

UNIVERSIDADE FEDERAL FLUMINENSE  
ESCOLA DE ENGENHARIA  
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

RICARDO BOHADANA MARTINS

DESENVOLVIMENTO DE SOLUÇÃO COMPUTACIONAL BASEADA EM MACHINE  
LEARNING PARA APOIO À MANUTENÇÃO PREDITIVA

NITERÓI  
2022

**RICARDO BOHADANA MARTINS**

**DESENVOLVIMENTO DE SOLUÇÃO COMPUTACIONAL BASEADA EM  
MACHINE LEARNING PARA APOIO À MANUTENÇÃO PREDITIVA**

Trabalho de Conclusão de Curso apresentado  
ao Corpo Docente do Departamento de  
Engenharia Elétrica da Escola de Engenharia da  
Universidade Federal Fluminense, como parte  
dos requisitos necessários à obtenção do título  
de Engenheiro Eletricista.

Orientador(a):

Prof. Dr. Vitor Hugo Ferreira

Niterói, RJ

2022

Ficha catalográfica automática - SDC/BEE  
Gerada com informações fornecidas pelo autor

M379d	<p>Martins, Ricardo Bohadana Desenvolvimento de solução computacional baseada em machine learning para apoio à manutenção preditiva / Ricardo Bohadana Martins ; Vitor Hugo Ferreira, orientador. Niterói, 2022. 82 f. : il.</p> <p>Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica)-Universidade Federal Fluminense, Escola de Engenharia, Niterói, 2022.</p> <p>1. Machine learning. 2. Manutenção preditiva. 3. Gestão de ativos. 4. Produção intelectual. I. Ferreira, Vitor Hugo, orientador. II. Universidade Federal Fluminense. Escola de Engenharia. III. Título.</p>
	CDD -

Bibliotecário responsável: Debora do Nascimento - CRB7/6368

RICARDO BOHADANA MARTINS

**DESENVOLVIMENTO DE SOLUÇÃO COMPUTACIONAL BASEADA EM  
MACHINE LEARNING PARA APOIO À MANUTENÇÃO PREDITIVA**

Trabalho de Conclusão de Curso apresentado  
ao Corpo Docente do Departamento de  
Engenharia Elétrica da Escola de Engenharia da  
Universidade Federal Fluminense, como parte  
dos requisitos necessários à obtenção do título  
de Engenheira(o) Eletricista.

Aprovado em 07 de fevereiro de 2022, com nota 10,0 (dez), pela banca examinadora.

**BANCA EXAMINADORA**

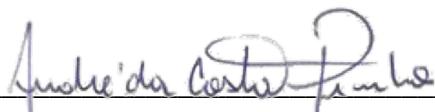


Assinado de forma digital por  
VITOR HUGO FERREIRA  
vhferreira@id.uff.br:01277679673  
Dados: 2022.02.10 13:36:56 -03'00'

---

Prof. Dr. Vitor Hugo Ferreira – Orientador(a)

UFF



---

Prof. Dr. André da Costa Pinho – Membro Convidado

UFF



---

Prof. Dr. Luciano de Oliveira Daniel – Membro Convidado

UFF

Niterói

2022

*Dedico este trabalho aos meus pais Ricardo e Itala, à minha irmã Thayssa e à minha querida Gabriella, que estiveram comigo nos momentos bons e ruins na busca do meu diploma.*

## **AGRADECIMENTOS**

Agradeço ao meu orientador, o Prof. Dr. Vitor Hugo Ferreira, por todo o esforço, contribuição e assistência providenciada durante a prospecção e elaboração deste trabalho. Acredito que, sem seu grande conhecimento e apoio, não seria possível o desenvolvimento deste documento.

Agradeço, também, aos professores John Reed e Rainer Zanghi, que tiveram uma contribuição indireta neste trabalho. Ambos me introduziram ao mundo da programação de computadores, no primeiro semestre de 2017 (disciplina de Programação de Computadores com a linguagem FORTRAN) e no primeiro semestre de 2018 (disciplina de Linguagens de Programação para Engenharia Elétrica com a linguagem C/C++ e Octave/Matlab), respectivamente. Atualmente, minha principal área de atuação, além de parte do foco deste trabalho, é a programação, então deixo aqui meus sinceros agradecimentos.

Agradeço, também, à minha irmã, Dra. Thayssa Bohadana Martins, que além de apoiar e auxiliar minha formação como engenheiro sempre foi, para mim, uma inspiração e um exemplo a ser seguido.

Agradeço, de coração, à Gabriella, por sua autêntica forma de ser, de me motivar, de me inspirar e de me fazer persistir nessa jornada que chega ao fim.

Por fim, agradeço aos meus pais, Ricardo e Itala, por toda a paciência, apoio e por nunca terem medido esforços para me proporcionar um ensino de qualidade durante toda a minha vida.

*“Só por que algo não faz o que você planejou que fizesse, não significa que é inútil.”*

*Thomas Edison*

## RESUMO

Automação industrial, robótica, internet das coisas, inteligência artificial, *Big Data* e computação em nuvem são algumas das inovações trazidas pela quarta revolução industrial. O desenvolvimento dessas tecnologias tem aplicação direta na gestão de ativos dentro de diversos setores que utilizam equipamentos e máquinas sujeitos a falhas e quebras. Nessa perspectiva, algumas empresas já iniciaram o desenvolvimento de soluções de gestão de ativos baseadas na manutenção preditiva, como a plataforma Predix da GE e a Siemens Energy Global. A partir disso, este trabalho realiza o desenvolvimento de uma solução baseada em *machine learning* com o objetivo de auxiliar a tomada de decisão em estratégias preditivas. A solução é desenvolvida utilizando algoritmos e técnicas de desenvolvimento e treinamento de redes neurais em *Python*, com o uso de bibliotecas de código aberto conhecidas (como *TensorFlow* e *Keras*), além de se basear em um banco de dados público da Microsoft Azure AI Gallery, disponível para aplicações de manutenção preditiva. Foi identificada a capacidade de prever, com um tempo razoável de antecedência, a falha de ativos, possibilitando aos engenheiros de manutenção responsáveis, avaliar e agendar intervenções, reduzindo a quantidade de falhas, manutenções e aumentando o tempo de disponibilidade e vida útil das máquinas e equipamentos.

**Palavras-Chave:** gestão de ativos, aprendizado de máquina, manutenção preditiva.

## **ABSTRACT**

Industrial automation, robotics, internet of things, artificial intelligence, Big Data and cloud computing are some of the innovations brought by the fourth industrial revolution. The development of these technologies has direct application in asset management within several sectors that use equipment and machines subject to failures and breakdowns. In this perspective, some companies have already started developing asset management solutions based on predictive maintenance, such as GE's Predix platform and Siemens Energy Global. Thus, this paper carries out the development of a machine learning solution in order to assist decision making in predictive strategies. The development is made using algorithms and techniques for modelling and training neural networks in Python, using well-known open-source libraries (such as TensorFlow and Keras) and a predictive maintenance database made publicly available by Microsoft Azure AI Gallery. The main achievement was the model's capability of predicting, with a reasonable amount of time in advance, asset failure, allowing responsible maintenance engineers to schedule interventions, hence reducing the total number of failures, maintenance costs, machinery downtime and increasing useful lifetime.

**Keywords:** machine learning, asset management, predictive maintenance.

## LISTA DE FIGURAS

Figura 2.1 – Diferentes estratégias de manutenção e seus objetivos.....	5
Figura 2.2 – Gráfico do desempenho de um equipamento de acordo com o tempo, representando o intervalo de manutenção corretiva ( $t_2 - t_1$ ). Neste caso, o reparo só é executado após o ativo apresentar alguma falha ou defeito.....	6
Figura 2.3 – Gráfico do desempenho de um equipamento de acordo com o tempo, representando os intervalos de manutenção preventiva e corretiva. Neste caso, o reparo é executado frequentemente antes do ativo apresentar alguma falha ou defeito, de maneira preventiva. ....	8
Figura 2.4 – Gráfico do desempenho de um ativo de acordo com o tempo, representando o intervalo de tempo de planejamento da intervenção para correção do problema ( $tp$ ) e o desempenho real do equipamento (linha pontilhada). Neste caso, o reparo é feito antes que a máquina falhe, mas tentando, ao máximo, aproveitar completamente a vida útil da máquina, reduzindo os custos de manutenção.....	10
Figura 3.1 – Modelo de um neurônio não linear $k$ com sinais de entrada $x$ , pesos sinápticos ( $w_k$ ), nó de soma, <i>bias</i> – ou viés – ( $b_k$ ), função de ativação ( $\phi$ ) e saída ( $y_k$ ).....	14
Figura 3.2 – Gráfico da função de ativação limiar. .....	15
Figura 3.3 – Gráfico da função de ativação sigmoide. .....	16
Figura 3.4 – Exemplo de uma rede neural com duas camadas escondidas. .....	17
Figura 3.5 - Ilustração das direções do fluxo de dois sinais importantes em uma rede neural: propagação direta de sinais de função e retropropagação de sinais de erro. ....	18
Figura 3.6 – Problema do ajuste excessivo no treinamento de redes neurais.....	20
Figura 5.1 – Amostras da variação da medição de tensão em operação normal. ....	29
Figura 5.2 – Amostras da variação da medição de vibração em operação normal.....	29
Figura 5.3 – Amostras da variação da medição de rotação em operação normal.....	30
Figura 5.4 – Amostras da variação da medição da pressão em operação normal. ....	30
Figura 5.5 – Amostra dos erros, onde 0 indica operação normal. ....	31
Figura 5.6 – Amostra das falhas, onde 0 indica operação normal. ....	31
Figura 5.7 – <i>Boxplot</i> da tensão em relação aos tipos de falha, com 0 indicando operação normal. ....	33
Figura 5.8 – <i>Boxplot</i> da vibração em relação aos tipos de falha, com 0 indicando operação normal. ....	33
Figura 5.9 – <i>Boxplot</i> da rotação em relação aos tipos de falha, com 0 indicando operação normal. ....	34

Figura 5.10 – <i>Boxplot</i> da pressão em relação aos tipos de falha, com 0 indicando operação normal.....	34
Figura 5.11 – Suavização das medidas da tensão com o cálculo da média móvel de 24 períodos (24 horas).....	35
Figura 5.12 – Suavização das medidas de vibração, computando a média móvel de 24 períodos (24 horas).....	36
Figura 5.13 – Suavização das medidas de rotação, computando a média móvel de 24 períodos (24 horas).....	36
Figura 5.14 – Suavização das medidas de pressão, computando a média móvel de 24 períodos (24 horas).....	37
Figura 5.15 – Amostra do comportamento da tensão e sua média durante falha.....	38
Figura 5.16 – Amostra do comportamento da vibração e sua média durante falha .....	38
Figura 5.17 – Amostra do comportamento da rotação e sua média durante falha .....	39
Figura 5.18 – Amostra do comportamento da pressão e sua média durante falha .....	39
Figura 5.19 – Amostra das primeiras variáveis de entrada definidas, máximo e mínimo (6hrs). .....	41
Figura 5.20 – Amostra das variáveis de média móvel simples e exponencial (24hrs) para a tensão.....	41
Figura 5.21 - Amostragem do ponto de falha, variável de tempo e a saída esperada do modelo. .....	42
Figura 5.22 - Validação cruzada para 8 arquiteturas distintas.....	46
Figura 5.23 - Perda de cada arquitetura testada por época .....	47
Figura 5.24 - Esforço computacional (tempo de cada época em segundos).....	47
Figura 5.25 - Precisão dos modelos selecionados para treinamento completo por época.....	49
Figura 5.26 - Perda dos modelos selecionados para treinamento completo por época .....	49
Figura 5.27 - Esforço computacional acumulativo dos modelos selecionados para treinamento completo .....	50
Figura 5.28 - Gráficos das Métricas de acordo com o valor limite de decisão aplicado .....	52
Figura 5.29 – Simulação da operação da solução para as amostras da máquina de número 96. .....	55
Figura 5.30 – Simulação da operação da solução para as amostras da máquina de número 97. .....	56
Figura 5.31 – Simulação da operação da solução para as amostras da máquina de número 98. .....	57

Figura 5.32 – Simulação da operação da solução para as amostras da máquina de número 99.	58
.....	.....
Figura 5.33 – Simulação da operação da solução para as amostras da máquina de número 100.	59
.....	.....
Figura 5.34 – Detalhes de falsos negativos da máquina 100 devido à variável de erro. ....	60

## **LISTA DE TABELAS**

Tabela 3.1 - Exemplo de uma matriz de confusão .....	22
Tabela 5.1 - Arquitetura das redes neurais utilizadas na validação cruzada. ....	45
Tabela 5.2 – Matriz de confusão e métricas de avaliação do modelo desenvolvido. ....	52

## **LISTA DE ABREVIATURAS E SIGLAS**

PdM	Predictive Maintenance
ABNT	Associação Brasileira de Normas Técnicas
IA	Inteligência Artificial
GA	Gestão de Ativos
RN	Redes Neurais
IAM	Institute of Asset Management
ICA-LA	International Copper Association Latin America
IoT	Internet of Things
ARIMA	<i>AutoRegressive Integrated Moving Average</i>
MME	Média móvel exponencial

## LISTA DE SÍMBOLOS

$t_n$	Tempo ou Instante $n$
$b_k$	<i>Bias</i> ou viés
$x_j$	Valor do estímulo da entrada $j$ do neurônio $k$
$w_{kj}$	Peso sináptico da entrada $j$ no neurônio $k$
$v_k$	Valor da saída do neurônio $k$ antes de sua ativação
$\varphi$	Função de ativação
$y_k$	Saída do neurônio $k$
$N$	Quantidade de amostras
$\mathcal{T}$	Conjunto de treinamento
$\Delta$	Variação ou Correção
$\delta$	Gradiente local
$e$	Erro ou desvio
$d$	Resposta desejada
$p$	Probabilidade

## SUMÁRIO

1	INTRODUÇÃO.....	1
1.1	MOTIVAÇÃO .....	1
1.2	OBJETIVO .....	2
1.3	ESTRUTURA DO TRABALHO .....	2
2	GESTÃO DE ATIVOS .....	3
2.1	MANUTENÇÃO CORRETIVA .....	5
2.1.1	MANUTENÇÃO CORRETIVA NÃO PLANEJADA .....	6
2.1.2	MANUTENÇÃO CORRETIVA PLANEJADA .....	7
2.2	MANUTENÇÃO PREVENTIVA .....	7
2.3	MANUTENÇÃO PREDITIVA .....	9
2.3.1	CONCEITO .....	9
2.3.2	TECNOLOGIAS APLICADAS .....	11
2.3.3	SOLUÇÕES DISPONÍVEIS .....	12
3	REDES NEURAIS PARA CLASSIFICAÇÃO .....	13
3.1	MODELO DE UM NEURÔNIO.....	14
3.2	FUNÇÕES DE ATIVAÇÃO .....	15
3.2.1	FUNÇÃO DE LIMIAR .....	15
3.2.2	FUNÇÃO SIGMOIDE .....	16
3.3	PERCEPTRON DE MÚLTIPHAS CAMADAS .....	16
3.4	APRENDIZAGEM EM LOTE.....	18
3.5	ALGORITMO DE RETROPROPAGAÇÃO DO ERRO.....	18
3.6	GENERALIZAÇÃO DA REDE.....	20
3.7	MÉTRICAS DE AVALIAÇÃO DA REDE .....	21
3.7.1	MATRIZ DE CONFUSÃO .....	21
3.7.2	ACURÁCIA .....	22
3.7.3	PRECISÃO .....	22
3.7.4	RECALL.....	23
4	PROCESSO DE DESENVOLVIMENTO DE UM ALGORITMO DE APRENDIZADO DE MÁQUINA.....	23
4.1	DEFINIÇÃO DO PROBLEMA E DO SUCESSO.....	24
4.2	ENTENDIMENTO E IDENTIFICAÇÃO DOS DADOS .....	25
4.3	PREPARAÇÃO DOS DADOS .....	25

4.4	DETERMINAÇÃO DAS CARACTERÍSTICAS E TREINAMENTO DO MODELO	26
4.5	AVALIAÇÃO DO DESEMPENHO E ESTABELECIMENTO DE METAS.....	26
5	PROCESSO DE DESENVOLVIMENTO DA SOLUÇÃO DE APOIO À MANUTENÇÃO PREDITIVA.....	27
5.1	TECNOLOGIAS UTILIZADAS PARA DESENVOLVIMENTO DO ALGORITMO	
		28
5.2	ANÁLISE DOS DADOS DISPONÍVEIS .....	28
5.3	FEATURE ENGINEERING .....	32
5.3.1	VISUALIZAÇÃO DA RELAÇÃO ENTRE VARIÁVEIS .....	32
5.3.2	IDENTIFICAÇÃO DO INTERVALO DE PREVISÃO.....	37
5.3.3	DEFINIÇÃO DAS VARIÁVEIS DE ENTRADA.....	40
5.3.4	DEFINIÇÃO DA VARIÁVEL DE SAÍDA.....	42
5.4	ARQUITETURA E TREINAMENTO DA REDE NEURAL .....	42
5.4.1	QUANTIDADE DE CAMADAS ESCONDIDAS .....	43
5.4.2	QUANTIDADE DE NEURÔNIOS EM CADA CAMADA ESCONDIDA .....	43
5.4.3	ESCOLHA FINAL DA ARQUITETURA E TREINAMENTO .....	44
5.5	AVALIAÇÃO DA PERFORMANCE E PÓS PROCESSAMENTO .....	50
5.6	SIMULAÇÃO DA OPERAÇÃO DA SOLUÇÃO.....	54
6	CONCLUSÃO E TRABALHOS FUTUROS .....	61
	REFERÊNCIAS BIBLIOGRÁFICAS .....	64

## 1 INTRODUÇÃO

### 1.1 MOTIVAÇÃO

Para diversos setores e companhias, um bom desempenho é consequência direta de seus ativos. As concessionárias de energia elétrica, por exemplo, dependem dos ativos dos sistemas de geração, transmissão e distribuição para suprir a demanda dos consumidores e gerar resultados. Nessa perspectiva, o que se observa para esses empreendimentos se estende para outras áreas que atuam com equipamentos, máquinas e componentes que necessitam de supervisão, substituição ou manutenção.

Com a chegada da quarta revolução industrial, vê-se um investimento crescente no desenvolvimento de estratégias para aumentar a confiabilidade, eficiência e produtividade na indústria. Isso porque a Indústria 4.0 marca uma série de mudanças nos processos de produção, nas operações e nos sistemas por meio da conectividade digital integrada, também classificada como fábrica inteligente, onde os processos físicos são monitorados e recriados em um ambiente virtual para tomada de decisões [1].

Em um mundo onde a demanda por bens, serviços, desempenho e lucro só aumenta, independentemente do setor, é economicamente inviável realizar a compra de novos equipamentos para substituir aqueles que apresentarem defeito. A estratégia mais comum utilizada é a de executar, periodicamente, manutenções preventivas. Entretanto, essa abordagem é ineficiente pelo fato de aumentar os custos de manutenção e não aproveitar toda a vida útil do equipamento. Nesse cenário, a gestão de ativos deixa de ser uma oportunidade e passa a ser uma necessidade para aquelas empresas que desejam reduzir seus custos de manutenção, aumentar sua produtividade e seu lucro.

Diante disso, o estudo sobre o desenvolvimento de um algoritmo de apoio a tomada de decisão, capaz de inferir, com certo grau de assertividade, quanto próximo de falhar está um determinado ativo, é relevante não somente para concessionárias de energia elétrica, mas para diversas companhias nas mais variadas áreas de atuação. Isso porque permitiria aplicar diferentes estratégias de gestão de ativos, agendando manutenções antes da falha e aproveitando ao máximo a vida útil do equipamento.

Com isso em mente, surgem algumas questões que serão abordadas neste trabalho, como:

- É possível prever quando um ativo passará a operar de fora de sua faixa ótima?
- É possível prever uma eventual falha?

## 1.2 OBJETIVO

O presente trabalho tem como objetivo principal o desenvolvimento de um algoritmo computacional na linguagem *Python* capaz de modelar uma rede neural (fazendo uso das bibliotecas *TensorFlow* e *Keras*) a partir de variáveis de processo de ativos para prever futuras falhas em seus componentes e possibilitar a implementação de uma estratégia de manutenção preditiva.

Através de uma análise orientada a dados, seguida de um pré-processamento para seleção das características de entrada (*features*) adequadas e uma divisão das amostras em conjuntos de treinamento e validação, será possível executar o treinamento da rede neural e o teste do seu desempenho. A partir disso, infere-se sobre a capacidade do modelo de obter sucesso em suas previsões, além de determinar a janela de tempo pré-falha que o algoritmo desenvolvido será capaz de detectar a anomalia.

O conjunto de amostras utilizadas como base para o desenvolvimento da solução são variáveis de processo e de estado (condição de operação) de diversas máquinas descaracterizadas (sem detalhamento do tipo de máquina, modelo, processo utilizado etc.), de um banco público da plataforma *Azure AI Gallery* da Microsoft, voltada para o desenvolvimento de soluções de manutenção preditiva, ou PdM (*Predictive Maintenance*). Assim, o principal questionamento a ser feito é sobre a capacidade de associar as variáveis de processo disponíveis com o desempenho, saúde e vida útil remanescente da máquina para decidir sobre a janela de tempo pré-falha possível de ser determinada.

## 1.3 ESTRUTURA DO TRABALHO

Este trabalho foi estruturado em 6 capítulos, contemplando a base teórica dos temas abordados além de desenvolvimento e conclusão. Os assuntos a serem abordados em cada capítulo são apresentados, de modo geral, a seguir:

- O capítulo inicial introduz o leitor ao tema, a motivação do trabalho e a estruturação do documento;

- O segundo capítulo apresenta a fundamentação teórica sobre diferentes estratégias de gerenciamento de ativos aplicadas por grandes empresas;
- O terceiro capítulo apresenta, em termos gerais, a teoria que envolve o desenvolvimento de redes neurais para classificação;
- O quarto capítulo explica o processo de desenvolvimento a ser seguido para a construção de soluções utilizando diferentes abordagens de aprendizado de máquinas;
- O quinto capítulo expõe a análise sobre os dados utilizados e detalha a escolha das características da rede neural, a comparação entre diferentes arquiteturas, o treinamento e os testes desse algoritmo com os dados de validação e de simulação em tempo real;
- Por fim, o sexto capítulo apresenta a conclusão do trabalho e uma proposta de estudo a ser realizada posteriormente.

## 2 GESTÃO DE ATIVOS

As seções 3.2.1 e 3.3.1 da norma ABNT NBR ISO 55000:2014 [2] – que fornece uma visão geral da gestão de ativos (GA) e de sistemas de GA – apresenta as seguintes definições:

- a) gestão de ativos: “atividade coordenada de uma organização para obter valor a partir dos ativos”;
- b) ativos: “item, algo ou entidade que tem valor real ou potencial para uma organização”.

Na verdade, a GA vai além do exposto na norma. É sobre agregar valor e alcançar os objetivos definidos pela organização, o que traz uma forma diferente de abordar e pensar, onde cada empresa é responsável por determinar, para seu caso específico, o que considera ser o seu valor e estratégias sobre como gerenciar seus ativos para obter o melhor valor total [3].

Em essência, a GA é um processo de orientação para compra, uso e alienação de ativos com o objetivo de obter o melhor desempenho e benefício, controlando os custos e riscos associados ao longo do seu ciclo de vida. No setor elétrico, onde as concessionárias de energia dependem de seus ativos para obter um bom desempenho, o sucesso na GA se traduz no sucesso da prestação de serviço. Isso porque todo o sistema elétrico é formado de ativos que são essenciais para que a produção, o transporte e a distribuição de eletricidade ocorram com qualidade e confiabilidade. Desse modo, os serviços prestados pela empresa de energia são

notavelmente influenciados pela forma como seus principais ativos são gerenciados, seja na opção por um ativo mais eficiente e que minimize os custos ao longo de sua vida útil ou na substituição de um equipamento antes que uma falha indesejada aconteça.

Nesse contexto, e com a chegada das ideias e conceitos da indústria 4.0, a GA se mostrou muito importante nos últimos anos devido à necessidade de manter o sistema elétrico seguro e operando, mesmo com o envelhecimento de seus ativos. A International Copper Association Latin America (ICA-LA) realizou duas pesquisas nesse tema (ano de 2011 e 2014), comparando as empresas do ramo de energia elétrica que já põem em prática os principais conceitos de GA e aquelas que ainda não realizam essa gestão, constatando seus principais benefícios e futuros desafios [4]. Na pesquisa mais recente, publicada em 2014, 77% das empresas dizem acreditar que a gestão de ativos impacta na tarifa de energia elétrica. Ainda, de acordo com a pesquisa da ICA-LA as principais vantagens obtidas pelas empresas brasileiras que já aplicam uma estratégia de GA vão desde a melhoria do reconhecimento dos investimentos na revisão tarifária, passando pela possibilidade de elaborar um plano de substituição para ativos antes da falha até a melhoria de indicadores de desempenho.

Assim, a gestão de ativos e a confiabilidade são conceitos estritamente relacionados e fundamentais para garantir a eficiência industrial. A confiabilidade de um ativo se baseia na probabilidade de um sistema ou componente cumprir sua função em um determinado período de tempo. Logo, quanto maior a confiabilidade menor a probabilidade de ocorrer uma falha. Contudo, a garantia de uma confiabilidade alta depende diretamente do conjunto de ações que acompanhem os resultados desses ativos. Assim, a gestão de ativos suporta e sustenta a confiabilidade dos ativos na engenharia de manutenção [5].

Analisando com maior detalhamento, observa-se que existem diferentes abordagens para a engenharia de manutenção dentro de uma estratégia de gestão de ativos, como mostra a figura 2.1.

Figura 2.1 – Diferentes estratégias de manutenção e seus objetivos



Fonte: [6] – Adaptado.

Dependendo da aplicação, há diferentes estratégias que podem ser seguidas e que providenciam diferentes vantagens e resultados. Estas diferentes possibilidades serão discutidas nas próximas seções.

## 2.1 MANUTENÇÃO CORRETIVA

Na manutenção corretiva, as máquinas e equipamentos são utilizados até o momento em que apresentarem um desempenho diferente do esperado ou um defeito. Com isso em mente, a estratégia de manutenção corretiva não é, exclusivamente, uma manutenção de emergência.

De fato, esta estratégia é bem simples e se aplica apenas a dispositivos cuja falha não causa impacto imediato, seja na geração, transmissão ou distribuição de eletricidade. De modo geral, pode-se pensar no caso de uma lâmpada que só é substituída após queima. Contudo, em sistemas mais complexos, a interrupção repentina e não programada no funcionamento de uma máquina ou equipamento pode causar grandes impactos negativos na segurança, no ambiente, na qualidade do produto ou até propagar os danos para outros ativos. Como mencionado, dois eventos podem ocasionar esse tipo de manutenção, são eles:

- Desempenho abaixo do esperado apontado pelo acompanhamento das variáveis de processo operacionais;
- Ocorrência de falha ou defeito.

Assim, o objetivo principal da ação corretiva é corrigir ou restaurar as condições de funcionamento do equipamento ou sistema e pode ser dividida em duas classes:

1. Manutenção Corretiva Não Planejada
2. Manutenção Corretiva Planejada

### **2.1.1 MANUTENÇÃO CORRETIVA NÃO PLANEJADA**

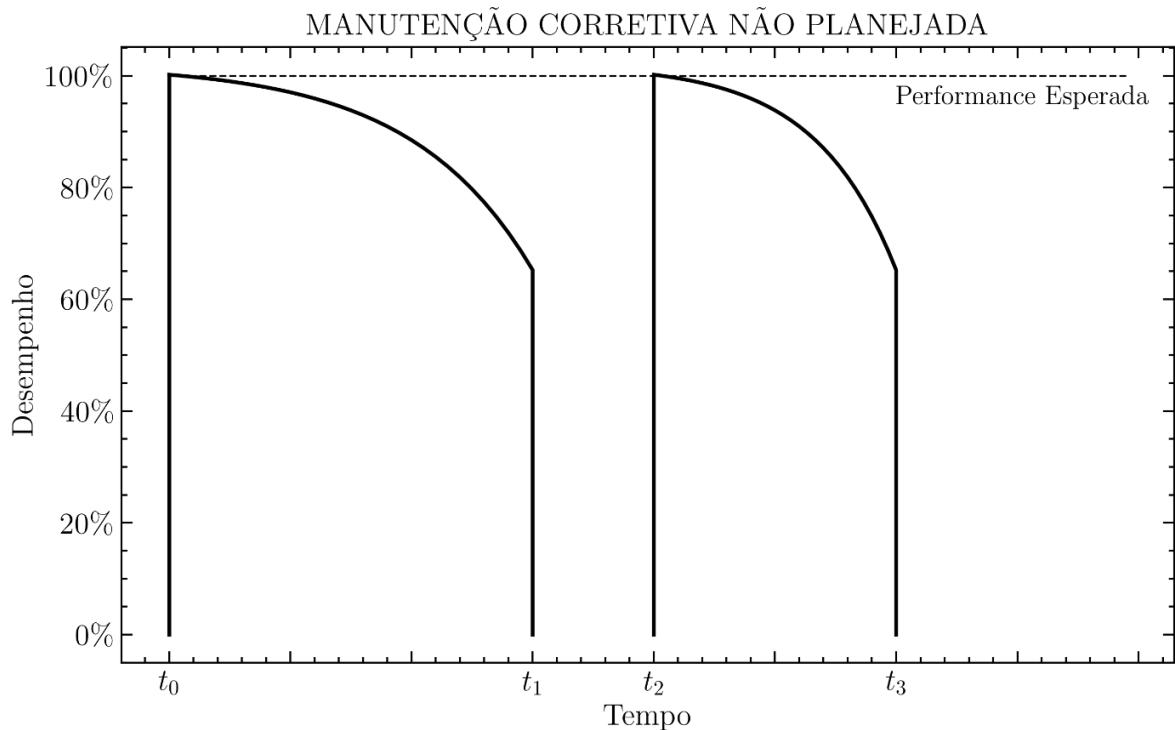
Também conhecida como não programada ou emergencial, a manutenção corretiva não planejada é caracterizada pela atuação após um evento já ocorrido, como uma falha, não havendo tempo suficiente para preparação do serviço.

Normalmente, este tipo de estratégia implica em custos elevados, pois a quebra inesperada pode acarretar perda na qualidade do produto final, perdas de produção e gerar custos altos indiretos de manutenção.

Vale ressaltar, ainda, que existem equipamentos que possuem um desempenho constante ao longo de sua vida útil, seguido de falha instantânea, como as lâmpadas.

A figura 2.2 ilustra uma estratégia de manutenção corretiva não planejada aplicada a um equipamento ou sistema que apresenta queda de desempenho com o tempo. Nela é possível perceber que o tempo para falha é desconhecido e  $t_3 - t_2$  é diferente de  $t_1 - t_0$ . Isso pode ocorrer devido ao desgaste de outros componentes do ativo que venham a falhar em seguida ou devido ao desgaste geral do ativo que provocou nova falha, nesse caso, em menor tempo. Além disso, o período de intervenção para correção da falha é  $t_2 - t_1$ .

Figura 2.2 – Gráfico do desempenho de um equipamento de acordo com o tempo, representando o intervalo de manutenção corretiva ( $t_2 - t_1$ ). Neste caso, o reparo só é executado após o ativo apresentar alguma falha ou defeito.



Fonte: Gráfico 3.1 – Manutenção Corretiva Não Planejada [6] - Adaptado.

### **2.1.2 MANUTENÇÃO CORRETIVA PLANEJADA**

Por outro lado, a manutenção corretiva planejada se baseia na correção do desempenho abaixo do esperado do equipamento. Nessa perspectiva, uma atividade planejada, de modo geral, acaba sendo sempre mais rápida, segura, econômica e de melhor qualidade do que aquela não planejada, mesmo que a decisão seja de realizar a manutenção após a falha do equipamento. Essa decisão, provavelmente, foi feita através da análise de informações fornecidas pelo acompanhamento do ativo e um plano de intervenção já estará pronto para entrar em ação após a ocorrência do defeito.

Assim, a manutenção corretiva planejada se torna dependente da qualidade das informações fornecidas pelo acompanhamento de seus ativos.

## **2.2 MANUTENÇÃO PREVENTIVA**

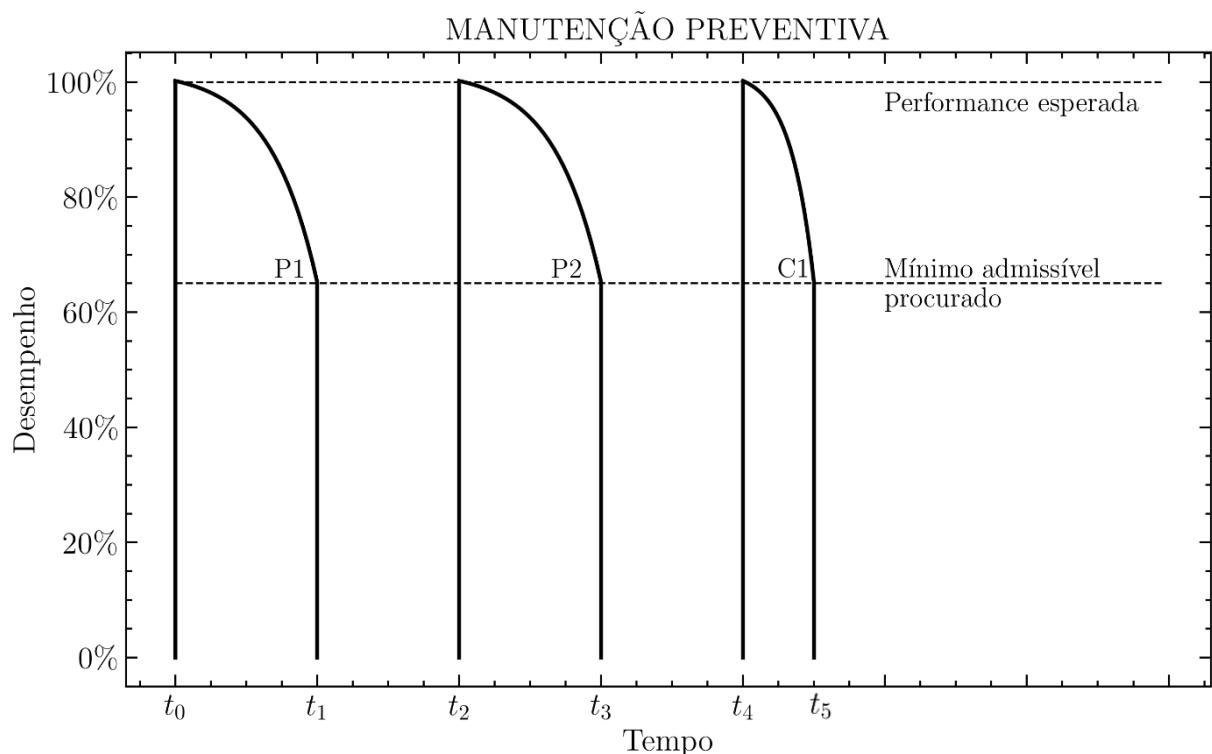
Diferentemente da estratégia corretiva, o principal objetivo da manutenção preventiva é reduzir a ocorrência de falhas de qualquer natureza ou a queda de desempenho a partir do planejamento de verificações e reparos frequentes com intervalos predeterminados.

Na figura 2.3 observa-se que P1 e P2 identificam as manutenções preventivas, que são executadas em intervalos pré-determinados em função da operação do ativo. Nesse caso,  $t_1$  –

$t_0$  e  $t_3 - t_2$  são iguais e representam o intervalo pré-determinado, enquanto  $t_2 - t_1$  e  $t_4 - t_3$  representam os períodos de intervenção.

Por outro lado, a execução de manutenção preventiva tenta evitar a ocorrência de falhas espontâneas e a necessidade de manutenções corretivas, mas não elimina sua possibilidade. Por esse motivo, C1 identifica a manutenção corretiva – que ocorre antes do tempo pré-determinado para nova intervenção preventiva, assim  $t_5 - t_4$  é menor que  $t_3 - t_2$ .

Figura 2.3 – Gráfico do desempenho de um equipamento de acordo com o tempo, representando os intervalos de manutenção preventiva e corretiva. Neste caso, o reparo é executado frequentemente antes do ativo apresentar alguma falha ou defeito, de maneira preventiva.



Fonte: Gráfico 3.2 – Autor.

Essa estratégia será muito bem vista em casos onde o preço de peças de reposição é baixo, o trabalho de manutenção é simples e o custo de falhas é elevado – tanto no âmbito de danificar outros componentes do sistema quanto de suas implicações na segurança operacional ou pessoal.

O ponto positivo da aplicação desse modelo de gestão, além da possibilidade de planejamento e conhecimento prévio das ações, é que os ativos estão, na maioria dos casos, sempre em boas condições de operação devido às fiscalizações regulares executadas. Por outro lado, um grande desafio nessa estratégia é determinar qual o momento certo de realizar o reparo

preventivo do equipamento. Isso porque cada ordem gera um custo e, logicamente, quanto maior o número de ordens maior o custo para a empresa que aplica essa estratégia.

Vale mencionar que, em determinados setores, como na aviação, a estratégia preventiva é quase obrigatória pois a segurança se torna o fator mais importante na análise da estratégia a ser utilizada.

Então, apesar de muito eficiente, em termos de eliminação quase total da falha de ativos, a gestão de ativos por meio de manutenções preventivas pode gerar um custo muito mais alto do que o realmente necessário, pois não aproveita ao máximo a vida útil do equipamento.

## **2.3 MANUTENÇÃO PREDITIVA**

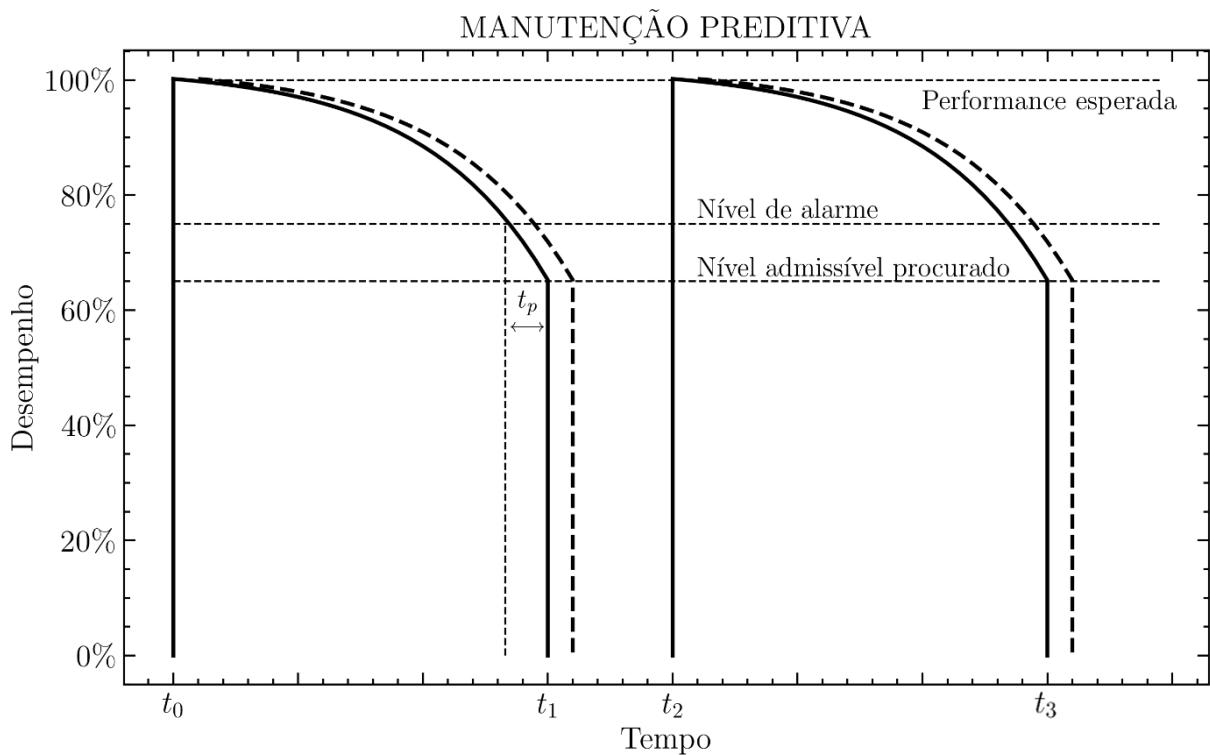
### **2.3.1 CONCEITO**

Diferentemente da manutenção corretiva, ela não permite a falha do equipamento e ao contrário da manutenção preventiva, não regulariza a quantidade de ordens de reparo, pois essa prática gera custos extras. Na verdade, seu objetivo é prever quando um ativo estará no fim de sua vida útil e necessitará de manutenção com base em parâmetros de condição e desempenho. Nesse sentido, a manutenção preditiva tem seu foco na disponibilidade do ativo, à medida que não promove um excesso de intervenções.

A intervenção, nesta estratégia, ocorre quando o grau de degradação do ativo se aproxima ou atinge um limite estabelecido previamente. Esse tipo de estratégia permite o planejamento prévio do serviço e de alternativas ou atividades de produção necessárias. Assim, pode-se dizer que quando a intervenção é decidida, a manutenção preditiva se transforma numa manutenção corretiva planejada.

Na figura 2.4 abaixo, observa-se o tempo de funcionamento do ativo ( $t_3 - t_2$  e  $t_1 - t_0$ ), o instante de detecção da necessidade de manutenção ( $t_1 - t_p$ ), o intervalo de planejamento da intervenção para correção do problema ( $t_p$ ), o período de manutenção ( $t_2 - t_1$ ), o desempenho real (linha pontilhada) e o previsto pela estratégia (linha contínua) do equipamento.

Figura 2.4 – Gráfico do desempenho de um ativo de acordo com o tempo, representando o intervalo de tempo de planejamento da intervenção para correção do problema ( $t_p$ ) e o desempenho real do equipamento (linha pontilhada). Neste caso, o reparo é feito antes que a máquina falhe, mas tentando, ao máximo, aproveitar completamente a vida útil da máquina, reduzindo os custos de manutenção.



Fonte: Autor.

Nessa perspectiva, com a estratégia preditiva, torna-se possível minimizar os custos de manutenção enquanto se maximiza o aproveitamento da vida útil do equipamento. Naturalmente, colocar em prática essa abordagem requer que haja uma digitalização da empresa seguindo os conceitos da indústria 4.0 - com dados históricos e monitoramento em tempo real dos ativos, além da capacidade de integrar seu funcionamento com soluções e sistemas digitalizados - o que nem sempre se observa na realidade, uma vez que são conceitos relativamente novos. Nesse sentido, para aplicação desta estratégia faz-se necessário, muitas vezes, realizar um investimento inicial tanto em infraestrutura quanto no desenvolvimento do algoritmo de previsão a ser utilizado para determinar a vida útil restante do ativo. Ainda assim,

de acordo com Kardec e Nascif (2009), o desenvolvimento de um programa de acompanhamento de ativos bem gerenciado apresenta uma relação de custo/benefício de 1 para 5.

Por fim, é fundamental que a equipe responsável pela análise dos dados coletados e diagnóstico seja bem treinada, pois mais importante do que realizar a medição no equipamento é a capacidade de leitura e interpretação para gerar conclusões significativas e acertadas dessas amostras.

### 2.3.2 TECNOLOGIAS APLICADAS

Em um momento de transformação digital global e indústria 4.0, o conceito de manutenção preditiva vem se tornando gradativamente mais comum no meio industrial. À medida que novas recursos e tecnologias são criados e aprimorados, essa estratégia se torna mais precisa, eficiente e econômica.

Existem diversas soluções e conceitos tecnológicos que são capazes de oferecer um suporte tanto operacional quanto informativo para a equipe de gestores e de manutenção. Em um contexto de aprimoramento e expansão, a internet das coisas, ou em inglês *internet of things* (IoT), se mostra como uma das grandes aliadas da manutenção preditiva e está cada vez mais presente em diversas empresas de diferentes setores.

A IoT, por intermédio de sensores, softwares e dispositivos de comunicação inteligentes, mantém toda a linha produtiva conectada. Com esse alto poder de integração, aumenta-se a capacidade de leitura e compartilhamento de dados em tempo real de diversas partes da cadeia produtiva com gestores e profissionais de manutenção. Na prática, esses dados disponibilizados fornecem informações valiosas a respeito das condições de operação e funcionamento dos equipamentos supervisionados. A manutenção preditiva é um foco da IoT Industrial, proporcionando economia de custos de até 12% em comparação com a manutenção planejada, reduzindo os custos totais de manutenção em até 30% e eliminando falhas em até 70% [1].

O aprendizado de máquina, ou em inglês *machine learning*, também se destaca como uma tecnologia de modernização no setor de manutenção. No contexto da manutenção preditiva, exerce o papel de tornar o supervisionamento e operações mais autônomos e menos dependentes de intervenções. Com o apoio de algoritmos e sensores específicos, é possível agir, de forma antecipada, em qualquer equipamento no qual seja identificado um ponto de atenção. Assim, é possível organizar uma estratégia de manutenção, programando análises de

desempenho e organizando a rotina de técnicos para inspecionar equipamentos onde foram identificados problemas.

Em um mundo impulsionado por informações, um dos bens intangíveis mais valiosos atualmente são os dados. No âmbito da manutenção, não é diferente. Quanto maior a quantidade de dados disponível para análise ou estudo, melhor o resultado. No contexto da manutenção preditiva, os dados coletados em tempo real pela IoT não devem ser descartados, pois históricos de falha, reparos e outras informações podem ser cruciais no projeto de uma estratégia ou em uma tomada de decisão. Assim, com o apoio de softwares de gestão ou centralização das informações e dados coletados, as equipes de manutenção têm mais visibilidade sobre toda a infraestrutura, performance e condição de operação da linha produtiva.

### **2.3.3 SOLUÇÕES DISPONÍVEIS**

Como mencionado anteriormente, já existem algumas soluções e softwares disponibilizados por empresas especializadas para dar suporte a equipes e gestores de manutenção com objetivo de traçar uma estratégia preditiva.

A Plataforma Predix da GE Digital, por exemplo, oferece segurança e escalabilidade para IoT em nuvem. É uma base de nuvem de internet das coisas (IoT) segura e escalável compartilhada por aplicativos da GE Digital, como o Gestão do Desempenho de Ativos, ou *Asset Performance Management (APM)*, e Gestão do Desempenho de Operações, ou *Operations Performance Management (OPM)* [7]. Mais especificamente, o aplicativo APM, ou Gestão do Desempenho de Ativos, seria equivalente à um sistema de supervisionamento preditivo, uma vez que, segundo o próprio website da empresa, é responsável por ajudar a garantir a confiabilidade e disponibilidade do equipamento ao mesmo tempo que reduz os custos operacionais e de manutenção.

Outro exemplo que pode ser mencionado é a *Asset Performance Management for Power Plants*, da Siemens Energy, uma plataforma orientada a análise de dados que facilita a transição para um modelo de manutenção com foco em confiabilidade e no agendamento de inspeções e intervenções de manutenção de acordo com o risco [8]. Ainda, de acordo com o website da empresa, o núcleo do sistema é composto por uma biblioteca de modelos de:

- ativos, classificados de acordo com a criticalidade de alguns aspectos como a segurança, saúde e ambiente;
- falha, modo, efeito e análise, onde são avaliados e descritos os ativos com potencial de falha, seus efeitos associados e as intervenções apropriadas;

- índice de saúde, apresentando uma indicação a respeito da condição dos ativos;
- vida útil remanescente, onde é estimada a vida útil restante de cada equipamento supervisionado, considerando dados históricos, leituras de indicadores, quantidade de alarmes, entre outros.

No entanto, essas ferramentas, além de outras disponíveis no mercado, não informam como foi desenvolvida essa solução, como suas previsões são feitas ou no que se baseiam. Isso é consequência de uma estratégia de propriedade intelectual, que tem como objetivo vender o seu produto ao mesmo tempo em que protege o conhecimento da empresa.

A aplicação de modelos de aprendizado de máquina para solucionar problemas de manutenção com a aplicação de estratégias preditivas é uma tendência atual por estar mostrando desempenho interessante. Esse desempenho é comprovado por uma abordagem com múltiplos classificadores aplicada à um exemplo simulado e um problema de manutenção na fabricação de semicondutores de referência [9]. Ainda, outra aplicação que comprova essa forte tendência e o bom desempenho é a aplicação da Média Móvel Auto-Regressiva Integrada, ou em inglês *AutoRegressive Integrated Moving Average (ARIMA)*, em dados de séries temporais coletados de sensores de máquinas de corte longitudinal para prever possíveis falhas e defeitos de qualidade, melhorando o processo geral de fabricação e reduzindo os custos de manutenção [10].

Com isso, o presente trabalho serve como ponto de partida para o desenvolvimento de um algoritmo capaz de auxiliar a implementação de uma estratégia de manutenção preditiva baseada em código aberto.

### **3 REDES NEURAIS PARA CLASSIFICAÇÃO**

De modo geral, pode-se dizer que uma rede neural é um modelo matemático projetado para se assemelhar a um cérebro na medida em que soluciona um problema ou realiza uma tarefa específica [11]. Entretanto, vale ressaltar que ainda há um longo caminho a ser percorrido antes de construir-se uma arquitetura computacional com a mesma capacidade e velocidade de processamento que o cérebro humano (se é que será possível). Nessa perspectiva, as redes neurais, apesar de não serem tão complexas e eficientes como um cérebro humano, trazem dois benefícios importantes:

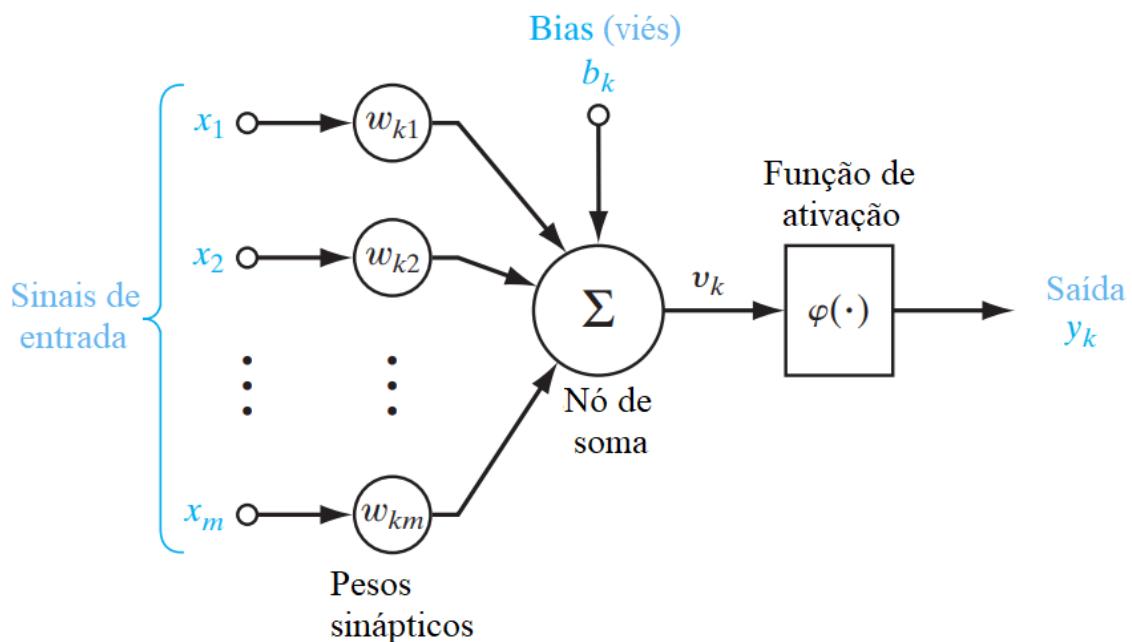
1. Aprendizagem: capacidade de aprender
2. Generalização: capacidade de produzir saídas adequadas para entradas que não estavam presentes durante o treinamento/aprendizagem.

### 3.1 MODELO DE UM NEURÔNIO

Um neurônio consiste em uma unidade de processamento de informações e é peça-chave na construção de uma rede neural. A figura 3.1 mostra o modelo de um único neurônio  $k$  e os elementos básicos que garantem seu funcionamento. Nela é possível observar 4 estruturas principais que constituem o modelo:

1. Um conjunto de conexões – ou sinapses –, cada uma com seu respectivo peso sináptico.
2. Um nó somador, responsável por computar a soma de todos os sinais após multiplicação com seus respectivos pesos sinápticos.
3. Um *bias* ou viés, responsável pelo efeito de incrementar ou subtrair o próprio valor da entrada da função de ativação, dependendo se for positivo ou negativo.
4. Uma função de ativação, responsável por limitar a amplitude da saída de um neurônio. Na maioria dos casos, a faixa de valores da saída normalizada de um neurônio pode ser escrita como  $[0, 1]$  ou  $[-1, 1]$ .

Figura 3.1 – Modelo de um neurônio não linear  $k$  com sinais de entrada  $x$ , pesos sinápticos ( $w_k$ ), nó de soma, *bias* – ou viés – ( $b_k$ ), função de ativação ( $\varphi$ ) e saída ( $y_k$ ).



Fonte: Figura 5 – Modelo de um neurônio não linear  $k$  [11] - Adaptado.

Matematicamente, representa-se o neurônio da figura 3.1 com as seguintes equações:

$$v_k = \left( \sum_{j=1}^m w_{kj} x_j \right) + b_k \quad (3.1)$$

$$y_k = \varphi(v_k) \quad (3.2)$$

Onde, na equação 3.1,  $x_1, x_2, \dots, x_m$  são os sinais de entrada,  $w_{k1}, w_{k2}, \dots, w_{km}$  são os pesos sinápticos do neurônio  $k$  e  $b_k$  é o viés (*bias*). Na equação 3.2,  $\varphi$  é a função de ativação e  $y_k$  é a saída do neurônio  $k$ .

## 3.2 FUNÇÕES DE ATIVAÇÃO

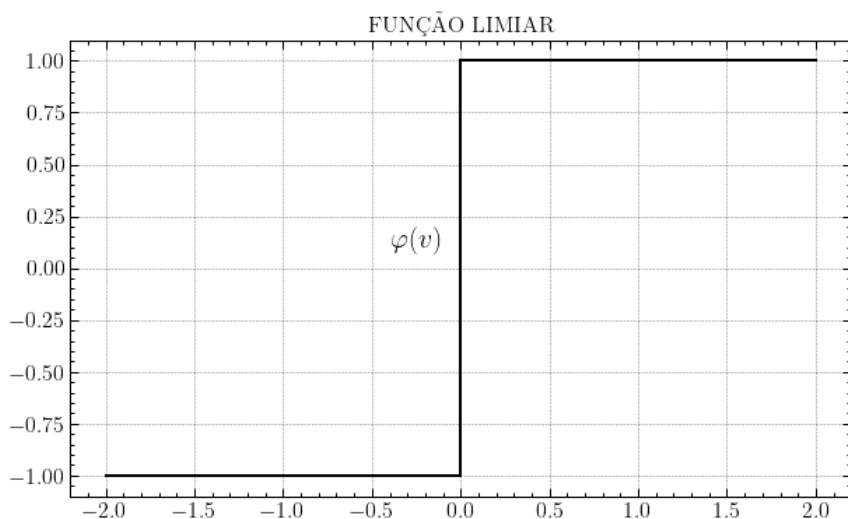
A saída de um neurônio é determinada pela função de ativação aplicada a  $v_k$ . A seguir serão apresentadas 3 das principais funções de ativação utilizadas na literatura.

### 3.2.1 FUNÇÃO DE LIMIAR

Também chamada de função de degrau binário, ela possui a seguinte característica:

$$\varphi(v) = \begin{cases} 1, & \text{se } v \geq 0 \\ 0, & \text{se } v < 0 \end{cases} \quad (3.3)$$

Figura 3.2 – Gráfico da função de ativação limiar.



Fonte: Autor.

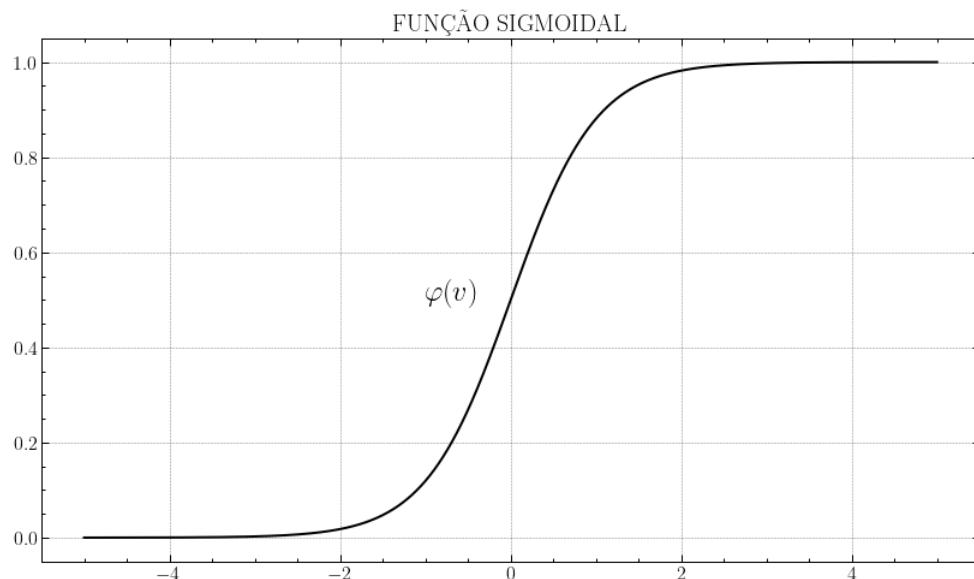
### 3.2.2 FUNÇÃO SIGMOIDE

É o tipo de função de ativação mais utilizada em redes neurais, apresenta um balanço entre comportamento linear e não-linear e é uma função diferenciável, característica importante para a aplicação em redes neurais [11]. Pode ser definida, matematicamente, da seguinte forma:

$$\varphi: \mathbb{R} \rightarrow [0, 1]$$

$$\varphi(v) = \frac{1}{1 + e^{-v}} \quad (3.4)$$

Figura 3.3 – Gráfico da função de ativação sigmoide.



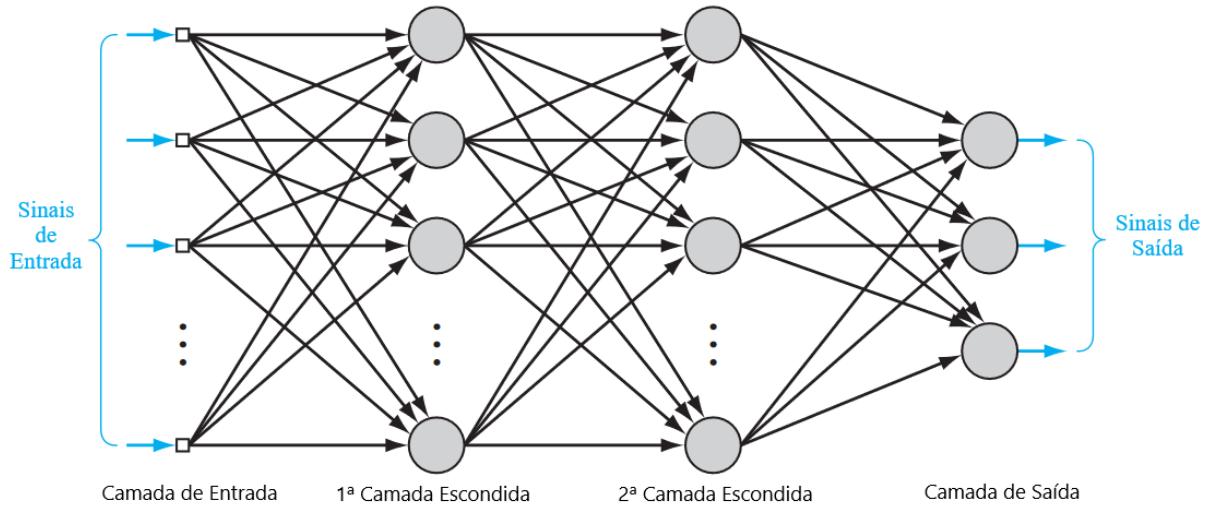
Fonte: Autor.

## 3.3 PERCEPTRON DE MÚLTIPLAS CAMADAS

A figura 3.4 representa uma rede neural de múltiplas camadas, também chamada de perceptron de múltiplas camadas. Nela estão representados a disposição dos neurônios em camadas e alguns sinais importantes para o entendimento de seu funcionamento, são eles:

- Sinal de Entrada
  - É entendido como o ponto de partida do algoritmo que se propaga de maneira direta, neurônio a neurônio, pela rede até a camada de saída;
- Sinal de Saída
  - É o resultado final de todas as operações e funções aplicadas ao sinal de entrada.

Figura 3.4 – Exemplo de uma rede neural com duas camadas escondidas.



Fonte: Adaptada [11].

Já a figura 3.5, por sua vez, representa o fluxo de dois sinais importantes que circulam na rede neural, são eles:

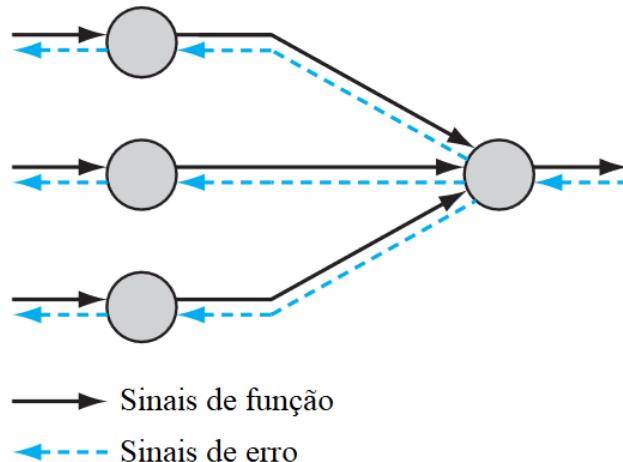
- Sinais de função
  - É um sinal de entrada ou estímulo aplicado na entrada de um neurônio ou da rede. Se propaga de maneira direta e emerge na saída como um sinal de saída. Recebe essa nomenclatura pois é presumido que seja uma função da saída da rede e a cada neurônio que passa o sinal é calculado como uma função das entradas e pesos associados a esse neurônio;
- Sinais de erro
  - Este sinal tem sua origem na saída da rede e se propaga na direção inversa (oposta aos sinais de função e de camada em camada). Recebe esse nome pois é responsável por carregar um valor resultado de um cálculo que envolve uma função de erro.

Além disso, cada neurônio da rede neural que pertença a camada de saída ou que seja uma unidade escondida é projetado para realizar duas operações:

- Encaminhar um sinal de função recebido em sua entrada para sua saída, expresso como uma função não linear dos sinais de entrada e seus pesos sinápticos associados;

- Computar uma estimativa do vetor gradiente (superfície de erro) em relação aos pesos sinápticos conectados às suas entradas, necessário para ocorrer a retropropagação do erro na rede.

Figura 3.5 - Ilustração das direções do fluxo de dois sinais importantes em uma rede neural: propagação direta de sinais de função e retropropagação de sinais de erro.



Fonte: Adaptada [11].

### 3.4 APRENDIZAGEM EM LOTE

Quando se fala de aprendizagem relacionada a uma rede neural refere-se, na verdade, a capacidade da rede de atualizar os pesos sinápticos das entradas de seus neurônios de acordo com os sinais de erro propagados, e com o objetivo de garantir uma saída ou resposta cada vez mais ajustada, isto é, mais próxima daquela desejada.

Existem diferentes formas de realizar o treinamento de uma rede neural, uma delas é chamada de aprendizagem em lote. Nesse caso, os ajustes aos pesos sinápticos da rede neural são realizados após a apresentação de  $N$  amostras de um conjunto de treinamento  $\mathcal{T}$ , o que constitui uma época. Assim, os ajustes dos erros sinápticos são feitos de época em época.

### 3.5 ALGORITMO DE RETROPROPAGAÇÃO DO ERRO

O algoritmo de retropropagação do erro tem como objetivo principal ajustar os pesos sinápticos e os *bias* associados a todos os neurônios de uma rede neural. O propósito dessa seção não é demonstrar o algoritmo, mas resumir-lo para que seu processo iterativo seja

suficientemente compreendido de modo a facilitar sua implementação. Para mais informações sobre o algoritmo recomenda-se a leitura de [11].

De modo geral, o algoritmo de retropropagação de erro utiliza o método de descida em gradiente, que consiste no cálculo do gradiente da função de custo e na atualização dos pesos sinápticos da rede neural na direção oposta do valor calculado, já que este aponta para o sentido de máximo crescimento da função e objetivo é minimizar a função de erro. Assim, a correção aplicada aos pesos sinápticos  $\Delta w_{kj}$  da camada  $l$  conectando o neurônio  $i$  ao neurônio  $j$  em determinada iteração  $n$  é definida da seguinte forma:

$$\begin{pmatrix} \text{Correção} \\ \text{nos pesos} \\ \Delta w_{ji}^{(l)}(n) \end{pmatrix} = \begin{pmatrix} \text{taxa de} \\ \eta \\ \text{aprendizado} \end{pmatrix} \times \begin{pmatrix} \text{gradiente} \\ \text{local} \\ \delta_j^{(l)}(n) \end{pmatrix} \times \begin{pmatrix} \text{Sinal de} \\ \text{entrada do} \\ \text{neurônio } j \\ y_i^{(l-1)}(n) \end{pmatrix} \quad (3.5)$$

Onde  $v_j(n)$  já foi definido na equação (3.1) e o gradiente local  $\delta_j^{(l)}(n)$  depende da função de função de erro  $e_j(n)$ . Também chamada de função de custo, é definida durante o processo de treinamento da rede neural e pode possuir diferentes formas, a depender do objetivo do projeto, preferência do desenvolvedor ou necessidade da rede. Para este trabalho, será utilizada a *binary cross-entropy loss* ou perda binária de entropia cruzada. Ela é responsável por computar o logaritmo da perda probabilística entre classificações previstas e verdadeiras. Isso significa que, se a probabilidade prevista para uma variável  $x$  pertencer a uma classe  $C_1$ , for alta e a variável pertencer, de fato, a  $C_1$ , então a perda computada terá um valor baixo. Por outro lado, caso essa variável  $x$  não pertencer a essa classe  $C_1$ , a perda computada será alta. A função de perda binária de entropia cruzada pode ser escrita da seguinte forma para um problema de classificação binário:

$$e_j(n) = (d_j(n) - 1) \cdot \log(1 - p(d_j(n))) - d_j(n) \cdot \log(p(d_j(n))) \quad (3.6)$$

Com  $d_j(n)$  representando a resposta desejada (0 ou 1) e  $p(d_j(n))$  a probabilidade de a saída do neurônio pertencer a classe de  $d_j(n)$  na camada de saída da rede neural para um estímulo  $x(n)$  aplicado à camada de entrada.

Portanto, essa função de erro penaliza a previsão da rede de acordo com a probabilidade de pertencimento a cada classe. Se a probabilidade de pertencer a classe real for alta, então a perda será baixa. Caso contrário, será alta. Por fim, a atualização dos pesos sinápticos dos neurônios da rede pode ocorrer a cada iteração (*online learning*) ou a cada época (aprendizado em lote, ou *batch learning*). No caso do aprendizado por época, a cada iteração são somadas as

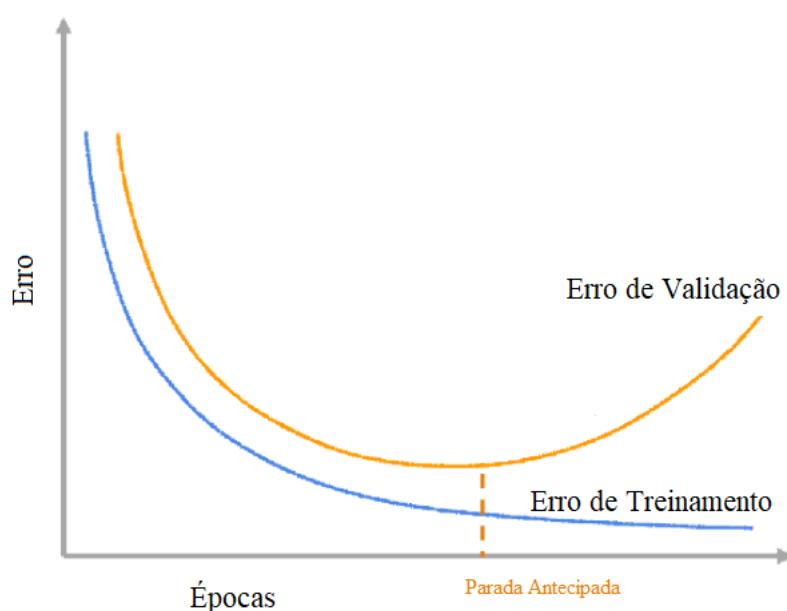
correções nos pesos  $\Delta w_{ji}^{(l)}(n)$  e somente após a apresentação das  $N$  amostras que constituem uma época ( $e$ ) que os pesos sinápticos da rede são atualizados, conforme a equação a seguir, onde  $\alpha$  é a constante de momento.

$$w_{ji}^{(l)}(e+1) = w_{ji}^{(l)}(e) + \alpha \sum_{n=1}^{N-1} \Delta w_{ji}^{(l)}(n) + \sum_{n=1}^N \Delta w_{ji}^{(l)}(n) \quad (3.7)$$

### 3.6 GENERALIZAÇÃO DA REDE

A generalização de uma rede neural pode ser definida como a sua capacidade de responder de maneira correta a um estímulo nunca visto anteriormente durante treinamento. Na prática isso quer dizer que, na grande maioria dos casos, para treinar uma rede neural visando solucionar um problema específico com  $N$  amostras, não se deve apresentar todas as amostras durante o treinamento da rede. Isso porque é muito provável que ocorra o problema de ajuste excessivo, ou em inglês *overfitting*, onde a rede neural se adapta de maneira excelente aos dados de treinamento, mas quando apresentada a casos nunca vistos anteriormente, não responde da forma esperada. A figura 3.6 apresenta, graficamente, o problema de *overfitting* durante o treinamento por época. O ideal, na prática, é encontrar o ponto em que o erro de validação seja o menor possível.

Figura 3.6 – Problema do ajuste excessivo no treinamento de redes neurais.



Fonte: Autor.

Detectar o *overfitting* é uma tarefa relativamente simples, evitá-lo de forma eficiente que é complexo, já que nem sempre consegue-se atingir o mínimo global ou generalizar os dados de maneira eficaz. Todavia, existem algumas técnicas que podem ser implementadas para evitar esse problema, como:

- Tratamento dos dados
  - A filtragem e balanceamento dos dados pode ajudar a evitar que o modelo fique viciado em algumas características particulares.
- Validação cruzada
  - Consiste na divisão do conjunto de treinamento em um conjunto de estimativa e outro conjunto de validação para treinar diversos modelos e selecionar aquele que melhor performa para os dados de validação e testá-lo para os dados de teste [11].
- Parada antecipada
  - Consiste em realizar o treinamento tradicional da rede e verificar, a cada iteração de treinamento ou época, o desempenho da rede para os dados de validação. Conforme o desempenho da rede melhorar, o treinamento é continuado. Contudo, no momento em que o desempenho apresentar uma piora para esse resultado, é feita a parada do treinamento e os pesos sinápticos e vieses anteriores são salvos.

## 3.7 MÉTRICAS DE AVALIAÇÃO DA REDE

Após a realização do treinamento de uma rede neural, deve-se avaliar seu desempenho apresentando novas amostras (não vistas durante o treinamento) e computando seus resultados. De posse das respostas, existem algumas técnicas para realizar a avaliação da performance dessa rede de classificação.

### 3.7.1 MATRIZ DE CONFUSÃO

A matriz de confusão é uma técnica de visualização dos resultados com grande relevância, pois permite analisar de forma rápida o desempenho de um algoritmo. Cada coluna dessa matriz representa as instâncias em uma classe predita, enquanto cada linha as instâncias de uma classe real. O nome se dá pelo fato de permitir rápida investigação sobre a confusão entre duas classes previstas. A tabela 3.1 representa o exemplo de uma matriz de confusão

legendada. Cada valor, dentro de uma matriz de confusão, possui um nome com um significado, são eles:

- Verdadeiro Positivo (VP): quando a predição do modelo é positiva e o *ground-truth*, ou a verdade de campo (saída desejada), também;
- Verdadeiro Negativo (VN): quando a predição do modelo é negativa e o *ground-truth* também;
- Falso Positivo (FP): quando a predição do modelo é positiva, enquanto o *ground-truth* é negativo;
- Falso Negativo (FN): quando a predição do modelo é negativa, enquanto o *ground-truth* é positivo.

Tabela 3.1 - Exemplo de uma matriz de confusão

		Predição	
		0	1
Real	0	Verdadeiro Negativo (VN)	Falso Positivo (FP)
	1	Falso Negativo (FN)	Verdadeiro Positivo (VP)

Fonte: Autor.

### 3.7.2 ACURÁCIA

A acurácia pode ser definida como a relação entre as amostras preditas corretamente e todas as predições realizadas.

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN} \quad (3.8)$$

Seu resultado pode ser enganoso ao lidar com dados de classes desbalanceadas, isto é, onde a quantidade de elementos por classe não é aproximadamente equivalente. Isso porque o treinamento da rede acaba colocando mais peso no aprendizado da classe com maior número de amostras.

### 3.7.3 PRECISÃO

A precisão pode ser definida como a relação entre quantidade de amostras preditas corretamente de determinada classe e a quantidade de previsões realizadas dessa classe.

$$Precisão = \frac{VP}{VP + FP} \quad (3.9)$$

Assim, é possível afirmar que quanto maior a quantidade de FP, ou seja, classificações erradas de uma amostra como pertencente a essa classe, menor a precisão.

### 3.7.4 RECALL

O *recall*, também chamado de sensibilidade, pode ser definido como a probabilidade de determinada amostra de uma classe X ser corretamente predita como tal. Na prática, indica a relação entre amostras da classe X corretamente preditas e todos as amostras pertencentes a essa classe.

$$Recall = \frac{VP}{VP + FN} \quad (3.10)$$

É possível perceber que a sensibilidade é inversamente proporcional à quantidade de falsos negativos, isto é, a quantidade de amostras não identificadas pela rede. Vale a pena ressaltar, também, a diferença entre recall e precisão. O modelo desenvolvido pode obter uma precisão muito alta, mas de nada adianta se não obtiver, também, uma sensibilidade alta. Isso porque estará deixando de reconhecer muitas amostras como pertencentes a classe de interesse. Com isso em mente, dependendo do problema em questão, a sensibilidade da rede neural pode ser mais relevante que a precisão para avaliar o seu desempenho.

## 4 PROCESSO DE DESENVOLVIMENTO DE UM ALGORITMO DE APRENDIZADO DE MÁQUINA

Durante a última década, o aprendizado de máquina passou por uma ampla democratização. Inúmeras palestras, livros, tutoriais e artigos já foram publicados relacionados ao tema. Embora aspectos técnicos sobre a construção e otimização de modelos sejam bem documentados e detalhados, essas fontes carecem de informações sobre como, de fato, desenvolver soluções de aprendizado de máquina em um contexto empresarial. Isso porque a construção e otimização do modelo – assunto muito bem documentado – é, na realidade, uma pequena parcela do processo de desenvolvimento da solução a ser implementada.

O desenvolvimento, implementação e gestão de uma solução de aprendizado de máquina geralmente segue um padrão. No entanto, as metodologias de desenvolvimento de projetos existentes não se aplicam devido à principal característica de qualquer solução de *machine learning* de ser orientada a dados. O aprendizado é derivado dos dados disponíveis, e não de código de programação. Assim, as abordagens e metodologias de desenvolvimento mais corretas originam-se de necessidades centradas em dados e resultam em projetos que se concentram nas etapas de descoberta, aquisição e filtragem desses dados.

Assim, apesar da democratização do aprendizado de máquina, o processo de desenvolvimento de um modelo desta natureza ainda é uma atividade muito nova e pode parecer intimidante para muitos profissionais ou organizações. Mesmo para aqueles com experiência na área, construir um modelo requer diligência, experimentação e criatividade. Nesse contexto, será apresentada, a seguir, uma metodologia que pode ser aplicada na construção de projetos de aprendizado de máquina orientados a dados.

#### **4.1 DEFINIÇÃO DO PROBLEMA E DO SUCESSO**

A primeira etapa de qualquer projeto, seja ele relacionado ou não com aprendizado de máquina, é a definição de seu objetivo. Compreender o que se está tentando resolver, restrições e requisitos é fundamental. É necessário entender o problema antes de tentar solucioná-lo.

O objetivo desta etapa é converter todo esse conhecimento e compreensão adquirida na definição de um problema adequado para solução com aprendizado de máquina. Também é trivial definir quais os resultados esperados, isto é, determinar as metas que se deseja atingir com o modelo e o que se espera dele. Alguns questionamentos que podem ser feitos para auxiliar esse processo são:

- Qual é o objetivo empresarial que requer uma solução de aprendizado de máquina?
- Quais são as características do problema que se está tentando resolver? É um problema de regressão, classificação ou clusterização?
- Qual a heurística aplicada atualmente na solução desse problema e o quanto melhor espera-se que o modelo seja?
- Quais são os critérios de sucesso definidos para o projeto?

Em projetos profissionais e empresariais, os objetivos devem ser relacionados aos objetivos de negócio, e não apenas de aprendizado de máquina. Assim, especificar e quantificar as metas do projeto que se desejar desenvolver auxiliará na quantificação e na medida de um retorno de investimento - que determinará a viabilidade do projeto.

Por fim, após determinar o problema a ser solucionado e o sucesso esperado, ainda é nesta etapa que se determina a viabilidade do projeto de aprendizado de máquina que se pretende implementar. Para isso, deve-se avaliar a viabilidade de negócio, dos dados e de implementação. Com uma clara definição do problema e um retorno de investimento suficiente, pode-se dizer que o projeto é viável no âmbito de negócio. No que tange aos dados, o projeto é viável se existem dados em quantidade suficiente ou se há possibilidade de coleta. No contexto da implementação, entende-se como apto se há tecnologia e infraestrutura suficientes para o desenvolvimento e implementação do projeto.

## **4.2 ENTENDIMENTO E IDENTIFICAÇÃO DOS DADOS**

Um modelo de aprendizado de máquina é construído aprendendo e generalizando a partir de dados de treinamento e, em seguida, aplicando esse conhecimento adquirido a novos dados, que nunca foram apresentados antes, para fazer previsões e cumprir seu propósito. Essas amostras de treinamento são resultado de um extenso processo de estudo, análise, filtragem e preparação dos dados disponíveis ou *raw data* – dados crus.

Nessa perspectiva, é necessário identificar as necessidades de dados para o desenvolvimento da solução e determinar se estão na forma adequada para atendê-las. O foco deve ser na identificação desses dados, de requisitos, da qualidade e de características potencialmente interessantes que valem uma maior investigação.

A compreensão e identificação desses requisitos e particularidades vão ajudar tanto no controle da quantidade e qualidade dos dados recebidos, quanto no entendimento das características necessárias para garantir que o modelo funcione como esperado.

## **4.3 PREPARAÇÃO DOS DADOS**

Uma vez que já se tenha identificado as características ótimas para seu modelo é necessário transformar os dados brutos em um formato que possa ser utilizado para treinar seu modelo de aprendizado de máquina. O foco passa a ser em atividades centradas na modelagem dos dados com o objetivo de construir um conjunto de amostras a ser usado para treinamento e

validação. Esse conjunto de tarefas pode ser entendido como *feature engineering*, e consiste, por exemplo, na filtragem, agregação, aumento, rotulagem, normalização e transformação – bem como qualquer outra atividade envolvendo o processamento de dados estruturados, semiestruturados ou não estruturados - dos dados brutos em amostras organizadas e prontas para alimentar seu modelo de aprendizado.

Tarefas de preparação e filtragem de dados podem levar um tempo relativamente alto. Pesquisas de desenvolvedores de IA e cientistas de dados revelam que essa etapa pode representar até 80% do tempo de um projeto [12]. Na prática, como os algoritmos de aprendizado de máquina aprendem dos dados a ele fornecidos, vale a pena investir uma boa quantidade de tempo para garantir que os dados entregues ao modelo sejam da melhor qualidade possível.

#### **4.4 DETERMINAÇÃO DAS CARACTERÍSTICAS E TREINAMENTO DO MODELO**

Uma vez que os dados estão em uma forma utilizável e o problema a ser resolvido é conhecido, finalmente é hora de iniciar a etapa de treinamento do modelo, que aprenderá a partir dos dados de qualidade que foram preparados anteriormente e de uma vasta quantidade de técnicas e algoritmos.

Esse treinamento consistirá em um processo iterativo de definição e teste de diversas características diferentes do modelo, como a seleção do algoritmo a ser utilizado, configuração e ajustes dos hiperparâmetros, treinamento, validação e otimização do modelo. Para isso, deve-se realizar as seguintes ações:

1. Selecionar o algoritmo certo baseado nos requisitos de seus dados e no objetivo de aprendizado definido;
2. Configurar, ajustar e otimizar os hiperparâmetros para obter uma performance ideal do modelo;
3. Identificar as características dos dados que geram os melhores resultados;
4. Treinar diversos modelos diferentes para encontrar a melhor versão.

#### **4.5 AVALIAÇÃO DO DESEMPENHO E ESTABELECIMENTO DE METAS**

Partindo de uma perspectiva do aprendizado de máquina, a avaliação de um modelo constitui a apresentação de amostras nunca vistas antes, durante treinamento, e inclui o cálculo das métricas (acurácia, sensibilidade, precisão, dentre outros), da matriz de confusão, de

medidas de desempenho computacional e uma determinação final se o modelo satisfaz os objetivos de negócio estabelecidos no início do projeto.

A avaliação do algoritmo de aprendizado de máquina desenvolvido pode ser considerada a garantia de qualidade do modelo. Avaliar o desempenho da solução em relação as métricas determinam como o modelo irá performar em um ambiente de produção, isto é, no mundo real. Deve ser verificado, ainda, se o modelo desenvolvido atinge as expectativas e requisitos pré-determinados no início do desenvolvimento, pois as métricas, sozinhas, não definem se o modelo está pronto para ser implantado.

Finalmente, após executar todas as etapas descritas neste capítulo, a construção de um modelo de aprendizado de máquina chegará ao fim, restando apenas a implementação e supervisão da performance do modelo em um ambiente de produção. Vale ressaltar que, apesar da construção do modelo ter terminado, um algoritmo de *machine learning*, em sua essência, está sempre em constante evolução. Nesse sentido, após a implementação do modelo no mundo real, o desenvolvimento não deve cessar. À medida que novos dados são apresentados ao modelo, novas amostras de treinamento são adquiridas e em uma nova iteração de treinamento poderá evoluir e otimizar suas respostas.

Além disso, em muitos casos, a saída do modelo de aprendizado de máquina não é diretamente utilizada e necessita, ainda, de algumas técnicas de pós processamento. Na prática, o pós-processamento é responsável pela interpretação da resposta do modelo computacional e sua transposição para o meio comercial ou empresarial. Pode-se, ainda, utilizar-se dessa etapa para identificar e corrigir alguns erros ou casos específicos sem a necessidade de realizar um novo treinamento.

## **5 PROCESSO DE DESENVOLVIMENTO DA SOLUÇÃO DE APOIO À MANUTENÇÃO PREDITIVA**

O objetivo principal deste trabalho consiste no desenvolvimento de uma rede neural capaz de auxiliar e apoiar decisões na implementação de uma estratégia de manutenção preditiva. O sucesso, neste caso, é definido como a capacidade de indicar o momento em que um ativo entra em iminência de falha, e quanto mais cedo ocorrer a identificação, melhor. Para isso, como explicado nos capítulos 2 e 4, é necessário conhecer suficientemente os dados disponíveis para entender as necessidades dos ativos e as características que mais influenciam na falha.

## 5.1 TECNOLOGIAS UTILIZADAS PARA DESENVOLVIMENTO DO ALGORITMO

Para realizar a implementação computacional do que será proposto ao longo deste capítulo, será utilizada a linguagem de programação de código aberto *Python* em sua versão 3.6. Além disso, o ambiente de desenvolvimento escolhido, isto é, onde o código será desenvolvido, foi o *Google Colaboratory*, que consiste em uma plataforma em nuvem para aplicações em *Python*, disponível sem custos e especialmente otimizado para o desenvolvimento e treinamento de algoritmos de inteligência artificial e aprendizado de máquina, sendo este último a principal razão para essa escolha.

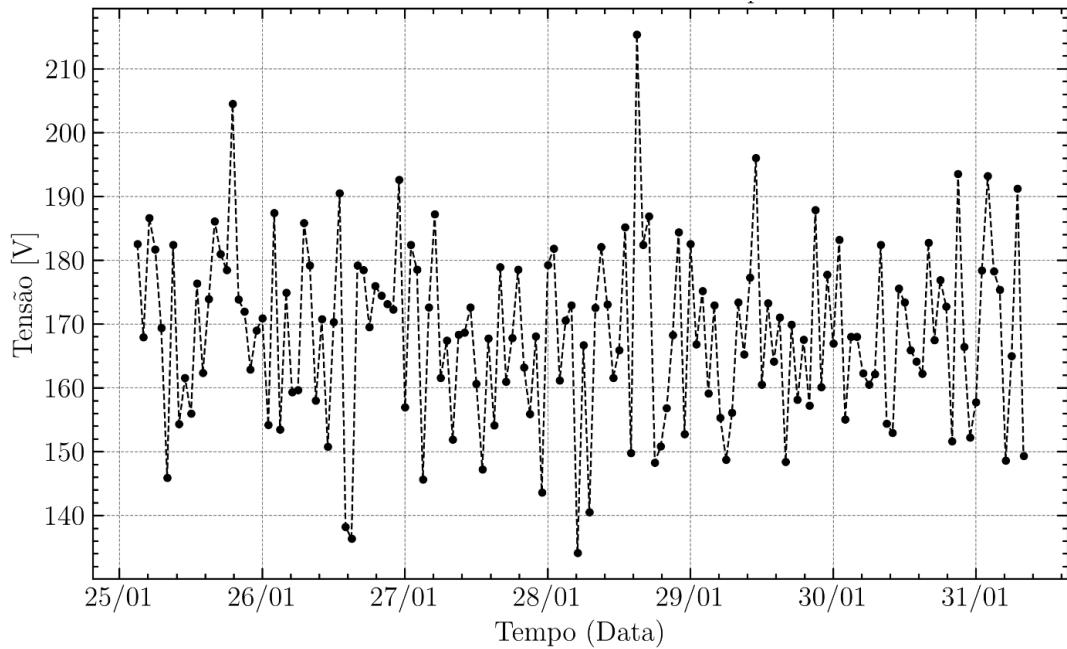
Sob a perspectiva algorítmica, foram utilizadas diversas bibliotecas disponíveis para o *Python* com destaque para duas: *Tensor Flow* e *Keras*. A primeira é uma interface para expressar algoritmos de aprendizado de máquina e um ambiente para executar esses algoritmos [13]. A segunda é responsável por fornecer a arquitetura do modelo, permitindo assim, a construção de redes neurais com múltiplas camadas e retropropagação do erro. Nesse modelo, cada neurônio possui seu peso sináptico, função de ativação, minimização de perdas e otimização, que podem ser definidos pelo usuário [14].

Ademais, para fins de referência, todo o algoritmo, lógica e dados utilizados para desenvolvimento da solução computacional de *machine learning* com uso de redes neurais resultado deste trabalho poderá ser encontrado em um repositório no Github do autor [15].

## 5.2 ANÁLISE DOS DADOS DISPONÍVEIS

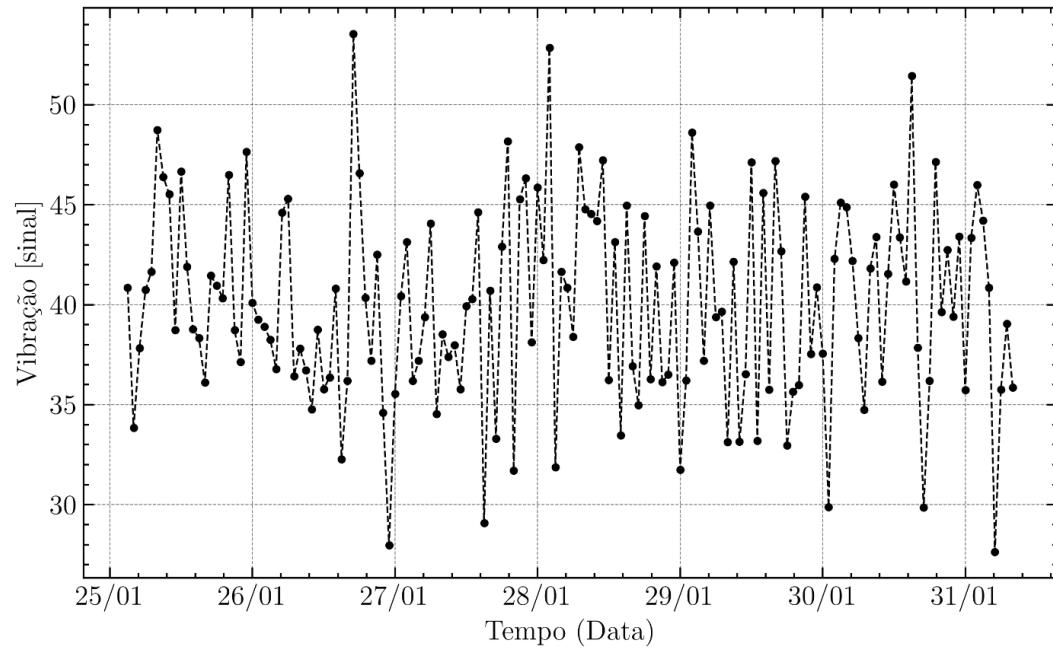
Os dados utilizados na elaboração da rede neural para apoio à manutenção preditiva foram disponibilizados pela Microsoft na plataforma Azure AI Gallery [16]. As amostras disponíveis constituem status de erro, falha e variáveis de processos, como vibração, tensão, pressão e rotação, medidas a cada hora de 100 máquinas de diferentes idades observadas durante um ano ao longo de 2015 até 2016. Nas figuras 5.1 até 5.4 estão representadas as amostras das variáveis de processo, enquanto nas figuras 5.5 e 5.6 as variáveis de erro e falha, de interesse para o modelo que será desenvolvido.

Figura 5.1 – Amostras da variação da medição de tensão em operação normal.



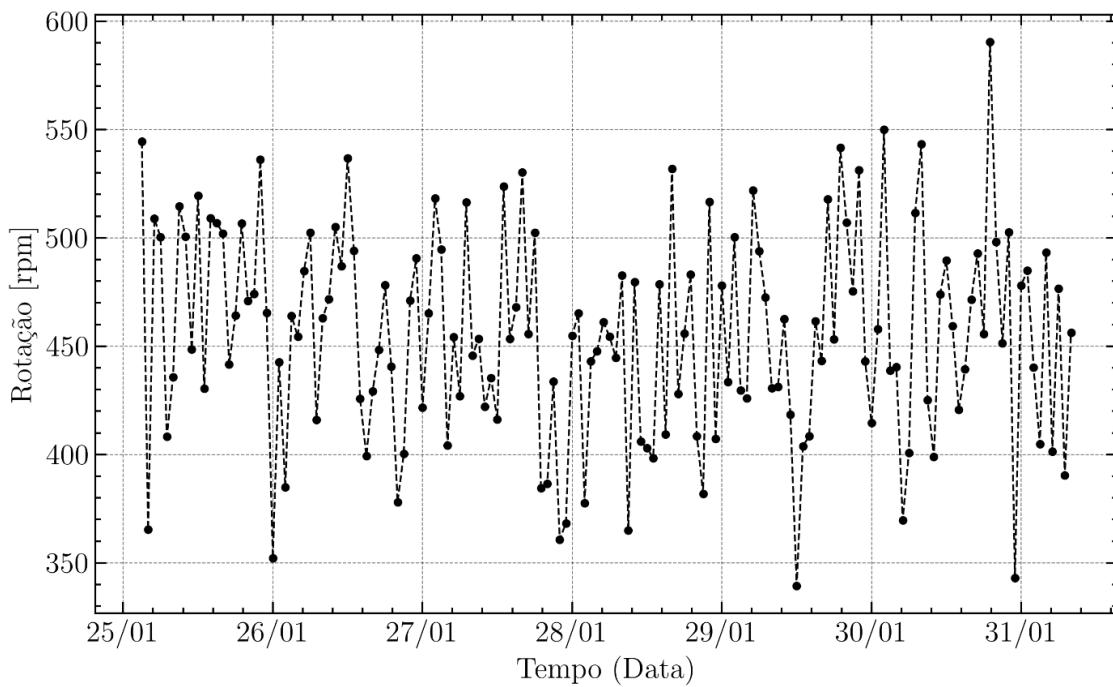
Fonte: Autor.

Figura 5.2 – Amostras da variação da medição de vibração em operação normal.



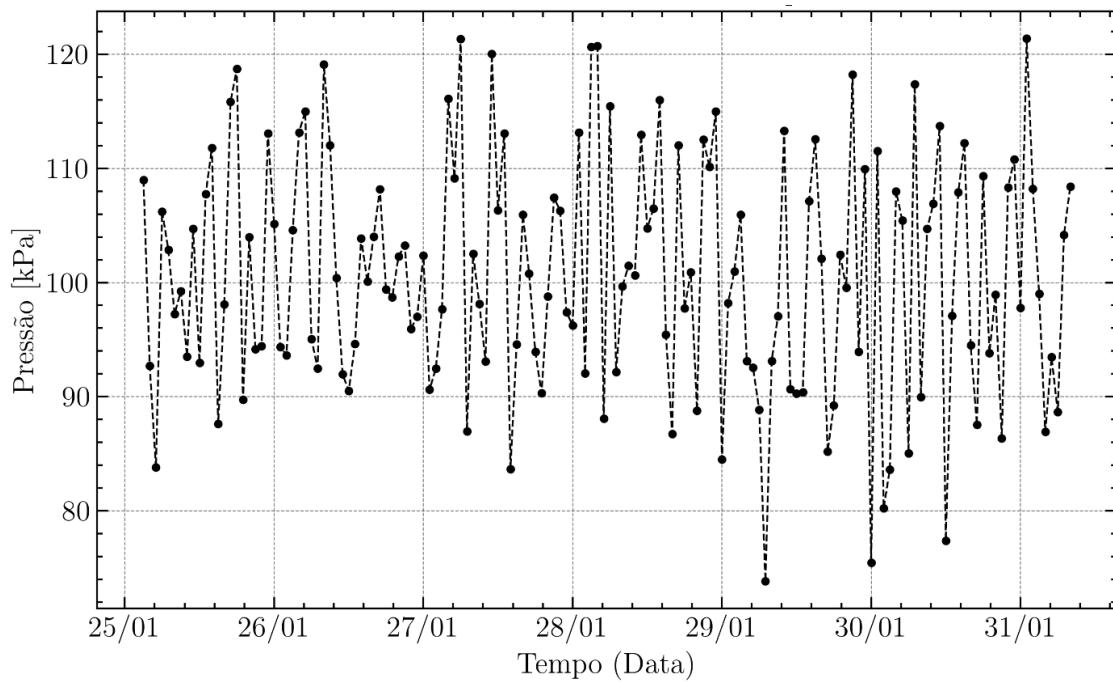
Fonte: Autor

Figura 5.3 – Amostras da variação da medição de rotação em operação normal.



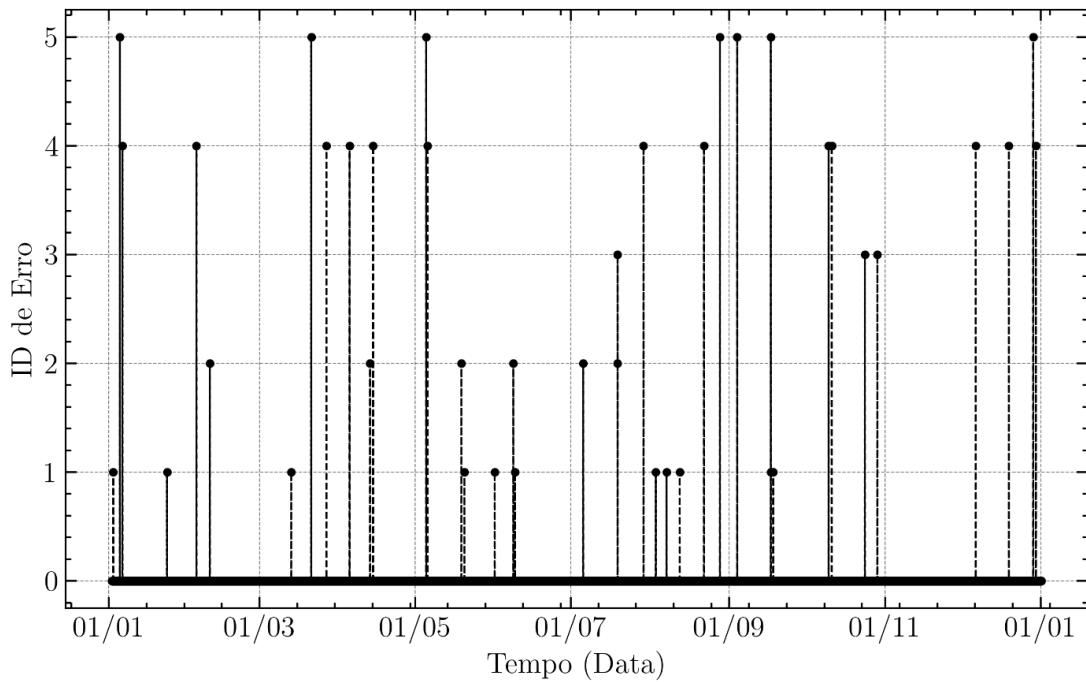
Fonte: Autor.

Figura 5.4 – Amostras da variação da medição da pressão em operação normal.



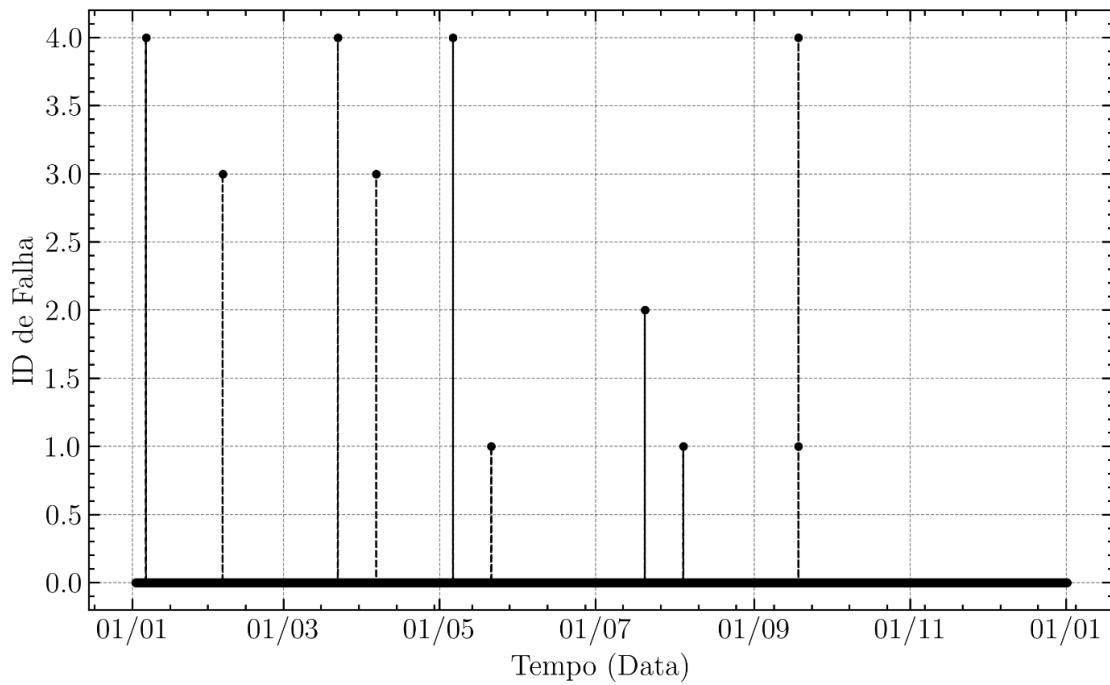
Fonte: Autor

Figura 5.5 – Amostra dos erros, onde 0 indica operação normal.



Fonte: Autor.

Figura 5.6 – Amostra das falhas, onde 0 indica operação normal.



Fonte: Autor.

Vale ressaltar que as amostras de erro consistem em problemas detectados na máquina supervisionada, mas que não geram uma falha, isto é, não é necessária qualquer intervenção e o ativo retorna ao funcionamento normal sozinho. Em contrapartida, os instantes de falha

constituem momentos em que é necessário realizar algum tipo de manutenção para garantir o retorno da operação normal.

A partir das amostras das variáveis supervisionadas dos ativos, é possível perceber que há uma variação das medições dentro de uma faixa de valores. Em vista disso, é provável que seja possível identificar um ponto de operação fora do ideal de acordo com as medições indicadas por essas variáveis, bem como um estado de transição entre operação normal e estado de falha.

Assim, é necessária uma análise mais profunda para entender o funcionamento dos ativos supervisionados e as alterações de comportamento sofridas na iminência da falha. Inicia-se, então, a tarefa inicial do desenvolvimento de qualquer algoritmo de *machine learning*, o *feature engineering*.

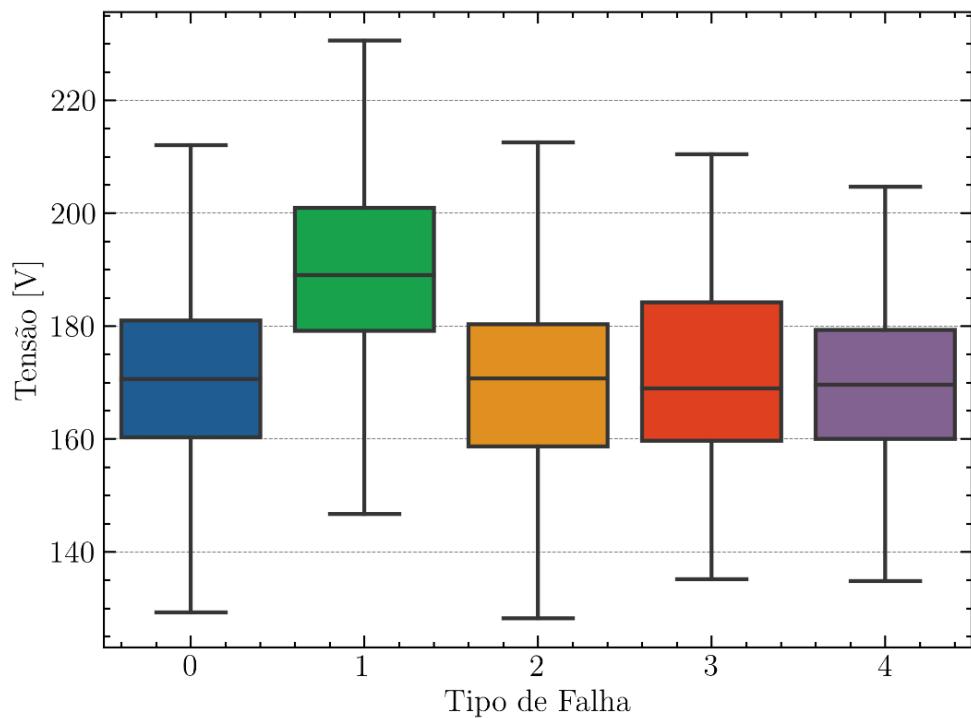
## 5.3 FEATURE ENGINEERING

### 5.3.1 VISUALIZAÇÃO DA RELAÇÃO ENTRE VARIÁVEIS

Como um dos objetivos da manutenção preditiva é prever o ponto de falha dos ativos supervisionados, deve-se verificar a relação existente entre as variáveis de processo e a variável que indica falha, aferidas no *dataset* utilizado. Nesse contexto, as figuras 5.7 até 5.10 apresentam as relações entre variáveis de processo e indicadores de falha em *boxplots* – gráfico onde a caixa representa a amplitude interquartílica e sua altura é definida pelo primeiro e terceiro quartis, contendo 50% das amostras. A linha horizontal interna à caixa indica a mediana da amostra, enquanto as duas semirretas verticais ligam os quartis Q3 e Q1 aos valores máximos e mínimos, respectivamente, delimitados pelas duas linhas horizontais externas.

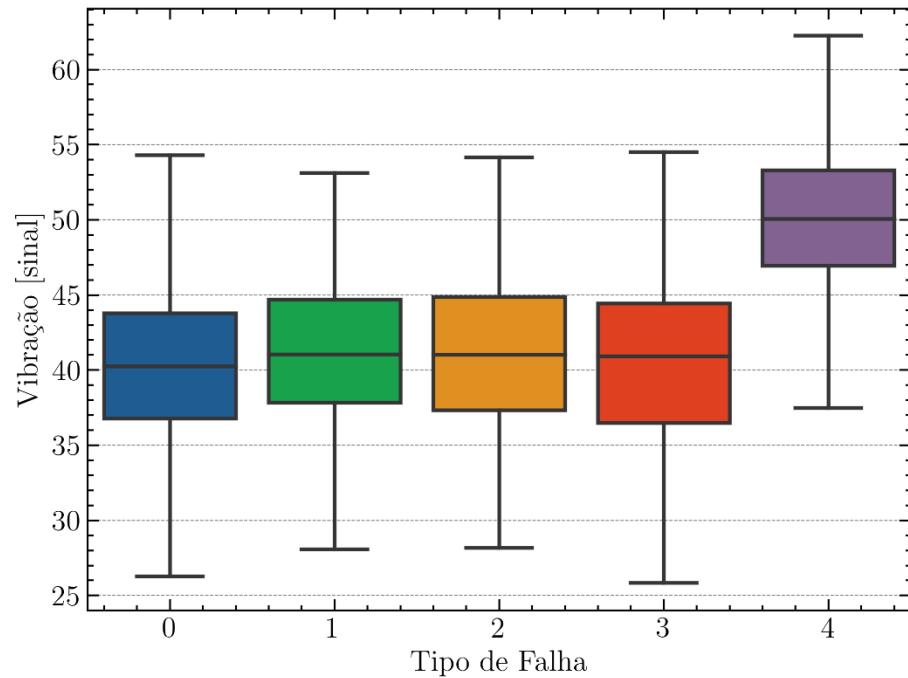
É possível observar que há uma relação evidente entre tipo de falha e uma determinada variável de processo. Da figura 5.7, observa-se que a tensão em condições de operação normais oscila entre 160 V e 180 V. Porém, para o tipo de falha 1, ocorre um aumento da oscilação da tensão da ordem de 20 V, passando a operar entre 180 V e 200 V, em, aproximadamente 75% das amostras para esse tipo de falha.

Figura 5.7 – Boxplot da tensão em relação aos tipos de falha, com 0 indicando operação normal.



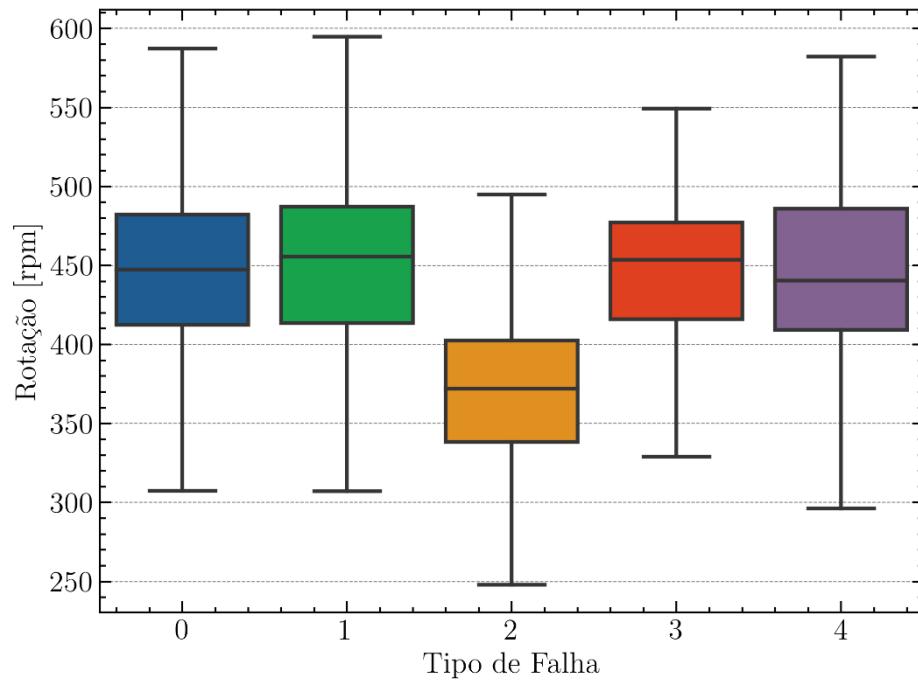
Fonte: Autor.

Figura 5.8 – Boxplot da vibração em relação aos tipos de falha, com 0 indicando operação normal.



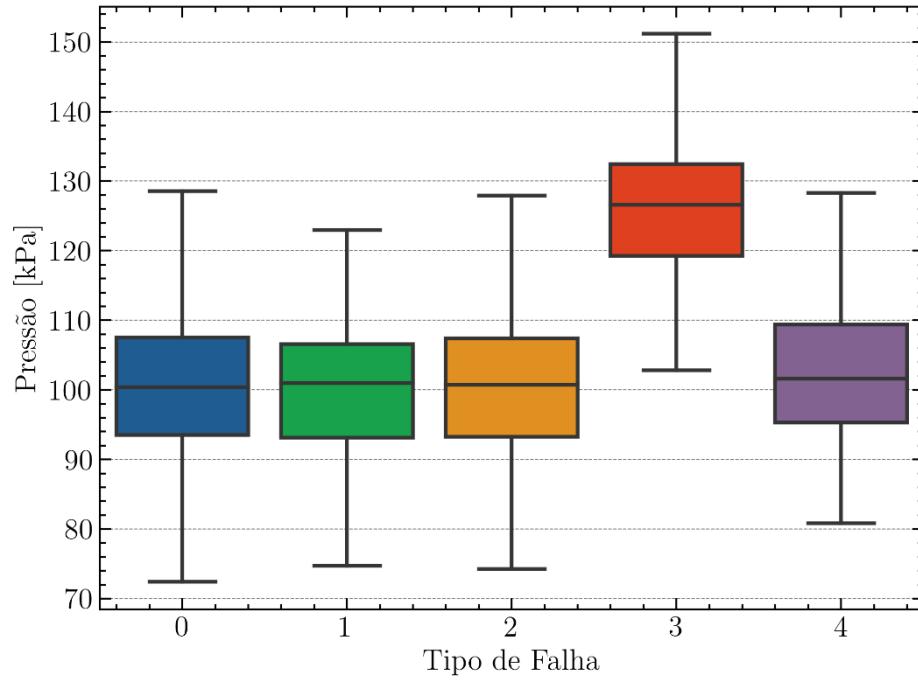
Fonte: Autor.

Figura 5.9 – Boxplot da rotação em relação aos tipos de falha, com 0 indicando operação normal.



Fonte: Autor.

Figura 5.10 – Boxplot da pressão em relação aos tipos de falha, com 0 indicando operação normal.



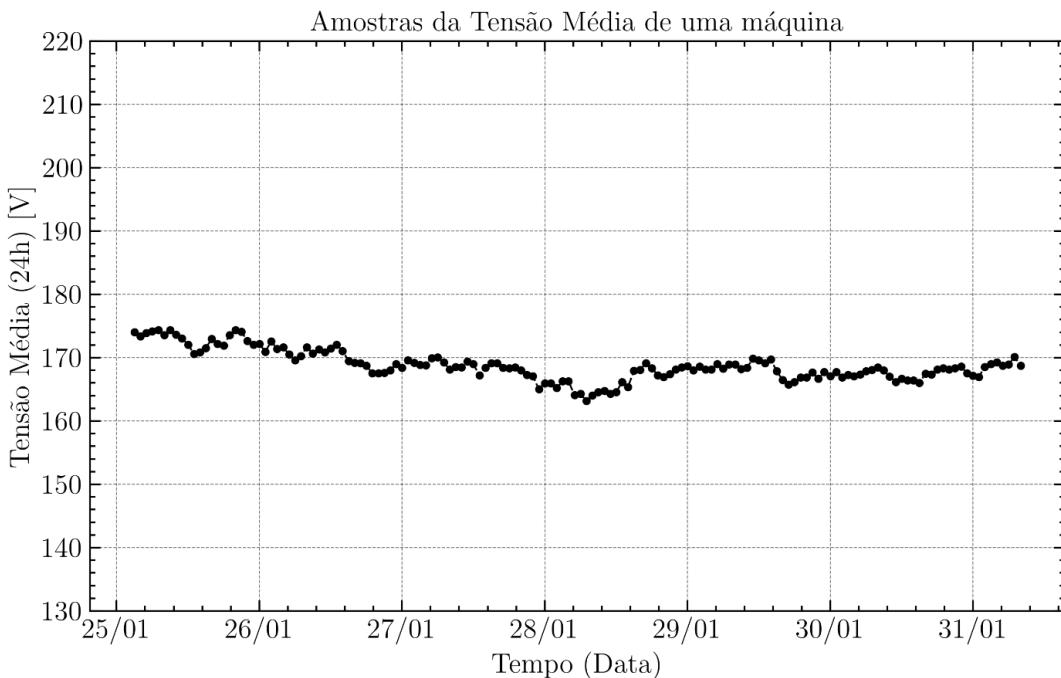
Fonte: Autor.

O mesmo se pode dizer para a figura 5.8, onde é evidente que em condições regulares os sinais de vibração transitam entre 35 e 45 unidades, mas para o tipo de falha 5, esse valor intervalo passa a ser de 47 até 53 unidades. Analogamente para a rotação, em operação usual

tem-se uma flutuação entre 410 e 480 rotações por minuto, já em condições de falha tipo 2, vê-se uma diminuição de aproximadamente 75 rotações por minuto. Semelhantemente, mas de modo mais acentuado, vê-se um acréscimo acentuado na pressão em instantes de operação com falha do tipo 3, de 100 kPa para 125 kPa, em média. É possível perceber, ainda, que as amostras possuem uma distribuição esparsa, com valores extremos mesmo em situações normais, o que pode ser prejudicial na identificação de padrões por parte da solução. Nessa perspectiva, pode ser entendido que as medições possuem um ruído que dificulta a identificação do status de operação.

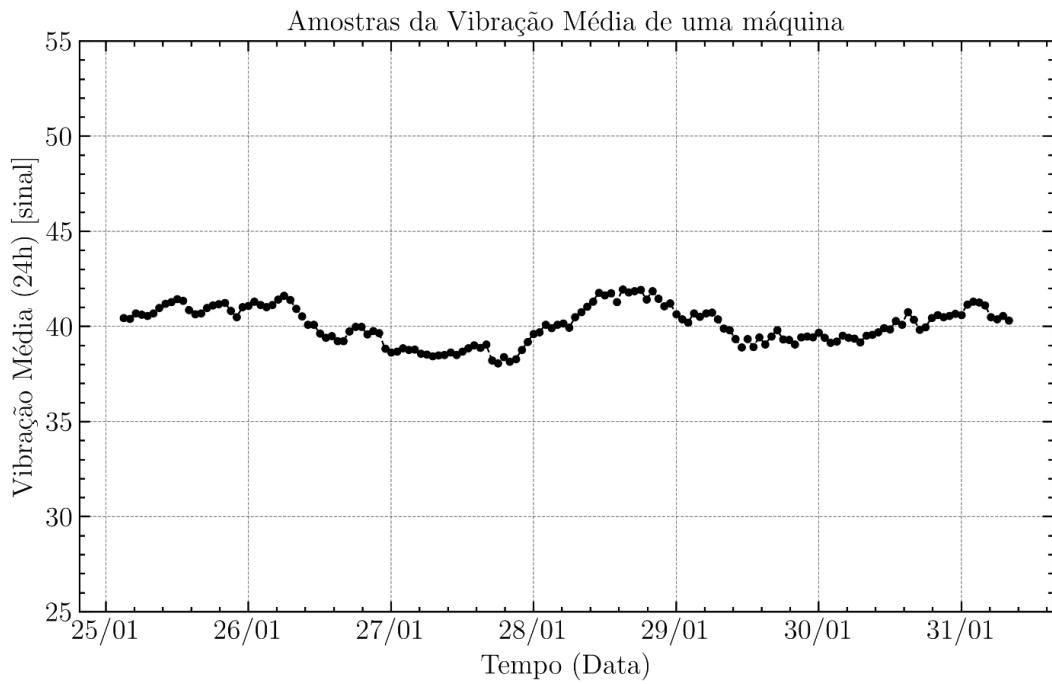
Uma estratégia comumente utilizada para reduzir esse ruído que as amostras apresentam é utilizar alguma medida de média para computar as medições em determinado instante, como a média móvel ou a média móvel exponencial. De modo geral, essa estratégia tem o objetivo de suavizar a variação das amostras e pode ter como consequência uma melhor definição dos modos de operação. As figuras 5.11 até 5.14 mostram essa suavização – no mesmo intervalo das figuras 5.1 até 5.4 para fins de comparação –, onde foi computada a média móvel de 24 períodos (no caso, um único período corresponde a uma única hora).

Figura 5.11 – Suavização das medidas da tensão com o cálculo da média móvel de 24 períodos (24 horas).



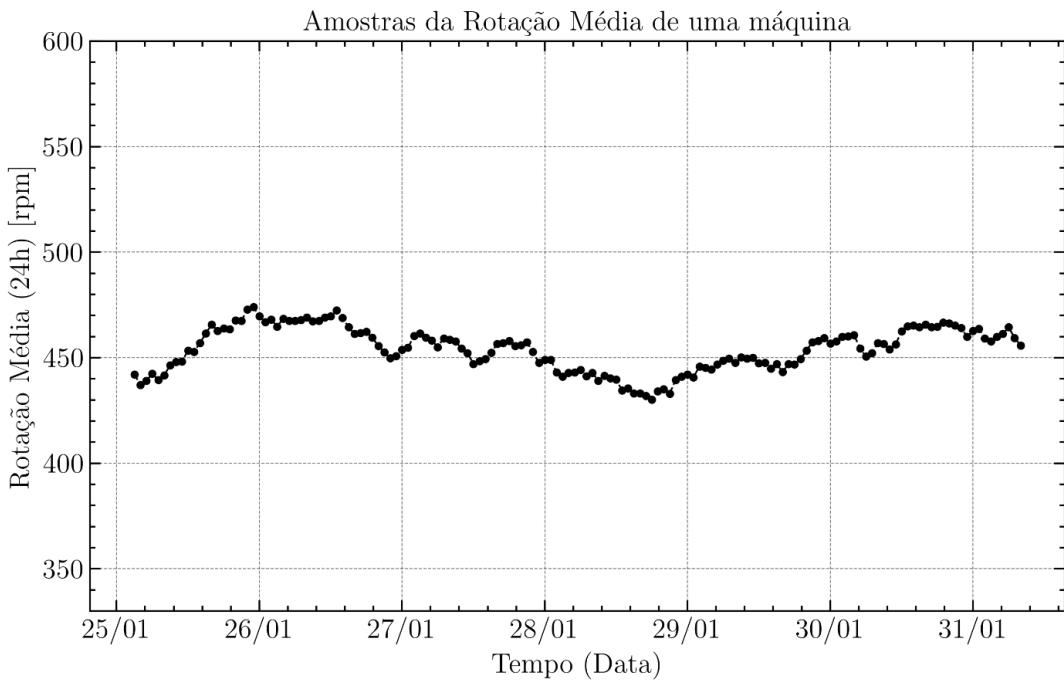
Fonte: Autor.

Figura 5.12 – Suavização das medidas de vibração, computando a média móvel de 24 períodos (24 horas).



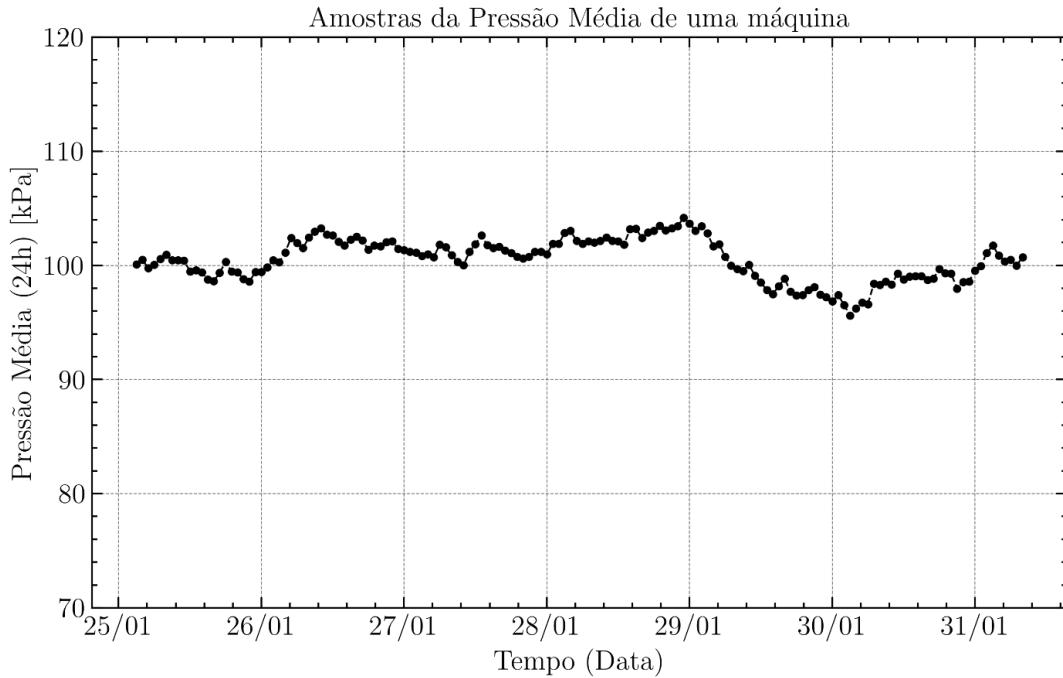
Fonte: Autor.

Figura 5.13 – Suavização das medidas de rotação, computando a média móvel de 24 períodos (24 horas).



Fonte: Autor.

Figura 5.14 – Suavização das medidas de pressão, computando a média móvel de 24 períodos (24 horas).



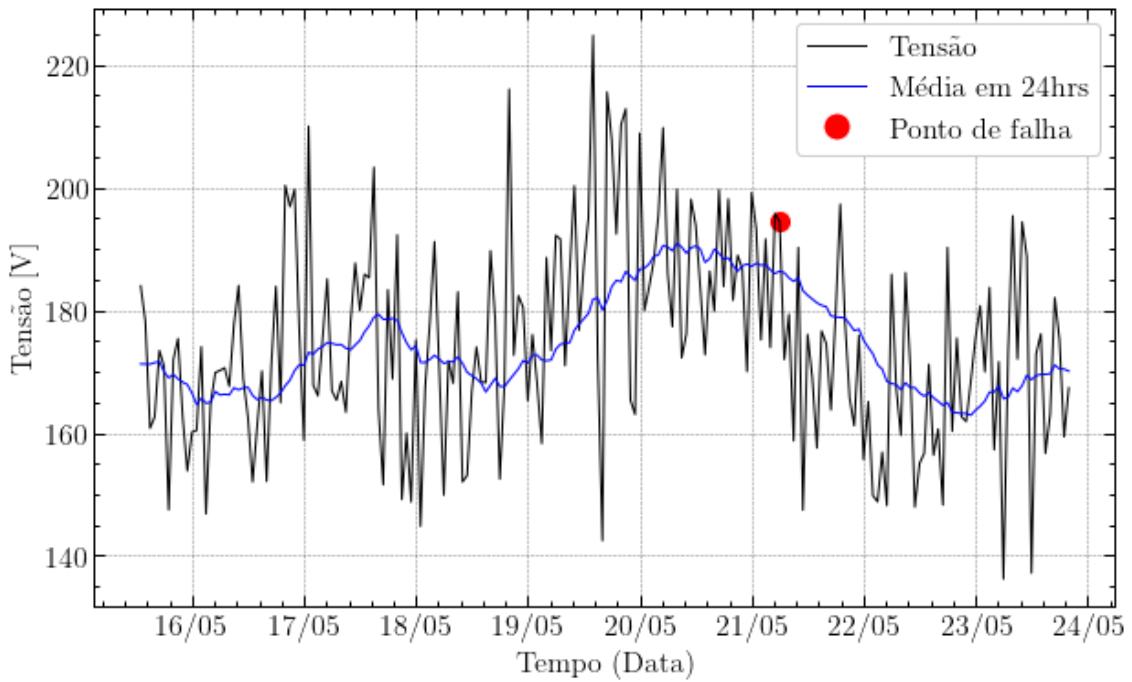
Fonte: Autor.

Nesse cenário, é possível enxergar uma relação entre as variáveis de processo e os instantes de falha, que pode ser atacada no desenvolvimento do modelo computacional de apoio à manutenção preditiva. O questionamento que deve ser feito, neste instante, passa a ser se essa relação ocorre após o evento de falha – como consequência –, ou se ocorre previamente com intervalo de tempo suficiente para identificação e disparo de alarme de aviso para gestores e engenheiros de manutenção.

### 5.3.2 IDENTIFICAÇÃO DO INTERVALO DE PREVISÃO

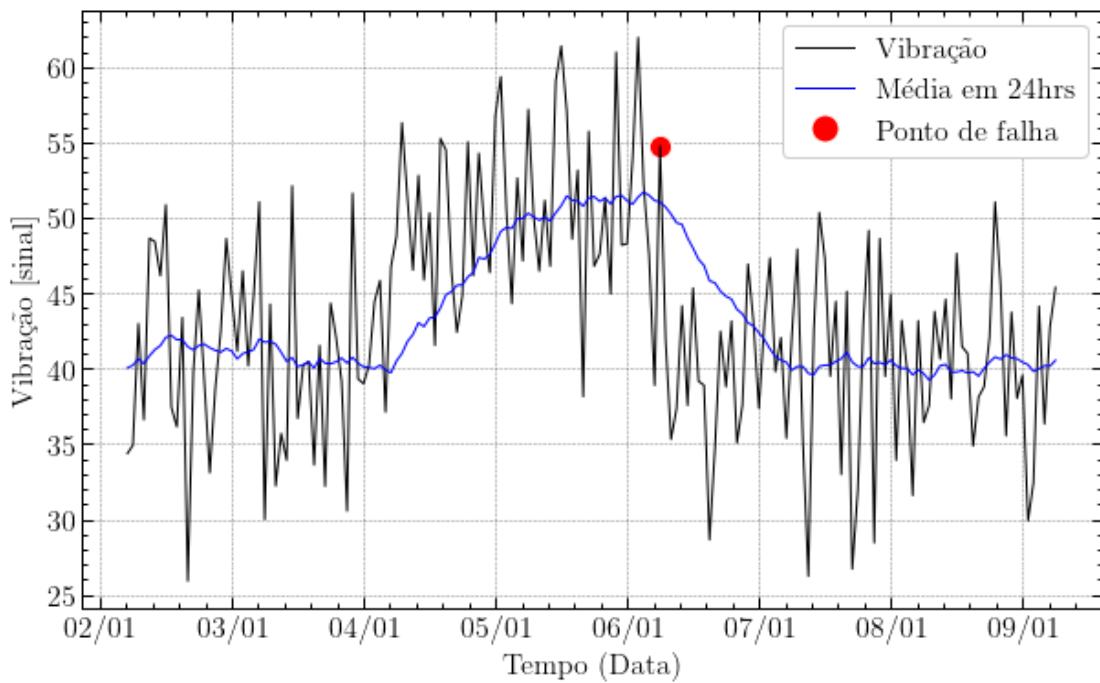
Para compreender o comportamento das variáveis de processo quando na iminência de falha e determinar se é possível desenvolver um modelo que, de acordo com essas variáveis, seja capaz de determinar esses momentos, é necessário visualizar as modificações que elas sofrem nesse período. Nesse contexto, as figuras 5.15 até 5.17 mostram o comportamento das variáveis de processo disponíveis, bem como suas médias móveis de 24 períodos nesse intervalo e o ponto exato de falha.

Figura 5.15 – Amostra do comportamento da tensão e sua média durante falha



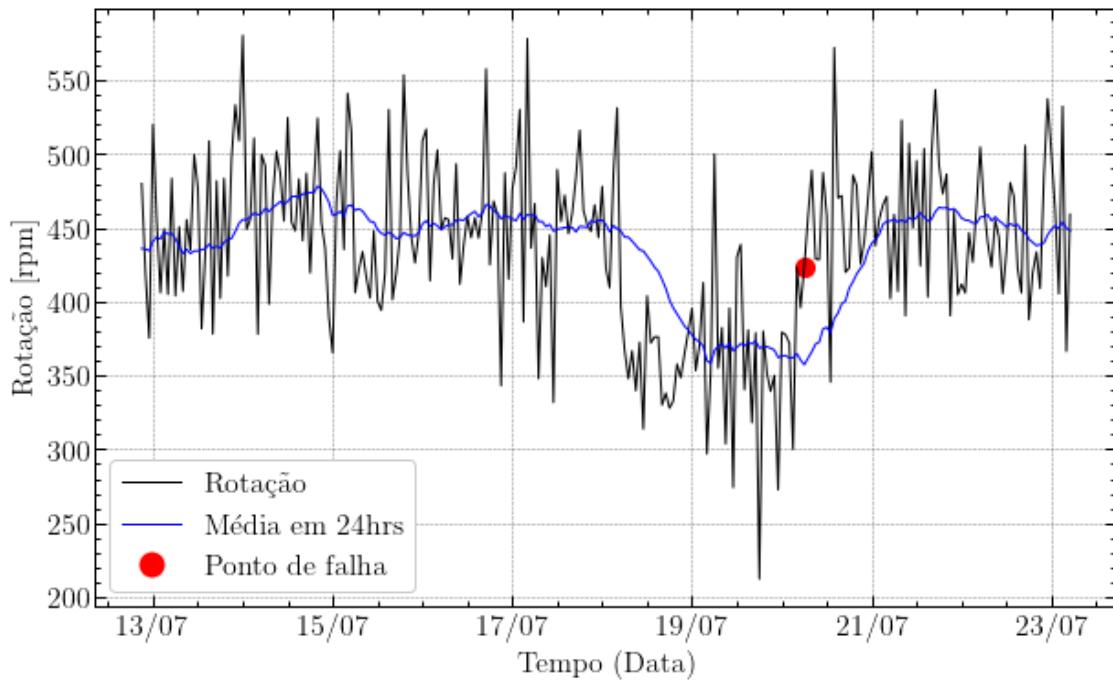
Fonte: Autor.

Figura 5.16 – Amostra do comportamento da vibração e sua média durante falha



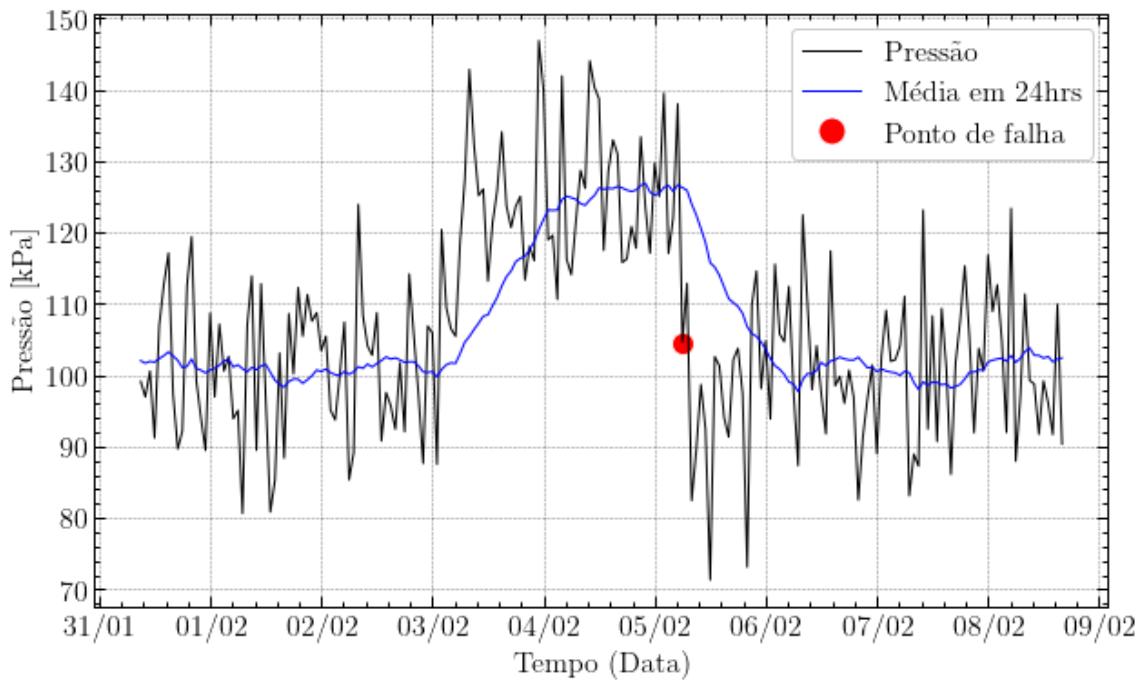
Fonte: Autor.

Figura 5.17 – Amostra do comportamento da rotação e sua média durante falha



Fonte: Autor.

Figura 5.18 – Amostra do comportamento da pressão e sua média durante falha



Fonte: Autor.

Pela análise das figuras 5.15 a 5.18, é possível perceber uma alteração considerável na distribuição das amostras durante a falha. Mais que isso, identifica-se que aproximadamente 48

horas antes do momento de falha, esses dados sofrem uma alteração significativa (da ordem de 10% até 20%) do seu valor médio.

Assim, observa-se um alvo para o modelo a ser desenvolvido por este trabalho. Conseguir detectar o início desse período de 48 horas antes da falha enquanto alerta aos engenheiros de manutenção da necessidade de intervenção.

Por outro lado, pode ocorrer que essas alterações identificadas sejam precursoras da falha, isto é, a falha somente acontece devido à operação do ativo fora da normalidade durante 48 horas e não ao contrário – como sinal de que falhará – como pensado anteriormente. Não cabe, neste trabalho, a identificação da razão do acontecimento da falha – causa ou consequência –, mas o resultado, para ambos os casos, será o mesmo. O modelo desenvolvido será responsável por alertar a provável falha da máquina ao identificar essas alterações.

### **5.3.3 DEFINIÇÃO DAS VARIÁVEIS DE ENTRADA**

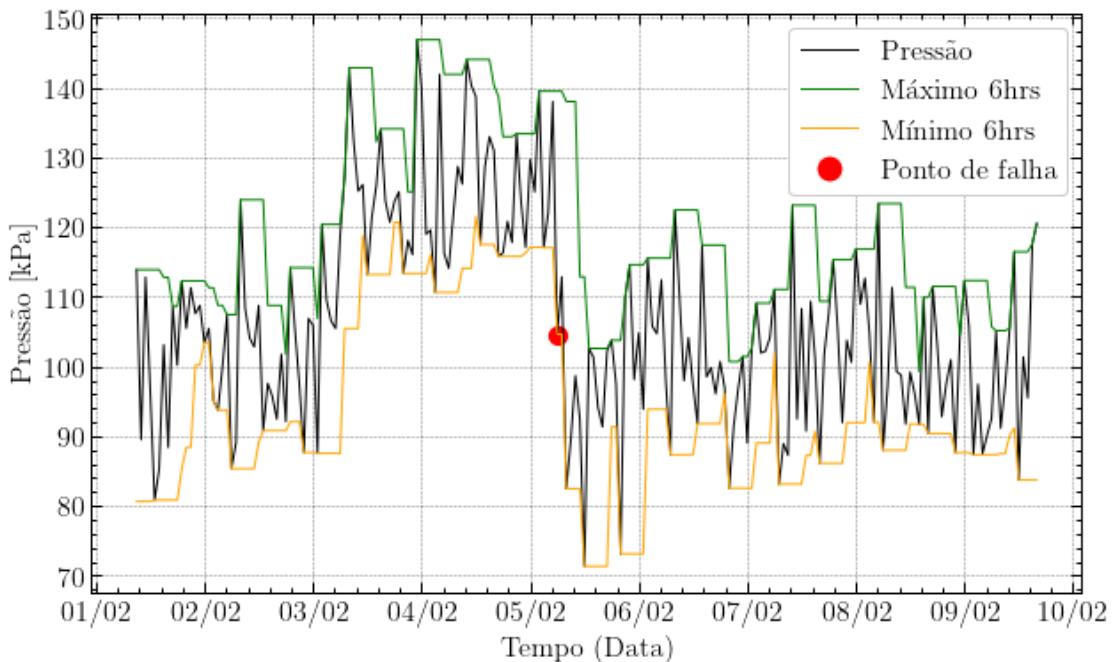
O próximo passo no desenvolvimento do algoritmo de *machine learning* é definir quais variáveis serão entregues à rede neural e qual será sua saída. Como já mencionado anteriormente, as amostras disponíveis constituem o que se chama de dados brutos, com ruído, grande variação e que não apresentam um comportamento bem definido. O ideal, em qualquer projeto, é realizar uma filtragem e um tratamento desses dados, tornando-os aptos a serem entregues à rede.

As primeiras variáveis de entrada definidas para a rede serão os valores máximos e mínimos computados das últimas 6 horas. Isso permite que a rede neural identifique quando há mínimos ou máximos consecutivos, caracterizando uma alteração no comportamento da variável estudada, que representa uma faixa de variação das amostras. O intervalo de análise móvel de 6 horas foi determinado de modo que seja suficientemente pequeno para responder de maneira rápida a modificações importantes, mas sem perder a característica de atenuação das variações, importante para evitar que novos valores de máximos e mínimos sejam computados repetidamente sem representar uma clara mudança de patamar no comportamento da faixa de variação. Desse modo, a figura 5.19 apresenta o comportamento dessas variáveis aplicadas às amostras de pressão no mesmo intervalo da figura 5.18.

Em um segundo momento, é fácil perceber que apenas as variáveis de mínimo e máximo nas últimas 6 horas não serão suficientes para que o modelo consiga atingir o objetivo esperado. Desse modo, as próximas variáveis que a rede neural desenvolvida deve receber são as informações de estado do ativo, mas sem o ruído e a variação constante observadas nos dados

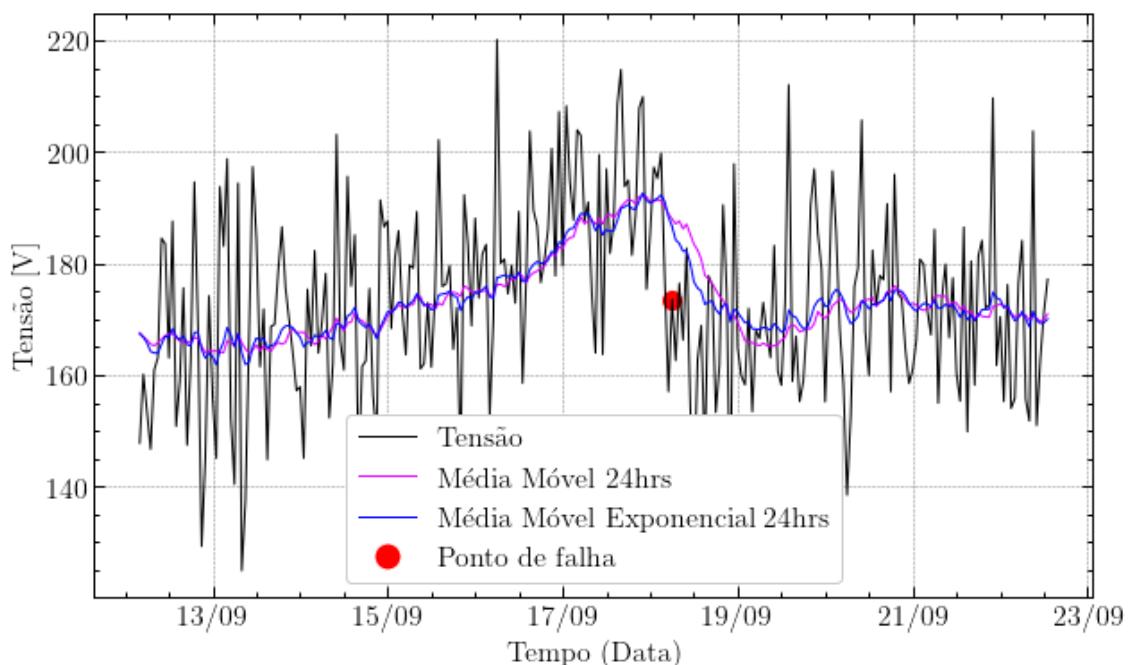
brutos. Nesse contexto, os dados brutos foram filtrados em duas variáveis distintas para cada variável de processo, a média móvel simples e a média móvel exponencial, ambas de 24 períodos, que podem ser analisadas, em relação à tensão, na figura 5.20. Tem-se, então, um total de 4 características por variável de processo, resultando em 16 entradas para o modelo.

Figura 5.19 – Amostra das primeiras variáveis de entrada definidas, máximo e mínimo (6hrs).



Fonte: Autor.

Figura 5.20 – Amostra das variáveis de média móvel simples e exponencial (24hrs) para a tensão.



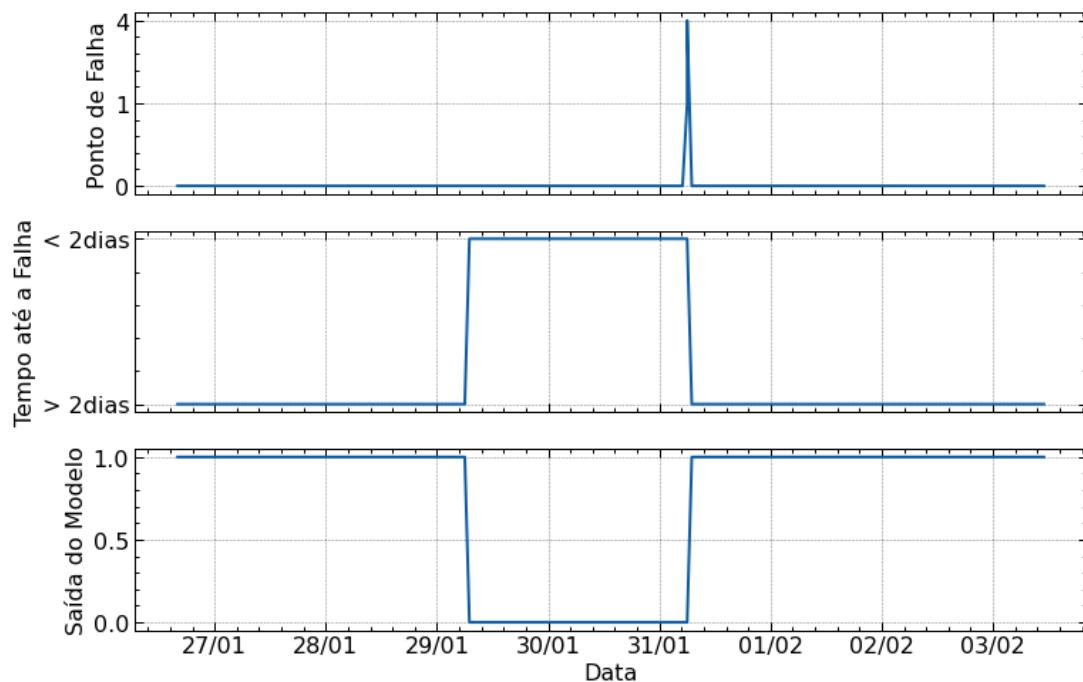
Fonte: Autor.

### 5.3.4 DEFINIÇÃO DA VARIÁVEL DE SAÍDA

Após a definição dos 16 estímulos que serão aplicados à primeira camada da rede neural, deve-se definir qual a saída esperada do modelo. Em linha com o principal objetivo do trabalho, o algoritmo a ser desenvolvido deve ser capaz de alertar sobre a provável falha do equipamento, e não somente indicar se o modo de operação atual está dentro da normalidade ou não.

Com isso em mente e após a identificação do intervalo de previsão no item 5.3.2 deste documento, a variável de saída corresponderá a uma variável binária onde o valor 1 indicará modo de operação normal, enquanto 0 indicará um estado de iminência de falha com menos de 2 dias para ocorrência de problema no ativo. O gráfico 5.21 mostra o comportamento dessa variável, onde logo após o dia 29/01 a saída do modelo deve ser 0, indicando a iminência de falha e alertando aos engenheiros de manutenção responsáveis pela supervisão dos ativos sobre a falha que está para ocorrer (logo após o dia 31/01).

Figura 5.21 - Amostragem do ponto de falha, variável de tempo e a saída esperada do modelo.



Fonte: Autor.

## 5.4 ARQUITETURA E TREINAMENTO DA REDE NEURAL

Para definir a arquitetura da rede neural desenvolvida, deve-se tomar duas decisões. A primeira delas é determinar a quantidade de camadas escondidas, enquanto a segunda a

quantidade de neurônios em cada camada escondida. Isso porque a camada de entrada e saída já foram definidas previamente, com 16 variáveis de entrada (4 por variável de processo), e um único neurônio na camada de saída.

#### **5.4.1 QUANTIDADE DE CAMADAS ESCONDIDAS**

O teorema da aproximação universal afirma que uma rede neural direta com uma única camada escondida contendo um número finito de neurônios pode aproximar funções contínuas com suposições suaves sobre a função de ativação. A primeira versão deste teorema foi proposta por Cybenko (1989) para funções de ativação sigmoidal [17]. Hornik (1991) expandiu essa abordagem, mostrando que não se limita a escolha específica da função de ativação, mas sim a própria arquitetura direta multicamadas que permite às redes neurais o potencial de serem aproximadores universais [18].

Em decorrência desse teorema, uma quantidade considerável da literatura recomenda o uso de uma única camada escondida. Contudo, tudo mudou com Hinton, Osindero & Teh (2006). O questionamento levantado é: se uma camada escondida é suficiente para replicar o comportamento de qualquer função contínua, por que investir pesado em *deep learning*, ou aprendizado profundo? Na verdade, embora o teorema da aproximação universal prove que uma rede neural com uma camada escondida é capaz de aprender e replicar o comportamento de qualquer função, ele não estabelece a arquitetura ótima para todo e qualquer problema, mas sim a arquitetura suficiente [19].

#### **5.4.2 QUANTIDADE DE NEURÔNIOS EM CADA CAMADA ESCONDIDA**

Uma parte fundamental da definição da arquitetura a ser utilizada é a definição da quantidade de neurônios em cada camada escondida. Apesar dessas camadas não interagirem diretamente com o ambiente externo, possuem grande influência na saída da rede.

Utilizar poucos neurônios nessas camadas pode resultar no *underfitting*, ou sub-ajuste, da rede. Esse fenômeno ocorre quando não há neurônios suficientes para detectar adequadamente as não-linearidades presentes nos padrões de saída do conjunto de treinamento. Por outro lado, utilizar uma quantidade exacerbada de neurônios nas camadas escondidas pode acarretar no *overfitting*, ou sobre ajuste. Nesse caso específico, o sobre ajuste acaba sendo caracterizado pela quantidade limitada de amostras em um conjunto de treinamento não ser suficiente para ajustar todos os graus de liberdade da rede neural, que possui uma enorme

capacidade de processamento de informações. Além disso, um grande número de neurônios nas camadas escondidas tem como consequência o incremento do tempo de treinamento, que pode atingir patamares suficientemente grandes de modo que seja inviável realizar o treinamento da rede dentro dos requisitos do projeto da solução.

Evidentemente, deve haver um balanço entre muitos ou poucos neurônios nas camadas escondidas. Para isso, deve-se testar diversas configurações em um processo de tentativa e erro, mas com pontos de partida razoáveis, pois a utilização de números aleatórios em busca da melhor performance pode custar muito tempo.

Existem muitas heurísticas para determinar uma faixa inicial de neurônios para investigar nas camadas ocultas. Algumas delas, são [20]:

- A quantidade de neurônios deve estar entre o tamanho da camada de entrada e o tamanho da camada de saída;
- O número de neurônios ocultos deve ser  $2/3$  do tamanho da camada de entrada acrescido do tamanho da camada de saída;
- O dobro da quantidade de neurônios da camada de entrada deve ser maior que a quantidade oculta.

#### **5.4.3 ESCOLHA FINAL DA ARQUITETURA E TREINAMENTO**

Nesse contexto, foi feita uma validação cruzada, com diferentes arquiteturas (com uma ou duas camadas escondidas) e para uma pequena amostra dos dados pré-processados de treinamento, com o objetivo de verificar aquela que possui o melhor desempenho. Essa arquitetura será, então, a que seguirá para o ciclo completo de treinamento.

Os modelos apresentados para a validação cruzada foram montados com base nas heurísticas apresentadas na seção 5.3.2 deste documento. Como não se pode afirmar que uma rede neural com apenas uma camada escondida será a mais adequada (aprendendo de forma mais rápida e performando melhor de modo geral), foram selecionadas 8 arquiteturas de rede, metade com duas camadas escondidas e a outra metade com apenas uma. A quantidade de neurônios em cada camada escondida foi selecionada respeitando a heurística pré-estabelecida. A tabela 5.1 mostra informações mais específicas sobre os modelos, como suas camadas, neurônios, parâmetros treináveis e função de ativação utilizada.

Tabela 5.1 - Arquitetura das redes neurais utilizadas na validação cruzada.

Modelo	Camadas Escondidas	Neurônios por Camada [Entrada, Escondida, Saída]	Parâmetros Treináveis	Função de Ativação
1	1	[16, 4, 1]	73	Sigmoide
2	1	[16, 8, 1]	145	
3	1	[16, 12, 1]	217	
4	1	[16, 16, 1]	289	
5	2	[16, 4, 2, 1]	81	
6	2	[16, 8, 4, 1]	177	
7	2	[16, 12, 6, 1]	289	
8	2	[16, 16, 8, 1]	417	

Fonte: Autor

As avaliações feitas em cima do treinamento de cada arquitetura foram respectivas à precisão (seção 3.7.3), *loss* (computada através da perda binária de entropia cruzada, vide seção 3.5) e esforço computacional do modelo (tempo por época e total de treinamento). O resultado dessa heurística está representado nas figuras 5.22, 5.23 e 5.24 (a partir da 5ª época, pois as épocas iniciais possuem valores substancialmente diferentes que alteram a proporção do eixo *y*, impossibilitando uma análise mais detalhada das figuras nas épocas finais).

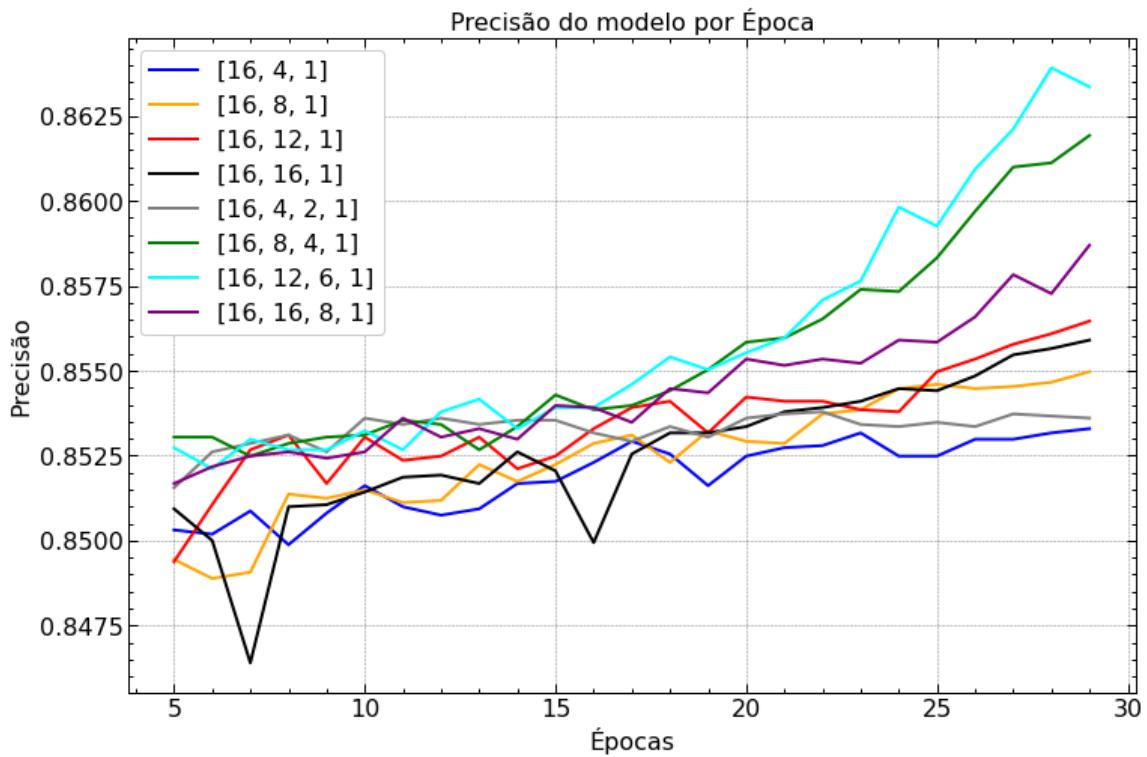
Em uma primeira análise, é possível observar que os modelos que possuíram melhor desempenho quanto à precisão foram os que possuíam 2 camadas escondidas, com destaque para os modelos 6 e 7. Vale ressaltar que o modelo 7, que atingiu a melhor precisão dentre todas as arquiteturas, constitui-se exatamente de 12 neurônios na sua primeira camada escondida – o que corresponde a exatos 2/3 da quantidade de neurônios da camada de entrada (10.67) adicionados da quantidade de neurônios na camada de saída (1) – conforme as heurísticas de Heaton (2008) apresentadas na seção 5.3.2 deste documento.

No que tange as perdas de cada arquitetura, é possível distinguir claramente os modelos de uma única camada escondida (com perda maior) daqueles com duas camadas escondidas. Isso significa que, apesar de apenas uma camada escondida ser suficiente para satisfazer o objetivo delimitado inicialmente, ela não parece a melhor opção, a princípio, neste problema específico.

Por fim, a respeito do esforço computacional das arquiteturas, observa-se claramente a disparidade entre as arquiteturas com uma camada escondida em relação as que possuem duas. O *Google Colaboratory*, ambiente utilizado na execução de todo o código deste trabalho, conta com um processador Intel Xeon com dois núcleos @2.3 GHz, com 13 GB de memória RAM, equipado com a placa de vídeo NVIDIA Tesla K80 com 12 GB RAM [21]. Para esse ambiente,

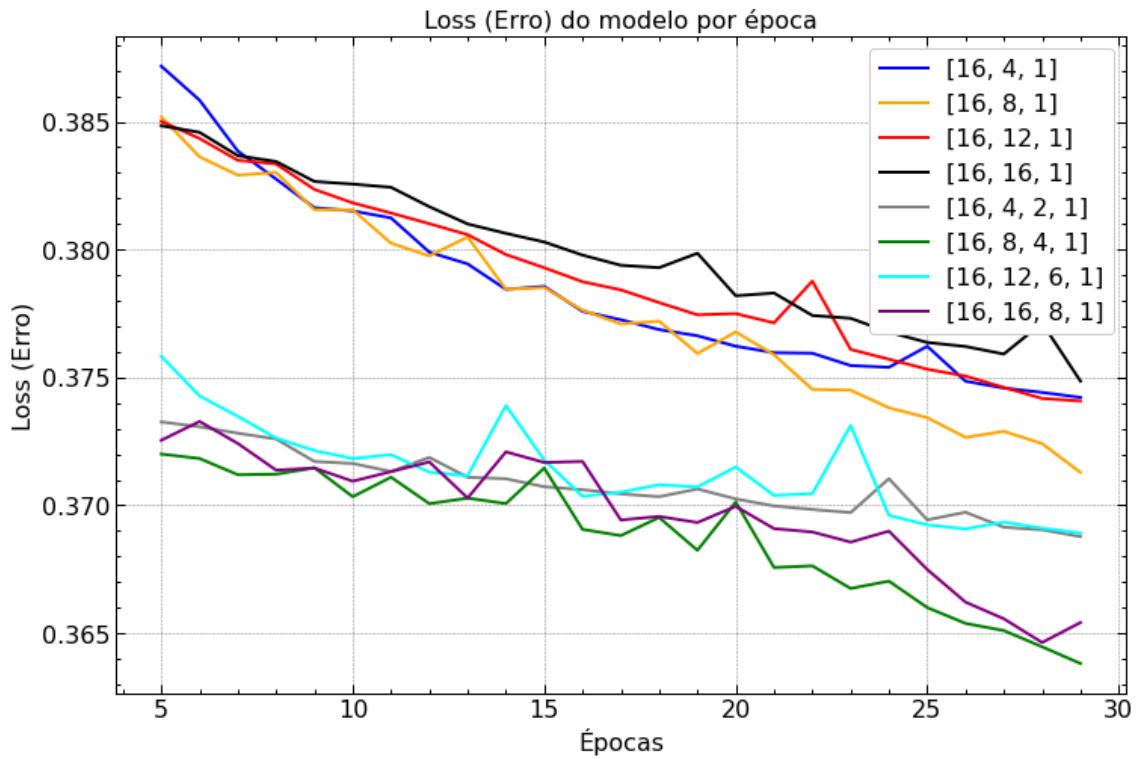
em específico, verifica-se um custo computacional, em média, de aproximadamente 0.5 segundos de diferença por época entre o tempo de treinamento de uma rede com camada escondida única e outra com duas. Nesse contexto, esse custo não é suficiente para impedir a escolha pela arquitetura mais complexa. Vale ressaltar que, dependendo das limitações de hardware do ambiente de desenvolvimento utilizado, uma diferente análise pode ser obtida e, consequentemente, uma conclusão diferente também.

Figura 5.22 - Validação cruzada para 8 arquiteturas distintas.



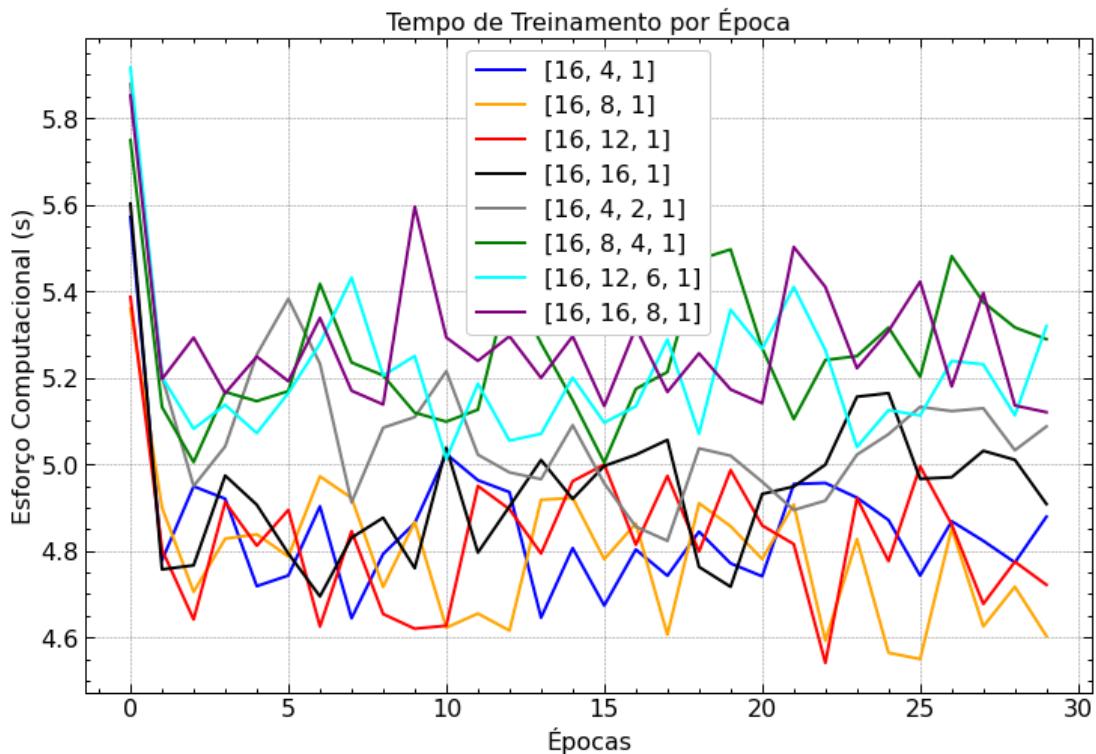
Fonte: Autor.

Figura 5.23 - Perda de cada arquitetura testada por época



Fonte: Autor.

Figura 5.24 - Esforço computacional (tempo de cada época em segundos)



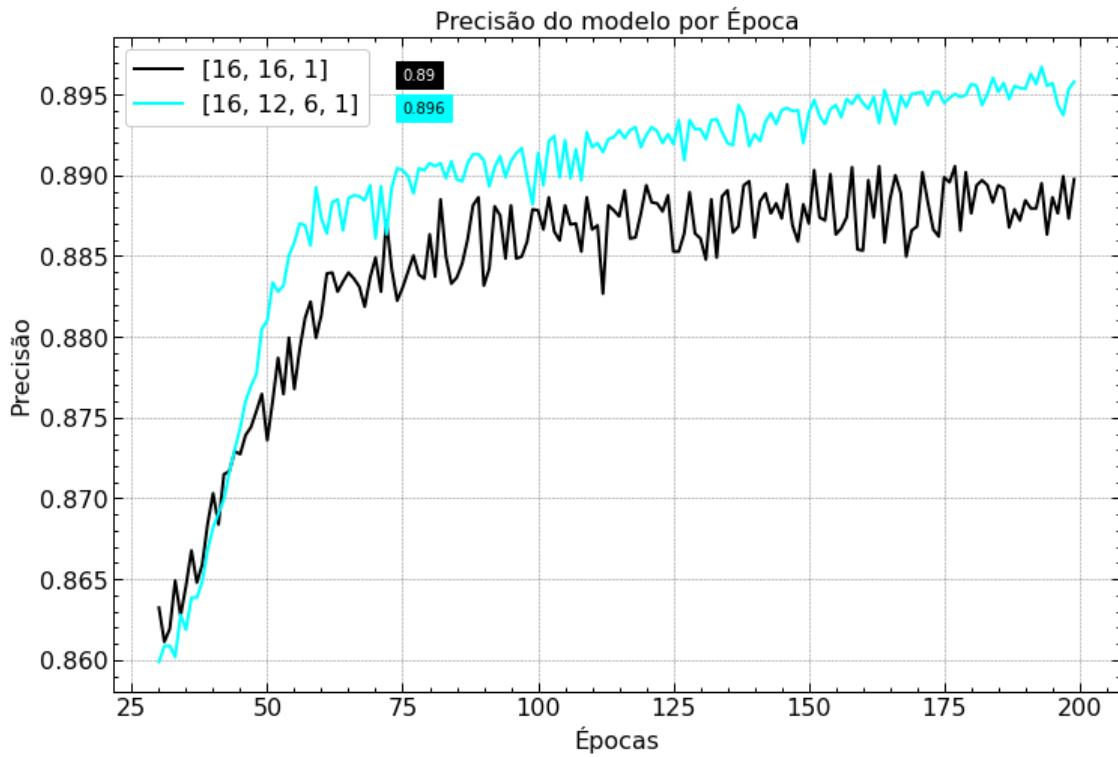
Fonte: Autor.

Por fim, realizou-se o treinamento completo de duas arquiteturas específicas, com o objetivo de verificar se, de fato, apenas uma camada escondida soluciona o problema em questão de forma equivalente às arquiteturas com duas camadas. Para isso, os modelos 4 e 7, que possuem a mesma quantidade de parâmetros de treinamento ajustáveis, receberam um treinamento completo de 200 épocas com 75% dos dados disponíveis (25% para teste) de 95 máquinas distintas – isso porque 5 máquinas foram separadas para avaliação posterior com a simulação do funcionamento do modelo desenvolvido em tempo real. As figuras 5.25, 5.26 e 5.27 mostram os resultados de precisão, perda e esforço computacional, respectivamente, da apresentação das amostras de teste para as arquiteturas por época.

É possível observar que, para o gráfico da precisão dos modelos, obteve-se uma diferença de apenas 0,6% de precisão entre as duas arquiteturas, valor que não é, neste momento, suficiente para justificar a adoção de uma rede neural mais complexa com duas camadas escondidas em detrimento da mais simples. Analisando a figura 5.26, verifica-se que a diferença entre as perdas dos modelos na última época de treinamento é próxima de 0.02, o que fortalece a teoria inicial de que o modelo mais complexo não é necessário. A análise do esforço computacional, desta vez representado na figura 5.27 como o esforço acumulado – somando-se o tempo de treinamento por época até obter-se o tempo total de treinamento, mostra uma clara vantagem no tempo de treinamento a favor da arquitetura mais simplista, em torno de 9,8% mais rápido.

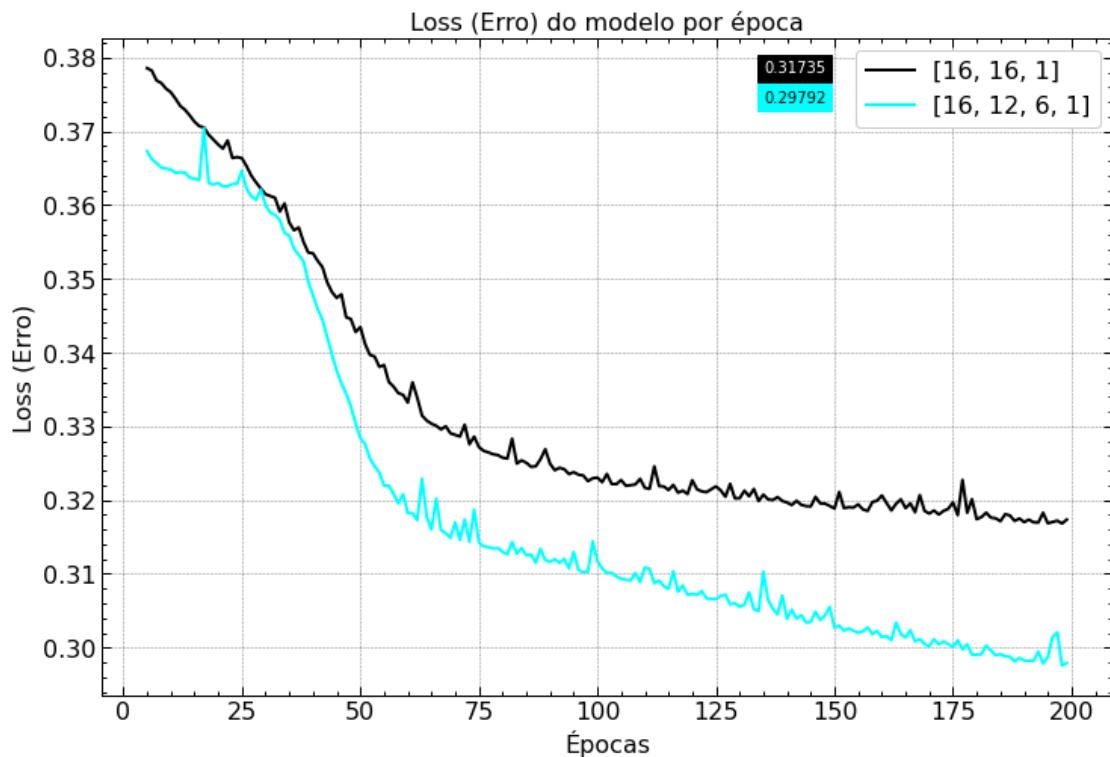
Assim, percebe-se que as arquiteturas não apresentam diferenças significativas em seu desempenho que justifiquem a escolha de um sobre o outro. Portanto, entende-se que os modelos são equivalentes. Nesse contexto, o modelo utilizado será o mais simples, com menor complexidade e que permite um treinamento mais rápido.

Figura 5.25 - Precisão dos modelos selecionados para treinamento completo por época.



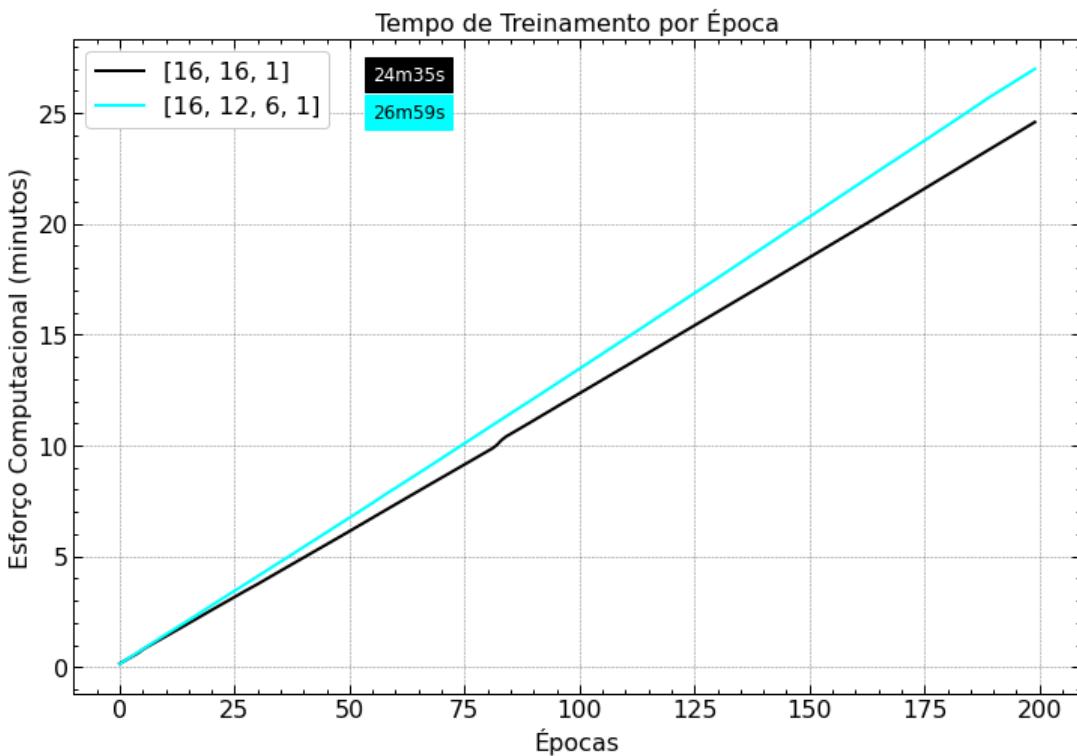
Fonte: Autor.

Figura 5.26 - Perda dos modelos selecionados para treinamento completo por época



Fonte: Autor.

Figura 5.27 - Esforço computacional acumulativo dos modelos selecionados para treinamento completo



Fonte: Autor.

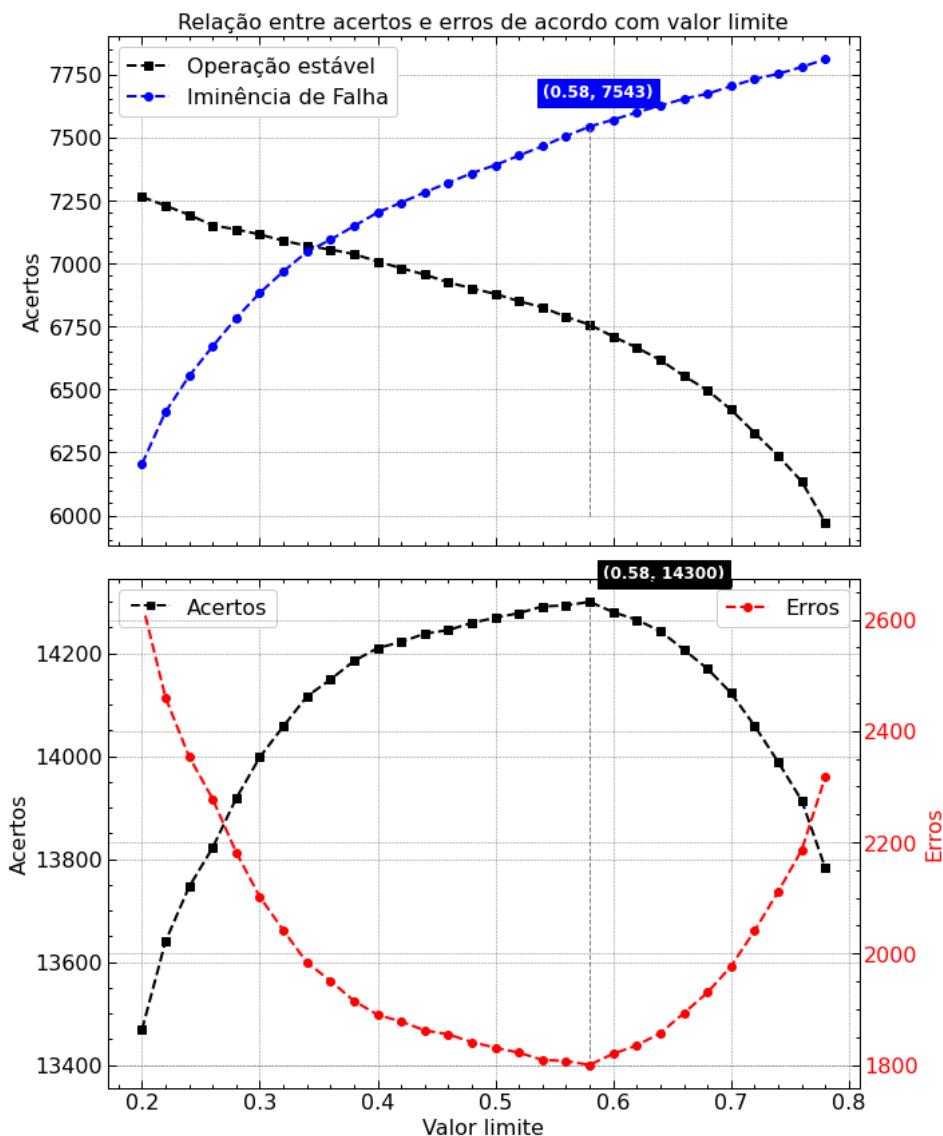
## 5.5 AVALIAÇÃO DA PERFORMANCE E PÓS PROCESSAMENTO

Após a finalização do treinamento da rede neural desenvolvida, é necessário avaliar seu desempenho em dados de teste previamente separados, para determinar sua aplicabilidade na solução do problema em questão. Assim, 25% dos dados separados, as amostras de teste, foram apresentados ao modelo. A saída do modelo consiste em um único valor entre 0 e 1 que pode ser entendido como uma medida de probabilidade de que o padrão apresentado na entrada pertença à determinada classe. Assim, quanto mais próximo de 0, maior a probabilidade de pertencer à classe 0 (iminência de falha) e, analogamente, quanto mais próximo de 1, maior a probabilidade de pertencer à classe 1 (operação estável).

Apesar disso, essa probabilidade não define, de fato, a qual classe será atribuída. Na verdade, essa classificação depende das regras de pós processamento, que são definidas com o objetivo de otimizar a performance do modelo desenvolvido. Para compreender melhor as alterações no desempenho do modelo de acordo com a variação do limiar de decisão para atribuição do padrão de entrada a uma das duas classes (falha em menos de 48 horas ou falha em mais de 48 horas), foram construídos os dois gráficos da figura 5.28. O primeiro é responsável por computar os acertos de cada classe, enquanto o segundo os acertos e erros

totais. O objetivo principal para o ajuste do limiar, neste caso, é a redução dos falsos positivos, ou seja, reduzir a quantidade de vezes em que a solução prevê operação estável quando, na realidade, o ativo se encontra em iminência de falha. Para atingir essa meta, deve-se ajustar o limiar de decisão de modo a incluir uma quantidade maior de previsões para a classe 0 (falha em menos de 48 horas). Por outro lado, como o gráfico mostra, a redução nos falsos positivos tem um custo: o aumento dos verdadeiros negativos, que ocorre quando o modelo prevê falha em menos de 48 horas, mas o ativo está operando normalmente. Assim, deve-se levar em conta o risco que se deseja assumir na estratégia adotada para ajuste desse valor limite, tendo em mente que o favorecimento de uma classe só é possível em detrimento da outra. Para este documento, foi adotada a seguinte premissa: falhas não detectadas são muito mais prejudiciais para os ativos em questão do que a indicação equivocada de iminência de falha. Assim, seguindo essa estratégia específica, foi feito um aumento no valor do limiar, para redução dos falsos positivos, até o valor em que, de acordo com o segundo gráfico, a solução tem a maior proporção entre acertos e erros. É possível observar que, a maior quantidade de acertos se dá quando temos um limiar de decisão igual a 0,58, o que significa que previsões menores que esse limite recebem a classe 0, representando iminência de falha, enquanto valores maiores recebem a classe 1, operação estável.

Figura 5.28 - Gráficos das Métricas de acordo com o valor limite de decisão aplicado



Fonte: Autor.

Assim, definido o valor limite, a tabela 5.2 apresenta a matriz de confusão e as métricas de avaliação da aplicação da rede neural treinada aos dados de treinamento previamente separados.

Tabela 5.2 – Matriz de confusão e métricas de avaliação do modelo desenvolvido.

	Previsão do Modelo		Métricas		
	0	1	Precisão	Recall	Acurácia
Verdades	0	7543	579	86,1%	92,9%
	1	1221	6757	92,1%	84,7%

Fonte: Autor.

Em um primeiro momento, destaca-se, positivamente, a acurácia obtida após o treinamento de 200 épocas. Por outro lado, analisar apenas essa métrica não é a melhor maneira de avaliar o desempenho da rede neural. Assim, deve-se observar, também, precisão, *recall* e a relação entre elas para determinar se a performance está como esperada ou não.

A precisão, neste caso, é responsável por determinar a quantidade de acertos para cada classe prevista. Com isso, observa-se que 86,1% de todas as 8764 previsões para a classe 0 (iminência de falha) estavam corretas, enquanto que 92,1% das 7336 previsões para a classe 1 (operação normal) foram acertadas. O fato de a precisão de acerto para a classe de provável falha não indica uma má performance do modelo, uma vez que falsos negativos são esperados na busca por minimizar os falsos positivos.

No caso do parâmetro *recall*, é responsável por indicar a quantidade de acertos para cada classe real, isto é, de todas as amostras pertencentes a determinada classificação, a quantidade de separações corretas. Verifica-se, então, que houve um *recall* de 92,9% para iminência de falha, enquanto apenas 84,7% para a operação normal dos ativos.

Assim, entende-se que ao maximizar a performance para prever corretamente os instantes de iminência de falha, potencializa-se a confiabilidade do modelo em evitar falsos positivos – isto é, não prever iminência de falha em momentos críticos de operação. No entanto, prejudica-se o desempenho em instantes de operação normal, resultando em alarmes isolados que podem dificultar a avaliação das condições de operação nessas circunstâncias. Por isso, faz-se necessário desenvolver uma função que contabiliza uma determinada quantidade de previsões anteriores para definir se o alarme de iminência de falha deve ser ou não acionado.

Nesse contexto, a função definida para ponderar esse alarme foi a média móvel exponencial. A equação da MME escolhida leva em consideração a previsão dos 23 instantes anteriores e do instante atual (logo, 24). Ainda, foi determinado um valor de 65% para acionamento do alarme, isto é, quando o nível do alarme atinge valores iguais ou superiores à 0,65, tem-se o acionamento do mesmo. Assim, falsos negativos não serão fortes o suficiente para impactar o processo de decisão, uma vez que ocorrem, na maioria dos casos, descontinuamente e de maneira isolada. Vale ressaltar que, assim como o valor limite de decisão, o nível de ativação do alarme também depende da estratégia a ser seguida. Nesse sentido, esse número pode aumentar ou diminuir de acordo com o risco que o gestor de manutenção responsável esteja disposto a correr. Por fim, resta avaliar a performance da rede neural desenvolvida em conjunto com as técnicas de pós processamento implementadas em simulações em tempo real.

## 5.6 SIMULAÇÃO DA OPERAÇÃO DA SOLUÇÃO

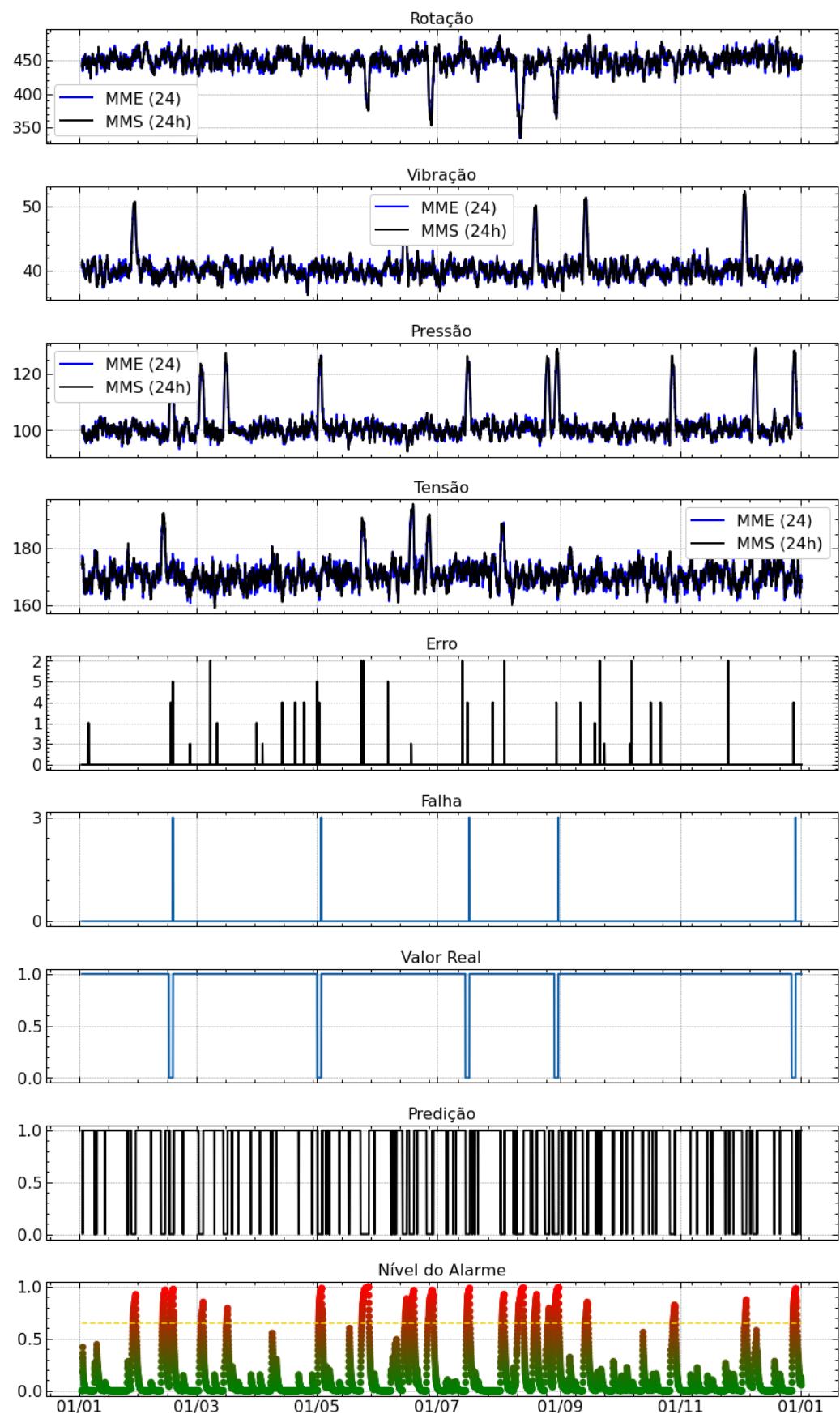
Com a conclusão de todas as etapas anteriores, tem-se a conclusão do desenvolvimento da solução de aprendizado de máquina de apoio à manutenção preditiva. O algoritmo é responsável por realizar desde as etapas iniciais de pré-processamento dos dados, até a variável de saída que compõe um alarme com o pós-processamento das inferências da rede neural.

Assim, antes de levar a aplicação desenvolvida para um ambiente de produção com máquinas e condições reais de operação em tempo real, é recomendável verificar a solução desenvolvida em ambiente similar. Para isso, como dito na seção 5.4.3, foi realizada a simulação da solução desenvolvida, operando com amostras de 5 máquinas distintas, representadas nas figuras 5.29 até 5.33. Esta etapa difere das etapas de testes anteriores pelo fato de as amostras apresentadas ao modelo seguirem a ordem cronológica de captura pelos sensores, diferentemente da etapa de testes, onde eram apresentadas em ordem aleatória.

Nessas simulações, é possível perceber que as amostras apresentam um comportamento similar na iminência de um erro àquele observado em instantes próximos à falha. Assim, nesses períodos, a resposta da solução desenvolvida é a ativação do alarme. Porém, não se deve entender esse alarme como uma confusão ou equívoco da solução desenvolvida. Na verdade, como nesses instantes de erro as variáveis de processo se comportam de maneira similar à iminência de falha, ao acionar o alarme, indicando possível falha, a solução desenvolvida está atuando de maneira preventiva e evitando que o ativo supervisionado opere fora da sua faixa nominal, preservando e prolongando, assim, a integridade e vida útil do ativo. É possível observar esse comportamento com mais detalhes na figura 5.34, onde estão representados um instante de erro e outro de iminência de falha, ambos com ativação do alarme. Nesta última, é perceptível, ainda, a influência da função de pós processamento responsável pelo acionamento do alarme, levando em consideração as previsões das últimas 24 horas.

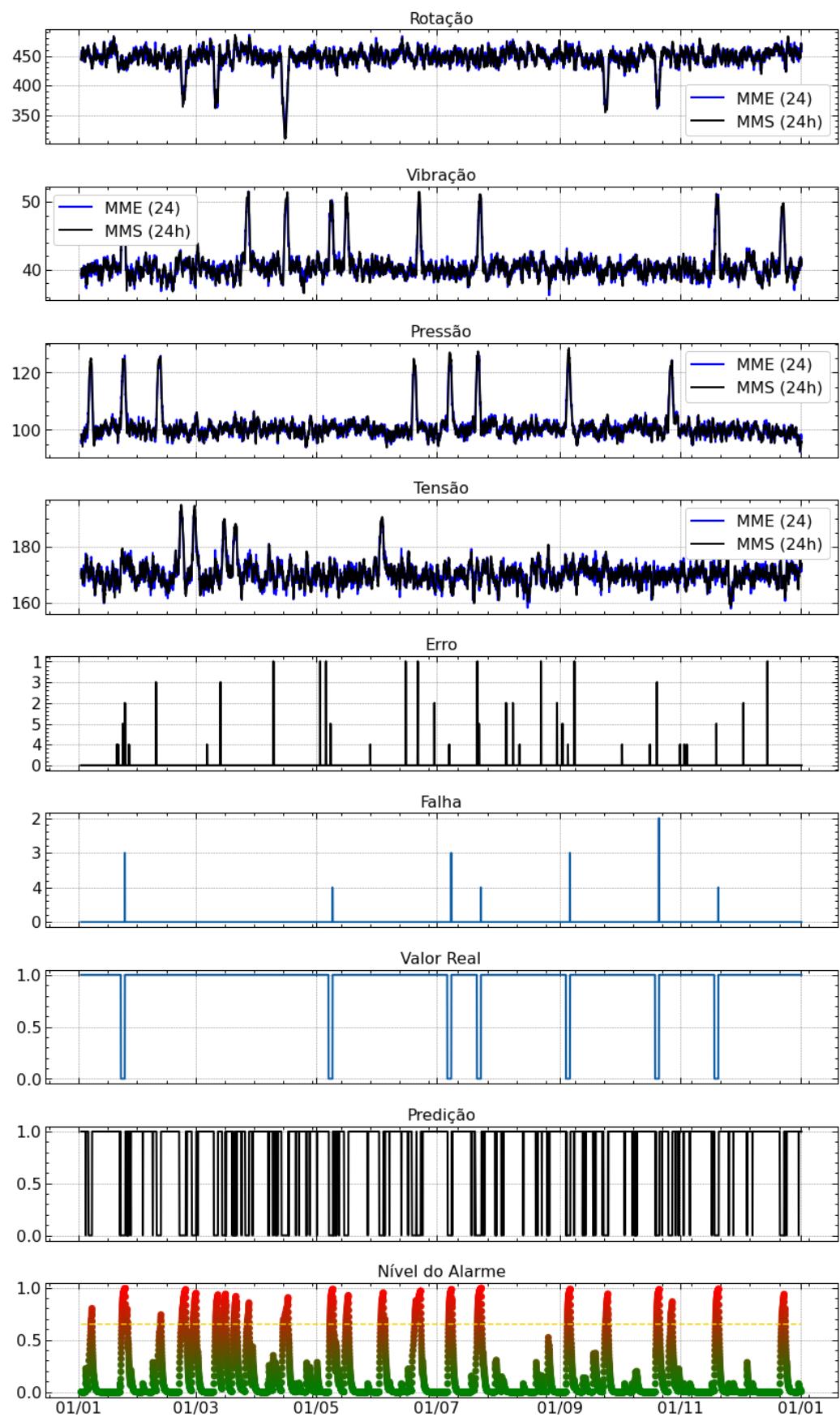
De acordo com o exposto, entende-se que a avaliação da performance da solução desenvolvida nas simulações vai além das métricas de desempenho, uma vez que atua preventivamente na ocorrência de erros – o que seria considerado uma imprecisão nas métricas. Ainda, essa qualificação do desempenho é totalmente dependente do nível escolhido para acionamento do alarme, neste caso 65%, mas que pode ser variável de acordo com o perfil da empresa. Uma abordagem mais conservadora pode usar um nível mais baixo, com maior prevenção, enquanto uma perspectiva mais agressiva pode elevar ainda mais.

Figura 5.29 – Simulação da operação da solução para as amostras da máquina de número 96.



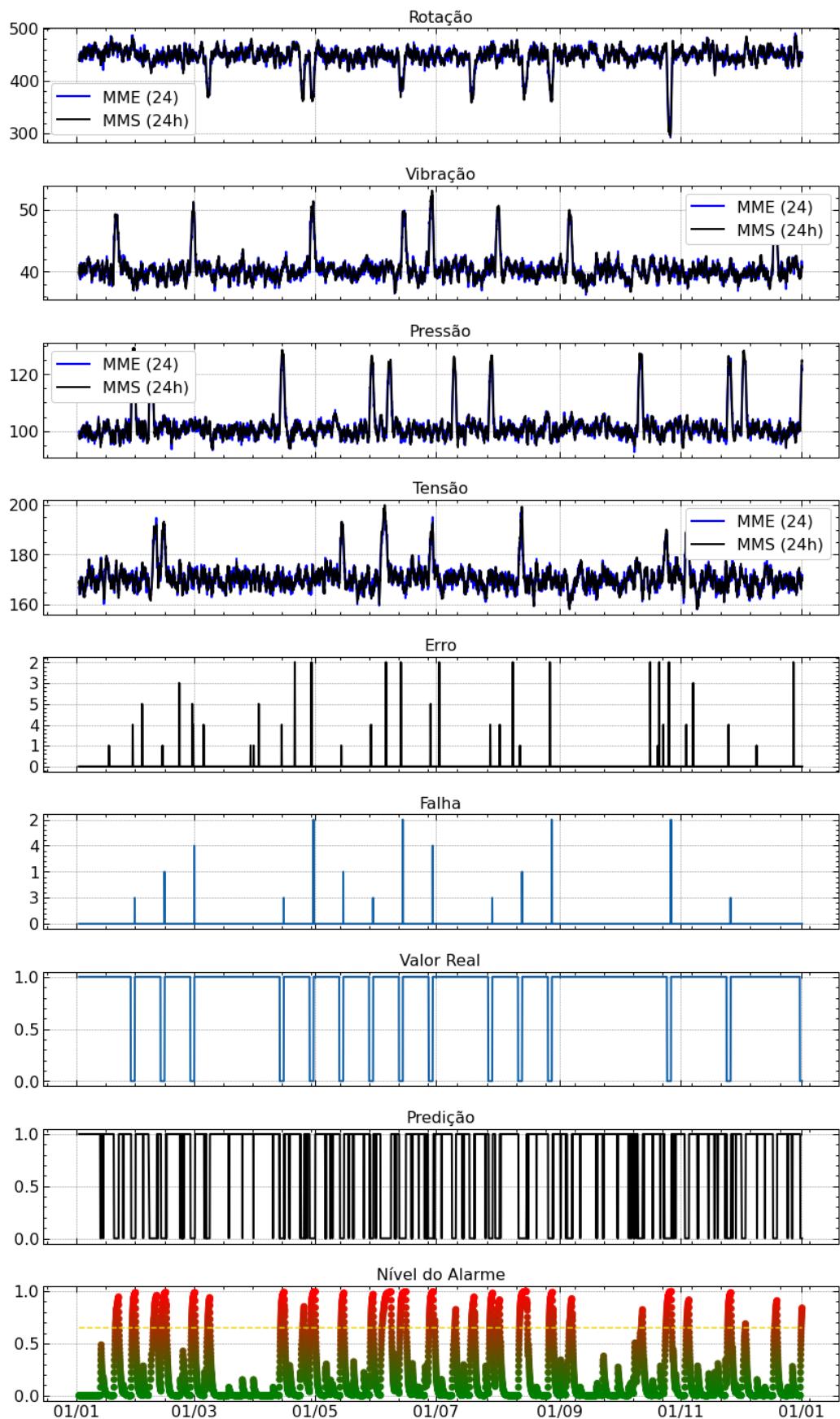
Fonte: Autor.

Figura 5.30 – Simulação da operação da solução para as amostras da máquina de número 97.



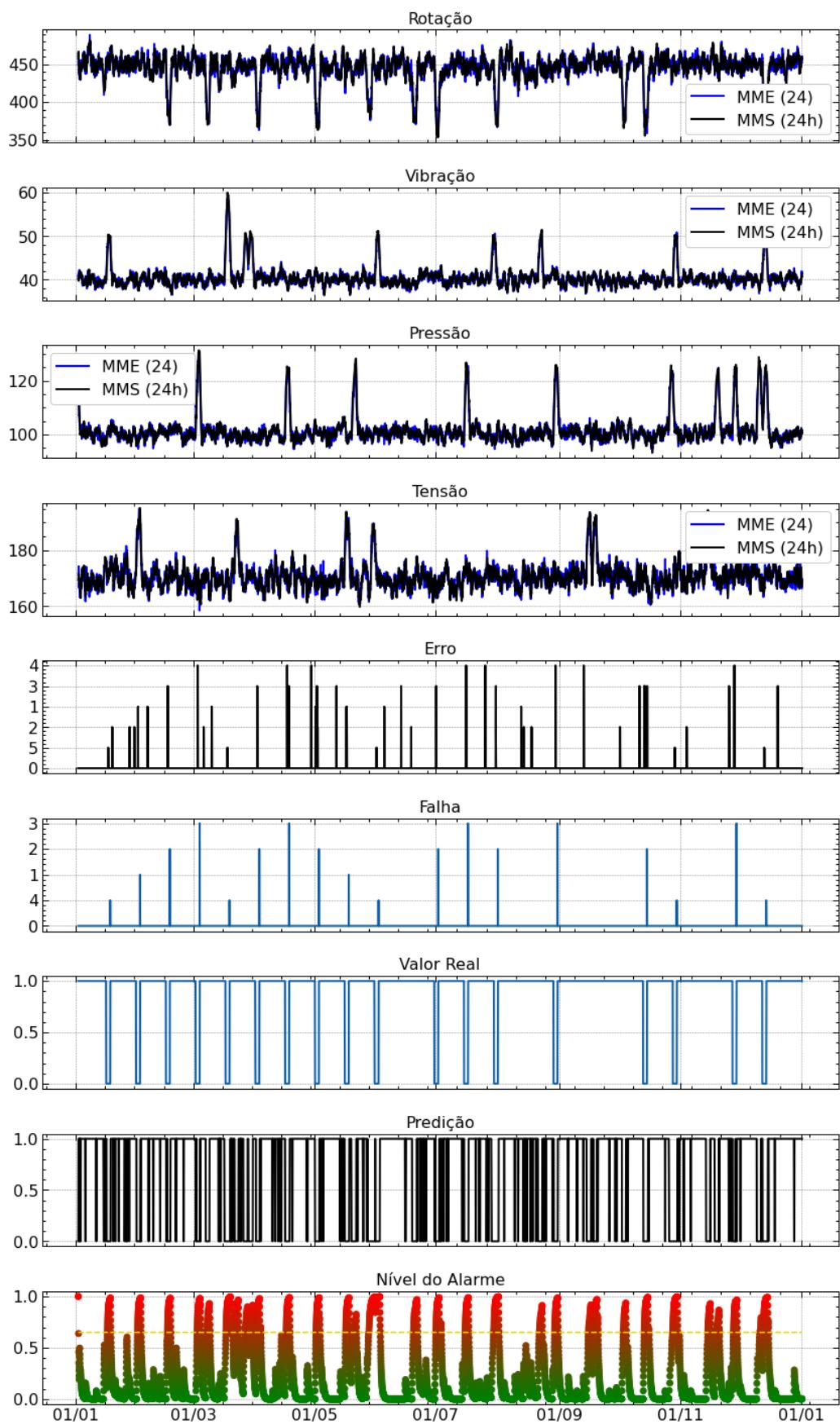
Fonte: Autor.

Figura 5.31 – Simulação da operação da solução para as amostras da máquina de número 98.



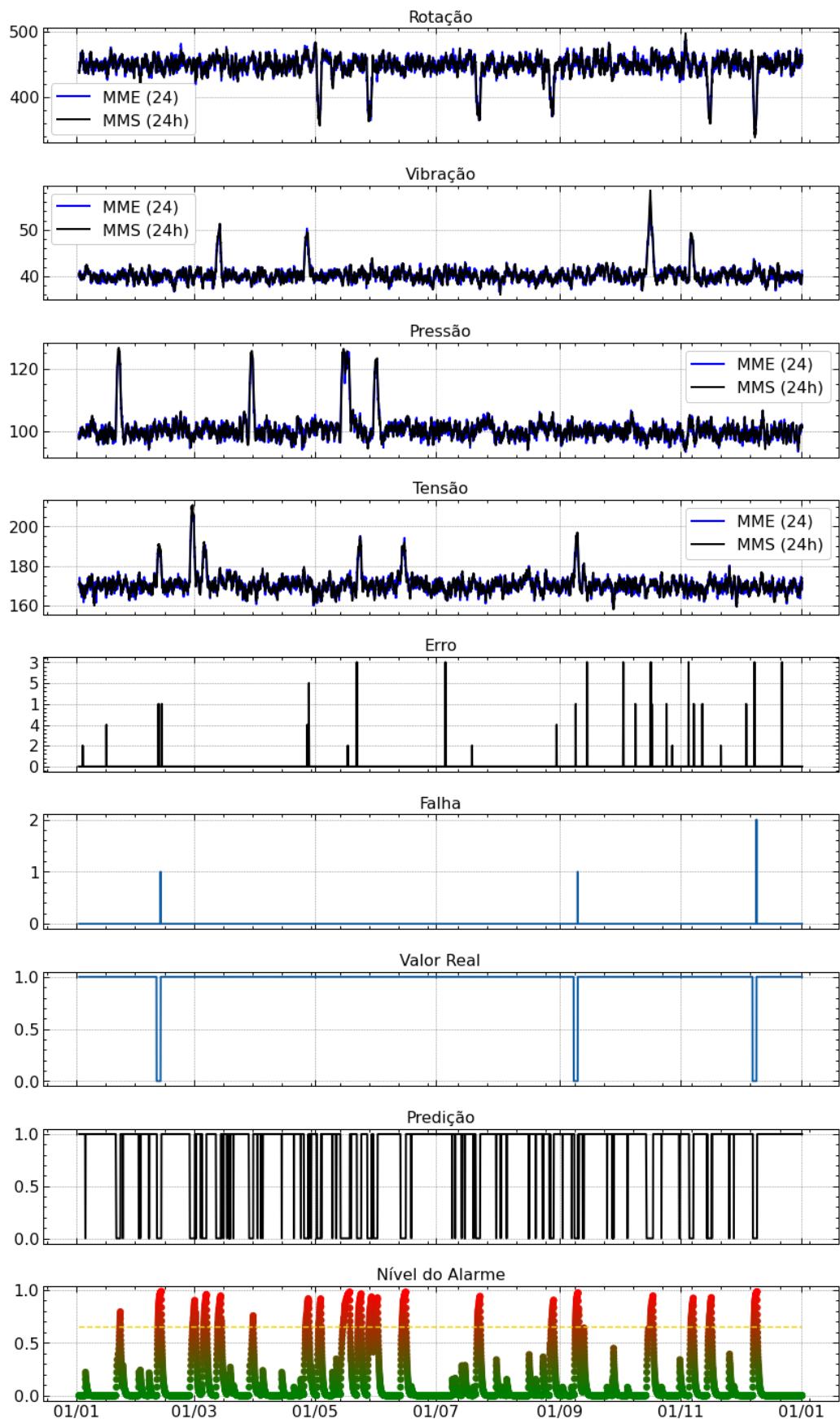
Fonte: Autor.

Figura 5.32 – Simulação da operação da solução para as amostras da máquina de número 99.



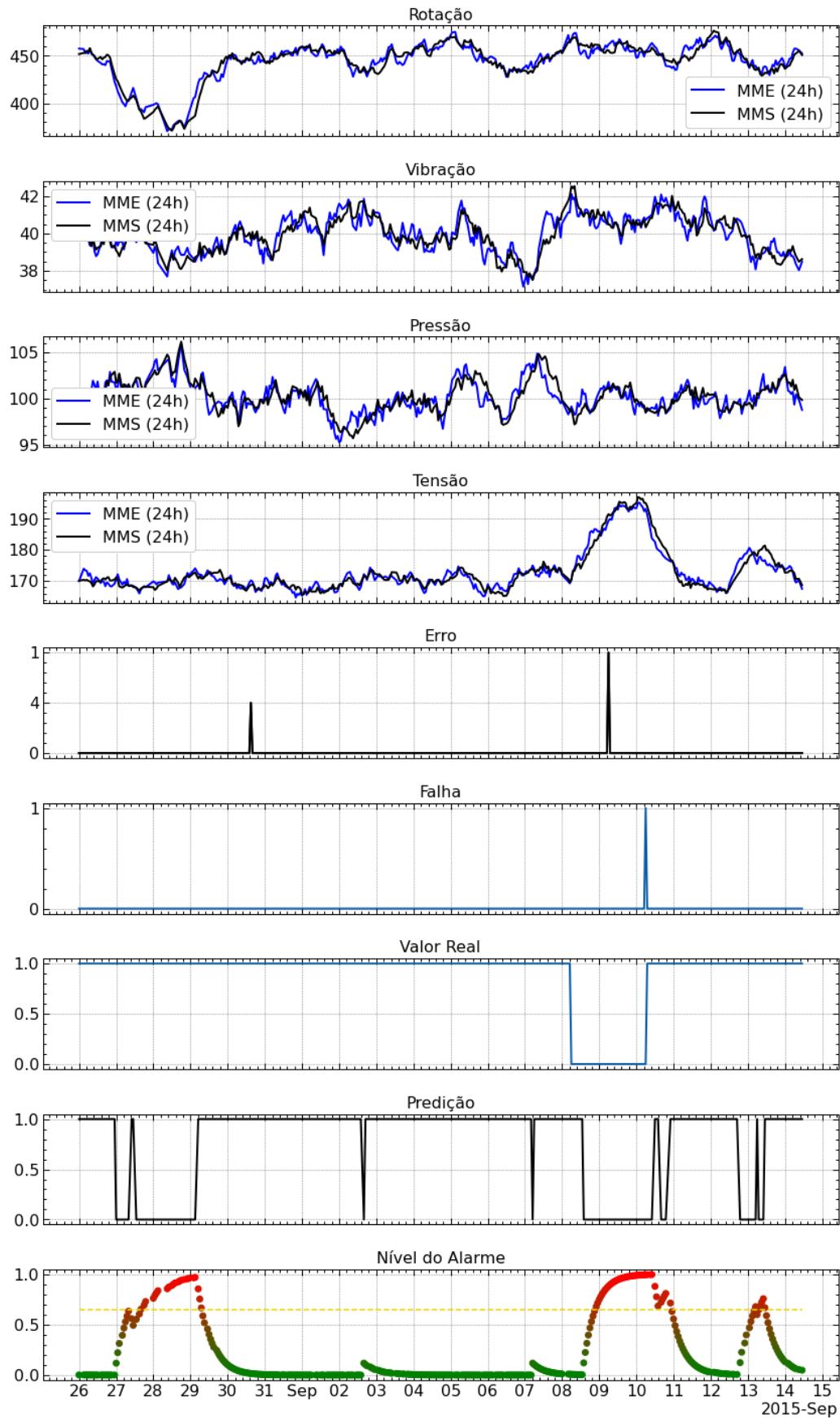
Fonte: Autor.

Figura 5.33 – Simulação da operação da solução para as amostras da máquina de número 100.



Fonte: Autor.

Figura 5.34 – Detalhes de falsos negativos da máquina 100 devido à variável de erro.



Fonte: Autor.

## 6 CONCLUSÃO E TRABALHOS FUTUROS

Em um ambiente onde as principais tecnologias e conceitos da indústria 4.0 – como computação em nuvem, *Big Data*, e outros – são aplicados na prática, é possível tirar vantagem dessas inovações para melhorar diversas atividades de uma empresa. Em particular, para diferentes estratégias de gerenciamento de ativos, destaca-se a implementação de uma abordagem que, utilizando as novas tecnologias citadas, minimiza os custos e maximiza a vida útil desses ativos.

Após a fundamentação teórica, desenvolvimento e exposição dos resultados, fica evidente a possibilidade de implementação de estratégias de gestão de ativos que utilizem a manutenção preditiva baseada em inferências de modelos e algoritmos computacionais de *machine learning*. O resultado é direto: redução na quantidade e no tempo de paradas, eliminação de efeitos de falha em cascata e queda nos custos de manutenção.

Desenvolvida com o objetivo de identificar falhas de maneira prévia, a solução foi além, superando as expectativas e mostrando, também, capacidade de identificação de erros e intervalos onde o comportamento das variáveis de processo se mostra fora da faixa nominal das máquinas. Mais especificamente, a variável de erro, que indica presença de problema momentâneo, mas que não causa a interrupção no funcionamento do ativo, foi responsável por intervalos de previsão muito similares a momentos de falha, além do acionamento da variável de alarme de pós processamento das previsões. Assim, verifica-se que a solução desenvolvida atua tanto na previsão quanto na prevenção de falhas.

De modo geral, os resultados obtidos pelas métricas de avaliação do modelo foram excelentes e também superaram as expectativas iniciais. Destaca-se o percentual de 92,9% atingido para o parâmetro recall, representando que dentre todos os 8122 instantes em que o ativo se encontrava em iminência de falha, 7543 foram previstos corretamente, com apenas 579 falsos positivos. Vale ressaltar, também, a acurácia de 88,82%. Esse percentual foi alcançado com 200 épocas de treinamento, mas acredita-se que, pelas figuras 5.25 e 5.26, ainda existe espaço para melhorias nos resultados, investindo-se um tempo maior no treinamento da rede neural a partir do aumento na quantidade de épocas.

Merece destaque, também, a capacidade de customização da solução desenvolvida neste documento. Isto é, a possibilidade de ajustar alguns parâmetros nas técnicas de pós processamento com o objetivo de que a solução se alinhe melhor com a estratégia desejada e os riscos associados. Mais detalhadamente, em um primeiro momento e para uma estratégia mais conservadora, o limiar de decisão de classificação do modelo pode ser alterado de forma

que se acerte ainda mais classificações de iminência de falhas e reduzindo falsos positivos, assumindo, em compensação, uma quantidade maior de verdadeiros negativos e prejudicando a precisão de classificações de operação normal. Ainda nessa estratégia, outro parâmetro que pode ser ajustado é o nível do alarme. Reduzir esse valor tornaria o alarme mais sensível a previsões de falha e eliminaria ainda mais o risco de falha repentina do equipamento sem aviso prévio, em troca de uma maior quantidade de acionamento. Pode-se realizar, também, um incremento do nível de alarme para reduzir a sensibilidade da solução e balancear com o limiar de decisão incrementado. Por outro lado, estratégias mais agressivas, de modo geral, estão sujeitas a correr riscos maiores e, assim, os parâmetros são ajustados com o objetivo de aumentar a acurácia e reduzir o número total de erros de classificação. Portanto, fica evidente que a solução desenvolvida é capaz de se ajustar a diferentes cenários, dependendo da estratégia que se deseja implementar e dos riscos que se está disposto a correr em busca da redução da quantidade de intervenções, da eliminação de falhas de equipamentos e do aproveitamento máximo da vida útil de ativos.

Em contrapartida, a aplicação de aprendizado de máquina na gestão de ativos não se limita ao exposto neste documento, uma vez que a rede neural desenvolvida representa apenas um ponto de partida para diversos outros modelos e algoritmos que podem ser construídos com o mesmo objetivo. Pode-se investigar ainda, em trabalhos futuros, formas de otimização das técnicas de pós processamento aplicadas em conjunto com a função de acionamento do alarme. Outra alternativa é realizar a comparação do desempenho de diferentes modelos de aprendizado de máquina aplicadas em um mesmo conjunto de dados ou ainda realizar um acoplamento de modelos com o objetivo de que operem em conjunto. Vale destacar, também, a possibilidade de aplicar as técnicas desenvolvidas neste trabalho para dados com mais detalhes e informações sobre sua origem, uma vez que o banco de dados utilizado não dispõe de informações específicas e é pouco informativo em relação à natureza das máquinas e aos tipos de processos em que são utilizadas.

Por fim, como qualquer nova atividade dentro de uma empresa, a adesão de uma estratégia de manutenção preditiva de ativos requer um investimento inicial em diferentes áreas, como no desenvolvimento de um sistema de aquisição e armazenamento de dados – em nuvem ou servidores físicos – e na integração entre as equipes de gestão, desenvolvedores e técnicos ou engenheiros responsáveis pela operação desses ativos. Entretanto, mesmo nesses casos, onde o ambiente não é o ideal, o investimento rapidamente se paga e se transforma em lucro. Isso, em parte, porque, de modo geral, o reparo ou substituição de ativos representa entre 15% e 30%

do custo dos bens produzidos [22] e uma estratégia executada com êxito proporcionará um retorno do investimento de seis a doze meses [23].

Nesse sentido, este trabalho possibilitou entender o processo de construção de uma solução baseada em algoritmos de aprendizado de máquina para apoiar uma estratégia de manutenção preditiva desde o início do projeto, onde são definidos os objetivos e metas, até o fim, onde é feito o pós-processamento das saídas do modelo.

## REFERÊNCIAS BIBLIOGRÁFICAS

- [1] TELES, Jhonata. Indústria 4.0 - Tudo sobre a Quarta Revolução Industrial. 31 out. 2017. Engeteles. Disponível em: <https://engeteles.com.br/industria-4-0/>. Acesso em: 16 jan. 2022.
- [2] NORMA ABNT NBR ISO 55000:2014, Gestão de ativos - visão geral, princípios e terminologia.
- [3] THE INSTITUTE OF ASSET MANAGEMENT. An Anatomy of Asset Management. Versão 3, dezembro, 2015. 8 p.
- [4] INTERNATIONAL COPPER ASSOCIATION LATIN AMERICA. Gestão Ativos No Setor Elétrico Da América Latina Melhores Práticas E Tendências. 2014.
- [5] LUCIANO, Bruno. Gestão de ativos na manutenção industrial: tendência ou realidade? 10 ago. 2021. Abecom Rolamentos SKF. Disponível em: <https://www.abecom.com.br/gestao-de-ativos-na-manutencao-industrial/>. Acesso em: 16 jan. 2022.
- [6] KARDEC, Alan; NASCIF, Júlio. Manutenção: função estratégica. 3. ed. rev. e ampl. Rio de Janeiro: Qualitymark: Petrobras. 2009.
- [7] Predix Platform | Industrial IoT Application Platform | GE Digital. 2018. Ge.com. Disponível em: <https://www.ge.com/digital/iiot-platform>. Acesso em: 6 nov. 2021.
- [8] LICHTENBERGER, Stefan. Asset Performance Management 4.0: Predict with confidence within the Digital Twin. Siemens Energy Global GmbH & Co. KG, 2021. Disponível em: <https://assets.siemens-energy.com/siemens/assets/api/uuid:9877421f-1180-4ffa-9451-f0fe67354329/technical-paper-apm-4-0-e-.pdf>. Acesso em: 11 nov. 2021.
- [9] SUSTO, Gian Antonio. *et al.* (2015). Machine Learning for Predictive Maintenance: A Multiple Classifier Approach. Industrial Informatics, IEEE Transactions on. 11. 812-820. 10.1109/TII.2014.2349359.
- [10] KANAWADAY, Ameeth & SANE, Aditya. (2017). Machine learning for predictive maintenance of industrial machines using IoT sensor data. 87-90. 10.1109/ICSESS.2017.8342870.
- [11] HAYKIN, SIMON. Redes Neurais: Princípios e prática. Porto Alegre, RS: Bookman, 2001.
- [12] COGNILYTICA RESEARCH DATA ENGINEERING. Preparation, and Labeling for AI. 31 de janeiro de 2019. Doc. ID: CGR-DE100.

- [13] ABADI, Martín. *et al.* TensorFlow: A System for Large-Scale Machine Learning This paper is included in the Proceedings of the 12th USENIX Symposium on Operating Systems Design and Implementation (OSDI '16). TensorFlow: A system for large-scale machine learning. Disponível em: <https://www.usenix.org/system/files/conference/osdi16/osdi16-abadi.pdf>.
- [14] CHOLLET, François. *et al.* 2015. Keras. Disponível em: <https://github.com/fchollet/keras>.
- [15] MARTINS, Ricardo Bohadana. (2022) Predictive Maintenance [Source Code]. Disponível em: <https://github.com/ricardobohadana/PredictiveMaintenance>.
- [16] UZ, Fidan Boylu. Predictive Maintenance Modelling Guide Data Sets. 2016. Azure AI Gallery. Disponível em: <https://gallery.azure.ai/Experiment/Predictive-Maintenance-Modelling-Guide-Data-Sets-1>. Acesso em: 22 ago. 2021.
- [17] CYBENKO, G. Approximations by superpositions of sigmoidal functions, *Mathematics of Control, Signals, and Systems*, 2 (4), 303-314, 1989.
- [18] HORNIK, Kurt. Approximation Capabilities of Multilayer Feedforward Networks, *Neural Networks*, 4(2), 251–257, 1991. doi:10.1016/0893-6080(91)90009-T
- [19] HINTON, G. E. OSINDERO, S. TEH, Y. W. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*. 18 (7): 1527–1554, 2006.
- [20] HEATON, Jeff. Introduction to Neural Networks for Java, 2nd Edition. Heaton Research Inc. 2008.
- [21] PESSOA, Tiago. *et al.* Performance Analysis of Google Colaboratory as a Tool for Accelerating Deep Learning Applications. 2008. IEEE Access. PP. 1-1.10.1109/ACCESS.2018.2874767.
- [22] GATEC. Gestão agroindustrial. Projeto Manutenção Preditiva Análise de vibrações. Disponivel em: <https://docplayer.com.br/9755487-Projeto-manutencao-preditiva-analise-de-vibracoes-www-gatec-com-br.html>. Acesso em: 22 dez. 2021.
- [23] RIO OIL & GAS EXPO AND CONFERENCE 2008, Rio de Janeiro. *Trabalho técnico...* BARABAS, Elizabeth. Ganhos com a manutenção preditiva baseada em dados de processo em tempo real. Disponível em: <https://www.osti.gov/etdeweb/servlets/purl/21197118>.