UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

VALESCKA RAYSA MOURA SOARES DE BARROS

APLICAÇÃO DE TÉCNICAS DE *MACHINE LEARNING* NA DETECÇÃO DE PERDAS NÃO TÉCNICAS

Recife
2022

VALESCKA RAYSA MOURA SOARES DE BARROS

APLICAÇÃO DE TÉCNICAS DE *MACHINE LEARNING* NA DETECÇÃO DE PERDAS NÃO TÉCNICAS

Projeto de Dissertação apresentado ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica.

Área de concentração: Processamento de Energia.

Orientador: Prof. Dr. Ronaldo Ribeiro Barbosa de Aquino.

Recife

2022

Catalogação na fonte
Bibliotecário Gabriel Luz CRB-4 / 2222

B277a Barros, Valescka Raysa Moura Soares de.
Aplicaçâo de tecnica de *machine learning* na detecção de perdas não técnicas / Valescka Raysa Moura Soares de Barros. 2022.
76 f: figs., tabs., abrev. e siglas.

Orientador: Prof. Dr. Ronaldo Ribeiro Barbosa de Aquino.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia Elétrica, Recife, 2022.
Inclui referências.

1. Engenharia elétrica. 2. Machine learning. 3. Perdas não técnicas. 4. Perdas comerciais. 5. K-means. 6. Multilayer perceptron. 7. agrupamento hierárquico. I. Aquino, Ronaldo Ribeiro Barbosa de (Orientador). II. Título.

UFPE

621.3 CDD (22. ed.) BCTG / 2022 - 420

VALESCKA RAYSA MOURA SOARES DE BARROS

APLICAÇÃO DE TÉCNICAS DE MACHINE LEARNING NA DETECÇÃO DE PERDAS NÃO TÉCNICAS

Projeto de Dissertação apresentado ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do grau de Mestre em Engenharia Elétrica. Área de concentração: Processamento de Energia.

Aprovado em: 29 de agosto de 2022

BANCA EXAMINADORA

_____

Prof. Dr. Ronaldo Ribeiro Barbosa de Aquino (Orientador)
Universidade Federal de Pernambuco

_____

Prof. Dr. Jonata Campelo de Albuquerque (Examinador Externo)
Universidade Federal de Pernambuco

_____

Profa. Dra. Aida Araujo Ferreira (Examinadora Externa)
Instituto Federal de Pernambuco

AGRADECIMENTOS

À minha avó Edenir, a maior responsável por eu ter conseguido trilhar este caminho, ao amor, ao cuidado, ao exemplo, às renuncias e ao suporte que sempre me ofereceu.

A minha mãe Suzana e meu irmão Willams pelo incentivo e apoio durante essa jornada.

A Jackson, um amigo incrível e meu amado namorado, que me apoiou durante a preparação desta tarefa. Obrigado por seus cafés, pela ajuda na limpeza de casa, por ouvir meus lamentos, por compartilhar os inúmeros momentos de ansiedade e estresse. Sem você ao meu lado o trabalho não seria concluído.

Ao professor Ronaldo, que me adotou como orientanda e não me deixou desistir de alcançar o titulo de mestre, me apoiou e me orientou com carinho e dedicação. Agradeço também ao meu amigo Murilo, que junto com o professor Ronaldo, colaborou no desenvolvimento deste trabalho.

Ao CNPq (Conselho Nacional de Desenvolvimento Científico e Tecnológico) e a CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), pela bolsa de estudos e auxílio financeiro que possibilitou a dedicação integral ao programa de pós-graduação e a operacionalização do estudo.

RESUMO

O alto índice de perdas não técnicas (PNT), também conhecidas como perdas comerciais (PC), afeta diretamente o faturamento das distribuidoras de energia elétrica no Brasil, onde, apenas no ano de 2020 chegaram a subtrair cerca de 7,5% do montante de energia adquirido pelas concessionárias. O prejuízo é repassado para os consumidores até o limite regulatório, a diferença entre o limite regulatório e o total de PNT é custeado pelas distribuidoras. As ações de combate às PNT nem sempre são assertivas, causando ainda mais prejuízos para as empresas. Afim de direcionar as inspeções vêm se utilizando algoritmos de *Machine Learning* (ML) para detectar clientes que possivelmente estão desviando energia ou possuem alguma falha em seu sistema de medição e faturamento. Este trabalho simula esta situação usando uma base de dados real e realizando testes com redes *Multilayer Perceptron* (MLP), *K-means* e Agrupamento hierárquico, aplicando padronização e normalização para o tratamento de dados. Foi utilizado o *Jupyter notebbok,* um ambiente de desenvolvimento integrado *(*IDE do inglês *Integrated Development Environment)*, de código aberto disponibilizado no *software* Anaconda. Foram testados três modelos, cada um utilizando a base de dados original, usando a base de nados normalizada e também com a base de dados padronizada. O modelo que obteve os melhores resultados referentes ao conjunto de simulação foi o que utilizou arquitetura de Redes Neurais artificiais (RNA) do tipo MLP, com uma acurácia de 69%, no entanto são necessários a implementação de outras técnicas de tratamento de dados para obter um melhor tempo de processamento.

Palavras-chave: *machine learning*; perdas não técnicas; perdas comerciais; *K-means*; *multilayer perceptron*; agrupamento hierárquico.

ABSTRACT

The high rate of non-technical losses (NTL), also known as commercial losses (CL), directly affects the billing of electricity distributors in Brazil, where, in 2020 alone, they subtracted about 7.5% of the amount of energy acquired by the concessionaires. The loss is passed on to consumers up to the regulatory limit, the difference between the regulatory limit and the total PNT is borne by the distributors. Actions to combat PNT are not always assertive, causing even more damage to companies. In order to direct the inspections, Machine Learning (ML) algorithms have been used to detect customers who are possibly diverting energy or have a failure in their metering and billing system. This work simulates this situation using a real database and performing tests with Multilayer Perceptron (MLP), K-means and Hierarchical Clustering networks, applying standardization and normalization for data processing. Jupyter notebbok, an open-source Integrated Development Environment (IDE) made available in Anaconda software, was used. Three models were tested, each one using the original database, using the normalized database and also with the standardized database. The model that obtained the best results regarding the simulation set was the one that used Artificial Neural Networks (ANN) architecture of the MLP type, with an accuracy of 69%, however, it is necessary to implement other data processing techniques to obtain a better processing time.

Keywords: machine learning; non-technical losses; commercial losses; K-means; multilayer perceptron; hierarchical grouping.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ANEEL	Agência Nacional de Energia Elétrica
BT	Baixa Tensão
CSV	*Comma Separeted Values* (Valores Separados por Vírgula)
DT	*Decision Tree* (Árvore de Decisão)
EMQ	Erro Médio Quadrático
IA	Inteligência Artificial
IDE	*Integrated Development Environment* (Ambiente de Desenvolvimento Integrado)
KNN	*K-Nearest Neighbors Algorithm* (K-Ésimo Vizinho mais Próximo)
LOF	Fator de *Outlier* Local
MIT	*Massachusetts Institute of Technology*
ML	*Machine Learning*
MLP	*Multilayer Perceptron* (Perceptron Multicamadas)
NAN	*Not a Number* (Valor Numérico Indefinido)
NMEP	Número Máximo de Épocas
PC	Perdas Comerciais
PCA	*Principal Component Analysis* (Análise De Componentes Principais)
PIB	Produto Interno Bruto
PNT	Perdas Não Técnicas
PRODIST	Procedimentos de Distribuição
PRORET	Procedimentos de Regulação Tarifária – ANEEL
PT	Perdas Técnicas
RL	*Reinforcement Learning* (Aprendizado por Reforço)
RNA	Rede Neural Artificial
SEP	Sistema Elétrico de Potência
SGCC	*State Grid Corporation of China*
SVM	*Support Vector Machine* (Máquina de Vetores de Suporte)

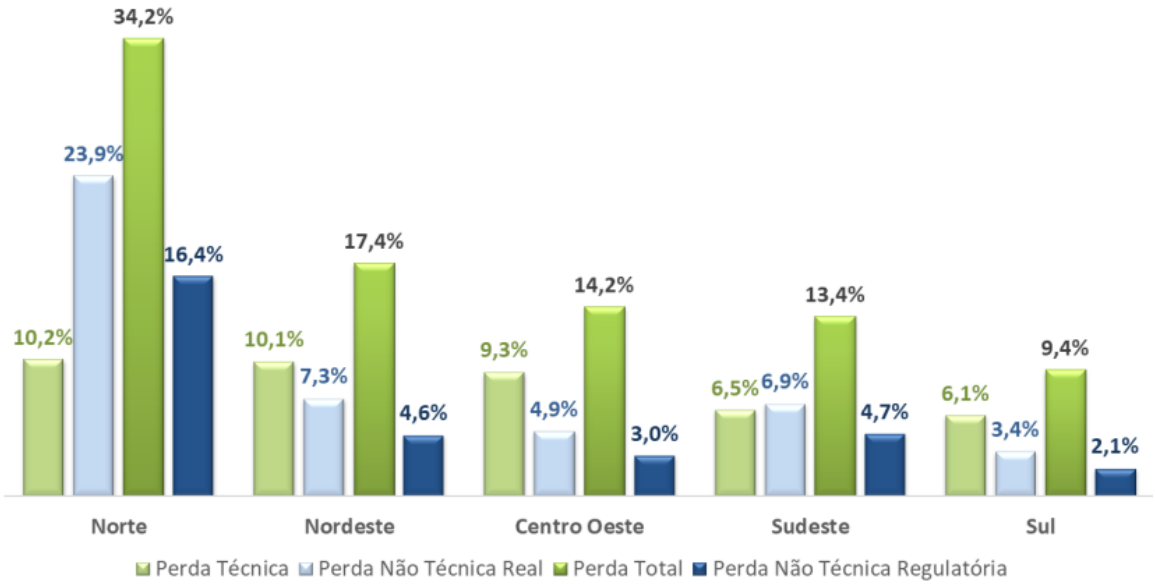
SUMÁRIO

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO E RELEVÂNCIA

O alto índice de perdas não técnicas (PNT), também conhecidas como perdas comerciais, afeta diretamente o faturamento das distribuidoras de energia elétrica no Brasil. Em 2020, as PNT chegaram a somar 37,9TWh, 7,5% do montante adquirido pelas concessionárias (ANEEL, 2021b), no entanto, os prejuízos referentes à essas perdas só são repassadas até o limite regulatório, definido na revisão tarifária periódica que ocorre a cada três ou a cada cinco anos. A Figura 1 mostra a diferença entre as perdas não técnicas totais e as perdas regulatórias, deixando evidente que apenas uma pequena parcela do montante perdido é recuperada pelas concessionárias.

Figura 1 - Perdas técnicas, não técnicas reais e regulatórias sobre a energia injetada por região (2020)



Fonte: ANEEL (2021b, p. 3).

O alto índice de perdas comerciais gera um círculo vicioso, pois desencadeia um consumo desenfreado acarretando em construção de novas usinas ou ao uso de fontes energéticas mais custosas, encarecendo a fatura de energia e fomentando um crescimento das PNT (MATOS, 2017).

Atualmente o processo de detecção de PNT é bem intuitivo, onde as concessionárias enviam equipes para realizar inspeções em unidades consumidoras baseando-se em indicadores como nos casos em que, havendo o interrompimento do fornecimento de energia a um cliente e o mesmo não solicita o religamento, indicando que o cliente pode estar fraudando seu consumo de energia, ou quando este reside em bairros com elevados índices de perdas. No entanto, não é possível realizar inspeções em todas as unidades consumidoras.

A realização de inspeções em unidades consumidoras com consumo regular gera prejuízo para a distribuidora, pois naquela ação é investido tempo de funcionários, combustível, equipamentos, entre outros insumos. O índice de sucesso dessas inspeções fica em torno de 10% (REIS FILHO, 2006).

Em contrapartida, estão surgindo muitas tecnologias no combate as PNT, essas tecnologias baseiam-se em ferramentas de análise que permitem um melhor direcionamento das equipes de inspeção. Sistemas de medição inteligentes, *smart grids*, e novos equipamentos de monitoramento estão sendo implementados com sucesso nos sistemas de distribuição de energia. Nesse conjunto, muitas técnicas de inteligência artificial têm sido desenvolvidas para atuar no combate às PNT, com o objetivo de encontrar padrões não detectáveis por humanos ou por sistemas computacionais mais simples em bancos de dados de grandes proporções, como é o caso do histórico de clientes de uma concessionária de energia.

Contudo, pouco se tem publicado sobre o uso de algoritmos de *Machine Learning* (ML) não supervisionados, tampouco comparando-os com algoritmos de aprendizagem supervisionada. Visando isto, o presente trabalho propõe a implementação de dois modelos de ML de aprendizagem não supervisionada, o *K-means* e o Agrupamento Hierárquico, e também uma implementação de um modelo de rede neural MLP de aprendizado supervisionado. As técnicas abordadas neste trabalho buscam a melhoria dos modelos, trazendo para as concessionárias mais um artefato no trabalho de combate as PNT.

1.2 OBJETIVOS

O objetivo geral deste trabalho é apresentar uma metodologia baseada em ML para auxiliar as concessionárias de energia no direcionamento de ações de combate às PNT, propondo um modelo de ML capaz de detectar com eficiência consumidores

fraudulentos ou com problemas no sistema de mediação e faturamento. Os objetivos específicos deste trabalho consistem em:

- Apresentar um resumo sobre PNT no Brasil;
- Identificar os principais métodos de fraude realizados;
- Apresentar novas tecnologias de combate às perdas;
- Realizar um estudo sobre ML e técnicas para criação de modelos;
- Discorrer sobre o funcionamento dos algoritmos *K-means*, Agrupamento hierárquico e algoritmos utilizados em redes *Multilayer Perceptron* (MLP), como o *Backpropagation*;
- Realizar o pré-processamento da base de dados obtida visando melhores resultados;
- Implementar computacionalmente os modelos escolhidos na ferramenta Anaconda, utilizando *Phyton*;
- Entender e confrontar os resultados obtidos, propondo melhorias em trabalhos futuros.

1.3 ESTRUTURA DA DISSERTAÇÃO

Esta dissertação está estruturada da seguinte forma:

No capítulo 1 é feita uma apresentação do tema, mostrando a relevância do assunto e citando os objetivos do trabalho, também é mostrado a estrutura de elaboração da dissertação.

No capítulo 2 é apresentado uma síntese sobre as perdas técnicas e não técnicas, diferenciando ambas e mostrando o impacto das PNT para as concessionárias de energia e para os consumidores, também mostra os principais tipos de fraude executadas por clientes desonestos, assim como o impacto socioeconômico trazido por esta situação. Ao final do capítulo são trazidas algumas das novas tecnologias usadas no combate às PNT.

No capítulo 3, é feita uma síntese sobre ML, expondo de forma teórica os processos de aprendizagem e abordando o funcionamento dos algoritmos escolhidos neste trabalho. No capítulo 4 é apresentada a metodologia utilizada para o desenvolvimento dos modelos. São denotadas todas as funcionalidades utilizadas dentro do *software* Anaconda.

O capítulo 5 traz os resultados obtidos na implementação dos três algoritmos nas três situações sugeridas. Por fim, apresenta-se as conclusões acerca dos resultados obtidos na implementação, além de propor melhorias para trabalhos futuros.

2 PERDAS NA DISTRIBUIÇÃO DE ENERGIA ELÉTRICA

2.1 INTRODUÇÃO

As perdas na distribuição se referem à parcela de energia que foi adquirida pela distribuidora, porém não foi comercializada, seja por motivos técnicos ou por motivos comerciais. A redução dessas perdas é um dos grandes desafios para as concessionárias, já que geram não só prejuízo financeiro como também condições de risco para os consumidores.

As perdas globais de energia, em um sistema de distribuição, podem ser encontradas subtraindo a energia despachada pela subestação, saída do alimentador, da energia mensal medida dos consumidores atendidos pelo alimentador em questão (BERNARDON *et al*., 2007), conforme Equação 2.1:

$$PG(i) = EG(i) - EF_i \tag{2.1}$$

Em que:

i = Ramal em questão

PG = Perdas globais em um ramal i

EG = Energia fornecida pela subestação para um determinado ramal i num período determinado em kWh.

$\sum EF_i$ = Energia total faturada pela concessionária de todos os consumidores conectados ao ramal i em questão.

As perdas globais podem ser classificadas em dois tipos: perdas técnicas e não técnicas. As perdas técnicas referem-se às condições da rede elétrica que afetam o fornecimento de energia como por exemplo efeitos da corrente elétrica sobre o condutor, potência reativa, entre outros. Já as perdas não técnicas referem-se à energia fornecida pela distribuidora que, todavia, não foi faturada devido a furtos, fraudes nos Sistemas distribuição de energia ou até mesmo falhas no sistema de medição.

O cálculo das perdas técnicas é determinado no PRODIST (Procedimentos de Distribuição de Energia Elétrica), um conjunto de documentos que normatiza as atividades técnicas dos sistemas de distribuição de energia elétrica. têm como objetivo regulamentar o relacionamento das distribuidoras com os consumidores de energia

elétrica, além de padronizar as operações do sistema de distribuição. É dividido em 11 módulos sendo estes:

- Módulo 1 - Glossário de Termos Técnicos do PRODIST
- Módulo 2 - Planejamento da Expansão do Sistema de Distribuição
- Módulo 3 - Conexão ao Sistema de Distribuição de Energia Elétrica
- Módulo 4 - Procedimentos Operativos do Sistema de Distribuição
- Módulo 5 - Sistemas de Medição e Procedimentos de Leitura
- Módulo 6 - Informações Requeridas e Obrigações
- Módulo 7 - Cálculo de Perdas na Distribuição
- Módulo 8 - Qualidade do Fornecimento de Energia Elétrica
- Módulo 9 - Ressarcimento de Danos Elétricos
- Módulo 10 - Sistema de Informação Geográfica Regulatório
- Módulo 11 - Fatura de Energia Elétrica e Informações Suplementares

De acordo com Piotrowski *et al*. (2021, p. 1), “As perdas de energia impactam no planejamento e operação da rede, principalmente dos investimentos que precisam ser realizados para suprir uma demanda por energia que não é devidamente utilizada ou faturada”.

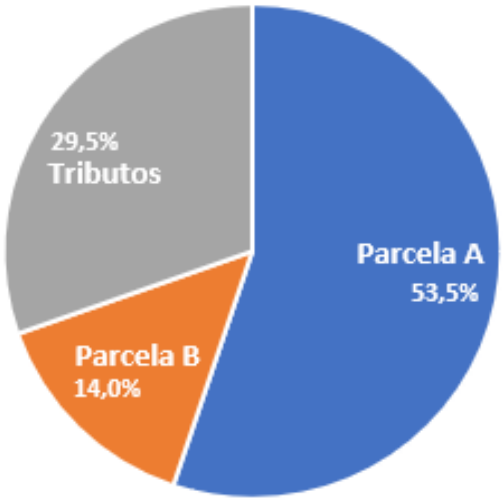
Em 2020 as perdas globais na distribuição de energia chegaram a aproximadamente 14,8% (ANEEL, 2021b). O conhecimento do nível de perdas bem como o tipo é de extrema importância para as concessionárias, já que esses valores influenciam diretamente nas revisões tarifárias periódicas na qual a Agência Nacional de Energia Elétrica (ANEEL) determina, a partir de regras definidas em procedimentos, qual será o valor do reajuste da tarifa de energia praticado para cada concessionária.

No Brasil, o setor de distribuição de energia é considerado um monopólio natural, ou seja, não existe concorrência entre as concessionárias tornando necessária a presença de um órgão regulador. A ANEEL é a responsável por fiscalizar e regular a comercialização de energia elétrica, bem como realizar as revisões tarifárias periódicas.

A tarifa de energia é composta basicamente pela parcela A, parcela B e impostos, conforme Figura 2. A parcela A representa cerca de 53% da tarifa de energia e é formada pelos gastos relativos à compra de energia, transmissão e encargos

setoriais, esses custos não estão diretamente relacionados com a distribuidora e são chamados de custos não gerenciáveis. A parcela B é considerada como custos gerenciáveis pela distribuidora e é composta por gastos referentes à distribuição de energia como despesas de operação e manutenção, a cota de depreciação e a remuneração dos investimentos.

Figura 2 - Composição da tarifa de energia



Fonte: Elaborado pela autora (2022).

O processo de revisão tarifária ocorre, em média, a cada quatro anos e revisa a parcela B, ou seja, tem como objetivo avaliar junto às distribuidoras os custos operacionais e a remuneração de investimentos, definindo e fixando parâmetros essenciais para a lucratividade da empresa e, dessa forma, garantindo o fornecimento de energia elétrica de qualidade. A parcela A é reajustada anualmente na data de aniversário de concessão de cada distribuidora.

As perdas técnicas são características do processo, por isso são consideradas nas revisões tarifárias periódicas a partir de regras definidas no módulo 7 do PRODIST no qual são determinadas as condições para o cálculo de perdas técnicas. Tais perdas ocorrem com mais frequência nos alimentadores primários, nos quais se tem um maior fluxo de potência e são calculadas a partir do somatório das perdas em equipamentos, condutores e medidores.

A determinação é feita através de cálculos de fluxo de potência nas diversas etapas do Sistema Elétrico de Potência (SEP), as etapas mais importantes para o

setor de distribuição de energia são: Subtransmissão (69kV à 138kV), subestações abaixadoras (transformação de alta tensão para média tensão), rede primária de distribuição, rede secundária de distribuição, ramais de ligação e medição. Além das etapas citadas também são consideradas as perdas nas conexões como efeito corona, fuga nos isolamentos, perdas em bancos de capacitores e reguladores de tensão, etc. Essas perdas são tratadas como perdas gerais e, por serem difíceis de serem calculadas, são estimadas entre 5% a 10% do valor total das perdas técnicas (PENIN, 2008).

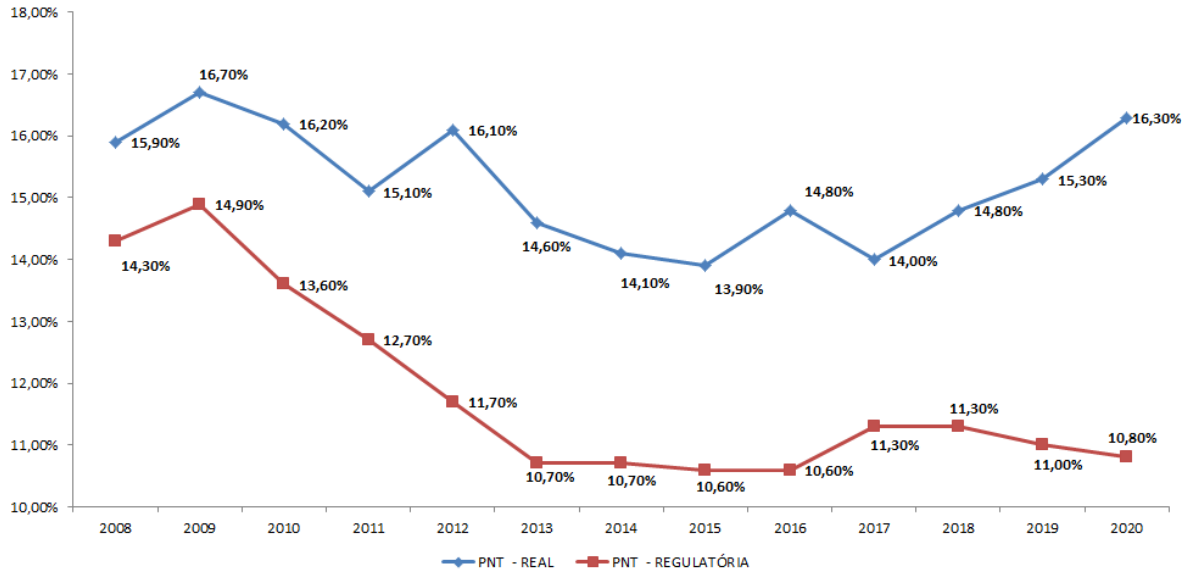
As perdas não técnicas são obtidas através da diferença entre as perdas globais e perdas técnicas regulatórias, conforme Equação 2.2, contudo, a ANEEL determina valores regulatórios para os níveis de perdas não técnicas que podem ser diferentes do real obtido pela distribuidora. A diferença de custos entre o valor regulatório e o real é de responsabilidade da concessionária. Os limites regulatórios de perdas não técnicas são calculados conforme as regras definidas no Submódulo 2.6 do PRORET (Procedimentos de Regulação Tarifária), conjunto de documentos que regulamentam e padronizam o processo de cálculos tarifários.

$$PNT(i) = PG(i) - PT(i) \tag{2.2}$$

Sendo i o ramal em questão analisado, $PG(i)$ as perdas globais em um ramal i em kWh, $PNT(i)$ são as perdas não técnicas de energia elétrica em um ramal i para um período mensal (kWh) e $PT(i)$ as perdas técnicas de energia elétrica em um ramal i para um período mensal (kWh).

Em 2020 as perdas não técnicas representaram um custo de R$8,6 bilhões, equivalente à 37,9 TWh de energia, sendo que, desse montante, R$5,6 bilhões são consideradas perdas não técnicas regulatórias, que é repassado para o consumidor na revisão tarifária periódica, representando um aumento de 2,9% na conta de energia. A diferença entre o custo real e o custo considerado regulatório fica sob responsabilidade das concessionárias (ANEEL, 2021b). A Figura 3 mostra que a diferença entre as PNT regulatórias e reais são historicamente discrepantes, gerando para as distribuidoras um prejuízo contínuo que é indispensável ser mitigado.

Figura 3 - Comparativo entre PNT real e PNT regulatórias ao longo dos anos



Fonte: ANEEL (2021b, p. 10).

2.2 PERDAS NÃO TÉCNICAS

As perdas não técnicas, também conhecidas como perdas comerciais, correspondem à adulteração nos medidores de energia, conexão ilegal à rede, erros de medição e falhas no processo de faturamento (CHAUHAN; RAJVANSHI, 2013). A partir desses tipos de PNT pode-se classificá-las em três tipos:

- PNT devido a falhas no faturamento ou sistema de medição;
- Furto;
- Fraude do sistema de medição.

As PNT devido a falhas no faturamento ou sistema de medição referem-se às irregularidades em medidores e no sistema de faturamento da concessionária. Essa situação acontece, por exemplo, quando faturistas registram o consumo de forma errada (em caso de medidores analógicos e digitais que não possuem sistema de faturamento remoto), ou quando o medidor apresenta algum defeito de fabricação e até mesmo por ligação errada feita pelo eletricista instalador. Nessa situação o consumidor não tem o controle sobre o valor faturado, muitas vezes sendo até prejudicado por cobranças indevidas e falha nos equipamentos.

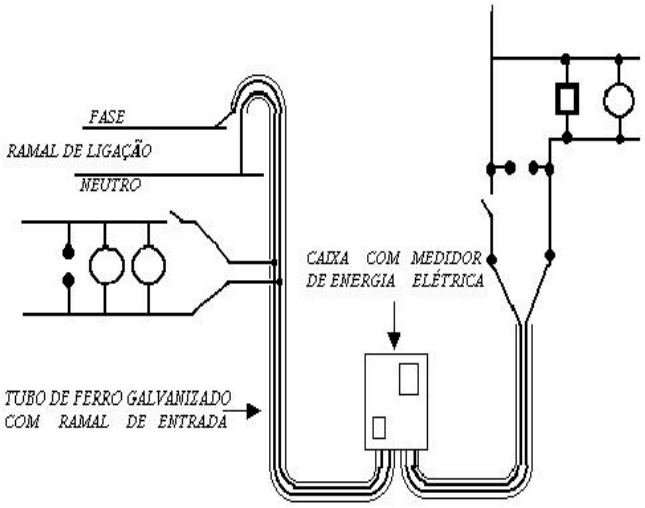
Pode-se considerar como furto quando o cliente realiza uma conexão diretamente à rede de baixa tensão (BT) da concessionária. Segundo NAGAMINE (2011, p. 31):

> As ligações clandestinas são feitas diretamente na rede da distribuidora de energia sem equipamento intermediário que oferece segurança aos consumidores. Elas provocam perigos à sociedade, pois provocam choques elétricos, curtos-circuitos que podem resultar em ferimentos e até a morte por descargas elétricas e até incêndios nas residências com ligação sem segurança e que utilizam fios quebrados ou sem um isolamento adequado, além de sobrecargas no sistema elétrico que compromete equipamentos da rede de distribuição como transformadores e pode ocasionar interrupções no abastecimento de outros consumidores.

As fraudes são caracterizadas por uma adulteração na medição ou no ramal de medição, logo o consumidor pode aumentar o consumo energético sem que isto seja contabilizado. Fraudes no sistema de medição de energia, apesar de representarem um perigo altíssimo para o executor, é uma das práticas comumente usadas para fazer desvio de energia elétrica. Algumas das técnicas mais comuns utilizadas são:

- Desvio do fluxo de potência antes da medição: consiste em derivar os condutores do ramal de entrada antes da conexão de entrada do medidor, dessa forma o fraudador pode alimentar parte de suas cargas sem realizar a medição. Essa ligação pode ser feita na caixa de medição ou perfurando o eletroduto do ramal de entrada, como mostrado na Figura 4;

Figura 4 - Desvio de energia com perfuração do eletroduto do ramal de entrada

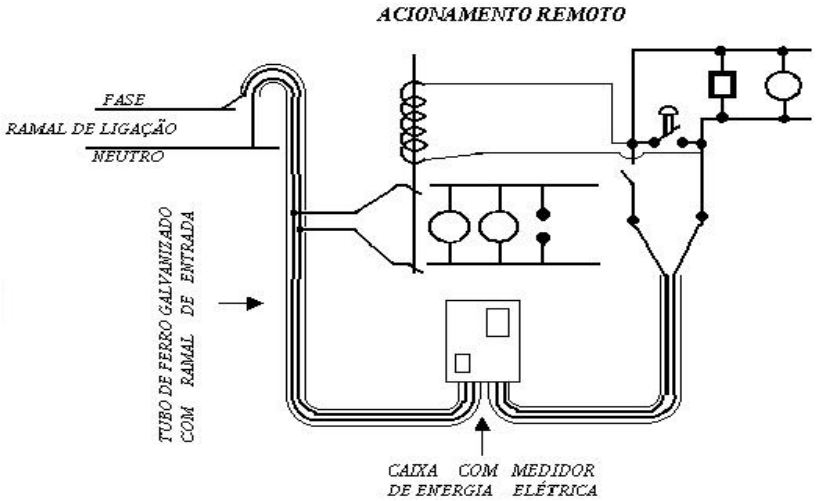


Fonte: Oliveira (2012).

- Desvio do fluxo de potência antes da medição com acionamento remoto: o método é similar ao citado anteriormente, com a diferença de inserir algum dispositivo automático que permita interromper o fluxo de corrente pelo caminho

desviado e dessa maneira dificultar a detecção da fraude. A Figura 5 exemplifica um método de instalação desse sistema;

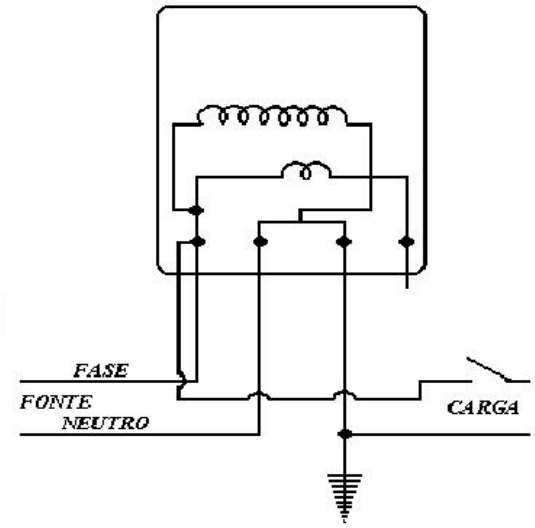
Figura 5 **-** Desvio do fluxo de potência antes da medição com acionamento remoto.



Fonte: Oliveira (2012).

- Medição monofásica com ligação direta de uma fase no terminal do medidor: este método consiste em derivar a alimentação para a carga diretamente do terminal de entrada, dessa forma interrompendo a medição, como mostrado na Figura 6.

Figura 6 - Medição monofásica com ligação direta de uma fase no terminal do medidor

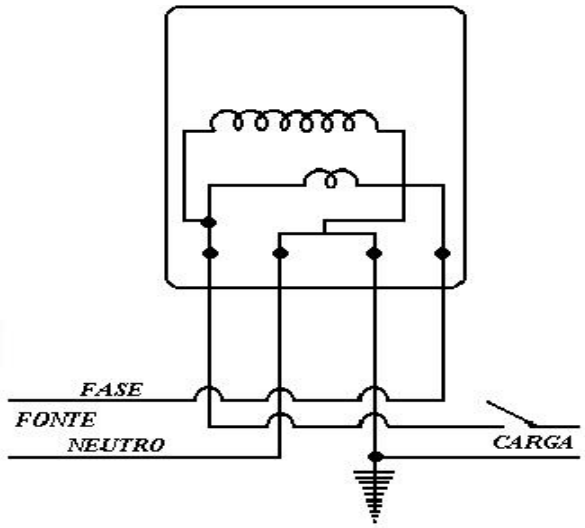


Fonte: Oliveira (2012).

- Medição Monofásica com Inversão da Fase de Entrada com a Saída: Nesse caso, a alimentação da carga se dá através do borne de saída, como os

medidores analógicos não detectam um fluxo de corrente invertido (Figura 7), logo, não vão medir a energia consumida.

Figura 7 - Medição monofásica com inversão da fase de entrada com a saída



Fonte: Oliveira (2012).

Existem diversos outros métodos de fraude de medição como o uso de imãs para frear o elemento móvel de medidores eletromecânicos, rompimento da bobina de potencial, adulteração do registrador, criação de neutro artificial, dentre outras técnicas.

2.3 IMPACTO SÓCIO ECONÔMICO

As fraudes no sistema elétrico de distribuição afetam não só as concessionárias, mas também seus clientes, pois, além de influenciarem na tarifa de energia dos consumidores pagantes, como mencionado anteriormente, também causam sobrecarga na rede elétrica, prejudicam a qualidade do fornecimento de energia provocando variações de tensões que muitas vezes chegam à danificar os aparelhos dos clientes. Além disso, as concessionárias têm dificuldade em prever a energia a ser contratada da geradora fazendo com que, em um momento de alta demanda, ocorra interrupção no fornecimento de energia e até mesmo apagões. As PNT sobrecarregam as usinas geradoras tornando a tarefa da confiabilidade do sistema de fornecimento de energia muito desafiadora.

Essa situação não é preocupante apenas para o Brasil. Na cidade do México milhões de clientes ilegais levaram uma rede sobrecarregada ao limite e milhares de residências e empresas foram atingidas por cortes de energia. Estima-se que os roubos de energia resultem na perda de até US $475 milhões em receitas anualmente (SULLIVAN, 2002). Na Índia, o roubo de eletricidade faz as perdas anuais estimadas atingirem US $4,5 bilhões, cerca de 1,5% do PIB (BHATIA; GULATI, 2004). As perdas não técnicas envolvem questões sociais, econômicas, regionais, administrativas, políticas, de infraestrutura, taxa de alfabetização, criminalidade e corrupção, efeito sobre consumidores regulares, qualidade da energia e segurança.

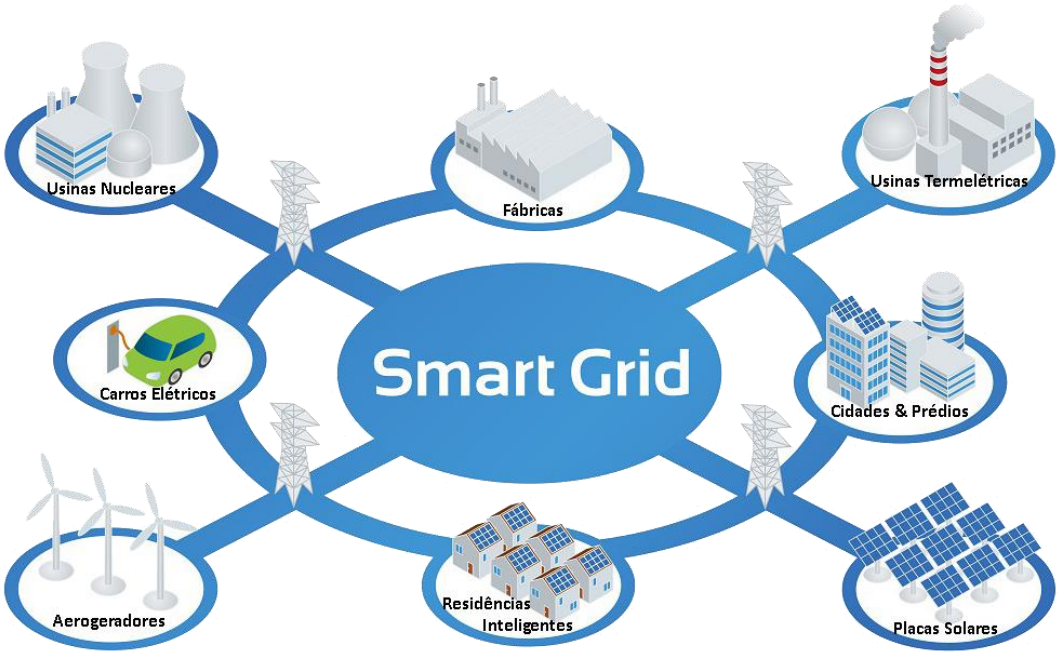
Dantas (2006) menciona como principais malefícios causados pelas fraudes de energia elétrica: a insegurança; a concorrência desleal; o desperdício de energia; e a não arrecadação de impostos. A insegurança está relacionada ao fato de que a maioria das pessoas que praticam intervenções no sistema de energia elétrica não possuem instrução técnica, colocando assim, a sua vida e a vida de outras pessoas em risco. A concorrência desleal ocorre quando empresas ao cometer fraude conseguem repassar aos seus clientes preços abaixo do praticado no mercado por empresas honestas, que por sua vez acabam ficando no prejuízo por não conseguirem competir com as empresas fraudadoras. Além disso, clientes que cometem fraude geralmente não possuem um consumo consciente de energia elétrica, acarretando maiores investimentos por parte das concessionárias para suprir esse aumento de demanda. Por fim, a não arrecadação de impostos embutidos na conta de energia traz prejuízos à sociedade pois não são destinados à seus fins.

2.4 COMBATE ÀS PNT

A prevenção das PNT remete aos métodos utilizados pelas concessionárias para evitar a ocorrência destas. As principais técnicas adotadas pelas concessionárias baseiam-se em campanhas de prevenção, modificações na infraestrutura da rede, sistemas pré-pagos, sistemas centralizados e redes inteligentes (*smart grid*) (PENIN, 2008).

O *smart grid* (redes inteligentes) permite que as concessionárias realizem cortes e desligamentos remoto, alimentação de banco de dados, faturamento mais eficiente, controle de qualidade do fornecimento de energia, e diversas outras vantagens (Figura 8).

Figura 8 - *Smart grid*



Fonte: Pelielo (2016).

As redes inteligentes têm como consequência a digitalização do setor elétrico, trazendo muitas soluções para redução das perdas técnicas e não técnicas. O uso de medidores inteligentes (*smart meters*), Sistemas de Medição Centralizada (SMC), e técnicas no campo da inteligência artificial (IA) têm se mostrado superiores aos métodos convencionais, em termos de precisão, tempo, custo e eficiência.

Os medidores inteligentes (Figura 9) possibilitam a medição do consumo em tempo real, transmitindo periodicamente para a concessionária as informações necessárias para tomada de decisões, que pode ser feita por funcionários ao analisarem os dados ou por sistemas de aprendizado de máquinas capazes de detectar anomalias não perceptíveis por pessoas.

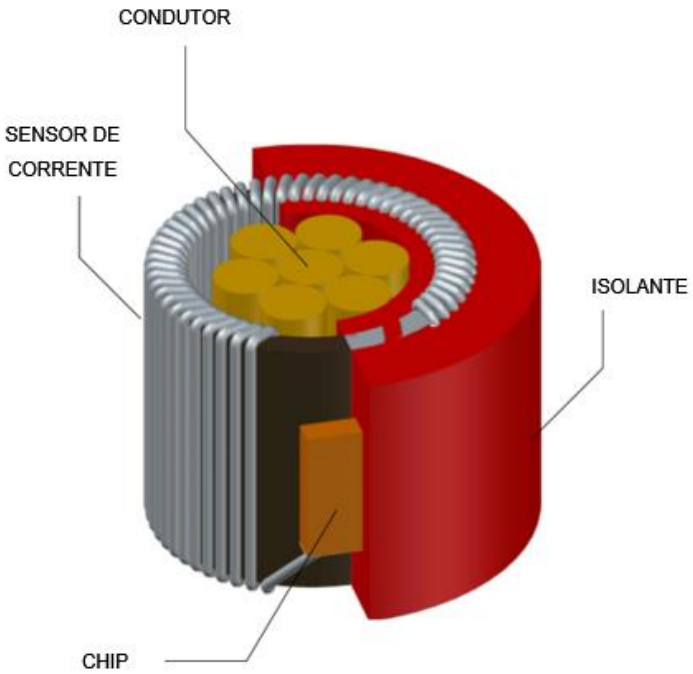
Pesquisadores da Universidade Federal da Paraíba (UFPB) desenvolveram um tipo de cabo microchipado capaz de detectar fraudes ao longo do circuito. O dispositivo contém chips dotados de sensores de corrente que se auto alimentam pela indução magnética da corrente que passa no condutor, como mostrado na Figura 10. Os chips ficam na camada isolante do condutor e fazem a medição da corrente elétrica em vários pontos ao longo da rede. Dessa forma, se em um determinado ponto foi

feita uma derivação para desviar energia, o sensor consegue detectar o local e a quantidade de energia desviada (OLIVEIRA, 2022).

Figura 9 - Medidor inteligente

Fonte: Pelielo (2016).

Figura 10 - Cabo microchipado



Fonte: Adaptado de Oliveira (2022).

2.5 TRABALHOS RELEVANTES

Vários trabalhos contribuíram como referência ou fonte de inspiração para o desenvolvimento desta dissertação. O uso de técnicas de ML para detecção de PNT é um tema já discutido no meio acadêmico, por isso, diversas técnicas já usadas foram utilizadas na presente obra e são citadas ao longo do texto.

Eichkoff *et al*. (2021) propõem um modelo de *Random Forest*, pré-classificado pelo algoritmo *K-means*, para criar grupos de consumidores que serão utilizados para a predição do modelo classificador. Ele utiliza a média e desvio padrão do consumo de três anos dos clientes como atributos no conjunto de dados (*dataset*). Matos (2017) e Reis Filho (2006) apresentam uma metodologia capaz de detectar fraudes no consumo de energia elétrica usando árvore de decisão. Em ambos os trabalhos o modelo proposto se comporta como bom classificador.

Chauhan e Rajvanshi (2013), apresenta uma análise de tecnologias, metodologias e ações a serem tomadas no combate às PNT, mostrando também as fontes dessas perdas e o impacto econômico provocado. Oliveira (2012) contribui em seu site esclarecendo as diversas formas de fraude em medidores de energia. Bhatia e Gulati (2004) e Sullivan (2002) são usados como referências acerca do impacto das PNT no mundo. Piotrowski (2021) apresenta uma análise geral das perdas elétricas das concessionárias do Rio Grande do Sul – RS, com destaque para as PNT.

Em sua tese de doutorado, Penin (2008) faz uma análise aprofundada sobre PNT, onde propõe métodos para redução de fraudes e furtos de energia, bem como processos para recuperação de receitas embasado nas normas de regulação tarifária e também aborda metodologias de cálculo para investimentos no combate às perdas de energia elétrica. Neste trabalho concluiu-se que um dos principais motivos das PNT é a desigualdade social, sugerindo ações governamentais visando generalizar o acesso e viabilizando o fornecimento de energia. Dantas (2006) propõe uma metodologia de estratificação e quantificação das PNT para categorizar áreas ou municípios, embasado em ferramentas estatísticas, utilizando dados como quantidade de consumidores, classes de consumo e posição geográfica.

Em Peng *et al*. (2021) é proposto um método para detecção PNT usando clusterização e o fator de outlier local (no inglês, LOF). Este método usa o *K-means* para agrupar os perfis de carga dos clientes, identificando aqueles cujos perfis de carga estão longe de seus centros de cluster. Estes clientes são selecionados como

candidatos atípicos e então o LOF é utilizado para calcular o nível de anomalia. Foi usado outros métodos de detecção de anomalias para comparar com o método sugerido. Segundo Khan *et al*. (2020), a identificação de fraudes é bastante desafiadora devido ao desequilíbrio da base de dados, levando à problemas de *overfiting*, causando altas taxas de falsos positivos, por isso, em seu trabalho é proposto o uso do algoritmo *Adasyn* para obter uma base de dados equilibrada, conseguindo resultados bem eficientes ao usar algoritmos de aprendizado supervisionado.

Bohani *et al*. (2021) usa os algoritmos *AdaBoost*, *Decision Tree* (DT), redes neurais artificiais e aprendizado profundo, todos com aprendizado supervisionado, para detectar PNT no mesmo banco de dados utilizado nesta dissertação. Foram realizados vários testes com diferentes proporções entre conjunto de treinamento e conjunto de teste. Para a substituição dos dados faltantes no *dataset* foi utilizado a média de consumo de cada cliente e em seguida feita a normalização.

3 APRENDIZADO DE MÁQUINAS

3.1 INTRODUÇÃO

Muito se discute atualmente sobre aprendizagem de máquinas ou, no inglês *Machine Learning* (ML), no entanto o conceito atual de ML surgiu na década de 1950 a partir de diversos trabalhos na área de inteligência artificial. Dentre eles podem ser citados alguns dos trabalhos pioneiros:

Em 1950, Alan Turing desenvolveu o “Teste de Turing” para determinar se uma máquina era realmente inteligente. Para passar no teste, uma máquina tinha que ser capaz de enganar um humano para pensar que era humano em vez de um computador (TURING, 2004).

Em 1959, Arthur Samuel, engenheiro do MIT, pioneiro da inteligência artificial e criou o primeiro programa de computador capaz de “aprender”, o *software* era um programa que jogava damas e melhorava seu desempenho, jogo após jogo (SAMUEL, 1959).

Em 1958, o psicólogo Frank Rosenblatt, da *Cornell University*, junto com sua equipe construiu uma máquina para reconhecer as letras do alfabeto com base em ideias sobre o funcionamento do sistema nervoso humano, esse trabalho tornou-se o primeiro protótipo de redes neurais artificiais (FRADKOV, 2020).

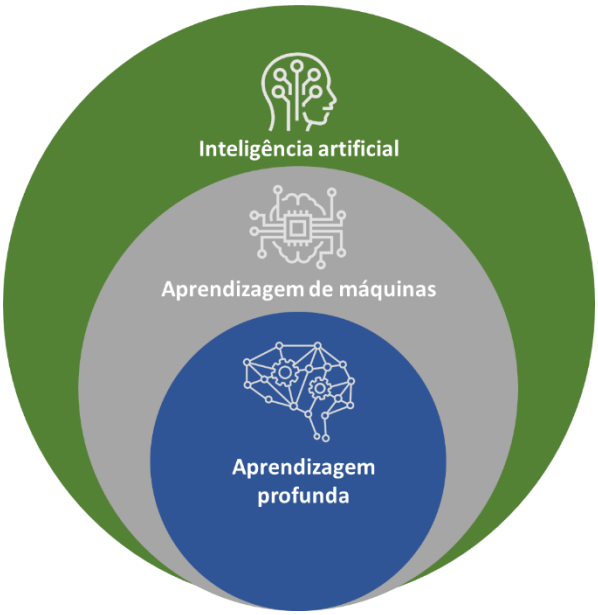
Em 1964, foi desenvolvido por Joseph Weizenbaum, no laboratório de Inteligência Artificial do MIT, o Elisa, o primeiro *chatbot* da história, que se baseando em palavras-chaves e estrutura sintática conversava de forma automática imitando uma psicanalista (BARBOSA, 2020).

O termo *Machine Learning* (ML) pode facilmente ser confundido com inteligência artificial e, apesar de ambos estarem intrinsecamente relacionados, é fundamental ressaltar que são conceitos diferentes. A Inteligência Artificial (IA) é uma área ampla dentro da Ciência da Computação que faz com que os computadores desenvolvam a capacidade de compreender e aprender, agindo de forma semelhante à inteligência humana. No entanto, IA é um campo amplo que comporta outras tecnologias, como: ML e *Deep Learning* (DL). Existem ainda outras subdivisões de IA, como: processamento de linguagem natural, robótica e visão computacional.

Na Figura 11 pode-se entender em que contexto o ML se encaixa dentro da inteligência artificial, dessa forma, segundo Monard e Baranauskas (2005), pode-se

definir ML como sendo “uma área da inteligência artificial cujo objetivo é o desenvolvimento de técnicas computacionais sobre o aprendizado bem como a construção de sistemas capazes de adquirir conhecimento de forma automática”.

Figura 11 - Diferença entre Inteligência Artificial, Aprendizagem de Máquinas e Aprendizagem Profunda



Fonte: Elaborado pela autora (2022).

3.2 TIPOS DE APRENDIZADO

Os algoritmos de ML imitam o processo de aprendizado humano. Isto é, segue o mesmo princípio de aprendizado que uma pessoa usa para entender, aprender e tomar decisões. Com o uso do ML pode-se analisar grandes volumes de dados de forma iterativa e obter resultados com maior rapidez e precisão.

Embora existam uma infinidade de algoritmos de ML pode-se dividi-los em três categorias: aprendizado supervisionado, aprendizado não supervisionado, e aprendizado por reforço. Para melhor entendimento dos tipos de aprendizado de máquinas se faz necessário apresentar alguns conceitos:

- Algoritmo: Uma sequência com números finitos de instruções que permitem chegar à solução de um determinado problema;
- Banco de dados: São conjuntos organizados de dados que se relacionam e formam uma informação, que estão relacionados entre si, por exemplo, registros sobre pessoas, lugares ou coisas;

- Cluster: Faz referência a um agrupamento de dados, aglomeração, também denominado instâncias;
- Atributo: é a característica de um exemplo, por exemplo a variável altura pode receber como atributo um número real ∈ R, também conhecido como *feature*;
- Rótulo ou classe: é a classificação que se deseja obter de um conjunto de dados.

3.2.1 Aprendizado supervisionado

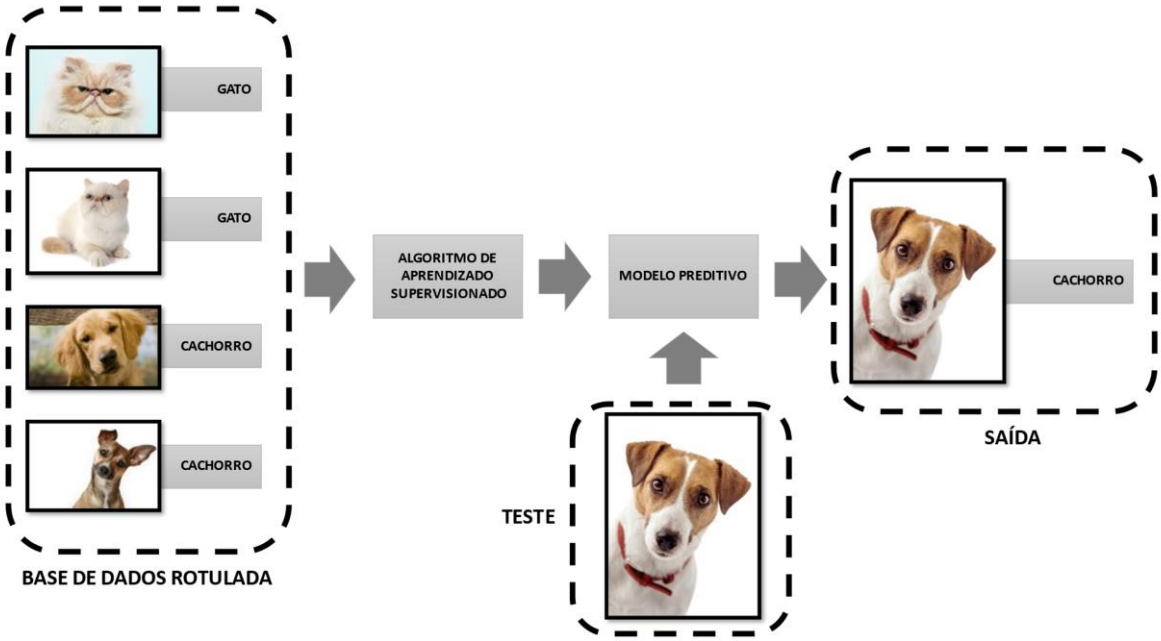
No aprendizado supervisionado, todo exemplo possui um atributo especial, o rótulo ou classe, que descreve o fenômeno de interesse, isto é, a meta que se deseja aprender e poder fazer previsões a respeito (MONARD; BARANAUSKAS, 2005).

A Figura 12 exemplifica um modelo de aprendizado supervisionado que tem uma base de dados com várias imagens de cachorros já rotuladas e imagens de gatos também já rotuladas. Esse conjunto de treinamento serve para dar um parâmetro ao algoritmo de aprendizagem, criando assim, um modelo preditivo que consegue identificar novos exemplos fornecidos à máquina, associando as imagens às respostas corretas. Entre os problemas de aprendizado supervisionado existem duas categorias, os problemas de classificação e os problemas de regressão.

Nos problemas de classificação, temos saídas discretas, ou seja, o dado é classificado de acordo com o seu tipo, como no exemplo da Figura 12 onde os animais são classificados de acordo com sua espécie. Outros exemplos de problemas de classificação são a detecção de spam na caixa de e-mail e a detecção de fraudes de cartão de crédito. A Figura 13 exemplifica como é feita a classificação.

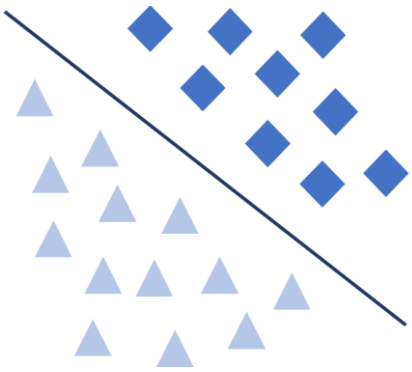
Existem também os problemas de regressão, onde a saída não é uma classe, mas sim um valor associado àquele dado. Por exemplo, o problema de mercado imobiliário, onde as entradas são dados como local, área e tempo e a saída seria a previsão do valor do imóvel. Na regressão o algoritmo identifica uma função que relaciona os dados de entrada e saída. A exatidão de um algoritmo de regressão é calculada com base na variação entre a saída precisa e a saída prevista, como mostrado na Figura 14. Segundo Ayodele (2010) os algoritmos de aprendizado supervisionado mais utilizados são: regressão logística; classificador *Naive Bayes*; *Perceptron*; máquinas de vetor de suporte (*Suport Vectors Machine* – SVM); árvore de decisão; redes neurais; e redes bayesianas.

Figura 12. Aprendizado supervisionado



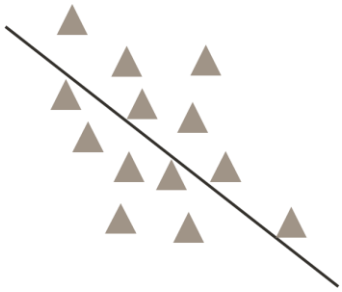
Fonte: Elaborado pela autora (2022).

Figura 13. Classificação



Fonte: Elaborado pela autora (2022).

Figura 14. Regressão

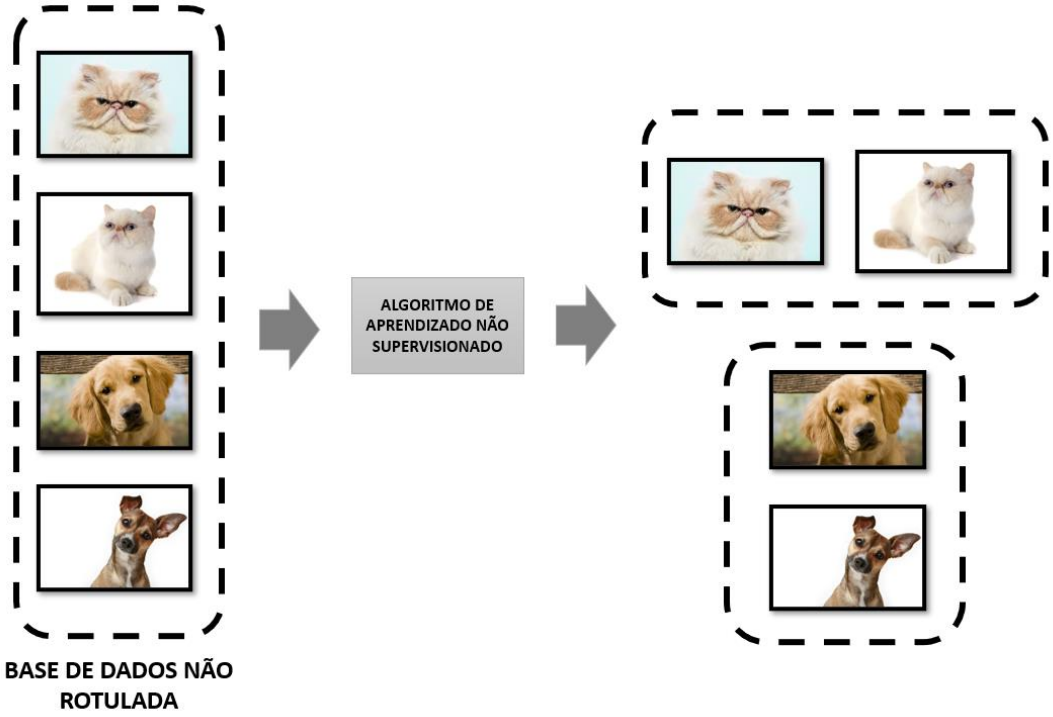


Fonte: Elaborado pela autora (2022).

3.2.2 Aprendizado não supervisionado

Nem sempre é possível ter os dados rotulados ou valores reais para aplicar o aprendizado supervisionado, pois coletar e/ou rotular um grande conjunto de exemplos pode custar muito tempo, esforço e dinheiro. No caso dos algoritmos de aprendizagem não-supervisionada, não é atribuído um rótulo para os dados de saída (Figura 15). A partir de uma grande base de dados, o algoritmo busca padrões e similaridades entre eles, permitindo identificar grupos de itens similares ou similaridade de itens novos com grupos já definidos (FONTANA, 2020).

Figura 15. Aprendizado não supervisionado



Fonte: Elaborado pela autora (2022).

Os algoritmos não supervisionados podem ser divididos em algoritmos de agrupamento e algoritmos de transformação. Os algoritmos de transformação criam uma nova versão do banco de dados original. Essa nova versão pode ajudar na compreensão de outros algoritmos ou na interpretação humana. Um exemplo de algoritmo de transformação é o *Principal Component Analysis* (PCA), cuja finalidade básica é a redução da base de dados utilizada, eliminando informações redundantes e pouco significativas na análise, exibindo apenas as informações importantes.

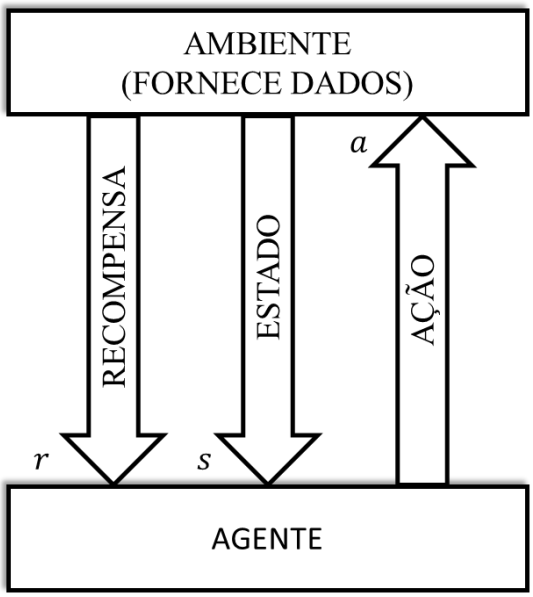
“Os algoritmos de agrupamento (*clustering*) particionam os dados em grupos com características similares com base em critérios pré-estabelecidos, permitindo encontrar padrões entre os dados fornecidos” (FONTANA, 2020, p. 6). O agrupamento pode ser por particionamento ou por agrupamento hierárquico. Os métodos de particionamento são rápidos e diretos. Alguns dos mais importantes algoritmos de clusterização são: *K-means*; clusterização hierárquica; e maximização da expectativa.

3.2.3 Aprendizado por reforço

No aprendizado por reforço (*Reforicement Learning* – RL) o algoritmo aprende através de iterações de tentativa e erro, empregando um sistema e/ou penalidades que obriga o programa a resolver o problema. O ambiente de aprendizado é dinâmico, ou seja, não são necessários novos exemplos. Neste tipo de sistema, o programa que está sendo treinado é conhecido como agente. O agente realiza uma ação aleatoriamente em um ambiente e recebe uma recompensa ou punição de acordo com o resultado dessa ação, sendo o objetivo do algoritmo maximizar a recompensa recebida (FONTANA, 2020). É importante ressaltar que, embora o programador defina um sistema de recompensas, não é dada ao agente nenhuma informação sobre como solucionar o problema.

Segundo Faria (2000), no RL, o agente e o ambiente são relacionados através da assimilação e ação. A cada iteração, o agente recebe como entrada um sinal de reforço r e uma indicação do estado atual s e por sua vez, o agente determina a ação a como resposta. Tal ação modifica o estado do ambiente e essa modificação gera um novo sinal de reforço, dessa forma o agente passa a definir ações que, com o passar das iterações, aumenta o somatório do sinal de reforço. A Figura 16 exemplifica o funcionamento deste tipo de aprendizado.

Figura 16. Aprendizado por reforço



Fonte: Elaborado pela autora (2022).

3.3 *K-MEANS*

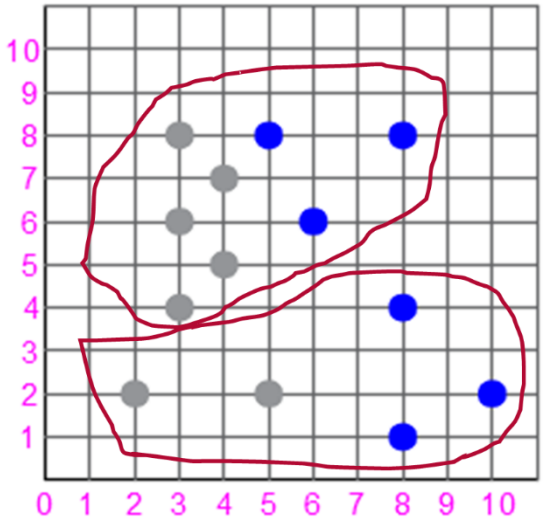
O *K-means* é um algoritmo de aprendizado não supervisionado, que usa a clusterização para agrupar dados que possuem características semelhantes, ou seja, sem nenhuma pré-classificação. O termo "*K-means*" foi empregado primeiramente por MacQueen (1967), desde então, é geralmente o método de agrupamento mais conhecido e utilizado. Existem várias extensões de *K-means* propostas na literatura. No entanto, apesar de sua popularidade, o algoritmo tem limitações relacionadas às variáveis de inicialização (AHMED; SERAJ; ISLAM, 2020). O desenvolvimento deste tópico é baseado nos trabalhos de Likas, Vlassis e Verbeek (2003), Nunes (2016) e Sinaga e Yang (2020).

O *K-means* é um algoritmo de particionamento, ou seja, ele divide um conjunto de dados em um número de clusters previamente definidos. Considere um conjunto de dados $P = \{x_1, x_2, \dots, x_n\}$ em um espaço euclidiano d-dimensional R^d, sendo n o número de pontos desse conjunto.

Inicialmente define-se o número de clusters k, então *K-means* divide aleatoriamente esse conjunto P em subconjuntos, Figura 17, $C_i = \{C_1, C_2, C_3, \dots, C_k\}$. Para cada cluster C_i existe um centróide y_i associado que será comparado com todos os elementos desse grupo. O centróide pode ser obtido através da média de todos os pontos pertencentes ao cluster, conforme Equação 3.1.

$$y_i = \frac{1}{n} \sum_{x_j \in C_i} x_j \tag{3.1}$$

Figura 17. *K-Means*: Inicialização



Fonte: Elaborado pela autora (2022).

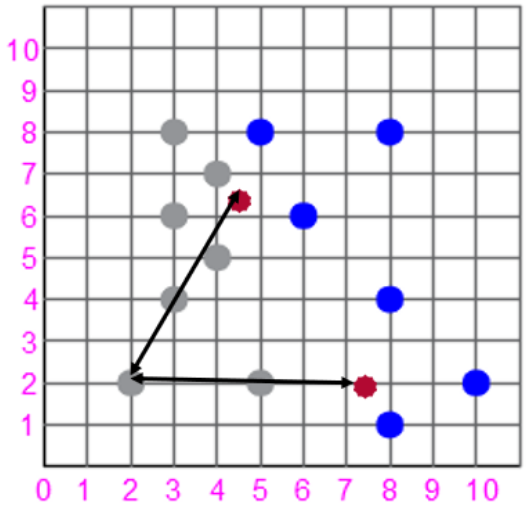
Seguidamente, o algoritmo inicia duas fases iterativas: a atribuição de clusters e a atualização dos centroides. Na primeira fase cada ponto x_j é atribuído a um cluster C_i cujo centróide y_i está mais próximo, o critério de agrupamento mais utilizado é a soma das distâncias euclidianas quadradas entre o ponto x_j e o centróide y_i, como mostra a Equação 3.2.

$$F_s(P) = \sum_{i=1}^{k} \sum_{x_j \in C_i} D(x_j, y_i) \tag{3.2}$$

Onde D é a distância euclidiana definida pela Equação 3.3, Figura 18. Dessa forma, F_s define a soma de todas as distâncias entre cada elemento de um cluster e seu centroide correspondente. O algoritmo busca minimizar F_s, convergindo para uma solução local mais próxima, isto significa que nem sempre o algoritmo encontra a solução ótima do problema.

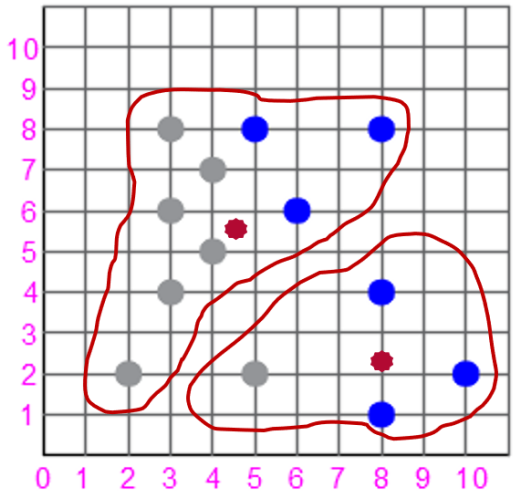
$$\sum_{i=1}^{k} |x_j - y_i|^2 \quad (3.3)$$

Figura 18. K-Means: Cálculo das distâncias



Fonte: Elaborado pela autora (2022).

Figura 19. K-Means: Escolha de novos clusters

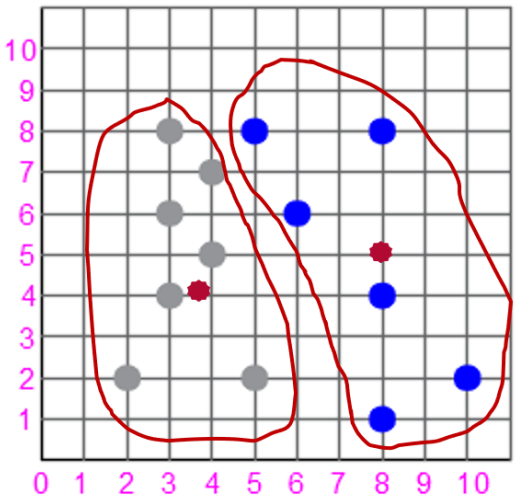


Fonte: Elaborado pela autora (2022).

Após a primeira fase alguns pontos terão mudado de grupo e dessa forma inicia-se a segunda fase atualizando o valor dos centroides através da Equação 3.1, Figura 19. Essas duas etapas se repetem, Figura 20, até todos os pontos estarem alocados em seus clusters sem que os centroides mudem novamente ou se a diferença entre os novos centroides calculados e os anteriores alcance um valor

mínimo estabelecido com critério de parada (erro). Pode-se usar a Equação 3.4 como teste de parada.

Figura 20. K-Means: Reinicialização

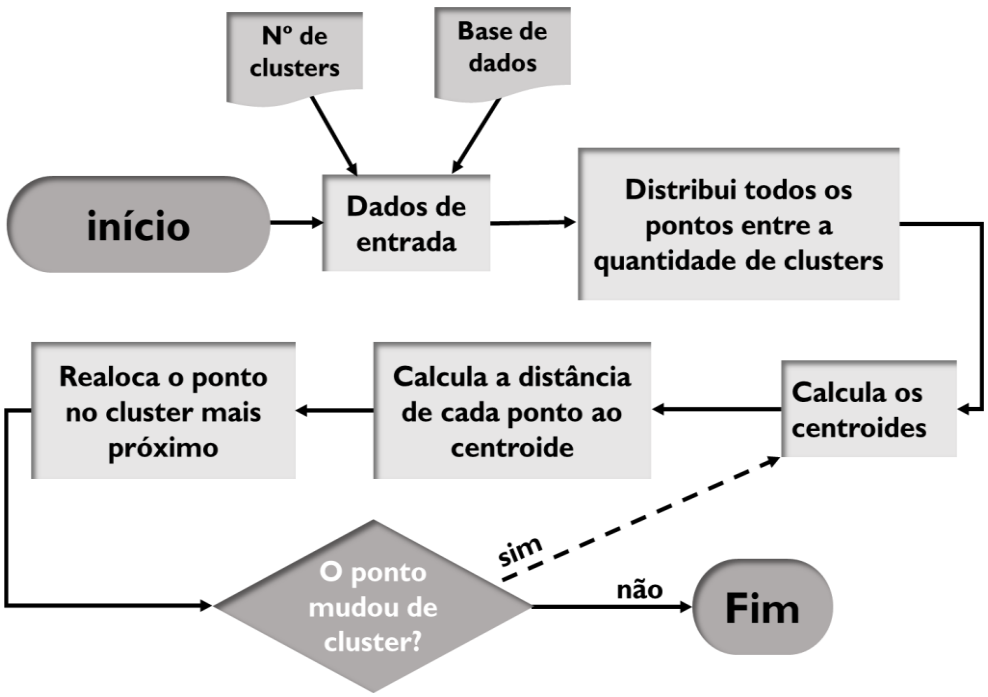


Fonte: Elaborado pela autora (2022).

$$\sum_{i=1}^{k} \left| y_i^{iteração} - y_i^{iteração-1} \right|^2 < erro \qquad (3.4)$$

O código é descrito de forma resumida através do fluxograma da Figura 21

Figura 21. Fluxograma: *K-Means*



Fonte: Elaborado pela autora (2022).

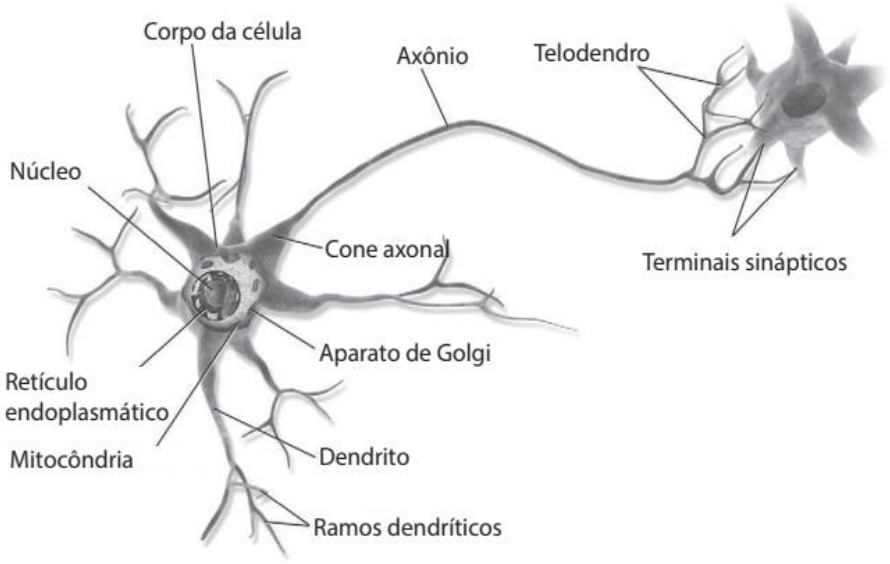
Apesar da simplicidade de funcionamento, o algoritmo apresenta problemas de inicialização dos centroides que são calculados aleatoriamente inicialmente e podem levar a solução para um mínimo local. Além disso, em base de dados onde os exemplos são muito diferentes, o agrupamento pode ser afetado. O *K-means* agrupa um ponto em cada cluster, sendo assim, podem ocorrer erros de agrupamento quando um ponto for equidistante dos centroides de dois clusters diferentes.

3.4 REDES NEURAIS ARTIFICIAIS (RNA)

Redes Neurais Artificiais (RNA's) são modelos matemáticos inspirados no funcionamento do sistema nervoso central de animais. Mcculloch e Pitts (1943) foram os primeiros a introduzir o conceito de redes neurais num trabalho que comparava células nervosas vivas a um circuito eletrônico. O modelo proposto consistia em uma estrutura de resistores variáveis e amplificadores que simulavam conexões sinápticas de um neurônio biológico.

O neurônio biológico é composto basicamente por três partes principais: os dendritos que recebem as informações, o corpo celular que soma e assimila essa informação e o axônio que transmite tal informação gerada pelos impulsos nervosos (Figura 22). A conexão entre os neurônios ocorre pelas sinapses.

Figura 22. Neurônio biológico

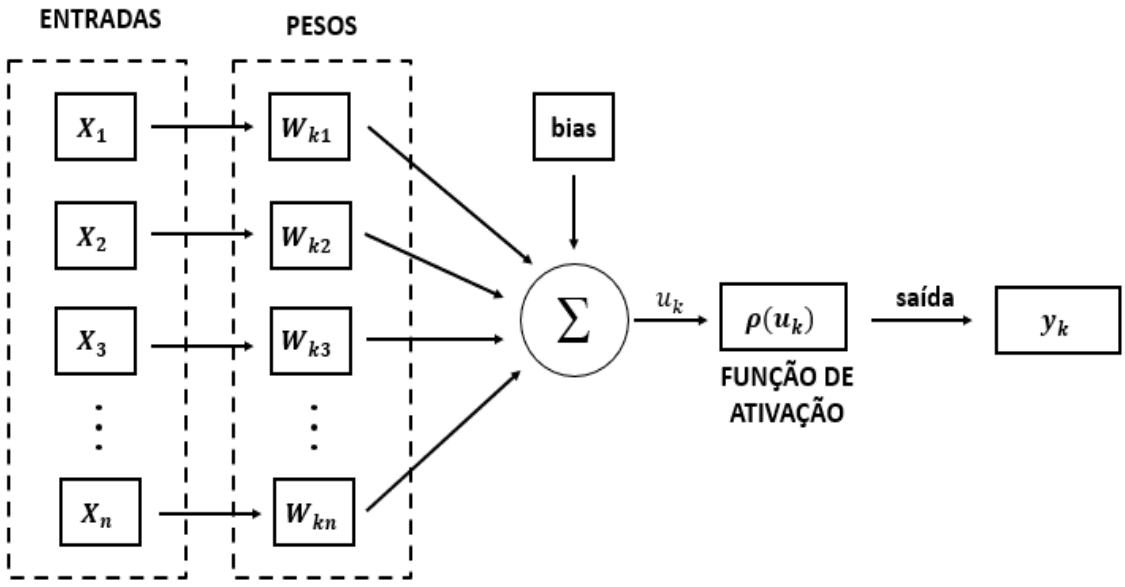


Fonte: Géron (2017, p. 261).

O modelo proposto por Mcculloch e Pitts (1943) sugere que os valores de entrada e saída são binários, portanto, um neurônio influencia ou não outro. Tal afirmação foi contestada por Hebb (1949) em seu trabalho *The Organization of Behavior*, onde afirma que as conexões entre os neurônios são fortalecidas quanto mais são utilizadas, ou seja, se dois neurônios são ativados ao mesmo tempo, suas conexões são fortalecidas; caso contrário, se apenas um neurônio estiver ativado, suas conexões são enfraquecidas. Rosenblatt (1958) propôs o *Perceptron* como o primeiro modelo para aprendizagem supervisionada, chamada de *Perceptron* de Camada Única, sendo essa a forma mais simples de rede neural, usada para classificar padrões linearmente separáveis.

O neurônio artificial é um modelo matemático que imita o neurônio biológico, dessa forma, os dendritos são equivalentes às entradas que são interligados através de pesos (funções sinápticas) ao corpo celular (Figura 23). Os pesos são definidos a partir do estimulo que as entradas oferecem ao neurônio, quanto maior o peso, mais excitatório é o estimulo, do contrário, mais inibitório é o estimulo. O somatório das entradas com seus respectivos pesos é processado por uma função de ativação que determina a saída y_k correspondente ao axônio.

Figura 23. Neurônio artificial



Fonte: Elaborado pela autora (2022).

Considerando o neurônio mostrado na Figura 23, tem-se n entradas $X_j = X_1, X_2, \dots, X_n$, com seus respectivos pesos $W_{jk} = W_{1k}, W_{2k}, \dots, W_{nk}$, em que k é o índice do neurônio analisado, logo:

$$u_k = \sum_{j=1}^{n} W\_jk\, X\_j + bias \tag{3.5}$$

Neste modelo foi considerado o bias, uma constante de valor igual a 1 associada a um peso b_k cuja função é deslocar a função de ativação no eixo. A saída desse neurônio é definida por:

$$y_k = \rho(u_k) \tag{3.6}$$

A função de ativação condiciona a amplitude de saída de um neurônio a um valor finito. Existem vários tipos de funções de ativação, as mais comuns são representadas na Tabela 1.

Tabela 1. Funções de ativação

Nome	Equação
Função degrau simétrico	$y_k = \rho(u_k) = \begin{cases} +1, & u_k \geq 0 \\ -1, & u_k < 0 \end{cases}$
Função lógica	$y_k = \rho(u_k) = \begin{cases} \lambda u_k, & \\ & 0 \leq u_k \leq t + 1, \\ & u_k > t \end{cases}$
Função sigmoide	$y_k = \rho(u_k) = \dfrac{1 - e^{-\lambda u_k}}{1 + e^{-\lambda u_k}}$

Fonte: Elaborado pela autora (2022).

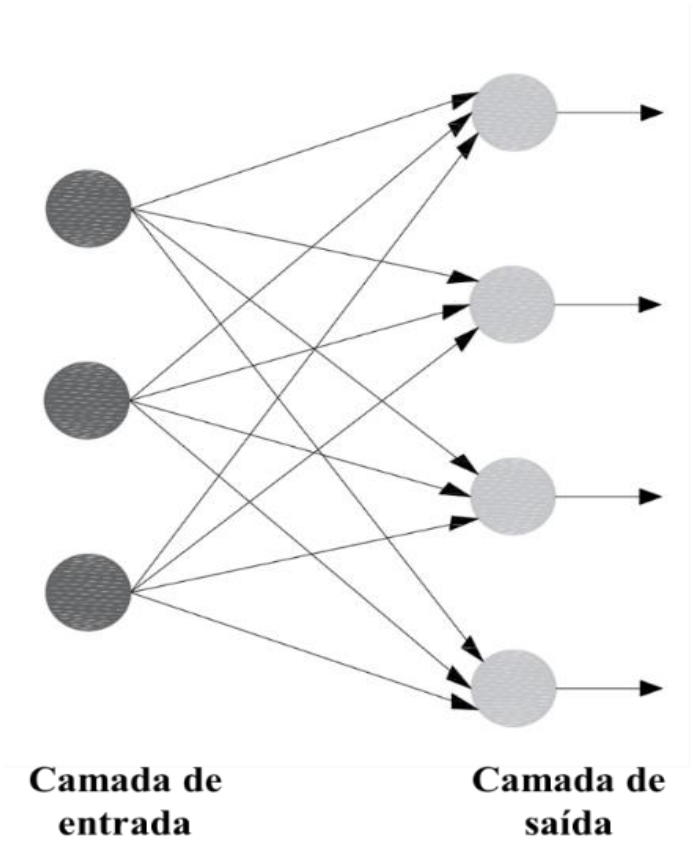
Sendo, λ e t números reais constantes que indicam a inclinação da reta e a suavidade da curva, respectivamente.

3.4.1 Topologias de Redes Neurais Artificiais (RNA)

A topologia de uma rede neural diz respeito à forma como os neurônios dessa rede estão ligados, sendo assim, os algoritmos de treinamento são considerados estruturados. Existem basicamente dois tipos de estruturas de redes:

a) Redes alimentadas adiante (*feed-forward networks*): são as arquiteturas mais utilizadas nas RNA. Os neurônios se dispõem em uma ou mais camadas de forma que o fluxo de dados se dá sempre num único sentido, da camada de entrada para a camada de saída (HAYKIN, 1999). É importante ressaltar que nas redes *feed-forward* não existem conexões entre neurônios de uma mesma camada e todos os neurônios de uma camada são conectados a todos os neurônios da camada seguinte (Figura 24);

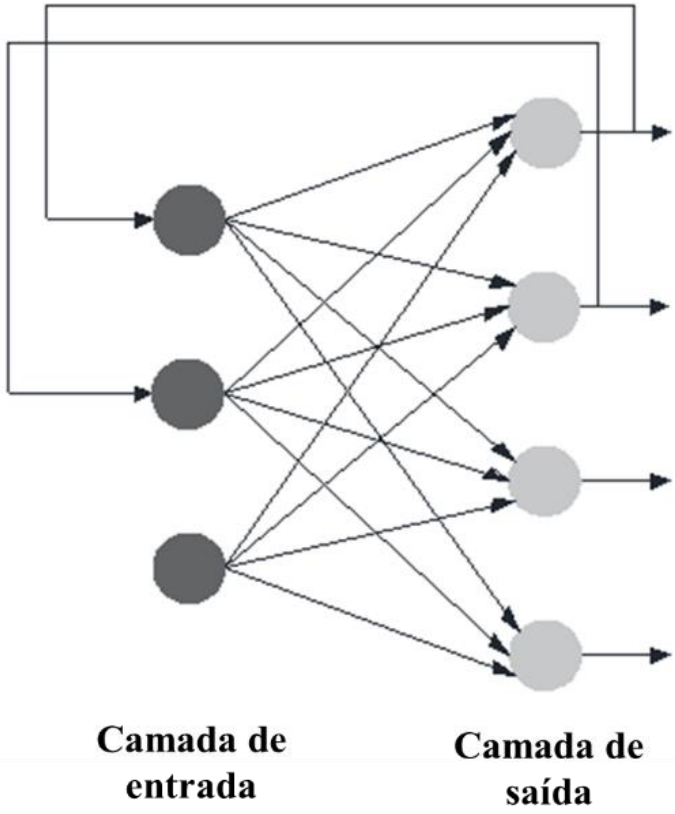
Figura 24. *Rede feed-forward*



Fonte: Elaborado pela autora (2022).

b) Redes recorrentes (*feed-backward networks*): são redes que têm ao menos um laço de realimentação, ou seja, a saída pode alimentar as entradas. As saídas podem alimentar as mesmas camadas ou camadas anteriores (Figura 25).

Figura 25. *Rede feed-backward*



Fonte: Elaborado pela autora (2022).

3.4.2 Processos de aprendizagem

As regras de aprendizagem definem o método pelo qual os pesos devem ser ajustados. Os algoritmos de aprendizagem são determinados por esses modelos. Segundo Haykin (1999), existem quatro métodos de aprendizagem:

a) Aprendizagem por correção de erro: é realizado um ajuste dos pesos através do erro obtido pela diferença entre o valor da saída e o valor esperado. Dessa forma o erro decresce a cada iteração;

b) Aprendizado *hebbiano*: é a mais antiga e conhecida de todas as regras de aprendizado. Em seu postulado Hebb (1948) afirma que se dois neurônios são ativados simultaneamente a conexão entre eles é fortalecida. Esse aprendizado ocorre localmente a medida em que os neurônios são ativados;

c) Aprendizagem de Boltzmann: nome dado em homenagem ao seu criador *Ldwig Boltzman* é um modelo de aprendizado estocástico inspirado na mecânica estatística. Uma RNA que usa esse tipo de aprendizado é denominada Máquina de

Boltzman (MB). A MB é constituída por uma rede de neurônios recorrentes simétrica que assumem apenas dois estados, ligado (+1) e desligado (-1). Se caracteriza por uma função de energia determinada pelos estados dos neurônios. A atualização da condição dos neurônios é feita aleatoriamente gerando uma saída segundo a distribuição de Boltzman de mecanismos estatísticos;

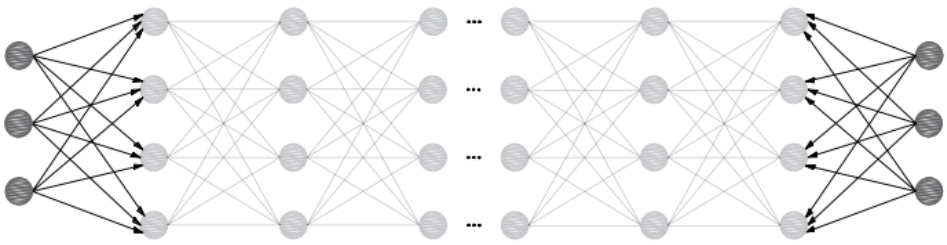
d) Aprendizagem competitiva: Neste modelo os neurônios competem entre si e somente um fica ativo por iteração, sendo o vencedor o que tiver maior similaridade com o padrão de entrada. Todos os pesos dos neurônios próximos ao neurônio vencedor terão seus valores ajustados.

3.4.3 Redes *Multilayer Perceptron* (MLP)

O modelo *Perceptron* desenvolvido por Rosenblatt (1958) é baseado no modelo de Mcculloch e Pitts (1943) acompanhado de uma regra de treinamento para ajuste dos pesos, é uma das redes neurais mais simples existentes consistindo de um único neurônio com pesos sinápticos ajustáveis e bias. É capaz de resolver apenas problemas linearmente separáveis, logo só consegue classificar dois grupos.

A MLP é uma rede neural semelhante ao *Perceptron* com mais de uma camada interna, como mostrado na Figura 26, o que a torna capaz de resolver problemas não linearmente separáveis. O treinamento da rede é feito através algoritmo *backpropagation*, sendo este dividido em duas etapas, a fase para frente (*forward)* onde as entradas são passadas através da rede e são obtidas as saídas (essa etapa também é conhecida como fase de propagação) e a fase para trás (*backward)* onde se calcula o gradiente da função de perda na camada final da rede e usa-se esse gradiente para aplicar a regra da cadeia (*chain rule*) para atualizar os pesos na rede (etapa também chamada de retro propagação).

Figura 26. Rede MLP



Fonte: Elaborado pela autora (2022).

Com base em Morais (2016) Pode-se resumir as etapas do *backpropagation* da seguinte forma:

a) Iniciar os pesos com valores aleatórios e não nulos;
b) Apresentar um padrão de entrada e propagá-lo até a saída da rede;
c) Comparar a saída predita (p) com a saída desejada (d). Caso seja diferente deve-se calcular o erro quadrático entre a diferença da saída desejada (d) e a saída predita (p) em todos os nós das camadas de saída (m), conforme Equação 3.7;

$$e = \frac{1}{2}\sum_{j=1}^{m}(d_j - p_j)^2 \tag{3.7}$$

d) Calcula-se os gradientes locais *(δ)* dos neurônios da camada de saída, dado pela Equação (3.9). O gradiente do neurônio j é resultante do produto entre o erro desse neurônio com a derivada da função de ativação *(φ′)* aplicada ao campo local induzido v_j calculado na Equação (3.8), onde w_{ij} representa o peso de entrada associado a cada neurônio j;

$$v_j = \sum_{i=0}^{m} w_{ij} - p_i \tag{3.8}$$

$$\delta_j = (d_j - p_j)\varphi' v_j \tag{3.9}$$

e) Ajustar os pesos da camada de saída utilizando as Equações em 3.10 e 3.11;

$$\Delta w_{ji(n)} = \eta \delta_{j(n)} p_{i(n)} \tag{3.10}$$

$$\Delta w_{ji(n+1)} = w_{ji} - \Delta w_{ji(n)} \tag{3.11}$$

f) Repetir os passos de e) a f) para todos os padrões de treinamento (1 época);

g) Calcular a cada época o erro médio quadrático (EMQ) para o treinamento utilizando a Equação (3.12), onde N representa o número de padrões utilizados no treinamento;

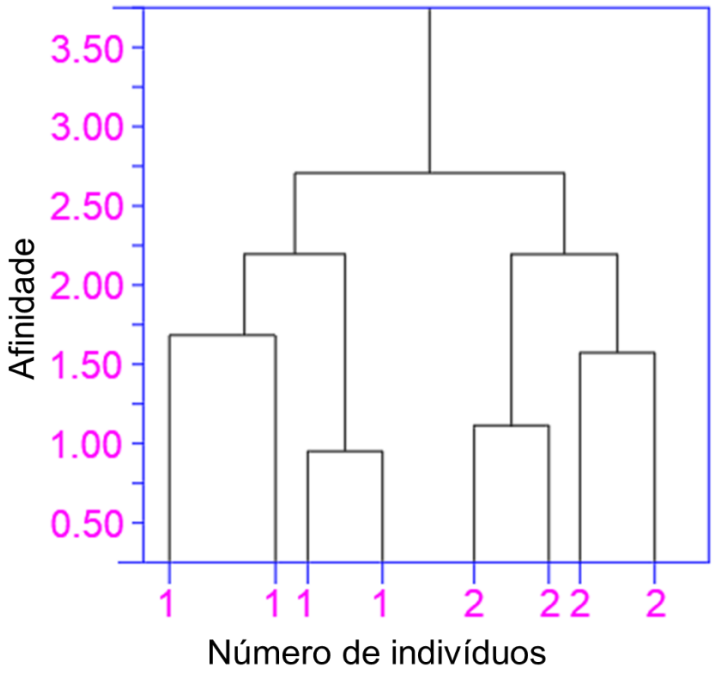
$$EMQ = \frac{1}{N}\sum_{j=1}^{N} E(j) \tag{3.12}$$

h) Se o EMQ for maior que o valor desejado (VD) ou se o contador de épocas for menor que o número máximo de épocas (NMEp), repetir o passo g). Em caso contrário pare.

3.5 CLUSTERIZAÇÃO HIERÁRQUICA

O método de clusterização hierárquico organiza o conjunto de dados em uma estrutura de classes conforme a proximidade dos exemplos. Os resultados são estruturados em um dendrograma que, basicamente, é uma árvore, onde o tronco representa todo o conjunto de dados e os ramos representam os grupos de indivíduos, a Figura 27 mostra um exemplo de um dendrograma. O resultado pode ser obtido cortando o dendrograma no ponto onde a quantidade de ramos dessa árvore for igual à quantidade de clusters desejado, ou seja, cada nó representa um cluster.

Figura 27. Dendrograma



Fonte: Elaborado pela autora (2022).

O agrupamento hierárquico pode ser dividido em dois tipos de abordagem de agrupamento, que são o método *bottom-up* (agrupamento aglomerativo) e o método *top-down* (agrupamento por divisão). Na primeira abordagem cada exemplo é considerado um grupo individual e vão se fundindo até formar um grupo principal com todos os indivíduos. Na metodologia *top-down*, inicialmente se considera o grupo completo que é repetidamente dividido até cada indivíduo ficar sozinho.

Segundo Murtagh e Contreras (2017) antes do agrupamento de dados é necessário realizar as medições entre os exemplos, ou seja, medir as distâncias em relação uns aos outros para então fazer o agrupamento, sendo essas conhecidas como medidas de similaridade. Os métodos para calcular a distância mais comuns são: distância Euclidiana, distância de *Manhattan*, distância de *Hamming* e ângulo entre vetores.

O que diferencia os algoritmos de agrupamento é a forma como é calculada a distância entre os exemplos, representada por $d(C_i, C_j)$. Os métodos mais comuns de agrupamento para determinar a distância entre agrupamentos são: ligação simples, ligação completa, centroides, mediana, médias das distâncias e método *Ward* (ANDERBERG, 1973).

No método *Ward*, utilizado neste trabalho, busca minimizar a soma de quadrados dentro do grupo. Considere um cluster i, SS_i é o somatório da distância de cada elemento do cluster até o centroide do cluster, Equação 3.13.

$$SS_i = \sum_{j=1}^{n_i} (x_{ij} - c_i)^2 \tag{3.13}$$

Sendo, x_{ij} o elemento j pertencente ao cluster i, n_i o número de elementos do cluster i e c_i é o centroide do cluster i. a distância entre os clusters é definida na equação 3.14:

$$d(C_i, C_l) = \left[\frac{n_i n_l}{n_i + n_l}\right](c_i - c_l)^2 \tag{3.14}$$

A cada passo do algoritmo de agrupamento, os dois clusters que obtiverem menor distância entre si serão agrupados, formando um único cluster novo.

CAPÍTULO 4: METODOLOGIA

4.1 INTRODUÇÃO

Este capítulo trata da metodologia utilizada para a implementação dos modelos fazendo uma breve descrição do software utilizado bem como as ferramentas que foram empregadas para realizar o processamento e tratamento da base de dados e para avaliação dos resultados.

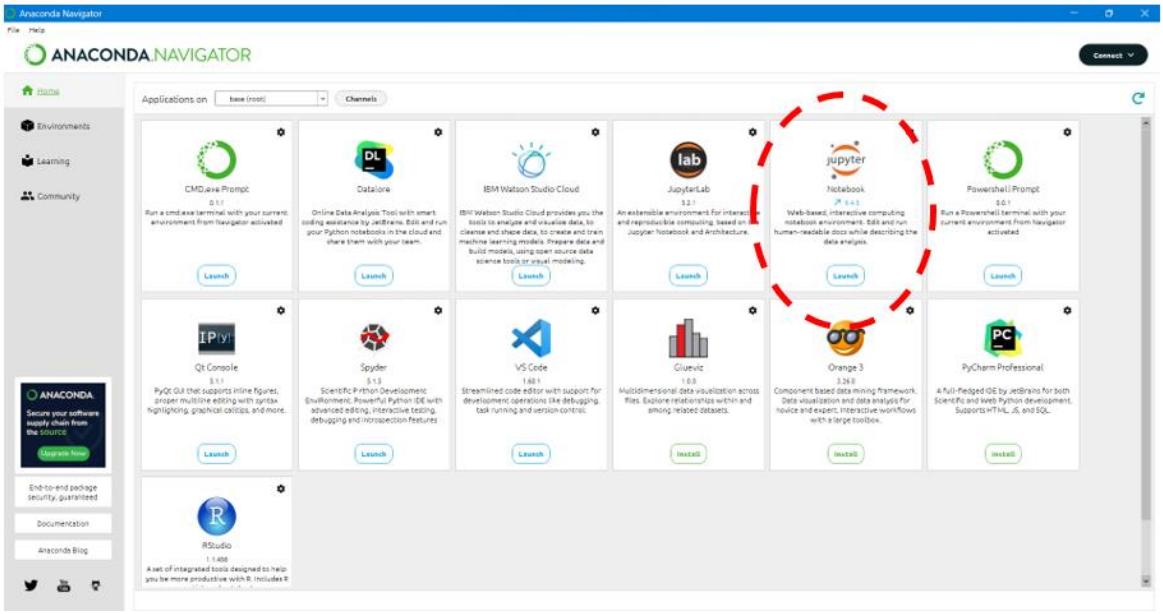
4.2 *SOFTWARE* UTILIZADO

Os resultados desse trabalho foram obtidos a partir de simulações utilizando o software Anaconda, uma distribuição de código aberto (*open source*) do *Python* direcionada para aplicações na área de ML. O programa concentra milhares de bibliotecas de ML pré instaladas além de possibilitar a instalação de pacotes adicionais, administrados pelo Conda, o gerenciador de pacotes do Anaconda. O software foi desenvolvido pela empresa Anaconda Inc. e está disponível em www.anaconda.com com uma versão gratuita para uso individual e uma versão paga voltada para empresas de desenvolvimento.

Na Figura 28 pode-se ver que o Anaconda conta com ambientes de desenvolvimento integrado (IDE, do inglês, *Integrated Development Environment*). Uma IDE é uma ferramenta para desenvolvimento de softwares com várias funções que facilitam o trabalho de programação, possibilitando escrever e compilar o código em um mesmo ambiente.

Dentre as IDE's disponíveis no Anaconda estão o *Orange*, *Spyder*, *PyCharm* e a *Jupyter Notebook*, escolhida aqui por sua possibilidade de mesclar texto e código com facilidade, permitindo uma melhor compreensão das etapas da implementação se assemelhando à um caderno onde será escrito uma história em blocos. Neste trabalho foi utilizada a versão do *Python* 3.8.12.

Figura 28. Interface do Anaconda

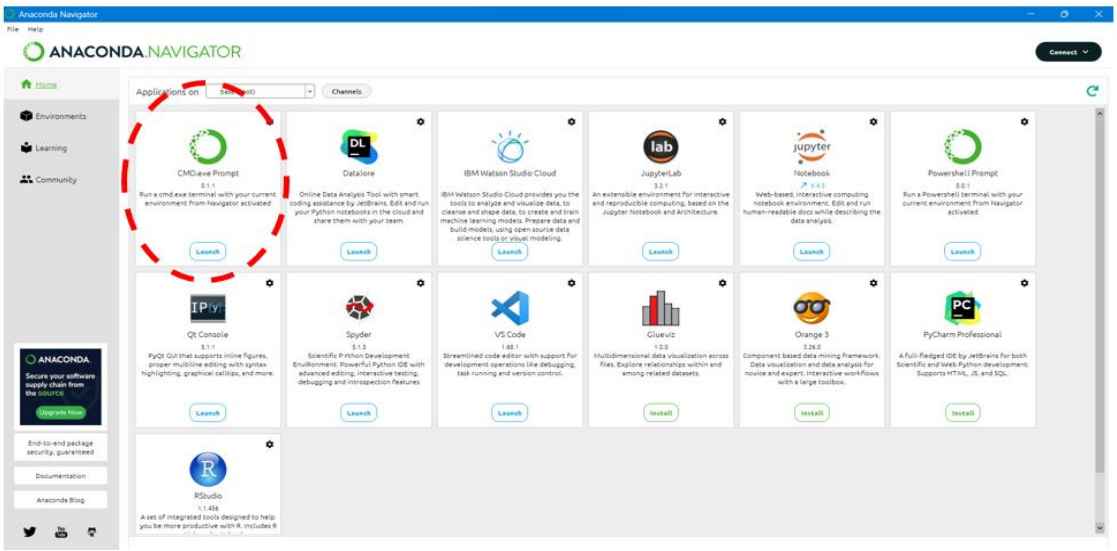


Fonte: Anaconda (2022).

4.2 BIBLIOTECAS

A distribuição Anaconda conta com um vasto acervo bibliotecas voltadas para ML. As bibliotecas tem várias funções, ferramentas e técnicas que ajudam a reduzir o uso de códigos de um programa. As bibliotecas podem ser instaladas facilmente utilizando *prompt* de comando do Anaconda mostrada na Figura 29.

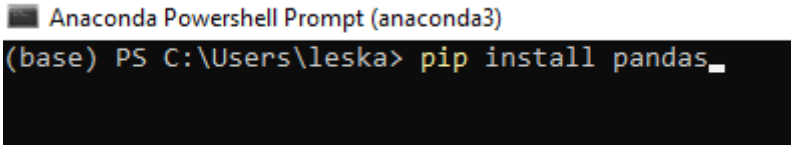
Figura 29. Prompt de comando do Anaconda



Fonte: Anaconda (2022).

Para proceder com a instalação basta digitar o comando *pip install* no *prompt* de comando do Anaconda mais o nome do pacote que precisa ser instalado conforme Figura 30 o *pip* é o gerenciador de pacotes do *python,* com ele a instalação das bibliotecas só precisa ser feita uma única vez e quando for necessário utilizá-las só será preciso importá-las no código.

Figura 30. *Pip install*



Fonte: Anaconda (2022).

O *pip* possui vários outros comandos leis no gerenciamento de pacotes, como o *pip uninstall* (desinstalação de um pacote), *pip install upgrade* (atualização de pacotes) e o *pip list* (entrega uma lista com todas as bibliotecas instaladas no *phyton* do seu computador e suas respectivas versões). As principais bibliotecas empregadas nas simulações foram:

- *NumPy* (*Numerical Python*): O *NumPy* é um pacote do *Phyton* que oferece os algoritmos necessários à maioria das aplicações científicas que envolvam dados numéricos em *Python*. Contém funções básicas de álgebra linear, transformações de *Fourier*, recursos avançados de números aleatórios e ferramentas para integração com outras linguagens de programação, como *Fortran*, *C* e *C++.* É destinada a realizar operações em *arrays* multidimensionais, conhecidas como *ndarray*. Tem disponível algoritmos para tratamento de dados, geração de subconjuntos e filtragens, estatísticas descritivas, manipulação de dados relacionais, manipulações de dados em grupos, entre outros tipos de processamento;
- Pandas (*Python and Data Analysis*): utilizada em operações e manipulações de dados estruturados. Amplamente utilizada para coleta e preparação de dados. A biblioteca permite trabalhar com diferentes tipos de dados como matrizes, dados em *Excel*, arquivos de extensão .**csv*, entre outros. Os principais objetos do Pandas são o *dataframe* e as séries. Neste trabalho, o

Pandas é utilizado para manipular um *dataframe* contendo os dados de consumo de 42.372 clientes;

- *Matplotlib* (*Math Plotting Library*): para gerar uma grande variedade de gráficos, tais como histogramas, gráfico de linhas e mapa de calor;
- *SkLearn* ou *Scikit Learn* (*SciPy Toolkit Learn*): Tem por base a *NumPy*, *SciPy* e *Matplotlib*. Contém muitas ferramentas para aprendizado de máquina e modelagem estatística, incluindo classificação, regressão, clusterização e redução de dimensionalidade. Esta biblioteca será melhor explanada no tópico 4.4 deste capítulo;
- *Seaborn* (*Statistical Data Visualization*): Geração de gráficos atraentes e informativos em *Python*. Tem por base a *Matplotlib*. Visa tornar a visualização uma parte central da exploração e compreensão dos dados;
- *Statsmodels* (*Statistical Models*): Permite explorar dados, estimar modelos estatísticos e executar testes. Uma lista extensa de estatísticas descritivas, funções de plotagem e de resultados estão disponíveis para diferentes tipos de dados e para cada estimador.

4.3 *SCIKITLEARN*

A *Scikitlearn* é uma biblioteca de código aberto criada a partir do *NumPy*, *SciPy* e *matplotilib*. Nela é possível encontrar algoritmos para diversas aplicações em ML como redes neurais, classificação, regressão, clusterização, redução de dimensionalidade, entre outros. As funcionalidades na *Sciktlearn* podem ser divididas em cinco grupos (PACHECO, 2021):

- Estimadores básicos;
- Pré-processamento e transformadores;
- Avaliação de modelos;
- Pipelines;
- Busca automática de parâmetros.

Os estimadores básicos são os algoritmos de ML dentro da biblioteca como *MLP classifier*, *K-means*, *KNN*, etc. Os estimadores seguem um padrão de funcionamento que inclui uma função de treinamento *fit* e uma função de predição *predict.* Cada algoritmo possui parâmetros específicos que devem ser definidos no

início da aplicação, do contrário, o *scikitlearning* aplicará o valor *default* (parâmetros padrão) daquele algoritmo.

As funções de pré-processamento e transformação são capazes de estruturar e reorganizar os dados para que as etapas de treinamento e predição sejam mais eficientes. Dentre as funções de pré-processamento estão normalização, padronização ao longo de um eixo, binarização de rótulos, e mais.

A avaliação de modelos é feita através de funções como *train_test_split* que divide o conjunto de dados em partição de treinamento e teste de forma aleatória para evitar bias (tendências) e *accuracy_score* que basicamente calcula a eficiência de determinado algoritmo. O objetivo dessas funções é avaliar os modelos de ML.

A *sklearn* também oferece uma maneira de criar um *pipeline* para juntar algumas etapas em apenas uma chamada de função utilizado o subpacote *sklearn.pipeline* dessa forma, diminuindo o tamanho do código e consequentemente o esforço envolvido. Além disso a *sklearn* conta também com algumas ferramentas para busca automática de parâmetros como a *RandomizedSearchCV* cujo objetivo é encontrar o melhor conjunto de parâmetros para um estimador através de amostras.

Neste trabalho a biblioteca *sklearn* foi utilizada para todas as funcionalidades citadas acima, exceto as *pipelines*. As funções que aqui merecem destaque são *sklearn.neural_network. MLPClassifier* e *sklearn.cluster.KMeans* descritos nos tópicos a seguir.

4.4 BANCO DE DADOS

A base de dados utilizada nesta dissertação é a mesma utilizada por Zheng *et al*. (2018) em seu trabalho, onde propõe um novo método de detecção de roubo de eletricidade baseado em redes neurais convolucionais amplas e profundas. O banco é fornecido pela *State Grid Corporation of China* (SGCC) e contém os dados de consumo de eletricidade de 42.372 clientes de eletricidade em 1.035 dias desde o dia 01/01/2014 até o dia 31/10/2016 (Figura 31).

Dentre os 42.372 consumidores 3.615 são rotulados como desonestos e 38.757 são honestos. Por questão de processamento da máquina utilizada (*Notebook Lenovo IdeaPad* S145-15API com processador *AMD Ryzen* 5 3500U *with Radeon Vega Mobile Gfx* 2.10 GHz, 8 GB de RAM e sistema operacional *Windows* 10) a base de

dados foi reduzida para 7.230 consumidores, dessa forma ficando 3.615 clientes honestos e 3.615 clientes desonestos.

Figura 31. *Dataframe*

	CONS_NO	FLAG	01/01/2014	02/01/2014	03/01/2014	04/01/2014	05/01/2014	06/01/2014	07/01/2014	08/01/2014	...	22/10/20
0	0387DD8A07E07FDA6271170F86AD9151	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	7
1	01D6177B5D4FFE0CABA9EF17DAFC2B84	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0
2	4B75AC4F2D8434CFF62DB64D0BB43103	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	10
3	B32AC8CC6D5D805AC053557AB05F5343	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	12
4	EDFC78B07BA2908B3395C4EB2304665E	1.0	2.90	5.64	6.99	3.32	3.61	5.35	4.73	3.68	...	14
...	...	...	...	...	...	...	...	...	...	...	...	
42367	F1472871E1AFF49D4289564B6377D76C	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	3
42368	F3C8BBCD2DC26C1E0249DEEF6A4256B7	0.0	2.70	0.00	0.00	5.72	6.05	5.81	3.07	4.04	...	3
42369	A9A0FE83467A680FBFB0DBFC910DF227	0.0	0.58	1.16	0.92	0.98	1.54	1.38	0.89	0.70	...	0
42370	D9A6ADA018FA46A55D5438370456AA45	0.0	16.89	15.15	19.28	17.19	16.80	17.48	17.86	23.99	...	15
42371	F3406636BAD1E6E0826E8EDDC9A1BF00	0.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	10

42372 rows × 1036 columns

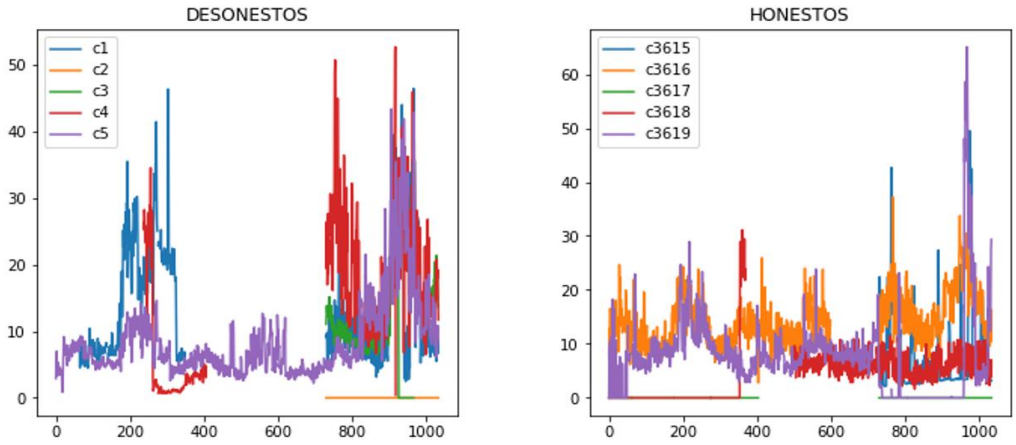
Fonte: Elaborado pela autora (2022).

As informações do banco de dados estão armazenadas em um arquivo tipo CSV (*Comma Separated Values*, em português, valores separados por vírgulas) e sua manipulação é feita utilizando a biblioteca Pandas que cria um objeto do tipo *DataFrame*.

O *DataFrame* se comporta como uma planilha onde as linhas são os consumidores e as colunas são os dias em que o cliente foi avaliado. Uma das colunas intitulada “FLAG” indica se o cliente cometeu fraude ou não, sendo o valor “zero” para clientes honestos e o valor “um” para clientes desonestos. Essa coluna foi retirada do *DataFrame* para aplicação dos modelos de ML e posteriormente foi inserida novamente no banco de dados para comparação entre o rótulo real e o rótulo predito pelo algoritmo.

A Figura 32 mostra o comportamento dos consumidores honestos e desonestos ao longo de três anos. Como se pode perceber, as diferenças no comportamento desses consumidores não são evidentes a olho nu. O que ratifica a necessidade de modelos de análise de dados capazes de expressar este comportamento e classificar os diversos tipos de clientes.

Figura 32. Consumidores honestos X Consumidores desonestos



Fonte: Elaborado pela autora (2022).

4.5 PRÉ-PROCESSAMENTO DE DADOS

O pré-processamento de dados é uma das etapas mais importantes do processo de *Machine learning*, pois a qualidade do banco de dados pode influenciar diretamente no resultado do modelo. Nessa etapa é feita a transformação do *dataset* puro para um banco de dados contendo as informações úteis para o problema já citado anteriormente de forma organizada e estruturada. No pré-processamento algumas das atividades mais comuns são: identificação de inconsistências, identificação de atributos duplicados, identificação e remoção de dados faltantes, identificação de outliers, balanceamento de classes, seleção de atributos, entre outros (BATISTA, 2003).

4.5.1 Tratamento de dados faltantes

Os dados de consumo de eletricidade geralmente contêm valores ausentes ou errôneos; isso é causado principalmente por vários motivos, como por exemplo a falha de medidores, a transmissão não confiável de dados de medição (no caso de medidores inteligentes), a manutenção não programada do sistema e problemas de armazenamento. Esta ausência de dados causa lacunas, “buracos de informações” na base de dados que torna o seu processamento inviável. Num *DataFrame* esses dados faltantes são representados por NAN (*Not a Number*), como mostrado na Figura 33.

Figura 33. Dados faltantes

	CONS_NO	FLAG	01/01/2014	02/01/2014	03/01/2014	04/01/2014	05/01/2014	06/01/2014	07/01/2014	08/01/2014	...	22/10/201
0	0387DD8A07E07FDA6271170F86AD9151	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	7.1
1	01D6177B5D4FFE0CABA9EF17DAFC2B84	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	0.0
2	4B75AC4F2D8434CFF62DB64D0BB43103	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	10.9
3	B32AC8CC6D5D805AC053557AB05F5343	1.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	...	12.8
4	EDFC78B07BA2908B3395C4EB2304665E	1.0	2.90	5.64	6.99	3.32	3.61	5.35	4.73	3.68	...	14.2
...	...	...	...	...	...	...	...	...	...	...	...	.
7225	C80C0A967166EDC265A32F0E6C79B768	0.0	0.40	1.10	0.89	0.90	0.94	1.93	0.55	0.00	...	0.4
7226	A4F8D505DDA447CFFB0CB0D355FD379D	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.0
7227	2B891A7C4EDE71FF060CE6F53DFF242D	0.0	10.91	9.32	9.07	6.35	10.33	10.17	9.27	9.28	...	14.3
7228	0A7F166D6A883A69F4C01FFDAA2D17B9	0.0	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	...	0.0
7229	16AAC19A409470767008364C0182A532	0.0	0.00	1.57	0.00	2.77	2.60	2.01	0.00	0.00	...	5.4

7230 rows × 1036 columns

Fonte: Elaborado pela autora (2022).

Para realizar a substituição desses dados faltantes foi usado a metodologia adotada em Zheng *et al*. (2018) como explica a Equação 4.1:

$$f(x_i) = \begin{cases} \dfrac{x_{i-1} + x_{i+1}}{2} & x_i \in NAN, x_{i-1}, x_{i+1} \notin NAN \\ 0, & x_i \in NAN, x_{i-1} \, ou \, x_{i+1} \in NAN \\ x_i & x_i \notin NAN \end{cases} \tag{4.1}$$

Sendo x_i a medição do consumo no dia i. Em resumo, se o dado em análise for um NAN e os dados antecessores e sucessores na base de dados deste não forem NAN então, o valor será substituído pela média dos dados antes e depois. Caso o dado seja um NAN e os dados antes ou depois também forem NAN, então o valor é substituído por zero. Por último, se o dado analisado não for NAN então o valor não é substituído.

4.5.2 Normalização

O banco de dados utilizado apresenta valores muito diferentes entre os consumidores já que as faixas de consumo variam de cliente para cliente. Essas diferenças devem ser levadas em conta pois podem interferir no resultado final da simulação. A normalização desses dados é necessária a fim de evitar que algum valor se sobreponha a outro inserindo bias no algoritmo.

A normalização por reescala é um dos métodos mais comuns de normalização (PADILHA; CARVALHO, 2017). No método por reescala um novo intervalo é definido, geralmente o intervalo [0,1] e pode ser calculado a partir da Equação 4.2.

$$x_{ij\ normalizado} = \frac{x_{ij} - min_j}{max_j - min_j} \tag{4.2}$$

Sendo x_{ij} o consumo do cliente j no dia i. Foi feita uma normalização por reescala, deixando todos os valores dentro do intervalo [0,1], para isso foi utilizado a biblioteca *sklearn.preprocessing* e o comando *minmax_scale*. O *minmax_scale* normaliza os dados por coluna, por isso foi necessário transpor o *DataFrame* para que a normalização fosse feita para cada cliente separadamente.

4.5.3 Padronização

A padronização tem o mesmo objetivo da normalização, ou seja, colocar todos os exemplos de um *dataset* na mesma ordem de grandeza. A diferença entre os dois métodos está no fato de que na padronização a média das variáveis é igual a zero e o desvio padrão é igual a 1. A padronização não coloca os exemplos dentro de um intervalo específico como na normalização, podendo ser um problema para alguns algoritmos como redes neurais, no entanto a padronização é mais eficiente em caso de *datasets* com muitos *outliers* (GÉRON, 2017).

Neste trabalho, foi utilizado o método de padronização dos dados, mais conhecido pela função *Z-Score*, mostrada na Equação 4.3.

$$Z_{ij\ padronizado} = \frac{x_{ij} - \mu_j}{\sigma_j} \tag{4.3}$$

Sendo x_{ij} o consumo do cliente j no dia i, μ_j e σ_j são, respectivamente, a média e desvio padrão dos consumos do cliente j. Para realizar a padronização foi utilizada a função *StandardScaler*, acondicionada na biblioteca *sklearn.preprocessing*.

4.6 MÉTRICAS DE AVALIAÇÃO EM *MACHINE LEARNING*

A avaliação dos modelos utilizados em ML pode afetar diretamente na tomada de decisões, podendo gerar desde a ineficiência de processos até a realização de prejuízos no caso de dados financeiros. A métrica de avaliação vai informar o quão confiável é o modelo utilizado e por isso é importante que a escolha do método de avaliação seja coerente com o modelo utilizado. As métricas de avaliação utilizadas neste trabalho são:

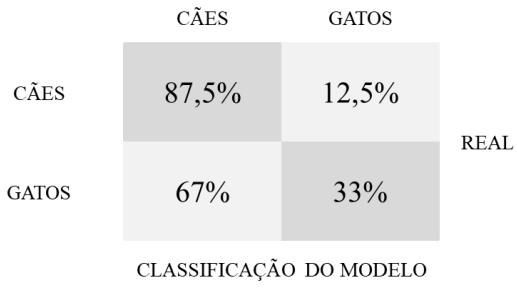
- Matriz de confusão;
- Acurácia;
- Precisão.

4.6.1 Matriz de confusão

A matriz de confusão indica quantos exemplos existem em cada grupo analisado. São analisados os rótulos reais e os rótulos preditos pelo modelo. Dessa forma pode-se identificar quantos exemplos o algoritmo rotulou corretamente. Na Figura 34 pode-se ver um exemplo do funcionamento de uma matriz de confusão em um modelo para classificação de cães e gatos.

Na Figura 34 a matriz de confusão mostra que 87,5% dos cães foram classificados corretamente pelo algoritmo e apenas 12,5% foram classificados incorretamente. Já para os gatos tem-se 33% de acertos e 67% de erros. A matriz de confusão ajuda a entender se o algoritmo está favorecendo uma das classes. Para utilizar a matriz de confusão no *Python* utiliza-se o subpacote *sklearn.metrics* instalado no *sklearn*.

Figura 34. Matriz de confusão



Fonte: Elaborado pela autora (2022).

4.6.2 Acurácia

A acurácia mostra quantos exemplos foram classificados corretamente, independente da classe. Pode ser definida pela equação 4.4

$$Acurácia = \frac{Total\ de\ acertos}{Total\ de\ exemplos} \tag{4.4}$$

Tomando como base o exemplo anterior temos uma acurácia de 67%, no entanto, sabe-se, pela matriz de confusão, que o algoritmo não teve uma boa proporção de acertos para gatos, dessa forma, o método da acurácia é bom para se ter uma avaliação global do algoritmo, mas não é muito confiável quando se quer analisar qual é a origem dos erros.

4.6.3 Precisão

O método da precisão traz uma abordagem mais minuciosa da acurácia. É definida como a razão entre os acertos de uma classe e o total de exemplos daquela classe, conforme Equação 4.5. Aproveitando o exemplo de cães e gatos novamente, é obtida uma precisão de 87,5% para cães e de 33,7% para gatos.

$$precisão = \frac{Total\ de\ acertos\ da\ classe\ x}{Total\ de\ exemplos\ da\ classe\ x} \tag{4.5}$$

CAPÍTULO 5: RESULTADOS

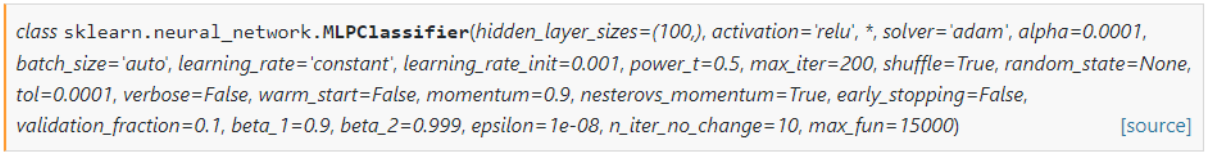
5.1 INTRODUÇÃO

Foram utilizados quatro modelos para realizar a classificação dos consumidores em honestos e desonestos, são o *MLPClassifier*, *K-means*, clusterização hierárquica e *DBscan*. Para cada modelo foram feitos três testes, sendo o primeiro utilizando a base de dados sem normalização, o segundo usando a base de dados normalizada e o terceiro teste utilizando a padronização dos dados.

5.2 *MLPCLASSIFIER*

O *MLPClassifier* faz parte do pacote *scikit-learn* inserido no subpacote *sklearn.neural_network*. Para utilizar esta função é necessário informar o valor de seus parâmetros como mostrado na Figura 35.

Figura 35. Parâmetros MPL*Classifier*



Fonte: Scikit-learn (2022).

Todos esses parâmetros já vêm com um valor pré-definido (*default*) e foram alterados apenas aqueles que impactaram no resultado do modelo, apresentados na Tabela 2.

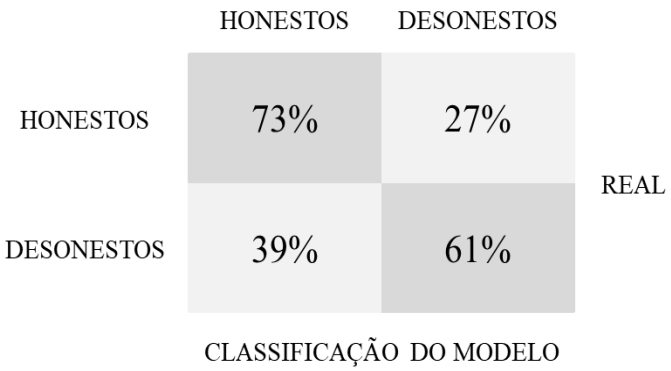
Tabela 2. Parâmetros *MLPClassifier*

Nome	Valor utilizado	Função
hidden_layer_sizes	(100,)	Número de neurônios em cada camada oculta
activation	relu	Função de ativação
solver	adam	Algoritmo de otimização dos pesos
learning_rate	'adaptive'	Roteiro de taxa de aprendizado para atualização dos pesos
learning_rate_init	0,00001	Taxa de aprendizado inicial usada. Controla o tamanho do passo na atualização dos pesos.
max_iter	10000	Número máximo de iterações

Fonte: Elaborado pela autora (2022).

O teste 1 foi realizado utilizando a base de dados sem normalização. Na Figura 36 é mostrada a matriz de confusão dessa simulação.

Figura 36. Matriz de confusão *MLPClassifier* – Teste 1

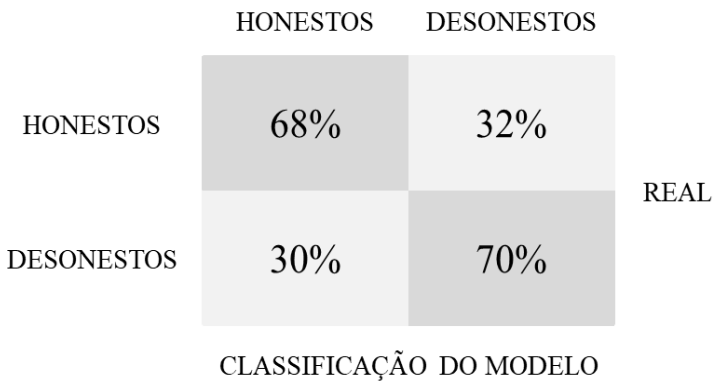


Fonte: Elaborado pela autora (2022).

Nota-se que a pontuação de acurácia do modelo é de 67%, um bom resultado, contudo, o grupo de perfil de consumidores desonestos apresentou pior performance enquanto que o grupo de consumidores honestos teve uma quantidade de acertos melhor. A diferença na quantidade de acertos de uma classe para outra pode indicar que o modelo tende a favorecer a classe de consumidores honestos. O tempo de processamento deste teste foi de 2,479 minutos.

O teste 2 foi realizado utilizando a base de dados normalizada. Na Figura 37 é mostrada a matriz de confusão desse teste.

Figura 37. Matriz de confusão *MLPClassifier* – Teste 2

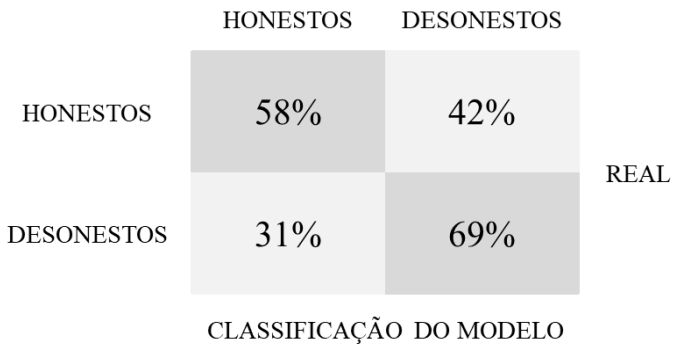


Fonte: Elaborado pela autora (2022).

Observa-se que a acurácia desse modelo foi de 69%, um pouco melhor que o modelo anterior. Ambas as classes de consumidores têm quantidades de acertos similares, dando a entender que o modelo não foi tendencioso, no entanto, o tempo de execução foi de 17,30 minutos.

No teste 3 utilizou-se a base de dados padronizada. Na Figura 38 é mostrada a matriz de confusão dessa situação.

Figura 38. Matriz de confusão *MLPClassifier* – Teste 3



Fonte: Elaborado pela autora (2022).

O teste 3 teve uma acurácia de 63,5%, sendo isto uma piora considerável, além disso o modelo favorece a classe de clientes “desonestos”. O tempo de execução desta situação foi de 4,72 minutos. A Tabela 3 resume os resultados dos três primeiros testes.

Tabela 3. Resumo *MLPClassifier*

Teste	Modelo	Tempo	Acurácia
Teste 1	MLP puro	2,47	67%
Teste 2	MLP normalizado	17,3	69%
Teste 3	MLP padronizado	4,72	63%

Fonte: Elaborado pela autora (2022).

5.3 *K-MEANS*

O *K-means* está incluso no subpacote *sklearn.cluster.KMeans*. A Figura 39 mostra os parâmetros de entrada deste algoritmo.

Figura 39. Parâmetros *K-means*

```
classe sklearn.cluster. KMeans ( n_clusters = 8 , * , init = 'k-means++' , n_init = 10 , max_iter = 300 , tol = 0.0001 , verbose = 0
, random_state = None , copy_x = True , algoritmo = 'lloyd' )                                                    [fonte]
```

Fonte: ScikitLearn (2022).

Os parâmetros utilizados diferente do valor *default* estão descritos na Tabela 4. O número de clusters utilizado foi 2 já que nessa base de dados lidamos apenas com a classe “honesto” e “desonesto”. O método de inicialização escolhido foi o ‘*K-means*++’ visando acelerar a convergência, o valor padrão desse parâmetro é o ‘*random*’. O número de inicializações selecionado foi de 300 já que o algoritmo pode convergir com diferentes inicializações. O *random state* escolhido foi 2 após vários testes verificou-se que este valor gerava melhores resultados.

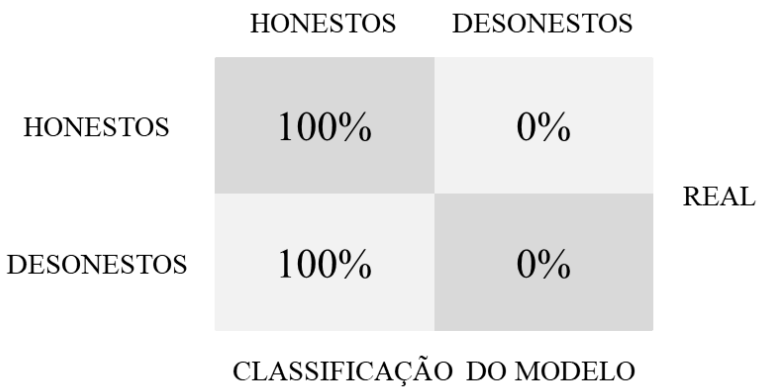
Tabela 4. Parâmetros *K-means*

Nome	Valor utilizado	Função
n_clusters	2	Número de clusters
init	'*K-Means*++'	Método para inicialização
n_init	300	Número de vezes que o algoritmo *K-means* será executado com diferentes sementes de centroides
max_iter	1000	Número máximo de iterações
random_state	2	Determina a geração de números aleatórios para inicialização do centroide

Fonte: Elaborado pela autora (2022).

O primeiro teste foi realizado com a base de dados sem normalização, gerando resultados extremamente insatisfatórios, como mostrado na Figura 40, percebe-se que o algoritmo classificou todos os exemplos como honestos, dando a entender que o modelo não consegue diferenciar as classes, algo bem comum quando os exemplos não seguem um padrão pois o ideal é que todos os atributos contribuam de maneira igual no cálculo da medida de proximidade, visto que um atributo não pode se sobrepor sobre o outro.

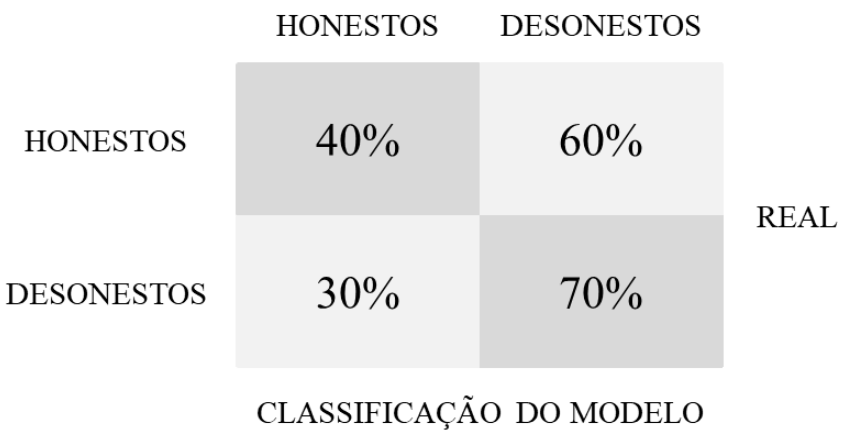
Figura 40. Matriz de confusão *K-means* – Teste 1



Fonte: Elaborado pela autora (2022).

O segundo teste foi realizado com a base de dados já normalizada e apresentou resultado melhor em relação à situação anterior pois, como mostrado na Figura 41, o algoritmo consegue agrupar os exemplos, contudo a classificação ainda é tendenciosa e indica a maioria dos exemplos como desonestos.

Figura 41. Matriz de confusão *K-means* – Teste 2

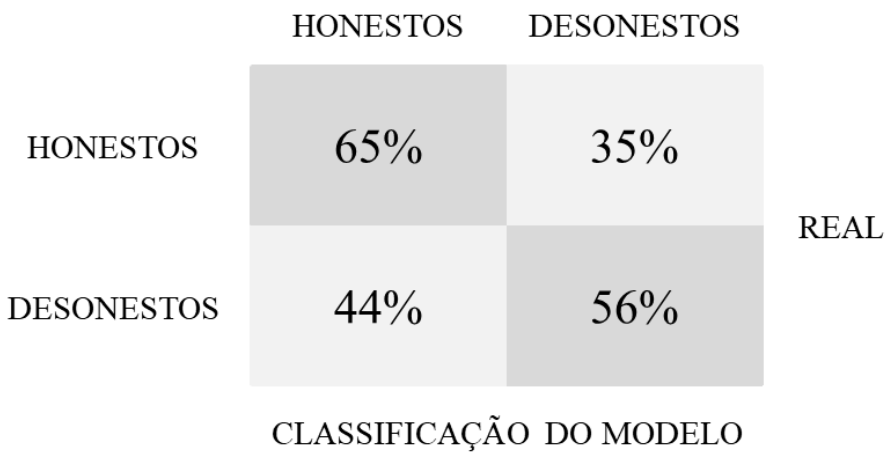


Fonte: Elaborado pela autora (2022).

O terceiro teste, representado na Figura 42, mostra uma melhor acurácia quando usado a padronização ao invés da normalização. Dentre os três testes realizados utilizando o *K-means*, o teste 3 se mostrou mais eficiente, apresentando uma acurácia maior com um tempo de processamento similar aos demais testes. Além disso o teste 3 teve maior equilíbrio na classificação, mostrando a que o algoritmo não

favoreceu nenhuma das classes. A Tabela 5 mostra o resumo dos resultados dos testes realizados com esse algoritmo.

Figura 42. Matriz de confusão *K-means* – Teste 3



Fonte: Elaborado pela autora (2022).

Tabela 5. Resumo *K-means*

Teste	Modelo	Tempo	Acurácia
Teste 1	*K-Means* puro	0,12	50%
Teste 2	*K-Means* normalizado	0,53	55%
Teste 3	*K-Means* padronizado	0,60	61%

Fonte: Elaborado pela autora (2022).

5.4 CLUSTERIZAÇÃO HIERÁRQUICA

O algoritmo de clusterização hierárquica está incluso no subpacote *sklearn.cluster.AgglomerativeClustering*. Na Figura 43 são mostrados os parâmetros de entrada desse algoritmo.

Figura 43. Parâmetros de clusterização hierárquica

class sklearn.cluster.**AgglomerativeClustering**(*n_clusters=2, *, affinity='euclidean', memory=None, connectivity=None, compute_full_tree='auto', linkage='ward', distance_threshold=None, compute_distances=False*) [source]

Fonte: ScikitLearn (2022).

A Tabela 6 abaixo descreve os parâmetros modificados para obter um melhor resultado neste modelo, a escolha dos parâmetros foi feita após testar várias combinações.

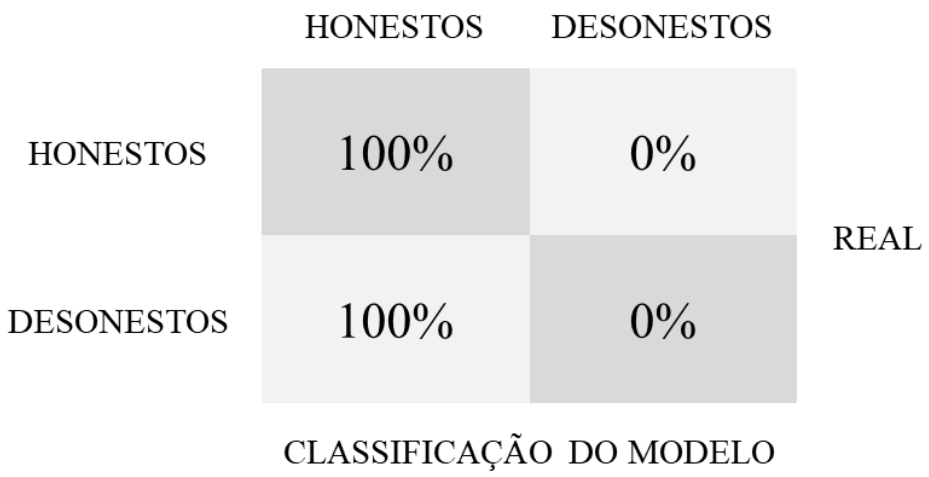
Tabela 6. Parâmetros de clusterização hierárquica

Nome	Valor utilizado	Função
n_clusters	2	O número de clusters a serem encontrados.
affinity	'euclidean'	Métrica usada para calcular a ligação
linkage	'ward'	O critério de ligação determina qual distância usar entre conjuntos de observação. O algoritmo irá mesclar os pares de clusters que minimizam este critério

Fonte: Elaborado pela autora (2022).

O teste 1 foi realizado com a base de dados pura e obteve resultados similares ao *K-means*, como mostrado na Figura 44 o modelo de agrupamento hierárquico não consegue detectar um padrão nos consumidores desonestos e classifica todos os exemplos como clientes honestos.

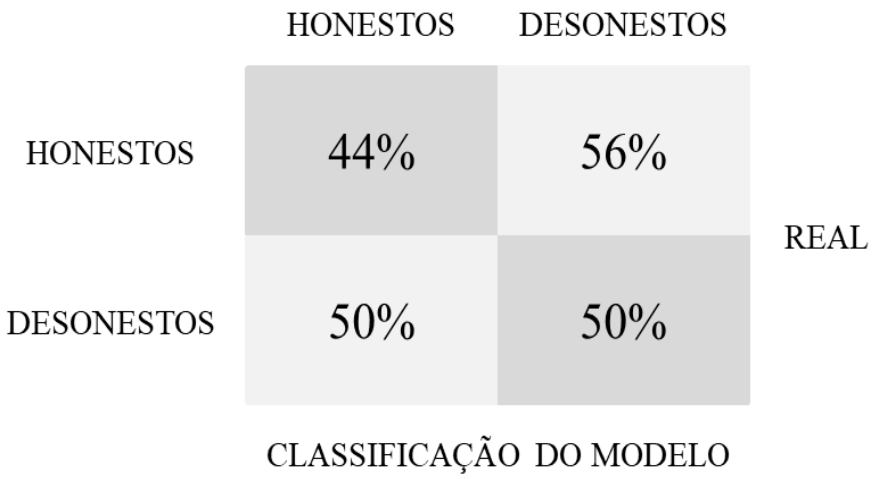
Figura 44. Matriz de confusão clusterização hierárquica – Teste 1



Fonte: Elaborado pela autora (2022).

O teste 2 foi realizado usando a base de dados normalizada e foi obtido um resultado mais balanceado e que o anterior demonstrando que o modelo consegue diferenciar consumidores honestos e desonestos, no entanto a quantidade de acertos é inferior ou menor que os erros (Figura 45).

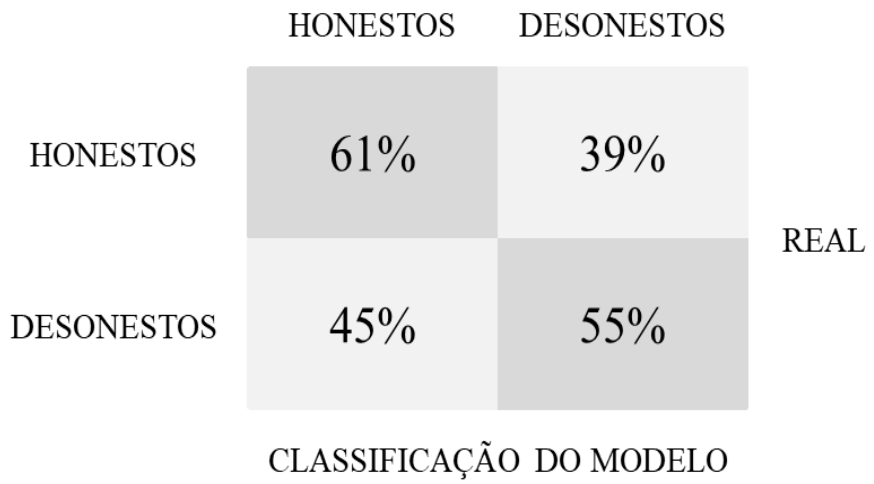
Figura 45. Matriz de confusão clusterização hierárquica – Teste 2



Fonte: Elaborado pela autora (2022).

O teste 3 foi feito com a base de dados padronizada e obteve melhor resultado com o algoritmo de clusterização hierárquica num tempo de 0,34 segundos. A Figura 46 mostra a matriz de confusão desse teste.

Figura 46. Matriz de confusão clusterização hierárquica – Teste 3



Fonte: Elaborado pela autora (2022).

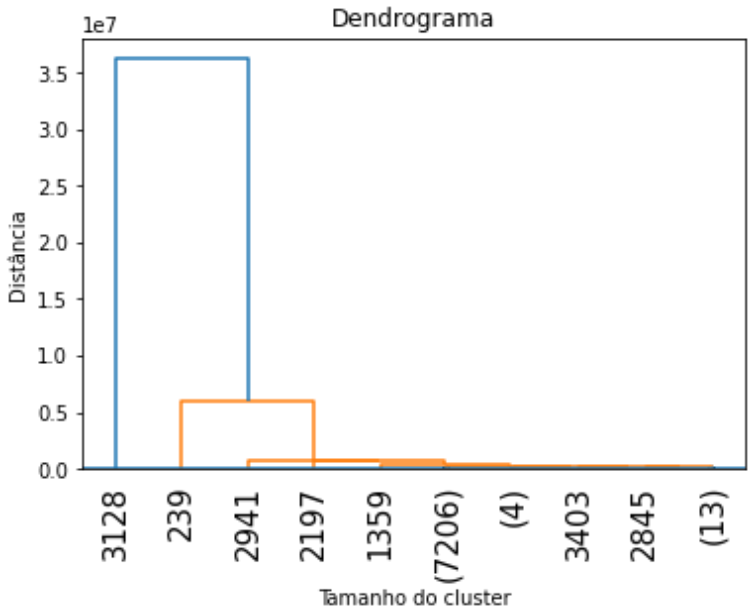
A Tabela 7 resume os testes realizados com o algoritmo de clusterização hierárquica, mostrando a acurácia e o tempo obtidos em cada teste. Nas Figuras 47, 48 e 49 são mostrados os dendrogramas obtidos nestas simulações.

Tabela 7. Resumo de clusterização hierárquica

Teste	Modelo	Tempo	Acurácia
Teste 1	Clusterização hierárquica puro	0,46	50%
Teste 2	Clusterização hierárquica normalizado	0,51	47%
Teste 3	Clusterização hierárquica padronizado	0,34	58%

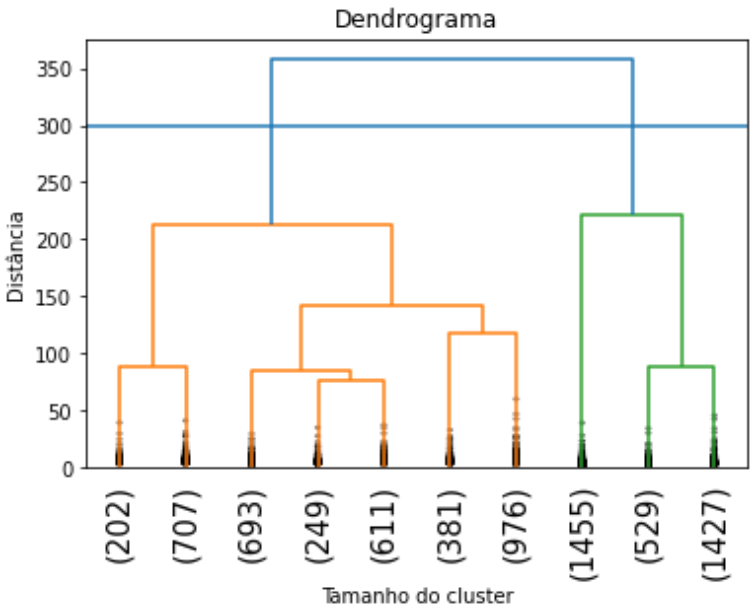
Fonte: Elaborado pela autora (2022).

Figura 47. Dendrograma – Teste 1



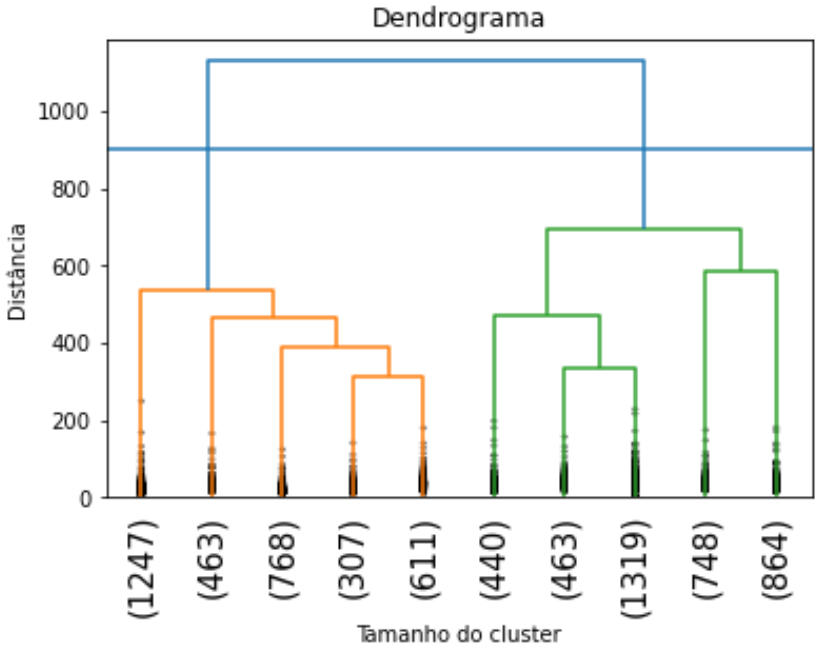
Fonte: Elaborado pela autora (2022).

Figura 48. Dendrograma – Teste 2



Fonte: Elaborado pela autora (2022).

Figura 49. Dendrograma – Teste 3



Fonte: Elaborado pela autora (2022).

CAPITULO 6: CONCLUSÕES

6.1 CONCLUSÕES GERAIS

Foi abordado neste trabalho o problema de perdas não técnicas no setor de distribuição de energia, evidenciando os prejuízos econômicos e sociais causados. Foram apresentados os métodos mais praticados para fraudar sistemas de medição, assim como algumas soluções já adotadas no Brasil e no mundo.

Aqui foi proposto uma solução utilizando *Machine Learning*, visando auxiliar as concessionárias na tomada de decisão ao direcionar equipes de inspeção de PNT em campo utilizando dados armazenados no histórico de clientes. O trabalho analisou três modelos de ML sendo o MLP de aprendizado supervisionado E o *K-means* e a Clusterização Hierárquica modelos de aprendizado não supervisionada.

A metodologia utilizada neste trabalho utilizou-se de uma base de dados real na qual foi necessário realizar um pré-processamento de dados empregando técnicas de ML e aplicando as bibliotecas disponíveis no *software* utilizado, o Anaconda. Os exemplos de clientes contidos nesta base têm características de consumo muito diferentes uns dos outros o que dificulta o aprendizado do modelo.

Os testes foram iniciados utilizando os parâmetros padrões dos algoritmos disponibilizados pelo software e foram alterados individualmente em busca da combinação que proporcionou o melhor resultado.

Dentre os modelos testados, a rede MLP teve o melhor desempenho nos três testes, obtendo uma acurácia de 69% com a base normalizada. Dentre os modelos de aprendizado não supervisionado, o *K-means* teve melhor desempenho, conseguindo uma acurácia de 61% com a rede padronizada. Observou-se que a padronização usada trouxe melhorias consideráveis para os modelos de aprendizado não supervisionado aumentando a acurácia destes em até 19%. Já no caso do modelo de aprendizado supervisionado a normalização e padronização pioraram ou não modificaram relativamente o resultado.

Em relação ao tempo de processamento, o *K-means* e Clusterização Hierárquica tiveram um desempenho consideravelmente melhor em relação ao MLP. O elevado tempo de execução do MLP deixa claro que a implementação precisa ser otimizada. Dessa forma, os resultados obtidos colocam o MLP como melhor classificador para esta base de dados.

Conclui-se que a utilização desse modelo como ferramenta de combate as PNT, associada à outras práticas de combate as perdas não técnicas mencionadas nesse trabalho, podem diminuir consideravelmente os prejuízos das concessionárias de energia.

6.2 TRABALHOS FUTUROS

Os resultados obtidos deixam claro que alguns pontos precisam ser melhorados. Seguem algumas sugestões para trabalhos futuros:

a) Realizar os testes utilizando uma base de dados que contenham outros atributos além do consumo do cliente, como classe de consumo, histórico de dívidas e informações de incidência de fraudes anteriores, de preferência em parceria com uma distribuidora local para validação dos resultados em campo;

b) Testar novos algoritmos de classificação para comparação quanto ao desempenho e tempo de processamento;

c) Utilizar um algoritmo de aprendizado semi supervisionado;

d) Utilização do MLP associado a outros algoritmos;

e) Utilizar outro método para o pré-processamento dos dados de forma que o padrão de consumidores honestos e desonestos fique mais evidente para o algoritmo e também diminuir o tempo de processamento;

f) Associação do modelo ao sistema de medição remota de forma a detectar fraudes em tempo real, permitindo atuação da concessionária de forma mais rápida e eficaz.

REFERÊNCIAS BIBLIOGRÁFICAS

AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA (ANEEL). **Perdas de Energia Elétrica**. Brasília: ANEEL, 2021. Disponível em: http://www2.aneel.gov.br/area.cfm?idArea=801&idPerfil =4. Acesso em: 29 jul. 2022.

AHMED, M.; SERAJ, R.; ISLAM, S. M. S. The k-means algorithm: A comprehensive survey and performance evaluation. **Electronics**, v. 9, n. 8, p. 1-12, 2020. Disponível em: https://doi.org/10.3390/electronics9081295. Acesso em: 19 jul. 2022.

ANDERBERG, M. R. The broad view of cluster analysis. *In*: ANDERBERG, Michael R. **Cluster analysis for applications**. London: Academic Press, 1973, p. 1-9.

AYODELE, T. O. Types of machine learning algorithms. *In*: ZHANG, Y. (ed.) **New Advances in Machine Learning**. London: IntechOpen, 2010, p. 19-48.

BARBOSA, X. C. Breve introdução à história da inteligência artificial. **Jamaxi**, v. 4, n. 2, p. 90-97, 2020. Disponível em: https://periodicos.ufac.br/index.php/jamaxi/article/view/4730. Acesso em: 12 ago. 2022.

BATISTA, G. E. A. P. A. **Pré-processamento de dados em aprendizado de máquina supervisionado**. 2003. Tese (Doutorado em Ciências de Computação e Matemática Computacional) – Universidade de São Paulo, São Carlos, 2003. Disponível em: https://teses.usp.br/teses/disponiveis/55/55134/tde-06102003-160219/publico/TeseDoutorado.pdf. Acesso em: 19 ago. 2022.

BERNARDON, D. P. *et al*. Perdas técnicas e comerciais de energia elétrica em sistemas de distribuição. *In*: CONFERÊNCIA BRASILEIRA DE QUALIDADE DE ENERGIA ELÉTRICA, 7., 2007, Santa Maria. **Anais** [...]. Santa Maria: AGEPOC, 2007.

BHATIA, B.; GULATI, M. P. **Reforming the power sector:** controlling electricity theft and improving revenue. Washington: The World Bank, 2004. Disponível em: https://documents1.worldbank.org/curated/en/175611520414087771/pdf/122142-BRI-272-BHATI-PUBLIC.pdf. Acesso em: 15 ago. 2022.

BOHANI, F. A. *et al*. A comprehensive analysis of supervised learning techniques for electricity theft detection. **Journal of Electrical and Computer Engineering**, v. 2021, p. 1-10, 2021. Disponível em: https://doi.org/10.1155/2021/9136206. Acesso em: 19 ago. 2022.

CHAUHAN, A.; RAJVANSHI, S. Non-technical losses in power system: a review. *In*: INTERNATIONAL CONFERENCE ON POWER, ENERGY AND CONTROL (ICPEC). 1., 2013, Dindigul. **Anais** [...]. Dindigul: IEE, 2013. p. 558-561. Disponível em: https://ieeexplore.ieee.org/document/6527720. Acesso em: 25 ago. 2022.

DANTAS, P. R. P. **Avaliação de perdas de energia elétrica não-técnicas metodologia aplicada no município de Salvador**. 2006. Dissertação (Mestrado em Energia) – Universidade Salvador, Salvador, 2006.

EICHKOFF, H. S. *et al*. Classificação de unidades consumidoras irrigantes de arroz para análise de perdas não técnicas utilizando o método de random forest. *In*: SEMINAR ON POWER ELECTRONICS AND CONTROL, 13., 2021, Santa Maria. **Anais** [...]. Santa Maria: SEPOC, 2021. Disponível em: https://repositorio.ufsm.br/handle/1/21809. Acesso em: 18 maio 2022.

FARIA, G. **Explorando o potencial de algoritmos de aprendizado com reforço em robôs móveis**. 2000. Dissertação (Mestrado em Ciências da Computação e Matemática Computacional) – Universidade de São Paulo, São Carlos, 2000. Disponível em: https://www.teses.usp.br/teses/disponiveis/55/55134/tde-19022020-091603/publico/GedsonFaria_ME.pdf. Acesso em: 12 set. 2022.

FONTANA, E. **Introdução aos algoritmos de aprendizagem supervisionada**. Curitiba: Universidade Federal do Paraná (UFPR), 2020. Disponível em: https://fontana.paginas.ufsc.br/files/2018/03/apostila_ML.pdf. Acesso em: 20 ago. 2022.

FRADKOV, A. L. Early history of machine learning. **IFAC-PapersOnLine**, v. 53, n. 2, p. 1385-1390, 2020. Disponível em: https://doi.org/10.1016/j.ifacol.2020.12.1888. Acesso em: 12 ago. 2022.

GÉRON, A. **Hands-on machine learning with scikit-learn and tensorflow:** concepts., tools, and techniques to build intelligent systems. Sebastopol: O'Reilly Media, 2017.

HAYKIN, S. O. **Neural networks:** a comprehensive foundation. 2. ed. Singapore: Pearson, 1999.

HEBB, D. O. The first stage of perception: growth of the assembly. *In*: HEBB, D. O. **The Organization of Behavior**. New York: John Wiley & Sons, 1949, p. 60-78.

KHAN, Z. A. *et al*. Electricity theft detection using supervised learning techniques on smart meter data. **Sustainability**, v. 12, n. 19, p. 1-25, 2020. Disponível em: https://doi.org/10.3390/su12198023. Acesso em: 27 ago. 2022.

LIKAS, A.; VLASSIS, N.; VERBEEK, J. J. The global k-means clustering algorithm. **Pattern recognition**, v. 36, n. 2, p. 451-461, 2003. Disponível em: https://doi.org/10.1016/S0031-3203(02)00060-2. Acesso em: 16 ago. 2022.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. *In*: BERKELEY SYMPOSIUM ON MATHEMATICAL STATISTICS AND PROBABILITY, 5., 1967, Berkeley. **Anais** [...]. Berkeley: University of California Press, 1967. p. 281-297.

MATOS, Y. C. C. **Detecção de fraudes no consumo de energia elétrica usando árvores de decisão**. 2017. Dissertação (Mestrado em Engenharia Elétrica) – Universidade Federal do Pará, Belém, 2017. Disponível em: https://ppgee.propesp.ufpa.br/ARQUIVOS/dissertacoes/DM%2028_2017%20Yasmin%20Christine%20Correa%20Matos.pdf. Acesso em: 12 ago. 2022.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. **The bulletin of mathematical biophysics**, v. 5, n. 4, p. 115-133, 1943. Disponível em: https://doi.org/10.1007/BF02478259. Acesso em: 29 ago. 2022.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. *In*: REZENDE, S. O. **Sistemas inteligentes:** fundamentos e aplicações. Barueri: Manole, 2005, p. 39-56.

MORAIS, L. C. *et al*. Previsão do preço da energia elétrica no mercado ibérico utilizando uma rede neural artificial multi-layer perceptron com backpropagation. *In*: ENCONTRO DE INICIAÇÃO CIENTÍFICA, 1., 2016, Toledo Prudente. **Anais** [...]. Toledo Prudente: ETIC, 2016. Disponível em: http://intertemas.toledoprudente.edu.br/index.php/ETIC/article/view/5456. Acesso em: 16 ago. 2022.

MURTAGH, F.; CONTRERAS, P. Algorithms for hierarchical clustering: an overview, II. **Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery**, v. 7, n. 2, p. 1-16, 2017. Disponível em: https://doi.org/10.1002/widm.1219. Acesso em: 25 ago. 2022.

NAGAMINE, G. K. **Estudo das perdas não técnicas no sistema elétrico de distribuição e as tecnologias utilizadas para seu combate**. 2011. Trabalho de Conclusão de Curso (Bacharelado em Engenharia Elétrica) – Universidade Tecnológica Federal do Paraná, Curitiba, 2011. Disponível em: https://repositorio.utfpr.edu.br/jspui/bitstream/1/9853/2/CT_COELE_2011_2_02.pdf. Acesso em: 16 set. 2022.

NUNES, D. H. F. **Um breve estudo sobre o algoritmo K-means**. 2016. Dissertação (Mestrado em Matemática) – Faculdade de Ciências e Tecnologia, Universidade de Coimbra, Coimbra, 2016. Disponível em: https://eg.uc.pt/bitstream/10316/48034/1/Tese_DiogoNunes.pdf. Acesso em: 15 jul. 2022.

OLIVEIRA, M. UFPB cria dispositivo com microchip que detecta furto de energia elétrica, o chamado ‘gato’. **Universidade Federal da Paraíba – UFPB**, 03 jun. 2022. Disponível em: https://www.ufpb.br/ufpb/contents/noticias/ufpb-cria-dispositivo-com-microchip-que-detecta-furto-de-energia-eletrica-o-chamado-2018gato2019#wrapper. Acesso em: 07 jun. 2022.

OLIVEIRA, P. B. A. Fraudes mais comuns na medição de energia elétrica. **Punaro Bley**, 2012. Disponível em: https://sites.google.com/site/punarobley/fraudes-mais-comuns-na-medicao-de-energia-eletrica. Acesso em: 16 maio 2022.

PACHECO, A. Introdução ao Scikit-learn – Parte 1: uma visão geral. **Computação Inteligente**, 10 jun. 2021. Disponível em: http://computacaointeligente.com.br/outros/intro-sklearn-part-1/. Acesso em: 27 maio 2022.

PADILHA, V. A.; CARVALHO, A. C. P. L. F. **Mineração de dados em Python**. São Paulo: Instituto de Ciências Matemáticas e de Computação da Universidade de São Paulo, 2017. Disponível em: https://edisciplinas.usp.br/pluginfile.php/3904960/mod_resource/content/3/mineracaodadosbiologicos-parte1.pdf. Acesso em: 18 ago. 2022.

PELIELO, G. *et al*. Smart grids: Redes inteligentes. **Smart grid**, 2016. Disponível em:https://www.gta.ufrj.br/ensino/eel878/redes1-2016-1/16_1/smartgrid/. Acesso em: 19 agosto 2022.

PENG, Y. *et al*. Electricity theft detection in AMI based on clustering and local outlier factor. **IEEE Access**, v. 9, p. 107250-107259, 2021. Disponível em: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9500192. Acesso em: 19 maio 2022.

PENIN, C. A. S. **Combate, prevenção e otimização das perdas comerciais de energia elétrica**. 2008. Tese (Doutorado em Engenharia) – Universidade de São Paulo, São Paulo, 2008. Disponível em: https://www.teses.usp.br/teses/disponiveis/3/3143/tde-14082008-092248/publico/DoctorPenin11.pdf. Acesso em: 19 maio 2022.

PIOTROWSKI, L. J. *et al*. Análise das perdas de energia no sistema elétrico de distribuição brasileiro. SEMINAR ON POWER ELECTRONICS AND CONTROL, 13., 2021, Santa Maria. **Anais** [...]. Santa Maria: SEPOC, 2021. Disponível em: https://repositorio.ufsm.br/handle/1/21753?locale-attribute=es. Acesso em: 18 maio 2022.

REIS FILHO, J. **Sistema inteligente baseado em árvore de decisão, para apoio ao combate às perdas comerciais na distribuição de energia elétrica**. 2006. Dissertação (Mestrado em Engenharias) – Universidade Federal de Uberlândia, Uberlândia, 2006. Disponível em: https://repositorio.ufu.br/bitstream/123456789/14493/1/JRFilhoDISSPRT.pdf. Acesso em: 12 set. 2022.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. **Psychological Review**, v. 65, n. 6, p. 386-408, 1958. Disponível em: https://doi.org/10.1037/h0042519. Acesso em: 19 set. 2022.

SAMUEL, A. L. Some studies in machine learning using the game of checkers. **IBM Journal of Research and Development**, v. 3, n. 3, p. 210-229, 1959. Disponível em: https://ieeexplore.ieee.org/document/5392560. Acesso em: 16 ago. 2022.

SINAGA, K. P.; YANG, M. Unsupervised K-means clustering algorithm. **IEEE Access**, v. 8, p. 80716-80727, 2020. Disponível em: https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=9072123. Acesso em: 12 jun. 2022.

SULLIVAN, K. Power pirates leave Mexico City in the dark. **The Washington Post**, 20 jan. 2002. Disponível em: https://www.washingtonpost.com/archive/politics/2002/01/20/power-pirates-leave-

mexico-city-in-the-dark/1daabd95-dd62-47ef-a2b7-8bd19b9facfa/. Acesso em: 12 set. 2022.

TURING, A. M. Computing machinery and intelligence. *In*: SHIEBER, S. **The Turing Test:** Verbal Behavior as the Hallmark of Intelligence. Cambridge: MIT Press, 2004, p. 68-95.

ZHENG, Z. *et al*. Wide and deep convolutional neural networks for electricity-theft detection to secure smart grids. **IEEE Transactions on Industrial Informatics**, v. 14, n. 4, p. 1606–1615, 2018. Disponível em: https://ieeexplore.ieee.org/document/8233155. Acesso em: 15 jun. 2022.