Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Departamento de Engenharia Elétrica

Trabalho de Conclusão de Curso

Diagnóstico preditivo em máquinas elétricas: análise com classificadores e técnicas de sobreamostragem

Juliana Batista Rosa de Souza

João Monlevade, MG
2022

Juliana Batista Rosa de Souza

Diagnóstico preditivo em máquinas elétricas: análise com classificadores e técnicas de sobreamostragem

Trabalho de Conclusão de Curso apresentado à Universidade Federal de Ouro Preto como parte dos requisitos para obtenção do Título de Bacharel em Engenharia Elétrica pelo Instituto de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto.

Orientadora: Profª. Drª. Sarah Negreiros de Carvalho Leite

Universidade Federal de Ouro Preto

João Monlevade

2022

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

S729d Souza, Juliana Batista Rosa de.
Diagnóstico preditivo em máquinas elétricas [manuscrito]: análise com classificadores e técnicas de sobreamostragem.. / Juliana Batista Rosa de Souza. - 2022.
32 f.: il.: color., gráf., tab..

Orientadora: Profa. Dra. Sarah Negreiros de Carvalho Leite.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Aplicadas. Graduação em Engenharia Elétrica .

1. Aprendizado do computador. 2. Controle preditivo. 3. Localização de falhas (Engenharia) - Manutenção. 4. Máquinas elétricas. 5. Máquinas - Manutenção e reparos. I. Leite, Sarah Negreiros de Carvalho. II. Universidade Federal de Ouro Preto. III. Título.

CDU 621.313

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

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE ENGENHARIA ELETRICA

FOLHA DE APROVAÇÃO

Juliana Batista Rosa de Souza

Diagnóstico preditivo em máquinas elétricas: análise com classificadores e técnicas de sobreamostragem

Monografia apresentada ao Curso de Engenharia Elétrica da Universidade Federal
de Ouro Preto como requisito parcial para obtenção do título de Bacharela em Engenharia Elétrica.

Aprovada em 13 de junho de 2022.

Membros da banca

Profa. Dra. Sarah Negreiros de Carvalho Leite - Orientadora - Universidade Federal de Ouro Preto
Prof. Dr. Wilingthon Guerra Zvietcovich - Universidade Federal de Ouro Preto
Prof. Dr. Fábio Lumertz Garcia - Instituto Federal de São Paulo

Profa. Dra. Sarah Negreiros de Carvalho Leite, orientadora do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 27/06/2022.

Documento assinado eletronicamente por **Sarah Negreiros de Carvalho Leite**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 27/06/2022, às 21:37, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0352654** e o código CRC **E4E7D0C4**.

R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35400-000
Telefone: (31)3808-0818 - www.ufop.br

Agradeço aos meus professores pelos ensinamentos, aos amigos e amigas pelo companheirismo em todos os momentos e em especial aos meus pais Maria dos Anjos e Adimar e irmãs Geovana e Laura, pelo amor incondicional e suporte ininterrupto.

"Se você tem conhecimento, deixe que os outros ascendam suas velas nele."
(Margaret Fuller)

Resumo

As máquinas elétricas são amplamente empregadas em processos de produção industrial. Dessa forma é de suma importância que haja mecanismos que façam o diagnóstico de uma falha inicial, enquanto as máquinas ainda estão em operação, para evitar uma falha mais significativa. Dentre as funções operacionais e administrativas, a produção e a manutenção apresentam grande relevância, uma vez que os produtos ou serviços precisam ser entregues com qualidade no tempo solicitado. Assim, os diferentes tipos de manutenção possuem uma enorme importância estratégica que reflete diretamente a nível operacional e logístico de uma empresa. A manutenção preditiva visa reduzir o tempo fora de serviço, perdas na produção, quebra dos equipamentos e ainda garante uma operação produtiva confiável e consistente dos sistemas industriais. Essa prática se dá a partir da coleta, monitoramento e análise de diversos parâmetros dos equipamentos. Neste trabalho é realizada a análise de dados sintéticos de uma base de dados pública, considerando medições em máquinas elétricas de qualidade baixa, média e alta. Foram propostos e implementados sistemas de identificação das máquinas elétricas que apresentavam falhas. Foram empregadas cinco técnicas de balanceamento de amostras entre as classes 'com falha' e 'sem falha', sendo: SMOTE (*Synthetic Minority Oversampling Technique*), ADASYN (*Adaptive Synthetic Sampling Method*) e *Borderline* SMOTE, exclusão de amostras da classe majoritária e, por último, foi considerado o caso sem balanceamento de amostras entre as classes. Os resultados mostraram a importância do equilíbrio das amostras entre as classes de máquinas 'com falha' e 'sem falha' para projetar o sistema de classificação. Também foram configurados dois classificadores: classificador linear baseado no método dos mínimos quadrados e redes neurais do tipo *Multilayer Perceptron* (MLP). A combinação entre as abordagens de balanceamento (ou não) das classes e os classificadores gerou 9 cenários de testes. O melhor desempenho (96,49% de acurácia) foi obtido combinando a técnica de sobreamostragem *Borderline* SMOTE com a rede MLP.

Palavras-chave: manutenção preditiva, aprendizado de máquina, identificação de falha, balanceamento de classes, classificação.

Abstract

Electric machines are extensively deployed in industrial production processes. Thus therefore it is of utmost importance to have mechanisms in place that can diagnose an initial failure while the machines are still in operation, to avoid a more significant failure. Among the operational and administrative functions, production and maintenance are of great relevance, since the products or services need to be delivered with quality in the required time. In this way, the different types of maintenance have an enormous strategic importance that reflects directly on the operational and logistical level of a company. Predictive maintenance aims to reduce out-of-service time, production losses, equipment breakdown, and also ensures a reliable and consistent productive operation of industrial systems. This practice is based on the collection, monitoring and analysis of various parameters of equipment. In this work, the analysis of synthetic data from a public database is performed, considering measurements on electrical machines of low, medium and high quality. Systems were proposed and implemented to identify the electric machines that presented failures. Five techniques of sample balancing between 'with fail' and 'without fail' classes were employed, electric machines of low, medium and high quality. Systems were proposed and implemented to identify the electric machines that presented failures. Five techniques of sample balancing between 'with fault' and 'without fault' classes were employed, as follows: SMOTE (Synthetic Minority Oversampling Technique), ADASYN (Adaptive Synthetic Sampling Method) and Borderline SMOTE, exclusion of samples from the majority class, and lastly, the case without balancing the samples between the classes. The results showed the importance of the classes of 'with fail' and 'without fail' machines in designing the classification system. Two classifiers were also configured: linear classifier based on the least-squares method and Multilayer Perceptron (MLP) neural networks. The combination between the class balancing approaches and the classifiers generated 9 test scenarios. The best performance (96.49% accuracy) was obtained by combining the Borderline SMOTE oversampling technique with the MLP network.

Keywords: Predictive maintenance, machine learning, fault identification, class balancing, oversampling, classification.

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

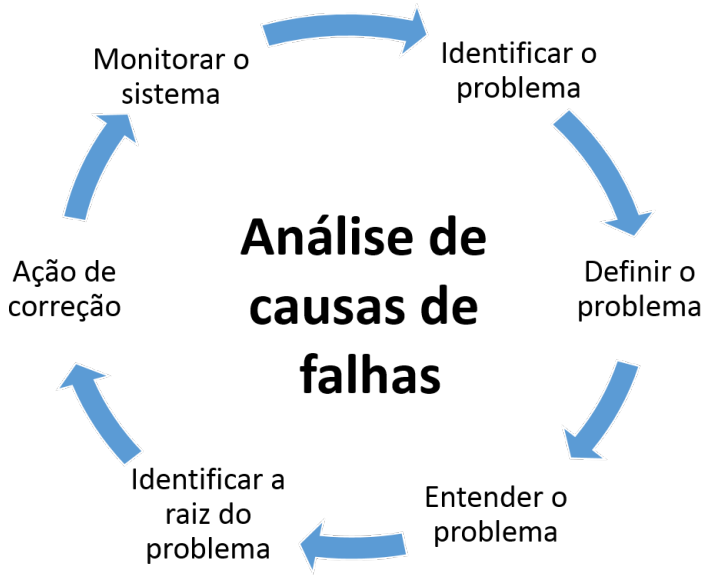
AUC	*Area Under the ROC Curve*
DEELT	Departamento de Engenharia Elétrica
IEEE	*Institute of Electrical and Electronics Engineers*
ICEA	Instituto de Ciências Exatas e Aplicadas
UFOP	Universidade Federal de Ouro Preto
UCI	*University of California, Irvine, School of Information and Computer Sciences*
MLP	*Multilayer Perceptron*
RBF	Rede de Funções de Base Radial
K	Kelvin
RPM	Rotações por minuto
Nm	Newton-metro
W	Watt
RNA	Rede Neural Artificial
SVM	Máquina de Vetores de Suporte
SMOTE	*Synthetic Minority Oversampling Technique*
KNN	*K-Nearest Neighbors*
ADASYN	*Adaptive Synthetic Sampling Method*
SGD	*Stochastic Gradient Descent*
ROC	*Receiver Operating Characteristic*
LDA	*Linear Discriminant Analysis*
QDA	*Quadratic Discriminant Analysis*
UDI	*Unique Device Identifier*
TanH	Tangente hiperbólica
ReLU	Unidade Linear Retificada

Sumário

1 INTRODUÇÃO

Em um mercado competitivo, diminuir o tempo de produção, reduzir os custos e manter a qualidade do produto adequada são características essenciais buscadas pelas indústrias (COSTA et al., 2019). Essas características são obtidas através da construção e implementação de estratégias que visam compreender as causas das falhas e na tomada de decisão, relacionadas às estratégias de manutenção. A Figura 1 ilustra um ciclo genérico de análise para identificar causas de falhas.

Figura 1 – Análise de causa de falha



Fonte: Adaptado de Vib Master (2021).

A manutenção atualmente conhecida é apresentada por Kardec e Nascif (2009) e é dividida em dois grupos:

- Manutenção planejada: É realizada com planejamento e não necessariamente corrige problemas. A manutenção planejada é dividida em três categorias:
  - Preventiva: Conjunto de ações que garantem o funcionamento do equipamento e que independe do estado operacional de seus componentes;
  - Preditiva: Ação preventiva baseada em conhecimento prévio dos parâmetros de funcionamento dos componentes do equipamento;
  - Detectiva: Manutenção que verifica possíveis falhas em sistemas de proteção dos equipamentos.

- Manutenção não planejada: Quando realizada sem planejamento prévio, direcionada para corrigir problemas. Também conhecida por manutenção corretiva, é dividida em duas categorias:
  - Ocasional: Realizada para reparar alguma falha que não compromete o funcionamento do equipamento;
  - Inesperada: Realizada para reparar defeitos repentinos durante a operação do equipamento.

Existe, ainda, a manutenção conhecida como Engenharia da Manutenção, que reúne os elementos de todos os tipos citados acima e presta suporte técnico de manutenção, dedicada a confirmar a rotina e implantar melhorias (KARDEC; NASCIF, 2009).

A manutenção preditiva pode ser usada na gestão da qualidade e auxiliar nas tomadas de decisão, como forma de otimizar a disponibilidade de máquina, redução de manutenções corretivas desnecessárias e, assim, reduzir os custos da produção (COSTA et al., 2019). A manutenção possui uma enorme importância estratégica que reflete diretamente no nível operacional e logístico da empresa (FUENTES et al., 2006). Nesse contexto, a manutenção de ativos requer um constante monitoramento do equipamento a fim de detectar e diagnosticar defeitos. Pois, é a partir da detecção de defeitos, que o plano de manutenção é posto em prática.(ZONTA et al., 2020).

A obtenção de dados por meio de monitoramento permite que a indústria desenvolva mecanismos na tomada de decisão e explore estes dados de diversas maneiras, de forma a agregar no capital da empresa (COSTA et al., 2019). Nos sistemas de monitoramento, a utilização de técnicas de processamento digital de sinais possibilita o desenvolvimento de sistemas de prevenção de manutenções não-programadas e paradas indesejadas, reduzindo custos e perdas evitáveis (SILVA, 2014). De acordo com Santos, Silva e Suetake (2012) os métodos de detecção de falhas com base em sistemas inteligentes são bastante difundidos. Dentre eles, o mais utilizado é a Rede Neural Artificial pela sua versatilidade para tratar soluções de diversos problemas, como predição e classificação de padrões (SILVA, 2014).

Detecção e diagnóstico de falha podem auxiliar todo o setor de manutenção de uma indústria de diversas maneiras. Por exemplo, a manutenção preditiva pode evitar uma parada não programada do equipamento diminuindo perdas (FUENTES et al., 2006). Utilizar métodos preditivos auxilia nessa tomada de decisão.

1.1 Objetivos

1.1.1 Objetivo geral

Identificar se uma máquina elétrica apresenta falha ou não considerando os atributos disponíveis no conjunto de dados analisado.

1.1.2 Objetivos específicos

- Analisar os atributos de máquinas elétricas operando normalmente e em condições de falha;
- Padronizar a ordem de grandeza dos atributos do conjunto de dados a serem utilizados pelos classificadores;
- Tratar o desbalanceamento de amostras entre as classes de máquinas: 'com falha' e 'sem falha' da base de dados;
- Implementar, treinar e avaliar o desempenho dos classificadores;
- Comparar o desempenho dos sistemas desenvolvidos combinando as técnicas de sobreamostragem e classificação.

2 REVISÃO BIBLIOGRÁFICA

No trabalho de Matzka (2020a) é apresentado e analisado um conjunto de dados de manutenção preditiva sintético, que simula condições reais, disponível no repositório de dados UCI Matzka (2020b). A principal contribuição foi o desenho de um modelo explicável e uma interface explicativa. Em uma de suas análises foi demonstrado que nem todas as características disponíveis no conjunto de dados, contribuem efetivamente para a detecção de falhas em máquinas e que, as variáveis 'torque' e 'velocidade de rotação', são dominantes na determinação de existência ou não de falha. O desempenho do modelo explicável foi avaliado utilizando 15 árvores de decisão com apenas 4 nós para facilitar a interpretação humana.Essas usaram 4 a 6 atributos do conjunto de dados para classificar os diferentes tipos de falha presentes. A falha por desgaste de ferramenta — tool wear foi o tipo mais difícil de identificar. Isso se deve ao baixo número desse tipo de falha,da natureza aleatória da substituição ou falha da ferramenta em tempo hábil. Observação frequente (e obstáculo para o aprendizado de máquina) em situações reais de manutenção preditiva. Já a interface explicativa em que cada atributo foi normalizado para um valor esperado de 0 e um desvio padrão de 1 e o usuário recebe a explicação baseada no recurso com os desvios absolutos máximos. E concluiu-se que as explicações fornecidas pelas árvores de decisão tendem a ser de maior qualidade, mas em um número considerável de casos não fornecem nenhuma explicação. Por outro lado, os desvios de recursos normalizados fornecem explicações de uma qualidade explicativa consistente, mas um pouco menor.

Utilizando o mesmo conjunto de dados de Matzka (2020b), Sridhar e Sanagavarapu (2021) analisaram e trataram o desbalanceamento entre as classes da base de dados. Os algoritmos de aprendizagem de máquina baseados em classificação, geralmente tem um melhor desempenho com uma distribuição equitativa de amostras entre as classes. Utilizando várias técnicas de sobreamostragem sendo elas *SMOTE*, *ADASYN*, *Borderline SMOTE* e métodos híbridos *SMOTE+Tomek*, *SVM+SMOTE*, *SMOTE+ENN*, o número de amostras de defeito ou falha, que corresponde a classe minoritária com 339 amostras do universo de 10 mil amostras, são replicadas para compensar o desequilíbrio. Após essa etapa vários classificadores de aprendizado de máquina sendo eles *Random Forest*, *Decision tree*, *XGBoost*, *Adaboost*, *k-NN*, *LDA*, *QDA*, *Gaussian Naive Bayes*, *Passive Aggressive* são utilizados para avaliar o conjunto de dados sobreamostrado com cada uma das técnicas utilizadas. O equilíbrio entre Precisão e *Recall* em caso de sobreamostragem também é analisado. Através das curvas *ROC* (*Receiver Operating Characteristic*), que avalia a qualidade da saída do classificador, notou-se que a acurácia inicial de 0,818 passou a ser 0,882 quando utilizado métodos de sobreamostragem.

Já Ferreira et al. (2019) utilizando uma outra base de dados com informações sobre a vibração em motores elétricos. Propuseram um sistema de manutenção preditiva empregando

diversas técnicas de aprendizado de máquinas, tais como regressão logística, *Support Vector Machine*, redes neurais artificiais, *random forest*, Naive Bayes e árvores de decisão aprimorada.

Em (SILVA, 2014), uma metodologia de detecção e classificação de falhas em motores de indução trifásica ligados na rede elétrica é proposta. Foram realizados ensaios com diferentes motores operando sob várias condições de desequilíbrio de tensão e diferentes regimes de torque e com vários tipos de falha. Durante os ensaios foram medidas as correntes do estator. Utilizando redes neurais do tipo *Perceptron Multilayer* (MLP), mapas auto-organizáveis e Kohonen e redes de funções de base radial conseguiu-se detectar a presença ou não de falhas nos rolamentos do estator e do rotor.

No trabalho de Carvalho et al. (2019), uma revisão sistemática da literatura sobre os métodos de aprendizado de máquina aplicados em manutenção preditiva é apresentada. Foram revisados 14 artigos publicados no *IEEEXplore Digital Library1* e 9 no *ScienceDirect2* a partir de 2009. Os autores verificaram uma tendência em aplicar os seguintes métodos de aprendizado de máquina: 37% dos trabalhos utilizam *Random Forest*, 27% Redes Neurais Artificiais (RNA), 25% Máquinas de Vetores de Suporte (SVM) e 13% *k-means*.

3 METODOLOGIA

A metodologia do sistema para detecção de falhas em máquinas elétricas é ilustrada na Figura 2 e estruturada em três partes:

1. Pré-processamento e adequação da base de dados contendo as características de máquinas elétricas.

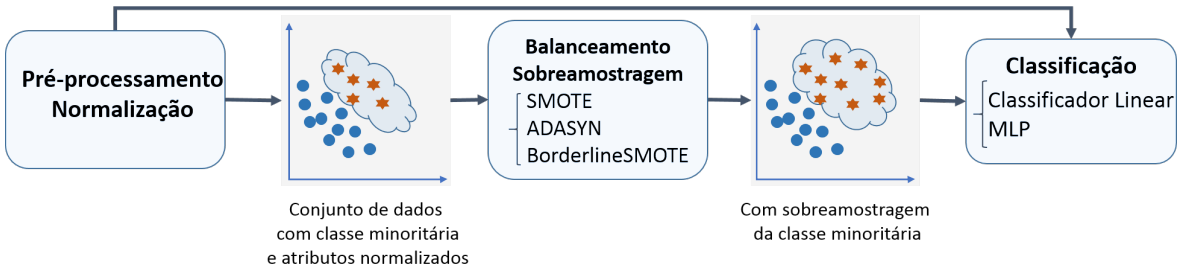
2. Balanceamento entre o quantitativo da classe minoritária de máquinas com falha e da classe majoritária de máquinas sem falha, empregando técnicas de sobreamostragem, como SMOTE (do inglês, *Synthetic Minority Oversampling Technique*), ADASYN (do inglês, *Oversample using Adaptive Synthetic*) e *Borderline SMOTE*. Para fins de comparação, também foi considerado o caso sem sobreamostragem, realizando o balanceamento das classes pela eliminação aleatória das amostras de máquinas sem falhas e o caso em que as classes foram usadas sem qualquer tratamento (classes desbalanceadas).

3. Classificação para identificar se a máquina apresenta ou não falha. Nesta etapa foram empregadas duas estruturas de classificação: uma linear — classificador linear baseado no método dos mínimos quadrados —, e uma rede neural artificial do tipo *Multilayer Perceptron* (MLP).

Cada uma destas etapas é tratada de maneira pormenorizada ao longo deste Capítulo.

No desenvolvimento deste trabalho, utilizou-se o *Google Colaboratory*. Também conhecido como *Google Colab*, que é uma ferramenta em nuvem do *Google Research* que permite criar e executar códigos na linguagem Python. Nesta plataforma é possível rodar os programas diretamente do navegador de forma simples, rápida e gratuita (GOOGLE COLAB,).

Figura 2 – Arquitetura do sistema.



Fonte: Adaptado de Sridhar e Sanagavarapu (2021).

3.1 Descrição do conjunto de dados

As análises deste trabalho empregaram a base de dados de manutenção preditiva disponível em (MATZKA, 2020b). Trata-se de um conjunto de 10 mil amostras de dados sintéticos, que refletem dados reais para realizar a manutenção preditiva. Uma vez que conjuntos de dados reais são de difícil obtenção e publicação para fins de estudos.

A Tabela 1 resume as informações apresentadas para cada amostra da base. A entrada "Falha de Máquina" indica se a máquina apresenta alguma falha ou não. São consideradas falhas de cinco naturezas: desgaste da ferramenta, dissipação de calor, falha de energia, falha de sobre tensão, falhas aleatórias.

Esta entrada "Falha da Máquina" assume o valor 1 quando pelo menos um dos tipos de falha ocorre. Na base de dados, há 339 amostras de máquinas com falhas e 9661 amostras de máquinas sem falhas. Este desbalanceamento entre as classes precisa ser tratado, a fim de garantir o funcionamento dos algoritmos de classificação.

Tabela 1 – Descrição dos atributos do conjunto de dados.

Atributos das máquinas		**Descrição dos atributos**
Qualidade	L - Baixo	50% dos máquinas do conjunto
	M - Médio	30% das máquinas do conjunto
	H - Alto	20% das máquinas do conjunto
Temperatura do ar [K]		Temperatura atmosférica
Temperatura do processo [K]		Temperatura de operação
Velocidade de rotação [rpm]		Velocidade a uma potência de 2860W
Torque [Nm]		Torque da máquina
Desgaste do equipamento	L - Baixo	2 minutos
	M - Médio	3 minutos
	H - Alto	5 minutos
Falha de Máquina		Sem Falha - 0 ou Com Falha - 1

Fonte: Adaptado de Sridhar e Sanagavarapu (2021).

3.2 Pré-processamento

3.2.1 Exploração dos dados

A Figura 3 ilustra a forma como os dados são organizados no conjunto de dados. Pode-se verificar que há 11 colunas referentes aos atributos da máquina. As três primeiras fazem referência aos dados de identificação da máquina. Em que: numeração, *UDI* e *Product ID*, que se referema nu meração da amostra, identificação da máquina e número de série da máquina, respectivamente. As seis variáveis seguintes apresentam os dados medidos e as informações das máquinas: *Type* se refere à qualidade da máquina, *Air temperature* indica a temperatura ambiente, *Process temperature* indica a temperatura de processo, *Rotational speed* se refere à velocidade de

Figura 3 – Conjunto de dados - algumas amostras

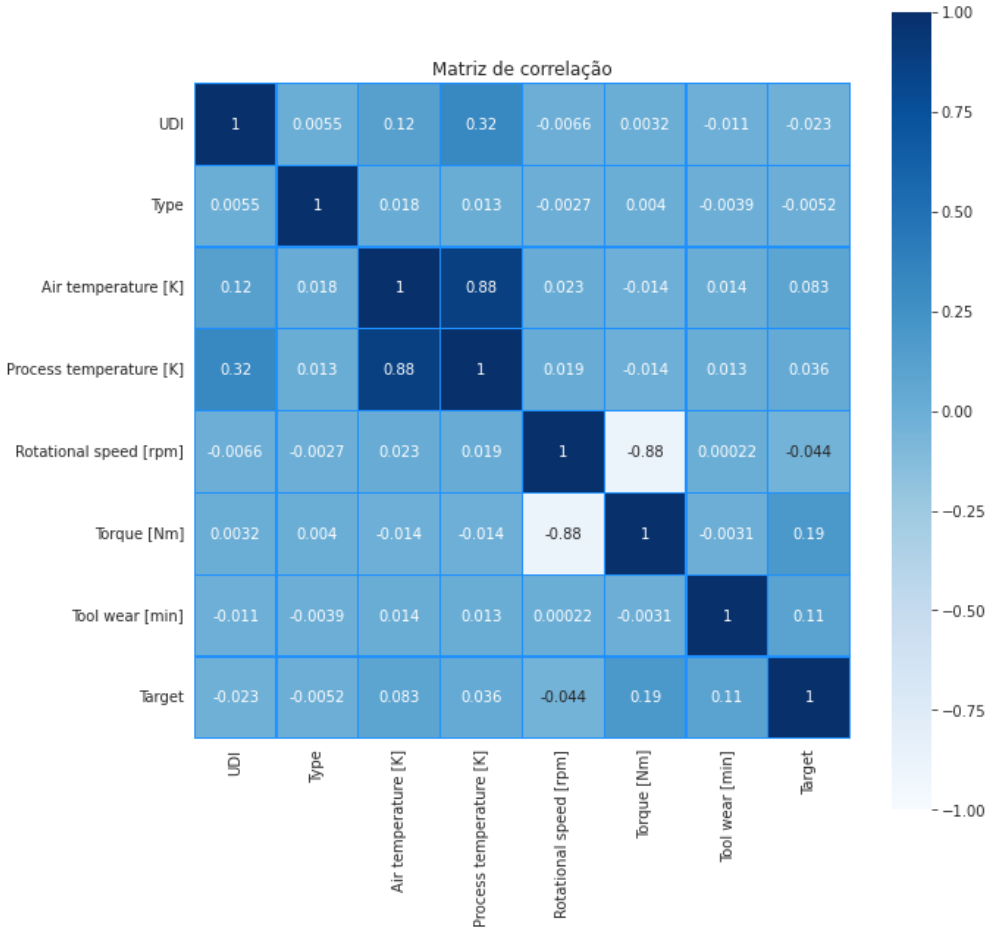
Identificação da máquina | Atributos da máquina | Informações sobre falha

	UDI	Product ID	Type	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Target	Failure Type
9013	9014	H38427	H	297.2	308.0	1435	43.0	202	0	No Failure
9014	9015	M23874	M	297.2	308.1	1340	61.3	207	1	Overstrain Failure
9015	9016	L56195	L	297.2	308.1	1431	49.7	210	1	No Failure
9016	9017	L56196	L	297.3	308.2	1396	52.7	212	1	Overstrain Failure
9017	9018	M23877	M	297.3	308.1	1369	52.0	214	0	No Failure
9018	9019	L56198	L	297.3	308.1	1615	35.4	217	1	Tool Wear Failure

Fonte: Da autora.

rotação, *Torque* indicando o torque da máquina e *Tool wear* que mostra o desgaste da máquina. As duas últimas colunas: *Target* apresenta se a máquina possui ou não falha, trata-se, portanto, do vetor de rótulos e, *Failure Type* se refere ao tipo de falha apresentado.

Figura 4 – Matriz de correlação dos atributos.



Fonte: Da autora.

Para analisar a relação entre as variáveis e delas com o vetor de rótulo (target) foi calculada a matriz de correlação dos dados, mostrada na Figura 4. Nesta análise, quanto mais próximo de +1 ou -1 o valor da célula, mais relação tem as variáveis especificadas no eixo x e y da matriz de correlação. E quanto mais próximo de zero, mais descorrelacionadas são as variáveis. Desta

análise, se observa que os atributos *Torque* e *Rotation Speed* apresentam uma alta correlação negativa (-0.88), ou seja, são variáveis inversamente proporcionais. De fato, para uma potência constante fixada em 2860 W, a Equação 3.1 ilustra essa relação:

$$Power = Torque \times RotationSpeed \tag{3.1}$$

Também é possível notar que os atributos *Air Temperature* e *Process Temperature* apresentam tom escuro de azul e uma correlação positiva igual a $0,88$ o que pode ser compreendida pelas leis da termodinâmica, uma vez que a temperatura do ambiente influência diretamente na temperatura do processo.

Desta análise, verificou-se que as variáveis em geral apresentam baixa correlação com o rótulo, sendo *Torque* e *Tool Wear* as que apresentam maiores valores.

Na sequência, verificou-se que dentre as colunas de características, os atributos *UDI* e *Product ID* contém valores únicos para cada amostra, que não contribuem no processo de classificação e identificação de máquinas com ou sem falhas. Assim, estas colunas foram excluídas.

Por fim, os símbolos H, M e L da coluna *Type* que categorizam a qualidade de cada máquina em Alto [H], Médio [M] e Baixo [L] foram substituídos pelos números 0, 2 e 1, respectivamente. A Figura 5 mostra um trecho do conjunto de dados após as modificações realizadas.

Figura 5 – Conjunto de dados modificado.

	Type	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]	Target	Failure Type
9013	0	297.2	308.0	1435	43.0	202	0	No Failure
9014	2	297.2	308.1	1340	61.3	207	1	Overstrain Failure
9015	1	297.2	308.1	1431	49.7	210	1	No Failure
9016	1	297.3	308.2	1396	52.7	212	1	Overstrain Failure
9017	2	297.3	308.1	1369	52.0	214	0	No Failure
9018	1	297.3	308.1	1615	35.4	217	1	Tool Wear Failure

Fonte: Da autora.

3.2.2 Normalização das entradas

A normalização é o processo de dimensionar amostras individuais para ter uma norma unitária (SCIKIT LEARN, a). Neste trabalho, a normalização *MinMaxScaler* foi aplicada no conjunto de dados já modificado da Figura 5, nas cinco características que correspondem a grandezas medidas com valores pertencentes ao conjunto dos reais, a saber: *Air temperature [K], Process temperature [K], Rotational speed [rpm], Torque [Nm], Tool wear [min]*. Essa técnica reescala os valores mínimo e máximo do conjunto de dados, de modo que pertençam ao intervalo zero e um, sem distorcer as diferenças nas faixas de valores. A expressão que descreve esse método de normalização é dada por:

$$X_c = \frac{X - X_{min}}{X_{max} - X_{min}} \tag{3.2}$$

sendo X_c a saída escalada, X a amostra tratada, X_{max} o maior valor assumido pela variável e X_{min} o menor valor assumido pela variável. Esta abordagem não trata os valores extremos *outliers*, sendo adequada para atributos que apresentam baixo desvio padrão (SCIKIT LEARN, d). Após a normalização dos dados, a matriz de dados foi dividida em uma matriz **X** de 6 colunas e 10 mil linhas, contendo os atributos que caracterizam as máquinas. E a coluna *Target*, que indica a presença ou não de falha, foi transformada em um vetor **y** com 10 mil linhas. Esses *arrays* **X** e **y** são as entradas do classificador. A Figura 6 apresenta a matriz de características **X** antes e depois da normalização.

Figura 6 – Matriz de características **X** original e normalizada.

X - Original:

	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]
9013	297.2	308.0	1435.0	43.0	202.0
9014	297.2	308.1	1340.0	61.3	207.0
9015	297.2	308.1	1431.0	49.7	210.0
9016	297.3	308.2	1396.0	52.7	212.0
9017	297.3	308.1	1369.0	52.0	214.0

X - Normalizado:

	Air temperature [K]	Process temperature [K]	Rotational speed [rpm]	Torque [Nm]	Tool wear [min]
9013	0.206522	0.283951	0.155413	0.538462	0.798419
9014	0.206522	0.296296	0.100116	0.789835	0.818182
9015	0.206522	0.296296	0.153085	0.630495	0.830040
9016	0.217391	0.308642	0.132712	0.671703	0.837945
9017	0.217391	0.296296	0.116997	0.662088	0.845850

Fonte: Da autora.

3.3 Balanceamento das classes

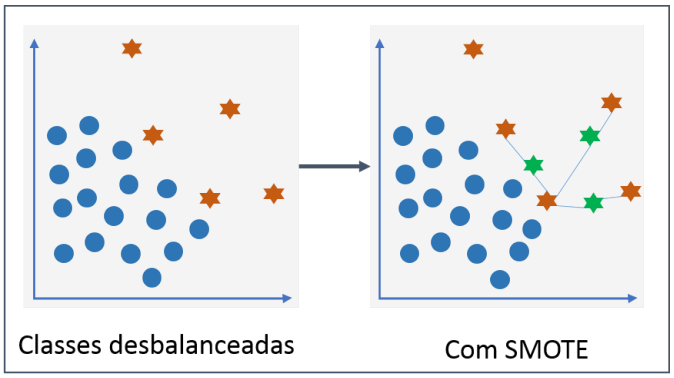
Tratar o desbalanceamento entre as classes da base de dados é essencial para garantir um bom desempenho da etapa de classificação. Na base de dados usada, há 9661 amostras de dados da classe majoritária (máquinas sem falhas) e somente 339 amostras da classe minoritária (máquinas com falhas). O balanceamento entre as classes foi tratado com duas abordagens:

1. Eliminação aleatória de dados da classe com amostras majoritárias de modo a equiparar a quantidade de amostras. Nesta abordagem foram selecionadas as amostras que seriam utilizadas para o treinamento e a validação do classificador. Desta forma, foram empregadas todas as 339 amostras de máquinas com falha que havia na base e realizou-se um sorteio a fim de escolher 339 amostras dentre as 9661 de máquinas sem falha. Isto resultou em 678 amostras selecionadas (das 10 mil disponíveis). Deste montante, 80% foram utilizadas para o treinamento dos classificadores e 20% foram empregadas na etapa de validação.

2. Emprego de técnicas de sobreamostragem para gerar amostras da classe minoritária. Nesta segunda abordagem, foram testadas três estratégias de sobreamostragem:

    - SMOTE: Método baseado no algoritmo KNN, em que a distância entre qualquer um de seus vizinhos é ponderada por um valor arbitrário e adicionado à amostra para gerar uma nova amostra. A Figura 7 indica esse processo de criação de novas amostras:

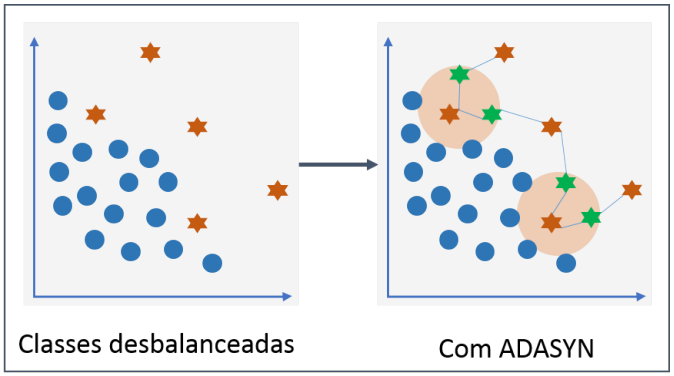
Figura 7 – Técnica SMOTE.



Da autora.

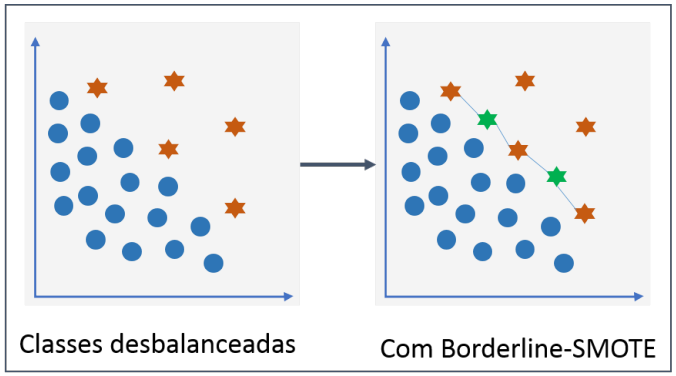
    - ADASYN: Trata-se de um algoritmo adaptativo que gera novas amostras priorizando as amostras da classe minoritária em áreas de fácil aprendizado - quando possuem menos vizinhos da classe majoritária, conforme ilustra a Figura 8.
    - Borderline SMOTE: Este método identifica as amostras da classe minoritária ao longo da fronteira de separação entre as classes (minoritária/ majoritária) e usa essas amostras para gerar novas, conforme mostra a Figura 9.

Figura 8 – Técnica ADASYN.



Da autora.

Figura 9 – Técnica *Borderline* SMOTE.



Da autora.

3.4 Classificação

A classificação é a etapa final do sistema capaz de identificar, a partir das variáveis de entrada, se uma máquina apresenta ou não falha. Nesta etapa, o algoritmo mapeia as características de entrada ao rótulo da saída, especificando a qual classe aquelas entradas melhor correspondem. Este processo é dividido em duas etapas: treinamento e validação (HAN; PEI; KAMBER, 2011).

Na etapa de treinamento, o algoritmo 'aprende' a partir de uma partição do conjunto de dados, dito conjunto de dados de treinamento. A matriz de características ($\mathbf{X}$) apresenta N amostras de entrada e k características.

$$\mathbf{X} = \begin{bmatrix} x_{11} & \dots & x_{1k} \\ & \ddots & \\ x_{N1} & \dots & x_{Nk} \end{bmatrix}$$

Cada uma das N entradas são associadas a uma classe, gerando o vetor de rótulo de

classe **y**.

$$\mathbf{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}$$

No processo de treinamento, o algoritmo gera uma função $\mathbf{y} = f(\mathbf{X})$ que mapeia as k entradas de $\mathbf{X}$ na posição i, no rótulo $\mathbf{y}_i$ associado, sendo $i = 0, ..., N$ (HAN; PEI; KAMBER, 2011).

Na etapa de validação, o modelo treinado é utilizado para realizar a classificação e avaliar a precisão da predição do classificador. Nesta etapa, um subconjunto disjunto ao conjunto usado durante o treinamento é empregado. Desta forma, é possível estimar o desempenho do classificador e detectar a capacidade de generalização do sistema e possíveis casos de *overfitting*. Esse último que ocorre quando, por exemplo, um modelo se ajusta aos dados de treinamento, mas tem baixa acurácia com dados novos (HAN; PEI; KAMBER, 2011).

Para realizar a partição dos dados para as etapas de treino e de validação, foi empregada a função *train_test_split*() disponível na biblioteca do *Scikit-learn*. Ela divide o conjunto de dados para treinamento e validação, de maneira aleatória e disjunta, tentando manter o equilíbrio de amostras entre as classes. Foi configurado que 80% dos dados seriam utilizados na etapa de treinamento e os 20% restantes, na etapa de validação do modelo.

A seguir são definidos os conceitos particulares dos classificadores empregados no presente trabalho. Na sequência, são apresentados os métodos de avaliação do desempenho dos classificadores utilizados.

3.4.1 Classificador linear baseado no método dos mínimos quadrados

Uma das mais simples ferramentas utilizadas para discriminação de amostras de um conjunto de dados é o classificador linear. O baixo custo computacional e a simples implementação são as vantagens desta técnica que permite separar as classes de máquinas ‘com falhas’ das ‘sem falhas’.

O objetivo do classificador linear é determinar uma função linear capaz de separar as amostras de cada classe, de acordo com as características inseridas em sua entrada, com a menor taxa de erro possível. Existem diversos critérios para implementar o hiperplano separador das classes, neste trabalho, optou-se pela técnica conhecida como Método dos Mínimos Quadrados (THEODORIDIS; KOUTROUMBAS, 2009).

A saída **y** do classificador linear pode ser definida como:

$$\mathbf{y} = \mathbf{w}^T\mathbf{X} \tag{3.3}$$

sendo **w** o vetor de pesos que define o hiperplano separador das classes e **X** a matriz de características que contém as informações da máquina (ver Tabela 1).

As entradas do vetor de pesos **w** são determinadas na etapa de treinamento do classificador. Inicialmente, são atribuídos os rótulos no vetor **r**, que assume o valor de $+1$ para as amostras correspondentes à classe 'com falha' e -1 para as amostras referentes aos dados da classe 'sem falha'. Desta forma, o vetor de pesos pode ser determinado minimizando o erro quadrático médio pela seguinte expressão:

$$\mathbf{w} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{r} \tag{3.4}$$

sendo $\mathbf{X}^T\mathbf{X}$ a matriz de correlação de amostras de dados de entrada e $\mathbf{X}^+ = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T$ a matriz que corresponde a pseudo inversa de Moore-Penrose de **X** (THEODORIDIS; KOUTROUMBAS, 2009).

Após determinado o vetor de pesos **w**, o hiperplano de separação das classes está definido. Assim, para validar ou testar o sistema basta calcular:

$$\mathbf{y} = \mathbf{w}^T\mathbf{X} \tag{3.5}$$

se $\mathbf{y} > 0$ a amostra pertence a classe 'com falha', caso contrário, pertence a classe 'sem falha'.

3.4.2 Redes Neurais do tipo *Multilayer Perceptron*

As redes neurais artificiais podem atuar como poderosos classificadores não lineares. Elas foram propostas baseadas na forma de funcionamento dos neurônios biológicos, de modo a permitir que as máquinas pudessem 'aprender' e apresentar 'Inteligência Artificial'. As RNAs são constituídas por unidades básicas de processamento, os neurônios artificiais, que são modelos simplificados dos neurônios biológicos (LEITE, 2020). A Figura 10 ilustra um modelo de neurônio artificial proposto em 1943 por (MCCULLOCH; PITTS, 1943) com suas partes mais importantes: as sinapses, com pesos sinápticos $\mathbf{w}_n$ associados a cada entrada $\mathbf{X}_n$. Cada entrada do neurônio é ponderada pelo peso sináptico e o resultado é somado ($\mathbf{u}_k$) e submetido a uma função, dita função de ativação. A função de ativação, tem a finalidade de modelar o processo de sinapse biológico, ela é responsável por determinar se o neurônio artificial dispara ou não, e gerar uma resposta, a saída $\mathbf{y}_k$.

Em termos matemáticos têm-se:

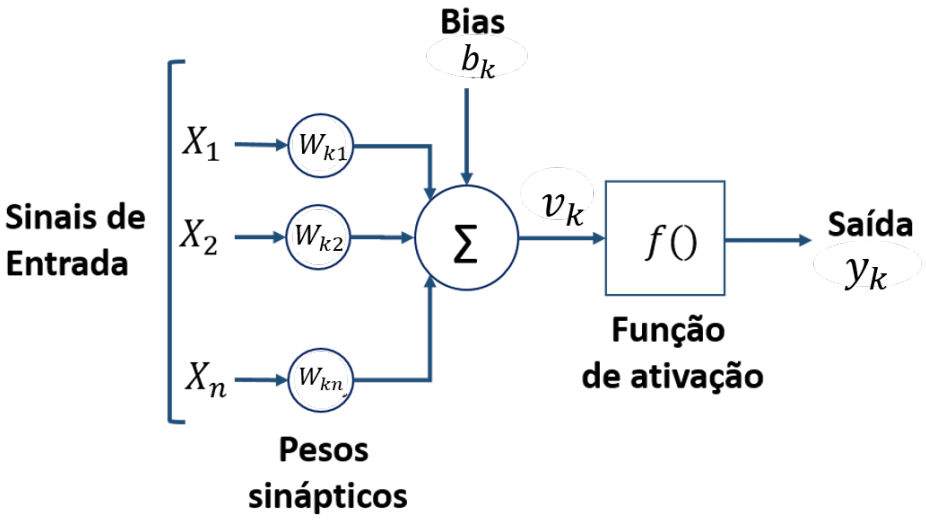
$$\mathbf{u}_k = \sum_{j=1}^{n} \mathbf{w}_{kj}\mathbf{X}_j \tag{3.6}$$

$$\mathbf{v}_k = \mathbf{u}_k + b_k \tag{3.7}$$

$$\mathbf{y}_k = f(\mathbf{v}_k) \tag{3.8}$$

onde: **u** é o valor da combinação linear dos sinais das n entradas **X** com os valores dos pesos sinápticos **w**. **v** é o potencial de ativação do neurônio k. b é o valor do *bias*, que tem o efeito de

Figura 10 – Modelo do neurônio artificial.



Adaptado de Haykin (2001).

diminuir ou aumentar o valor da entrada da função de ativação, caso seja positivo ou negativo. E **y** é a saída do neurônio.

As funções de ativação exercem o papel de garantir como e quando o neurônio artificial dispara, uma vez que nem sempre as entradas geram estímulo suficiente para tal. Tipicamente, elas atuam no intervalo de $[0, 1]$ ou $[-1, 1]$.

Nas redes neurais, o erro e_i pode ser definido por:

$$e_i = d_i - y_i \tag{3.9}$$

onde:

d_i é a saída desejada;

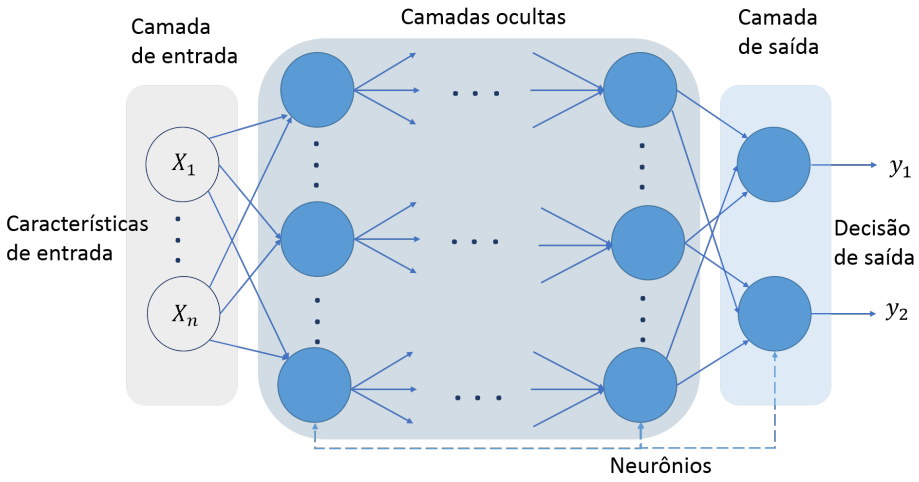
y_i é a saída estimada pela rede neural.

Para minimizar o erro de estimação, os pesos sinápticos da RNA devem ser ajustados de maneira que a saída estimada pela rede neural, seja cada vez mais próxima da saída desejada. Os ajustes dos pesos sinápticos w_i são feitos com: o valor do erro calculado, valor das variáveis de entrada e a taxa de aprendizado. Matematicamente, considerando o estímulo de entrada no instante i e o passo de treinamento η, tem-se:

$$w_{i+1} = w_i + \eta e_i x_i \tag{3.10}$$

A rede neural adotada neste trabalho foi a *Multilayer Perceptron* (MLP). Trata-se de uma estrutura do tipo *feedforward* multicamadas, conforme ilustra a Figura 11. Nas redes neurais MLP a informação vai de maneira unidirecional dos nós de entrada aos de saída, passando pelas camadas ocultas (HAYKIN, 2001).

Figura 11 – Rede neural *feedfoward* MLP.



Fonte:Adaptado de Haykin (2001).

Nessa rede, a camada de entrada recebe as características extraídas da base de dados. Cada entrada é ponderada pelo vetor de pesos sinápticos os quais são estimados durante a etapa de treinamento da rede neural artificial. A soma de cada entrada ponderada passa pela função de ativação definida em cada neurônio. Este processo se repete em todas as camadas ocultas. As saídas da última camada oculta são ponderadas e somadas pelos neurônios da camada de saída da rede neural, que após passar pela função de ativação, gera a saída estimada pela rede, para aquelas variáveis de entrada correspondente (NEVES, 2018).

As camadas da rede MLP são completamente conectadas, também são ditas ‘densas’, ou seja, cada neurônio se conecta a todos os outros neurônios da camada seguinte.

Outro tipo de camada que pode ser empregada na topologia das redes MLPs são as camadas de *dropout*. Estas camadas definem um percentual de neurônios, que são escolhidos aleatoriamente para serem desconectados durante o processo de treinamento dos pesos sinápticos da rede neural. Esta abordagem visa evitar o problema de sobreajuste da rede neural.

3.4.2.1 Treinando a rede MLP

Na etapa de treinamento da rede neural, deseja-se encontrar uma superfície capaz de separar as duas classes do problema, ou seja, máquinas ‘com falhas’ das ‘sem falhas’.

Primeiramente, é necessário definir a estrutura da rede neural. Não existe uma regra para determinar a topologia mais adequada para solucionar cada problema, geralmente a configuração dos parâmetros é determinada de modo empírico (LEITE, 2020).

Geralmente, os principais parâmetros a serem determinados são:

- Número de camadas ocultas,
- Número de neurônios em cada camada oculta,

- Função de ativação empregada em cada neurônio,
- Emprego ou não de camadas de *dropout* e percentual de apagamento de neurônios.

Neste projeto, o número de camadas ocultas testados e o número de neurônios variou entre 2 e 20.

A função de ativação deve ser configurada para cada neurônio da rede neural. As funções de ativação testadas nesse projeto foram:

- **Sigmoid**: a partir de uma curva em S executa classificações com múltiplas saídas. Seu intervalo é limitado entre 0 e 1, mas a função sofre com a dissipação do gradiente que estagna o aprendizado da rede em algumas circunstâncias;
- **TanH**: a função tangente hiperbólica também é do tipo sigmoidal, mas seu intervalo vai de -1 a +1. Ela também apresenta com o problema da dissipação do gradiente, entretanto, trata melhor a detecção de pequenas diferenças;
- **ReLU** - a função 'Unidade Linear Retificada' tem a vantagem de não ativar todos os neurônios ao mesmo tempo, desta forma ela gera uma rede esparsa exigindo menos processamento, por ser mais fácil de ser computada. A ReLU deve ser usada apenas nas camadas ocultas.

Para o treinamento da MLP e ajuste adequado dos pesos sinápticos e do bias, o algoritmo utilizado foi o *backpropagation*. Trata-se de um algoritmo supervisionado em que o treinamento ocorre em duas fases: *forward* e *backward*.

Na fase *forward* a amostra de dados entra na rede pela camada de entrada e é processada por cada camada oculta e passada para camada seguinte até que chegue na camada de saída que calcula a saída da RNA. Esse resultado é comparado com a saída desejada.

Na fase *backward* percorre o caminho inverso, ou seja, vai da camada de saída em direção à camada de entrada. Durante este processo os pesos sinápticos dos neurônios são ajustados de maneira iterativa com o objetivo de diminuir o erro. Esses ajustes ocorrem observando a taxa de aprendizagem até que a rede seja treinada ou ocorra um critério de parada.

Existem algumas maneiras de considerar as amostras da base de dados de treinamento durante a etapa de ajustes dos pesos sinápticos:

- **Modo *online***: Após cada amostra percorrer a rede, os pesos sinápticos são atualizados como um fluxo de aprendizado em tempo real. Esta abordagem é sensível a valores discrepantes e a taxa de aprendizagem deve ser mantida baixa.

- **Modo batelada ou em lote**: Os pesos sinápticos são atualizados após o processamento de todas as amostras de treinamento. Dessa forma, o ajuste dos pesos sinápticos é mais rápido e menos sensível a discrepâncias de valor. Nesse modo, o algoritmo de *backpropagation* considera a soma de todos os gradientes das amostras de treinamento.

- **Modo em minilote ou estocástico**: Esse modo combina características dos modos anteriores, pois faz a atualização dos pesos sinápticos após o processamento de um subconjunto com amostras aleatórias do conjunto de treinamento. Assim tem um baixo uso de memória e convergência mais rápida.

Neste trabalho foi empregado o método estocástico com *batch* de tamanho 32 e 64.

Após a configuração da arquitetura da rede neural MLP, para a compilação do modelo da rede neural é necessário definir um otimizador, uma função de perda e uma métrica. Esses hiperparâmetros vão guiar o ajuste do modelo no cálculo dos pesos sinápticos. O otimizador utilizado nesse projeto foi o *Adam*.

Dentre as funções de custo ou de perda mais comumente utilizadas, foram testadas:

- *mse*: erro quadrático médio;
- *mae*: erro médio absoluto;
- *binary_crossentropy*: entropia binária cruzada;

Para avaliar se o treinamento está satisfatório são utilizadas métricas. As mais frequentemente empregadas em problemas de classificação é a acurácia que exibe a porcentagem de acertos.

Por fim, para executar o treinamento é definido o número de épocas (*epochs*) ou de iterações que o modelo executará para determinar os pesos da superfície de separação das classes (THERRIEN, 1989).

3.4.2.2 Validando a rede MLP

A validação do modelo ocorre após a finalização do treinamento. Nessa fase o subconjunto com as amostras destinadas à validação é empregado nas seguintes funções da biblioteca *Keras* do Python:

- *model.evaluate*(): avalia a entrada de dados e retorna o valor da função de perda e a métrica.
- *model.predict*(): gera a saída da rede, ou seja, gera o resultado de classificação para o dado de entrada.

3.4.3 Métodos de avaliação dos classificadores

De acordo com Han, Pei e Kamber (2011), vários critérios podem ser utilizados para validar ou comparar os classificadores:

- Interpretabilidade: refere-se ao nível de compreensão e entendimento que é fornecido pelo classificador e por ser muito subjetivo é difícil de ser analisado;
- Robustez: é a habilidade do classificador de fazer predições corretas de dados, mesmo na presença de valores faltantes ou ruídos;
- Velocidade: refere-se ao custo computacional envolvido no uso e na geração de um classificador;
- Acurácia: refere-se à habilidade de um classificador fazer predições corretas do rótulo de classe de um dado novo - de uma amostra sem a informação de rótulo de classe.
- *Recall*: refere-se a habilidade do classificador de prever a classe pretendida de maneira correta
- Precisão: refere-se à proporção de identificações corretamente positivas, ou seja, quão bem o modelo trabalhou;

A acurácia citada acima é uma das mais utilizadas e corresponde à porcentagem de amostras do conjunto de testes que foram classificados de maneira correta pelo classificador. Uma ferramenta muito útil para avaliar a acurácia é a matriz de confusão. A partir dela é possível analisar o quão bem um classificador reconhece as classes das amostras. Para o problema proposto nesse trabalho, a matriz de confusão binária é a indicada para tal função. Nesse tipo de matriz são apresentados os valores positivos *versus* negativos como exemplificado na Tabela 2:

Tabela 2 – Matriz de Confusão Binária.

		Preditas	
Atual	CLASSE	**C1**	**C2**
	C1	Verdadeiro Positivo	Falso Negativo
	C2	Falso Positivo	Verdadeiro Negativo

Fonte: Da autora.

Sendo:

- Verdadeiro Positivo (VP): acontece quando a classe de interesse, presença de falha na máquina, é prevista corretamente;
- Falso Positivo (FP): acontece quando a classe de interesse é prevista incorretamente. E dito que possui falha e o correto seria sem falha;

- Verdadeiro Negativo (VN): ocorre quando a classe que não é de interesse, sem falha na máquina, é prevista corretamente;
- Falso Negativo (FN): ocorre quando a classe que não interessa é prevista incorretamente. É dito que não possui falha e o correto seria com falha.

A partir da matriz de confusão binária é possível calcular as seguintes métricas (SCIKIT LEARN, b):

Acurácia:

$$acurácia = \frac{VP + VN}{VP + FP + VN + FN} = \frac{predições\ corretas}{todas\ as\ predições} \tag{3.11}$$

Acurácia média: A acurácia média das iterações realizadas:

$$acurácia\ média = \frac{acurácia}{n^{\circ}\ de\ iterações} \tag{3.12}$$

Precisão:

$$precisão = \frac{VP}{VP + FP} \tag{3.13}$$

f-score: mostra o balanço entre a precisão e o *recall*:

$$f\_score = 2 * \frac{precisão * recall}{precisão + recall} \tag{3.14}$$

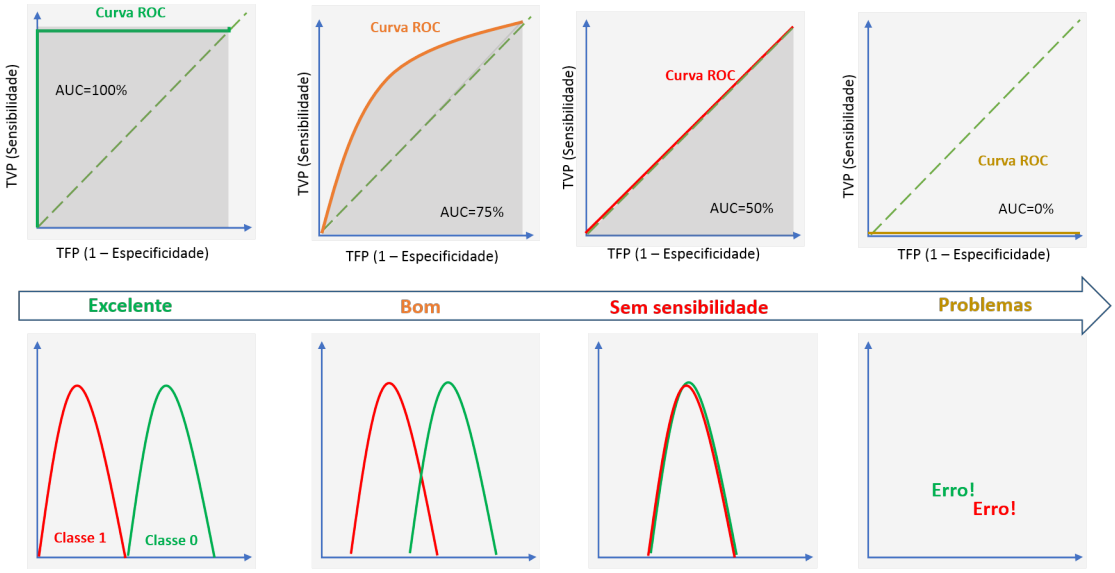
Existe também uma maneira gráfica de avaliar o desempenho de um modelo de aprendizado de máquina. A curva ROC (*Receiver Operating Characteristic*) avalia a qualidade da saída do classificador e o quão bem esse modelo consegue distinguir as duas classes.(SCIKIT LEARN, c)

A curva ROC possui dois parâmetros:

- Taxa de Verdadeiro Positivo, dado por: $\frac{VP}{VP+FN}$;
- Taxa de Falso positivo, dado por $\frac{FP}{FP+VN}$.

A Figura 12 abaixo ilustra da esquerda para a direita os possíveis comportamentos da curva ROC, do melhor para o pior:

Figura 12 – Entendendo a Curva ROC em uma imagem



Fonte: Adapatado de Glen (2019).

O eixo Y apresenta a taxa de Verdadeiro Positivo enquanto o eixo X apresenta a taxa de Falso Positivo. Isso indica que o canto superior esquerdo do gráfico é o ponto ideal da curva quando há uma taxa de falsos positivos de zero e uma taxa de verdadeiros positivos de um. É importante observar a inclinação da curva ROC, pois é ideal que a taxa de verdadeiros positivos seja maximizada e a taxa de falsos positivos minimizada. (SCIKIT LEARN, c)

A área sob a curva (AUC) agrega todos os limiares da curva ROC em um valor único. Seu valor varia de $0,0$ até $1,0$ e o limiar entre classes é $0,5$ representado pela linha tracejada em verde no gráfico da figura 12. Sendo que se o valor de AUC está no intervalo de $0,0$ até $0,5$ indica que o algoritmo classifica em uma classe e se o valor de AUC está no intervalo $0,5$ até $1,0$ classifica na outra classe. Quanto maior o valor de AUC, o melhor desempenho. É importante destacar que AUC trabalha com a precisão das classificações ao invés de seus valores absolutos, por isso, é uma métrica invariante em escala.

4 RESULTADOS E DISCUSSÃO

Neste Capítulo são apresentados os resultados dos sistemas de identificação das máquinas nas classes ‘com falha’ e ‘sem falha’. Todos os códigos foram escritos na linguagem *Python* e executados no ambiente do *Google Colaboratory*.

Foram testados dois classificadores distintos: classificador linear baseado no método dos mínimos quadrados e rede neural MLP. Também foram implementadas três técnicas de sobreamostragem e considerado o caso sem sobreamostragem e o caso de eliminação de amostras da classe majoritária. As combinações de técnicas de sobreamostragem e classificação resultaram em 9 cenários de testes.

Em todas as avaliações, a partição da base de dados foi de 80% das amostras para treinamento do classificador e 20% para validação do modelo treinado. Para avaliar e comparar os cenários foram utilizados o desvio padrão, taxa de acurácia média e a matriz de confusão. Em todos os testes, foi empregada a validação cruzada, utilizando múltiplas partições da base de dados.

4.1 Cenários de teste

Os Cenários 1-4 trazem informações referentes aos testes realizados com o classificador linear baseado no método dos mínimos quadrados (THEODORIDIS; KOUTROUMBAS, 2009) e sua combinação com as técnicas de balanceamento entre as classes, sendo sem amostragem (com eliminação das amostras da classe majoritária) e técnicas de sobreamostragem *SMOTE*, *ADASYN* e *Borderline SMOTE*. Já os Cenários 5-8 trazem informações referentes aos testes realizados com a rede neural MLP e sua combinação com as mesmas técnicas de balanceamento das amostras das classes.

1. **Cenário 1 - Classificador linear sem sobreamostragem**

   Nesse cenário não foi usada nenhuma técnica de sobreamostragem. Dessa forma, manteve-se as 339 amostras da classe ‘com falha’ e realizou-se um sorteio aleatório das amostras das amostras da classe ‘sem falha’ de modo a selecionar 339 amostras dentre as 9661 e obter um conjunto de dados balanceados. O rótulo das classes foi definido -1 para as amostras ‘sem falha’ e 1 para amostras ‘com falha’.

   O cálculo do vetor dos parâmetros do hiperplano separador foi feito utilizando a pseudo-inversa de acordo com o método de mínimos quadrados (THEODORIDIS; KOUTROUMBAS, 2009). Durante as iterações, para cada amostra, o valor predito era comparado com o valor de saída do classificador. Assim, a quantidade de acertos, o desvio padrão e a matriz

de confusão de cada iteração foram calculados. E ao final da rotina a acurácia média, o desvio padrão e a matriz de confusão média foram estimados.

2. **Cenário 2 - Classificador linear com *ADASYN***

   No Cenário 2 o conjunto de dados passou pelo processo de sobreamostragem utilizando o algoritmo da técnica *ADASYN*. Com isso, o conjunto de amostras 'com falha' foi aumentado ficando bem próximo ao número de amostras 'sem falha'. Passou de 339 amostras para 9655 amostras 'com falha', sendo que o conjunto de amostras 'sem falha' possui 9661 amostras. O *ADASYN* é um algoritmo dinâmico e adaptativo ao conjunto de dados, nos testes, foram utilizados os parâmetros de *default* da função *adasyn* da biblioteca *imblearn.over_sampling* do Python.

3. **Cenário 3 - Classificador linear com *SMOTE***

   No Cenário 3 o conjunto de dados passou pelo processo de sobreamostragem utilizando o algoritmo da técnica *SMOTE*. Com isso, a quantidade de amostras da classe 'com falha' foi igualada a da classe 'sem falha'. Sendo 9661 amostras para cada classe, com um total de 19322 amostras. Durante a aplicação do algoritmo *SMOTE* foi testada a influência da quantidade de vizinhos (*k_neighbors*) que influenciavam a geração das novas amostras, considerando os seguintes valores: 3, 5, 7 e 9.

4. **Cenário 4 - Classificador linear com *Borderline SMOTE***

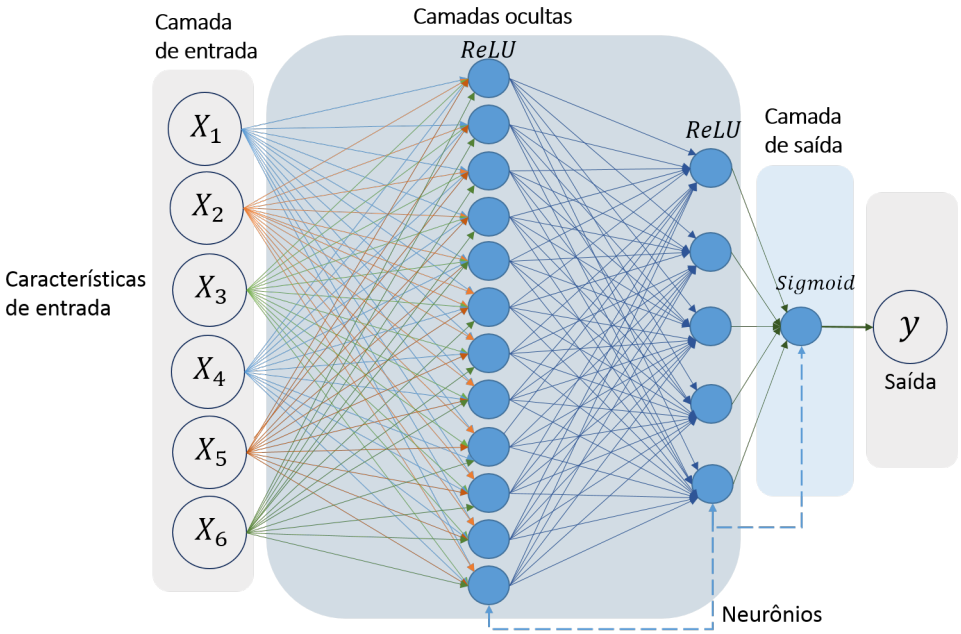
   No Cenário 4 a técnica de sobreamostragem empregada foi a *Borderline SMOTE*. Sendo assim, o conjunto de amostras 'com falha' foi sobreamostrado passando de 339 amostras para 9661. O *Borderline SMOTE* é uma variação do *SMOTE*, então decidiu-se por testar também a influência da quantidade de vizinhos (*k_neighbors*) escolhidos ao realizar a sobreamostragem. Os valores de *k_neighbors* avaliados foram: 3, 5, 7 e 9. Foram realizadas 10 mil iterações para estimar a matriz de confusão, acurácia média e desvio padrão.

5. **Cenário 5 - Rede neural MLP sem sobreamostragem**

   Nesse cenário o primeiro procedimento foi definir a topologia da rede neural MLP. Por se tratar de um processo empírico, uma série de testes foram realizados até se obter um desempenho equivalente ou superior ao do classificador linear. Tentou-se encontrar a estrutura mais simples da MLP capaz de solucionar o problema com alta confiabilidade. As configurações testadas estão apresentadas com mais detalhes na Figura 17 no Anexo A. A Figura 13 apresenta a estrutura da rede neural MLP selecionada após os testes e utilizada ao longo do trabalho.

   A rede MLP resultante possui 6 entradas, correspondentes as variáveis características das máquinas $X_1, X_2, X_3, X_4, X_5, X_6$. Todas as camadas da rede neural são completamente conectadas. A primeira camada oculta tem 12 neurônios e a segunda 5 neurônios. A função de ativação *ReLU* foi adotada. A camada de saída tem apenas um neurônio ativado pela

Figura 13 – Topologia da rede neural MLP utilizada.



Fonte: Da autora.

função de ativação *Sigmoid*. Esta topologia resultou em 155 parâmetros a serem estimados pesos sinápticos. Para a compilação e ajuste do modelo foi adotado o otimizador *Adam*, taxa de aprendizagem (*learning_rate)* de $0,01$, 500 épocas e *batch_size* igual a 64.

Após a configuração da Rede MLP, realizou-se um sorteio aleatório das amostras da classe 'sem falha' de modo a selecionar 339 amostras dentre as 9661 e obter um conjunto de dados balanceado. Esse conjunto foi dividido em amostras de treino e validação e utilizando a métrica *binary_acccuracy* a classificação da rede MLP foi avaliada em 2, 10 e 100 iterações.

6. **Cenário 6 - Rede neural MLP com *ADASYN***

   No Cenário 6 o conjunto de dados passou pelo processo de sobreamostragem das amostras com falha e foi utilizado o algoritmo da técnica *ADASYN*. Dessa forma, o conjunto com falha passou a ter 9661 amostras. Sendo o *ADASYN* um algoritmo dinâmico e adaptativo ao conjunto de dados que ele é exposto, nenhum de seus parâmetros configuráveis foi alterado e foram utilizados na sua configuração *default* em 2, 10 e 100 iterações.

7. **Cenário 7 - Rede neural MLP com *SMOTE***

   Nesse cenário de testes o conjunto de dados passou pelo processo de sobreamostragem utilizando a técnica *SMOTE*. Dessa forma, o conjunto de amostras com falha que tinha 339 amostras passou a ter 9661, a mesma quantidade de amostras que o conjunto sem falha. Dos parâmetros configuráveis do algoritmo *SMOTE* apenas a influência do número de vizinhos (*k_neighbors*) foi analisada. O valor de *k_neighbors* adotou os valores 3, 5, 7 e 9 que foram testados com 2, 10 e 100 iterações.

8. **Cenário 8 - Rede neural MLP com *Borderline SMOTE***

   Neste cenário o algoritmo de sobreamostragem do conjunto de amostras com falha foi o da técnica *Borderline SMOTE*. Nesse cenário o conjunto de amostras com falha foi sobreamostrado em 9661 amostras sem falha. Dos parâmetros configuráveis, apenas a influência do número de vizinhos foi avaliada e para isso *k_neighbors* assumiu os valores 3, 5, 7 e 9.

9. **Cenário 9 - Classificadores com classes desbalanceadas**

   Neste cenário foi realizado um teste a título de comparação, no qual o conjunto de dados foi submetido aos classificadores sem balanceamento entre as classes 'com falha'e 'sem falha'.

Para o classificador Linear finalizar a rotina de 10 mil iterações, foi gasto em média 13 minutos em contra partida, a Rede MLP necessitou em média 1 hora para realizar a rotina de 10 iterações. E isso se deve a complexidade elevada de uma rede neural e da simplicidade de um classificador linear. A Figura 14 apresenta os resultados obtidos em cada um dos cenários descritos.

Observa-se que a técnica de sobreamostragem *Borderline SMOTE* demonstrou excelente desempenho tanto para o classificador linear, empregando $k = 5$ vizinhos, quanto para a rede neural MLP com $k = 7$ vizinhos. A partir dos resultados, é possível perceber o quanto as técnicas de sobreamostragem baseadas em *SMOTE* melhoram a acurácia dos classificadores. Uma vez que os valores de acurácia dos cenários com sobreamostragem são maiores que nos cenários sem sobreamostragem.

O Cenário 2 que utilizou *ADASYN* teve uma acurácia média equivalente ao Cenário 1, no qual foram eliminadas as amostras da classe majoritária.

Ao observar os Cenários 1 e 5 é possível notar que a rede neural MLP tem um desempenho superior, mesmo sem a utilização de técnicas de sobreamostragem, se comparada ao classificador linear. Este desempenho superior da rede neural se mantém também nos testes sucessivos. De fato, o Cenário 8 com *Boderline SMOTE* usando $k = 7$ vizinhos e MLP foi o cenário com o melhor resultado geral.

Apesar dos valores de acurácia média do Cenário 9 para o Classificador Linear e Rede Neural MLP terem sido elevados, eles não podem ser considerados como o melhor desempenho. Pois, como o conjunto de dados é severamente desequilibrado, esse resultado pode esconder vícios ou aprendizagem equivocada. E para analisá-los utilizam-se as matrizes de confusão.

Figura 14 – Resultados dos cenários de teste.

Métodos de sobreamostragem		Métricas	Classificadores			
			Classificador Linear		Rede MLP	
			10 mil iterações		10 iterações	
SEM SOBREAMOSTRAGEM		Acurácia média	Cenário 1	81,56%	Cenário 5	86,54%
		Desvio padrão		3,204		3,222
ADASYN		Acurácia média	Cenário 2	80,48%	Cenário 6	93,58%
		Desvio padrão		0,618		0,707
SMOTE variando k	3	Acurácia média	Cenário 3	81,70%	Cenário 7	94,71%
		Desvio padrão		0,597		0,835
	5	Acurácia média		82,86%		94,28%
		Desvio padrão		0,596		0,761
	7	Acurácia média		82,72%		93,87%
		Desvio padrão		0,594		0,741
	9	Acurácia média		82,72%		89,52%
		Desvio padrão		0,589		13,631
BORDERLINE variando k	3	Acurácia média	Cenário 4	82,72%	Cenário 8	96,14%
		Desvio padrão		0,5907		1,694
	5	**Acurácia média**		**87,23%**		96,28%
		Desvio padrão		**0,521**		0,795
	7	**Acurácia média**		87,00%		**96,49%**
		Desvio padrão		0,5222		**0,373**
	9	Acurácia média		86,79%		95,84%
		Desvio padrão		0,529		0,967
SEM SORTEIO DE AMOSTRAS OU SOBREAMOSTRAGEM		Acurácia média	Cenário 9	96,77%	Cenário 9	97,66%
		Desvio padrão		0,295		0,449

Fonte: Da autora.

As matrizes de confusão e as curvas ROC (do inglês, *Receiver Operating Characteristic*) completam a análise desses cenários de melhor desempenho. Inicialmente, quando se comparam as matrizes de confusão dos cenários 4 e 9, a partir da Tabela 3a, nota-se que no Cenário 9, o Classificador Linear não acertou a classificação de nenhuma amostra da classe 'com falha'. Isso indica que ele 'aprendeu' de maneira equivocada a classificar majoritariamente para a classe 'sem falha' devido ao desbalanceamento entre as classes de dados. Mas, ao utilizar a técnica de sobreamostragem, esse desbalanceamento é corrigido. O classificador consegue ter um excelente desempenho classificando corretamente amostras de ambas as classes com uma taxa de acerto semelhante, como indica a Tabela 3b.

Ao observar a Figura 15a percebe-se que a curva ROC do Cenário 9 está sobre a reta de

Tabela 3 – Matrizes de confusão - Classificador Linear.

(a) Sem sorteio e sem sobreamostragem.

Cenário 9		Preditas	
	CLASSE	**Falha**	**Sem Falha**
Atual	**Falha**	0%	3,2%
	Sem falha	0%	96,8%

(b) Borderline SMOTE k=5.

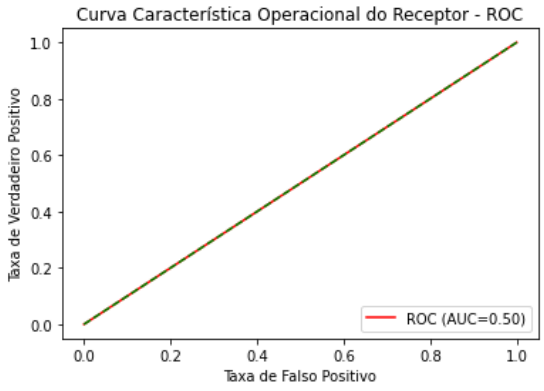
Cenário 4		Preditas	
	CLASSE	**Falha**	**Sem Falha**
Atual	**Falha**	85,3%	10,6%
	Sem falha	14,7%	89,4%

Fonte: Da autora.

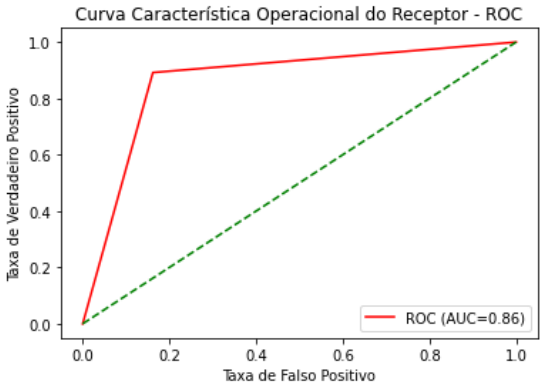
limiar e AUC = 0,5. Esse comportamento, reforça a informação dada pela matriz de confusão da tabela 3a e indica que o modelo sem o uso de sorteio de amostras e sem o uso de técnicas de sobreamostragem não é capaz de distinguir as classes das amostras. Porém, ao observar a Figura 15b, nota-se que a curva ROC do Cenário 4 está mais à esquerda do gráfico e apresenta AUC = 0,86, indicando que a taxa de falso positivo sofreu diminuição e que a taxa de verdadeiro positivo aumentou em relação ao observado no Cenário 9.

Figura 15 – Curva ROC - Classificador Linear.

(a) Sem sorteio e sobreamostragem - Cenário 9.



(b) Borderline SMOTE k=5 - Cenário 4.



Fonte: Da autora.

Ao comparar as matrizes de confusão dos Cenários 8 e 9, na Tabela 4.1 nota-se que a rede MLP foi capaz de identificar 76,9% das amostras da classe 'com falha' de maneira correta no Cenário 9, com taxa de falsas acusações de falha de 23,1%. Mas, na Tabela 4b, ao utilizar a técnica de sobreamostragem, a Rede Neural MLP passou a classificar corretamente 95,3% das amostras da classe 'com falha' e teve uma redução nas falsas acusações de falha para 4,7%.

Tabela 4 – Matrizes de confusão Rede Neural MLP.

(a) Sem sorteio e sem sobreamostragem.

Cenário 9		Preditas	
	CLASSE	**Falha**	**Sem Falha**
Atual	**Falha**	76,9%	1,8%
	Sem falha	23,1%	98,2%

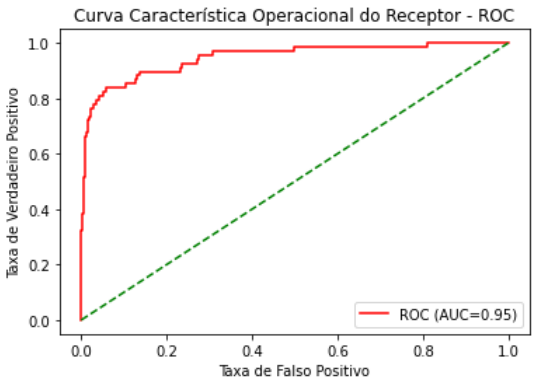
(b) Borderline SMOTE k=7.

Cenário 8		Preditas	
	CLASSE	**Falha**	**Sem Falha**
Atual	**Falha**	95,3%	2,2%
	Sem falha	4,7%	97,8%

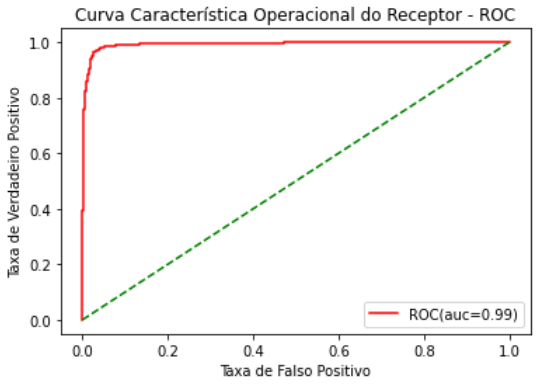
Fonte: Da autora.

Figura 16 – Curva ROC - Rede Neural MLP.

(a) Sem sorteio e sobreamostragem - Cenário 9.



(b) Borderline SMOTE k=7 - Cenário 8.



Fonte: Da autora.

Ao observar a Figura 16a nota-se que a a curva ROC do Cenário 9 está bem à esquerda no gráfico e apresenta AUC = 0,95. Esse comportamento, reforça a informação dada pelas matrizes de confusão da Tabela e indica que, mesmo sem o uso de sorteio de amostras e sem o uso de técnicas de sobreamostragem, o modelo é capaz de distinguir as classes das amostras de maneira muito eficaz. Porém, ao observar a Figura 16b, nota-se que a curva ROC do Cenário 8 está ainda mais à esquerda do gráfico e apresenta AUC = 0,99, indicando ser um modelo excelente e capaz de distinguir muito bem as classes das amostras. Em relação ao Cenário 9, teve a taxa de falso positivo diminuída e a taxa de verdadeiro positivo aumentada, chegando bem próximo a uma situação de desempenho ideal.

5 CONCLUSÕES

Este trabalho tratou de um sistema de diagnóstico preditivo de falhas em máquinas elétricas. Utilizou-se o Classificador Linear, a Rede Neural Artificial MLP e técnicas de sobreamostragem para analisar um conjunto sintético de dados de máquinas elétricas obtido no repositório público UCI - *Machine Learning Repository*, com o objetivo de identificar as máquinas elétricas que possuíam ou não falha e compreender o desempenho dos classificadores. Na etapa da classificação diversos testes foram realizados de maneira a configurar os parâmetros da Rede MLP e ajustar os dados para as particularidades de cada classificador.

Como a base de dados apresenta 9661 amostras da classe 'sem falha' e somente 339 amostras da classe 'com falha', foi indispensável balancear as classes de dados. Para isto, adotou-se quatro estratégias, excluir de maneira aleatória as amostras de dados da classe majoritária, utilizar as técnicas de sobreamostragem *SMOTE, Borderline SMOTE* e *ADASYN*.

O melhor resultado foi obtido empregando o método de *Borderline SMOTE* para ambos os classificadores testados. Sendo a acurácia média das 10 mil iterações de 87,24% usando o classificador linear e 96,49% para 10 iterações empregando a Rede MLP. No caso da rede neural, empregaram-se menos iterações para gerar o desempenho médio devido ao elevado tempo de execução.

Diante dos resultados, observa-se que o modelo gerado foi capaz de solucionar o problema de identificação das máquinas com falhas a partir de suas características, sendo uma técnica eficaz para colaborar nas rotinas de manutenção.

Como trabalhos futuros, propõe-se a implementação de um sistema de classificação multiclasses, que permita identificar o tipo de falha nas máquinas identificadas 'com falhas'. Também pode ser aprofundada a análise de outras técnicas de sobreamostragem.

REFERÊNCIAS

CARVALHO, T. P. et al. A systematic literature review of machine learning methods applied to predictive maintenance. *Computers & Industrial Engineering*, Elsevier, v. 137, p. 106024, 2019. 5

COSTA, G. K. da et al. Comparação de modelos de previsão voltados à manutenção na industria automobilistica a partir de dados de inspeção de qualidade. 2019. 1, 2

FERREIRA, R. H. d. M. S. et al. Sistema de manutenção preditiva para motores elétricos utilizando de sinais de vibração e aprendizado de máquina. Universidade Federal de Campina Grande, 2019. 4

FUENTES, F. F. E. et al. Metodologia para inovação da gestão de manutenção industrial. Florianópolis, SC, 2006. 2

GLEN, S - TECHTARGET - DATA SCIENCE CENTRAL -. *ROC Curve Explained in One Picture*. 2019. Disponível em: <https://www.datasciencecentral.com/roc-curve-explained-in-one-picture/>. Acesso em: 23 jun 2022. 21

GOOGLE. *Colaboratory - Perguntas Frequentes*. Disponível em: <https://research.google.com/colaboratory/intl/pt-BR/faq.html>. Acesso em: 4 mai 2022. 6

HAN, J.; PEI, J.; KAMBER, M. *Data mining: concepts and techniques*. [S.l.]: Elsevier, 2011. 12, 13, 19

HAYKIN, S. *Redes neurais: princípios e prática*. [S.l.]: Bookman Editora, 2001. 15, 16

KARDEC, A. P.; NASCIF, J. Manutenção: Função estratégica–3. ed, ver. e ampl. *Rio de Janeiro: Qualitymark: Petrobras*, 2009. 1, 2

LEITE, P. D. S. N. de C. *Ciência de Dados - Redes Neurais Artificiais e Deep Learning*. João Monlevade, MG, Brasil, 2020. 14, 16

MATZKA, S. Explainable artificial intelligence for predictive maintenance applications. In: IEEE. *2020 Third International Conference on Artificial Intelligence for Industries (AI4I)*. [S.l.], 2020. p. 69–74. 4

MATZKA, S. *UCI Machine Learning Repository*. 2020. Disponível em: <https://archive.ics.uci.edu/ml/datasets/AI4I+2020+Predictive+Maintenance+Dataset>. 4, 7

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943. 14

NEVES, S. A. d. Técnicas de aprendizado de máquina aplicadas a classificação da qualidade de pavimentos asfálticos utilizando smartphones. 2018. 16

SANTOS, F. M. d. C.; SILVA, I. N. d.; SUETAKE, M. Sobre a aplicação de sistemas inteligentes para diagnóstico de falhas em máquinas de indução-uma visão geral. *Sba: Controle & Automação Sociedade Brasileira de Automatica*, SciELO Brasil, v. 23, p. 553–569, 2012. 2

SCIKIT LEARN. *Dados de pré-processamento*. Disponível em: <https://scikit-learn.org/stable/modules/preprocessing.html#normalization>. Acesso em: 12 mai 2022. 9

SCIKIT LEARN. *Métricas e pontuação: quantificando a qualidade das previsões*. Disponível em: <https://scikit-learn.org/stable/modules/model_evaluation.html#confusion-matrix>. Acesso em: 2 mai 2022. 20

SCIKIT LEARN. *Receiver Operating Characteristic (ROC)*. Disponível em: <https://scikit-learn.org/stable/auto_examples/model_selection/plot_roc.html>. Acesso em: 23 jun 2022. 20, 21

SCIKIT LEARN. *sklearn.preprocessing.MinMaxScaler*. Disponível em: <https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html#sklearn.preprocessing.MinMaxScaler>. Acesso em: 12 mai 2022. 10

SILVA, L. R. B. *Classificação de falhas em maquinas eletricas usando redes neurais, modelos wavelet e medidas de informação*. Dissertação (Mestrado) — Universidade Tecnológica Federal do Paraná, 2014. 2, 5

SRIDHAR, S.; SANAGAVARAPU, S. Handling data imbalance in predictive maintenance for machines using smote-based oversampling. In: IEEE. *2021 13th International Conference on Computational Intelligence and Communication Networks (CICN)*. [S.l.], 2021. p. 44–49. 4, 6, 7

THEODORIDIS, S.; KOUTROUMBAS, K. Chapter 3. linear classifiers. In: ____. [S.l.: s.n.], 2009. p. 91–150. ISBN 9781597492720. 13, 14, 22

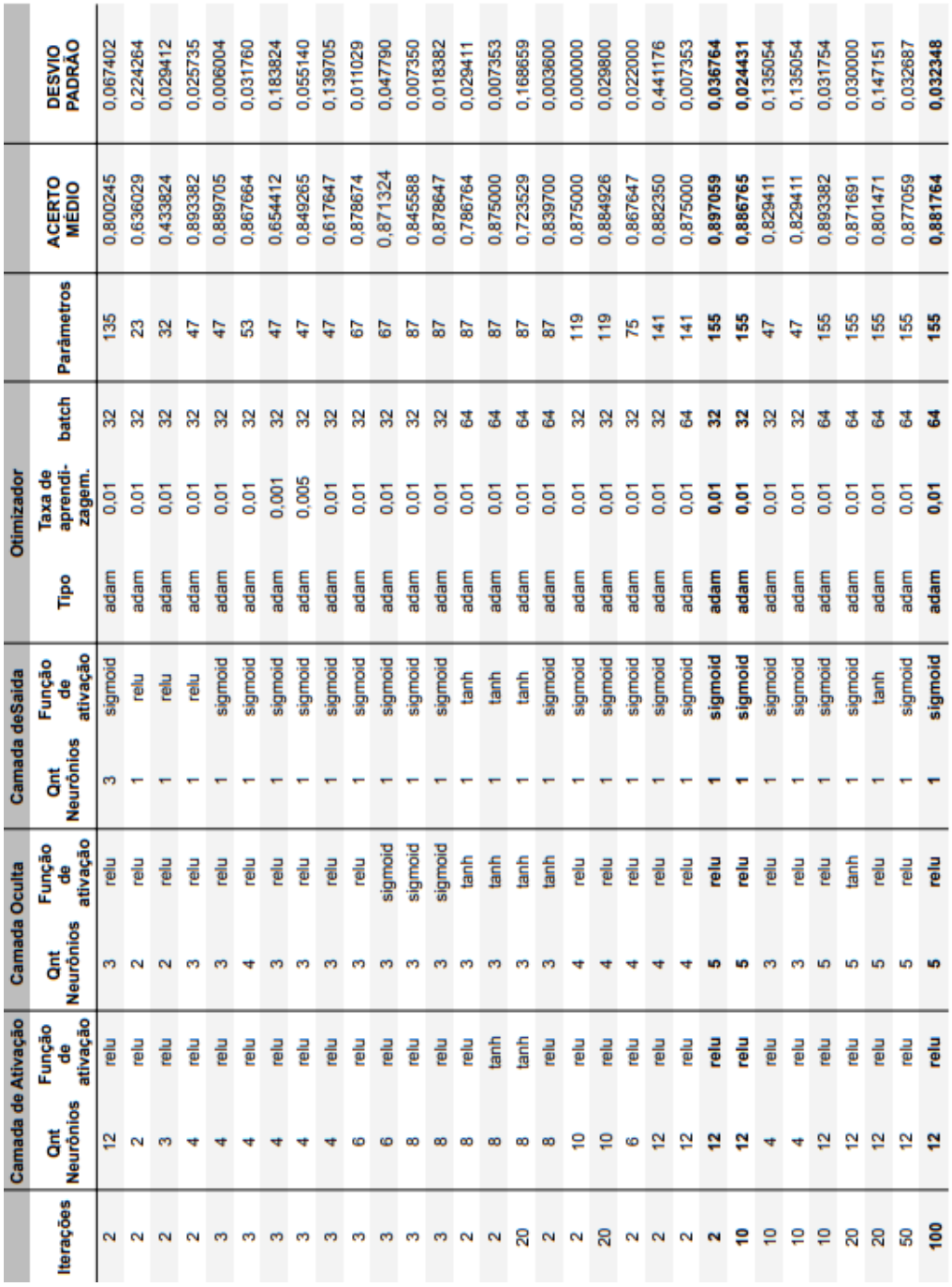
THERRIEN, C. W. *Decision estimation and classification: an introduction to pattern recognition and related topics*. [S.l.]: John Wiley & Sons, Inc., 1989. 18

VIB MASTER. *Causas raiz de falha de máquinas*. 2021. Disponível em: <https://www.vibmaster.com.br/causas-raiz-de-falha-de-maquinas/>. Acesso em: 2 mai 2022. 1

ZONTA, T. et al. Predictive maintenance in the industry 4.0: A systematic literature review. *Computers & Industrial Engineering*, Elsevier, v. 150, p. 106889, 2020. 2

ANEXO A – Configuração da Rede Neural Artificial MLP.

Figura 17 – Testes de parâmetros para ajuste da MLP.



	Camada de Ativação		Camada Oculta		Camada deSaida		Otimizador					
Iterações	Qnt Neurônios	Função de ativação	Qnt Neurônios	Função de ativação	Qnt Neurônios	Função de ativação	Tipo	Taxa de aprendi-zagem.	batch	Parâmetros	ACERTO MÉDIO	DESVIO PADRÃO
2	12	relu	3	relu	3	sigmoid	adam	0,01	32	135	0,800245	0,067402
2	2	relu	2	relu	1	relu	adam	0,01	32	23	0,636029	0,224264
2	3	relu	2	relu	1	relu	adam	0,01	32	32	0,433824	0,029412
2	4	relu	3	relu	1	relu	adam	0,01	32	47	0,893382	0,025735
3	4	relu	3	relu	1	sigmoid	adam	0,01	32	47	0,889705	0,006004
3	4	relu	4	relu	1	sigmoid	adam	0,01	32	53	0,867664	0,031760
3	4	relu	3	relu	1	sigmoid	adam	0,001	32	47	0,654412	0,183824
3	4	relu	3	relu	1	sigmoid	adam	0,005	32	47	0,849265	0,055140
3	4	relu	3	relu	1	sigmoid	adam	0,01	32	47	0,617647	0,139705
3	6	relu	3	relu	1	sigmoid	adam	0,01	32	67	0,878674	0,011029
3	6	relu	3	sigmoid	1	sigmoid	adam	0,01	32	67	0,871324	0,047790
3	8	relu	3	sigmoid	1	sigmoid	adam	0,01	32	87	0,845588	0,007350
3	8	relu	3	sigmoid	1	sigmoid	adam	0,01	32	87	0,878647	0,018382
2	8	relu	3	tanh	1	tanh	adam	0,01	64	87	0,786764	0,029411
2	8	tanh	3	tanh	1	tanh	adam	0,01	64	87	0,875000	0,007353
20	8	tanh	3	tanh	1	tanh	adam	0,01	64	87	0,723529	0,168659
2	8	relu	3	tanh	1	sigmoid	adam	0,01	64	87	0,839700	0,003600
2	10	relu	4	relu	1	sigmoid	adam	0,01	32	119	0,875000	0,000000
20	10	relu	4	relu	1	sigmoid	adam	0,01	32	119	0,884926	0,029800
2	6	relu	4	relu	1	sigmoid	adam	0,01	32	75	0,867647	0,022000
2	12	relu	4	relu	1	sigmoid	adam	0,01	32	141	0,882350	0,441176
2	12	relu	4	relu	1	sigmoid	adam	0,01	64	141	0,875000	0,007353
2	**12**	**relu**	**5**	**relu**	**1**	**sigmoid**	**adam**	**0,01**	**32**	**155**	**0,897059**	**0,036764**
10	**12**	**relu**	**5**	**relu**	**1**	**sigmoid**	**adam**	**0,01**	**32**	**155**	**0,886765**	**0,024431**
10	4	relu	3	relu	1	sigmoid	adam	0,01	32	47	0,829411	0,135054
10	4	relu	3	relu	1	sigmoid	adam	0,01	32	47	0,829411	0,135054
10	12	relu	5	relu	1	sigmoid	adam	0,01	64	155	0,893382	0,031754
20	12	relu	5	tanh	1	sigmoid	adam	0,01	64	155	0,871691	0,030000
20	12	relu	5	relu	1	tanh	adam	0,01	64	155	0,801471	0,147151
50	12	relu	5	relu	1	sigmoid	adam	0,01	64	155	0,877059	0,032687
100	**12**	**relu**	**5**	**relu**	**1**	**sigmoid**	**adam**	**0,01**	**64**	**155**	**0,881764**	**0,032348**

Fonte: Da autora.