UNIVERSIDADE FEDERAL DE MINAS GERAIS
PROGRAMA DE PÓS GRADUAÇÃO EM ENGENHARIA QUÍMICA

ANA BRANDÃO BELISÁRIO

ANÁLISE DE EMISSÕES EM CALDEIRAS DE RECUPERAÇÃO QUÍMICA DE FÁBRICAS DE CELULOSE KRAFT: PREDIÇÃO E ANÁLISE DE SENSIBILIDADE COM REDES NEURAIS ARTIFICIAIS

BELO HORIZONTE - MG

2020

ANA BRANDÃO BELISÁRIO

ANÁLISE DE EMISSÕES EM CALDEIRAS DE RECUPERAÇÃO QUÍMICA DE FÁBRICAS DE CELULOSE KRAFT: PREDIÇÃO E ANÁLISE DE SENSIBILIDADE COM REDES NEURAIS ARTIFICIAIS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Química da Escola de Engenharia da Universidade Federal de Minas Gerais, como requisito parcial para obtenção do Grau de Mestre em Engenharia Química.

Orientador: Professor Gustavo Matheus de Almeida
Linha de Pesquisa: Simulação e Otimização de Processos

BELO HORIZONTE - MG

2020

Belisário, Ana Brandão.
B431a Análise de emissões em caldeiras de recuperação química de fábricas de celulose kraft [recurso eletrônico] : predição e análise de sensibilidade com redes neurais artificiais / Ana Brandão Belisário. - 2020.
1 recurso online (xix, 80 f. : il., color.) : pdf.

Orientador: Gustavo Matheus de Almeida.

Dissertação (mestrado) - Universidade Federal de Minas Gerais, Escola de Engenharia.

Anexos: f. 79-80.

Bibliografia: f. 70-78.
Exigências do sistema: Adobe Acrobat Reader.

1. Engenharia química - Teses. 2. Predição (Lógica) - Teses. 3. Análise de sensibilidade - Teses. 4. Redes neurais (Computação) - Teses. 5. Banco de dados - Teses. I. Almeida, Gustavo Matheus de. II. Universidade Federal de Minas Gerais. Escola de Engenharia. III. Título.

CDU: 66.0(043)

Ficha catalográfica elaborada pela bibliotecária Roseli Alves de Oliveira CRB/6 2121
Biblioteca Prof. Mário Werneck, Escola de Engenharia da UFMG

UNIVERSIDADE FEDERAL DE MINAS GERAIS
ESCOLA DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA QUÍMICA

"Análise de emissões em caldeiras de recuperação química de fábricas de celulose Kraft: Predição e análise de sensibilidade com redes neurais artificiais"

Ana Brandão Belisário

Dissertação submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Química da Escola de Engenharia da Universidade Federal de Minas Gerais, como parte dos requisitos à obtenção do título de **MESTRE EM ENGENHARIA QUÍMICA**.

269ª DISSERTAÇÃO APROVADA EM 21 DE FEVEREIRO DE 2020 POR:

.....
Prof. Dr. Eduardo Coutinho de Paula
DESA/UFMG

.....
Prof. Dr. Marcelo Cardoso
DEQ/UFMG

.....
Prof. Dr. Gustavo Matheus de Almeida
Orientador - DEQ/UFMG

Dedico este trabalho a todas e todos que se dedicam
sinceramente em fazer dos seus mundos
lugares melhores para a coletividade.

AGRADECIMENTOS

Gostaria de agradecer a todas as pessoas, situações e vivências que me influenciam ou me influenciaram ao longo de toda essa trajetória. Em especial, minha imensa gratidão à minha Mãe, ao meu Pai, aos meus irmãos, família e a Deus que sempre me acompanham e apoiam em minhas decisões. Agradeço o professor Gustavo, excelente orientador com quem aprendi muito e para além do acadêmico, ao professor Marcelo, pelas conversas, atenção, apoio e aprendizados, aos membros da banca, que contribuíram bastante para o aprimoramento do trabalho, e também às professoras e aos professores que me influenciam e me apoiam em meu desenvolvimento. Agradeço também ao professor Esa pelos ensinamentos e pela oportunidade de estudar na LUT, Finlândia. Sou muito grata a todas amigas e amigos que dividiram comigo esse tempo do mestrado, tanto nos problemas quanto nos divertimentos. A todos que prezam, investem e desenvolvem softwares livres e de código aberto. Agradeço, por fim, ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo financiamento desta pesquisa.

RESUMO

Mundialmente, há um movimento de maior conscientização socioambiental, pautado em interesses da sociedade em geral, de instituições e governos. Disso resulta uma pressão maior sobre setores industriais para o desenvolvimento de processos mais limpos e sustentáveis. Uma das preocupações está relacionada com as emissões de gases, dentre eles, os óxidos de carbono (CO e CO_2), de nitrogênio (NO_x) e de enxofre (SO_x), além das emissões mercaptanas e de material particulado em suspensão, especialmente quando as indústrias se localizam nas proximidades de cidades. Nesse cenário, o monitoramento e controle de emissões ganha importância, e metodologias alternativas a métodos tradicionais, em geral custosos, têm sido desenvolvidas. Dentre as soluções em monitoramento de processos, destacam-se os sensores virtuais, baseados diretamente em dados históricos sobre as operações. Nesse contexto, o objetivo do presente trabalho foi construir sensores virtuais para emissões de gases e material particulado em fábricas de celulose, e então analisar quais variáveis de processo apresentam maior influência sobre tais emissões. O primeiro estudo de caso diz respeito à caldeira de recuperação química de uma indústria de celulose no Brasil, com o foco na construção de um sensor virtual para estimativa das emissões de óxido de enxofre (SO_2). O segundo estudo de caso se refere à caldeira de recuperação química de uma indústria de celulose na Finlândia, com o foco nas emissões de material particulado. Para ambas as aplicações, os sensores virtuais se baseiam em redes neurais artificiais com arquitetura MLP (*Multi-Layer Perceptrons*), a mais usual em aplicações para Engenharia Química em geral. De modo a verificar o seu desempenho, compararam-se os resultados com o modelo de regressão linear múltipla (MLR). Após a variação de um conjunto de parâmetros, como por exemplo, tipo de função de ativação e número de neurônios ocultos, a rede neural que melhor se ajustou aos dados alcançou coeficiente de correlação linear (r) significativamente superior àquele da regressão linear múltiplas, para ambos os estudos de caso, $r_{mlr} = 0.764$ e $r_{mlp} = 0.939$, para a caldeira do Brasil, e $r_{mlr} = 0.6974$ and $r_{mlp} = 0.86$, para a caldeira da Finlândia. Ao final, as variáveis de processo que mais influenciam os resultados dos sensores virtuais, para cada um dos casos, foram identificadas por meio de um estudo de análise de sensibilidade. Para o primeiro estudo de caso, as variáveis mais influentes foram vazão do ar terciário, vazão do ar primário, vazão do licor preto e temperatura do ar primário. Para o segundo estudo de caso, as variáveis mais influentes foram vazão do ar secundário, vazão do licor preto e seu teor de sólidos, vazão de gases não condensáveis diluídos e vazão de óleo combustível.

Palavras-chaves: Fábrica de celulose Kraft. Caldeira de recuperação química. Emissões. Predição. Análise de sensibilidade. Rede neural artificial. Bancos de dados industriais..

ABSTRACT

Globally, there is a movement of greater social-environmental awareness, based on the interests of society, institutions and governments. As a result, there is an increasing pressure on industrial sectors for the development of cleaner and more sustainable processes. One of the concerns is related to emissions of gases, among them, carbon oxides (CO and CO_2), nitrogen (NO_x) and sulfur (SO_x), in addition to dust emissions, suspended particulate matter, especially when industries are located close to cities. In this scenario, the monitoring and control of emissions gains importance, and alternative methodologies to traditional, usually costly, methods have been developed. Among the solutions in process monitoring, stand out the virtual sensors based directly on historical data from the operations. In this context, the objective of the present work was to build virtual sensors for gas and particulate material emissions monitoring from equipment in pulp mills and then analyse the variables that present the greate influence over those emissions. The first case involves a chemical recovery boiler of a pulp industry in Brazil. A virtual sensor was constructed for the estimation of sulfur oxide (SO_2) emissions. And the second case study refers to the chemical recovery boiler in a cellulose industry in Finland, with a focus on emissions of particulate matter. For both applications, the virtual sensors are based on a neural network with MLP (*Multi-Layer Perceptrons*) architecture, the most usual in chemical engineering applications in general. In order to verify their performance, the results were compared with multiple linear regression model. After varying a set of parameters, such as type of activation function and number of hidden neurons, the neural network that best fitted the data reached a linear correlation coefficient (*r*) higher than that of linear regression multiple, for both case studies, $r_{mlr} = 0.764$ e $r_{mlp} = 0.939$, for the boiler in Brazil, and $r_{mlr} = 0.6974$ and $r_{mlp} = 0.86$, for the boiler in Finland. In the end, the process variables that most influence the results of the virtual sensors, for each case, were identified by applying a sensitivity analysis technique. For the first case study, the most influential variables were tertiary air flow, primary air flow, black liquor flow and primary air temperature. For the second case study, the most influential variables were secondary air flow, black liquor flow and its solids content, diluted non-condensable gas flow and fuel oil flow

Key-words: Kraft pulp mill. Chemical recovery boiler. Emissions. Prediction. Sensitivity analysis. Artificial neural network. Industrial data sets.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E DE SIGLAS

ANN Artificial Neural Network

CFD Computational Fluid Dynamics

CaO Óxido de Cálcio

DCS Distributed Control System

DNCG Diluted Non-Condensable Gas

GMM Gaussian Mixture Model

ICA Independent Component Analysis

MAD Median Absolute Deviation

MLP MultiLayer Perceptron

MSE Mean Squared Error

NLPCA Nonlinear Principal Component Analysis

NNPLS Neural Network Partial Least Squares

Na_2S Sulfeto de Sódio

Na_2CO_3 Carbonato de Sódio

NaOH Hidróxido de Sódio

OFA Overfire Air

PCA Principal Component Analysis

PCR Principal Component Regression Model

PLS Partial Least Squares

PRESS Predicted Residual Error Sum of Squares

SO_2 Dióxido de Enxofre

SPE Squared Prediction Error

SVM Support Vector Machine

TRS Total Reduced Sulfur

SUMÁRIO

1 INTRODUÇÃO

Desde os anos de 1960, o advento do desenvolvimento e da expansão de Sistemas de Controle Distribuído (DSC, do inglês *distributed control systems*) relacionado aos avanços da capacidade computacional e de processamento tem aberto espaço para a evolução das formas de se coletar, armazenar, analisar e visualizar enormes quantidades de dados, contribuindo para a compreensão mais aprofundada de problemas muito complexos. No contexto da indústria de processos, associam-se grandes bases de dados disponíveis, poder computacional e metodologias de análise no sentido do desenvolvimento de técnicas e ferramentas que podem auxiliar supervisores e operadores das plantas industriais na obtenção de informações relevantes, a partir da análise e da interpretação dos dados, de modo a identificar problemas e aumentar a eficiência e a efetividade da operação (WANG; MCGREAVY, 1999).

Em paralelo, existe uma demanda crescente por melhoria de performance, produtividade, qualidade de processo e produto, associada a fatores econômicos e de exigências ambientais, de saúde e segurança do trabalho. Desse modo, por razões econômicas, governamentais e demandas sociais, acentua-se, cada vez mais, uma tendência interna do setor industrial para a implementação de inovações tecnológicas nos campos de monitoramento e controle de processos. Como consequência, tem-se a incorporação dos avanços tecnológicos de armazenamento e processamento de dados às rotinas fabris, e o crescimento em importância do campo de estudo voltado ao monitoramento de processos baseado em dados reais (WANG; MCGREAVY, 1999).

O papel desempenhado pelos sistemas de monitoramento de processos se relaciona à criação de um ambiente virtual capaz de apresentar, de forma abrangente, avaliações de performance e identificar seus fatores críticos, permitindo maior segurança das operações e proporcionando alta eficiência, contínua e duradoura, ao processo. Neste ponto, é fundamental mencionar que as pessoas responsáveis pela operação (operadores e supervisores, por exemplo) são uma parte importante de qualquer solução aplicável ao processo, portanto elas devem ser consideradas em qualquer implementação de novos métodos e sistemas de monitoramento e controle. Sendo estas as pessoas mais próximas à operação, a interface dos sistemas de monitoramento e controle para operadores e supervisores deve ser de compreensão clara e rica em informações úteis. Na interface desses sistemas devem ser apresentados o estado atual do processo, as principais causas potenciais de eventuais problemas, bem como dados que sustentem tomadas de decisão, sejam para melhoria geral de performance ou para intervenções que evitem ou remedeiem possíveis falhas (WANG; MCGREAVY, 1999).

Sistemas de monitoramento de processos podem ser concebidos seguindo três tipos de abordagens fundamentais: baseados em modelos fenomenológicos, baseados em conhecimento e baseados em dados. Os mais tradicionais são os métodos baseados em modelos fenomenológicos, construídos a partir do conhecimento das relações físico-químicas existentes entre as variáveis. A assertividade dos resultados obtidos dessa forma é dependente da adequação do modelo matemático proposto ao processo real. Os métodos baseados em conhecimento/experiência são formulados segundo aprendizados majoritariamente tácitos assimilados a partir da vivência da pessoa que convive com o processo específico, sendo, portanto, muito dependente das experiências fundamentadas nas possibilidades de comportamento da operação. Por fim, os modelos baseados em dados dependem apenas da base de dados gerada pelo próprio processo, assim, para que o modelo seja assertivo, é importante ter dados apropriados e representativos. Quanto mais complexos os processos e os sistemas industriais forem, mais difícil é a formulação de modelos fenomenológicos e mais trabalhoso e demorado é a construção de modelos baseados em conhecimento. Neste campo, os modelos baseados em dados ganham importância, afinal, as capacidades de armazenamento e processamento das máquinas já não são consideradas problemas. Cabe ressaltar, entretanto, que à medida que as demandas e as pesquisas evoluem, os métodos que combinam a experiência dos operadores e/ou modelos fenomenológicos com modelos baseados em dados têm se tornado relevantes (GE; SONG; GAO, 2013).

Tratando especificamente dos modelos de monitoramento baseados em dados, existe uma variedade de metodologias para atender processos com características diversas. Dentre os modelos mais tradicionais, destacam-se as abordagens que envolvem estatística multivariada. Em tais abordagens, assume-se, em geral, que os dados apresentam distribuição normal apenas e correlação espacial. Contudo, tais considerações não são válidas para todos os casos. Assim, têm-se desenvolvido evoluções nos métodos de monitoramento de processos baseados em estatística multivariada, bem como se introduzido outros tipos de técnicas e metodologias baseadas em dados. Podem-se citar a análise por componentes independentes (ICA, do inglês *Independent Component Analysis*), modelos de mistura Gaussiano (GMM , do inglês *Gaussian Mixture Models*), máquinas de vetores de suporte (SVM , do inglês *Support Vector Machines*), redes neurais artificiais (ANN , do inglês *Artificial Neural Networks*), dentre outros (GE; SONG; GAO, 2013).

As técnicas mais recentes de monitoramento e controle de processos trazem de forma muito aplicada conceitos e metodologias de inteligência artificial e aprendizado de máquina (*machine learning*), com o objetivo de colocar a capacidade computacional e de processamento a serviço dos interesses das pessoas responsáveis pelas unidades de operação das indústrias. Segundo Everitt e Skrondal (2010), o significado de aprendizado de máquina está associado ao estudo e à construção de algoritmos

computacionais capazes de melhorarem o próprio desempenho, tendo como base experiências passadas, sucedendo que tais informações de experiência são fornecidas por meio de bases de dados.

Em matéria de aprendizado de máquina, quatro são os tipos clássicos de problemas: (i) aprendizado supervisionado: quando o conjunto de treinamento apresenta os valores ou rótulos correspondentes à(s) variável(eis) de saída, sendo, em geral, trabalhados com algoritmos de regressão ou classificação; (ii) aprendizado não supervisionado: as entradas para treinamento do modelo não estão associadas a rótulos ou valores de saída; neste caso, são utilizados algoritmos de agrupamento (*clustering*) para descobrir classes dentro do conjunto de dados; (iii) aprendizado semi supervisionado: há a utilização de entradas com e sem rótulos, sendo que as entradas com rótulos são usualmente aplicadas para a construção do modelo básico, e as entradas não rotuladas são utilizadas para refinamento da solução; (iv) aprendizado ativo: ocorre quando os usuários atuam ativamente no processo de aprendizado do algoritmo, objetivando otimizar a qualidade do modelo pela aquisição ativa de conhecimento a partir da interação com o usuário (HAN; PEI; KAMBER, 2011).

De forma geral, para qualquer sistema de monitoramento e controle construído com base em conjuntos de dados, é fundamental a garantia da qualidade dos dados a fim de que os algoritmos sejam assertivos e eficazes. Assim, no setor industrial é importante que os processos e seus equipamentos estejam bem instrumentados com variados sensores e analisadores bem calibrados. Usualmente, os sensores físicos (*hardware sensors*), instalados na linha de produção (*online*), são capazes de enviar informações sobre o estado da variável a intervalos de tempo muito curtos, sendo bastante úteis para supervisão, monitoramento e controle instantâneo de variáveis como temperatura, vazão, pressão, acidez, dentre outras. Entretanto, existem variáveis de interesse, especialmente aquelas que indicam qualidade de produto, restrições ambientais e desempenho econômico, que são difíceis de medir automaticamente por sensores físicos. Essa relativa impossibilidade de medição se deve, principalmente, à inexistência de sensores, à elevada taxa de desgaste ou ao elevado custo de instalação e/ou manutenção.

Algumas variáveis de qualidade cujos sensores *online* não estão disponíveis são medidas em laboratório. Essas dependem da retirada da amostra, do seu recebimento no laboratório, seguida de preparação, análise e obtenção do resultado. Tal procedimento pode demorar de minutos a algumas horas. Isso significa que o resultado de laboratório, importante, por exemplo, para o controle de qualidade, é entregue de forma tardia e, caso exista qualquer necessidade de intervenção no processo, o estado atual já não corresponde ao momento quando a amostra foi retirada, sendo o ajuste feito tardiamente (QIN; YUE; DUNIA, 1997). Para remediar esse problema, é possível

utilizar as variáveis de processo medidas automaticamente para inferir os valores das variáveis que não são medidas em tempo real, aplicando métodos de inteligência computacional. Os sensores virtuais ou inferenciais (*software 'soft'* ou *inferential sensors*) são aqueles que utilizam ferramentas computacionais para compreender as relações funcionais entre variáveis, de modo a produzir estimativas para variáveis difíceis de medir a partir das variáveis medidas *online*, seguindo a mesma taxa de medição dos sensores físicos (SOUZA; ARAÚJO; MENDES, 2016).

Segundo Souza, Araújo e Mendes (2016), os sensores virtuais são aplicações importantes em diversos tipos de processos industriais, como indústria de celulose e papel, sistemas de tratamento de água e esgoto, cimenteiras, refinarias, processos de polimerização e reatores biológicos. Paralelamente às aplicações de sensores virtuais, têm-se o crescimento da responsabilidade ambiental. Segundo Bajpai (2011), a indústria tem feito modificações e inovações em seus processos produtivos em prol de alcançar uma produção mais limpa. Quanto aos resultados da implementação de novas práticas, têm-se observado consequências positivas para além da responsabilidade ambiental, como redução de custo, aumento de eficiência, produtividade e competitividade, além de redução de passivos ambientais.

Essas tendências vêm em concordância com o conceito internacional de 'Produção Mais Limpa' (*Cleaner Production*), definido pelo Programa Ambiental das Nações Unidas como "Aplicação contínua de estratégia ambiental integrada a processos, produtos e serviços para aumentar eficiência e reduzir riscos a pessoas e ao meio ambiente" (UNEP, 1990). Para cada uma dessas perspectivas (processo, produto, serviço) estão envolvidas estratégias integradas distintas. Por exemplo, para processos produtivos, estão incluídas políticas de conservação de matérias-primas, energia e água, de prevenção do uso de substâncias tóxicas, de redução da quantidade e toxicidade de emissões e efluentes; para produtos, a questão está na redução dos impactos negativos desde a extração da matéria-prima até a disposição final após o uso, ou seja, os efeitos produzidos pelo ciclo de vida completo do produto; já para serviços, o conceito de ambientalmente sustentável pode estar integrado desde a sua concepção até a entrega (BAJPAI, 2011). Uma das formas de contribuir para a implementação de práticas que promovam a sustentabilidade ocorre por meio de estudo, desenvolvimento e implementação de tecnologias que utilizem ferramentas computacionais para monitoramento, controle e suporte de decisões em termos imediatos e estratégicos.

Como temática deste trabalho, é discutida a aplicação de sensor virtual, também denominado sensor inferencial, no contexto da indústria de celulose e papel, com ênfase no monitoramento de emissões em caldeira de recuperação química. Em específico, aplicação de sensores para predição das concentrações de dióxido de enxofre (SO_2) e de material particulado presentes nos gases emitidos por esse equipamento.

2 OBJETIVO GERAL

Desenvolver sensores virtuais baseados em redes neurais artificiais com aplicações em estudos de caso reais, envolvendo emissões em caldeiras de recuperação química de indústrias de celulose *Kraft*.

2.1 OBJETIVOS ESPECÍFICOS

- Executar um pré-processamento dos dados que seja condizente com os objetivos das etapas subsequentes;
- Construir modelo preditivo para a concentração de SO_2, a partir de sensores virtuais baseados em redes neurais e em regressão linear múltipla, para fins de comparação;
- Construir modelo preditivo para o teor de material particulado, a partir de sensores virtuais baseados em redes neurais e em regressão linear múltipla, para fins de comparação;
- Utilizar os modelos de predição baseados em redes neurais associados à técnica de análise de sensibilidade para avaliar quais variáveis exercem mais influência sobre o parâmetro de saída, em cada um dos casos.

3 REVISÃO BIBLIOGRÁFICA

3.1 INDÚSTRIA DE CELULOSE KRAFT

Há aproximadamente 2000 anos, na China, foi desenvolvido o que seria, provavelmente, uma das mais importantes invenções humana: o papel, um depósito aquoso de fibras vegetais e outros materiais em forma de tecido ou camada. O processo se fundamenta na conversão de matéria-prima fibrosa em um volume de fibras livres dissolvidas, em geral fibras celulósicas, seja por processos mecânicos, químicos ou pela combinação de ambos (SÄRKKÄ; GUTIÉRREZ-POCH; KUHLBERG, 2018).

A tecnologia mais importante para fins de registro e comunicação de informações foi o papel, até a expansão dos meios de comunicação digitais. Segundo Särkkä, Gutiérrez-Poch e Kuhlberg (2018), quando essas novas possibilidades foram se popularizando e a preocupação com os rumos da indústria de papel começaram a aparecer, foi observado um crescimento significativo nas demandas para os setores de higiene, saúde e embalagens por exemplo, o que significa que o setor ainda está em alta, a Figura 1 ilustra a mudança de tendência em termos de finalidade de uso do papel. Para que exista uma ampla variedade de produtos, podem ser utilizadas uma variedade de matérias-primas e elaboradas uma gama de possibilidades de modificações nos tratamentos mecânicos e químicos.

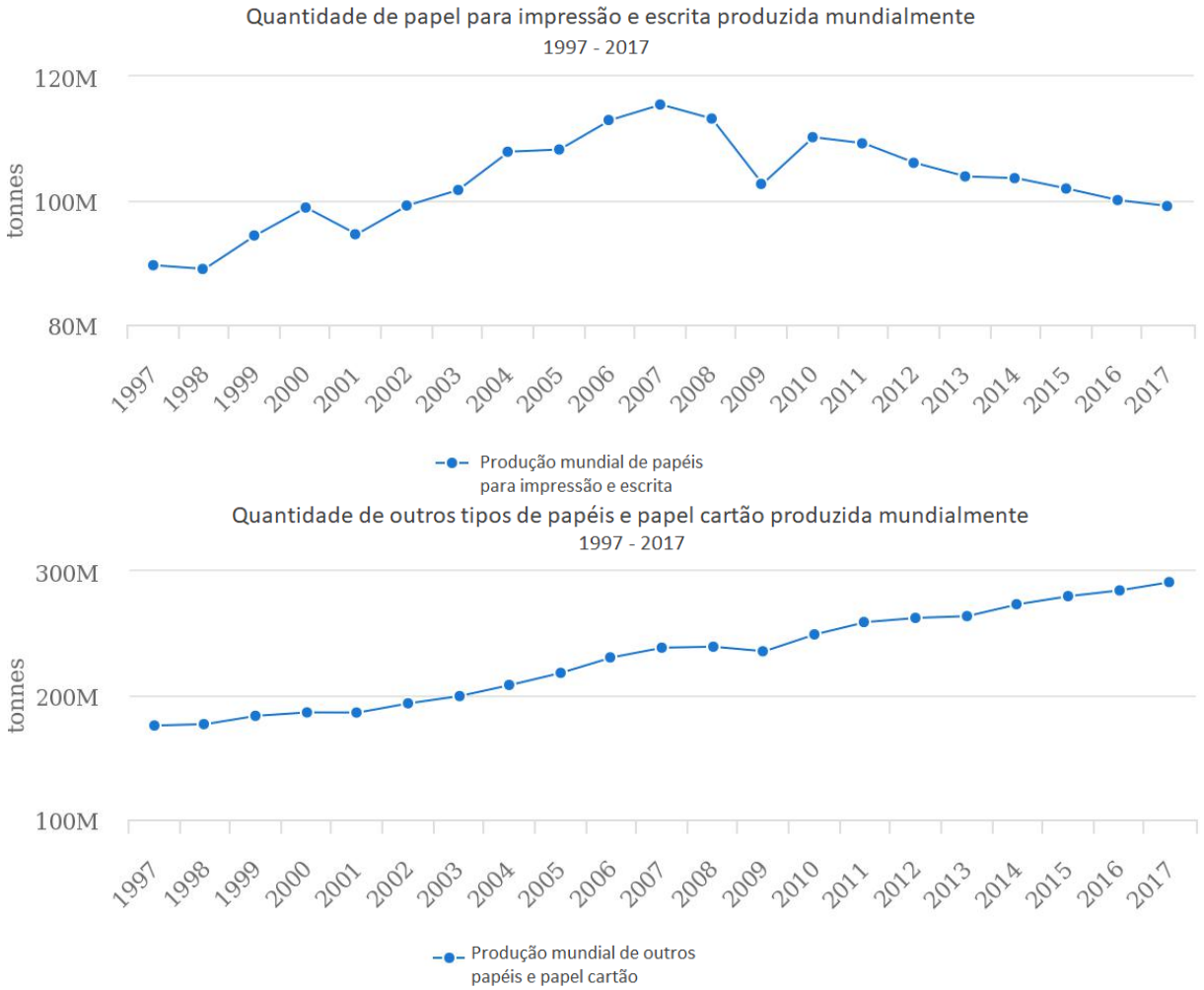


FIGURA 1 – Série histórica 1997 a 2017: Produção mundial de papéis. (a) Papel para impressão e escrita. (b) Outros papéis e papel cartão. Fonte: Adaptado de FAOSTAT (2019)

A produção de papel pode ser dividida em duas fases: a primeira envolve a transformação do material fibroso em polpa de celulose e, a segunda, a conversão dessa polpa em papel. A principal fonte de fibras de celulose são os materiais lenhosos, sendo 90% proveniente da madeira, e o restante provém de fibras de fontes não lenhosas, como resíduos agrícolas, gramíneas e vegetações nativas. A celulose, material fibroso vastamente distribuído em sua forma natural no reino vegetal, é o biopolímero natural mais abundante. O principal processo orgânico de síntese de celulose é a fotossíntese (celulose de plantas), mas existem outros processos, como a produção microbiana de carboidrato extracelular, a bio celulose ou celulose bacteriana (SÄRKKÄ; GUTIÉRREZ-POCH; KUHLBERG, 2018).

Tendo como foco as matérias-primas lenhosas, a primeira etapa de produção de celulose é a deslignificação, que consiste na dissolução da lignina (componente que mantém unidas as fibras celulósicas) para obtenção da massa de fibras livres. Essa etapa, denominada polpação, pode ser realizada por diferentes métodos, como

apresentado na Tabela 1, que indica a relação entre o método de polpação usual, seus rendimentos médios de polpa de celulose em relação à quantidade de madeira, e as principais aplicações de cada produto.

TABELA 1 – Métodos de polpação

Processo	Coloração da Polpa	Rendimento (%)	Aplicações
Polpação termo-mecânica	Marrom	>96	Papelão para caixas, papel de jornal, sacolas de papel
Polpação química, térmica e mecânica	Marrom claro	85-95	Papel de jornal, papéis especiais
Plantas Semi química	Marrom-bege	60-80	Papel de jornal, sacolas de papel
Polpação química – *Kraft*, sulfito	Marrom claro	40-55	Papel de jornal, papéis finos

Fonte: Adaptado de Bajpai (2011)

Métodos de polpação química contabilizam cerca de 70% do total da produção mundial, sendo que cerca de 90% desses executam o processo *Kraft* (utilização de sulfato), o que significa que 64% das indústrias de celulose e papel no mundo operam com fábricas desse tipo. Alguns fatores podem justificar a dominância desse processo: pode ser alimentado por madeiras de baixa qualidade e ainda produzir polpas de ótimas propriedades de resistência. Além disso, a recuperação dos produtos químicos de cozimento, de energia e de subprodutos é eficiente e bem estabelecida (BAJPAI, 2011).

As condições típicas para a polpação *Kraft* incluem elevadas temperaturas, pressões, e alcalinidade (solução aquosa de hidróxido e sulfeto de sódio, NaOH e Na_2S, respectivamente). A seletividade desse processo é muito baixa, e aproximadamente metade dos componentes da madeira bruta se degradam e são dissolvidas no licor de cozimento, denominado licor preto. Dentre as substâncias dissolvidas no licor, cabe destacar fragmentos de lignina degradada, materiais derivados de carboidratos (como ácidos carboxílicos alifáticos) e extrativos. Além do licor preto, alguns subprodutos de destaque da polpação *Kraft* de madeiras de fibra longa (coníferas, *softwood*) são o sulfato de terebintina (*sulphate turpentine*) e o sabão de *tall oil*. A quantidade desses produtos dependem da espécie de madeira utilizada, das condições de crescimento e dos métodos de estocagem (SÄRKKÄ; GUTIÉRREZ-POCH; KUHLBERG, 2018). As etapas de polpação do processo *Kraft* estão ilustradas na Figura 2.

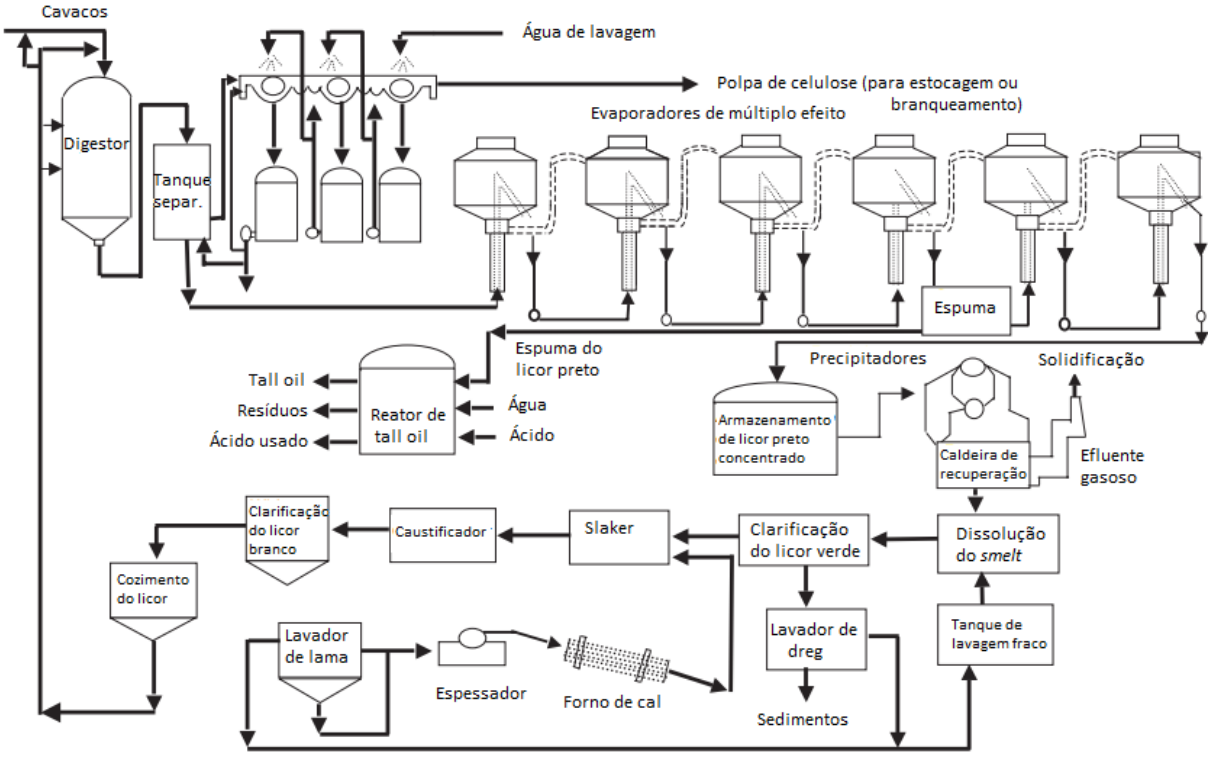


FIGURA 2 – Processo de polpação *Kraft* e os ciclos de recuperação química e energética. Fonte: Adaptado de Bajpai (2011)

A primeira parte do processo consiste na preparação da matéria prima: descascamento das toras de madeira e cominuição para obtenção de cavacos de tamanho uniforme e adequado para entrar no processo. É importante que os cavacos sejam homogêneos, caso contrário poderá ocorrer aumento do consumo de matéria prima e redução da eficiência energética (BAJPAI, 2011). Os cavacos seguem para o digestor onde reagem com o licor branco, solução aquosa de hidróxido e sulfeto de sódio, sob condições de elevadas temperatura (160℃-170℃) e pressão (7 bar), para que os componentes não-celulósicos sejam removidos. O produto resultante passa pelos lavadores, onde a polpa é separada do licor, então chamado licor preto fraco. A polpa, que equivale a, aproximadamente, 45-55% da massa de madeira inicial, segue para a estocagem ou diretamente para o branqueamento e etapas posteriores para fabricação de papel. Em paralelo, o licor preto fraco ingressa no ciclo de recuperação química e energética característicos do processo *Kraft*. Para cada tonelada de polpa produzida são geradas 10 toneladas de licor preto fraco (cerca de 1,5 ton de sólidos secos) (BAJPAI, 2011).

O processo de recuperação química envolve as etapas de concentração do licor preto fraco, combustão dos compostos orgânicos, redução dos inorgânicos e reconstituição do licor branco de cozimento. A Figura 3 resume, esquematicamente, as principais etapas do ciclo de recuperação química, que tem três objetivos principais: minimização dos impactos ambientais dos resíduos (licor preto), reciclagem dos

elementos químicos usados na polpação (NaOH e Na_2S), e cogeração de vapor e eletricidade (TRAN; VAKKILAINEN, 2008).

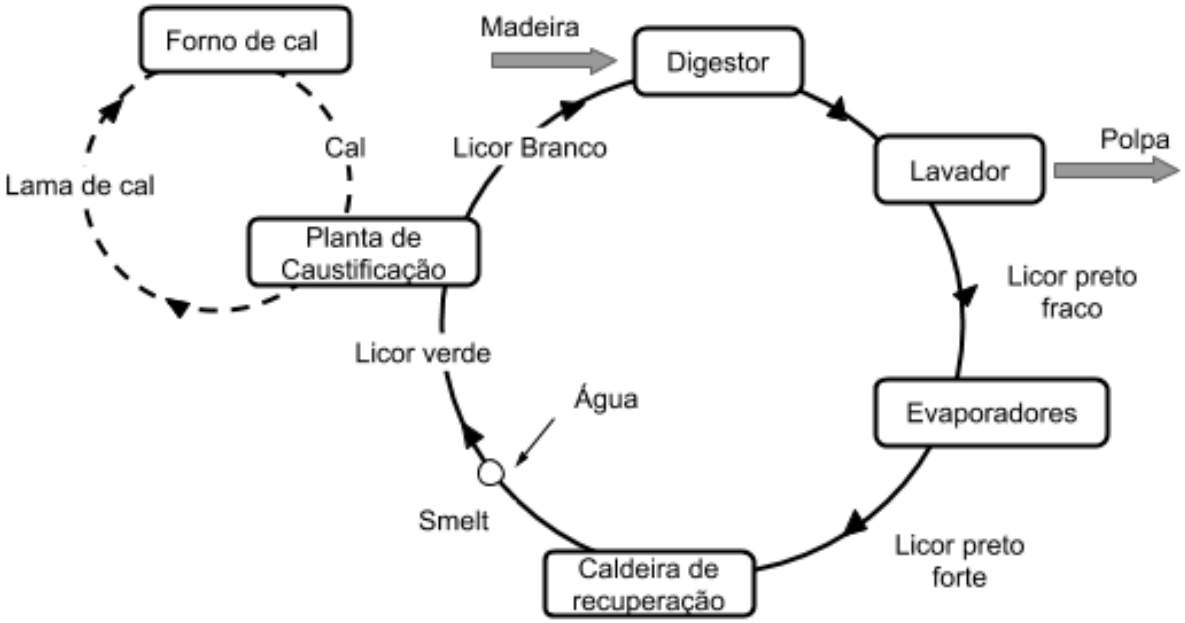


FIGURA 3 – Processo de polpação *Kraft* e os ciclos de recuperação química e energética. Fonte: Adaptado de **tran2008Kraft**

.

O licor preto fraco consiste de uma solução diluída (cerca de 12-15% de sólidos) de fragmentos de lignina, materiais orgânicos, compostos inorgânicos oxidados (sulfato e carbonato de sódio) e licor branco (NaOH e Na_2S). Esta solução segue para os evaporadores de múltiplo efeito e concentradores, onde é concentrada acima de 60% de sólidos e passa a ser denominada licor preto forte. Este último é aspergido na região inferior da caldeira de recuperação e queimado em atmosfera relativamente redutora (deficiente em oxigênio) para que ocorra a queima de orgânicos e a reconstituição do Na_2S. A extensão de formação desse componente caracteriza a eficiência de redução da caldeira, em torno de 90%. A energia gerada pela combustão é usada para geração de vapor e eletricidade. Os efluentes gasosos gerados seguem para o sistema de tratamento correspondente e o efluente fundido, o *smelt*, é despejado em tanque de dissolução onde se adiciona água para a formação do licor verde.

O licor verde consiste em uma mistura aquosa cujas maiores concentrações são de sulfeto e carbonato de sódio (Na_2S e Na_2CO_3, respectivamente). Este licor entra na planta de caustificação, reage com óxido de cálcio (CaO CaO) para que o Na_2CO_3 seja convertido em NaOH. Esse processo é mensurado pela eficiência de caustificação, que, em geral, é em torno de 80% a 83%. O licor resultante é o próprio licor branco (solução aquosa de Na_2S e NaOH), que retorna para o processo de polpação, e o precipitado, constituído, basicamente, por $CaCO_3$ (lama de cal), é lavado e direcionado

para o forno de cal, para a regeneração da cal (CaO).

Segundo (BAJPAI, 2011), apesar da importância do ciclo de recuperação químico e energético para o processo, ele corresponde a cerca de um terço dos custos de capital em uma fábrica de celulose moderna. Consequentemente, são de grande interesse as iniciativas de otimização da combustão do licor e de melhoria da eficiência energética de toda a planta.

3.2 REDE NEURAL ARTIFICIAL

As redes neurais artificiais são modelos computacionais inspirados no sistema nervoso de seres vivos, podendo ser descritas como um conjunto de unidades de processamento (neurônios artificiais) interligados por um grande número de conexões (sinapses artificiais). As suas características mais relevantes são: adaptação dos parâmetros internos por experiência; capacidade de aprendizado a partir de métodos de treinamento; habilidade de estimar soluções que eram desconhecidas (capacidade de generalização); organização de dados por agrupamento conforme características comuns; tolerância a falhas quando parte da estrutura interna da própria rede é sensivelmente corrompida; armazenamento distribuído, o que leva à robustez da arquitetura; e facilidade de prototipagem, já que, em geral, as operações matemáticas envolvidas são elementares. Existem várias possibilidades de arranjos dos neurônios em uma rede neural. Tais configurações são denominadas arquiteturas. Alguns exemplos de arquitetura são: *feedforward* em camada simples, *feedforward* em camadas múltiplas, recorrente ou realimentada e em estrutura reticulada. Cada arquitetura permite uma infinidade de topologias distintas, associadas às variadas composições estruturais definidas pelo número de neurônios em cada camada e pelas funções de ativação utilizadas (SILVA; SPATTI; FLAUZINO, 2010).

Um exemplo de arquitetura *feedforward* em camada simples é a rede Perceptron, composta por um único neurônio, é configurada por n sinais de entrada representativos do problema, e apenas uma saída. Cada uma das entradas é ponderada por um peso sináptico, e o valor resultante da composição de todas as entradas ponderadas, adicionado o limiar de ativação, é enviado para a função de ativação, cujo resultado é a saída produzida pelo *Perceptron*. Cabe ressaltar que o ajuste dos pesos e a determinação do limiar de ativação são realizados por processo de treinamento supervisionado, ou seja, durante o treinamento, para cada sinal de entrada, é informada a respectiva saída (SILVA; SPATTI; FLAUZINO, 2010).

Por ser composta por apenas um neurônio, a rede Perceptron apresenta algumas limitações, como a não convergência em casos de problemas não linearmente separáveis. Para aumentar a capacidade de resolução de problemas mais complexos, foram desenvolvidas as redes Perceptron de múltiplas camadas (*Multilayer Perceptron*,

MLP), altamente versáteis em termos de aplicabilidade. As principais áreas de aplicação desse tipo de rede incluem aproximação de funções, reconhecimento de padrões, identificação e controle de processos, previsão de séries temporais e otimização de sistemas. A propagação dos sinais de entradas desse tipo de rede é sempre realizada no mesmo sentido: a partir da camada de entrada em direção à camada neural de saída. Dado que o processo de treinamento dessa rede é supervisionado, os ajustes dos pesos e dos limiares são conduzidos pelo algoritmo de aprendizado, como o algoritmo de retro propagação do erro ou *backpropagation*. Em suma, os sinais de entrada se propagam adiante (*forward*), camada por camada, em direção à saída. As respostas produzidas pelas saídas são comparadas com as respostas existentes e os desvios (erros) são calculados. Em função dos valores destes erros, o mecanismo de retro propagação é executado de forma a ajustar os pesos sinápticos e os limiares de cada neurônio. Desse modo, redes MLP são capazes de predizer valores ou classes de variáveis a partir da combinação não linear de variáveis de entrada (SILVA; SPATTI; FLAUZINO, 2010).

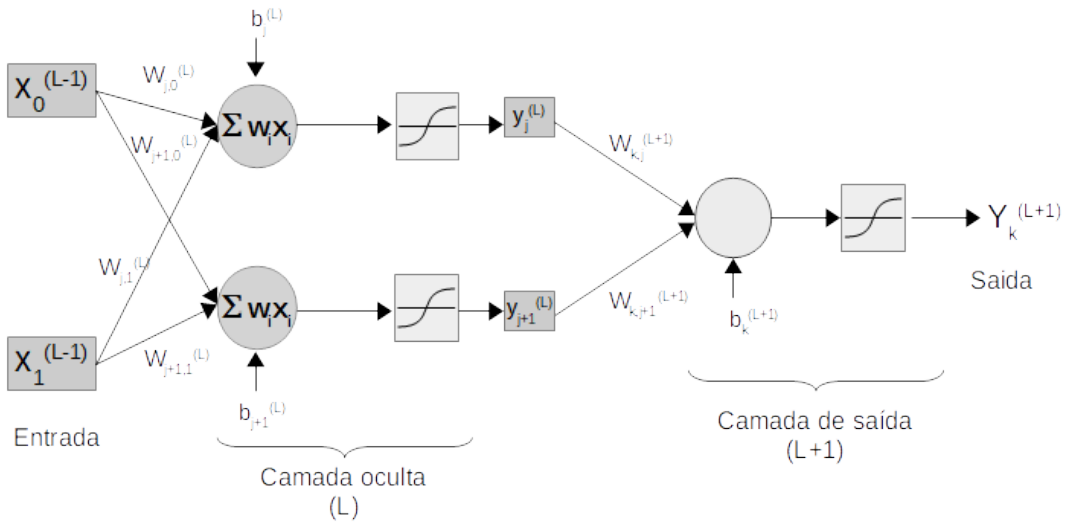


FIGURA 4 – Esquema de uma rede perceptron de múltiplas camadas, com dois neurônios na camada oculta. Fonte: Adaptado de Lugade (2011)

Conforme representado na Figura 4, $w_{j,i}^{(L)}$ indica o peso sináptico conectando a entrada i (ou saída do i-ésimo neurônio da camada L-1) ao j-ésimo neurônio da camada L, $b_j^{(L)}$ corresponde ao limiar de ativação (também denominado por *bias*) e $y_j^{(L)}$

corresponde à saída do j-ésimo neurônio da camada L. Inicialmente as observações de treinamento são alimentadas diretamente à primeira camada de neurônios ocultos. Para cada neurônio (ou unidade) é calculada uma entrada ponderada, $I_j^{(L)}$, dada pela combinação linear entre valores das entradas e seus pesos correspondentes, adicionado ao limiar $b_j^{(L)}$, que funciona como limite para variar a atividade do neurônio, conforme a Equação 3.1.

$$I_j^{(L)} = \sum_i (w_{ji}^{(L)} x_i) + b_j^{(L)} \tag{3.1}$$

Sobre a entrada ponderada ($I_j^{(L)}$) é aplicada uma função de ativação, com o objetivo de limitar a saída do neurônio a um intervalo de valores razoáveis. Usualmente, as funções de ativação são diferenciáveis, não lineares e com imagem limitada. O tipo mais usado é o sigmoide, com destaque para as funções logística e tangente hiperbólica, representadas na Figura 5 (ALMEIDA et al., 2010). Sendo a função de ativação representada por $g(.)$, a saída $y_j^{(L)}$ do neurônio é calculada conforme a Equação 3.2.

$$y_j^{(L)} = g(I_j^{(L)}) \tag{3.2}$$

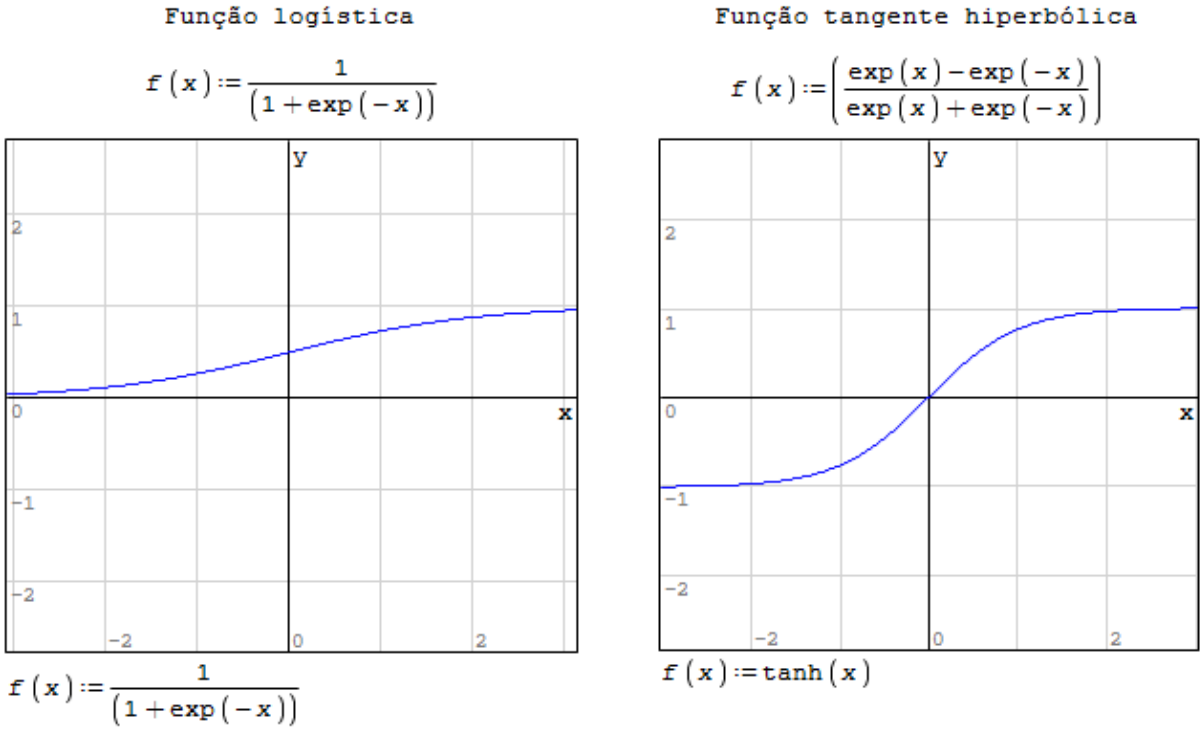


FIGURA 5 – Representação gráfica das funções logística e tangente hiperbólica. Fonte: Autoria própria

Após a realização dos cálculos em direção à saída (*forward*), são ajustados os pesos e limiares a partir da propagação do erro, no sentido da saída para a entrada

(*backward*). Esse procedimento, iniciado pela camada de saída, é conduzido pela minimização do erro em relação a dada amostra de treinamento para o k-ésimo neurônio da camada de saída. O ajuste da matriz de pesos é realizado no sentido oposto ao gradiente do erro, considerando cada unidade neuronal.

$$Err_j^{(L+1)} = 1/2 * \left(T_k - y_j^{(L+1)}\right)^2 \tag{3.3}$$

$$\frac{\partial Err_j^{(L+1)}}{\partial w_{ji}^{(L+1)}} = -\left(T_k - y_j^{(L+1)}\right) * g'\left(I_j^{(L+1)}\right) * y_j^{(L)} \tag{3.4}$$

$$\frac{\partial Err_j^{(L+1)}}{\partial b_{ji}^{(L+1)}} = -\left(T_k - y_j^{(L+1)}\right) * g'\left(I_j^{(L+1)}\right) * (-1) \tag{3.5}$$

Conforme as Equações 3.3, 3.4 e 3.5, Err_k é o erro de predição do k-ésimo neurônio da camada de saída (no caso, denominada L+1), T_k é o valor referente à informação da variável de saída (o aprendizado é supervisionado), $y_k^{(L+1)}$ é a estimativa dada pelo neurônio de saída, e $y_j^{(L)}$ corresponde às entradas do neurônio de saída (equivalente às saídas dos neurônios da camada L).

$$\begin{gathered}\Delta w_{ji}^{(L+1)} = -\eta \frac{\partial Err_j^{(L+1)}}{\partial w_{ji}^{(L+1)}} \quad \rightarrow \quad \Delta w_{ji}^{(L+1)} = \eta \delta_k^{(L+1)} y_j^{(L)} \quad , \text{com} \\ \delta_k^{(L+1)} = \left(T_k - y_j^{(L+1)}\right) * g'\left(I_j^{(L+1)}\right)\end{gathered} \tag{3.6}$$

$$\Delta w_{ji}^{(L+1)_{current}} \leftarrow w_{ji}^{(L+1)_{previous}} + \Delta w_{ji}^{(L+1)} \tag{3.7}$$

Os pesos w_{ji} são atualizados em reflexo à propagação do erro conforme as Equações 3.6 e 3.7, em que é a taxa de aprendizagem do algoritmo de propagação reversa, utilizada com o propósito de encontrar o mínimo global, evitando que a solução fique presa nos mínimos locais, e $_k^{(L+1)}$ é definido como o gradiente local em relação ao k-ésimo neurônio da camada de saída. Os limiares b_j são calculados de forma análoga conforme apresentado pelas Equações 3.8 e 3.9.

$$\begin{gathered}\Delta b_{ji}^{(L+1)} = -\eta \frac{\partial Err_j^{(L+1)}}{\partial b_{ji}^{(L+1)}} \quad \rightarrow \quad \Delta b_{ji}^{(L+1)} = \eta \delta_k^{(L+1)} * (-1) \quad , \text{com} \\ \delta_k^{(L+1)} = \left(T_k - y_j^{(L+1)}\right) * g'\left(I_j^{(L+1)}\right)\end{gathered} \tag{3.8}$$

$$\Delta b_{ji}^{(L+1)_{current}} \leftarrow b_{ji}^{(L+1)_{previous}} + \Delta b_{ji}^{(L+1)} \tag{3.9}$$

Após o ajuste dos parâmetros da camada de saída, o erro é retropropagado para as camadas anteriores, tendo como referência de erro a diferença entre os valores obtidos inicialmente e aqueles ajustados pela camada posterior. Ao final do treinamento, tem-se o conjunto de pesos e limiares ótimos que representam o problema especificado pelo conjunto de dados de treinamento. A partir de então, é possível inferir estimativas para as variáveis de saída, dado um conjunto de observações desconhecidas, proveniente do mesmo problema usado no treinamento (HAN; PEI; KAMBER, 2011).

Cabe ressaltar que a decisão acerca do número de neurônios em cada camada oculta bem como o número de camadas necessários para descrever apropriadamente os dados é empírica. Em geral, as redes começam com uma topologia mais simplificada e vão crescendo em complexidade até que o grau de exatidão desejado seja atingido (SOUZA; ARAÚJO; MENDES, 2016). Cabe ressaltar que o aumento da complexidade pode levar a sobretreinamento (modelo se especializa excessivamente no subconjunto de identificação, *overtraining*) e redução da capacidade de generalização (ALMEIDA et al., 2010).

3.3 SENSOR VIRTUAL BASEADO EM REDE NEURAL PARA EMISSÕES EM GERAL E SETOR DE CELULOSE EM PARTICULAR

Em 1984, Uronen publicou um artigo discutindo a expansão da presença de computadores, sensores e de sistemas de controle digitais no cenário da indústria de papel e celulose. Segundo esta publicação, haveria o aprimoramento dos conceitos de controle descentralizado e sistemas de comunicação e supervisão centralizados. Desse modo, sistemas de instrumentação digital tratariam os níveis mais baixos da hierarquia de controle e, a partir de uma rede de dados local seria construída uma estrutura hierárquica para toda a unidade fabril, incluindo controle e coordenação de área, planejamento e programação da produção, bem como sistemas de gerenciamento de informação a níveis mais elevados (URONEN, 1984).

Neste sentido, uma questão importante existente no setor industrial é a ausência de medições em tempo real de características de produtos e processos. Medições infrequentes podem causar atrasos na tomada de decisões, atrasando execução de ações corretivas e/ou de melhoria da operação, podendo gerar, por exemplo, ao longo de todo o tempo de atraso, produtos fora de especificação, desperdícios, aumento de consumo energético e amplificação de impactos ambientais. Os sensores virtuais podem contribuir em vários pontos para a minimização de problemas desse tipo, já que são construídos para predizer valores para as variáveis difíceis de medir, à mesma taxa de medição das variáveis medidas *online*. Desse modo, podem portanto ser incluídos em sistemas de controle e de monitoramento, contribuindo para a identificação precoce

de falhas e a transmissão da informação para sistemas de análise e produção de diagnóstico relativos a distúrbios no processo (SOUZA; ARAÚJO; MENDES, 2016).

Segundo Champagne, Amazouz e Platon (2005), alguns fatores são determinantes para uma implementação bem-sucedida de sensores virtuais:

- Bom motivador para o negócio (redução de custos, eliminação da produção fora da especificação, minimização de impactos socioambientais negativos);
- Especialista em tecnologia de sensores virtuais;
- Especialista no processo onde o sensor será implementado;
- Infraestrutura adequada para a coleta de dados de processo (exemplo: sistema de controle distribuído, controlador lógico programável conectado ao histórico de dados);
- Taxa de amostragem adequada para o processo;
- Bom sistema de amostragem, para que as amostras do parâmetro a ser medido sejam representativas do processo;
- Conjunto de dados que representem o estado de operação estável e usual para o desenvolvimento e a validação do sensor;
- Bom padrão de respostas para desenvolvimento do sensor;
- Processo de produção relativamente estável em termos de projeto (processos que não sejam redesenhados constantemente);

As legislações que regem as possibilidades de monitoramento de emissões gasosas, em geral, permitem a utilização de sensores inferenciais (sensores virtuais) para a apresentação dos resultados de monitoramento, desde que o modelo apresente um coeficiente de correlação acima de um valor específico. No Brasil, a Resolução CONAMA n° 436, de 22 de dezembro de 2011, no Anexo XIV, possibilita aplicação de diferentes métodos contínuos de monitoramento (CONAMA; MMA, 2011).

Nos Estados Unidos, o uso de sensores virtuais é permitido e regulamentado por lei, sendo obrigatória a existência de mecanismos de validação do sensor. Uma das maneiras de realizar esse processo ocorre por ferramentas de autovalidação, envolvendo etapas de detecção de falhas dos sensores (especialmente, sensores físicos), identificação e reconstrução de sensores defeituosos, construção dos modelos inferenciais e detecção de extrapolações. Qin, Yue e Dunia (1997) propõem um sensor inferencial com mecanismo de auto validação para a predição do teor de NO_x emitido por uma caldeira industrial. São selecionadas oito variáveis relacionadas

com as emissões de NO_x. As amostras são divididas entre conjunto de treinamento e conjunto de teste, e um modelo de análise por componentes principais (*Principal component analysis*, PCA) é construído para identificação e reconstrução de falhas nos sensores, seguida de extrapolação para valores considerados adequados. Posteriormente, são comparados os métodos: regressão linear por componentes principais (*Principal components regression model*, PCR) e rede neural associada à regressão por componentes principais (*neural net PCR*). Como resultados dessa integração, foi observado que a identificação e reconstrução de sensores defeituosos foi efetiva, e que a predição não é confiável quando se utilizam valores calculados por extrapolação. Em termos gerais, os resultados foram positivos e capazes de justificar a economia que se pode alcançar aumentando-se o uso de sensores virtuais em detrimento do uso de análises de laboratório ou de analisadores que exijam manutenção e calibração muito frequentes.

A maioria das aplicações industriais de sensores virtuais são baseadas em redes neurais artificiais (RNA), modelos, em geral, do tipo “caixa-preta”, que consideram relações de não linearidade entre os conjuntos de variáveis de entrada e de saída de sistemas complexos utilizando dados históricos. Deste modo, dados de entrada e de saída característicos de uma operação padrão podem ser utilizados para o treinamento de uma rede neural. Na sequência, para um processo em particular, sempre que novos dados de entrada são apresentados à uma rede treinada, espera-se que ela seja capaz de inferir, com certo grau de precisão, o valor correspondente à saída. O resultado é um sensor virtual funcional para a previsão de variável de processo (CHAMPAGNE; AMAZOUZ; PLATON, 2005).

Algumas aplicações industriais de sensores virtuais baseados em redes neurais são apresentados na Tabela 2.

Descrevem-se, a seguir, aplicações de sensores virtuais, baseados em redes neurais, com o foco em emissões em geral. Dong, McAvoy e Chang (1995) propuseram um modelo de monitoramento de emissões de NO_x a partir da construção de sensor virtual baseado em rede neural de mínimos quadrados parciais (*neural network partial least squares*, NNPLS) seguido da aplicação de análise por componentes principais não linear (*nonlinear principal component analysis*, NLPCA), cuja base de dados é proveniente de um aquecedor industrial. Os resultados da aplicação do NNPLS são comparados com resultados obtidos pela aplicação do método dos mínimos quadrados parciais (*partial least squares*, PLS), sendo utilizado para avaliar as diferenças o parâmetro erro quadrático de predição (*squared prediction error*, SPE) e a estatística soma dos quadrados do erro residual de predição (*predicted residual error sum of squares*, PRESS). É verificado que a rede neural oferece resultados melhores que a abordagem linear.

TABELA 2 – Aplicações industriais típicas de sensores virtuais baseados em RNA

Indústria	Variáveis preditas	Benefícios
Ambiental: Monitoramento e controle de emissões	Oxigênio nos gases de exaustão, CO, Nox, SO2, CO2, Opacidade. Aplicações: gás, turbina de vapor, caldeiras e fornalhas.	Metade do investimento inicial comparado aos analisadores físicos, redução dos custos de manutenção (em geral em 1/3), aumento da eficiência da produção.
Polímeros	Índice de fusão, densidade, índice isotático, concentrações de co-monômero e monômero, fluxo de catalisador, concentração de polímero e tempo de residência.	Aumento na eficiência do processo, melhor controle de qualidade, redução de desperdícios.
Plantas químicas	Composição do produto, estimação de impurezas, em reatores e na destilação.	-
Alimentos e bebidas	Massa e umidade dos alimentos produzidos. Teor de solventes nos produtos, propriedades dos materiais pulverizados.	-
Celulose e papel	Número Kappa, viscosidade, grau de branqueamento, força, rigidez, módulo E, rugosidade, opacidade, porosidade, área superficial específica, umidade, testes químicos e físicos.	Aumento de produção, redução nos tempos de transição, otimização do uso de energia, monitoramento preditivo de emissões na caldeira, predição de falhas.
Refino	Pontos de ebulição, número de octanagem, destilação, pontos flash, viscosidade de congelamento, pontos de corte, qualidade do destilado, resíduos de fundo, produção de H2S (plantas de enxofre), produção de intermediários.	-

Fonte: Adaptado de Champagne, Amazouz e Platon (2005)

Tronci, Baratti e Servida (2002) construíram um sensor virtual para monitoramento simultâneo das concentrações de monóxido de carbono (CO), óxidos de

nitrogênio (NO_x) e oxigênio (O_2), em emissões de caldeira termoelétrica com potência nominal de 4,8 MW. As variáveis de entrada utilizadas foram: percentual de ar em excesso, fluxos de cada tipo de combustível, razões entre ar superior (*ovefire air*, OFA) e ar total, fluxo de ar primário, fluxo de ar secundário, temperatura de entrada do ar, temperatura de saída dos fumos, e teor de oxigênio na câmara de combustão. Baseou-se a seleção de variáveis de entrada em conhecimentos e experiências em relação ao processo. Antes da construção do modelo, foi aplicado um filtro para remoção de dados anômalos, e os dados de entrada foram normalizados em escala [-1,1] e os dados de saída, em escala [0,1]. Os resultados foram satisfatórios, com erros de predição inferiores a 10%, para 95% dos dados de teste.

Uma caldeira de queima tangencial, com potência nominal de 600 MW, foi o equipamento de estudo de Zhou, Cen e Fan (2004) para a construção de sensores virtuais para monitoramento da concentração de NO_x nos gases de combustão e das características de queima do carbono (tendo como referência o teor de carbono não queimado). Foram utilizadas 29 variáveis de entrada, sendo 23 características da operação da caldeira (fluxo de combustível, fluxo de ar total, taxa de alimentação do carvão, posição do amortecedor do queimador de ar secundário, posição do amortecedor da entrada do ar superior (*overfire*), queda de pressão entre fornalha e exaustor, concentração de O_2 nos gases de saída, fluxo de ar primário, inclinação do pulverizador), e 6, do carvão (percentual de carbono, percentual de hidrogênio, percentual de oxigênio, percentual de nitrogênio, percentual de voláteis, capacidade calorífica). Os sensores virtuais foram modelados a partir de uma rede neural artificial, com 31 neurônios na camada oculta e 1 neurônio de saída (correspondente à variável a ser inferida). Os resultados são comparados com um modelo em fluidodinâmica computacional (*Computational Fluid Dynamics*, CFD). Conclui-se que a modelagem da rede neural é mais simples, direta e oferece resultados precisos para a predição de NO_x.

Uma abordagem de seleção de variáveis interessante foi proposta por Shakil et al. (2009) em um estudo envolvendo caldeira de tubos de água com queima de gás natural misturado com outros combustíveis. A metodologia de análise por componentes principais (*principal component analysis*, PCA) foi aplicada para a seleção das componentes que representam as seis medições de temperatura de parede da caldeira, a serem utilizadas como variáveis de entrada, junto às variáveis: fluxo de gás natural, fluxo de combustíveis secundários e fluxo de ar. As variáveis inferidas foram as concentrações de NO_x e de O_2 nos gases efluentes. Todas as variáveis foram normalizadas em escala [0,1]. Dois modelos de predição foram aplicados: uma rede neural estática e uma rede neural dinâmica. Para o modelo estático, os resultados de treinamento se apresentaram melhores que os de teste, indicando reduzida capacidade de generalização. Para o modelo dinâmico, o resultado geral, em termos de capacidade de generalização, foi satisfatório. O erro quadrático médio para o conjunto de validação,

para o modelo estático, foi de $4.8 * 10^{-3}$ (predição de NO_x) e $4.4 * 10^{-3}$ (predição de O_2; e para o modelo dinâmico de $9.9 * 10^{-4}$, predição de NO_x, e $1.1 * 10^{-3}$, predição de O_2 (SHAKIL et al., 2009).

Outra proposta de estudo para a construção de modelos de predição de variáveis se baseia na simulação de determinado equipamento e utilização dos dados gerados. Iliyas et al. (2013) aplicaram CFD em 3D para a modelagem de uma caldeira de 160 MW de queima de gás natural, com tubos de água e dois queimadores verticais alinhados. A simulação abrangeu diferentes condições operacionais. Os dados gerados pela simulação foram utilizados para treinar e testar dois tipos de redes neurais: Perceptron de camadas múltiplas (*multilayer perceptron*, MLP) e função de base radial (*radial basis functions*, RBF). São testados modelos com 6 e 8 variáveis de entrada: razão ar-combustível, fluxo de combustível, fluxo de ar, temperatura máxima da câmara de combustão, temperatura média da câmara de combustão, temperatura de saída dos gases, temperatura de entrada do ar e ângulo do redemoinho (*swirl angle*). Ressalta-se que o modelo com 6 entradas se mostrou mais preciso. As variáveis a serem inferidas foram as concentrações de NO_x e O_2 nos efluentes gasosos. Os resultados indicaram que a rede RBF alcançou resultados mais consistentes e reprodutíveis que a rede MLP.

Ramakalyan et al. (2016) propuseram uma metodologia híbrida para classificação e estimação de efluentes gasosos de caldeira, a partir de dados obtidos dos sensores instalados na própria linha de efluentes gasosos. Os dados provenientes dos sensores passaram por etapas de pré-processamento, foram normalizados e as observações semelhantes foram agrupadas pela técnica de agrupamento K-means (*K-means clustering*). Em seguida, é feita a classificação dos dados utilizando máquinas de vetores de suporte (*support vector machines*, SVM), com núcleo de função de base radial (*radial basis function kernel*). Os vetores de suporte são utilizados para treinar a rede neural de regressão generalizada (*generalized regression neural networks*, GRNN), e por fim, obter as estimativas de concentração de cada gás presente no efluente (RAMAKALYAN et al., 2016).

3.4 ANÁLISE DE SENSIBILIDADE

Um contraponto à aplicação de redes neurais com intenção de predição está no fato de o modelo não indicar, dentre as variáveis de entrada, aquelas que mais contribuem para o resultado de saída. Tendo em vista monitoramento e controle de processos, compreender as influências entre variáveis de sistemas complexos e não-lineares é de grande interesse (YEH; CHENG, 2010).

Segundo Papadokonstantakis, Lygeros e Jacobsson (2006) existem, basicamente, três categorias de abordagens para explicitar as relações entre variáveis de entrada e de saída, as diferenças entre tais categorias estão no momento de aplicação

da análise de sensibilidade. A primeira categoria envolve metodologias aplicadas sobre o conjunto de dados a ser considerado, antes da criação do modelo, em geral são empregadas técnicas multivariadas (como PCA e NLPCA) em conjunto com informações técnicas. A segunda, envolve modificações do modelo realizadas durante o procedimento de treinamento, que alia conhecimentos a priori à modelagem neural. A intenção desse tipo de modelagem "híbrida" está na consideração de conhecimentos do processo para o ajuste dos parâmetros internos da rede, usualmente essa abordagem utiliza métodos baseados em estruturas Bayesianas para a determinação automática de relevância das variáveis de entrada durante o procedimento de treinamento. Por fim, a terceira abordagem envolve aplicação de técnicas de análise de sensibilidade após as etapas de treinamento e validação, como exemplo podem ser citados o cálculo de medida de influência geral (*general influence measure*, GIM), zerar sequencialmente os pesos de cada variável de entrada (*sequentially zeroes the weights of each input variable*, SZW) e técnicas de variação sequencial das variáveis (*sequential varying of variables technique*, SVV).

Papadokonstantakis, Lygeros e Jacobsson (2006) comparam quatro diferentes abordagens de análise de sensibilidade a fim de classificar, em termos gerais, a importância das variáveis de entrada para a saída do modelo de redes neurais. Os modelos foram implementados em quatro conjuntos de dados artificiais, baseados em diferentes funções algébricas, e um conjunto de dados reais, referente ao processo de craqueamento catalítico de uma refinaria de petróleo. Como resultado do estudo, foi concluído que dependendo do tamanho e da complexidade do conjunto de dados, modelos diferentes podem gerar resultados bastante distintos, sendo que as tendências das variáveis de influência podem variar conforme o modelo. Nesse sentido, uma alternativa proposta por esse estudo seria a combinação de diferentes modelos escolhidos conforme critérios qualitativos e quantitativos.

No caso do presente trabalho, o interesse está na indicação das variáveis que mais influenciam o resultado da regressão, considerando que o modelo neural já foi treinado e validado. Portanto os exemplos que se seguem se classificam dentro dessa abordagem de interesse.

No campo da ecologia, Gevrey, Dimopoulos e Lek (2003) propõem um modelo de predição da densidade de desova de trutas marrons considerando como variáveis de entrada características do habitat. Após a obtenção de um modelo de predição adequado, os autores se preocupam em apresentar as contribuições relativas dos fatores de entrada, comparando sete métodos distintos de análise: (i) método PaD (de Derivadas Parciais), consiste no cálculo das derivadas parciais da saída conforme cada uma das entradas; (ii) método dos "Pesos", calculado com base nos pesos neuronais; (iii) método "*Perturb*", envolvendo a perturbação das variáveis de entrada; (iv) método

"*Profile*", obtido pela variação sucessiva de uma variável de entrada enquanto as outras são mantidas constantes em um valor fixo; (v) método "*classical stepwise*"que observa a alteração no valor do erro quando variáveis de entrada são adicionadas ou eliminadas; (vi) método "*improved stepwise a*", semelhante ao "*classical stepwise*"sendo que a eliminação de variáveis ocorre durante o treinamento da rede neural; (vii) método "*improved stepwise b*", envolvendo o treinamento da rede com a substituição de cada variável de entrada pela média para verificar as consequências sobre o erro. Como resultado, o método de derivadas parciais (PaD) se mostrou o mais eficiente por obter os resultados mais completos, seguido pelo método "*profile*" com o perfil de contribuição das variáveis de entrada.

Em 1999, Scardi; Harding Jr propuseram um modelo para estimar a produção primária de fitoplânctons duas redes *perceptron* em camadas múltiplas (MLP), uma com três e a outra com doze variáveis de entrada. Após o treinamento, validação e teste do modelo, com obtenção de resultados melhores comparados aos de modelos empíricos tradicionais, foi aplicado um método de análise de sensibilidade com o objetivo de verificar o efeito de pequenos distúrbios sobre cada uma das variáveis de entrada sobre a saída da rede. Com esses resultados foi possível compreender com mais profundidade o processo a ser modelado e os mecanismos da rede, bem como analisar efeitos de primeira ordem decorrentes da perturbação de variáveis sobre a saída da rede neural.

De modo específico, em caldeira de recuperação química, Almeida et al. (2010) construíram um mapa das relações de causa e efeito de cada variável de processo sobre o vapor produzido por uma caldeira de recuperação química. Inicialmente, eles propuseram um modelo neural de predição da produção de vapor, em seguida, com um modelo adequado, conduziram estudo de análise de sensibilidade a fim de conhecer as influências entre as variáveis de entrada e a de saída e, com isso, construir o mapa de causa e efeito. O método de perturbação de variáveis foi utilizado para essa análise, observando os desvios do erro quadrático médio na saída da rede neural conforme a perturbação de cada uma das variáveis preditoras.

4 DESCRIÇÃO DOS CASOS DE ESTUDO

Conforme apresentado na Introdução, caldeiras de recuperação química em indústrias *Kraft* de celulose são responsáveis por recuperar, de forma eficiente, energia e componentes químicos, que retornarão ao processo na forma de reagentes. No processo de recuperação química, o ideal é obter a combustão completa dos componentes orgânicos e, simultaneamente, uma elevada taxa de redução dos componentes inorgânicos. Como processos de oxidação e redução são inversos, não é simples alcançar a alta eficiência de ambos (VAKKILAINEN, 2016).

A Figura 6 ilustra os principais componentes de uma caldeira de recuperação química convencional. O liquor preto concentrado é pulverizado na região inferior da fornalha, as gotículas do liquor passam por um processo de secagem acelerada, seguida da volatilização de compostos orgânicos e inorgânicos e da queima de alguns componentes orgânicos. Os materiais resultantes se depositam no leito, onde as reações de oxidação de orgânicos e redução de inorgânicos se realizam. Os componentes gasosos provenientes do leito são queimados pelo ar secundário, que exerce as funções de manter estável a altura do leito assim como controlar a temperatura da fornalha. Em conjunto com o ar secundário, muitas vezes também ocorre a queima de gases não condensáveis diluídos (*Diluted Non-Condensable Gas*, DNCG), geralmente contendo compostos com nitrogênio e enxofre, e alguma umidade, provenientes de outros equipamentos da fábrica. Abaixo do ar secundário, é injetado o ar primário, uniformemente distribuído para criar simetria no leito e manter a região de adequada para a redução dos compostos inorgânicos. Os inorgânicos reduzidos e fundidos saem no fundo da caldeira e são denominados *smelt*. Em região superior as lanças do liquor, o ar terciário é introduzido com o objetivo de finalizar a queima dos combustíveis remanescentes da fornalha inferior, principalmente pela promoção do aumento da taxa de mistura na região gasosa. Os gases resultantes da queima e, eventualmente, algum material particulado carreado pelo fluxo gasoso seguem para a região de troca térmica, onde a energia térmica proveniente da fornalha e da fase gasosa é utilizada para geração de vapor superaquecido, seja para suprimento de energia térmica para outras utilidades ou para geração de energia elétrica (VAKKILAINEN, 2005; 2016).

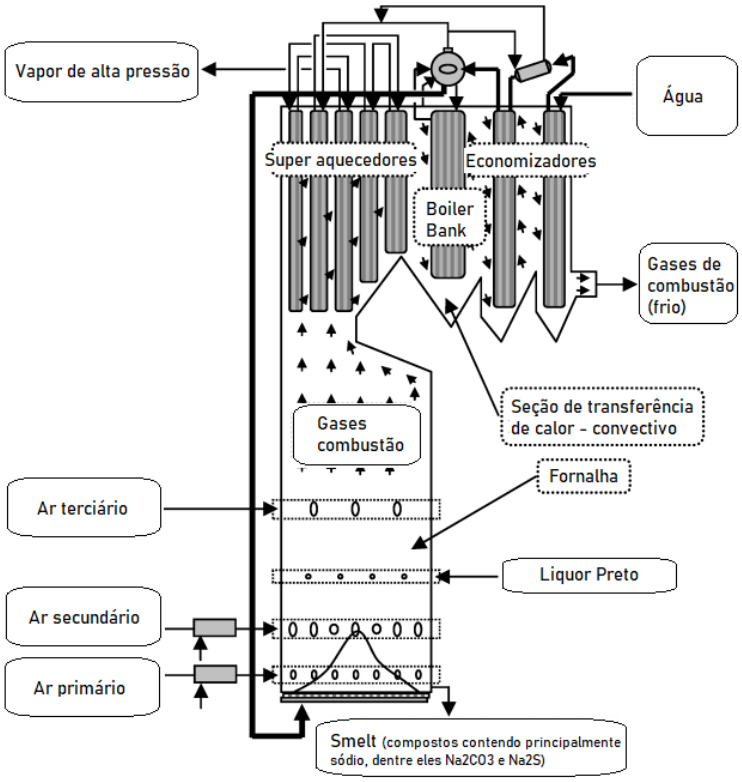


FIGURA 6 – Diagrama esquemático de uma caldeira de recuperação química. Fonte: Adaptado de Almeida et al. (2010)

Ressalta-se, ainda, que é muito desejável atingir rendimento térmico elevado; produzir cinzas mais limpas, que não produzam incrustações; gerar emissões menos poluentes; e ser compatível com as regulamentações ambientais. Em resumo, o processo é bastante complexo e envolve variadas reações simultâneas, conforme apresentado na Figura 7.

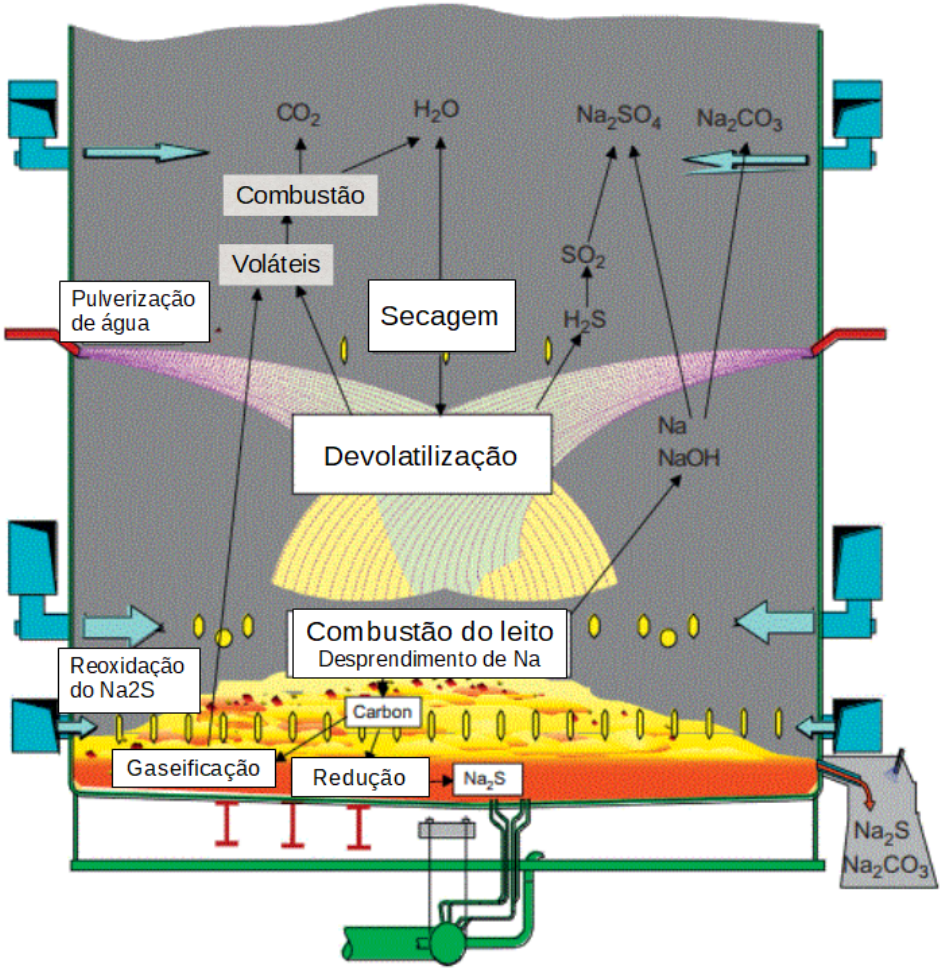


FIGURA 7 – Exemplos de reações na fornalha. Fonte: Adaptado de Vakkilainen (2016)

Além de energia, dois são os produtos principais da caldeira: o *smelt* e os gases de combustão. O *smelt* contém os produtos inorgânicos das reações que acontecem na fornalha, sendo o objetivo com relação ao *smelt* alcançar elevada concentração de sulfeto de sódio (Na_2S) em razão ao sulfato de sódio (Na_2SO_4). Neste contexto, a sulfidez é um parâmetro importante a ser avaliado ($sulfidez = S_{total}/(Na_2+K_2)$), já que o seu aumento indica problemas na operação da caldeira, especialmente relacionados ao aumento das emissões de gases contendo enxofre e, em particular, as emissões de SO_2. Em relação ao monitoramento das emissões de material particulado na corrente gasosa, a sua importância diz respeito ao controle das incrustações nos trocadores de calor e ao consumo de energia pelo precipitador eletrostático (VAKKILAINEN, 2005). Neste trabalho, construiu-se um sensor virtual para SO2, para uma caldeira em operação no Brasil, e um outro sensor virtual para material particulado, para uma caldeira em operação na Finlândia.

Óxidos de enxofre (SO_x) são óxidos ácidos que em condição ambiente se encontram no estado gasoso. A forma predominantemente encontrada é o dióxido

de enxofre (SO_2), um gás incolor, com odor característico, que reage com a água presente na atmosfera formando ácido sulfuroso (H_2SO_3). Outra forma comum dessa classe de óxidos é o trióxido de enxofre (SO_3), emitido diretamente na atmosfera ou gerado a partir do SO_2, gás que, em contato com a água, reage rapidamente formando ácido sulfúrico (H_2SO_4). Estes gases podem ser produzidos por fontes naturais (especialmente por vulcões) ou por fontes antropogênicas, sendo a queima de combustíveis (em especial os carvões minerais, ricos em enxofre) a principal fonte. A presença desses óxidos na atmosfera pode ser bastante nociva aos seres humanos e ao meio ambiente. Para a saúde humana, os principais efeitos incluem o comprometimento das funções pulmonares, as doenças respiratórias, irritações nos olhos, nariz e garganta e a mortalidade prematura. Com relação ao meio ambiente, as principais consequências são sobre a vegetação natural e sobre os cultivos agrícolas, que podem perder folhagem, diminuir produtividade ou até morrer, além da possibilidade de impactos causados pelo aumento da acidez da água e do solo. Esse aumento de acidez pode trazer prejuízos inclusive para formações rochosas e para a construção civil, aumentado as taxas de corrosão em metais (como ferro, aço e zinco) e de erosão em construções e monumentos (especialmente aqueles fabricados em mármore e rochas calcária e dolomítica) (WBC; WHO, 1999).

Outra questão importante em relação às emissões gasosas diz respeito ao material particulado que, quando não é devidamente tratado, pode sair pelas chaminés e prejudicar a qualidade do ar, especialmente em áreas urbanas nas proximidades das fábricas, podendo causar aumento das ocorrências de problemas respiratórios. No caso das caldeiras de recuperação química, o material particulado é produzido por reações entre gases do interior do forno e metais alcalinos vaporizados com a queima das gotas de licor injetadas na caldeira. Disso decorrem alguns aspectos importantes: o particulado em suspensão na corrente gasosa pode se depositar nas paredes dos trocadores de calor da região superior da fornalha, causando incrustações e entupimentos; a remoção do material é realizada por precipitadores eletrostáticos, sendo a eficiência de remoção dependente da carga de particulado e da potência do equipamento. O material recuperado retorna ao processo já que contém elementos precursores do licor branco, especialmente o sódio. Quando os precipitadores não alcançam alta eficiência, pode ocorrer aumento das emissões de particulado para a atmosfera, prejudicando a qualidade do ar no entorno das fábricas e redução da eficiência de recuperação química, já que há redução da quantidade de material que deveria retornar ao processo para ser recuperado como licor branco (VAKKILAINEN, 2005).

4.1 ESTUDO DE CASO 1: CALDEIRA NO BRASIL

4.1.1 Descrição da Base de Dados

A base de dados utilizada no primeiro estudo de caso é constituída por dados crus referentes a quatro meses de operação (junho, julho, agosto e setembro de 2001) de uma caldeira de recuperação química, obtidos de uma indústria de celulose em operação no Brasil.

Estavam disponíveis 2860 observações de quinze variáveis de processo (dessas, seis estão relacionadas as linhas de combustível e três à cada linha de ar), e quatro variáveis dependentes, relacionadas às emissões (H_2S e SO_2) e à eficiência do equipamento (emissões de H_2S, emissões de SO_2, fluxo de vapor, eficiência de redução), com taxa horária de amostragem. A Tabela 3 apresenta as variáveis e suas unidades de medida, em conjunto com estatísticas descritivas.

TABELA 3 – Resumo das variáveis disponíveis na base de dados crus - Estudo de Caso 1

	Variável	Média	Desvio-padrão	Mínimo	Máximo	Unidade
Entrada (Combustível)	Fluxo de entrada de licor	109.43	9.13	51.70	125.80	ton/h
	Temperatura do licor	126.45	0.98	115.40	131.70	℃
	Pressão do licor (paredes 2 e 4)	0.86	0.08	0.60	1.40	mmH2O
	Pressão do licor (paredes 1 e 3)	0.90	0.07	0.70	1.40	mmH2O
	Teor de sólidos do licor (medição 1)	68.16	1.66	62.40	73.60	%
	Dry solids content (measure 2)	68.24	1.62	62.40	74.10	%
Entrada (Ar)	Fluxo de ar primário	153.91	6.97	134.20	176.20	ton/h
	Temperatura do ar primário	150.03	1.95	136.50	158.60	℃
	Pressão do ar primário	37.37	7.04	16.40	105.40	mmH2O
	Fluxo de ar secundário	187.51	20.96	114.40	260.30	ton/h
	Temperatura do ar secundário	166.99	3.93	129.20	173.30	℃
	Pressão do ar secundário	202.77	18.54	118.00	265.90	mmH2O
	Fluxo de ar terciário	48.44	3.00	33.60	54.30	ton/h
	Temperatura do ar terciário	29.90	4.58	17.80	44.70	℃
	Pressão do ar terciário	200.23	17.11	100.80	263.50	mmH2O
Saída	Emissões de H_2S	0.25	0.36	0.00	2.50	ppm
	Emissões de SO_2	119.42	106.46	0.00	630.00	ppm
	Fluxo de vapor	297.85	26.53	203.10	359.90	ton/h
	Eficiência de redução	92.23	3.20	68.50	99.60	%

Fonte: Autoria própria

O foco deste primeiro estudo foi a predição do nível de emissões de SO_2. As razões para a escolha dessa variável foram duas, a primeira associada à perspectiva de melhoria do monitoramento e controle sobre a geração de efluentes gasosos, intencionando a redução dos impactos negativos causados ao meio ambiente, e a segunda associada à disponibilidade e características dos dados, afinal os dados de SO_2 se mostraram mais interessantes e adequados à aplicação de sensor virtual quando comparados aos dados de H_2S. A Figura 8 apresenta a série temporal para

a variável de saída escolhida. Em resumo, o conjunto inicial de variáveis consiste em dezesseis parâmetros de entrada e um de saída, conforme ilustrado pela Figura 9.

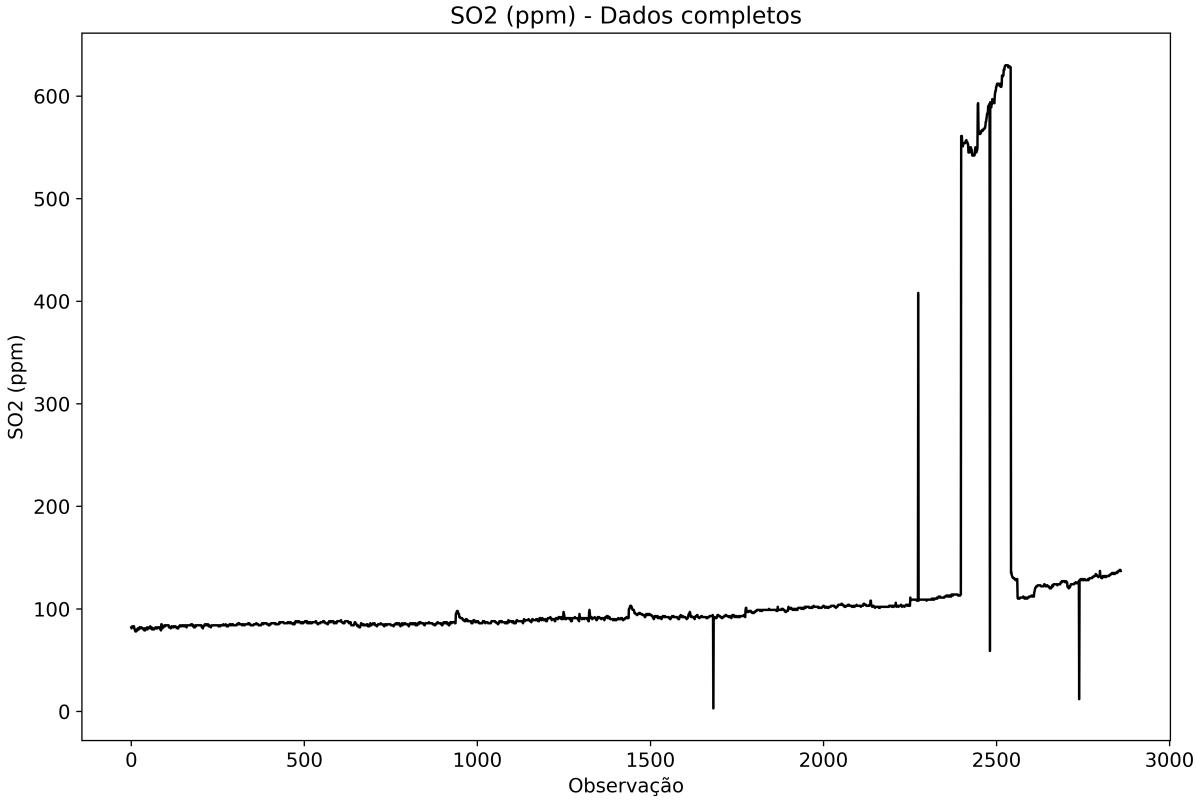


FIGURA 8 – Série temporal da variável teor de dióxido de enxofre (ppm) utilizada como variável de saída. Fonte: Adaptado de Valmet (2018)

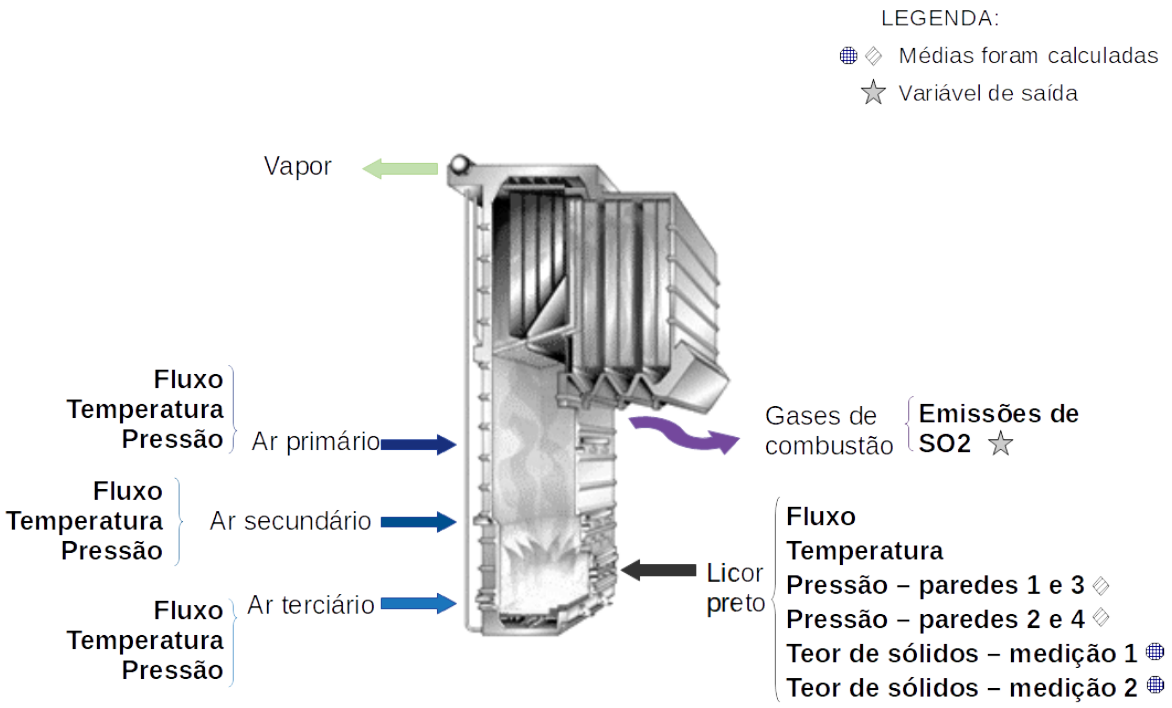


FIGURA 9 – Representação das variáveis utilizadas na construção do sensor. Fonte: Adaptado de Valmet (2018)

4.2 ESTUDO DE CASO 2: CALDEIRA NA FINLÂNDIA

4.2.1 Descrição da Base de Dados

O segundo estudo de caso envolve dados de uma caldeira de recuperação química de indústria de celulose localizada na Finlândia, compreendendo doze meses de operação, de janeiro a dezembro de 2003, sendo a amostragem condizente a médias horárias.

No total, estavam disponíveis 8760 observações de 34 variáveis, sendo 10 delas associadas à alimentação de combustível (licor e óleo), 10 associadas à alimentação dos ares primário, secundário e terciário, 8 variáveis dependentes, relacionadas às emissões e outras 6 relacionadas à geração de vapor, as quais não foram consideradas por não se relacionarem ao foco de estudo deste trabalho. A Tabela 4 apresenta as variáveis consideradas inicialmente neste estudo de caso e suas unidades de medida, em conjunto com estatísticas descritivas.

TABELA 4 – Resumo das variáveis disponíveis na base de dados crus - Estudo de Caso 2

	Variável	Média	Desvio-padrão	Mínimo	Máximo	Unidade
Entrada (Combustível)	Teor de sólidos do Licor	73.00	6.81	50.00	79.85	%
	Fluxo de entrada de licor (lado direito)	5.51	1.41	0.00	7.00	L/s
	Fluxo de entrada de licor (lado esquerdo)	5.45	1.49	0.02	7.00	L/s
	Fluxo de entrada de licor (medição total)	11.08	3.35	0.00	15.00	L/s
	Fluxo de entrada de óleo combustível	0.14	0.17	0.00	1.40	L/s
	Pressão do licor (frente)	1.15	0.42	0.00	6.00	bar
	Pressão do licor (atrás)	1.15	0.39	0.00	6.00	bar
	Pressão do licor (esquerda)	1.13	0.35	0.00	1.70	bar
	Pressão do licor (direita)	1.13	0.36	0.00	1.69	bar
	Temperatura de injeção do licor	117.87	26.67	3.59	138.96	℃
Entrada (Ar)	Fluxo de ar primário	10.21	2.05	0.00	13.60	L/s
	Pressão do ar primário	1091.60	305.38	0.00	1597.48	Pascal
	Temperatura do ar primário	166.39	38.70	13.09	191.11	℃
	Fluxo de ar secundário	21.20	6.19	0.00	25.00	Nm^3/s
	Pressão do ar secundário	2554.36	810.46	0.00	3987.17	Pascal
	Temperatura do ar secundário	140.44	31.65	15.99	205.32	℃
	Fluxo de ar terciário	3.24	1.88	0.00	7.18	Nm^3/s
	Pressão do ar terciário	2709.05	1192.10	0.00	4000.00	Pascal
	Fluxo de gases não condensáveis diluídos	3.54	1.19	0.00	5.04	Nm^3/s
	Temperatura dos gases não condensáveis diluídos	147.07	29.71	18.61	161.28	℃
Saída	Fluxo do gás de saída	2.93	1.30	0.00	7.49	Nm^3/s
	Teor de O2 no gás de saída (lado direito)	3.68	1.87	0.00	10.00	%
	Teor de O2 no gás de saída (lado esquerdo)	2.82	2.03	0.00	10.00	%
	Teor de SO2 no gás de saída	50.67	110.87	0.00	500.00	mg/Nm^3
	Teor de CO no gás de saída	172.17	208.83	0.00	1000.00	mg/Nm^3
	Teor de compostos de enxofre reduzido (TRS) no gás de saída	3.75	10.14	0.00	376.56	mg/Nm^3
	Teor de particulado no gás de saída	56.62	30.16	0.00	200.00	mg/Nm^3
	Temperatura do gás de saída	167.98	35.82	15.14	198.12	℃

Fonte: Autoria própria

Após verificar dentre as variáveis de saída aquelas que trariam o maior aprendizado em termos de aplicação de técnicas de regressão e de análise de sensibilidade, bem como uma possibilidade de contribuição mais significativa para a literatura, decidiu-se considerar o teor de material particulado no efluente gasoso. A Figura 10 apresenta a série temporal das emissões de material particulado, considerada variável de saída.

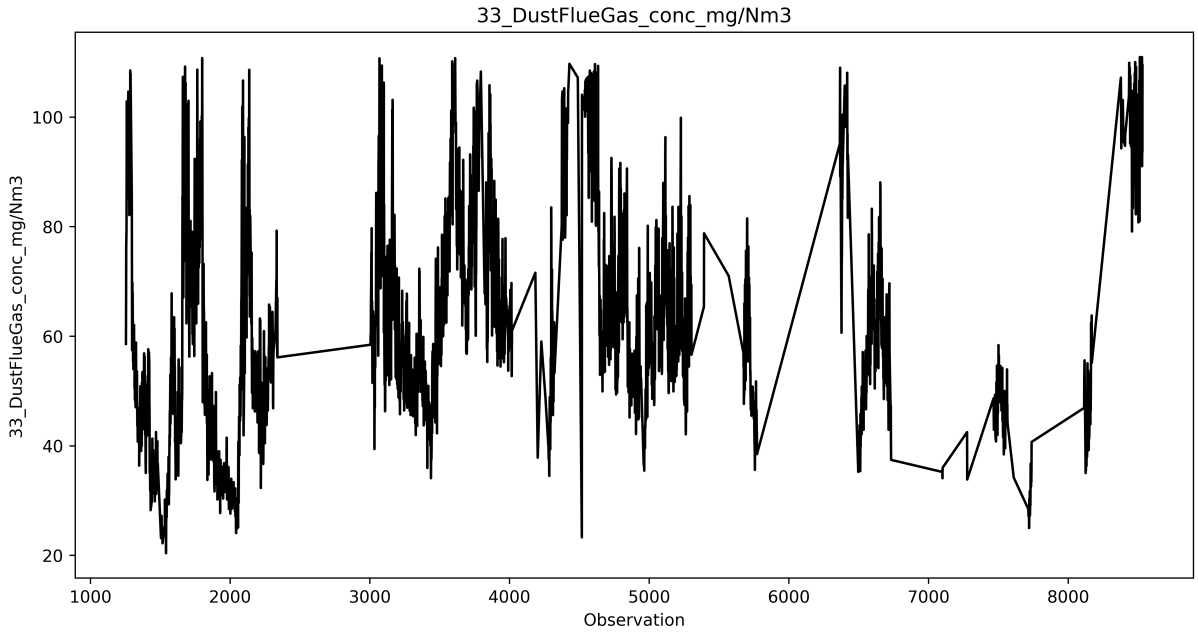


FIGURA 10 – Série temporal da variável teor de material particulado (mg/Nm^3) utilizada como variável de saída. Fonte: Autoria própria.

5 METODOLOGIA

Destaca-se que toda a metodologia foi construída utilizando-se a linguagem de programação Python (ROSSUM, 1995) e Jupyter Notebook como ambiente de desenvolvimento integrado (*Integrated Development Environment*, IDE) (KLUYVER et al., 2016). A escolha pelo Python se deve ao fato de ser uma linguagem de programação de "alto nível", gratuita, com bibliotecas em constante expansão e de código aberto. Além disso, a comunidade de programadoras/es Python é bastante ativa e colaborativa, permitindo um ambiente favorável ao desenvolvimento de novas pessoas programadoras, e as regras e estruturas de escrita de códigos é simples e contribui para o desenvolvimento de códigos adaptáveis e de leitura e compreensão facilitados.

A Figura 11 apresenta, de forma esquemática, a metodologia utilizada que será descrita nos tópicos a seguir.

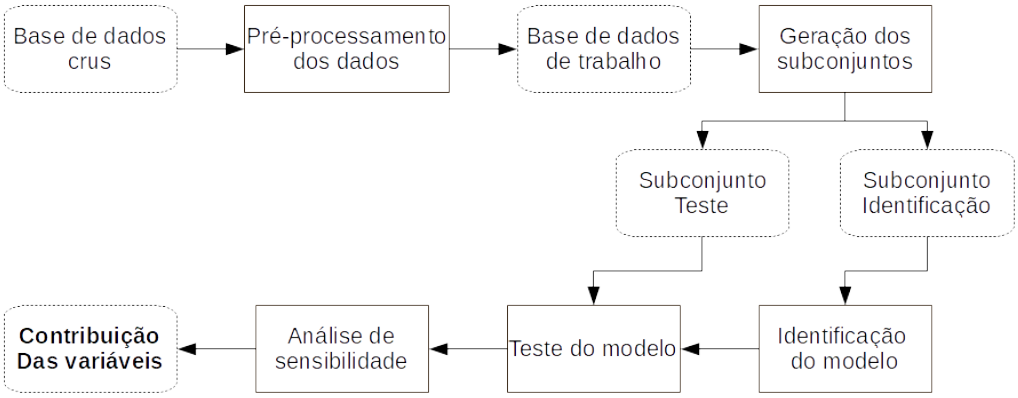


FIGURA 11 – Diagrama da metodologia empregada no trabalho

5.1 PRÉ-PROCESSAMENTO DE DADOS

Segundo Kotsiantis, Kanellopoulos e Pintelas (2006), apenas a aplicação do modelo baseado em dados para a predição de variáveis não é suficiente, já que a qualidade do modelo depende intrinsecamente da qualidade dos dados. Desse modo, cabe ressaltar a importância da etapa de pré-processamento de dados, bem como sua variedade de abordagens, sendo a escolha dos métodos de pré-processamento

dependente do conjunto de dados e do processo em questão. Resumidamente e de modo mais recorrente, a primeira etapa inclui a seleção do período de coleta de dados representativos que serão utilizados para a construção do modelo; e a remoção de dados discrepantes (*outliers*), que pode ser conduzida por filtros uni ou multivariados, ou por abordagens de envelope (*wrapper approach*). Também é necessário observar a presença de dados faltantes, e, se houver, é importante compreender as razões para a ausência desses dados e decidir como será o tratamento com relação às lacunas, sendo possível ignorar as observações incompletas ou utilizar alguma ferramenta para inferir valores às variáveis que não foram medidas pontualmente. Algumas possibilidades seriam a substituição das lacunas pelo valor mais comum, pela média, pela mediana, por regressão, por classificação, entre outras. Posteriormente, é feita a normalização dos dados. As técnicas mais usuais são a normalização mínimo-máximo (*min-max normalization*) e a padronização em z (*z-score normalization*). A partir disso, é realizada a seleção de variáveis, ou características, conforme sua classificação como relevante, irrelevante ou redundante, em relação à variável de interesse. Por fim, é possível ainda a criação de novas variáveis, a partir das variáveis originais. A resposta desse novo conjunto de variáveis pode ser mais precisa, entretanto, é possível que a variável transformada não apresente qualquer significado físico.

Neste trabalho, para ambos estudos de caso, foi observado que todas as variáveis apresentaram a mesma taxa de coleta de dados, todas as observações seguiram o mesmo padrão, com intervalos de amostragem de 1 hora, não sendo necessário conduzir uma etapa de sincronização das medições. Assim, a primeira etapa de pré-processamento aplicada foi a seleção de variáveis. Para constituir a entrada do modelo, foram escolhidas variáveis manipuláveis da operação e que alimentam a caldeira definindo seu ponto de operação. Para a saída, para cada estudo de caso, foi escolhida uma variável relacionada ao potencial de geração de poluição atmosférica pela caldeira. Com relação aos dados faltantes, foi escolhida a abordagem mais simples e eficiente: remoção das linhas de observações que, eventualmente, continham alguma lacuna ou algum valor nulo. Essa escolha é justificada por duas razões: a primeira se relaciona ao fato de que não se está considerando qualquer dinâmica do processo, ou seja, não estão sendo avaliadas as interações entre observações consecutivas, e por isso, a existência de intervalos de coleta maiores que 1 hora não comprometeria os resultados; e a segunda, deve-se à quantidade de dados disponíveis, ou seja, o descarte de linhas contendo observações faltantes não reduz de forma significativa o total de dados.

Em seguida, o objetivo foi identificar e remover valores discrepantes. Pode-se definir ’valor discrepante’ em um conjunto de dados como sendo uma observação (ou subconjunto de observações) que aparentemente é inconsistente com o restante dos dados (BARNETT; LEWIS, 1974). Inicialmente foi realizada uma análise visual

para a identificação de inconsistências. Na sequência, empregou-se o filtro de Hampel [1] para a remoção de dados discrepantes, devido à sua robustez e simplicidade de implementação (ROUSSEEUW; CROUX, 1993). Essa técnica foi aplicada para cada variável separadamente, sendo inicialmente calculada a mediana (Equação 5.1), em seguida o desvio absoluto da mediana (*median absolute deviation*, MAD) (Equação 5.2) e, por fim, os limites superior e inferior (Equações 5.3 e 5.4) utilizando o parâmetro multiplicador b para que exista equivalência comparativa ao desvio-padrão (Equação 5.5). Com isso, retiraram-se as observações cujos valores estivesse abaixo do limite inferior ou acima do limite superior, calculados para a variável em questão (Hampel (1974), Hampel (1985), Mathworks (2017)).

$$med_j = med(X_j) \tag{5.1}$$

$$MAD_j = med(|x_{ij} - med_j|) \tag{5.2}$$

$$b = \frac{1}{\sqrt{2} * erfc^{-1} * (1/2)} \approx 1.4826 \tag{5.3}$$

$$lowerlimit_j = med_j - b * MAD_j \tag{5.4}$$

$$upperlimit_j = med_j + b * MAD_j \tag{5.5}$$

A variável med_j corresponde à mediana da variável X_j, MAD_j corresponde ao desvio da mediana para a variável X_j, b indica o número de desvios com relação à mediana a ser considerado, e $lowerlimit_j$ e $upperlimit_j$ são os limiares inferior e superior, respectivamente, considerados no filtro.

Após a etapa de pré-processamento dos dados, tem-se, ao final, um banco de dados de trabalho, gerado a partir do banco de dados crus, coletado na fábrica.

5.2 GERAÇÃO DOS CONJUNTOS DE DADOS: IDENTIFICAÇÃO E TESTE

Após o pré-processamento, o conjunto resultante foi dividido em dois: conjunto de identificação e conjunto de teste. Para conduzir esta partição, é necessário garantir que, para qualquer técnica de construção de sensor virtual utilizada, sempre um mesmo conjunto de observações deve ser aplicado para identificação, e sempre o mesmo conjunto (complementar ao de identificação) deve ser aplicado para teste.

[1] Procedimento estatístico que utiliza parâmetros como a mediana e o desvio-padrão da mediana para identificar valores discrepantes (*outliers*) dado um conjunto de observações (HAMPEL, 1985)

Neste sentido, inicialmente, foram separados os valores máximo e mínimo de cada variável, de modo a garantir que estes estejam sempre no conjunto de identificação. Isso é importante para que a normalização, cujos valores devem ser provenientes do conjunto de identificação, seja conduzida apropriadamente. Sobre o conjunto restante, foi empregada a função de divisão aleatória de uma base de dados em dois subconjuntos. Nesta função (*sklearn.model_selection.train_test_split*; (PEDREGOSA et al., 2011)) é possível definir uma semente aleatória [2] e, com isso, garantir que os índices de cada observação sempre devem ser alocados no mesmo conjunto. Posteriormente, o conjunto contendo os valores máximo e mínimo de cada variável é inserido no conjunto de identificação, de tal forma que, ao final desse procedimento, cerca de 75% das observações são alocadas no conjunto de identificação e os 25% restantes, no de teste. Tendo em vista que cada variável apresenta faixas de valores bastante distintas entre si, é importante executar a normalização dos dados. Foi utilizada a escala mínimo-máximo, entre -1 e 1, conforme a Equação 5.6 (PEDREGOSA et al., 2011).

$$xnorm_{ij} = 2\frac{x_{ij} - x_{j_{min}}}{x_{j_{max}} - x_{j_{min}}} - 1 \tag{5.6}$$

Sendo x_{ij} o valor da observação i da variável j, e $x_{j_{min}}$ e $x_{j_{max}}$ os valores mínimo e máximo assumidos pela variável j, respectivamente.

Os valores para a construção da escala foram calculados a partir do conjunto de identificação. Com os valores $x_{j_{min}}$ e $x_{j_{max}}$ determinados, a normalização foi aplicada sobre os conjuntos de identificação e de teste. Após a obtenção dos resultados da regressão, a transformação inversa foi conduzida para apresentação dos resultados.

5.3 CONSTRUÇÃO DO SENSOR VIRTUAL

5.3.1 Regressão Linear Múltipla

Com a intenção de se obterem resultados comparativos, tendo como fundamento sensores virtuais para aplicação em regressão, um modelo de regressão linear múltipla (*Multiple Linear Regression*, MLR) foi construído a partir do banco de dados de trabalho resultante da etapa de pré-processamento. A regressão linear múltipla permite que uma variável resposta, y, seja modelada por uma função linear composta por duas ou mais variáveis regressoras. A Equação 5.7 apresenta uma forma matricial para modelo de regressão linear múltipla, em que X é a matriz de variáveis de entrada, β são os coeficientes da regressão, ϵ é o erro, diferença entre o valor real da saída e o

[2] Número utilizado como padrão para a geração de números aleatórios, permitindo que uma mesma sequência seja gerada repetidas vezes

valor obtido pelo modelo de regressão, n é o número de observações e p é o número de variáveis.

$$y = X\beta + \epsilon \quad , \text{com}$$

$$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad ; \quad X = \begin{bmatrix} 1 & x_{11} & x_{12} & \dots & x_{1p} \\ 1 & x_{21} & x_{22} & \dots & x_{2p} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & x_{n2} & \dots & x_{np} \end{bmatrix} \quad ; \quad \beta = \begin{bmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_n \end{bmatrix} \quad ; \quad \epsilon = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix} \tag{5.7}$$

Os parâmetros β são calculados a partir da minimização da função erro (L), pelo método dos mínimos quadrados, conforme a Equação 5.8, resultando na estimativa para β dada pela Equação 5.9. (MONTGOMERY; RUNGER, 2014).

$$L = \epsilon'\epsilon = (y - X\beta)'(y - X\beta) \quad \rightarrow \quad \text{minimizar} \tag{5.8}$$

$$\frac{\partial L}{\partial \beta} = 0 \quad \rightarrow \quad \hat{\beta} = (X'X)^{-1}X'y \tag{5.9}$$

No presente trabalho, toda a metodologia foi desenvolvida em Python, sendo utilizada a função *'sklearn.linear_model.LinearRegression'* para a obtenção e aplicação do modelo de regressão linear múltipla (PEDREGOSA et al., 2011).

5.3.2 Rede Perceptron Multicamadas

Com os resultados da etapa de pré-processamento, foram treinadas redes MLP, utilizando a função *'sklearn.neural_network.MLPRegressor'*, com a variação de alguns de seus hiperparâmetros internos. Os hiperparâmetros [3] modificados foram:

- Função de Ativação: logística e tangente hiperbólica.
- Método de resolução da otimização dos pesos (w_{ij}): LBFGS (método de otimização da família dos métodos quasi-Newton) e SGD (gradiente descendente estocástico).
- Taxa de aprendizagem: constante, *invscaling* (diminui a taxa de aprendizado a cada passo usando escala exponencial inversa) e adaptativa. As variações nas taxas de aprendizagem são aplicadas apenas para a função de ativação SGD. Para LBFGS, conforme estabelecido internamente ao código usado, apenas a função de ativação constante é adequada (PEDREGOSA et al., 2011).

[3] Hiperparâmetros são ps parâmetros do algorítimo de aprendizagem (não do modelo). Eles devem ser estabelecidos à priori e não são modificados durante o treinamento. O ajuste de hiperparâmetros é uma parte importante para a construção de sistemas de aprendizado de máquinas (GÉRON, 2019).

- Número de neurônios na camada oculta: variação de 1 a 30. Utilizou-se a possibilidade mais simples de rede neural, com uma única camada oculta.

Para a seleção da topologia mais adequada para a rede MLP, foi aplicada a técnica validação cruzada com k-partições (*k-fold cross-validation*) (KRIEGESKORTE, 2015). Para a validação cruzada, apenas o conjunto de identificação é utilizado. Cabe reforçar que, para garantir reprodutibilidade dos testes, todas as etapas de partição dos dados foram realizadas a partir de uma mesma semente aleatória. As observações separadas para o conjunto de identificação foram divididas em k partições. No caso, utilizou-se k igual a 5, sendo que $(k-1)$ delas foram usadas efetivamente para treinamento, e o grupo restante foi utilizado para validação. Este processo é repetido k vezes. Para cada uma dessas repetições de treinamento e validação, é calculado o coeficiente de correlação entre a saída real e aquela estimada pela rede. Um esquema representativo desse procedimento é apresentado na Figura 12.

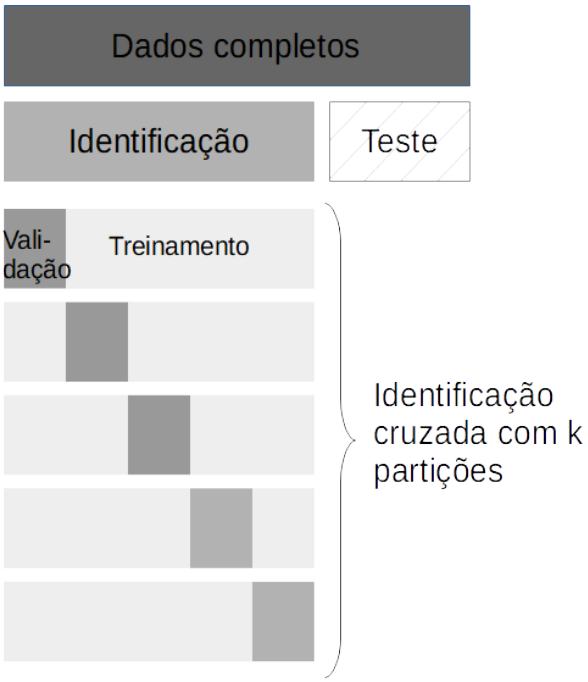


FIGURA 12 – Representação esquemática do procedimento de validação cruzada. Fonte: autoria própria

Desse modo, com base nos resultados obtidos a partir da validação cruzada, é selecionada a topologia que melhor se ajusta aos dados. Por fim, a rede é re-treinada com todos os dados de identificação. Posteriormente, o conjunto de teste é usado para verificar a capacidade de predição e generalização do modelo selecionado.

5.4 ANÁLISE DE SENSIBILIDADE

Decorridas as etapas de treinamento, validação e teste do modelo neural de predição da variável de saída é conduzida a análise de sensibilidade. Cada uma das entradas, uma por vez, é sujeita a uma perturbação dada pela substituição do seu valor original pela sua média adicionada de uma fração crescente do seu desvio-padrão, conforme a Equação 5.10.

$$VariavelAlterada_i(n) = media_i + k * stdev_i \tag{5.10}$$

Em que $k = 0.5$, para $1 <= n <= 250$; $k = 1$, para $250 < n <= 500$; $k = 1.5$, para $500 < n <= 750$; $k = 2$, para $750 < n <= N$, com N o total de observações do conjunto de treinamento.

Dado o modelo neural de predição, são calculados o coeficiente de determinação (r^2_{ref}) e o erro quadrático médio (MSE_{ref}) de referência, ou seja, para o conjunto de identificação sem qualquer perturbação. Considerando o conjunto de identificação, uma variável por vez é substituída pela sua média e o coeficiente de determinação (r^2_i) e o erro quadrático médio (MSE_i) são calculados. Segue-se então, a comparação entre o coeficiente de correlação e o erro quadrático médio associados à cada perturbação, com os respectivos valores de referência. A contribuição de influência de cada variável é calculada pela Equação 5.11. Quanto maior o valor de MSE, maior a contribuição da variável sobre a variável resposta.

$$\%contribuicao_i = \frac{(MSE_i - MSE_{ref})}{\sum_{i=1}^{p} MSE_i} * 100\% \tag{5.11}$$

As contribuições de cada uma das variáveis de processo sobre a variável de saída é representada graficamente por um diagrama em que a espessura da seta é proporcional ao percentual de contribuição correspondente à variável, dado pela diferença entre MSE_i e MSE_{ref}.

6 RESULTADOS E DISCUSSÕES

6.1 ESTUDO DE CASO 1: CALDEIRA NO BRASIL

6.1.1 Pré-Processamento

A matriz de dados inicial contém 15 variáveis de entrada, uma variável de saída e 2860 linhas de observações. Gráficos de dispersão para cada uma das variáveis foram construídos, e observou-se, para a variável teor de SO_2, variável de saída, uma região muito distante em relação às demais observações. Este conjunto foi retirado, restando 2706 amostras antes da aplicação do filtro de Hampel. A Figura 6.1.1 apresenta a série temporal para a variável de saída com destaque para os dados discrepantes. Conforme pode ser observado, a remoção da região de dados com valores muito distantes não altera a tendência geral do teor de SO_2 ao longo do período considerado.

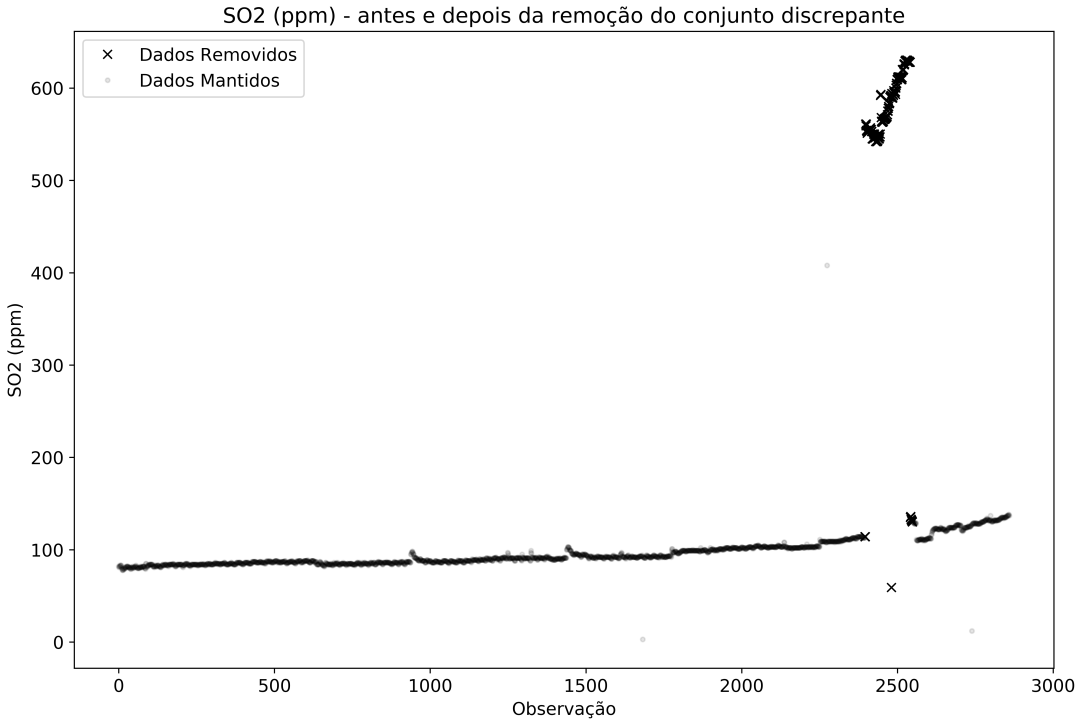


FIGURA 13 – Gráfico temporal para a variável teor de SO_2 nos gases de emissão, com destaque para a região de dados discrepantes. Fonte: autoria própria.

Considerou-se que dois pares de variáveis apresentavam informações redundantes, ou seja, a informação contida em uma variável de cada um dos pares corresponde à informação contida na outra variável. Por isso, cada par foi substituído

pelas respectivas médias, conforme a Equação 6.1.

$$X = \frac{X_i + X_j}{2} \tag{6.1}$$

Após o cálculo das médias das variáveis pressão do licor nas paredes 1 e 3 e pressão do licor nas paredes 2 e 4, foram dispostos em gráficos de dispersão os resultados comparativos entre as variáveis originais e a variável substituta. O mesmo procedimento foi realizado para as variáveis teor de sólidos medição 1 e teor de sólidos medição 2. A Figura 14 mostra os resultados em termos das correlações entre as variáveis. A correlação existente entre os pares de variáveis citados se explicita na Figura 14, justificando a substituição de cada par pela respectiva média sem prejuízo em termos de informação armazenada.

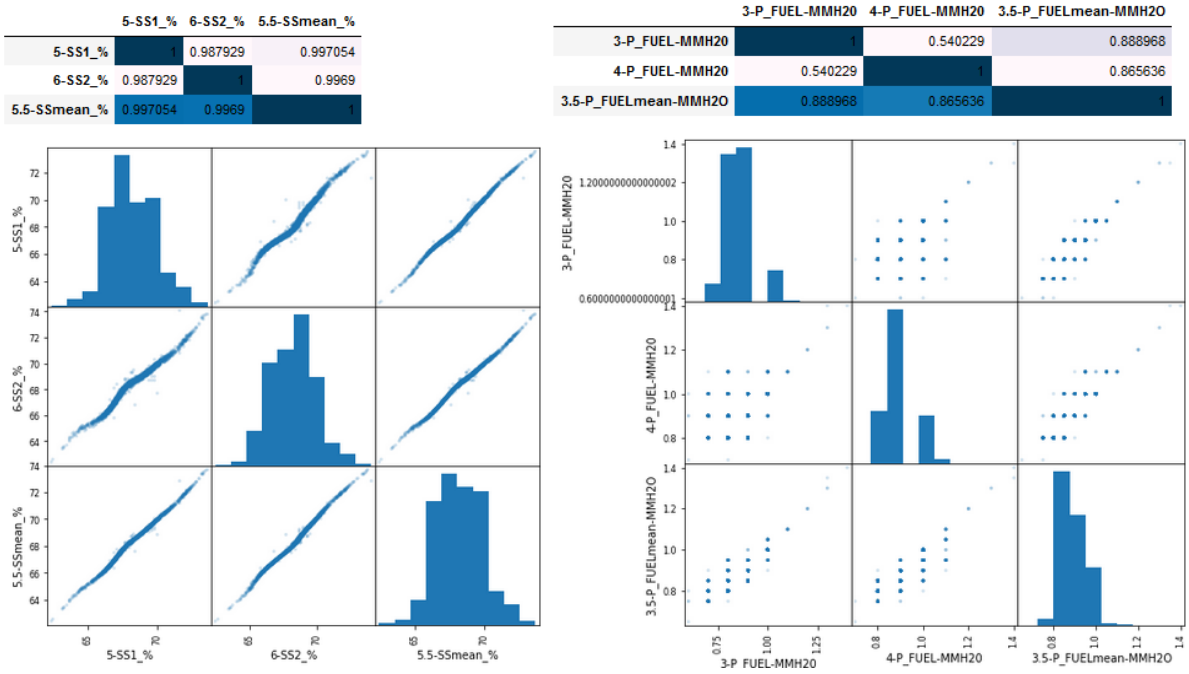


FIGURA 14 – Apresentação, por meio de matriz de gráficos de dispersão, das correlações entre as variáveis originais e a média; (a) pressão do combustível; (b) teor de sólidos. Fonte: autoria própria.

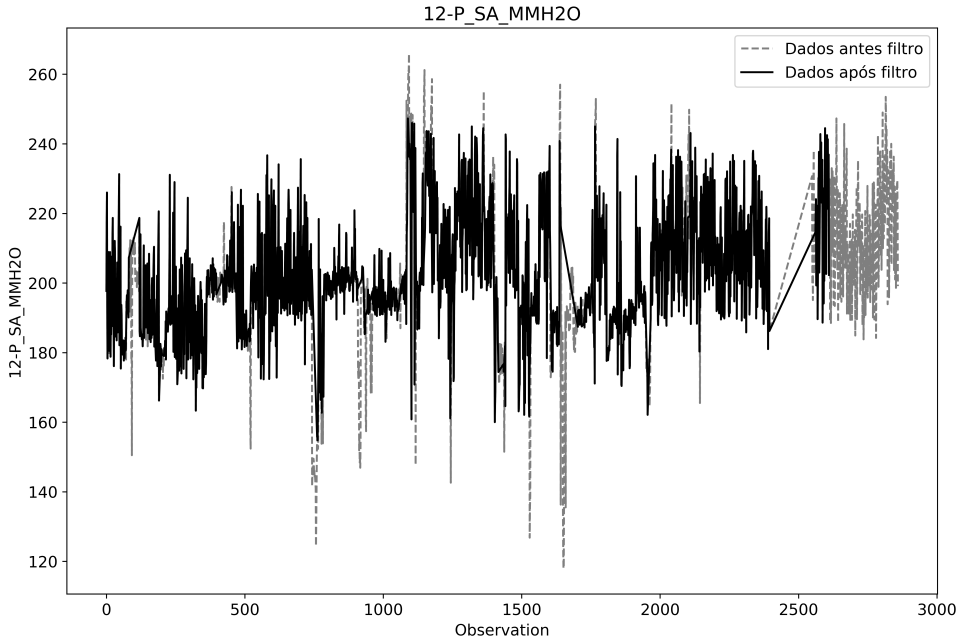
Em seguida, aplicou-se o filtro de Hampel, para cada variável individualmente. Os valores discrepantes foram removidos, restando, ao final, 2108 observações. A Tabela 5 apresenta a quantidade de observações discrepantes por variável.

TABELA 5 – Número de observações discrepantes por variável, segundo o filtro de Hampel

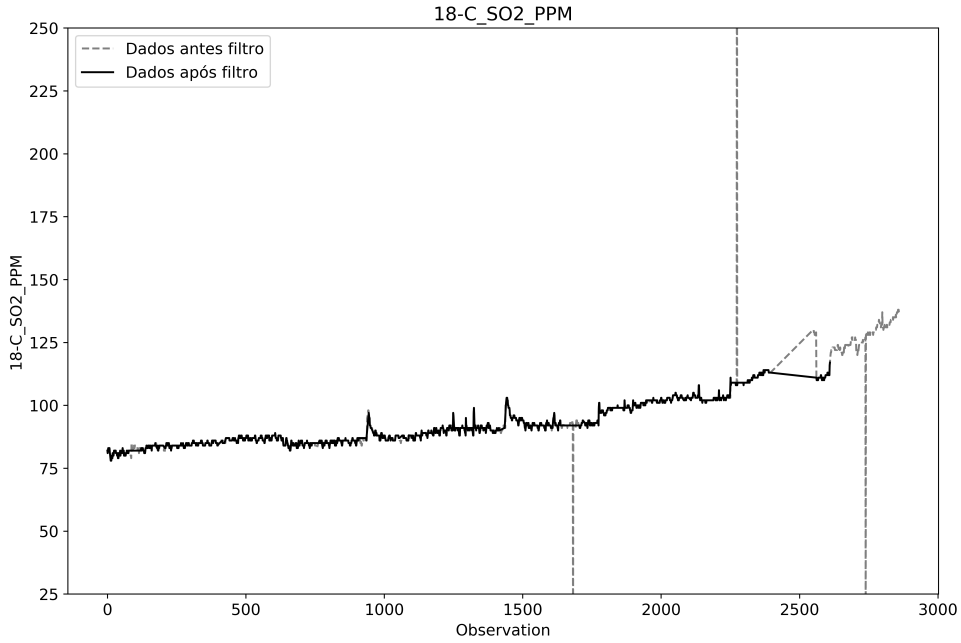
Variável	Número de Observações Discrepantes
1-F_FUEL_TH	55
2-T_FUELoC	53
3.5-P_FUELmean-MMH2O	9
5.5-SSmean_%	6
7-F_PA_TH	58
8-T_PA_oC	12
9-P_PA_MMH2O	84
10-F_SA_TH	0
11-T_SA_oC	34
12-P_SA_MMH2O	69
13-F_TA_TH	154
14-T_TA_oC	0
15-P_TA_MMH2O	15
18-C_SO2_PPM	263
Total de observações discrepantes	812
Total de linhas removidas	598
Tamanho final da base de dados	(2108, 14)

Fonte: autoria própria.

De forma a exemplificar este procedimento, a Figura 15 mostra as diferenças nos conjuntos de dados das variáveis pressão do ar secundário e teor de SO_2, antes e depois da aplicação do filtro.



(a) Pressão do ar secundário (mmH_2O).



(b) Teor de SO_2.

FIGURA 15 – Diferenças antes e após a aplicação do filtro de Hampel. Fonte: autoria própria.

A aplicação do filtro de Hampel consegue eliminar as observações mais discrepantes ao longo das primeiras 2600 observações, e a partir dessa todas as observações são consideradas discrepantes e eliminadas. Retomando a Figura , percebe-se que, aproximadamente, a partir da região em que os teores de SO_2 estão muito acima da tendência geral, a maioria da observações seguintes seriam consideradas discrepantes pelo filtro de Hampel.

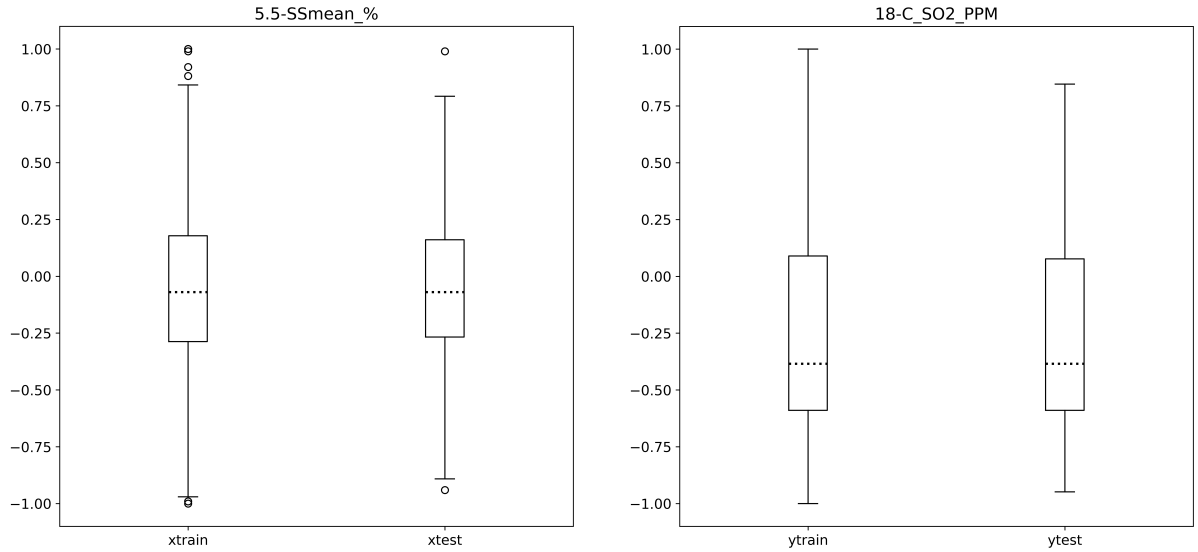
Finalizadas estas etapas de pré-processamento, o banco de dados de trabalho

gerado foi submetido à divisão entre conjunto de identificação e conjunto de teste, em seguida à normalização e, posteriormente, à construção dos modelos de sensores virtuais com aplicação em regressão.

6.1.2 Geração dos conjuntos de dados: Identificação e Teste

Inicialmente, foram filtradas as observações com o valor máximo e mínimo de cada variável, a fim de garantir que estariam alocadas no conjunto de identificação. Foi estabelecido um número que seria utilizado como semente de geração de números aleatórios necessários à separação dos dados de identificação e teste, com a intenção de certificar que esses seriam sempre os mesmos. Empregou-se a função *train_test_split* do módulo *model_selection* do pacote *scikit-learn* do Python, com semente igual a 42, e com a proporção 75% para identificação e 25% para teste. Do conjunto inicial de 2108 observações, destinaram-se 1580 observações para identificação e 528 para teste. Os dados foram normalizados utilizando escala mínimo máximo com amplitude entre -1 e 1. Cabe ressaltar que quando a rede neural foi testada com função de ativação logística, a normalização da variável resposta foi realizada com amplitude entre 0 e 1, coerente com o seu conjunto imagem.

Após a etapa de normalização, gráficos do tipo boxplot foram construídos para cada variável a fim de se comparar as faixas de valores em cada conjunto. A Figura 16 apresenta, comparativamente, os dados de identificação e de teste, para duas variáveis de entrada.



(a) Teor de sólidos no licor. (b) Teor de SO_2.

FIGURA 16 – Boxplot dos conjuntos de identificação e teste para as variáveis. Fonte: autoria própria.

A separação do banco de dados entre conjunto de identificação e conjunto

de teste não descaracteriza, em termos estatísticos, um dos conjuntos em relação ao outro. Como se pode verificar nos gráficos da Figura 16, a estatística descritiva do conjunto de teste se assemelha bastante com a do conjunto de identificação.

Com os conjuntos separados e já normalizados, seguem-se as etapas de construção de modelos de sensores virtuais, o primeiro com base em regressão linear múltipla, e o segundo baseado em rede perceptron multicamadas.

6.1.3 Modelos de Sensores Virtuais

6.1.3.1 *Regressão Linear Múltipla*

O modelo de regressão linear múltipla foi construído utilizando a função *LinearRegression* do módulo *linear_model* do pacote *scikit-learn*. Os dados de entrada e saída normalizados foram usados para a construção do modelo de regressão. Após a obtenção do modelo, a matriz de entrada do conjunto de teste foi aplicada para verificar a capacidade de predição do modelo. Os valores preditos foram, então, comparados com os valores de saída do conjunto de teste.

Os coeficientes da regressão calculados foram:

$\beta 0 = -0.418878\ (intercepto)$ e

$\beta = [0.48759;\ 0.10392;\ -0.27802;\ -0.002405;\ 0.17434;\ -0.11417;\ 0.023235;\ -0.12405;\ -0.50648;\ -0.02927;\ -0.40424;\ 0.30765;\ -0.19730]$

A saída do modelo está normalizada. Portanto, é necessário aplicar a transformação inversa e retornar os valores à escala original. Esse procedimento foi realizado antes da apresentação dos resultados. Pela Figura 17 é possível comparar as estimativas do modelo com os valores reais assumidos pela variável de saída, o teor de SO_2.

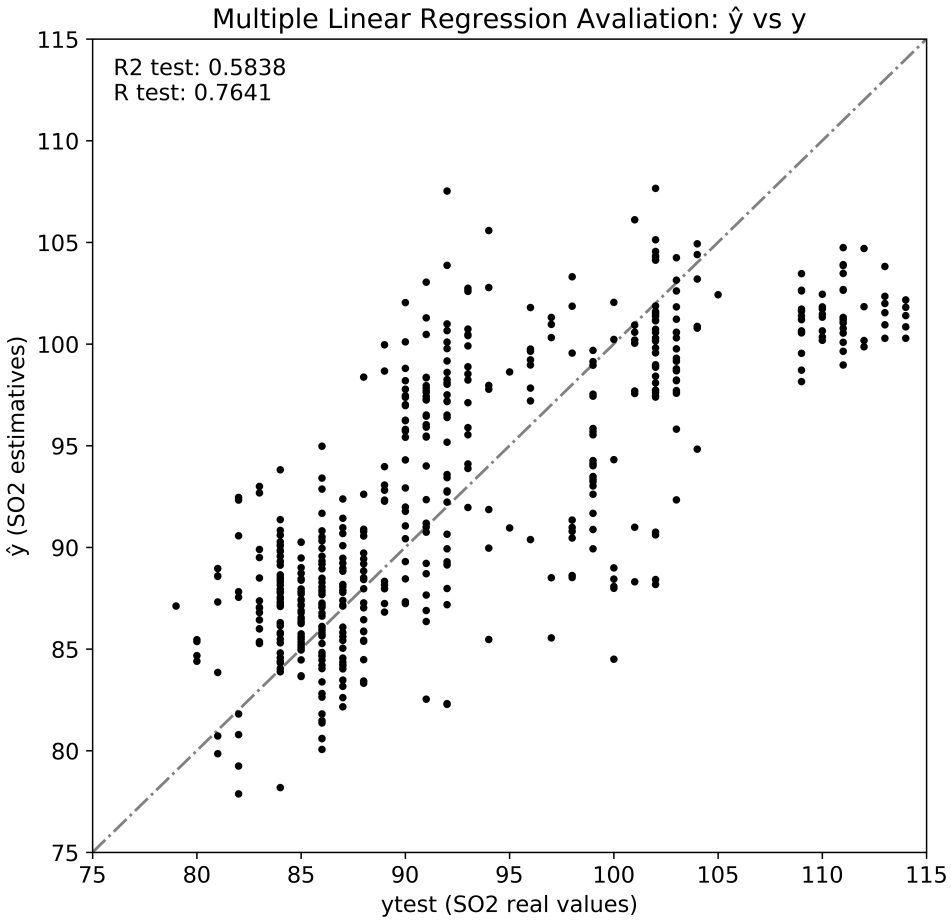
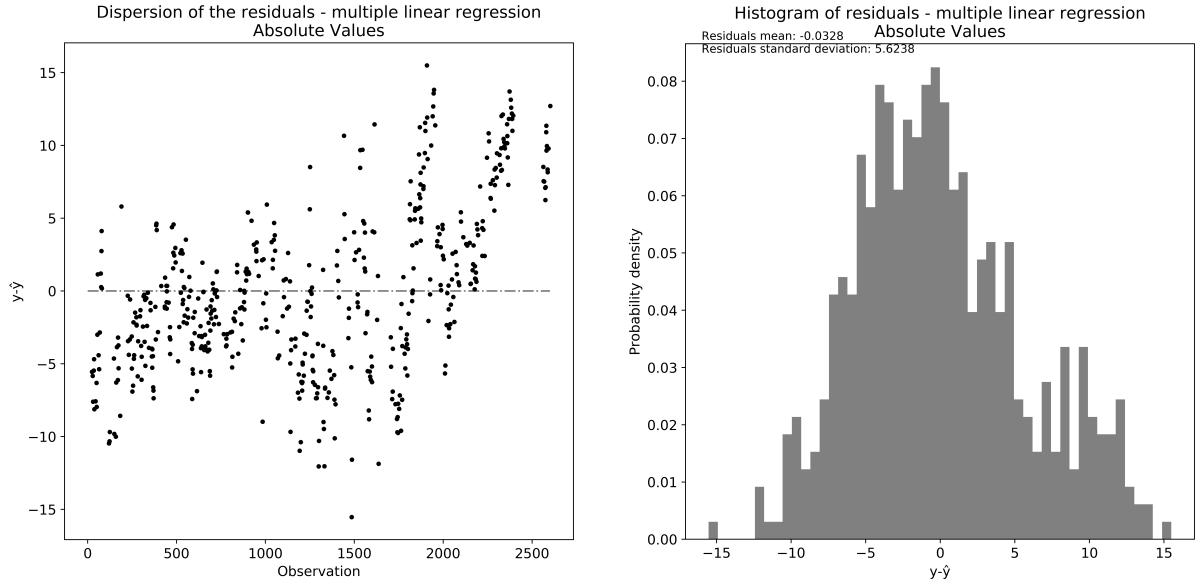


FIGURA 17 – Avaliação do resultado da Regressão Linear Múltipla, ŷ versus y.Fonte: autoria própria.

O coeficiente de determinação, R^2 foi 0.5838. Isso indica que aproximadamente 58% da variância da variável de saída, no caso, teor de SO_2 nos gases de combustão, é predito pelas variáveis independentes, com base no modelo de regressão linear múltipla. A Figura 18 exibe o comportamento dos resíduos (valor real subtraído do valor estimado, ou seja, y-ŷ), por meio do gráfico de dispersão (18a) e de histograma (18b) dos resíduos.



(a) Gráfico de dispersão dos resíduos (b) Histograma dos resíduos.

FIGURA 18 – Visualização do perfil dos resíduos (y-ŷ) obtidos a partir da regressão linear múltipla. Fonte: autoria própria.

Observando o histograma, a média dos resíduos se aproxima do zero, entretanto sua curva de distribuição se distancia da curva normal. Essa tendência de não homogeneidade na distribuição dos resíduos também pode ser observada a partir da análise do gráfico de dispersão, indicando que a relação entre as variáveis de entrada e a variável resposta tende a ser não linear.

6.1.3.2 *Rede Perceptron Multicamadas*

Para a rede neural, foram testadas variações em alguns hiperparâmetros a fim de escolher a topologia que melhor se adéque ao problema. A técnica de validação cruzada foi aplicada com a intenção de avaliar qual seria a topologia escolhida.

A validação cruzada foi conduzida com base na função *KFold*, do módulo *model_selection*, do pacote *scikit-learn*, do Python. O conjunto de identificação (com o total de 1580 observações) foi dividido em 5 partições, cada uma contendo 316 observações, que foram usadas para treinar e validar todas as possíveis combinações de hiperparâmetros para a construção da rede. A divisão em partições, assim como a separação anterior em identificação e teste, foi conduzida de forma pseudo aleatória, usando uma semente para a geração dos números aleatórios que orientam a distribuição das observações para as 5 partições. Isso permite a comparação isonômica dos resultados obtidos por cada combinação de hiperparâmetros.

Foi observado que, para ambas as funções de ativação, logística e tangente hiperbólica, o método de otimização e a taxa de aprendizado que oferecem o maior coeficiente de correlação, independentemente do número de neurônios na camada

oculta, são, respectivamente, o método de otimização LBFGS e a taxa de aprendizagem constante. Cabe ressaltar que os valores do coeficiente de correlação (r_{medio},) são uma média daqueles obtidos pela validação cruzada, e portanto, para cada valor de r existe um desvio-padrão associado. A Figura 19 ilustra os resultados da validação cruzada, em termos das médias e dos desvios-padrões dos coeficientes de correlação, para cada um dos tipos de função de ativação testados, segundo a variação do número de neurônios na camada oculta.

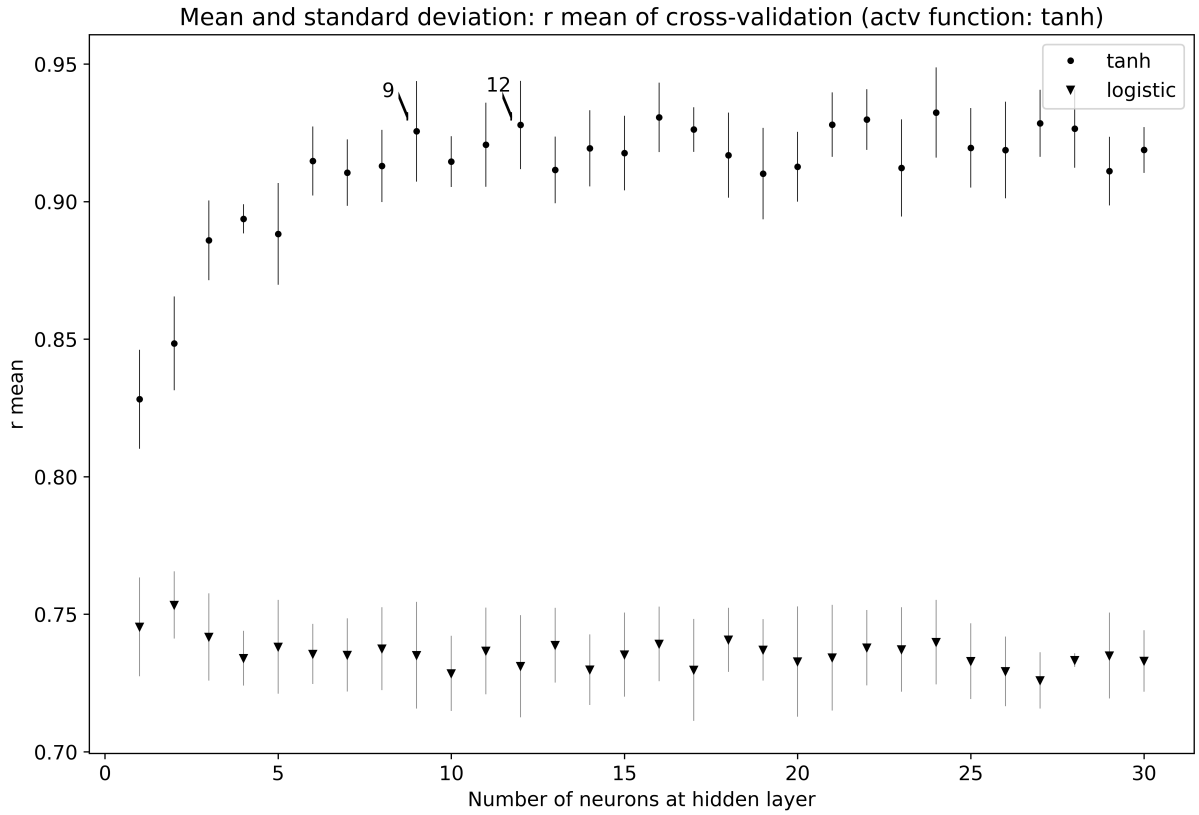


FIGURA 19 – Evolução do r_{medio} da validação cruzada conforme o número de neurônios na camada oculta, para as funções de ativação logística e tangente hiperbólica, destaque para as redes com melhores resultados. Fonte: autoria própria.

Tendo como base a Figura 19, é possível concluir que a rede que emprega a função de ativação tangente hiperbólica alcança melhores resultados em comparação àquela que utiliza a função logística, tendo em vista o coeficiente de correlação médio, obtido da validação cruzada. Essa função foi selecionada para ser utilizada na rede que deve funcionar como sensor virtual para predição de SO_2. O passo seguinte é a determinação do número de neurônios na camada oculta.

De acordo com a Figura 19, as redes com 9 e 12 neurônios na camada oculta apresentaram os melhores resultados. Entretanto, apesar de o coeficiente de correlação médio (r_{medio}) da rede com 12 neurônios ($r_{medio} = 0.92708$) ser relativamente pouco

maior que o da rede com 9 ($r_{medio} = 0.92508$), pelo princípio da parcimônia [1], observa-se que a diferença é muito pequena (0.22%), assim não justificando o aumento da complexidade da rede. Portanto, a rede com 9 neurônios na camada oculta foi escolhida como a mais adequada para a regressão do teor de SO_2 em função das 13 variáveis de processo em questão. Pode-se observar que o valor de r $(0, 92508)$ é significativamente maior do que aquele obtido para a regressão linear múltipla, igual a 0.7641.

Uma rede neural MLP, com os hiperparâmetros definidos: função de ativação tangente hiperbólica, taxa de aprendizado constante e 9 neurônios na camada oculta, foi treinada com o conjunto de identificação completo e testada com o conjunto de teste. O coeficiente de correlação foi da ordem de 0.94, e a Figura 20 apresenta os resultados da regressão com o modelo neural, enquanto a Figura 21 apresenta a qualidade da regressão ilustrada pelo comportamento dos resíduos.

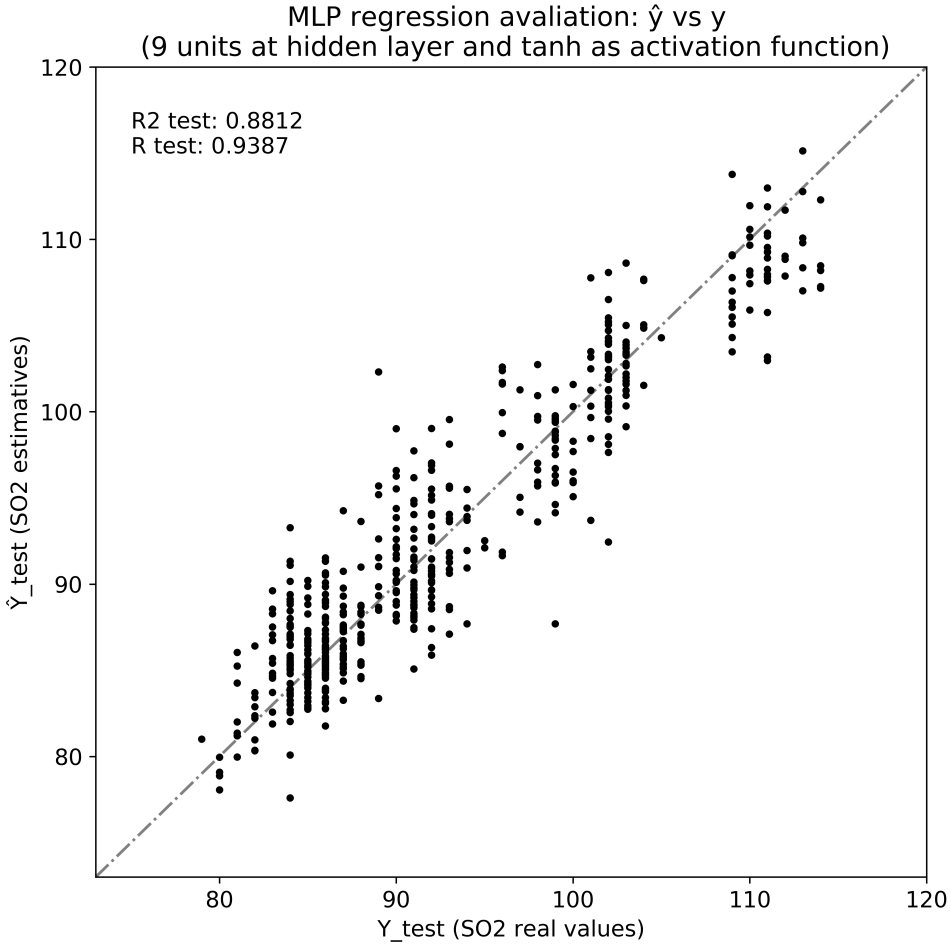
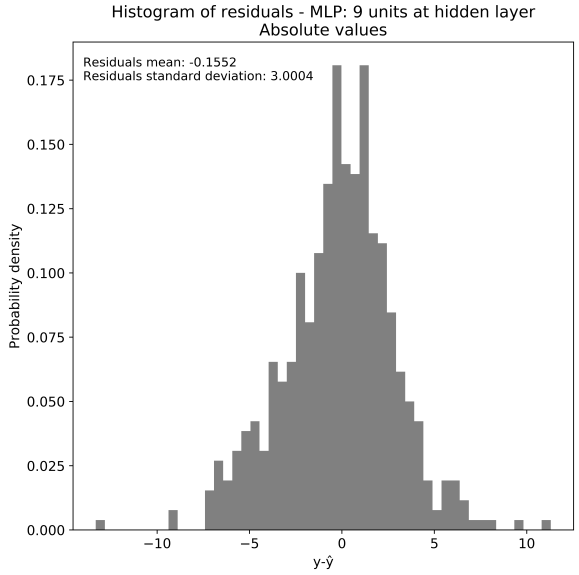
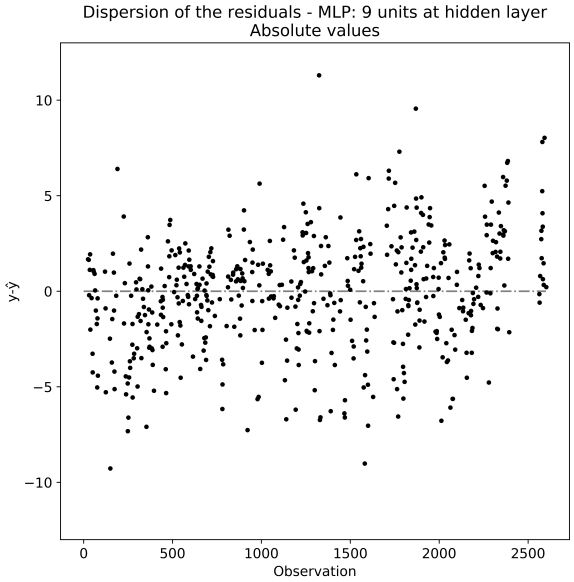


FIGURA 20 – Apresentação dos resultados do modelo de regressão MLP. Dispersão do valor estimado (ŷ) em função do valor real (y). Fonte: autoria própria.

[1] Princípio geral segundo o qual entre todos modelos concorrentes, dentre aqueles que oferecem ajuste adequado para o conjunto de dados, o que for mais simples deve ser preferido (EVERITT; SKRONDAL, 2010)



(a) Gráfico de dispersão dos resíduos ($y v s y_{estimado}$).



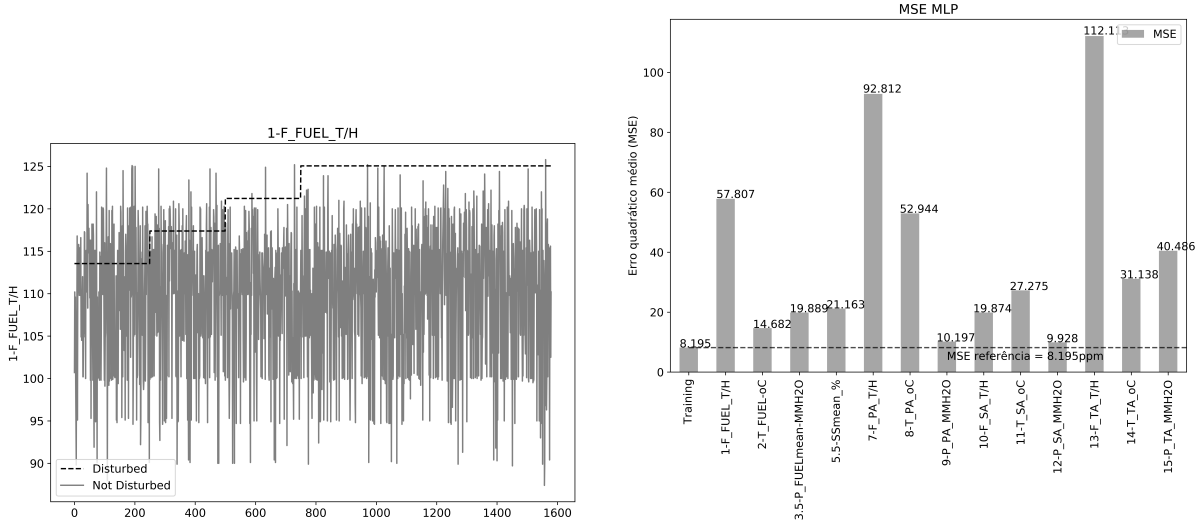
(b) Histograma dos resíduos.

FIGURA 21 – Apresentação dos resultados do modelo de regressão MLP. Fonte: autoria própria.

Observando-se a Figura 20 percebe-se que as estimativas se aproximam consideravelmente dos valores reais do teor de SO_2, ou seja, os pontos $y_{estimado}$ vs y_{real} se distribuem próximos à bissetriz (linha tracejada). A partir da análise do histograma e da dispersão dos resíduos (Figura 21), verifica-se que o erro de predição apresenta distribuição aproximadamente normal e homogênea, com média próxima de zero e desvio-padrão de 3 ppm. Tal resultado demonstra uma melhora significativa quando comparado ao modelo de regressão linear múltipla, que obteve um erro de predição médio de -0.24 ppm e um desvio-padrão de 15.8 ppm. Com essas análises, valida-se o modelo neural selecionado, cujos resultados são consideravelmente superiores ao modelo de regressão linear múltipla.

6.1.4 Análise de Sensibilidade

Considerando o conjunto definido como de identificação, foram aplicadas perturbações em cada variável de processo, uma por vez, a fim de compreender quais as variáveis teriam maior influência sobre a saída do modelo neural. A Figura 22 exemplifica a forma da perturbação aplicada a cada variável, sendo representada a vazão de licor preto, e apresenta o resultado da perturbação sobre a saída com base no cálculo do erro quadrático médio (do inglês*Mean Squared Error*, MSE).



(a) Representação do distúrbio na variável vazão de licor preto (tonh).

(b) Erro quadrático médio (MSE) resultante do distúrbio em cada variável.

FIGURA 22 – Representação das variáveis após a perturbação. Fonte: autoria própria.

A linha tracejada paralela à abscissa indica o erro quadrático médio (MSE) de referência ($MSE_{ref} = 8.20ppm$), calculado a partir do conjunto original de identificação sem qualquer perturbação. Quanto maior o valor de MSE, maior é a distância entre o valor real da variável de saída (teor de SO_2) e o valor estimado pelo modelo, isso indica que para as predições com as variáveis modificadas, quanto maior o MSE maior tende a ser a influência da variável alterada sobre a variável estimada. Assim, observa-se que as variáveis vazão de ar terciário ($MSE = 112.11ppm$), vazão do ar primário ($MSE = 92.81ppm$) e vazão do combustível ($MSE = 57.81ppm$) exercem uma maior influência sobre o teor de SO_2 estimado; ao passo que as variáveis pressão do ar primário ($MSE = 10.20ppm$) e pressão do ar secundário ($MSE = 9.93ppm$) apresentam a menor influência.

A Figura 23 representa o diagrama de causa-efeito construído com base nos resultados anteriores de MSE. Os percentuais se referem à diferença entre MSE da variável e o MSE de referência em relação à soma total dessas diferenças, conforme na Equação 5.11 (Capítulo 5).

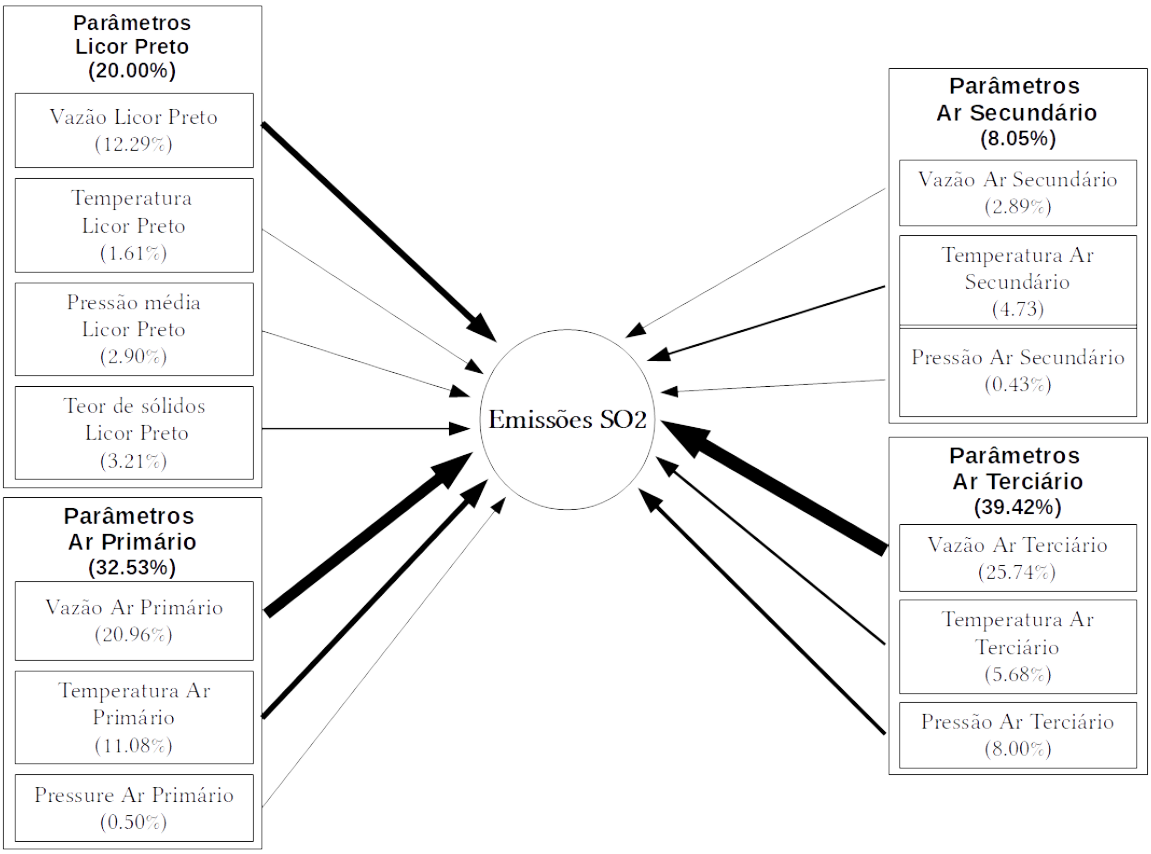


FIGURA 23 – Representação gráfica das relações causais entre as variáveis de entrada do processo e o teor de SO_2, variável de saída. Fonte: autoria própria.

A visualização dos dados a partir da Figura 23 permite uma compreensão mais global das relações entre as variáveis de um sistema multivariado. As entradas foram agrupadas em quatro conjuntos: variáveis relacionadas ao combustível, ao ar primário, ao ar secundário e ao ar terciário.

A influência da vazão de licor sobre as emissões de dióxido de enxofre é um resultado esperado, tendo em vista que tais emissões dependem diretamente da quantidade de enxofre que entra na caldeira, que advém da carga de licor e da sua sulfidez. A entrada de ar terciário é responsável pelo aumento da mistura dos efluentes gasosos e consequente queima de compostos não oxidados remanescentes, inclusive da combustão de compostos de enxofre reduzidos (*Total Reduced Sulfur*, TRS), o que poderia justificar a grande influência que a variável vazão de ar terciário exerce sobre a saída da rede. O ar primário tem a função de oxidar a matéria orgânica o suficiente para manter os inorgânicos não oxidados (para a recuperação dos reagentes da digestão da madeira, contendo hidróxido e sulfeto de sódio - NaOH e Na_2S) e manter o leito uniformemente distribuído no fundo da fornalha, assim, o ar primário provavelmente exerce influência nas emissões de TRS que, em regiões mais elevadas da caldeira, podem ser convertidos em SO_2 (VAKKILAINEN, 2005).

Contudo, era de se esperar que a vazão do ar secundário também exercesse grande influência sobre a saída do modelo, já que este é responsável pelo controle da temperatura da fornalha e, com isso, pelas emissões de voláteis contendo sódio e potássio, capazes de reagir com o enxofre gasoso e reduzir as emissões de SO_2. Este resultado não foi observado pela análise de sensibilidade, uma das razões pode decorrer do fato de que existe uma correlação elevada entre a vazão de combustível e de ar secundário (correlação 73%) podendo ter levado a rede neural a atribuir um peso maior para uma variável em detrimento da outra, que transpareceu no resultado da análise de sensibilidade. Neste sentido, uma das possibilidades de amenizar essa resposta seria aplicar alguma técnica para transformar as variáveis originais em variáveis menos correlacionadas, um exemplo seria a substituição das variáveis originais por um novo conjunto de variáveis ortogonais.

O resultado dessa análise é importante para compreender o conjunto de parâmetros manipuláveis possíveis de serem modificados para que sejam obtidos resultados mais significativos sobre a variável resposta. Em específico, o controle mais preciso e atento das variáveis levantadas pela análise de sensibilidade pode levar a um maior controle do SO_2, proporcionando uma operação mais estável.

6.2 ESTUDO DE CASO 2: CALDEIRA NA FINLÂNDIA

6.2.1 Pré-Processamento

Inicialmente, estavam disponíveis 8760 observações e 28 variáveis, dessas 20 seriam consideradas variáveis de entrada e 8 variáveis de saída, dependentes. A Tabela 4 apresentada na seção Descrição dos casos de estudo, tópico 4.2, mostra as variáveis e suas respectivas unidades de medida. Selecionou-se apenas o teor de material particulado (legenda *33_DustFlueGas_conc_mgNm3*) como variável de saída. Essa escolha deve-se à distribuição e disponibilidade dos dados, consideradas mais adequadas para a aplicação de técnicas de regressão e de análise de sensibilidade quando comparada às outras possibilidades de variáveis de saída. Além disso, do ponto de vista operacional, a compreensão do comportamento das emissões de particulados torna-se bastante importante, tendo em vista os problemas ambientais e relacionados ao desenvolvimento de doenças respiratórias em pessoas que circulam nas proximidades das fábricas, bem como os problemas de incrustação nas superfícies de trocas térmicas dos equipamentos de geração de vapor na caldeira (VAKKILAINEN, 2005).

As variáveis de entrada foram observadas individualmente e em conjunto, a fim de serem selecionadas apenas aquelas que seriam mais representativas e que poderiam trazer informações mais consistentes ao processo. Foram observadas as relações entre as três variáveis associadas ao fluxo de licor preto (fluxo total, medição

do fluxo do lado esquerdo e medição do fluxo do lado direito). A Tabela 6 apresenta o coeficiente de correlação entre essas três variáveis. Como pode ser observado, existe uma alta correlação entre as variáveis de fluxo do licor e, por isso, apenas a variável fluxo de licor total foi mantida, sendo as demais removidas da base de dados a ser considerada.

TABELA 6 – Coeficiente de correlação entre as variáveis de fluxo de entrada do licor preto

	Fluxo Total (Ls)	Fluxo Esquerdo (Ls)	Fluxo Direito (Ls)
Fluxo Total (Ls)	1.000	0.984	0.980
Fluxo Esquerdo (Ls)	0.984	1.000	0.951
Fluxo Direito (Ls)	0.980	0.951	1.000

Fonte: autoria própria.

Na base de dados original, havia quatro medições da pressão de entrada do licor preto: esquerda, direita, a frente e atrás. Calculou-se a média das pressões e observou-se o coeficiente de correlação entre as quatro medidas e a sua média. Observando a Tabela 7, percebe-se uma correlação razoável entre a pressão média e cada uma das outras medições de pressão, com isso as quatro medições de pressão foram substituídas pela pressão média do licor preto.

TABELA 7 – Coeficiente de correlação entre as variáveis de pressão de entrada do licor preto

	Pressão a frente (bar)	Pressão atrás (bar)	Pressão esquerda (bar)	Pressão direita (bar)	Pressão média (bar)
Pressão a frente (bar)	1	0.699	0.750	0.762	0.886
Pressão atrás (bar)	0.699	1	0.761	0.748	0.879
Pressão esquerda (bar)	0.750	0.761	1	0.965	0.945
Pressão direita (bar)	0.762	0.748	0.965	1	0.945
Pressão média (bar)	0.886	0.879	0.945	0.945	1

Fonte: autoria própria.

Em resumo, considerou-se um conjunto com 15 variáveis de entrada e uma variável de saída. Dada a seleção de variáveis, foram removidas as linhas de observações contendo pelo menos uma variável com valor igual a zero, resultando em uma base de dados com 16 variáveis e 7005 observações, no total. Em seguida, aplicou-se o filtro de

Hampel, para cada variável individualmente. Os valores discrepantes foram removidos, restando, ao final, 3058 observações. A Figura 24 apresenta a série temporal do teor de material particulado nos gases efluentes (variável de saída), nos momentos anterior e posterior à aplicação do Filtro de Hampel.

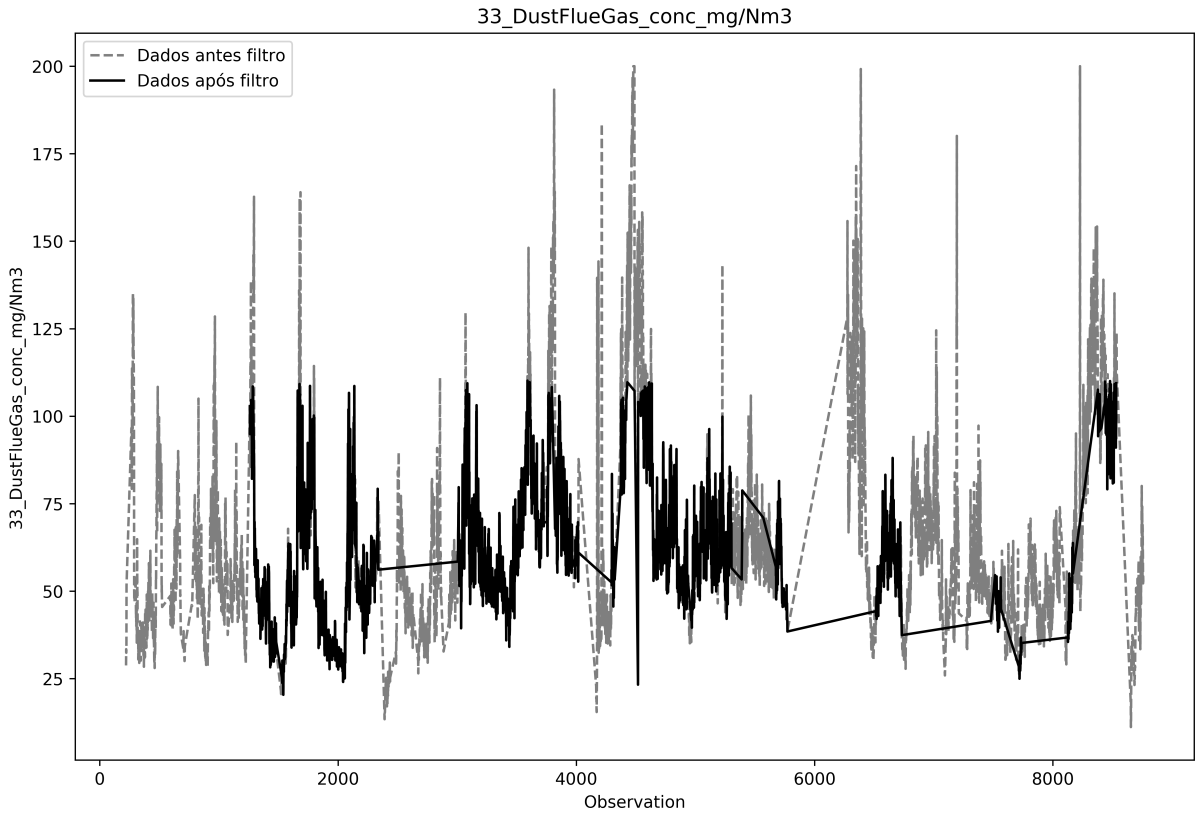


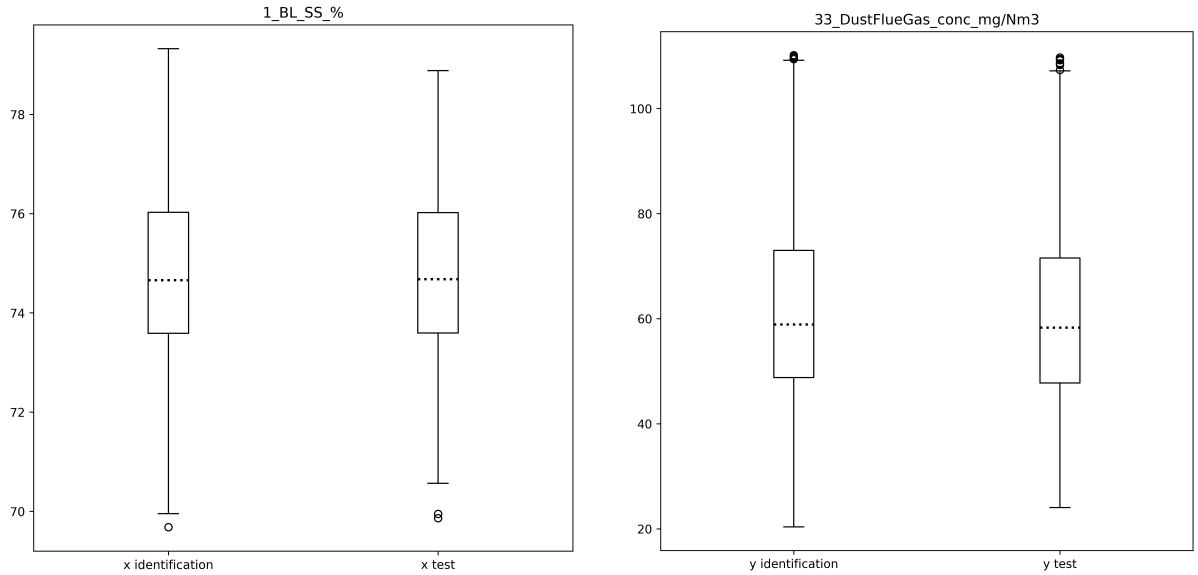
FIGURA 24 – Gráficos temporais para a variável teor de sólidos nos gases de emissão, nos momentos de antes da aplicação do Filtro e após a aplicação do Filtro de Hampel. Fonte: autoria própria.

6.2.2 Geração dos conjuntos de dados: Identificação e Teste

Da mesma forma como no primeiro estudo de caso, o conjunto de dados com 15 variáveis de entrada e 3058 obervações foi separado nos conjuntos de identificação e de teste, empregando a função *train_test_split* do módulo *model_selection* do pacote *scikit-learn* do Python, com semente igual a 42, e com a proporção 75% para identificação e 25% para teste. Isso significa que 2295 observações foram alocadas no conjunto de identificação e 763, no conjunto de teste. Os dados foram normalizados utilizando escala mínimo máximo com amplitude entre -1 e 1. Cabe ressaltar que quando a rede neural foi testada com função de ativação logística, a normalização da variável resposta foi realizada com amplitude entre 0 e 1, coerente com o seu conjunto imagem.

Após a etapa de normalização, gráficos do tipo boxplot foram construídos para cada variável a fim de se comparar as faixas de valores em cada conjunto. A Figura 25

apresenta, comparativamente, os dados de identificação e de teste, para duas variáveis de entrada.



(a) Teor de sólidos no licor (%). (b) Teor de material particulado (mg/Nm^3).

FIGURA 25 – Boxplot dos conjuntos de identificação e teste para variáveis. Fonte: autoria própria.

Como representado na Figura 25, as demais variáveis apresentam comportamento bastante semelhante ao dessas, o que indica que a distribuição estatística do conjunto de teste se assemelha bastante com a do conjunto de identificação. Ou seja, a separação do banco de dados entre conjunto de identificação e conjunto de teste não descaracteriza, em termos estatísticos, um dos conjuntos em relação ao outro.

Com os conjuntos separados e já normalizados, seguem-se as etapas de construção de modelos de sensores virtuais, o primeiro com base em regressão linear múltipla, e o segundo baseado em rede perceptron multicamadas.

6.2.3 Modelos de Sensores Virtuais

6.2.3.1 *Regressão Linear Múltipla*

O modelo de regressão linear múltipla foi construído utilizando a função *LinearRegression* do módulo *linear_model* do pacote *scikit-learn*. Os dados de entrada e saída normalizados foram usados para a construção do modelo de regressão. Após a obtenção do modelo, a matriz de entrada do conjunto de teste foi aplicada para verificar a capacidade de predição do modelo. Os valores preditos foram, então, comparados com os valores de saída do conjunto de teste. Os coeficientes da regressão calculados foram:

$\beta 0 = -0.152370\ (intercepto)$ e

$$\beta = [0.30466598;\ 0.09448704;\ -0.19617666;\ 0.05219924;\ -0.0404217;\ 0.26437493;\ 0.16051346;\ -0.01189688;\ 0.06750854;\ 0.50967292;\ -0.15384248;\ 0.23456869;\ -0.43419502;\ 0.10756773;\ -0.10165126]$$

A saída do modelo está normalizada. Portanto, é necessário aplicar a transformação inversa e retornar os valores à escala original. Esse procedimento foi realizado antes da apresentação dos resultados. Pela Figura 26 é possível comparar as estimativas do modelo com os valores reais assumidos pela variável de saída, o teor de material particulado.

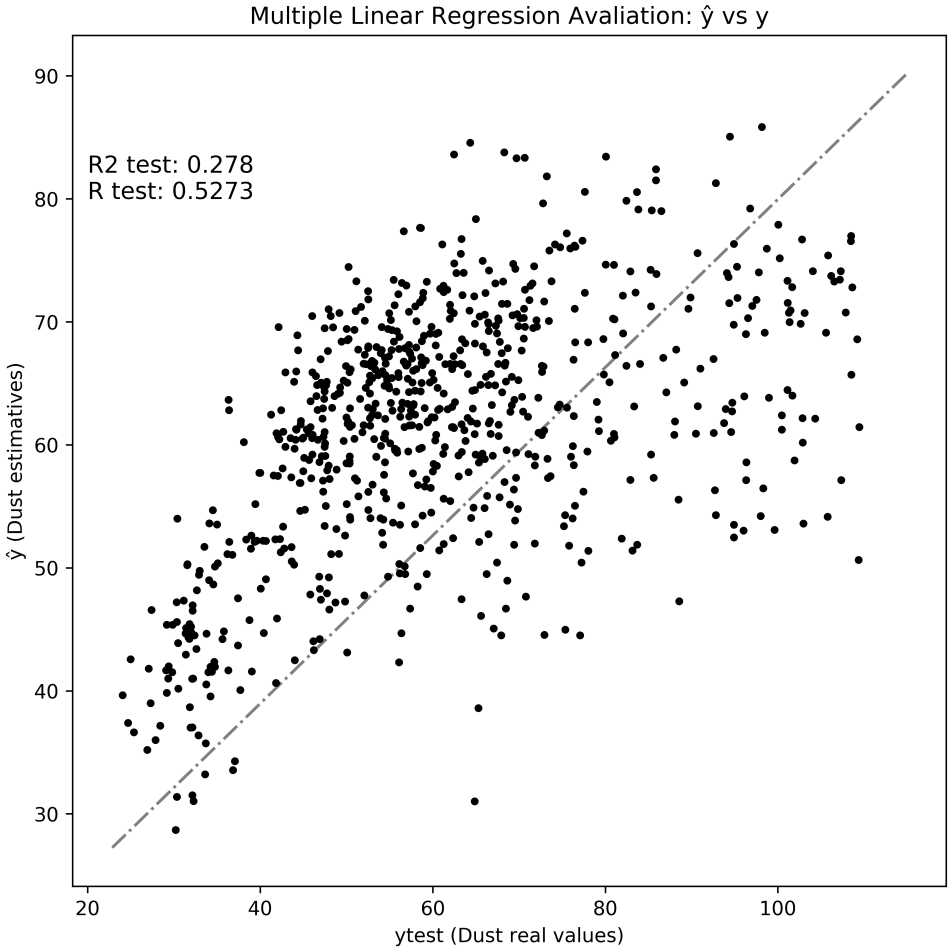
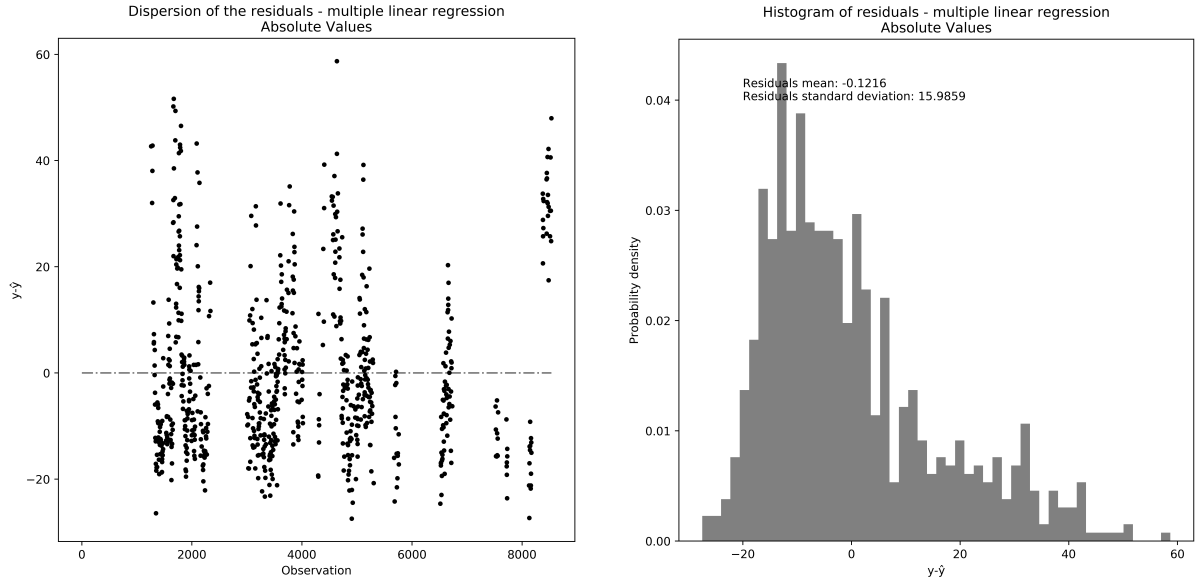


FIGURA 26 – Avaliação do resultado da Regressão Linear Múltipla, teor de particulado estimado versus teor de particulado medido.Fonte: autoria própria.

O coeficiente de determinação, R^2 foi 0.278. Isso indica que aproximadamente 28% da variância da variável de saída, no caso, teor de material particulado nos gases de emissão, é predito pelas variáveis independentes, com base no modelo de regressão linear múltipla. A Figura 27 exibe o comportamento dos resíduos (valor real subtraído do valor estimado, ou seja, y-ŷ), por meio do gráfico de dispersão e de histograma dos resíduos.



(a) Gráfico de dispersão dos resíduos. (b) Histograma dos resíduos.

FIGURA 27 – Visualização do perfil dos resíduos (y-ŷ) obtidos a partir da regressão linear múltipla. Fonte: autoria própria.

Observando o gráfico da regressão (y versus ŷ) percebe-se que para valores mais baixos de material particulado, o modelo superestima o valor real, ou seja propõe um valor estimado maior que o real; e, para valores mais elevados de teor de particulado ocorre o inverso: o modelo gera estimativas menores que os valores originais. Em geral, em casos de superestimação, o desvio absoluto entre o valor original e o resultado do modelo (erro) é menor que para os casos de subestimação. Essa constatação pode ser confirmada pelo histograma e pelo gráfico de dispersão do erro. Isso indica que o modelo linear não é capaz de predizer de forma satisfatória, a partir das variáveis de processo, o teor de material particulado nas emissões da caldeira.

6.2.3.2 *Rede Perceptron Multicamadas*

Para a rede neural, com o objetivo de escolher a topologia que melhor se adéque ao problema, foram testadas variações em alguns hiperparâmetros internos da rede. Os resultados da técnica de validação cruzada, aplicada a cada uma das topologias testadas, foram comparados e assim definiram-se os hiperparâmetros a serem utilizados.

Seguindo os mesmos passos do estudo de caso anterior, a validação cruzada foi conduzida com base na função *KFold*, do módulo *model_selection*, do pacote *scikit-learn*, do Python. O conjunto de identificação (com o total de 2295 observações) foi dividido em 5 partições, cada uma contendo 459 observações, que foram usadas para treinar e validar todas as possíveis combinações de hiperparâmetros para a construção da rede. A divisão em partições, assim como a separação anterior em identificação e

teste, foi realizada de forma pseudo aleatória, usando uma semente para a geração dos números aleatórios que orientam a distribuição das observações para as 5 partições. Isso permitiu a comparação isonômica dos resultados obtidos por cada combinação de hiperparâmetros.

Dos resultados da validação cruzada foi observado que, assim como no estudo de regressão do SO_2, para ambas as funções de ativação, logística e tangente hiperbólica, o método de otimização e a taxa de aprendizado que oferecem o maior coeficiente de correlação, independentemente do número de neurônios na camada oculta, são, respectivamente, o método de otimização LBFGS e a taxa de aprendizado constante.

A Figura 28 ilustra os resultados da validação cruzada, em termos do coeficiente de correlação médio (r_{medio}, média dos coeficientes de correlação obtido por cada conjunto de validação cruzada), com o desvio-padrão associado, para cada um dos tipos de função de ativação testados, segundo a variação do número de neurônios na camada oculta.

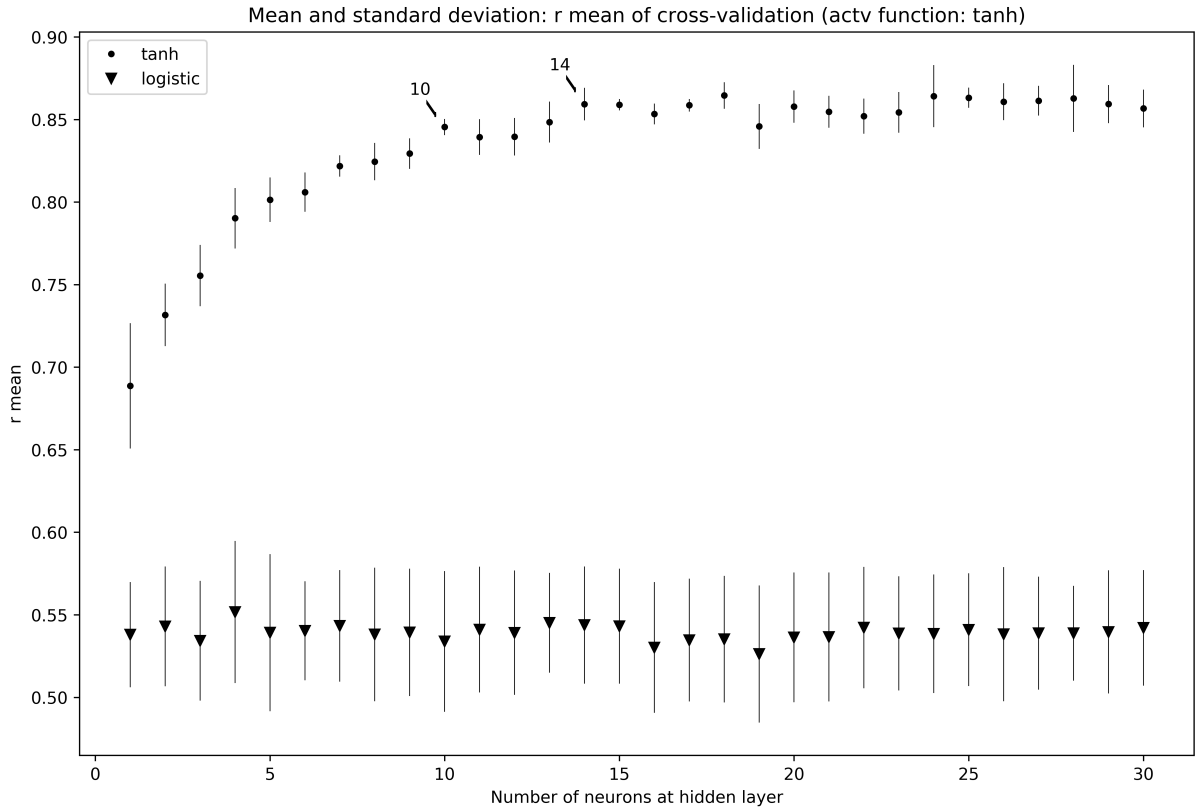


FIGURA 28 – Evolução do r_{medio} da validação cruzada conforme o número de neurônios na camada oculta, para as funções de ativação logística e tangente hiperbólica. Fonte: autoria própria.

Observando as médias e os desvios-padrões do coeficiente de correlação calculado a partir do procedimento de validação cruzada, representados pela Figura 28, pode-se concluir que a rede que emprega a função de ativação tangente hiperbólica

($tanh$) alcançou melhores resultados em comparação àquela que utiliza a função logística. Portanto, a função de ativação $tanh$ será utilizada na rede que deve funcionar como sensor virtual para predição do teor de material particulado. O próximo passo é a determinação do número de neurônios na camada oculta.

De acordo com a Figura 28, as redes com 10 e 14 neurônios na camada oculta apresentaram resultados satisfatórios, com coeficientes de correlação (r_{medio}) maiores que 0.80. Entretanto, apesar de o coeficiente de correlação médio (r_{medio}) da rede com 14 neurônios ($r_{medio} = 0.8593$) ser maior que o da rede com 10 ($r_{medio} = 0.8455$), pelo princípio da parcimônia, é interessante investigar se o aumento de complexidade causado pelo acréscimo de uma unidade neuronal compensa o aparente ganho na capacidade de predição. A partir da Figura 29 é possível perceber que o aumento percentual em r_{medio} quando se transita de 9 para 10 neurônios (aumento de 1.94%) é maior do que o aumento percentual decorrente da mudança de 13 para 14 neurônios (aumento de 1.28%).

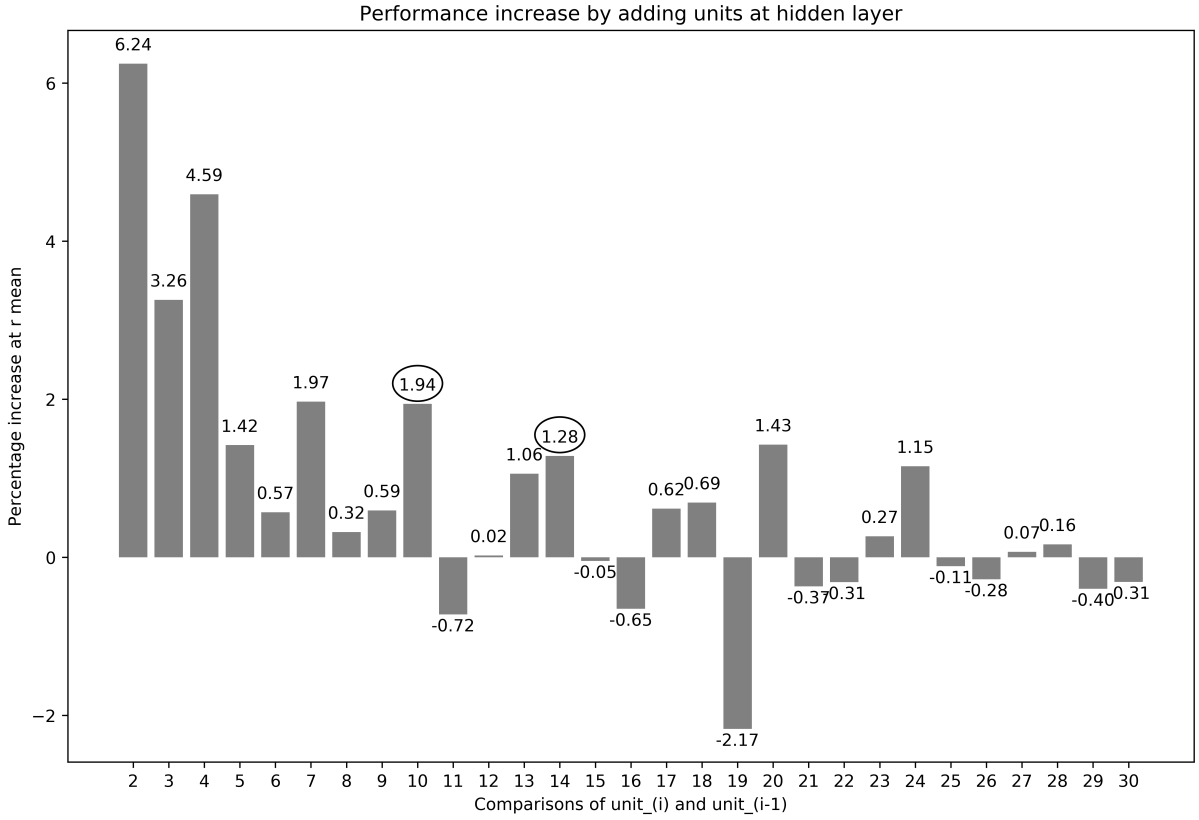


FIGURA 29 – Aumento percentual em r_{medio} pelo acréscimo de neurônios na rede MLP. Fonte: autoria própria.

Sendo razoavelmente pequena a diferença na qualidade dos resultados entre o modelo com 10 e com 14 neurônios, com um aumento em r_{medio} pouco maior que 1%, e a melhoria na qualidade da regressão de 9 para 10 neurônios mais significativa que de 13 para 14, concluiu-se que a rede com 10 neurônios ocultos é a mais adequada

para o modelo de regressão do teor de material particulado. Deste modo, a rede com a topologia definida (função de ativação tangente hiperbólica, método LBFGS de estimação de parâmetros, taxa de aprendizagem constante e 10 neurônios na camada oculta) foi treinada com o conjunto completo de identificação. Em seguida, testou-se a rede neural com o conjunto de dados de teste. Os resultados para o conjunto teste são apresentados nas Figuras 30 e 31.

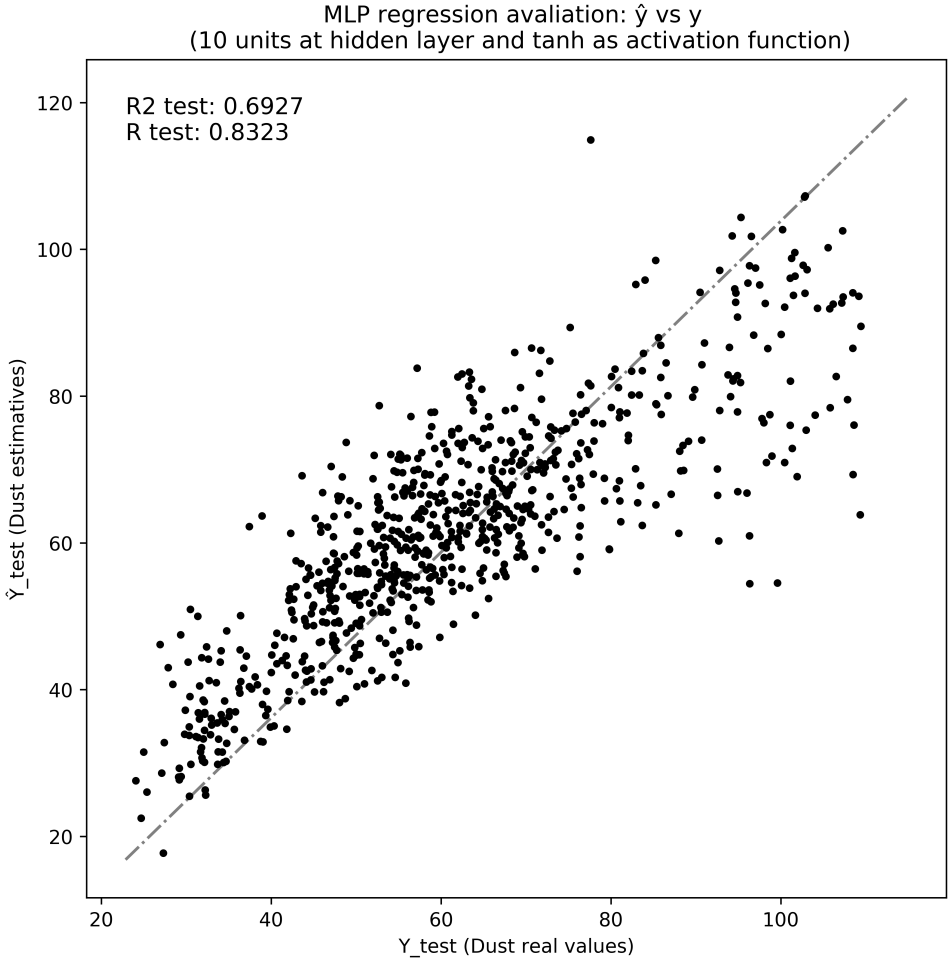
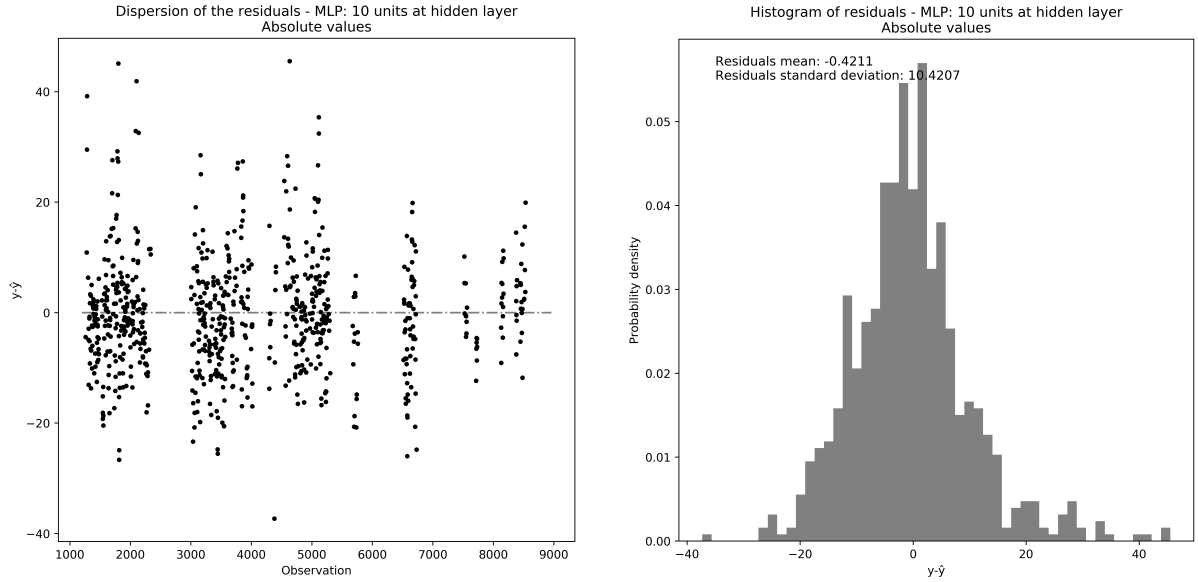


FIGURA 30 – Dispersão do valor estimado (ŷ) em função do valor real (y). Resultado para a rede MLP com 10 neurônios na camada oculta. Fonte: autoria própria.

Para a avaliação dos resíduos, dado pela diferença entre o valor real e o estimado (y-ŷ), foram construídos gráfico de dispersão e histograma da distribuição dos resíduos. A Figura 31 apresenta os resultados.

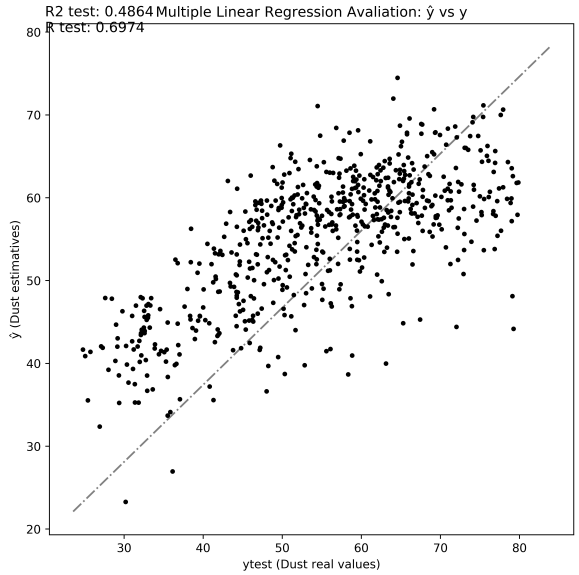


(a) Gráfico de dispersão dos resíduos. (b) Histograma dos resíduos.

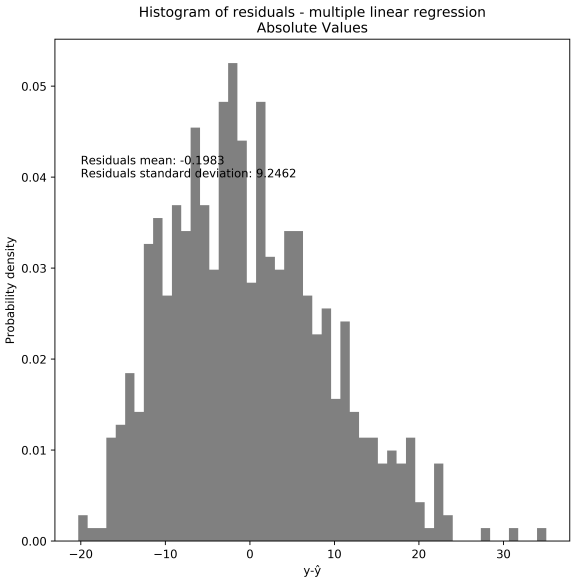
FIGURA 31 – Visualização do perfil dos resíduos ($y - y_{estimado}$) obtidos a partir da rede MLP com 10 neurônios na camada oculta. Fonte: autoria própria.

Como apresentado nas figuras, o modelo consegue predizer de forma satisfatória o teor de material particulado nos gases de saída da caldeira a partir das variáveis de processo consideradas. Sendo o coeficiente de correlação (r) referente aos dados de teste igual a 0.8323, a média dos resíduos $y - y_{estimado}$ em torno de zero (-0.4211) e, como pode ser observado pelo histograma, os resíduos seguem aproximadamente uma distribuição normal, com desvio-padrão de 10.4207. Observa-se que o modelo neural obteve resultado de regressão consideravelmente superior que o modelo linear com múltiplas variáveis ($r = 0.5273$). Isso indica que o mecanismo de produção e emissão do material particulado tende a ser não linear. Entretanto, analisando as Figuras 30 e 31, pode-se perceber que a qualidade da regressão piora significativamente a partir de um valor alto de concentração de material particulado, em torno de 80 mg/Nm^3.

Com a finalidade de aprimorar a acurácia do modelo e realizar uma análise de sensibilidade mais refinada, os valores de concentração de material particulado acima do limite de 80mg/Nm^3 foram descartados da base de dados de trabalho, e uma nova rede neural MLP foi identificada, treinada e testada. Os resultados do procedimento de identificação de um modelo de regressão linear múltipla são apresentados na Figura 32.



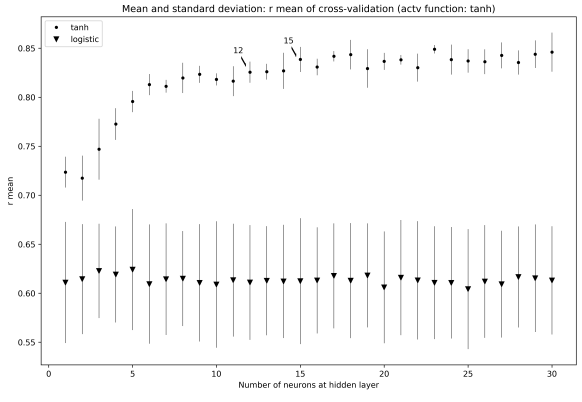
(a) Dispersão do valor estimado (ŷ) em função do valor real (y). Fonte: autoria própria.



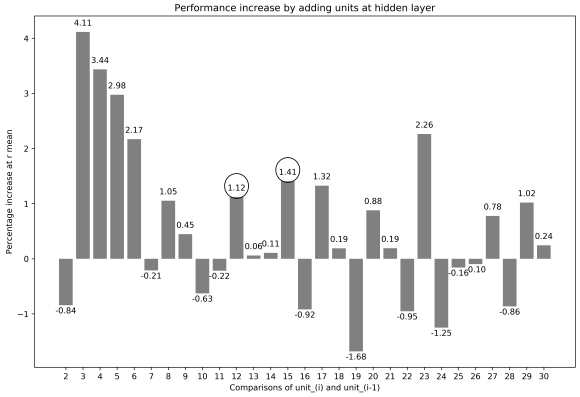
(b) Histograma dos resíduos.

FIGURA 32 – Visualização dos resultados obtidos pela regressão linear múltipla. Fonte: autoria própria.

O modelo de regressão linear múltipla considerando apenas os dados de trabalho que apresentaram teor de material particulado acima de 80mg/Nm^3 alcançou coeficiente de correlação r igual a 0.6974. Ou seja, a regressão nesse caso obteve resultado melhor que aquela obtida para a base de dados de trabalho completa, indicando que, possivelmente, os resultados da rede neural também seriam superiores. Assim, procedeu-se à validação cruzada para identificação do modelo MLP, empregando-se o conjunto de identificação, criado a partir dessa nova base de dados (material particulado inferior a 80mg/Nm^3), o resultado é apresentado na Figura 33.



(a) Evolução do r_{medio} da validação cruzada conforme o número de neurônios na camada oculta, para as funções de ativação logística e tangente hiperbólica. Fonte: autoria própria.



(b) Aumento percentual em r_{medio} pelo acréscimo de neurônios na rede MLP, considerando a função de ativação tangente hiperbólica.

FIGURA 33 – Evolução dos coeficientes de correlação médio conforme o número de neurônios na camada oculta. Fonte: autoria própria.

De acordo com a Figura 33, os resultados da rede neural com função de ativação tangente hiperbólica (*tanh*) se mostraram superiores aos da rede com função de ativação logística, independentemente do número de neurônios na camada oculta. Observando a evolução de r_{medio}, Figura 33a, para a rede que emprega a função *tanh*, pode se destacar as redes com 12 e 15 neurônios, com coeficientes de correlação (r_{medio}) iguais a 0.8255 e 0.8386. Para verificar se o aumento no coeficiente de correlação justifica o acréscimo de complexidade, a partir da Figura 33b, percebe-se que o aumento percentual em r_{medio} quando se transita de 11 para 12 neurônios (aumento de 1.12%) é menor do que o aumento percentual decorrente da mudança de 14 para 15 neurônios (aumento de 1.41%). Como ambas as possibilidades apresentaram r_{medio} entre 0.80 e 0.85, e o acréscimo nesse coeficiente calculado a partir do aumento de uma unidade neuronal apresenta uma diferença relativamente pequena (0.3%), decidiu-se pela rede de menor complexidade, com 12 neurônios na camada oculta.

Deste modo, a rede com a topologia definida (função de ativação tangente hiperbólica, método LBFGS de estimação de parâmetros, taxa de aprendizagem constante e 12 neurônios na camada oculta) foi treinada com o conjunto de identificação. Deve-se lembrar que ambos os conjuntos foram gerados a partir da base de dados contendo apenas valores para o teor de material particulado inferiores a 80mg/Nm^3. Em seguida, testou-se a rede neural com o conjunto de dados de teste. Os resultados para o conjunto teste são apresentados nas Figuras 34 e 35.

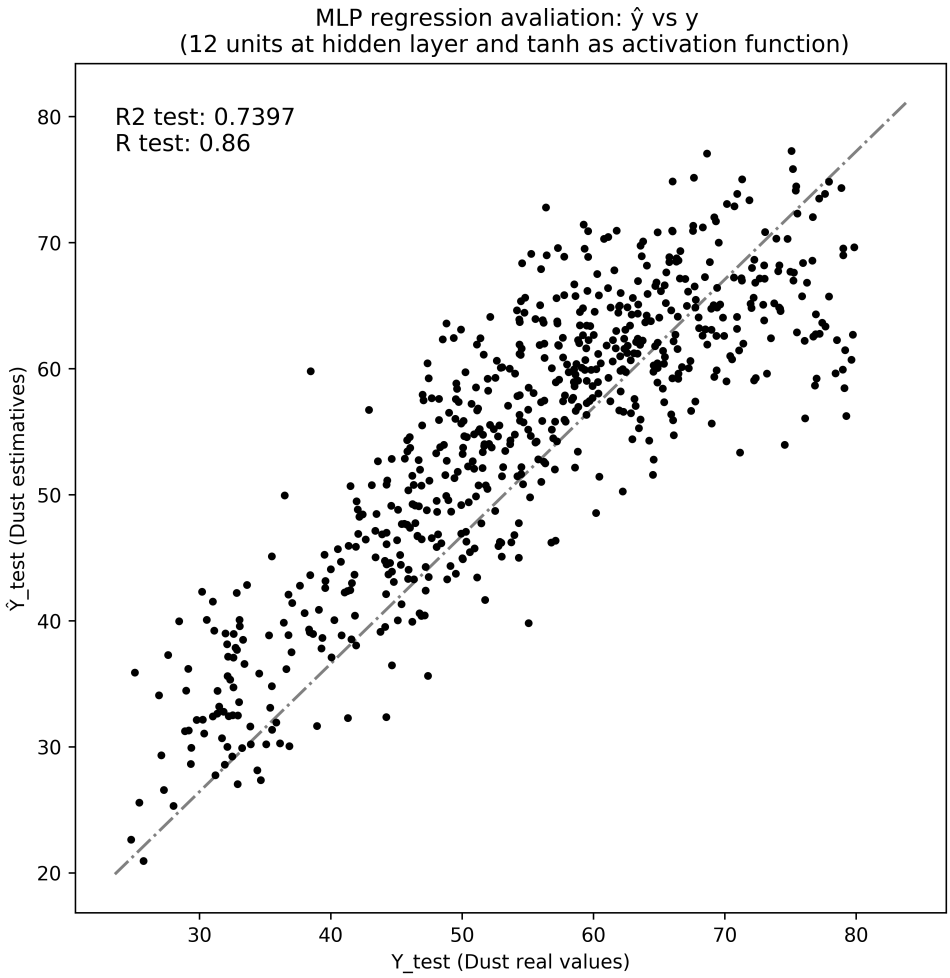
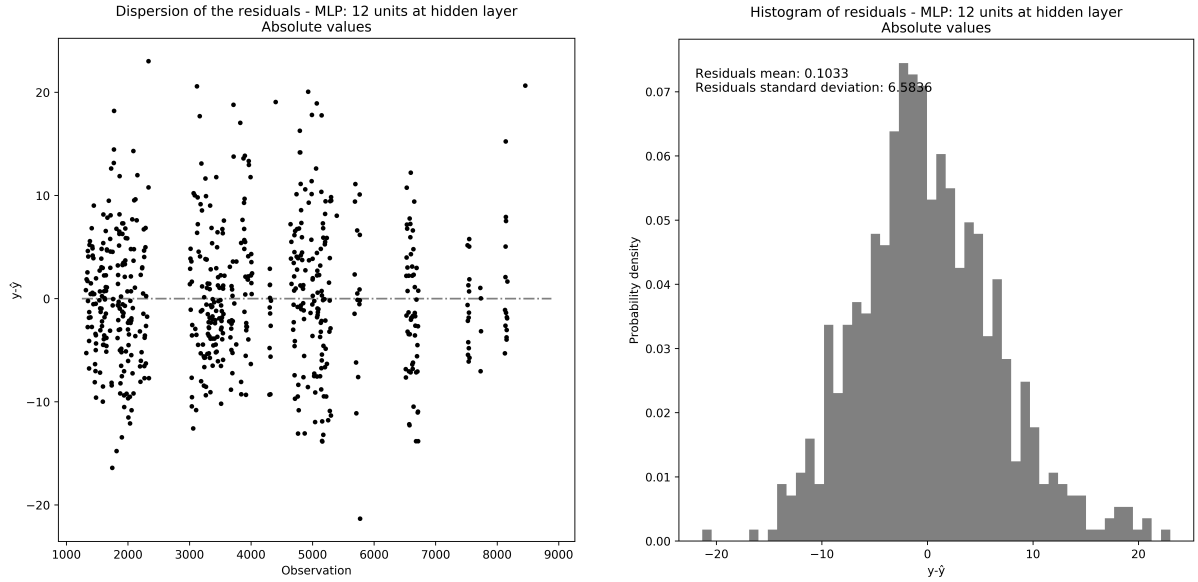


FIGURA 34 – Dispersão do valor estimado (ŷ) em função do valor real (y). Resultado para a rede MLP com 12 neurônios na camada oculta. Fonte: autoria própria.

Para a avaliação dos resíduos, dado pela diferença entre o valor real e o estimado (y-ŷ), foram construídos gráfico de dispersão e histograma da distribuição dos resíduos. A Figura 35 apresenta os resultados.



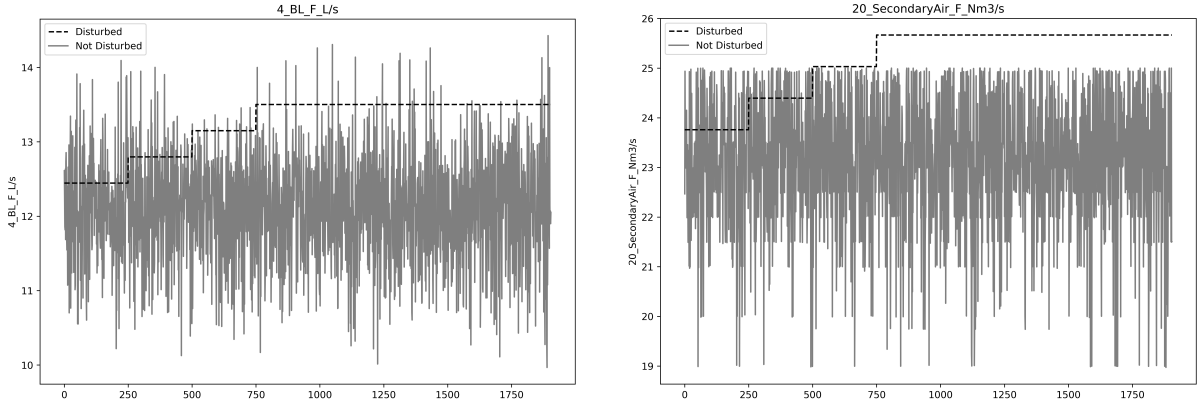
(a) Gráfico de dispersão dos resíduos. (b) Histograma dos resíduos.

FIGURA 35 – Visualização do perfil dos resíduos ($y - y_{estimado}$) obtidos a partir da rede MLP com 12 neurônios na camada oculta. Fonte: autoria própria.

Como apresentado nas figuras, o modelo consegue predizer de forma satisfatória o teor de material particulado nos gases de saída da caldeira a partir das variáveis de processo consideradas. Sendo o coeficiente de correlação (r) referente aos dados de teste igual a 0.860, a média dos resíduos $y - y_{estimado}$ entorno de zero (0.1033) e, como pode ser observado pelo histograma, os resíduos seguem aproximadamente uma distribuição normal, com desvio-padrão de 6.5836. Com essas análises foi possível validar o modelo neural selecionado. Segue a análise de sensibilidade, dado o conjunto de dados selecionado a partir da base de dados de trabalho em que o teor de material particulado é inferior a 80mg/Nm^3.

6.2.4 Análise de Sensibilidade

Seguindo a mesma sequência do estudo de caso anterior, considerando o conjunto definido como de identificação, foram aplicadas perturbações em cada variável de processo, uma por vez, a fim de compreender quais as variáveis teriam maior influência sobre a saída do modelo neural. A Figura 36 exemplifica a forma da perturbação aplicada a cada variável, sendo representado o teor de sólidos presente no licor preto e a vazão do ar secundário.



(a) Representação do distúrbio na variável teor de sólidos no licor (Nm^3/s).

(b) Representação do distúrbio na variável vazão de ar secundário (%).

FIGURA 36 – Representação das variáveis após a perturbação. Fonte: autoria própria.

A rede neural previamente treinada, identificada e validada foi empregada, inicialmente, para predição do teor de material particulado utilizando o conjunto de identificação sem qualquer perturbação, ao final o erro quadrático médio (MSE_{ref}) de referência foi calculado. Posteriormente, o procedimento foi repetido para cada variável modificada, calculando-se o MSE correspondente. Com os resultados, o gráfico apresentado na Figura 37 foi construído. A linha tracejada paralela à abscissa indica o erro quadrático médio de referência (MSE_{ref}).

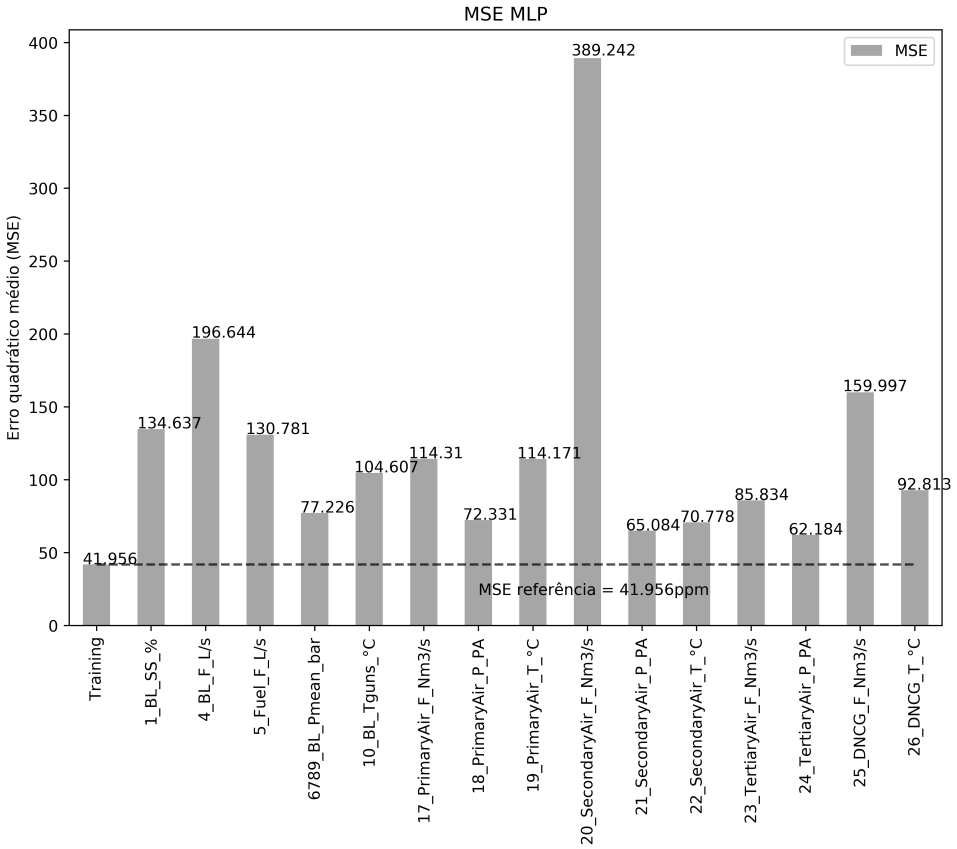


FIGURA 37 – Erro quadrático médio (MSE) resultante do distúrbio em cada variável.

Quanto maior o valor de MSE, maior é a distância entre o valor real da variável de saída (teor de particulado) e o valor estimado pelo modelo. Isso indica que, para as predições com as variáveis modificadas, quanto maior o MSE, maior tende a ser a influência da variável alterada sobre a variável estimada. Assim, observa-se que as variáveis vazão do ar secundário ($MSE = 389.24mg/Nm^3$), vazão do licor preto ($MSE = 196.64mg/Nm^3$) e vazão de gases não condensáveis diluídos ($MSE = 160.00mg/Nm^3$), e teor de sólidos no licor ($MSE = 134.64mg/Nm^3$) e vazão de óleo combustível ($MSE = 130.78mg/Nm^3$) exercem influência bastante significativa sobre a estimativa do teor de material particulado na emissão; ao passo que as variáveis pressão do ar terciário ($MSE = 62.18mg/Nm^3$), pressão do ar secundário ($MSE = 65.08$), pressão do ar primário ($MSE = 72.33mg/Nm^3$) e pressão média do licor ($MSE = 77.23mg/Nm^3$) exercem a menor influência.

A Figura 38 representa o diagrama de causa-efeito construído com base nos resultados anteriores de MSE. Os percentuais se referem à diferença entre o MSE da variável e o MSE de referência em relação à soma conforme na Equação 5.11 (Capítulo 5).

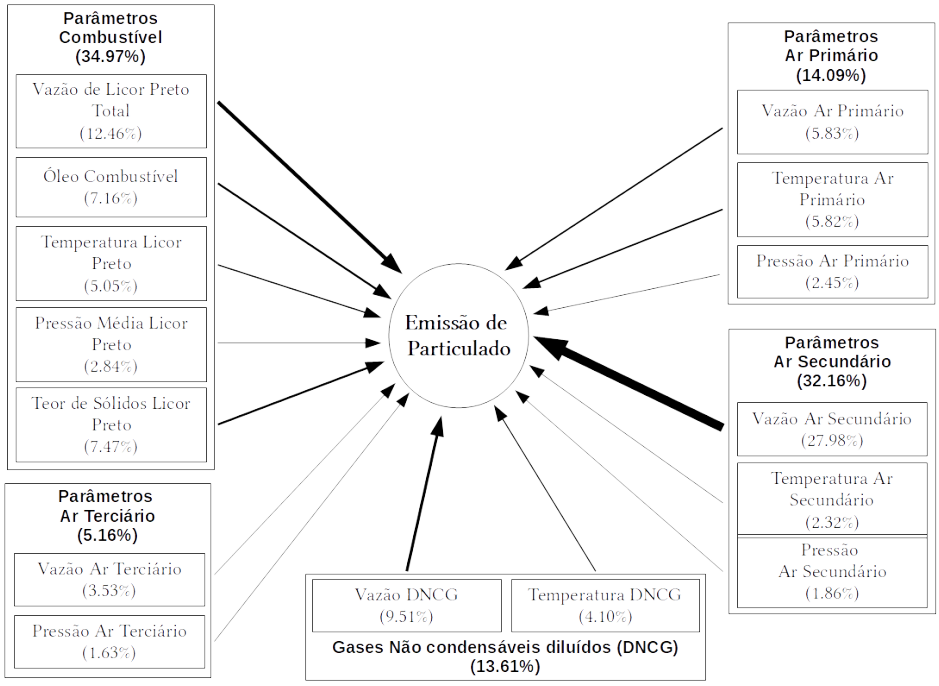


FIGURA 38 – Representação gráfica das relações causais entre as variáveis de entrada do processo e as emissões de material particulado, variável de saída. Fonte: autoria própria.

As variáveis de entrada foram agrupadas em cinco conjuntos: variáveis relacionadas ao combustível, ao ar primário, ao ar secundário, ao ar terciário e aos gases não condensáveis diluídos (DNCG). Dessa forma, é possível compreender, a partir da visualização, as interações entre os grupos de variáveis de entrada e a saída.

Segundo Vakkilainen (2005), a emissão de material particulado é dependente das reações com álcalis vaporizados (especialmente, sódio e potássio) e da quantidade de material, em estado fundido ou sólido, carregado pelos gases de combustão. A vaporização de álcalis depende, principalmente, da quantidade de sólidos no fluxo de licor de entrada e da temperatura da superfície do leito, enquanto o carreamento de particulado é dependente da configuração e do sistema dos fluxos de ar (primário, secundário e terciário). Neste sentido, os resultados da análise de sensibilidade são coerentes com a operação real da caldeira, descrita pela referência. A vazão de ar secundário interfere diretamente na queima do combustível, na temperatura da fornalha e nos fluxos gasosos, o que pode influenciar a taxa de volatilização de álcalis, bem como as reações que acontecem dentro da caldeira, além do carreamento de particulados, por isso é esperado que, sobre o resultado da análise de sensibilidade essa variável se apresente como muito influente.

A vazão de licor em conjunto com o teor de sólidos apontam a carga de orgânicos e inorgânicos que alimenta a caldeira. Considerando as emissões de particulados, os orgânicos se relacionam com temperatura do forno e os inorgânicos constituem a

fonte de elementos que podem se volatilizar para formar cinzas (material particulado), o que confirma a coerência entre os resultados apontados pela análise de sensibilidade e a operação real da caldeira.

Os gases não condensáveis diluídos (DNCG) podem ser formados durante os processamentos de licor preto e de licor branco, durante a lavagem da polpa de celulose, e ao longo dos canais com resíduos de licor preto e branco. Deve-se lembrar que as principais fontes de emissão desses gases são os equipamentos de lavagem, separação e armazenamento da polpa, os equipamentos de lavagem e armazenamento do licor preto na planta de evaporação e os equipamentos de armazenamento de licor branco na planta de caustificação (SUHR et al., 2015). Os DNCG são constituídos, em geral e principalmente, por compostos contendo enxofre e nitrogênio. A relação entre DNCG e emissão de material particulado acontece especialmente em função das reações entre compostos de enxofre, para formação de SO_2. Este SO_2 formado se transforma em sulfato (SO_4) que substitui o carbonato (CO_3) das cinzas (VAKKILAINEN, 2005). Com isso, o consumo de álcalis volatilizados para a formação de particulados aumenta e, consequentemente, o seu teor nas emissões gasosas. Isso indica a coerência dos resultados da análise de sensibilidade.

A quinta variável, dentre aquelas apontadas como mais influentes pela análise de sensibilidade, é a vazão de óleo combustível. Uma das principais diferenças entre óleo e licor preto, em termos práticos relacionados à emissão de material particulado, relaciona-se ao fato de que o licor, além da matéria orgânica, contém componentes inorgânicos (sódio, potássio e cloro, por exemplo) provenientes da digestão da madeira, enquanto o óleo é constituído, basicamente, por hidrocarbonetos com um teor relativamente baixo de outros elementos, como nitrogênio, enxofre e oxigênio. Isso significa que, óleos combustíveis, praticamente e de modo geral, não contém metais alcalinos. Com isso, a formação de material particulado contendo sódio e potássio, por exemplo, é muito baixa. Portanto, o apontamento dado pela análise de sensibilidade acerca da influência da alimentação de óleo combustível sobre a predição da variável material particulado também condiz com o modo de operação real da caldeira.

Da mesma forma que no estudo de caso anterior, os resultados da análise de sensibilidade são importantes para uma atuação mais consciente sobre o conjunto de parâmetros manipuláveis, tendo em vista a busca por resultados mais significativos sobre a variável resposta. Em específico, o controle mais preciso e atento das variáveis levantadas pela análise de sensibilidade pode levar a um maior controle das emissões de material particulado, proporcionando assim uma operação mais estável e com menores emissões.

7 CONCLUSÕES

O desenvolvimento de sensores virtuais para aplicações industriais já é uma realidade e tende a se popularizar no setor, por oferecer soluções mais econômicas e capazes de inferir, de modo satisfatório, valores para parâmetros difíceis ou caros de se medir. Além de obter uma estimativa para variáveis, os sensores virtuais podem ser usados para avaliar o comportamento de sensores *online*, assim apurando necessidades de manutenção e calibragem de tais instrumentos.

Com o suporte de modelos baseados em dados adequados aos processos em estudo, é possível compreender em termos multivariados os parâmetros que exercem maior influência sobre a saída do modelo e, por conseguinte, sobre o processo real (caso o modelo seja capaz de descrever apropriadamente o sistema), especialmente nos casos em que o modelo é fechado, ou seja, quando sua estrutura interna não é conhecida ou não pode ser interpretada com significado físico, e de alta complexidade (como os modelos neurais). Essa análise de sensibilidade direcionada para as variáveis de maior peso sobre a resposta do sistema pode auxiliar na precisão de grupos de parâmetros que necessitam de monitoramento mais detalhado para a manutenção do controle do processo, com redução de variabilidade e de desperdícios, assim como para embasar a tomada de decisões em termos de otimização e controle de qualidade.

Para ambos os estudos de caso, partindo de variáveis de processo pré-selecionadas, características da operação de uma caldeira de recuperação química, e uma variável de qualidade, foi possível construir sensores virtuais baseados, exclusivamente, nos dados reais do processo. Inicialmente, os dados passaram por uma etapa de pré-tratamento, a fim de que apresentassem qualidade suficiente para se obter modelos mais representativos e precisos. Os sensores foram construídos a partir de rede neural artificial, a rede Perceptron em camadas múltiplas, e comparado com um caso-base de sensor desenvolvido a partir de regressão linear múltipla. Com os sensores preditivos adequados, foi possível aplicar a ferramenta de análise de sensibilidade e, assim, compreender as relações de influência das variáveis de processo sobre a variável estimada, considerando o sistema multivariado.

Considerando o primeiro estudo de caso, a Caldeira no Brasil, para a aplicação da regressão linear múltipla e para a aplicação de redes neurais artificiais, o conjunto de observações de treinamento (75% do total de dados após o pré-processamento), e o conjunto de observações para teste foram os mesmos. A partir das 13 variáveis de processo, a saber: fluxo de entrada de licor, temperatura do licor, pressão média de entrada do licor, teor de sólidos médio do licor, fluxo de ar primário, temperatura do ar primário, pressão do ar primário, fluxo de ar secundário, temperatura do ar secundário,

pressão do ar secundário, fluxo de ar terciário, temperatura do ar terciário e pressão do ar terciário; pretendeu-se predizer, em forma de regressão, as emissões de SO_2 de uma caldeira de recuperação química de uma indústria de celulose no Brasil.

O resultado da regressão linear múltipla conseguiu predizer os valores para o teor de SO_2 nos gases de combustão com um coeficiente de correlação, r, de 0,7641. Entretanto, a partir do gráfico de dispersão dos resíduos, observou-se a existência de indicação de não linearidade em tal distribuição. Essa observação era esperada, em função da complexidade do processo; a expectativa era de que modelos lineares não são suficientes para compreender o processo.

No caso da rede neural artificial, no entanto, espera-se obter melhores resultados, por ser capaz de modelar funções não lineares. Os resultados da rede neural, com função de ativação tangente hiperbólica, corroboram tal expectativa, já que para qualquer número de neurônios na camada oculta, é obtido coeficiente de correlação, r, acima de 0,8281. A rede com nove neurônios na camada oculta se apresentou como mais adequada para esta situação, demonstrando boas capacidades de predição e de generalização. Para este caso, o coeficiente de correlação para a predição do teor de SO_2 nos gases emitidos pela caldeira foi de 0,9387.

O resultado da análise de sensibilidade foi, coerente com o esperado, ao apontar que a vazão de ar terciário, a vazão do ar primário e a vazão do combustível são as variáveis que mais influenciaram a predição das emissões de dióxido de enxofre. Tais variáveis definem, especialmente, a quantidade de enxofre que entra na caldeira e a qualidade da mistura dos gases em seu interior, indicando se a combustão será ou não completa, ou seja, se o enxofre gasoso estará na forma reduzida (H_2S) ou oxidada (SO_2).

Avaliando o segundo estudo de caso, a Caldeira na Finlândia, para a aplicação da regressão linear múltipla e para a aplicação de redes neurais artificiais, o conjunto de observações de treinamento (75% do total de dados após o pré-processamento), e o conjunto de observações para teste foram os mesmos. A partir das 15 variáveis de processo, a saber: teor de sólidos no licor, fluxo de entrada de licor, fluxo de entrada de óleo combustível, pressão média de entrada do licor, temperatura do licor, fluxo de ar primário, temperatura do ar primário, pressão do ar primário, fluxo de ar secundário, temperatura do ar secundário, pressão do ar secundário, fluxo de ar terciário, temperatura do ar terciário, pressão do ar terciário, fluxo de gases não condensáveis diluídos e temperatura dos gases não condensáveis diluídos; pretendeu-se predizer, em forma de regressão, o teor de material particulado de uma caldeira de recuperação química de uma indústria de celulose na Finlândia. E, com um modelo de predição adequado, pretendeu-se avaliar as influências entre as variáveis de entrada e a variável estimada.

O resultado da regressão linear múltipla, considerando os dados de trabalho completos, conseguiu predizer os valores para o teor de material particulado nas emissões com um coeficiente de correlação,r, de 0.5273. Entretanto, a partir do gráfico da regressão y versus y estimado, observou-se que para valores maiores de particulado, o modelo produz resultado subestimado. Em função da complexidade do processo, era esperado que regressões lineares não fossem suficientes para modelar o processo.

Redes neurais artificiais do tipo MLP são capazes de modelar funções não lineares, por isso, esperam-se melhores resultados a partir da aplicação desta técnica de regressão. Os resultados da rede neural, com função de ativação tangente hiperbólica, corroboram tal expectativa, já que para qualquer número de neurônios na camada oculta, é obtido coeficiente de correlação, r, acima de 0.68. A rede com dez neurônios na camada oculta se apresentou como a mais adequada para esta situação, demonstrando boas capacidades de predição e de generalização, em termos numéricos, alcançando um coeficiente de correlação de 0.8323. Entretanto, foi observado que para valores de material particulado acima de 80 mg/Nm^3 a capacidade de predição é bastante reduzida. Desse modo, foi proposta a criação de um novo modelo considerando, a partir da base de dados de trabalho, apenas as observações com teor de material particulado igual ou menor que 80 mg/Nm^3.

Para esta base de dados reduzida, a partir da validação cruzada, obteve-se que a combinação de hiperparâmetros da rede MLP mais adequada para predição do teor de material particulado foi função de ativação tangente hiperbólica, método de otimização LBFGS, taxa de aprendizado constante e 12 neurônios na camada oculta. Após o treinamento da rede com o conjunto de identificação e validação do modelo utilizando o conjunto de teste, o coeficiente de correlação alcançado foi de 0.86. Tal resultado valida a utilização do modelo para a predição do teor de material particulado nas emissões, sendo possível seguir para uma análise de sensibilidade com grau de confiabilidade aceitável.

Por fim, o resultado da análise de sensibilidade foi coerente com o esperado, ao apontar que a vazão de ar secundário, a vazão de licor preto, a vazão de gases não condensáveis diluídos, o teor de sólidos do licor preto e a vazão de óleo combustível são as variáveis que mais influenciaram a predição das emissões de material particulado. Tais variáveis estão intimamente ligadas ao controle da temperatura do leito e da região inferior da fornalha, bem como à taxa de vaporização de álcalis, às reações que ocorrem no meio gasoso e ao tipo de fluxo que os gases da combustão podem assumir. Tais características contribuem para a formação de poeira, principalmente de sais e compostos contendo sódio e potássio, e para o carreamento de partículas sólidas e fundidas, fatores decisivos nas emissões de material particulado.

Diante do exposto, a partir da metodologia proposta, o trabalho cumpre os

objetivos de identificar e testar modelos de regressão aplicados como sensores virtuais para a predição de variáveis de interesse, a partir de variáveis de processo. Igualmente, o estudo atende o objetivo de avaliar quais os parâmetros de entrada têm maior peso, ou maior influência, sobre a variável estimada.

7.1 SUGESTÃO DE TRABALHOS FUTUROS

Seria interessante identificar os regiões de operação dentro das bases de dados e, para cada uma delas, aplicar a metodologia deste trabalho. Com isso, seria possível predizer com mais acurácia a variável de saída, bem como identificar as diferenças com relação à contribuição das variáveis de processo sobre essa variável resposta, para cada região de operação. Esse procedimento possibilitaria obter informações mais precisas sobre pontos de operação das caldeiras e as variações entre eles.

Existe uma diversidade de técnicas de análise de sensibilidade. Nessa direção, outra sugestão seria testar outras técnicas, além daquela utilizada neste trabalho, e comparar os seus resultados. Assim, poderia-se alcançar maior confiabilidade acerca dos efeitos das variáveis de entrada sobre aquela de saída.

Também poderia-se utilizar a metodologia deste trabalho para predizer outras variáveis relacionadas a emissões. Por exemplo, poderia-se desenvolver um sensor virtual para a predição de níveis de emissões de NO_x. Em seguida, poderia-se realizar uma análise de sensibilidade de modo a determinar as influências das variáveis de processo sobre as emissões de NO_x.

Além disso, a partir de um contato mais próximo do processo industrial, poderia-se propor um procedimento de coleta de dados mais específico para os objetivos traçados, com maior interação com os operadores especialistas no processo para definição das variáveis de entrada e escolha de variáveis de saída que sejam de maior interesse para a fábrica em específico.

REFERÊNCIAS

ALMEIDA, G. M. de et al. Graphical representation of cause-effect relationships among chemical process variables using a neural network approach. **International Journal of Computational Intelligence and Applications**, World Scientific, v. 9, n. 01, p. 69–86, 2010.

BAJPAI, P. **Environmentally friendly production of pulp and paper**. [S.l.]: John Wiley & Sons, 2011.

BARNETT, V.; LEWIS, T. **Outliers in statistical data**. [S.l.]: Wiley, 1974.

CHAMPAGNE, M; AMAZOUZ, M; PLATON, R. The application of soft sensors in the pulp and paper and cement manufacturing sectors for process and energy performance improvement: Opportunity analysis and technology assessment. **Canmet Energy Technology Centre, Tech. Rep**, 2005.

CONAMA, Conselho Nacional do Meio Ambiente; MMA, Ministério do Meio Ambiente Brasil. **Resolução n°436, de 22 de Dezembro de 2011**. [S.l.], 2011. Disponível em: <http://www2.mma.gov.br/port/conama/res/res11/res43611.pdf>.

DONG, D.; MCAVOY, T. J.; CHANG, L. J. Emission monitoring using multivariate soft sensors. In: IEEE. PROCEEDINGS of 1995 American Control Conference-ACC'95. [S.l.: s.n.], 1995. v. 1, p. 761–765.

EVERITT, B.; SKRONDAL, A. **The Cambridge dictionary of statistics**. [S.l.]: Cambridge University Press Cambridge, 2010.

FAOSTAT, Food Agriculture Organization. **UN data: A world of information**. 2019. Disponível em: <http://data.un.org/Search.aspx?q=pulp>.

GE, Z.; SONG, Z.; GAO, F. Review of recent research on data-based process monitoring. **Industrial & Engineering Chemistry Research**, ACS Publications, v. 52, n. 10, p. 3543–3562, 2013.

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

GEVREY, M.; DIMOPOULOS, I.; LEK, S. Review and comparison of methods to study the contribution of variables in artificial neural network models. **Ecological modelling**, Elsevier, v. 160, n. 3, p. 249–264, 2003.

HAMPEL, F. R. The breakdown points of the mean combined with some rejection rules. **Technometrics**, Taylor & Francis, v. 27, n. 2, p. 95–107, 1985.

HAMPEL, F. R. The influence curve and its role in robust estimation. **Journal of the american statistical association**, Taylor & Francis, v. 69, n. 346, p. 383–393, 1974.

HAN, J.; PEI, J.; KAMBER, M. **Data mining: concepts and techniques**. [S.l.]: Elsevier, 2011.

ILIYAS, S. A. et al. RBF neural network inferential sensor for process emission monitoring. **Control Engineering Practice**, Elsevier, v. 21, n. 7, p. 962–970, 2013.

KLUYVER, T. et al. Jupyter Notebooks-a publishing format for reproducible computational workflows. In: ELPUB. [S.l.: s.n.], 2016. p. 87–90.

KOTSIANTIS, S. B.; KANELLOPOULOS, D.; PINTELAS, P. E. Data preprocessing for supervised learning. **International Journal of Computer Science**, Citeseer, v. 1, n. 2, p. 111–117, 2006.

KRIEGESKORTE, N. Crossvalidation. Edição: Arthur W. Toga. Academic Press, Waltham, p. 635–639, 2015. DOI: https://doi.org/10.1016/B978-0-12-397025-1.00344-4. Disponível em: <http://www.sciencedirect.com/science/article/pii/B9780123970251003444>.

LUGADE, V. **Neural Networks – A Multilayer Perceptron in Matlab**. 2011. Disponível em: <https://matlabgeeks.com/tips-tutorials/neural-networks-a-multilayer-perceptron-in-matlab/>.

MATHWORKS. **Hampel Filter**. 2017. Disponível em: <https://uk.mathworks.com/help/dsp/ref/hampelfilter.html>.

MONTGOMERY, D. C.; RUNGER, G. C. **Applied statistics and probability for engineers**. [S.l.]: John Wiley e Sons, 2014.

PAPADOKONSTANTAKIS, S.; LYGEROS, A.; JACOBSSON, S. P. Comparison of recent methods for inference of variable influence in neural networks. **Neural Networks**, Elsevier, v. 19, n. 4, p. 500–513, 2006.

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

QIN, S. J.; YUE, H.; DUNIA, R. Self-validating inferential sensors with application to air emission monitoring. **Industrial & engineering chemistry research**, ACS Publications, v. 36, n. 5, p. 1675–1685, 1997.

RAMAKALYAN, A. et al. Development of KSVGRNN: A hybrid soft computing technique for estimation of boiler flue gas components. **Journal of Industrial Information Integration**, Elsevier, v. 4, p. 42–51, 2016.

ROSSUM, G van. Python tutorial, technical report CS-R9526. **Centrum voor Wiskunde en Informatica (CWI), Amsterdam**, 1995.

ROUSSEEUW, P. J.; CROUX, C. Alternatives to the median absolute deviation. **Journal of the American Statistical association**, Taylor & Francis, v. 88, n. 424, p. 1273–1283, 1993.

SÄRKKÄ, T.; GUTIÉRREZ-POCH, M.; KUHLBERG, M. Technological Transformation in the Global Pulp and Paper Industry: Introduction. Springer, p. 1–10, 2018.

SCARDI, M.; HARDING JR, L. W. Developing an empirical model of phytoplankton primary production: a neural network case study. **Ecological modelling**, Elsevier, v. 120, n. 2-3, p. 213–223, 1999.

SHAKIL, M. et al. Soft sensor for NOx and O2 using dynamic neural networks. **Computers & Electrical Engineering**, Elsevier, v. 35, n. 4, p. 578–586, 2009.

SILVA, I. N. da; SPATTI, D. H.; FLAUZINO, R. A. Redes neurais artificiais para engenharia e ciências aplicadas. **São Paulo: Artliber**, v. 23, n. 5, 2010.

SOUZA, F. A. A.; ARAÚJO, R.; MENDES, J. Review of soft sensor methods for regression applications. **Chemometrics and Intelligent Laboratory Systems**, Elsevier, v. 152, p. 69–79, 2016.

SUHR, M. et al. Best available techniques (BAT) reference document for the production of pulp, paper and board. **European Commission**, 2015.

TRAN, H.; VAKKILAINEN, E. K. The kraft chemical recovery process. **Tappi Kraft Pulping Short Course**, p. 1–8, 2008. Disponível em: <https://www.researchgate.net/publication/267565045_THE_KRAFT_CHEMCIAL_RECOVERY_PROCESS>.

TRONCI, S.; BARATTI, R.; SERVIDA, A. Monitoring pollutant emissions in a 4.8 MW power plant through neural network. **Neurocomputing**, Elsevier, v. 43, n. 1-4, p. 3–15, 2002.

UNEP, United Nations Environment Programme Division of Technology Industry Economics. **Resource Efficient and Cleaner Production**. 1990. Disponível em: <http://www.unep.fr/scp/cp/>.

URONEN, P. Trends in digital control applications in pulp and paper industry. In: REAL Time Digital Control Application. [S.l.]: Elsevier, 1984. p. 309–314.

VAKKILAINEN, E. K. Kraft recovery boilers–Principles and practice. Valopaino Oy, 2005.

_____. **Steam generation from biomass: construction and design of large boilers**. [S.l.]: Butterworth-Heinemann, 2016.

VALMET. **Optimized recovery boiler**. 2018. Disponível em: <https://www.valmet.com/pulp/chemical-recovery/recovery-boilers/>. Acesso em:

WANG, X. Z.; MCGREAVY, C. Data Mining and Knowledge Discovery for Process Monitoring and Control. Springer-Verlag, 1999.

WBC, World Bank Group; WHO, World Health Organization. **Pollution Prevention and Abatement Handbook, 1998: Toward Cleaner Production**. [S.l.]: World Bank Publications, 1999.

YEH, I-C.; CHENG, WL. First and second order sensitivity analysis of MLP. **Neurocomputing**, Elsevier, v. 73, n. 10-12, p. 2225–2233, 2010.

ZHOU, H.; CEN, K.; FAN, J. Modeling and optimization of the NOx emission characteristics of a tangentially fired boiler with artificial neural networks. **Energy**, Elsevier, v. 29, n. 1, p. 167–183, 2004.

APÊNDICES

APÊNDICE A – CÓDIGO UTILIZADO

A versão completa do código implementado na resolução do problema pode ser encontrada na plataforma *GitHub*, em formato de um código livre. O arquivo pode ser encontrado sob o título `CódigoPython_MLP_AnaliseSensibilidade.ipynb`, ou diretamente no endereço

`https://github.com/ABBelisario/mestrado/blob/master/C%C3%B3digoPython_MLP_AnaliseSensibilidade.ipynb`

Segue-se uma versão resumida, apenas com os tópicos considerados mais relevantes.

Python_MLP_AnaliseSensibilidade_Resumido

February 13, 2020

1 1. Organização prévia

1.1 1.1 Módulos que serão utilizados

1.2 1.2 Funções implementadas

[ ]:
```
#IMPORT MODULES
import os
import scipy
import numpy as np
import pandas as pd
import sklearn as sk
from sklearn.model_selection import train_test_split
from sklearn.neural_network import MLPRegressor
from sklearn.preprocessing import MinMaxScaler
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
import matplotlib.pyplot as plt
import operator
import xlwt
from xlwt.Workbook import *
from pandas import ExcelWriter
import xlsxwriter
import math as m
import matplotlib.cm as cm
from pandas.plotting import scatter_matrix
import itertools
from matplotlib import patches
```

[ ]:
```
#Function 1: Obtaining the DataBase
def database(path, file):
    os.chdir(path) #Change directory
    db=pd.read_csv(file,sep=",",encoding="ISO-8859-1") #Data set
    [l,c]=db.shape #Data Matrix shape
    header=db.dtypes.index #Columns Header (Variables)
    return (db,l,c,header)
#Function 2.0: Three-Sigma Rule
def nsigma(var_orig, n=3):
```

```
    '''n -> multiplication factor of the threshold'''
    var=var_orig.copy()
    mean=var.mean(0)
    std=var.std(0)
    difference=abs(var-mean)
    threshold = n*std
    outlier_idx=difference>threshold
    var[outlier_idx]=np.nan
    return (var,outlier_idx,threshold)
#Function 2.1: Hampel Filter - moving window
def hampel(var_orig, rol='y', k=7, t0=3):
    '''vals: pandas series of values from which to remove outliers
    k: size of window (including the sample; 7 is equal to 3 on either side of␣
 ↪value)'''
    var=var_orig.copy() #Make copy so original not edited
    L= 1.4826 #Hampel Filter
    if rol=='y':
        rolling_median=var.rolling(k).median()#Median calculatade for each␣
 ↪window, correspondent to each row -> the first (k-1) lines are set to NaN
        difference=np.abs(rolling_median-var)# The k first are valued as NaN
        median_abs_deviation=difference.rolling(k).median()
    else:
        median=var.median()#original Hampel
        difference=np.abs(median-var)
        median_abs_deviation= difference.median()
    threshold= t0 *L * median_abs_deviation
    outlier_idx=difference>threshold # NaN compered to anything always returns␣
 ↪False -> the first k-1 values are never considered outlier
    var[outlier_idx]=np.nan
    return(var,outlier_idx,threshold)
#Function 3: Data Normalization
def scale(train,test,sk=1,rang=(0,1)):
    if sk==1:#sk=1 -> MinMaxScaler
        scaler=MinMaxScaler(feature_range=rang)
    else: # sk=2 -> StandardScaler
        scaler=StandardScaler()
    if len(train.shape)==1:
        tr=train.values.reshape(-1,1)
        te=test.values.reshape(-1,1)
        tr=np.reshape(scaler.fit_transform(tr),len(train))
        te=np.reshape(scaler.transform(te),len(test))
    else:
        tr=scaler.fit_transform(train)
        te=scaler.transform(test)
    return (tr,te,scaler)
#Function 4: Separate Database into input variables and output variables
def inout(data,header,output_number=1):
```

```
    yname=header[-output_number]#Output variables
    y=data[yname]
    xname=header[:-output_number]#Input variables
    x=data[xname]
    xs=x.shape
    ys=y.shape
    return(x,xname,xs,y,yname,ys)
```

2 2. Leitura da base de dados

2.1 2.1 Base de dados por completo

2.2 2.2 Base de dados apenas com as variáveis que serão efetivamente utilizadas

[ ]:
```
#OBTAIN THE DATABASE
path="C:/Users/anab-/Documents/EQ Mestrado/A Pesquisa/Dados e Resultados/
 ↪Caldeira de Recuperação"
file="db_20vc.csv"
db_orig,l,c,header_orig=database(path,file)
#The first and second columns are index and month -> should be removed
header=header_orig[2::]
dbb=db_orig.loc[:,header]
#Descriptive statistics
stat=dbb.describe()
```

[ ]:
```
# CREATE A DIRECTORY TO SEND THE RESULTS
db=dbb.drop(['16-P_BD_KG/CM2'],axis=1)
dir = 'Results_new_outPBD'
if not os.path.isdir(dir): os.makedirs(dir)
figsize=(12,8)
plt.rcParams.update({'font.size': 12})
db.describe()
```

[ ]:
```
#EXCLUDE NOT USED VARIABLES
db0=db.drop(['17-C_H2S_PPM', '19-F_STREAM_T/H', '20-E_REDUC_%'],axis=1).
 ↪replace(0,np.nan).dropna()
header0=db0.dtypes.index
#Data base after deletion of the SO2 disparate data
db01=db0.copy() #Unly the input variable included in the model
db01=db01.iloc[0:2395].append(db01.iloc[2545::])
#CORRELATION MATRIX
cor=db0.corr() #Correlation Matrix
cor.style.background_gradient(cmap='PuBu')
```

3 3. PRÉ PROCESSAMENTO

3.1 3.1 Cálculo das médias das variáveis e verificação da coerência

3.2 3.2 Detecção de outlier

[ ]:
```
# 3.1 SUBSTITUTION OF '3-P_FUEL-MMH20' and '4-P_FUEL-MMH20' BY THE AVERAGE
pfuel=db01.loc[:,('3-P_FUEL-MMH20','4-P_FUEL-MMH20')].copy()
#Mean of the values
meanfuel=(pfuel['3-P_FUEL-MMH20']+pfuel['4-P_FUEL-MMH20'])/2
pfuel['3.5-P_FUELmean-MMH20']=meanfuel
#SCATTER MATRIX
scatter_matrix(pfuel, alpha=0.2, figsize=(8,8), diagonal='hist')
fname = 'Scatter-matrix p fuel.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
#Correlation Matrix
corfuel=pfuel.corr()
corfuel.style.background_gradient(cmap='PuBu')
```

[ ]:
```
# 3.1 SUBSTITUTION OF '5-SS1_%' and '6-SS2_%' BY THE AVERAGE
ssolid=db01.loc[:,('5-SS1_%', '6-SS2_%')].copy()
#Mean of the values
meansolid=(ssolid['5-SS1_%']+ssolid['6-SS2_%'])/2
ssolid['5.5-SSmean_%']=meansolid
#SCATTER MATRIX
scatter_matrix(ssolid, alpha=0.2, figsize=(8, 8), diagonal='hist')
fname = 'Scatter-matrix solids.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
#Correlation Matrix
corsolid=ssolid.corr()
corsolid.style.background_gradient(cmap='PuBu')
```

[ ]:
```
## 3.1 SUBSTITUTION OF THE TWO PRESSURES AND TWO SOLIDS CONTENTS MEASUREMENTS␣
 ↪BY THEIR MEANS
db1 = db01.copy()
# Unification of '3-P_FUEL-MMH20' and '4-P_FUEL-MMH20' - Mean of pressures
Fm=(db1['3-P_FUEL-MMH20']+db1['4-P_FUEL-MMH20'])/2
#Modification of the database
db1['3-P_FUEL-MMH20']=Fm
db1.drop(['4-P_FUEL-MMH20'], axis=1, inplace=True)# Removing columns
db1.rename(columns={'3-P_FUEL-MMH20': '3.5-P_FUELmean-MMH20'}, inplace=True)
# Unification of '5-SS1_%' and '6-SS2_%' - Mean Solids Content
SSm=(db1['5-SS1_%']+db1['6-SS2_%'])/2
#Modification of the database
db1['5-SS1_%']=SSm #Adding the %SS mean values to the new database
db1.drop(['6-SS2_%'], axis=1, inplace=True)# Removing columns
```

```
db1.rename(columns={'5-SS1_%': '5.5-SSmean_%'}, inplace=True)
head1= db1.dtypes.index
db1.shape
```

[ ]:
```
# 3.2 OUTLIER DETECTION
workfilter=db1.copy()
#Filter
db_f2,out_idxf2,threshold2 = hampel(workfilter,rol='n') #Just 1.4826*3*MAD
db_f2=db_f2.dropna()
# 3.2 Chosen Filter -> Hampel identifier
work=db_f2.copy()
head=work.dtypes.index
print(np.sum(out_idxf1),db_f1.shape, np.sum(out_idxf2),db_f2.shape, np.
 ↪sum(out_idxf3),db_f3.shape)
```

[ ]:
```
#CORRELATION MATRIX
corw=work.corr() #Correlation Matrix
corw.style.background_gradient(cmap='PuBu')#'coolwarm')
```

[ ]:
```
# 3.2 Univariate plots
for i in range(db1.shape[1]):
    plt.figure(figsize=(12,8))
    plt.plot(db1.iloc[:,i],'k--',alpha=0.5,label='Dados antes filtro')#Data␣
 ↪before filtering
    plt.plot(work.iloc[:,i],'k-',alpha=1, label='Dados após filtro') #Data␣
 ↪after filtering
    plt.legend()
    if i==db1.shape[1]-1:
        plt.ylim((25,250))
        plt.legend(loc='upper left')
    plt.xlabel('Observation')
    plt.ylabel(head1[i])
    plt.title(head1[i])
    fname = 'AfterFilter-' + str(i+1) + '.png'
    plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
```

4 4. Validação cruzada e Normalização

4.1 4.1 Validação cruzada

4.2 4.2 Normalização

4.3 4.3 Verificação dos resultados

[ ]:
```
# 4.1 Separation of the rows with the highest and te lowest values of each␣
 ↪variable, in order to ensure that they will be assigned to training set
kn=5
rs=42
#DATABASE SELECTED:
data=work.copy()
head=work.dtypes.index
#Separation of the data: Input and Output
x,xname,xs,y,yname,ys=inout(data,head,output_number=1) #xs,ys = shapes of x and␣
 ↪y
ind=[]
for i in range(x.shape[1]):
    ind.append(x[xname[i]].idxmax())
    ind.append(x[xname[i]].idxmin())
ind.append(y.idxmax())
ind.append(y.idxmin())
xminmax=x.loc[set(ind)]
xrest=x.drop(set(ind))
yminmax=y.loc[set(ind)]
yrest=y.drop(set(ind))
#Separation of the data into training set and testing set
ts= 0.2535 #0.2005 #Testar 0.2535
xtr, xte, ytr, yte=train_test_split(xrest, yrest, test_size=ts, random_state=rs)
xtr=xtr.append(xminmax)
ytr=ytr.append(yminmax)
xtr.shape,xtr.shape[0]/5, xte.shape[0]/data.shape[0], xte.shape[0]
```

[ ]:
```
# 4.2 Normalização
#MinMaxScaler
        #For tanh: x and y => [-1,+1]
        #For logistic: x=>[-1,+1]; y=>[0,+1]
xtrain,xtest,scalerx=scale(xtr,xte,sk=1,rang=(-1,1))
ytrain,ytest,scalery=scale(ytr,yte,sk=1,rang=(-1,1))
ytrainl,ytestl,scaleryl=scale(ytr,yte,sk=1,rang=(0,1))
    #StandardScaler
xtrain1,xtest1,scalerx1=scale(xtr,xte,sk=2)
ytrain1,ytest1,scalery1=scale(ytr,yte,sk=2) #StandardScaler does not need a␣
 ↪range definition
ytrai=ytrain.copy()
ytes=ytest.copy()
```

[ ]:
```
# 4.3 Boxplot
medianprops = dict(linestyle=':', linewidth=2, color='k')
for j in range(x.shape[1]):
    plt.figure(figsize=(8,8))
    plt.boxplot([xtrain[:,j],xtest[:,j]],labels=['xtrain','xtest'],␣
 ↪medianprops=medianprops)
    plt.title(head[j])
    fname = 'Boxplot-' + str(j+1) + '.png'
    plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
plt.figure(figsize=(8,8))
plt.boxplot([ytrain,ytest],labels=['ytrain','ytest'], medianprops=medianprops)
plt.title(head[-1])
fname = 'Boxplot-' + str(15) + '.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
```

5 5. Modelos de Regressão

5.1 5.1 Regressão Linear Múltipla

[ ]:
```
# 5.1 Results from Multiple linear regression
n=0
kn=5 #number of splits at cross-validation procedure
#Writing the answer
column=['Y test','ŷ-cv1','ŷ-cv2',␣
 ↪'ŷ-cv3','ŷ-cv4','ŷ-cv5','ŷ-all','r2-cv1','r2-cv2','r2-cv3','r2-cv4','r2-cv5','r2-cvmean','r
mlr=pd.DataFrame(index=yte.index,columns=column).fillna(0)
mlr.iloc[:,0]=yte #Y value from database
# MULTIPLE LINEAR REGRESSION
reg=sk.linear_model.LinearRegression(fit_intercept=True, normalize=False,␣
 ↪copy_X=True, n_jobs=None)
R2=np.zeros([1,kn])
#Cross-Validation
kf=sk.model_selection.KFold(n_splits=kn, shuffle=True, random_state=rs)
    #The training data is used for cross validation procedure. Te first testg␣
 ↪set separeted is used after, for testing (https://towardsdatascience.com/
 ↪train-test-split-and-cross-validation-in-python-80b61beca4b6)
    #Data set for cross validation: xtrain, ytrain
i=0
for train_index, test_index in kf.split(xtrain,ytrain):
    x_train, x_test = xtrain[train_index], xtrain[test_index]
    y_train, y_test = ytrain[train_index], ytrain[test_index]
    #Treining
    freg=reg.fit(x_train,y_train)
    parr=freg.get_params(deep=True)
```

```
    #Prediction
    y_hat=freg.predict(x_test) #Cross-validation prediction
    yhat=freg.predict(xtest) #Test Prediction
    #Return to original scale
    Y_hat=scalery.inverse_transform(y_hat.reshape(-1,1))
    Yhat=scalery.inverse_transform(yhat.reshape(-1,1))
    #Coefficient of Determination
    r2=freg.score(x_test,y_test) # Cross validation test set
    R2test=freg.score(xtest,ytest) #Real test set
    mlr.iloc[:,i+1]=Yhat #Crossvalidation
    mlr.iloc[:,i+7]=r2#R2test
    R2[0,i]=r2
    i+=1
#Treinar o modelo com todos os dados
freg=reg.fit(xtrain,ytrain)
parr=freg.get_params(deep=True)
#Prediction
yhat=freg.predict(xtest) #Test Prediction
Yhat=scalery.inverse_transform(yhat.reshape(-1,1)) #Return to original scale
R2test=freg.score(xtest,ytest) #Real test set
mlr.iloc[:,6]=Yhat
mlr.iloc[:,-2]=mlr.iloc[:,[7,8,9,10,11]].mean(axis=1)
mlr.iloc[:,-1]=R2test
stdev=mlr.iloc[:,[7,8,9,10,11]].std(axis=1).iloc[0]
# MSE: Mean Squared Error
mse_mlr=sum((mlr.iloc[:,0]-mlr.iloc[:,6])*(mlr.iloc[:,0]-mlr.iloc[:,6]))/mlr.
 ↪iloc[:,0].shape[0]
```

[ ]:
```
# 5.1 Results of MLR
import math as m
print(" r2_cv mean: ",mlr.iloc[0,-2],"\n","r2_complete data: ", mlr.iloc[0,-1])
print(" r_cv mean: ",m.sqrt(mlr.iloc[0,-2]),"\n","r_complete data: ", m.
 ↪sqrt(mlr.iloc[0,-1]))
print(" residuals mean and stdev: ", (mlr.iloc[:,0]-mlr.iloc[:,6]).mean(0),(mlr.
 ↪iloc[:,0]-mlr.iloc[:,6]).std(0))
print(" Maximum and Minimum error: ", abs(mlr.iloc[:,0]-mlr.iloc[:,6]).
 ↪max(),abs(mlr.iloc[:,0]-mlr.iloc[:,6]).min())
print(" Mean absolute error: ",sum(abs(mlr.iloc[:,0]-mlr.iloc[:,6]))/mlr.iloc[:
 ↪,6].shape[0])
#Coeficientes da regressão linear
print(freg.coef_ , freg.intercept_)
print(parr)
# 5.1 Plot results from multiple linear regression
plt.figure(3,figsize=(8,8))
plt.plot([75,115],[75,115],'-.', color='grey')
plt.plot(mlr.iloc[:,0],mlr.iloc[:,6],'k.') #r2 cv mean
```

```
plt.text(76,112,'R2 test: '+str(round(mlr.iloc[0,-1],4))+'\n'+'R test:␣
 ↪'+str(round(np.sqrt(mlr.iloc[0,-1]),4)), fontsize=12)
plt.title('Multiple Linear Regression Avaliation: ŷ vs y')
plt.ylim((75,115))
plt.ylabel('ŷ (SO2 estimatives)')
plt.xlim((75,115))
plt.xlabel('ytest (SO2 real values)')
fname = 'MLR_ŷ-y' + '.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)

plt.figure(4,figsize=(8,8))
plt.hist(sorted(mlr.iloc[:,0]-mlr.iloc[:,6]),50, density=True, color='gray')
plt.xlabel('y-ŷ')
plt.ylabel('Probability density')
plt.title('Histogram of residuals - multiple linear regression'+'\n'+'Absolute␣
 ↪Values')
plt.text(-16,0.085,'Residuals mean: '+str(round((mlr.iloc[:,0]-mlr.iloc[:,6]).
 ↪mean(0),4))+'\n'+'Residuals standard deviation: '+str(round((mlr.iloc[:
 ↪,0]-mlr.iloc[:,6]).std(0),4)), fontsize=10)
fname = 'MLR-Histogram of residuals-absolute' + '.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)

plt.figure(5,figsize=(8,8))
plt.plot([0,2600],[0,0],'-.', color='grey')
plt.plot((mlr.iloc[:,0]-mlr.iloc[:,6]),'k.')
plt.ylabel('y-ŷ')
plt.xlabel('Observation')
plt.title('Dispersion of the residuals - multiple linear␣
 ↪regression'+'\n'+'Absolute Values')
fname = 'MLR-Dispersion of the residuals MLR-absolute' + '.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
```

5.2 5.2 Rede Perceptron em multicamadas

[ ]:
```
## 5.2 Multilayer Perceptron
#Number of hidden layers variation
N=30 #maximal number of neurons at hidden layer
kn=5 #number of splits at cross-validation procedure
t=1e-3 #Tolerance (modifications at r2 - > training the net)
niter=7
#Neural Network parameters
n_neurons=list(range(1,N+1))
activs=['logistic', 'tanh']
```

```
solvs = ['lbfgs', 'sgd']
learn_rate= ['constant', 'invscaling', 'adaptive']
param_test=[] #MLP - Parameter names mapped to their values. For each test
#Writing the answer
cvn=ytr.shape[0]/5
ind1=np.arange(cvn)
ind2=np.array(yte.index)
column1=['ŷ-cv1','ŷ-cv2',␣
 ↪'ŷ-cv3','ŷ-cv4','ŷ-cv5','r2-cv1','r2-cv2','r2-cv3','r2-cv4','r2-cv5','r2_mean','r2_stdev']
column2=['Y_test','ŷ-test','r2-all','r2-cv1','r2-cv2','r2-cv3','r2-cv4','r2-cv5','r2_mean']
column3=['r-cv1','r-cv2','r-cv3','r-cv4','r-cv5','r_mean','r_stdev']
idx = pd.IndexSlice
index1= pd.MultiIndex.from_product([n_neurons, activs, solvs, learn_rate,ind1],
                                   ␣
 ↪names=['ActivFunc','Solver','LearnRate','NumberNeurons','Observation'])
index2= pd.MultiIndex.from_product([n_neurons, activs, solvs, learn_rate, ind2],
                                   ␣
 ↪names=['ActivFunc','Solver','LearnRate','NumberNeurons','Observation'])
mlp_cv=pd.DataFrame(index=index1,columns=column1)#Results from crossvalidation␣
 ↪procedure
mlp_test=pd.DataFrame(index=index2,columns=column2)#Results all training set␣
 ↪and test set (reidentificação da rede)
r_coef=pd.DataFrame(index=index1,columns=column3)#Results Correlation␣
 ↪coefficient(r), crossvalidation
for m in range(N):#N #Neurons number
    m=m+1
    for h in range(2):#2 #activation
        if activs[h]=='logistic':
            ytrai=ytrainl.copy()
            ytes=ytestl.copy()
        else:
            ytrai=ytrain.copy()
            ytes=ytest.copy()
        for j in range(2):#2 #solver
            for k in range(3):#3 #learning rate
                nn=MLPRegressor(hidden_layer_sizes=(m+1,),
                                activation=activs[h],
                                solver=solvs[j],
                                batch_size='auto',
                                learning_rate= learn_rate[k],
                                max_iter=1000,
                                random_state=rs,
                                tol=t,
                                n_iter_no_change=niter,
                                verbose=False)
```

```
                #https://scikit-learn.org/stable/modules/generated/sklearn.
↪model_selection.KFold.html
                kf=sk.model_selection.KFold(n_splits=kn, shuffle=True,␣
↪random_state=rs)
                #The training data is used for cross validation procedure. The␣
↪first testset sorted is used after, for testing (https://towardsdatascience.
↪com/train-test-split-and-cross-validation-in-python-80b61beca4b6)
                #Data set for cross validation: xtrain, ytrain
                i=0
                for train_index, test_index in kf.split(xtrain,ytrai):
                    x_train, x_test = xtrain[train_index], xtrain[test_index]
                    y_train, y_test = ytrai[train_index], ytrai[test_index]

                    #Training
                    rb=nn.fit(x_train,y_train)
                    par=rb.get_params(deep=True)
                    #Prediction
                    y_hat=nn.predict(x_test)
                    #Test with the extra testing set - For each cross validation
                    yhat=nn.predict(xtest)
                    #Reshaping
                    y_h=y_hat.reshape(-1,1)
                    yh=yhat.reshape(-1,1)
                    if activs[h]=='logistic' or activs[h]=='relu':
                        Y_hat=scaleryl.inverse_transform(y_h)
                        Yhat=scaleryl.inverse_transform(yh)
                        y_t=scaleryl.inverse_transform(y_test.reshape(-1,1))
                    else:
                        Y_hat=scalery.inverse_transform(y_h)
                        Yhat=scalery.inverse_transform(yh)
                        y_t=scalery.inverse_transform(y_test.reshape(-1,1))
                    r2=rb.score(x_test,y_test)
                    R2=rb.score(xtest,ytes)

                    mlp_cv.loc[idx[m,activs[h], solvs[j],learn_rate[k],:
↪],column1[i]]=Y_hat
                    mlp_cv.loc[idx[m,activs[h], solvs[j],learn_rate[k],:
↪],column1[i+5]]=r2
                    r_coef.loc[idx[m,activs[h], solvs[j],learn_rate[k],:
↪],column3[i]]=np.sqrt(r2)
                    mlp_test.loc[idx[m,activs[h], solvs[j],learn_rate[k],:
↪],column2[i+3]]=R2
                    i+=1
                #Training with all data
                rb=nn.fit(xtrain,ytrai)
                #Test with the extra testing set - For each cross validation
```

```
                yhat=nn.predict(xtest)
                yh=yhat.reshape(-1,1)
                R2=rb.score(xtest,ytes)
                if activs[h]=='logistic':
                    Yhat=scaleryl.inverse_transform(yh)
                else:
                    Yhat=scalery.inverse_transform(yh)
                #Writing the answer
                mlp_test.loc[idx[m,activs[h], solvs[j],learn_rate[k],:
↪],column2[0]]=yte.values.reshape(-1,1)
                mlp_test.loc[idx[m,activs[h], solvs[j],learn_rate[k],:
↪],column2[1]]=Yhat
                mlp_test.loc[idx[m,activs[h], solvs[j],learn_rate[k],:
↪],column2[2]]=R2
                param_test.append([m,activs[h],solvs[j],learn_rate[k],nn.
↪get_params()])
```

[ ]:
```
# 5.2 MEAN OF R2 FROM CROSSVALIDATION (Determination coefficient)
mlp_cv.loc[idx[:],column1[-2]]=mlp_cv.iloc[:,[5,6,7,8,9]].mean(axis=1)
mlp_cv.loc[idx[:],column1[-1]]=mlp_cv.iloc[:,[5,6,7,8,9]].std(axis=1)
mlp_test.loc[idx[:],column2[-1]]=mlp_test.iloc[:,[2,3,4,5,6]].mean(axis=1)
# 5.2 MEAN OF R FROM CROSSVALIDATION (correlation coefficient)
r_coef.loc[idx[:],column3[-2]]=r_coef.iloc[:,[0,1,2,3,4]].mean(axis=1)
r_coef.loc[idx[:],column3[-1]]=r_coef.iloc[:,[0,1,2,3,4]].std(axis=1)
```

[ ]:
```
##CROSS VALIDATION
# 5.2 Take the better results per activation function for each neuron number at␣
↪hidden layer
best_log=np.zeros((N,6)).astype(object)
best_tanh=np.zeros((N,6)).astype(object)
for i in range(N):
    best_log[i,:5]=r_coef.loc[idx[i+1,'logistic',:,:,:]].
↪sort_values(['r_mean'], ascending=[False]).iloc[0].name
    best_log[i,-2]=r_coef.loc[idx[i+1,'logistic',:,:,:]].
↪sort_values(['r_mean'], ascending=[False]).iloc[0,-2] #r_mean
    best_log[i,-1]=r_coef.loc[idx[i+1,'logistic',:,:,:]].
↪sort_values(['r_mean'], ascending=[False]).iloc[0,-1] #r_stdev
    best_tanh[i,:5]=r_coef.loc[idx[i+1,'tanh',:,:,:]].sort_values(['r_mean'],␣
↪ascending=[False]).iloc[0].name
    best_tanh[i,-2]=r_coef.loc[idx[i+1,'tanh',:,:,:]].sort_values(['r_mean'],␣
↪ascending=[False]).iloc[0,-2] #r_mean
    best_tanh[i,-1]=r_coef.loc[idx[i+1,'tanh',:,:,:]].sort_values(['r_mean'],␣
↪ascending=[False]).iloc[0,-1] #r_stdev
best_log, best_tanh
```

```
[ ]: # 5.2 Plot restrictly tanh, lbfgs, constant - Cross validation
plt.figure(1,figsize)
for i in range(1,N+1):
    plt.errorbar(i,r_coef.loc[idx[i,'tanh','lbfgs','constant']].
 ↪iloc[0,-2],yerr=r_coef.loc[idx[i,'tanh','lbfgs','constant']].
 ↪iloc[0,-1],elinewidth=0.5, ecolor='black')#Mean of R2 of the crossvalidation␣
 ↪regression
    plt.plot(i,r_coef.loc[idx[i,'tanh','lbfgs','constant']].iloc[0,-2],'k.')
plt.text(8.,0.938,9)
plt.arrow(8.5,0.938,0.23,-0.007)
plt.text(11.,0.94,12)
plt.arrow(11.5,0.938,0.23,-0.007)
plt.plot(xaxis,best_log[:,-2],'k.',label='logistic')
plt.errorbar(xaxis, best_log[:,-2],best_log[:,-1],linewidth=0,elinewidth=0.5,␣
 ↪color='grey')
plt.title('Mean and standard deviation: r mean of cross-validation (actv␣
 ↪function: tanh)')
plt.ylabel('r mean')
plt.xlabel('Number of neurons at hidden layer')
fname = 'r mean of cross-validation x nn_logtanh.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
# Performance improvement provided by addition of units at hidden layer
bla=[]
xs = np.arange(2,N+1,1)
fig = plt.figure(2,figsize)
ax = fig.add_subplot(111)
for i in range(N-1):
    dif=(best_tanh[i+1,-2]-best_tanh[i,-2])/best_tanh[i,-2]#Relative increase
    bla.append(100*dif)
    plt.bar(i+2,100*dif, color='gray')
    plt.xticks(np.arange(2, 31, 1))
e1 = patches.Ellipse(xy=(9,1.15), width=1.4,height=0.45, fill=False)
ax.add_patch(e1)
e = patches.Ellipse(xy=(12,1.3), width=1.4,height=0.45, fill=False)
ax.add_patch(e)
ax.set_title('Performance increase by adding units at hidden layer')
ax.set_ylabel('Percentage increase at r mean')
ax.set_xlabel('Comparisons of unit_(i) and unit_(i-1)')
for x,y in zip(xs,bla):
    if y < 0:
        space = -10
    else:
        space = 8
    label = "{:.2f}".format(y)
    plt.annotate(label, # this is the text
                 (x,y), # this is the point to label
```

```
                textcoords="offset points", # how to position the text
                xytext=(0,space), # distance from text to points (x,y)
                ha='center') # horizontal alignment can be left, right or␣
 ↪center
fname = 'Percentage increase at r mean-unit-unit' + '.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
```

[ ]:
```
best_test=pd.DataFrame(columns=mlp_test.columns)
for i in range(1,N+1):
    best_test.loc[i] = mlp_test.loc[idx[i,'tanh','lbfgs','constant']].iloc[0,:]
best_test['r2_stdev']=best_test.iloc[:,3:8].std(axis=1)
# 5.2 Plot restrictly tanh, lbfgs, constant - Cross validation
plt.figure(1,figsize)
for i in range(1,N+1):
    plt.errorbar(i,best_test.iloc[i-1,-2],yerr=best_test.
 ↪iloc[i-1,-1],elinewidth=0.5, ecolor='black')#Mean of R2 of the␣
 ↪crossvalidation regression
    plt.plot(i,best_test.iloc[i-1,-2],'k.')
    if i==9:
        plt.plot(i,best_test.iloc[i-1,-2],'rx')
plt.title('Mean and standard deviation: r2 mean of test set (actv function:␣
 ↪tanh)')
plt.ylabel('r2 mean')
plt.xlabel('Number of neurons at hidden layer')
fname = 'r2 mean of test set x nn' + '.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
```

[ ]:
```
neurons=9
# 5.2 Plot y vs ŷ -> TEST SET
plt.figure(1,figsize=(8,8))
plt.plot([73,120],[73,120],'-.', color='grey')
plt.plot(mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:
 ↪,0],mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,1],'k.')
plt.xlabel('Y_test (SO2 real values)')
plt.ylabel('Ŷ_test (SO2 estimatives)')
plt.ylim((73,120))
plt.xlim((73,120))
plt.title('MLP regression avaliation: ŷ vs y '+"\n (" +str(neurons)+' units at␣
 ↪hidden layer and tanh as activation function)')
plt.text(75,115,'R2 test: '+str(round(mlp_test.
 ↪loc[idx[neurons,'tanh','lbfgs','constant']].iloc[0,2],4))+'\n'+'R test:␣
 ↪'+str(round(np.sqrt(mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].
 ↪iloc[0,2]),4)), fontsize=12)
fname = 'MLP-Regression_y vs ŷ ' +str(neurons)+ '.png'
```

```
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
#Histogram -> y-ŷ -> ABSOLUTE VALUES
dify=(mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,0]-mlp_test.
 ↪loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,1])
plt.figure(2,figsize=(8,8))
plt.hist(sorted(dify),50, density=True, color='gray')
plt.title('Histogram of residuals - MLP: '+str(neurons)+' units at hidden␣
 ↪layer'+'\n'+'Absolute values')
plt.xlabel('y-ŷ')
plt.ylabel('Probability density')
plt.text(-14,0.175,'Residuals mean: '+str(round(dify.
 ↪mean(0),4))+'\n'+'Residuals standard deviation: '+str(round(dify.std(0),4)),␣
 ↪fontsize=11)
fname = 'MLP-Histogram of residuals_y-ŷ_absolute-'+str(neurons) + '.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
#Residuals -> ABSOLUTE VALUES
plt.figure(3,figsize=(8,8))
plt.plot([0,2600],[0,0],'-.', color='grey')
plt.plot(dify,'k.')
plt.ylabel('y-ŷ')
plt.xlabel('Observation')
plt.ylim((-13,13))
plt.title('Dispersion of the residuals - MLP: '+str(neurons)+' units at hidden␣
 ↪layer'+'\n'+'Absolute values')
fname = 'MLP-Dispersion of the residuals MLP_absolute-' +str(neurons)+ '.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
#Print Results
print("r2_complete data: ", mlp_test.
 ↪loc[idx[neurons,'tanh','lbfgs','constant']].iloc[0,2])
print("r_complete data: ", np.sqrt(mlp_test.
 ↪loc[idx[neurons,'tanh','lbfgs','constant']].iloc[0,2]))
print(" residuals mean and stdev: ", dify.mean(0),dify.std(0))
print(" Maximum and Minimum error: ", abs(dify).max(),abs(dify).min())
print(" Mean absolute error: ",sum(abs(dify))/dify.shape[0])
```

[ ]:
```
# MSE: Mean Squared Error
mse_mlp=sum((mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:
 ↪,0]-mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:
 ↪,1])*(mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:
 ↪,0]-mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,1]))/
 ↪mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,0].shape[0]
np.mean(mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,0]-mlp_test.
 ↪loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,1])
```

[ ]:
```
# MSE: Mean Squared Error
mse_mlp=sum((mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:
 ↪,0]-mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:
 ↪,1])*(mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:
 ↪,0]-mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,1]))/
 ↪mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,0].shape[0]
print(mse_mlp)
np.mean(mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,0]-mlp_test.
 ↪loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,1])
print(dify.mean(0))
print(dify.std(0))
print(np.mean((mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:
 ↪,0]-mlp_test.loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,1])/mlp_test.
 ↪loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,0]),np.mean(dify/mlp_test.
 ↪loc[idx[neurons,'tanh','lbfgs','constant']].iloc[:,0]))
print(work.columns)
print(work.shape)
```

5.3 Sensitivity analysis

[ ]:
```
#Variables Identification
listvar=data.columns[:-1]
data.columns
```

[ ]:
```
#Neural network Model
## 4.2 Multilayer Perceptron
#Number of hidden layers variation
N=30 #maximal number of neurons at hidden layer
kn=5 #number of splits at cross-validation procedure
t=1e-3 #Tolerance (modifications at r2 - > training the net)
niter=7
ns=9
nn=MLPRegressor(hidden_layer_sizes=(ns+1,),
                            activation='tanh',
                            solver='lbfgs',
                            batch_size='auto',
                            learning_rate= 'constant',
                            max_iter=1000,
                            random_state=rs,
                            tol=t,
                            n_iter_no_change=niter,
                            verbose=False)
#Training with all data
rb=nn.fit(xtrain,ytrain)
#Test with the extra testing set - For each cross validation
yhat=nn.predict(xtest)
```

```
yh=yhat.reshape(-1,1)
R2=rb.score(xtest,ytest)
Yh=scalery.inverse_transform(yh)
Yhat=np.reshape(Yh,-1)
#Mean Squared Error - Prediction
mse_mlp=sum((yte-Yhat)*(yte-Yhat))/yte.shape[0]
```

[ ]:
```
#PREDICTION TRAINING SET
ytrhat=nn.predict(xtrain) #Test Prediction
ytrh=scalery.inverse_transform(ytrhat.reshape(-1,1)) #Return to original scale
R2tr=nn.score(xtrain,ytrain) #Real test set
Ytrhat=np.reshape(ytrh,-1)
#Mean Squared Error - Prediction
mse_mlptrain=sum((ytr-Ytrhat)*(ytr-Ytrhat))/ytr.shape[0]
```

[ ]:
```
### ---------  SENSITIVITY ANALYSIS  -------- ###
yMLP = {'Y':np.array(ytr),'yhat':Ytrhat}
#Coefficient of Determination
R2MLP={'Training':R2tr}
#Mean Squared Error (MSE)
MSE_mlp={'Training':mse_mlptrain}
#sum all deviations
sumdevR2MLP=0
#Mean of each variable - original range
meano=xtr.mean(axis=0)
stdevo=xtr.std(axis=0)
#Mean of each variable - scaled
mean=xtrain.mean(axis=0)
stdev=xtrain.std(axis=0)
#Modifications in the input data: substitution of each variable for the its␣
 ↪mean value
for i in range(len(listvar)):# -1 -> due to output
    x1=xtrain.copy()
    x1o=xtr.copy()
    #x1[:,i]=mean[i]
    #Disturbancy: Only mean - original range
    x1o.iloc[:250,i]=meano[i]+0.5*stdevo[i]
    x1o.iloc[250:500,i]=meano[i]+1*stdevo[i]
    x1o.iloc[500:750,i]=meano[i]+1.5*stdevo[i]
    x1o.iloc[750:,i]=meano[i]+2*stdevo[i]
    #mst='m' #Disturbancy: Only mean - Scaled
    x1[:250,i]=mean[i]+0.5*stdev[i]
    x1[250:500,i]=mean[i]+1*stdev[i]
    x1[500:750,i]=mean[i]+1.5*stdev[i]
    x1[750:,i]=mean[i]+2*stdev[i]
    mst='m+kstd' #Disturbancy: Mean+-k*stdev
    #Prediction
```

```
    y1=nn.predict(x1)
    yh1=scalery.inverse_transform(y1.reshape(-1,1)) #Return to original scale
    Y1=np.reshape(yh1,-1)
    r21=nn.score(x1,ytrain)
    #Mean Squared Error - Prediction
    mse_mlp1=sum((ytr-Y1)*(ytr-Y1))/ytr.shape[0]
    #Register
    yMLP.update({'yhat_mean_'+str(listvar[i]):Y1})
    R2MLP.update({str(listvar[i]):r21})
    MSE_mlp.update({str(listvar[i]):mse_mlp1})
    sumdevR2MLP+=(R2tr-r21)
    #Plot Disturbances
    plt.figure(figsize=(10,7))
    plt.plot(x1o.reset_index().iloc[:,i+1],'k--',label='Disturbed')
    plt.plot(xtr.reset_index().iloc[:,i+1],'k',alpha=0.5,label='Not Disturbed')
    plt.legend()
    plt.title(x1o.columns[i])
    plt.ylabel(x1o.columns[i])
    fname = 'Disturbance'+str(x1o.columns[i][:-4])+'.png'
    plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
↪dpi=600)
```

[ ]:
```
#Transform in DataFrame
MSEmlp=pd.DataFrame.from_dict(MSE_mlp,orient='index', columns=['MSE_MLP'])
R2mlp=pd.DataFrame.from_dict(R2MLP,orient='index', columns=['R2_MLP'])
Ymlp=pd.DataFrame.from_dict(yMLP)
MLP_results=MSEmlp.copy()
MLP_results['R2_MLP']=R2mlp.copy()
#Save
fname = '\Sensitivity_MLP.xlsx'
writer = pd.ExcelWriter(dir+fname, engine='openpyxl')
Ymlp.to_excel(writer,sheet_name='ymlp')
MLP_results.to_excel(writer,sheet_name='MSEr2')
writer.save()
writer.close()
```

[ ]:
```
#MEAN SQUARED ERROR CHART
msemlp = MSEmlp.sort_values('MSE_MLP',ascending=False)
plt.figure()
mse=msemlp.plot.bar(figsize=(10,7), title='MSE MLP', color='gray')
plt.plot((0,14),(MSEmlp.iloc[0],MSEmlp.iloc[0]),'k--')
mse.set_ylabel('MSE')
for p in mse.patches:
    mse.annotate(str(round(p.get_height(),2)), (p.get_x() * 0.995, p.
↪get_height() * 1.005))
fname = 'MLP-Sensitivity.png'
```

```
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
```

[ ]:
```
#MEAN SQUARED ERROR CHART
plt.figure()
mse=MSEmlp.plot.bar(figsize=(10,7), title='MSE MLP', color='gray',alpha=0.7)
plt.plot((0,15),(MSEmlp.iloc[0],MSEmlp.iloc[0]),'k--',alpha=0.7)
mse.set_ylabel('Erro quadrático médio (MSE)')
for p in mse.patches:
    mse.annotate(str(round(p.get_height(),3)), (p.get_x() * 0.995, p.
 ↪get_height() * 1.005))
plt.annotate('MSE referência = '+str(round(MSEmlp.iloc[0][0],3))+'ppm',(7,0.
 ↪5*MSEmlp.iloc[0]))
fname = 'MLP-Sensitivity-desorder.png'
plt.savefig(os.path.join(dir, fname), bbox_inches='tight', format='png',␣
 ↪dpi=600)
```

5.3.1 Fim