação da base de dados

No presente trabalho, a base de dados foi dividida em 70% para treino e 30% para teste, conforme Código 4.1.

Código 4.1 – Divisão da base em treino e teste.

```
1
2  y = df1.iloc[:,(df1.shape[1]-1)]
3  x = df1_padr.iloc[:,0:(Col-1)]
4
5  #Carregar base para simulação
6  arq = df1_padr.copy()
7
8  #Split
9  df_x_train, df_x_test, df_y_train, df_y_test = train_test_split(x,y,
10 test_size = 0.3, random_state = 0)
```

4.2 Calibragem de hiper-parâmetros

Os hiper-parâmetros definidos para rede neural deste trabalho foram ajustados em função de testes gradativos realizados no modelo utilizado. Inicialmente foram definidas 50 épocas. Nesta etapa de testes foi observado que ao aumentar as épocas a rede convergiu para um melhor resultado, ou seja, a curva de aprendizado melhorou. Tal fato incentivou o aumento das épocas até alcançar o modelo final com 1.000 épocas, onde o erro médio na saída se estabilizou.

As funções de ativação das camadas também foram ajustadas em decorrência dos testes. A função de perda *Mean Squared Error* - MSE e a função de ativação da camada de saída "Linear", foram escolhidas desde o primeiro teste, por se tratar de um problema de regressão ao qual a rede foi submetida a resolver. Essa estratégia vem de encontro à literatura utilizada e referenciada neste trabalho.

A quantidade de neurônios em cada camada também foi ajustada gradativamente à medida em que os testes foram realizados. Um comportamento identificado foi que ao aumentar a quantidade de camadas e de neurônios, aumentava também o tempo de processamento, porém, também foi observada uma melhora no erro médio absoluto.

Informações adicionais que auxiliaram nesta configuração podem ser encontradas no trabalho de Furtado (2019).

Para avaliar o desempenho da rede foram feitos alguns ensaios alterando o número de camadas, número de neurônios e, consequentemente, as funções de ativação. Foram exploradas algumas possibilidades e avaliado o desempenho de cada configuração para servir de critério de escolha da melhor solução. A seguir estão descritas três configurações testadas.

4.2.1 RNA - Configuração 1

Na configuração apresentada no Código 4.2 e Tabela 2, foi montada uma rede com 4 camadas. Já nos primeiros treinos a rede apresentou erro médio abaixo de 1 ponto, indicando que a configuração proposta trouxe um resultado promissor. Foram feitos cinco treinamentos para avaliação dos resultados. A rede demandou em média 8 minutos e 40 segundos para o treinamento. A Figura 31 ilustra um dos treinamentos do modelo e a Tabela 3 mostra os resultados obtidos com esta configuração.

Código 4.2 – RNA - configuração 1.

```
#Qtd de variáveis de entrada
n_variaveis = x_train.shape[1]
print('Input para a rede: ', n_variaveis)

model = Sequential()

#Teste 1
model.add(Dense(256, activation='relu', input_shape=(n_variaveis,))),
model.add(Dense(128, activation='sigmoid')),
model.add(Dense(64, activation='tanh')),
model.add(Dense(1, activation='linear'))
model.summary()

model.compile(optimizer='Adadelta', loss='mse', metrics=['mae'])

history = model.fit(x_train, y_train, epochs=1000, validation_data=
(x_test, y_test), verbose=2)
plt.plot(history.history['mae'])
plt.plot(history.history['val_mae'])
plt.title('Erro Médio Absoluto (MAE)')
plt.ylabel('Erro Médio Absoluto')
plt.xlabel('Epoch')
plt.legend(['treinamento', 'validação'], loc='upper left')
plt.show()

erro_medio = history.history['val_mae'][(len(history.
history['val_mae']))-1]

print('\nErro médio: ', erro_medio)
```

Mais testes foram realizados na tentativa de melhorar o desempenho do modelo e o resultado da rede proposta.

Tabela 2 – Variáveis *RNA - configurações do teste 1*.

Hiperparametro	**Valor**
Otimizador	Adadelta
Métrica	MAE
Função de perda	MSE
Épocas	1000
Variáveis	25
Camada de entrada	256 neurônios - Relu
Camada intermediaria C2	128 neurônios - Sigmoid
Camada intermediaria C3	64 neurônios - Tanh
Camada de Saída	1 neurônio - Linear
Conjunto de teste	30%

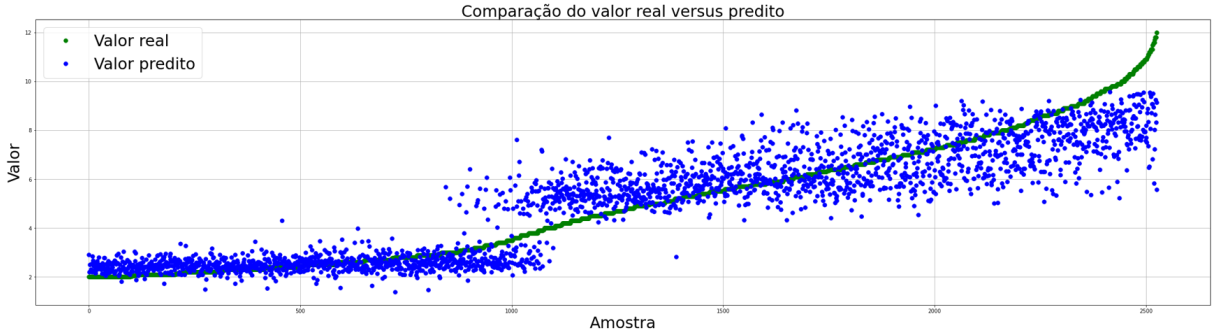


Figura 31 – Gráfico valores reais X valores preditos no teste 1.

Tabela 3 – Resultados apresentados no Teste 1.

Métrica	**Valor**
Erro médio	0,781
Desvio padrão do erro	0,002
Total de amostas	2.527
Critério para aceitação	-2 < erro > +2 pontos
Total de erros não aceitáveis	58
Percentual de erros	2,29%
Percentual de acertos	97,71%

4.2.2 RNA - Configuração 2

A rede com os hiper-parâmetros do Código 4.3 e Tabela 4 foi composta de 7 camadas, sendo a primeira camada com 256 neurônios, a segunda com 128 neurônios, a terceira com 128 neurônios, a quarta com 128 neurônios, a quinta com 128 neurônios, a sexta com 64 neurônios e a sétima camada com 1 neurônio. Assim como na rede anterior, foram feitos cinco treinamentos para análise dos resultados. Apresentou resultado melhor em relação à rede anterior, com erro médio de 0,759 pontos e tempo de processamento de 11 minutos e 52 segundos. A Figura 32 ilustra um dos treinamentos do modelo e a Tabela 5 mostra os resultados obtidos com esta configuração.

Código 4.3 – RNA - configuração 2.

```
#Qtd de variáveis de entrada
n_variaveis = x_train.shape[1]
print('Input para a rede: ', n_variaveis)

model = Sequential()

#Teste 2
model.add(Dense(256, activation='relu', input_shape=(n_variaveis,))),
model.add(Dense(128, activation='sigmoid')),
model.add(Dense(128, activation='relu')),
model.add(Dense(128, activation='sigmoid')),
model.add(Dense(128, activation='relu')),
model.add(Dense(64, activation='linear')),
model.add(Dense(1, activation='linear'))
model.summary()

model.compile(optimizer='Adadelta', loss='mse', metrics=['mae'])

history = model.fit(x_train, y_train, epochs=1000, validation_data=
(x_test, y_test), verbose=2)
plt.plot(history.history['mae'])
plt.plot(history.history['val_mae'])
plt.title('Erro Médio Absoluto (MAE)')
plt.ylabel('Erro Médio Absoluto')
plt.xlabel('Epoch')
plt.legend(['treinamento', 'validação'], loc='upper left')
plt.show()

erro_medio = history.history['val_mae'][(len(history.
history['val_mae']))-1]

print('\nErro médio: ', erro_medio)
```

Seguindo na tentativa de melhorar o desempenho do modelo aplicado, foram realizados mais testes com uma configuração de rede mais robusta.

Tabela 4 – Variáveis *RNA - configurações do teste 2*.

Hiperparametro	**Valor**
Otimizador	Adadelta
Métrica	MAE
Função de perda	MSE
Épocas	1000
Variáveis	25
Camada de entrada	256 neurônios - Relu
Camada intermediaria C2	128 neurônios - Sigmoid
Camada intermediaria C3	128 neurônios - Relu
Camada intermediaria C4	128 neurônios - Sigmoid
Camada intermediaria C5	128 neurônios - Relu
Camada intermediaria C6	128 neurônios - Linear
Camada de Saída	1 neurônio - Linear
Conjunto de teste	30%

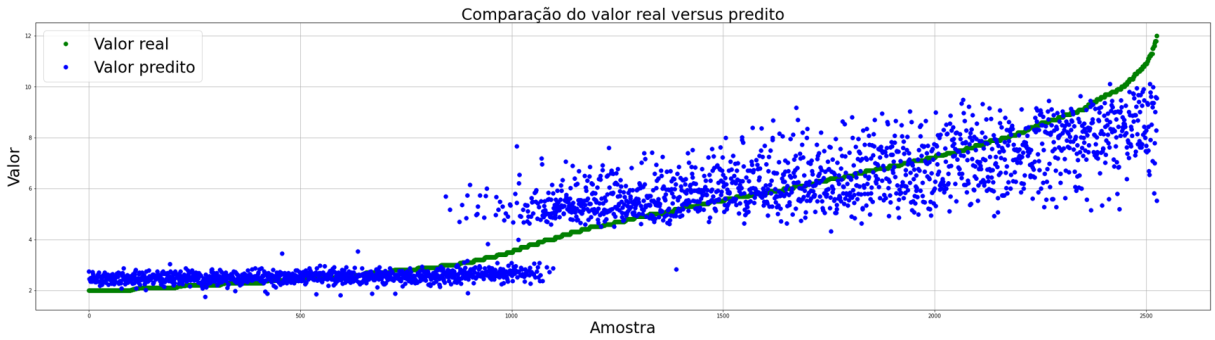


Figura 32 – Gráfico valores reais X valores preditos do Teste 2.

Tabela 5 – Resultados apresentados no Teste 2.

Métrica	**Valor**
Erro médio	0,758
Desvio padrão do erro	0,002
Total de amostas do teste	2.527
Critério para aceitação	-2 < erro > +2 pontos
Total de erros não aceitáveis	57
Percentual de erros	2,25%
Percentual de acertos	97,75%

4.2.3 RNA - Configuração 3

A rede descrita no Código 4.4 e Tabela 6 é composta de 6 camadas com um número de neurônios duplicados em relação à configuração anterior. Esta configuração aumentou o tempo de treinamento para 1 hora e 40 minutos. Foram feitos cinco treinamentos na rede para avaliação dos resultados. A configuração proposta não apresentou melhora significativa no resultado final. Silva (2022) em seu trabalho, menciona que aumentar o número de neurônios e camadas deixa a rede mais robusta, consome mais recursos computacionais, no entanto não garante a melhora nos resultados, além de aumentar o risco de gerar *overfit*. A Figura 33 ilustra um dos treinamentos do modelo e a Tabela 7 mostra os resultados obtidos com esta configuração.

Código 4.4 – RNA - configuração 3.

```
#Qtd de variáveis de entrada
n_variaveis = x_train.shape[1]
print('Input para a rede: ', n_variaveis)

model = Sequential()

#Teste 3
model.add(Dense(1024, activation='relu', input_shape=(n_variaveis,))),
model.add(Dense(512, activation='sigmoid')),
model.add(Dense(512, activation='tanh')),
model.add(Dense(512, activation='sigmoid')),
model.add(Dense(512, activation='tanh')),
model.add(Dense(1, activation='linear'))
model.summary()

model.compile(optimizer='Adadelta', loss='mse', metrics=['mae'])

history = model.fit(x_train, y_train, epochs=1000, validation_data=
(x_test, y_test), verbose=2)
plt.plot(history.history['mae'])
plt.plot(history.history['val_mae'])
plt.title('Erro Médio Absoluto (MAE)')
plt.ylabel('Erro Médio Absoluto')
plt.xlabel('Epoch')
plt.legend(['treinamento', 'validação'], loc='upper left')
plt.show()

erro_medio = history.history['val_mae'][(len(history.
history['val_mae']))-1]

print('\nErro médio: ', erro_medio)
```

Com os resultados dos testes, optou-se por seguir os estudos com segunda configuração de rede neural, a de 7 camadas, apresentada no Código 4.3.

Tabela 6 – Variáveis *RNA - configurações do teste 3*.

Hiperparametro	**Valor**
Otimizador	Adadelta
Métrica	MAE
Função de perda	MSE
Épocas	1000
Variáveis	25
Camada de entrada	256 neurônios - Relu
Camada intermediaria C2	512 neurônios - Sigmoid
Camada intermediaria C3	512 neurônios - Tanh
Camada intermediaria C4	512 neurônios - Sigmoid
Camada intermediaria C5	512 neurônios - Tanh
Camada de Saída	1 neurônio - Linear
Conjunto de teste	30%

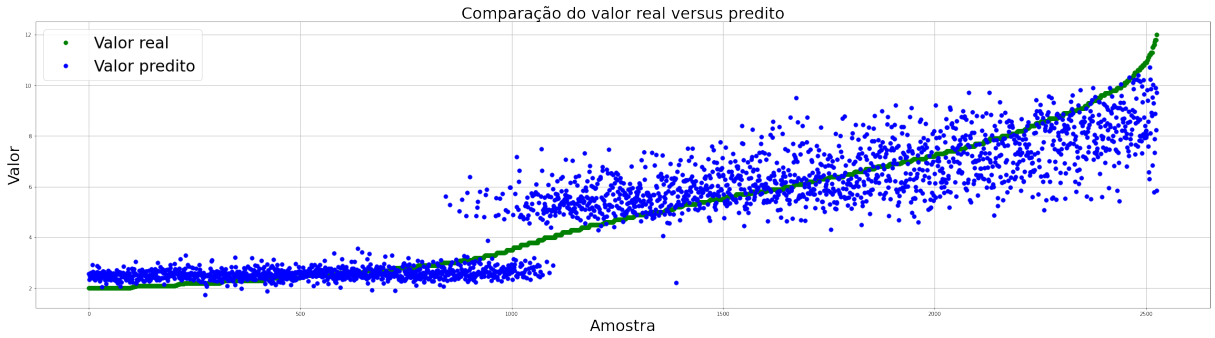


Figura 33 – Gráfico valores reais X valores preditos no teste 3.

Tabela 7 – Resultados apresentados no Teste 3.

Métrica	**Valor**
Erro médio	0,757
Desvio padrão do erro	0,003
Total de amostas do teste	2.527
Critério para aceitação	-2 < erro > +2 pontos
Total de erros não aceitáveis	57
Percentual de erros	2,25%
Percentual de acertos	97,75%

4.3 Validação do modelo

O parâmetro de erro de referência para as áreas técnica e operacional do lingotamento das aciarias da Usiminas Ipatinga é + ou - 2 pontos. Assim, o critério de validação do modelo proposto neste estudo obedece a mesma referência.

Para validação do modelo criado neste trabalho foi utilizada a técnica de avaliação do MAE, utilizando os valores preditos em comparação aos valores reais de medição contidos na base de dados. O erro médio obtido foi de 0,76 pontos, que após avaliação junto aos especialistas do processo de aciaria, foi considerado satisfatório em relação ao critério de aceitação do resultado esperado.

O erro médio alcançado e as demais saídas do modelo foram apresentados na seção anterior. Foi criado um novo *dataframe* com o resultado da medição predita pelo modelo em relação à medição real para as observações e os dados foram apresentado em um gráfico de dispersão, conforme mostrado na Figura 34.

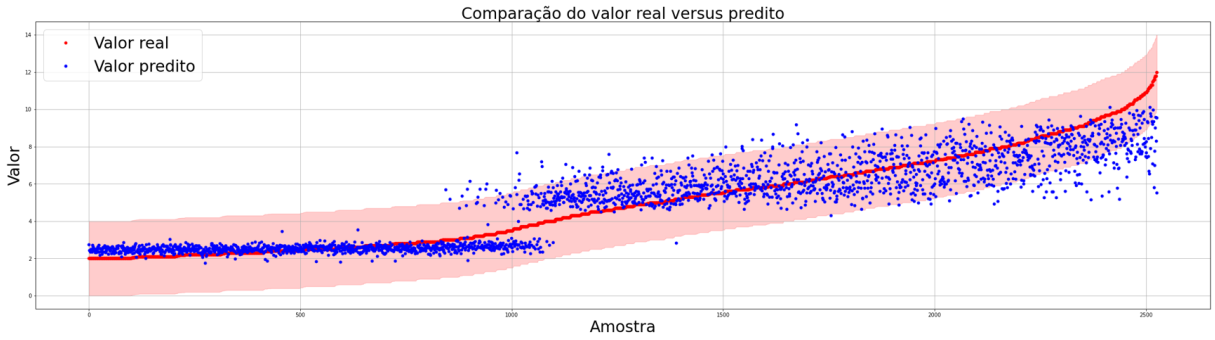


Figura 34 – Gráfico de dispersão considerando o critério de aceitação do processo (RNA-configuração 2).

Como o critério de validação é um erro de mais ou menos 2 (dois) pontos, margem aceita pelo processo de lingotamento, é possível observar no gráfico que o modelo teve bom desempenho, pois a dispersão dos pontos de predição acompanha a linha referente aos dados reais. Ao aplicar a margem aceitável aos valores preditos, foi observado que somente 2,25% dos dados ficaram fora da margem aceitável, conforme mostrado na Tabela 5, apresentada na sessão anterior. Os erros não aceitáveis consistem nos valores retornados pelo modelo que extrapolam o range de + ou - 2 ppm em relação aos valores reais medidos, utilizados para validar o modelo.

No dia 28/11/2022 ocorreu um acidente operacional devido a um rompimento de pele na corrida 553738. O hidrogênio medido pelo hydris em campo foi de 9,7 ppm. A fim de avaliar o modelo, os dados dessa corrida foram processados pela rede neural. O valor de hidrogênio predito foi de 7,6 ppm. Considerando apenas o erro médio da rede de 0,76 ppm, o valor do hidrogênio poderia chegar a 8,4 ppm. Conforme padrão operacional do lingotamento, quando o teor de hidrogênio da corrida é maior que 8 ppm devem ser feitas alterações no procedimento operacional, visando garantir a segurança das pessoas e do equipamento. As alterações são definidas de acordo com o teor de hidrogênio medido e são imperativas em relação às regras gerais de operação.

5 Considerações Finais

A base de dados utilizada apresenta alta correlação entre as variáveis de entrada e a variável de resposta teor de hidrogênio, que é o objetivo do estudo. Foi compreendido que as condições operacionais e a coleta de dados dos processos de refino secundário são as principais limitações para o bom funcionamento da RNA proposta.

O modelo gerado respondeu satisfatoriamente com um erro médio de 0,76 ppm, atendendo as expectativas do público técnico e operacional da área de aciaria. A implantação do modelo no processo para determinação do teor de hidrogênio do produto em tempo real precisa de baixo investimento. O objetivo da predição em tempo real é oferecer suporte à equipe operacional do lingotamento para que a mesma possa se preparar antes de receber um material com alto teor de hidrogênio, adaptando suas rotinas de produção.

De forma geral, os resultados obtidos neste trabalho são aplicáveis para a aciaria da Usiminas e demais empresas com processo semelhante como ferramenta de apoio à decisão, porém, devem ser consideradas as particularidades de cada processo, que podem gerar alterações nos resultados do modelo proposto.

5.1 Trabalhos futuros

As ligas adicionadas para produzir diferentes tipos de aço podem contribuir de formas diferentes para a inclusão de hidrogênio, portanto, é necessário refinar o treinamento do modelo adaptando o mesmo a cada família de aço, a fim de construir uma solução otimizada para uma qualidade específica do material.

Neste trabalho foram utilizados dados históricos de processo, comparando os valores preditos pelo modelo com os valores reais medidos no lingotamento. Para que seja efetivamente aplicado, é necessário implantar o modelo nos processos de refino secundário, configurando-o para fazer predições em tempo real.

O intervalo de datas entre os dados corresponde a 47 meses (janeiro/2019 a novembro/2022). Uma sugestão seria conduzir a coleta de dados por um intervalo de tempo maior para trazer mais robustez ao modelo implementado e retreiná-lo periodicamente à medida que novos dados são coletados.

Uma das limitações desse trabalho foi a qualidade dos dados de entrada, devido principalmente a erros de lançamento e medição. Também é adequado que ao longo do tempo sejam verificados novos estudos na literatura para aprimorar o modelo.

Referências

AZEVEDO, L. A. **Uma análise empírica do efeito do Ruído de Classe no aprendizado de Redes Neurais Artificiais**. 2022. Projeto Final (Bacharelado), Universidade Federal Rural do Rio de Janeiro.

BARCA, M. C. S.; SILVEIRA, T. R. S. Treinamento de redes neurais artificiais: o algoritmo backpropagation. **IX Encontro Latino Americano de Iniciação Científica**, Universidade do Vale do Paraíba, p. 46–49, 2005. Disponível em: <https://www.inicepg.univap.br/cd/INIC_2005/inic/IC120anais/IC1-17.pdf>. Acesso em: 28 jan. 2023.

BENATTI, K. A. **Sistemas não lineares via região de confiança:o algoritmo de Levenberg-Marquardt**. 2017. Dissertação (Mestrado). Universidade Federal do Paraná.

BOSCARIOLI, C.; BEZERRA, A.; BENEDICTO, M.; DELMIRO, G. Uma reflexão sobre banco de dados orientados a objetos. **IV CONGED - Congresso de Teconologias para Gestão de Dados e Metadados do Cone Sul**, Anais do IV CONGED, 2006. Disponível em: <https://conged.deinfo.uepg.br/artigo4.pdf>. Acesso em: 28 jan. 2023.

BRAGA, A. S. M. **Um Estudo Sobre As redes Neurais Aplicadas na Detecção de Comportamento Humano**. Tese (Doutorado) — Universidade Estadual de Campinas, Campinas, Brasil, 2022.

BRANDÃO, M. S.; GODINHO-FILHO, M.; JR., W. A.; BATTISSACCO, B. C.; MARÇOLA, J. A. Melhoria da categorização de produtos a partir do uso de algoritmos de aprendizado de máquina e medidas de similaridade. **Revista Produção Online**, Universidade Federal de Santa Catarina, v. 21, n. 4, p. 2093–2124, 2022. Disponível em: <https://www.producaoonline.org.br/rpo/article/view/4483>. Acesso em: 28 jan. 2023.

BROWN, M.; KROS, J. Data mining and the impact of missing data. **Industrial Management and Data Systems**, v. 103, p. 611–621, 11 2003.

CORREA, R. S.; CARVALHO, D. A. G.; CERCHIARI, B. S.; RIBEIRO, J. C.; MOURA, E. L. Utilização de redes neurais para prever o teor de hidrogênio em aços. **50° Seminário de Aciaria, Fundição e Metalurgia de Não-Ferrosos**, ABM, 2019. Disponível em: <https://abmproceedings.com.br/ptbr/article/download-pdf/utilizao-de-redes-neurais-para-prever-teor-de-hidrognio-em-aos>. Acesso em: 16 mar. 2023.

COSTA, L. H. **Sistemas de Apoio à Tomada de Decisão: Um Estudo Empírico Sobre o Uso do Sistema Visual Flash na Gestão Industrial**. 2016. Monografia (Especialização), Universidade Federal de Minas Gerais.

DANTAS, C. A. **Seleção de Atributos Baseado em Algoritmos de Agrupamento para Tarefas de Classificação**. 2017. Dissertação (Mestrado), Universidade Federal do Rio Grande do Norte.

FONTANA, E. **Introdução aos Algoritmos de Aprendizagem Supervisionada**. 2020. Apostila, Universidade Federal do Paraná.

FUCUN, L.; WU, j.; DONG, F.; LIN, J.; SUN, G. Ensemble machine learning systems for the estimation of steel quality control. **2018 IEEE International Conference on Big Data (Big Data)**, p. 2245–2252, 2018. Disponível em: <http://dx.doi.org/10.1109/BigData.2018.8622583>. Acesso em: 09 fev. 2023.

FUJII, T.; MANETTA, H. R.; NETO, O. A. F.; CASTRO, L. F. A. Estudo da incorporação e controle de hidrogênio na aciaria da V&M do Brasil. In: **XXXIV Seminário de Fusão, Refino e Solidificação de Metais**. Brasil: Associação Brasileira de Metalurgia, 2003. p. 149–157.

FURTADO, M. I. V. **Redes Neurais Artificiais: Uma Abordagem Para Sala de Aula**. Ponta Grossa, PR: Atena Editora, 2019.

GARDENGHI, J. L. C.; SANTOS, S. A. S. **Sistemas não lineares via região de confiança:o algoritmo de Levenberg-Marquardt**. 2011. Relatório de Pesquisa. Universidade de Campinas.

GHARAT, S. **What, Why and Which?? Activation Functions**. 2019. Disponível em https://medium.com/@snaily16/what-why-and-which-activation-functions-b2bf748c0441 . Acesso em 27 mar. 2023.

GONÇALVES, R. M.; COELHO, L. S.; KRUEGER, C. P. Modelagem preditiva de linha de costa utilizando redes neurais artificiais. **Boletim de Ciências Geodésicas**, Universidade Federal do Paraná, v. 16, n. 3, p. 420–444, 2010. ISSN 1982-2170. Disponível em: <https://revistas.ufpr.br/bcg/article/view/18725>. Acesso em: 28 jan. 2023.

GUPTA, D. **Fundamentals of Deep Learning – Activation Functions and When to Use Them?** 2022. Disponível em https://www.analyticsvidhya.com/blog/2020/01/fundamentals-deep-learning-activation-functions-when-to-use-them/ . Acesso em 27 mar. 2023.

HENRIQUES, B. R. **Estudo da Incorporação de Hidrogênio no Aço Líquido**. Dissertação (Mestrado) — REDEMAT - Universidade Federal de Ouro Preto, Ouro Preto, MG, 2010.

KHOLIEF, E. A.; DARWISH, S. H.; FORS, M. N. **Detection of steel surface defect based on machine learning using deep auto-encoder network**. 2017. International Conference on Industrial Engineering and Operations Management Rabat, Marrocos. Disponível em: <https://www.researchgate.net/publication/331872197_Detection_of_Steel_Surface_Defect_Based_on_Machine_Learning_Using_Deep_Auto-encoder_Network>. Acesso em: 09 fev. 2023.

LEE, H. D. **Seleção de Atributos Baseado em Algoritmos de Agrupamento para Tarefas de Classificação**. 2005. Tese (Doutorado), Universidade de São Paulo.

MEDRI, W. **Análise Exploratória de Dados**. 2011. Apostila, Universidade Estadual de Londrina.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. In: **Sistemas Inteligentes Fundamentos e Aplicações**. 1. ed. Barueri-SP: Manole Ltda, 2003. p. 89–114. ISBN 85-204-168.

NIED, A. **Treinamento De Redes Neurais Artificiais Baseado em Sistemas de Estrutura Variavel com Taxa de Aprendizado Adaptativa**. Tese (Doutorado) — Universidade Federal de Minas Gerais, Belo Horizonte, Brasil, 2007.

PAIXÃO, G. M. M.; SANTOS, B. C.; ARAUJO, R. M.; RIBEIRO, M. H.; MORAES, J. L.; RIBEIRO, A. L. Machine learning na medicina: Revisão e aplicabilidade. **Arquivos Brasileiros de Cardiologia**, Universidade Federal de Minas Gerais, v. 118, n. 1, p. 95–102, 2022. ISSN 1678-4170. Disponível em: <https://doi.org/10.36660/abc.20200596>. Acesso em: 13 set. 2022.

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

SCHWERTNER, E. S. D. **Uma classe de métodos do tipo Levenberg-Marquardt com passos múltiplos para problemas de Otimização de Menor Valor Ordenado**. 2020. Dissertação (Mestrado). Universidade Estadual de Maringá.

SILVA, J. J. S. **O Algoritmo de Retropropagação**. 2022. Monografia (Licenciatura), Instituto Federal da Paraíba.

TEIXEIRA, B. O. S.; TORRES, L. A. B.; AGUIRRE, L. A. Filtragem de kalman com restrições para sistemas não-lineares: Revisão e novos resultados. **Revista Controle & Automação**, Sociedade Brasileira de Automática, v. 21, p. 127–146, 2010. Disponível em: <https://doi.org/10.1590/S0103-17592010000200003>. Acesso em: 28 jan. 2023.

TREVISAN, V. **Comparing Robustness of MAE, MSE and RMSE**. 2022. Disponível em https://towardsdatascience.com/comparing-robustness-of-mae-mse-and-rmse-6d69da870828 . Acesso em 27 mar. 2023.

VERONEZE, R. **Tratamento de Dados Faltantes Empregando Biclusterização com Imputação Múltipla**. 2011. Dissertação (Mestrado), Universidade Estadual de Campinas.

WU, S. **3 Best metrics to evaluate Regression Model**. 2020. Disponível em https://towardsdatascience.com/what-are-the-best-metrics-to-evaluate-your-regression-model-418ca481755b . Acesso em 27 mar. 2023.

ZHANG, B.; ZHANG, X.; FAN, L. **Breakout Prediction Based on BP Neural Network in continuous Casting Process**. 2016. MATEC Web of Conferences. Disponível em: <https://www.matec-conferences.org/articles/matecconf/pdf/2016/24/matecconf_apop2016_05020.pdf>. Acesso em: 09 fev. 2023.

ZORZATO, M. G. **Análise Termodinâmica da Incorporação de Hidrogênio pelo Aço Líquido Através da Escória de Refino Secundário**. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Sul, Porto Alegre, RS, 2013.

APÊNDICE A – Variáveis da base de dados

Nome da variável	Descrição
Data	Data da Corrida
Corrida	Número da Corrida
UMIDADE_R	Umidade do ar
Familia	Família do aço
Trat	Tipo de Tratamento
ROTA_CORRIDA	Rota da corrida
RH	Tratamento RH
H_y	Hidrogênio medido Hydris
PesoAcoPAN	Peso aço na panela
TempoCASOB	Tempo de CASOB
TempoFP	Tempo de FP
TempoRH	Tempo de RH
TEMPO_Vacuo	Tempo de Vácuo
Just_Vacuo	Justificativa Vácuo >2 tor
VOL_OX_LD	Volume oxigênio LD
VOL_OX_RH	Volume oxigênio RH
VOL_OX_CASOB	Volume oxigênio CASOB
ARG_CASOB	Volume argônio CASOB
ARG_FP	Volume argônio FP
COQUE_LD	Coque fino calculado no LD
CALE_LD	Cal na panela no LD
ESIN_LD	Escoria sintetica no LD
CARB_LD	Carbureto SI no LD
FLUO_LD	Fluorita na panela no LD
NEFE_LD	Nefelina no LD
BRIF_LD	Briquete cobertura - Aço IF no LD
BRIQ_LD	Briquete dessulfurante no LD
BRID_LD	Briquete Desfosforante no LD
RALO_LD	Borra de aluminio no LD
PGRAF_LD	Grafite no LD
COQ_CALC_LD	Coque calculado no LD
COQUE_FP	Coque fino calculado no FP
C_FIO_FP	Carbono fio no FP
COQ_CALC_FP	Coque calculado no FP

Nome da variável	**Descrição**
FECA_FP	Ferro Calcio em fio horizontal no FP
CAFI_FP	Cal fina para injeção no FP
POCS_FP	CaSi em po no FP
CASI_FP	Calcio silicio fio horizontal no FP
FLUO_FP	Fluorita na panela no FP
CALE_FP	Cal na panela no FP
ESIN_FP	Escoria sintetica no FP
BREF_FP	Briquete Refrigerante no FP
AGRE_FP	Agregado Residual no FP
FCAL_FP	FeCaAl em fio (vertical) no FP
NEFE_FP	Nefelina no FP
BRIQ_FP	Briquete dessulfurante no FP
CSAL_FP	CaSiAl em fio vertical no FP
CASV_FP	Ca silicio fio vertical no FP
FECV_FP	Fe calcio fio vertical no FP
BRIF_FP	Briquete cobertura - Aço IF no FP
FCAH_FP	Ferro Calcio Aluminio Fio Horizontal no FP
NBRQ_FP	Novo Briquete Dessulfurante no FP
AGRB_FP	Agregado Briquetado no FP
AGRG_FP	Agregado Granel no FP
AGRP_FP	Agregado Peneirado no FP
CARB_FP	Carbureto SI no FP
BRID_FP	Briquete Desfosforante no FP
RALO_FP	Borra de aluminio no FP
COQUE_OB	Coque fino calculado no CASOB
FECA_OB	Ferro Calcio em fio horizontal no CASOB
CAFI_OB	Cal fina para injeção no CASOB
POCS_OB	CaSi em po no CASOB
CASI_OB	Calcio silicio fio horizontal no CASOB
FLUO_OB	Fluorita na panela no CASOB
CALE_OB	Cal na panela no CASOB
ESIN_OB	Escoria sintetica no CASOB
BREF_OB	Briquete Refrigerante no CASOB
AGRE_OB	Agregado Residual no CASOB
FCAL_OB	FeCaAl em fio (vertical) no CASOB
NEFE_OB	Nefelina no CASOB
BRIQ_OB	Briquete dessulfurante no CASOB
CSAL_OB	CaSiAl em fio vertical no CASOB

Nome da variável	Descrição
CASV_OB	Ca silicio fio vertical no CASOB
FECV_OB	Fe calcio fio vertical no CASOB
BRIF_OB	Briquete cobertura - Aço IF no CASOB
FCAH_OB	Ferro Calcio Aluminio Fio Horizontal no CASOB
NBRQ_OB	Novo Briquete Dessulfurante no CASOB
AGRB_OB	Agregado Briquetado no CASOB
AGRG_OB	Agregado Granel no CASOB
AGRP_OB	Agregado Peneirado no CASOB
CARB_OB	Carbureto SI no CASOB
BRID_OB	Briquete Desfosforante no CASOB
RALO_OB	Borra de aluminio no CASOB
PGRAF_OB	Grafite no CASOB
C_FIO_OB	Carbono fio no CASOB
COQ_CALC_OB	Coque calculado no CASOB
COQUE_RH	Coque fino calculado no RH
FECA_RH	Ferro Calcio em fio horizontal no RH
CAFI_RH	Cal fina para injeção no RH
POCS_RH	CaSi em po no RH
CASI_RH	Calcio silicio fio horizontal no RH
FLUO_RH	Fluorita na panela no RH
CALE_RH	Cal na panela no RH
ESIN_RH	Escoria sintetica no RH
BREF_RH	Briquete Refrigerante no RH
AGRE_RH	Agregado Residual no RH
FCAL_RH	FeCaAl em fio (vertical) no RH
NEFE_RH	Nefelina no RH
BRIQ_RH	Briquete dessulfurante no RH
CSAL_RH	CaSiAl em fio vertical no RH
CASV_RH	Ca silicio fio vertical no RH
FECV_RH	Fe calcio fio vertical no RH
BRIF_RH	Briquete cobertura - Aço IF no RH
FCAH_RH	Ferro Calcio Aluminio Fio Horizontal no RH
NBRQ_RH	Novo Briquete Dessulfurante no RH
AGRB_RH	Agregado Briquetado no RH
AGRG_RH	Agregado Granel no RH
AGRP_RH	Agregado Peneirado no RH
CARB_RH	Carbureto SI no RH
BRID_RH	Briquete Desfosforante no RH

Nome da variável	**Descrição**
RALO_RH	Borra de aluminio no RH
C_FIO_RH	Carbono fio no RH
COQ_CALC_RH	Coque calculado no RH
CAL_INJ_FP	Cal Micro no FP
CAL_INJ_CAS	Cal Micro no CASOB
CASI_INJ_CAS	Cal Micro no CASOB
ALBA	Aluminio em barra
ALFI	Aluminio em fio (jumbo)
ALGO	Alumínio gotão
ALGR	Alumínio granulado
ANTI	Antimonio
BLOG	Briquete teste
COQC	Coque
CRBC	Ferro cromo baixo carbono
FE_S	Ferro Enxofre
FE-B	Ferro Boro a granel
FEBE	Ferro Boro embalado
FEMO	Ferro molibdenio
FENB	Ferro niobio
FEP	Ferro fosforo
FETI	Ferro titanio a granel 10-50mm
FETR	Ferro titanio a granel RH 3-13
FEV	Ferro vanadio
MNAC	Ferro manganes alto carbono
MNBC	Ferro manganes baixo carbono
MNBP	Fe manganes baixo P
MNEL	Manganes eletrolitico
MNMC	Ferro manganes medio carbono
NI-E	Níquel eletrolitico aparas
SCCO	Sucata de cobre
SIBC	Fe silicio baixo C 10-50 mm
SIBR	Fe silicio baixo C 10-30 mm
SIGL	Fe silicio no gusa l¡quido
SIMN	Ferro silicio manganes
SIMT	Silício Metálico
SMFC	Silício
AL	Aiumínio da análise química
AS	Arsênio da análise química

Nome da variável	**Descrição**
AT	Astato da análise química
B	Boro da análise química
C	Carbono da análise química
CA	Cálcio da análise química
CO	Cobalto da análise química
CR	Cromo da análise química
CU	Cobre da análise química
H	Hidrogênio da análise química
MN	Manganês da análise química
MO	Molibdênio da análise química
N	Nitrogênio da análise química
NB	Nióbio da análise química
NI	Níquel da análise química
O	Oxigênio da análise química
P	Fósforo da análise química
S	Enxofre da análise química
SB	Antimônio da análise química
SI	Silício da análise química
SN	Estanho da análise química
TI	Titânio da análise química
V	Vanádio da análise química
ZR	Zircônio da análise química
PB	Chumbo da análise química
BI	Bismuto da análise química
W	Tungstênio da análise química
CD	Cádmio da análise química
BE	Berílio da análise química
ZN	Zinco da análise química
BA	Bário da análise química
HG	Mercúrio da análise química
CH	CH da análise química
BSOL	Boro Solúvel na análise química
AO	Alumínio solúvel na análise química
AI	Alumínio insolúvel na análise química
TS	Tenessino da análise química
C_VIS	Carbono Visado
SI_VIS	Silício Visado
MN_VIS	Manganês Visado

Nome da variável	**Descrição**
P_VIS	Fósforo Visado
S_VIS	Enxofre Visado
AL_VIS	Alumínio Visado
CU_VIS	Cobre Visado
NB_VIS	Nióbio Visado
V_VIS	Vanádio Visado
TI_VIS	Titânio Visado
CR_VIS	Cromo Visado
NI_VIS	Níquel Visado
MO_VIS	Molibdênio Visado
SN_VIS	Estanho Visado
N_VIS	Nitrogênio Visado
H2_VIS	Hidrogênio Visado
CO_VIS	Cobalto Visado
AS_VIS	Arsênio Visado
O2_VIS	Oxigênio Visado
B_VIS	Boro Visado
CA_VIS	Cálcio Visado
AT_VIS	Astato Visado
SB_VIS	Antimônio Visado
ZR_VIS	Zircônio Visado
BI_VIS	Bismuto Visado
CE_VIS	Cério Visado
W_VIS	Tungstênio Visado
PB_VIS	Chumbo Visado
CD_VIS	Cádmio Visado
BE_VIS	Berílio Visado
ZN_VIS	Zinco Visado
BA_VIS	Bário Visado
HG_VIS	Mercúrio Visado
CH_VIS	CH Visado
BSOL_VIS	Boro Solúvel Visado
AO_VIS	Alumínio solúvel Visado
AI_VIS	Alumínio insolúvel Visado
TS_VIS	Tenessino Visado
DIF_C	Diferença Carbono Visado x Real
DIF_SI	Diferença Silício Visado x Real
DIF_MN	Diferença Manganês Visado x Real

Nome da variável	**Descrição**
DIF_P	Diferença Fósforo Visado x Real
DIF_S	Diferença Enxofre Visado x Real
DIF_AL	Diferença Alumínio Visado x Real
DIF_CU	Diferença Cobre Visado x Real
DIF_NB	Diferença Nióbio Visado x Real
DIF_V	Diferença Vanádio Visado x Real
DIF_TI	Diferença Titânio Visado x Real
DIF_CR	Diferença Cromo Visado x Real
DIF_NI	Diferença Níquel Visado x Real
DIF_MO	Diferença Molibdênio Visado x Real
DIF_SN	Diferença Estanho Visado x Real
DIF_N	Diferença Nitrogênio Visado x Real
DIF_CO	Diferença Cobalto Visado x Real
DIF_AS	Diferença Arsênio Visado x Real
DIF_B	Diferença Boro Visado x Real
DIF_CA	Diferença Cálcio Visado x Real
DIF_AT	Diferença Astato Visado x Real
DIF_SB	Diferença Antimônio Visado x Real
DIF_ZR	Diferença Zircônio Visado x Real
DIF_BI	Diferença Bismuto Visado x Real
DIF_W	Diferença Tungstênio Visado x Real
DIF_PB	Diferença Chumbo Visado x Real
DIF_CD	Diferença Cádmio Visado x Real
DIF_BE	Diferença Berílio Visado x Real
DIF_ZN	Diferença Zinco Visado x Real
DIF_BA	Diferença Bário Visado x Real
DIF_HG	Diferença Mercúrio Visado x Real
DIF_CH	Diferença CH Visado x Real
DIF_BSOL	Diferença BSOL Visado x Real
DIF_AO	Diferença AO Visado x Real
DIF_AI	Diferença AI Visado x Real
DIF_TS	Diferença Tenessino Visado x Real