



Universidade Federal da Bahia
Escola Politécnica
Colegiado do Curso de Eng. Elétrica



Thiago de Almeida Ribeiro

Utilização de Técnicas de Aprendizado de Máquina para Identificação de Fraudes na Distribuição de Energia Elétrica

Orientadora: Profa. Dra. Luciana Martinez

Salvador-BA – Brasil

2022

Thiago de Almeida Ribeiro

Utilização de Técnicas de Aprendizado de Máquina para Identificação de Fraudes na Distribuição de Energia Elétrica

Trabalho apresentado ao Curso de Graduação em Engenharia Elétrica da Universidade Federal da Bahia como parte dos requisitos para a obtenção do grau de Engenheiro(a) Eletricista.

Orientadora: Profa. Dra. Luciana Martinez

Salvador-BA – Brasil

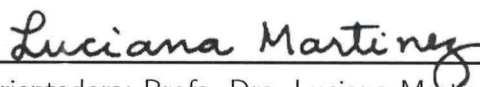
2022

Thiago de Almeida Ribeiro


Utilização de Técnicas de Aprendizado de Máquina para Identificação de Fraudes na Distribuição de Energia Elétrica

Trabalho apresentado ao Curso de Graduação em Engenharia Elétrica da Universidade Federal da Bahia como parte dos requisitos para a obtenção do grau de Engenheiro(a) Eletricista.

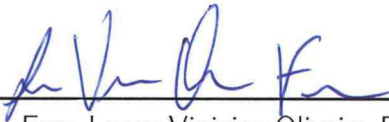
Trabalho aprovado. Salvador-BA – Brasil, 2022:



Orientadora: Profa. Dra. Luciana Martinez


5142 23157127

Prof. Dr. Renato José Pino de Araújo



Eng. Lucas Vinícios Oliveira Filgueiras

Salvador-BA – Brasil
2022

Agradecimentos

À minha família, sobretudo aos meus pais e avós, pelo suporte recebido ao longo de toda a vida.

À minha namorada, Luiza, minha principal companheira e incentivadora.

Aos meus amigos, que tornaram essa caminhada mais leve.

Aos meus colegas Bernardo e Daniel, e ao meu supervisor, Lucas, pelo apoio no desenvolvimento deste Trabalho.

À esta universidade, corpo docente e funcionários por toda dedicação à educação e, em especial, à minha orientadora, Luciana, por todo o auxílio e disponibilidade.

“As questões mais importantes da vida são, na maioria das vezes, apenas problemas de probabilidade”.

(Pierre-Simon Laplace)

Resumo

As perdas no Sistema de Distribuição de Energia Elétrica podem ser definidas como a diferença entre a energia elétrica adquirida pelas distribuidoras e a faturada aos seus consumidores. Essas perdas podem ser técnicas ou não técnicas. O combate às Perdas Não Técnicas é extremamente importante, tanto para as distribuidoras quanto para a sociedade em geral, tornando interessante o estudo e aperfeiçoamento de técnicas que visam encontrar e estancar as fontes dessas perdas. O presente trabalho tem como objetivo desenvolver uma ferramenta de combate às Perdas Não Técnicas, sendo focado em soluções sem o uso de *hardware*, fazendo uso de técnicas de aprendizado de máquina. Nesse sentido, foram selecionados e adequados os dados a serem utilizados, a partir da base de dados de consumidores de uma empresa brasileira. De posse de tais dados, foi feita uma análise exploratória e, com as informações obtidas, foram escolhidas as variáveis a serem utilizadas no treinamento dos modelos, após passarem pela etapa de pré-processamento. Para a construção de tais modelos, foram utilizadas e avaliadas as técnicas de Máquinas de Vetores de Suporte, Redes Neurais Artificiais (mais especificamente Perceptron Multicamadas), Florestas de Caminhos Ótimos, *Random Forest* e *Gradient Boosting*, tendo a última apresentado melhores resultados nos testes teóricos. Tal modelo foi utilizado para selecionar instalações para serem inspecionadas em campo, havendo alguma divergência em relação aos testes teóricos, o que pode ser explicado por fatores como qualidade da base de dados e do serviço da distribuidora, desconsideração da evolução temporal dos dados e sazonalidade, bem como o uso de uma amostra reduzida. Por fim, foram propostas ainda outras metodologias para impulsionar o resultado dos modelos, que podem ser postas em prática em trabalhos futuros.

Palavras-chave: Perdas Não-técnicas. Distribuição. Análise de Dados. Aprendizado de Máquina.

Abstract

Losses in the Electric Energy Distribution System can be defined as the difference between the electric energy purchased by the distributors and that billed to their consumers. These losses can be technical or non-technical. Combating Non-Technical Losses is extremely important, both for distributors and for society in general, making it interesting to study and improve techniques that aim to find and stop the sources of these losses. This work aims to develop a tool to combat Non-Technical Losses, focusing on solutions without the use of hardware, making use of machine learning techniques. In this sense, the data to be used were selected and adapted from the consumer database of a Brazilian company. With such data, an exploratory analysis was carried out and, with the information obtained, the variables to be used in the training of the models were chosen, after going through the pre-processing stage. For the construction of such models, the techniques of Support Vector Machines, Artificial Neural Networks (more specifically Multilayer Perceptron), Optimum-Path Forests, Random Forests and Gradient Boosting were used and evaluated. The last one presented better results in the theoretical tests. This model was used to select installations to be inspected in the field, with some divergence in relation to the theoretical tests, which can be explained by factors such as the quality of the database and the distributor's service, disregard of the temporal evolution of the data and seasonality, as well as the use of a reduced sample. Finally, other methodologies were proposed to boost the results of the models, which can be put into practice in future works.

Keywords: Non-Technical Losses. Distribution. Data Analysis. Machine Learning.

Lista de ilustrações

Fig. 1 – Perdas no Sistema Elétrico	23
Fig. 2 – Árvore de Decisão	35
Fig. 3 – Representação de uma rede neural. (a) Rede neural multicamadas. (b) Modelo de um neurônio.	37
Fig. 4 – Representação da lógica de Máquinas Vetores de Suporte. (a) Máquina de Vetores de Suporte (SVM) linear. (b) SVM não-linear.	38
Fig. 5 – (a) Grafo completo. (b) MST (Árvore Geradora Mínima) do grafo completo. (c) Protótipos escolhidos. (d) Floresta de caminhos ótimos resultante. (e) Uma amostra de teste (triângulo). (f) O caminho ótimo do protótipo.	39
Fig. 6 – Exemplo de Matriz de Confusão	40
Fig. 7 – Processo de Validação Cruzada <i>K-fold</i>	42
Fig. 8 – Fluxograma da metodologia empregada	43
Fig. 9 – Matriz de Correlação de Pearson	49
Fig. 10 – Matriz de Confusão - <i>Gradient Boosting</i>	53
Fig. 11 – Matriz de Confusão - Floresta de Caminhos Ótimos	54
Fig. 12 – Matriz de Confusão - <i>Random Forest</i>	55
Fig. 13 – Matriz de Confusão - <i>Rede Neural Artificial</i>	56
Fig. 14 – <i>K-fold</i> estratificado - Precisão	58
Fig. 15 – <i>K-fold</i> estratificado - Sensibilidade	59
Fig. 16 – <i>K-fold</i> estratificado - <i>F1-Score</i>	59
Fig. 17 – Matriz de Confusão - <i>Gradient Boosting</i> sem INSTPARCEIROFRAUDE	61
Fig. 18 – Matriz de Confusão - <i>Gradient Boosting</i> considerando evolução temporal	63

Lista de tabelas

Tab. 1 – Vantagens e desvantagens das soluções propostas nos diferentes tipos de estudo	28
Tab. 2 – Variáveis de Inspeção	44
Tab. 3 – Variáveis de Leitura e Medição	45
Tab. 4 – Variáveis de Cadastro	45
Tab. 5 – Variáveis de Consumo	46
Tab. 6 – Variáveis de Serviços e Pagamentos	47
Tab. 7 – Parâmetros dos Modelos	52
Tab. 8 – Métricas de Avaliação - <i>Gradient Boosting</i>	53
Tab. 9 – Métricas de Avaliação - Floresta de Caminhos Ótimos	54
Tab. 10 – Métricas de Avaliação - <i>Random Forest</i>	55
Tab. 11 – Métricas de Avaliação - Rede Neural Artificial	56
Tab. 12 – Comparação dos Modelos	57
Tab. 13 – Métricas de Avaliação - <i>Gradient Boosting</i> sem INSTPARCEIROFRAUDE	60
Tab. 14 – Métricas de Avaliação - <i>Gradient Boosting</i> considerando evolução temporal	63

Lista de abreviaturas e siglas

AM	Aprendizado de Máquina
ACP	Análise de Componentes Principais
ANEEL	Agencia Nacional de Energia Elétrica
FN	Falso Negativo
FP	Falso Positivo
IA	Inteligência Artificial
ICMS	Imposto sobre Circulação de Mercadorias e Serviços
ML	<i>Machine Learning</i>
OPF	Floresta de Caminhos Ótimos
OPFs	Florestas de Caminhos Ótimos
PNTs	Pernas Não Técnicas
RNA	Rede Neural Artificial
RNAs	Redes Neurais Artificiais
SVM	Máquina de Vetores de Suporte
SVMs	Máquinas de Vetores de Suporte
TOI	Termo de Ocorrência e Inspeção
VN	Verdadeiro Negativo
VP	Verdadeiro Positivo
VPN	Valor Preditivo Negativo

Lista de símbolos

$cov(x, y)$ Covariância entre X e Y

r Coeficiente de Pearson

Sumário

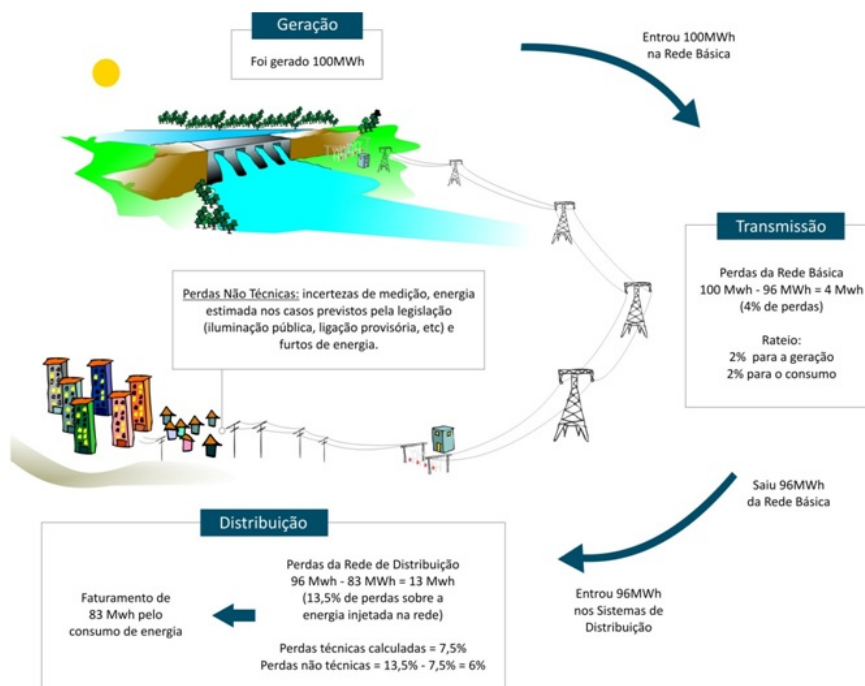
1	INTRODUÇÃO	23
1.1	Objetivo Geral	25
1.2	Objetivos Específicos	25
1.3	Apresentação dos Capítulos	26
2	REVISÃO BIBLIOGRÁFICA	27
2.1	Contextualização	27
2.2	Fundamentação Teórica	29
2.2.1	Concessionária e Consumidor	29
2.2.2	Medição e Faturamento	30
2.2.3	Perdas no Sistema de Distribuição	31
2.2.4	Aprendizado de Máquina	34
2.2.4.1	<i>Random Forest</i>	34
2.2.4.2	<i>Gradient Tree Boosting</i>	36
2.2.4.3	Redes Neurais Artificiais	36
2.2.4.4	Máquinas de Vetores de Suporte	37
2.2.4.5	Florestas de Caminhos Ótimos	38
2.2.5	Avaliação de Desempenho	39
2.2.6	Validação Cruzada	41
3	PROPOSTA DO TRABALHO E ANÁLISE DAS VARIÁVEIS	43
3.1	Preparação dos Dados e Criação das Variáveis	44
3.2	Análise Exploratória	47
3.3	Tratamento dos Dados	50
4	CONCEPÇÃO E TESTE DOS MODELOS	51
4.1	Criação dos modelos	51
4.2	Testes Teóricos	52
4.2.1	Validação dos Resultados e Testes em Campo	57
4.3	Análise e Adequação dos Modelos	60
4.4	Discussão dos Resultados	61
5	CONCLUSÃO	65
	REFERÊNCIAS	67

	APÊNDICE A – CÓDIGOS EM PYTHON	71
A.1	Treinamento	71
A.2	Seleção	85

1 Introdução

O sistema elétrico é composto por atividades de geração, transmissão e distribuição. As perdas de energia se referem à energia elétrica gerada que passa pelas linhas de transmissão e redes da distribuição, mas que não chega a ser comercializada, seja por motivos técnicos ou comerciais (ANEEL, 2021a). As perdas são definidas, portanto, como a diferença entre a energia elétrica adquirida pelas distribuidoras e a faturada aos seus consumidores, e são divididas em técnicas e não técnicas. A Figura 1 demonstra tais perdas.

Fig. 1 – Perdas no Sistema Elétrico



Fonte: ANEEL (2015)

As perdas técnicas são inerentes à atividade de distribuição de energia elétrica, pois parte da energia é dissipada no processo de transporte, transformação de tensão e medição, em decorrência das leis da física. Aplicam-se então modelos específicos utilizando informações simplificadas das redes e dos equipamentos existentes, como, por exemplo, comprimento e bitola dos condutores, potência dos transformadores e energia fornecida às unidades consumidoras. Com base nessas informações, estima-se o percentual de perdas técnicas eficientes relativas à energia injetada na rede.

As Pernas Não Técnicas ([PNTs](#)) ou comerciais decorrem principalmente de furto (ligação clandestina, desvio direto da rede) ou fraude de energia (adulterações no medidor) - popularmente conhecidos como “gatos” - erros de medição e de faturamento. De maneira simplificada, as perdas não técnicas são apuradas pela diferença entre as perdas totais e as perdas técnicas. Seus valores regulatórios, ou seja, que são reconhecidos na tarifa de energia, são calculados pela Agência Nacional de Energia Elétrica ([ANEEL](#)) por uma metodologia de comparação de desempenho das distribuidoras, observando critérios de eficiência e as características socioeconômicas das áreas de concessão. De acordo com a Resolução Normativa 1000/2021 ([ANEEL, 2021b](#)), a diferença de custos entre o valor regulatório e o real é de responsabilidade da concessionária.

As [PNTs](#) reais no Brasil, obtidas pela multiplicação dos montantes pelo preço médio da energia nos processos tarifários, sem considerar tributos, representaram um custo de aproximadamente R\$ 8,6 bilhões em 2020. No entanto, as perdas não técnicas regulatórias, considerou um custo de aproximadamente R\$ 5,6 bilhões ao ano, o que representa aos consumidores cerca de 2,9% do valor da tarifa de energia elétrica, variando por distribuidora ([ANEEL, 2021a](#)). Nota-se, portanto, que tais perdas impactam financeiramente tanto a concessionária quanto o consumidor final.

Além dos impactos financeiros descritos acima, em [Dantas et al. \(2006 apud FÁRIA, 2016\)](#), são destacados alguns outros impactos sociais provocados por tal tipo de perda:

- Insegurança: em geral, as ligações clandestinas são realizadas sem rigor técnico e sem um estudo prévio da rede elétrica local. As consequências disso são acidentes graves, redução do nível de tensão local e aumento das interrupções no fornecimento de energia para clientes normais que compartilham a mesma rede.
- Concorrência desleal: o furto de energia permite reduzir ilicitamente os custos de atividades comerciais ou industriais, gerando uma concorrência desleal em relação às empresas honestas. Essas empresas são, dessa forma, estimuladas a também aderir a essa prática fraudulenta por uma questão de sobrevivência no mercado.
- Desperdício de energia: consumidores fraudadores ou ligados clandestinamente não pagam a energia elétrica que consomem e, por isso, não têm hábitos de racionalização, o que ocasiona grande desperdício de energia.
- Não arrecadação de impostos: a arrecadação de vários impostos é reduzida por fraudes e ligações clandestinas. Dentre esses, destaca-se o Imposto sobre Circulação de Mercadorias e Serviços ([ICMS](#)), que é proporcional à venda de energia elétrica. Tais recursos não arrecadados pelo Estado deixam de ser aplicados em benefício da própria sociedade.

Percebe-se, portanto, que o combate às PNTs é importante tanto para as distribuidoras quanto para a sociedade em geral. Para identificá-las, normalmente é necessária a visita de uma equipe técnica à instalação, o que gera custos. Dessa forma, é interessante otimizar as inspeções, enviando equipes apenas para instalações onde exista algum indício prévio de perda. Devido ao elevado número de consumidores, é vantajoso para as distribuidoras automatizar tal processo de seleção, e a aplicação de técnicas de Aprendizado de Máquina (AM), que auxilia na construção de modelos analíticos, tem sido cada vez mais recorrente.

Apesar de muitos autores já terem tratado sobre a utilização de técnicas de AM na detecção de fraudes – Paulo (2020), Barros (2021), Ramos (2010), entre outros – as abordagens propostas variam consideravelmente a depender dos tipos de dados disponíveis para a construção dos modelos (consumo, cadastro etc.), tornando interessante a utilização de diferentes abordagens de investigação. Comparar o desempenho de diferentes abordagens, buscando identificar quais são mais produtivas, pode ser uma forma de possibilitar a recuperação de uma maior quantidade de energia com um menor número de inspeções em campo realizadas. Além disso, com a pandemia de Covid-19, os perfis de consumo, muito utilizados em boa parte dos trabalhos, sofreram modificações consideráveis, tornando interessante a realização de um estudo atual, de forma a verificar que as metodologias já utilizadas continuam aplicáveis.

1.1 Objetivo Geral

Este Trabalho tem como objetivo desenvolver uma ferramenta computacional para detecção de fraudes de energia em grandes clientes do grupo B de uma concessionária de energia elétrica brasileira, fazendo uso de técnicas de AM. A ferramenta proposta poderá ser utilizada para automatizar o processo de seleção de instalações que receberão inspeções, buscando melhorar a efetividade das ações realizadas em campo.

1.2 Objetivos Específicos

Para que o objetivo geral fosse alcançado, foi necessário:

- Selecionar, a partir do banco de dados de consumidores da distribuidora, as informações mais relevantes para identificar uma fraude.
- Fazer um comparativo entre diferentes técnicas de AM para a detecção de fraudes, analisando tanto taxas de acerto na identificação da fraude como o custo computacional associado à metodologia.

- Analisar a efetividade das inspeções em campo na identificação de fraudes, comparando o retorno obtido com o uso de técnicas de [AM](#) em relação à metodologia atual da distribuidora.

1.3 Apresentação dos Capítulos

O presente Trabalho está dividido em 5 capítulos, conforme descrito abaixo.

No capítulo 1 é apresentada uma introdução sobre o tema abordado, conjuntamente com a justificativa do Trabalho, bem como seus objetivos gerais e específicos.

O capítulo 2 apresenta a revisão de alguns artigos sobre utilização de conceitos de [AM](#) no combate [PNTs](#) no sistema de distribuição. Além disso, é apresentada a fundamentação teórica com os conceitos sobre o funcionamento da distribuição de energia no Brasil, o combate às [PNTs](#), as técnicas de [AM](#) utilizadas e as métricas de avaliação.

O capítulo 3 apresenta a metodologia utilizada no Trabalho e as variáveis disponíveis. Além disso, é realizada uma análise de tais variáveis, fazendo o uso do coeficiente de correlação de Pearson, e descrito o tratamento realizado para que elas pudessem ser utilizadas no treinamento dos modelos.

No capítulo 4 é explicado o processo de criação dos modelos de [AM](#), incluindo técnicas de programação, de treinamento e os parâmetros utilizados. É apresentado ainda o resultado dos testes teóricos e práticos, trazendo as métricas de avaliação. Além disso, são apresentadas as adequações que foram necessárias após alguns testes.

Por fim, o capítulo 5 apresenta e discute as conclusões do Trabalho desenvolvido e potenciais desenvolvimentos futuros.

2 Revisão Bibliográfica

2.1 Contextualização

Em [Viegas et al. \(2017\)](#) foi estruturada uma revisão bibliográfica dos trabalhos na área de detecção de **PNTs**. O artigo divide os estudos analisados, 103 no total, em três categorias: estudos teóricos (6), soluções de *hardware* (25) e soluções sem *hardware* (72), destacando as vantagens e limitações de cada uma delas.

Conforme [Paulo \(2020\)](#), para os estudos teóricos e com uso de *hardware*, os autores citam as limitações de uma baixa precisão de detecção de **PNTs** ou de um capital significativo de despesas para a distribuidora. Entretanto, essas soluções podem se tornar viáveis para identificar e combater perda em locais considerados críticos.

As soluções sem *hardware* são definidas como: “estudo em que o foco principal é a caracterização e descrição de uma solução não *hardware*, normalmente *software* ou algoritmo, que permite a detecção ou estimativa de perdas não técnica”. Elas compreendem a maior parte dos estudos presentes na literatura, por serem soluções mais acessíveis e por aproveitarem e transformarem as informações dos consumidores e medidores em dados para a detecção da probabilidade de um comportamento ilegal. Este Trabalho é focado em soluções sem *hardware*, mas informações de *hardwares* já existentes na rede de distribuição, como alguns Sensores Inteligentes, discutidos em [Oliveira \(2021\)](#), também são utilizadas.

Essa categoria abrange alguns tipos de técnicas, como classificação, estimação, teoria dos jogos, entre outros. A primeira fornece previsões quanto à presença ou não de **PNTs** significativas em uma zona ou instalação específica, enquanto a segunda realiza um estimativa absoluta ou relativa de tais perdas. Para os fins deste Trabalho, considera-se mais adequada a utilização de técnicas de classificação. Já a teoria dos jogos, como o próprio nome indica, diz respeito a estudos que utilizam técnicas de detecção baseadas em tal teoria para modelar consumidores legítimos, fraudadores e as relações entre eles e a concessionária. A Tabela 1, retirada de [Viegas et al. \(2017\)](#), mostra um resumo das vantagens e desvantagens de cada tipo de solução.

Entres as técnicas de classificação mais comuns na literatura, estão presentes: Máquinas de Vetores de Suporte (**SVMs**), Redes Neurais Artificiais (**RNAs**), Árvore de Decisão (tal qual suas derivadas, como *Random Forest* e *Gradient Boosting*) e Florestas de Caminhos Ótimos (**OPFs**), nessa ordem. Neste Trabalho, tais técnicas de classificação são testadas, para fins de comparação de desempenho.

É possível encontrar na literatura vários trabalhos científicos que utilizam as técnicas citadas acima para a detecção de **PNTs** em distribuidoras nacionais, dentre os quais estão

Tab. 1 – Vantagens e desvantagens das soluções propostas nos diferentes tipos de estudo

Tipo	Técnica	Vantagens	Desvantagens
Estudos Teóricos	Análise de variáveis e fatores	Gera informações valiosas para criação de políticas e tomada de decisão	Incapaz de detectar fontes específicas de PNTs
Soluções com Hardware	Hardware de medição	Pode prevenir todos os tipos de PNTs resultantes do medidor	Custos com equipamentos
	Infraestrutura de medição	Pode prevenir todos os tipos de PNTs resultantes da rede elétrica	Custos com equipamentos
	Geração e processamento de sinal	Pode prevenir todos os tipos de PNTs resultantes da rede elétrica	Requer medidores inteligentes
Soluções sem Hardware	Classificação	Baixo custo e uso de recursos disponíveis	A detecção não é garantida e os dados necessários podem não estar disponíveis
	Estimação	Estimativa de estado: baixo custo e alta precisão / Modelagem de perda técnica: baixo custo	Estimativa de estado: requisitos de dados significativos / Modelagem de perda técnica: apenas estima PNTs agregada
	Teoria dos jogos	Estimativas precisas de desempenho	Precisa fazer suposições fortes sobre comportamento fraudulento

Fonte: Adaptado de [Viegas et al. \(2017\)](#)

os trabalhos de [Paulo \(2020\)](#) e [Barros \(2021\)](#) (Árvore de Decisão, *Random Forest*, *Gradient Boosting*, *Rede Neural Artificial (RNA)* e *SVM*) e [Ramos \(2010\)](#) (Floresta de Caminhos Ótimos, *RNA* e *SVM*), adotados como os principais referenciais metodológicos (etapas, variáveis utilizadas etc.) para a construção do presente Trabalho.

É interessante citar ainda o trabalho de [Saeed et al. \(2020\)](#), que pode ser considerado

uma atualização dos trabalhos de [Viegas et al. \(2017\)](#), trazendo algumas informações adicionais. Entre tais informações, tem-se uma lista das métricas mais utilizadas para avaliar os modelos, como: acurácia, precisão, sensibilidade, valor preditivo negativo, F1-score, tempos de classificação e treinamento.

2.2 Fundamentação Teórica

Neste tópico, são apresentados os principais conceitos utilizados na elaboração deste Trabalho, sendo discutidas noções gerais de fornecimento de energia e perdas comerciais, bem como os conceitos necessários da área de [AM](#) e métricas de avaliação dos modelos.

2.2.1 Concessionária e Consumidor

Segundo a Resolução Normativa 1000/2021 ([ANEEL, 2021b](#)), que trata das condições gerais de fornecimento de energia elétrica, a concessionária (ou distribuidora) e o consumidor são definidos, respectivamente, como:

Agente titular de concessão federal para prestar o serviço público de distribuição de energia elétrica.

Pessoa física ou jurídica que solicite o fornecimento do serviço à distribuidora, assumindo as obrigações decorrentes desta prestação à sua unidade consumidora.

Tais consumidores são subdivididos em grupos a depender das condições de fornecimento, sendo o grupo B o foco deste Trabalho. São eles:

- Grupo A: grupamento composto de unidades consumidoras com fornecimento em tensão igual ou superior a 2,3 kV, ou atendidas a partir de sistema subterrâneo de distribuição em tensão secundária, caracterizado pela tarifa binômia e subdividido nos seguintes subgrupos:
 - subgrupo A1 – tensão de fornecimento igual ou superior a 230 kV;
 - subgrupo A2 – tensão de fornecimento de 88 kV a 138 kV;
 - subgrupo A3 – tensão de fornecimento de 69 kV;
 - subgrupo A3a – tensão de fornecimento de 30 kV a 44 kV;
 - subgrupo A4 – tensão de fornecimento de 2,3 kV a 25 kV;
 - subgrupo AS – tensão de fornecimento inferior a 2,3 kV, a partir de sistema subterrâneo de distribuição.

- Grupo B: grupamento composto de unidades consumidoras com fornecimento em tensão inferior a 2,3 kV, caracterizado pela tarifa monômnia e subdividido nos seguintes subgrupos:
 - subgrupo B1 – residencial;
 - subgrupo B2 – rural;
 - subgrupo B3 – demais classes; e
 - subgrupo B4 – iluminação pública.

Além da divisão por grupo, os consumidores possuem ainda uma divisão por classe, a depender da finalidade da instalação:

- Classe Residencial: caracteriza-se pelo fornecimento à unidade consumidora com fim residencial.
- Classe Industrial: caracteriza-se pelo fornecimento à unidade consumidora em que seja desenvolvida atividade industrial, conforme definido na Classificação Nacional de Atividades Econômicas – CNAE, assim como o transporte de matéria-prima, insumo ou produto resultante do seu processamento.
- Classe Comercial: caracteriza-se pelo fornecimento à unidade consumidora em que seja exercida atividade comercial ou de prestação de serviços.
- Classe Rural: caracteriza-se pelo fornecimento à unidade consumidora que desenvolva atividades de agricultura, pecuária ou aquicultura.
- Classe Poder Público: caracteriza-se pelo fornecimento à unidade consumidora solicitado por pessoa jurídica de direito público que assuma as responsabilidades inerentes à condição de consumidor.
- Classe Iluminação Pública: caracteriza-se pelo fornecimento para iluminação de ruas, praças, avenidas, túneis, passagens subterrâneas, jardins, vias, estradas, passarelas etc. Que se caracterizam como atividades sem fins econômicos.
- Classe Serviço Público: caracteriza-se pelo fornecimento exclusivo para motores, máquinas e cargas essenciais à operação de serviços públicos de água, esgoto, saneamento e tração elétrica urbana ou ferroviária, explorados diretamente pelo Poder Público ou mediante concessão ou autorização.

2.2.2 Medição e Faturamento

Ainda segundo a resolução 1000/2021 (ANEEL, 2021b), é de responsabilidade da distribuidora a instalação de equipamentos de medição nas unidades consumidoras, salvo

algumas exceções. Tais medidores podem ser, em alguns casos, lidos remotamente, mas o mais comum, sobretudo no grupo B, são os medidores que necessitam de uma leitura presencial (mais baratos), efetuada por um profissional conhecido como leiturista. Tal leitura deve ser efetuada em intervalos de aproximadamente 30 dias, observados o mínimo de 27 e o máximo de 33 dias, de acordo com o calendário de leitura.

Para as unidades do grupo B, conforme citado anteriormente, aplica-se tarifa monômnia: aquela que é constituída por valor monetário aplicável unicamente ao consumo de energia elétrica ativa. A tarifa pode ser do tipo convencional, sem fazer distinção horária, ou do tipo branca, que distingue o valor tarifário em três horários distintos, os quais variam de acordo com a região.

Além do valor efetivo do consumo, a resolução 1000/2021 ([ANEEL, 2021b](#)) garante a aplicação na fatura das perdas de transformação (2,5% para clientes ligados no secundário) e do custo de disponibilidade do sistema elétrico, que corresponde ao valor mínimo de consumo a ser faturado na unidade consumidora, sendo 30 kWh para ligações monofásicas ou bifásicas a 2 condutores, 50kWh para ligações bifásicas a três condutores e 100 kWh para ligações trifásicas.

Em caso de impedimento de leitura, os valores faturados de consumo devem corresponder à média aritmética dos valores faturados nos 12 ciclos anteriores ao impedimento. A aplicação da média ao faturamento pode acontecer por até três ciclos consecutivos de impedimento, em caso de responsabilidade da distribuidora, sendo que, a partir do quarto ciclo, o faturamento aplicado será o custo de disponibilidade do sistema elétrico à respectiva unidade consumidora. Em caso de responsabilidade do cliente, a aplicação da média por tempo superior aos 3 meses estabelecidos é permitida.

Em caso de faturamento incorreto, o limite de retroatividade do acerto depende da responsabilidade pelo erro. Em caso de responsabilidade da distribuidora e faturamento a menor, a cobrança é limitada aos últimos três ciclos de faturamento imediatamente anteriores ao ciclo vigente. Em caso de faturamento a maior, a devolução ao consumidor tem limite de 36 ciclos de faturamento anteriores. Quando a responsabilidade pelo faturamento incorreto é atribuída ao consumidor, tanto a cobrança quanto a devolução tem período limite de 36 meses anteriores ao ciclo vigente de faturamento.

2.2.3 Perdas no Sistema de Distribuição

Os Procedimentos de Distribuição - PRODIST são documentos elaborados pela ANEEL e normatizam e padronizam as atividades técnicas relacionadas ao funcionamento e desempenho dos sistemas de distribuição de energia elétrica. O Módulo 7 ([ANEEL, 2018](#)) trata especificamente sobre o cálculo de perdas na distribuição, definindo a metodologia, os procedimentos e as formas de apuração das perdas de energia no sistema de distribuição

de energia elétrica.

Alguns dos indicadores utilizados para os cálculos definidos em tal módulo são:

- Energia Injetada - EI: energia ativa medida proveniente de agentes supridores (transmissores, outras distribuidoras e geradores) e da geração própria no sistema de distribuição;
- Energia Fornecida - EF: corresponde à energia ativa entregue medida ou estimada;
- Energia Passante - EP: total de energia ativa que transita em cada segmento do sistema de distribuição;
- Perdas na Distribuição - PD: diferença entre a Energia Injetada e a Energia Fornecida;
- Perda Técnicas - PT: corresponde à energia dissipada no sistema de distribuição devido a fenômenos da física;
- Perda Técnicas do Segmento - PTS: consiste nas perdas técnicas em cada segmento do sistema de distribuição;
- Perda Não Técnicas - PNT: É a diferença entre as Perdas na Distribuição e as Perdas Técnicas.

A partir dos indicadores apresentadas acima, são calculados os indicadores de perdas do sistema de distribuição:

Percentual de Perdas Técnicas – PPT: percentual de perdas técnicas em relação à energia injetada:

$$PPT = \frac{PT}{EI} * 100 \text{ [%]} \quad (2.1)$$

Percentual de Perdas na Distribuição – PPD: percentual de perdas totais em relação à energia injetada:

$$PPD = \left(1 - \frac{EF}{EI}\right) * 100 \text{ [%]} \quad (2.2)$$

Percentual de Perdas Não Técnicas – PPNT: percentual de perdas não técnicas em relação à energia injetada:

$$PPNT = PPD - PPT \text{ [%]} \quad (2.3)$$

As perdas técnicas são inerentes à atividade de distribuição de energia elétrica, pois parte da energia é dissipada no processo de transporte, transformação de tensão e medição em decorrência das leis da física. Essas perdas, portanto, estão associadas às características de carregamento e configuração das redes das concessionárias de distribuição. Os montantes

de perdas técnicas são divididos pela energia injetada, que é a energia elétrica inserida na rede de distribuição para atender aos consumidores, incluindo as perdas. (ANEEL, 2021a)

Segundo Ferreira et al. (2008 apud ORTIZ, 2017) as PNTs podem ser classificadas de acordo com as suas causas da seguinte forma:

1. Causas internas às distribuidoras, como erros de faturamento e leitura do medidor, problemas no cadastramento dos seus clientes e equipamentos obsoletos ou defeituosos.
2. Causas que são externas às distribuidoras e promovidas por parte dos consumidores, como a adulteração de equipamentos de medição e ligações clandestinas.
3. Causas que são externas às distribuidoras e relacionadas com questões tanto sociais quanto governamentais, como, por exemplo, o processo de favelização. A diferença fundamental entre este item e o anterior é que, neste caso, quem executa a fraude não é cliente da concessionária, não há cadastro e tampouco medidor instalado. Além disso, muitas das vezes, o fraudador não possui condições de pagar a fatura.

Nesse contexto, são denominadas, respectivamente, como: defeito, fraude e clandestinos. O foco deste Trabalho são as fraudes, mas alguns defeitos também poderão ser detectados com o auxílio da ferramenta desenvolvida.

Alguns dos procedimentos irregulares mais comuns que caracterizam uma fraude são:

- Ligação direta ou desvio na medição.
- Ligação invertida.
- Auto-religação de unidade cortada em campo.
- Uso de super imã.
- Ponte nos blocos de terminais.
- Neutro isolado.
- Medidor inclinado/deitado.
- Medidor avariado.

Identificada uma fraude através de uma inspeção, a distribuidora deve levantar o conjunto de evidências através de perícia técnica, avaliação do histórico de consumo e

grandezas elétricas, medição fiscalizadora e/ou recursos visuais. Em caso de violação do medidor, é necessário também elaborar um relatório de avaliação técnica. O Termo de Ocorrência e Inspeção (TOI) deve ser emitido e uma cópia deve ser entregue ao consumidor ou a quem acompanhar a inspeção. Comprovada a irregularidade, para apurar as diferenças não faturadas, a concessionária deve utilizar os critérios da Resolução Normativa N° 1000 (ANEEL, 2021b) para cobrar o consumo devido do cliente. (PAULO, 2020)

2.2.4 Aprendizado de Máquina

Os termos AM ou *Machine Learning* (ML) são utilizados para se referir a um campo da Inteligência Artificial (IA) dedicado ao desenvolvimento de algoritmos para análise automática de dados. Os algoritmos se baseiam na ideia de que sistemas podem aprender com dados, identificar padrões e tomar decisões com o mínimo de intervenção humana. Em outras palavras, a partir do aprendizado com exemplos históricos, um modelo computacional pode executar uma tarefa sem que haja instruções explicitamente programadas para isso (SIMON, 2013 apud BARROS, 2021).

Uma das principais aplicações do AM é no problema de classificação, cujo objetivo é atribuir um rótulo a um indivíduo no conjunto de dados, de modo a classificá-lo em uma das possíveis classes existentes. Nota-se que o problema de detecção de fraudes é um caso típico de classificação. Entre os principais algoritmos de classificação, os quais são utilizados ao longo deste Trabalho, tem-se: SVM, RNA, entre outros.

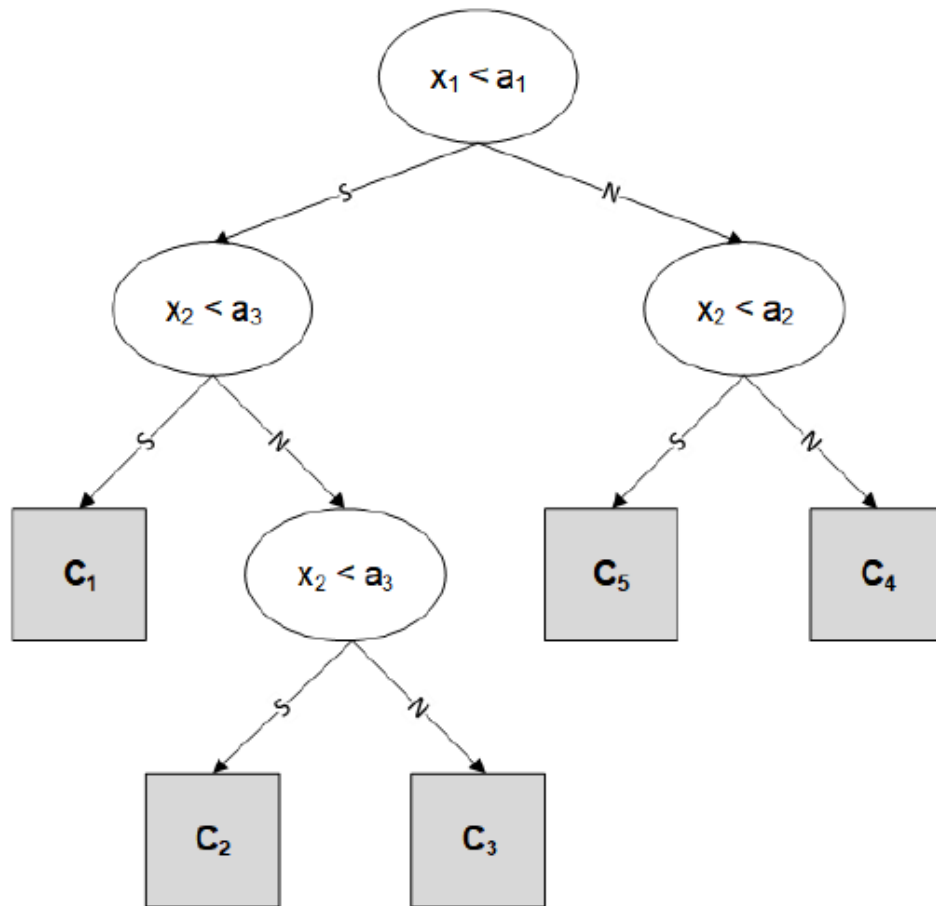
Apesar deste não ser o foco principal do Trabalho, o entendimento de tais algoritmos pode ser interessantes para uma melhor definição das variáveis a serem utilizadas. Dessa forma, serão trazidas a seguir descrições simplificadas de tais algoritmos, baseadas em Paulo (2020) e Ramos (2010).

2.2.4.1 *Random Forest*

Um dos algoritmos mais conhecidos e simples de ser interpretado, existentes na área de aprendizado de máquina, são as Árvores de Decisão. Uma árvore de decisão clássica é um grafo acíclico direcionado, conforme a Figura 2, sendo comumente gerada com base em cálculos de entropia e ganho de informação, variáveis que dizem respeito à falta de uniformidade e desorganização dos dados.

Por outro lado, esses modelos não apresentam bons resultados de precisão quando comparados a outras técnicas de AM. Isso ocorre pois são instáveis, ou seja, uma pequena alteração nos dados pode gerar uma grande alteração na árvore final já que, a cada nó, o critério para dividir a árvore tem como base o melhor atributo, sendo que dois ou mais atributos podem ser classificados similarmente e pequenas variações podem levar a árvores completamente diferentes (FACELI et al., 2011). No entanto, a performance pode ser

Fig. 2 – Árvore de Decisão



Fonte: Paulo (2020)

aprimorada substancialmente ao se agregar várias árvores de decisão ao mesmo modelo, como ocorre para as florestas aleatórias, ou como são mais conhecidas, *Random Forests*, propostas por Breiman (2001).

As *Random Forests* estão dentro da área de *Ensemble Learning* (Aprendizagem em Conjunto), que é baseada, em analogia ao mundo real, na ideia de “consultar diferentes profissionais antes de tomar uma decisão”. Elas consistem em combinações de árvores de decisão treinadas a partir do método de *bagging*, no qual cada árvore possui um conjunto determinado de atributos escolhidos aleatoriamente para o cálculo métrica de pureza/impureza (como a entropia), o que as mantém não correlacionadas. Tratando-se de um algoritmo de classificação, são utilizados os “votos da maioria” para dar a resposta final.

2.2.4.2 Gradient Tree Boosting

O *Gradient Boosting*, assim como o *bagging*, se enquadra na área de *Ensemble Learning*. Construídos os modelos, o *Gradient Boosting* generaliza-os a partir da otimização de uma função de perda diferenciável arbitrária. Quando baseado em árvores de decisão, o que é mais comum, é conhecido como *Gradient Tree Boosting*.

O *Gradient Tree Boosting* se diferencia do *bagging* pois agora o conjunto de treinamento é gerado de forma que as observações são ponderadas e algumas farão parte do conjunto com mais frequência, enquanto que anteriormente tal processo era equiprovável. Cada classificador do *boosting* é treinado considerando o sucesso do classificador anterior. A cada etapa de treinamento, os pesos das ponderações são redistribuídos de modo a enfatizar os casos mais difíceis de se classificar, para que os próximos classificadores se concentrem nele.

Nesse caso, a resposta final também será obtida considerando uma ponderação. Um modelo com um bom resultado de classificação deverá receber um peso maior, e a saída será determinada pela média ponderada das estimativas de cada modelo.

2.2.4.3 Redes Neurais Artificiais

A Rede Neural Artificial é uma das técnicas mais conhecidas de aprendizado de máquina, tendo sido projetada tomando como inspiração a maneira como o cérebro humano adquire conhecimento e a sua capacidade de aprendizado. Suas raízes estão nos trabalhos do neuroanatomista e psiquiatra Warren McCulloch, do Instituto Tecnológico de Massachusetts, e do matemático Walter Pitts, da Universidade de Illinois (MCCULLOCH; PITTS, 1943). Tais ideias foram evoluindo até o ano de 1958, no qual Frank Rosenblatt criou o *Perceptron* (ROSENBLATT, 1958), considerado o tipo mais simples de RNA *feedforward*, na qual a informação flui sem realimentação de valores de saídas.

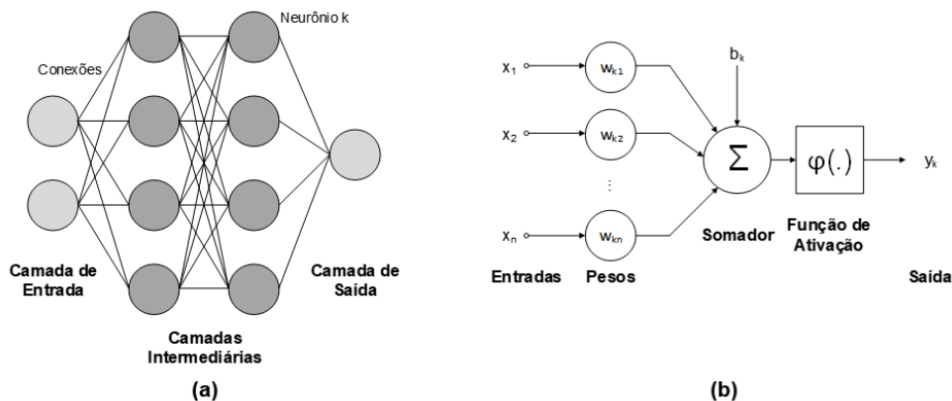
Uma RNA é composta por diversas unidades de processamento chamadas de neurônios, em analogia aos sistemas biológicos. Tais neurônios computam funções matemáticas e são, na maioria dos casos, dispostos em camadas, as quais são ligadas pelos axônios. Cada um deles está associado a um peso, de onde vem o conhecimento das redes neurais. Dessa forma, o processo de aprendizado é dado pelo ajuste de tais pesos, o que pode ser feito através de diferentes algoritmos. Entre os principais, tem-se o Gradiente Descendente, o RMSProp e o Adam, que buscam minimizar uma função objetivo, normalmente uma função de erro.

Em geral, o treinamento da RNA é realizado em conjunto com a técnica de retropropagação (*backpropagation*) para computar o gradiente da função objetivo. O termo *backpropagation* refere-se apenas ao método para calcular o gradiente, enquanto o algoritmo de otimização é usado para realizar o aprendizado, ou seja, atualizar os pesos, usando esse

gradiente (GOODFELLOW et al., 2016).

Na Figura 3 (a) é possível observar uma rede *feedforward* multicamadas composta por: camada de entrada, em que os padrões são apresentados ao modelo; camadas intermediárias, onde é feita a maior parte do processamento; camada de saída, em que são entregues as saídas do sistema. Como pode ser observado na Figura 3 (b), as entradas de um neurônio recebem os valores e estes são ponderados e combinados através de um somador para então serem convertidos na saída após aplicar uma função de ativação. Essa função é normalmente do tipo sigmoidal, ao menos para os neurônios da camada de saída, já que permite que o resultado final seja interpretado de maneira probabilística (BISHOP, 1995). Para as camadas intermediárias, a função linear retificada ou *Rectified Linear Unit* (ReLU) têm sido amplamente utilizada graças a sua simplicidade e eficácia (RAMACHANDRAN; ZOPH; LE, 2017).

Fig. 3 – Representação de uma rede neural. (a) Rede neural multicamadas. (b) Modelo de um neurônio.



Fonte: Paulo (2020)

2.2.4.4 Máquinas de Vetores de Suporte

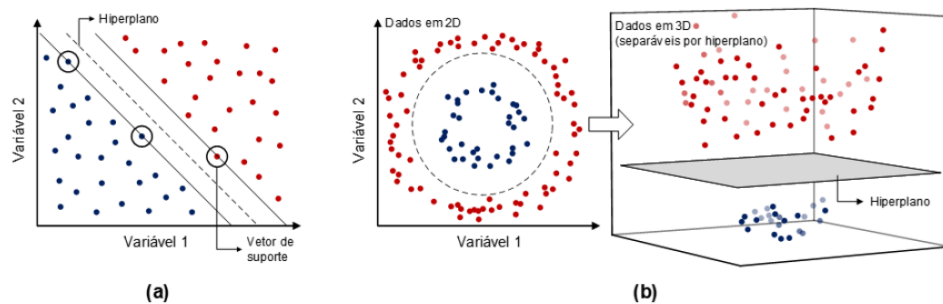
A Máquina Vetores de Suporte é um algoritmo de AM, por vários anos, foi considerado como o mais eficiente, e ainda supera a maior parte dos algoritmos mais simples, como as árvores de decisão. Atualmente tem eficiência comparável às RNAs. Tal algoritmo é utilizado sobretudo para tarefas de classificação e regressão, tendo como vantagens o fato de não sofrer tanta influência de ruídos ou *outliers*, aprender conceitos não presentes nos dados originais e ser normalmente mais simples de ser utilizado que as RNAs (menos parâmetros para se configurar).

Nesse algoritmo, cada item é tido como um ponto em um espaço n -dimensional, onde n é o número de recursos disponíveis, e sendo o valor de cada recurso correspondente a uma coordenada. Dessa forma, a classificação é realizada encontrando-se o hiperplano

que melhor diferencia as classes, ou seja, de modo que a margem entre as classes seja maximizada. Os pontos que tocam as margens são chamados de vetores de suporte. Na Figura 4 (a) é possível observar um exemplo para uma classificação binária em duas dimensões, com o hiperplano sendo representado por uma reta.

SVMs lineares são eficazes em conjuntos de dados aproximadamente lineares mesmo com a presença de ruídos, como citado anteriormente. Entretanto, quando não é possível separar as classes diretamente, **SVMs** não lineares mapeiam o conjunto de dados de seu espaço original para um novo de maior dimensão, conhecido por espaço de características, utilizando-se de uma técnica conhecida como *Kernel Trick*. A escolha apropriada desse novo espaço transforma o problema não linear em um problema linear, fazendo com que o conjunto de treinamento mapeado possa ser separado por um hiperplano. Esse caso é ilustrado pela Figura 4 (b).

Fig. 4 – Representação da lógica de Máquinas Vetores de Suporte. (a) **SVM** linear. (b) **SVM** não-linear.



Fonte: Paulo (2020)

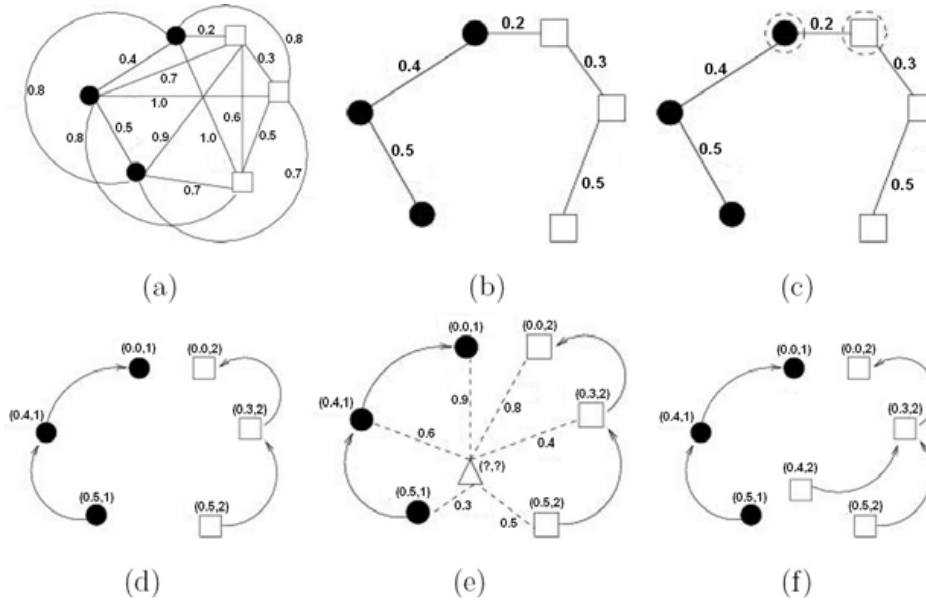
2.2.4.5 Florestas de Caminhos Ótimos

O algoritmo de Floresta de Caminhos Ótimos (**OPF**) foi introduzido em Papa et al. (2007), e tem sido cada vez mais utilizado para realização da tarefa de classificação. Isso se dá pelo fato do algoritmo dispor de alguns benefícios, como ser livre de parâmetros, possuir tratamento nativo de problemas multiclases e não fazer alusão sobre a forma e/ou separabilidade das classes.

A técnica modela as amostras como sendo os nós de um grafo completo. Os elementos mais representativos de cada classe do conjunto de treinamento, isto é, os protótipos, são escolhidos como sendo os elementos pertencentes às regiões de fronteiras entre as classes. Os protótipos participam de um processo de competição disputando as outras amostras oferecendo-lhes caminhos de menor custo e seus respectivos rótulos (ou classes). Ao final deste processo, obtém-se um conjunto de treinamento particionado em

árvores de caminhos ótimos, sendo que a união dessas árvores remete a uma floresta de caminhos ótimos (RAMOS, 2010). Tal processo é ilustrado na Figura 5.

Fig. 5 – (a) Grafo completo. (b) MST (Árvore Geradora Mínima) do grafo completo. (c) Protótipos escolhidos. (d) Floresta de caminhos ótimos resultante. (e) Uma amostra de teste (triângulo). (f) O caminho ótimo do protótipo.



Fonte: Ramos (2010)

2.2.5 Avaliação de Desempenho

Para avaliar os resultados obtidos a partir de cada modelo, é necessário definir algumas métricas. No caso de problemas de classificação, é muito comum utilizar conceitos relacionados à Matriz de Confusão.

A Matriz de Confusão é uma ferramenta padrão de análise de desempenho para classificadores. Tal ferramenta consiste de uma representação, em forma de matriz, indicando a frequência de classificação para cada classe do modelo e suas possíveis combinações. Cada linha da matriz representa os indivíduos em uma classe prevista, enquanto cada coluna representa os indivíduos em uma classe real (ou vice-versa) (KELLEHER; NAMEE; D'ARCY, 2020). Em Barros (2021), é possível encontrar um exemplo de Matriz de Confusão, bem como a definição de alguns parâmetros interessantes, como segue.

Tal matriz divide os resultados em 4 categorias:

- **Verdadeiro Positivo (VP):** a classe predita e real fazem parte da classe positiva;
- **Falso Positivo (FP):** a classe predita retornou positivo mas a real era negativa;

Fig. 6 – Exemplo de Matriz de Confusão

		Classe Preditiva		
		negativo	positivo	
Classe Real	negativo	4 (VN)	1 (FP)	80% (especificidade)
	positivo	2 (FN)	3 (VP)	60% (sensibilidade)
		67% (valor preditivo negativo)	75% (precisão)	70% (acurácia)

Fonte: Adaptado de [Barros \(2021\)](#)

- **Verdadeiro Negativo (VN):** os valores preditos e reais fazem parte da categoria negativa;
- **Falso Negativo (FN):** o valor predito resultou na classe negativa mas o real era da classe positiva.

A partir das quantidades de **VP**, **FP**, **VN** e **FN**, é possível determinar várias métricas estatísticas associadas ao desempenho do classificador, sendo que as cinco principais estão destacadas na Figura 6 e são descritas a seguir:

- **Sensibilidade:** indica o percentual de indivíduos da classe positiva que foram identificados corretamente pelo classificador. A sensibilidade é definida conforme a Equação (2.4). No exemplo da Figura 6, dentre os cinco positivos existentes no grupo, apenas três foram identificadas pelo classificador (sensibilidade = 60%);

$$Sensibilidade = \frac{VP}{VP + FN} \quad (2.4)$$

- **Especificidade:** é o equivalente da sensibilidade para a classe dos negativos e é definida conforme a Equação (2.5). No exemplo da Figura 6, dentre os cinco negativos existentes no grupo, quatro foram identificados pelo classificador (especificidade = 80%);

$$Especificidade = \frac{VN}{VN + FP} \quad (2.5)$$

- **Precisão:** indica o nível de acerto da classificação positiva e é definida conforme a Equação (2.6). No exemplo da Figura 6, dentre os quatro indivíduos classificados como positivos, apenas três realmente eram (precisão = 75%);

$$Precisão = \frac{VP}{VP + FP} \quad (2.6)$$

- **Valor Preditivo Negativo (VPN):** é o equivalente da precisão para a classe dos negativos e é definido conforme a Equação (2.7). No exemplo da Figura 6, dentre os seis indivíduos classificados como negativos, apenas quatro realmente eram (valor preditivo negativo = 67%);

$$VPN = \frac{VN}{VN + FN} \quad (2.7)$$

- **Acurácia:** indica o nível de acerto geral do classificador, incluindo as classes positivas e negativas. A acurácia é definida conforme a Equação (2.8). No exemplo da Figura 6, do total de 10 indivíduos, sete foram classificados corretamente em suas respectivas classes (acurácia = 70%).

$$Acurácia = \frac{VP + VN}{VP + FP + VN + FN} \quad (2.8)$$

Dentre as métricas destacadas, as mais importantes para um classificador binário, como é o caso dos construídos neste Trabalho, são a sensibilidade e a precisão. Tais métricas guardam uma relação de compromisso, de forma que é simples obter uma alta precisão com baixa sensibilidade ou uma alta sensibilidade com uma baixa precisão. Assim, o desafio para um bom classificador é obter a combinação de alta sensibilidade com alta precisão, o que pode ser mensurado a partir do *F1-score*, que consiste na média harmônica entre a sensibilidade e a precisão, conforme expresso na Equação (2.9).

$$F1\text{-score} = \frac{2}{\frac{1}{sensibilidade} + \frac{1}{precisão}} = 2 \cdot \frac{sensibilidade \times precisão}{sensibilidade + precisão} \quad (2.9)$$

O valor de *F1-score* varia entre zero e um (ou 0% e 100%). Tais valores ocorrem, respectivamente, nos casos de maior desequilíbrio e equilíbrio, ou seja, quando sensibilidade e precisão são 0% e 100% ou 100% e 100%, não importando a ordem.

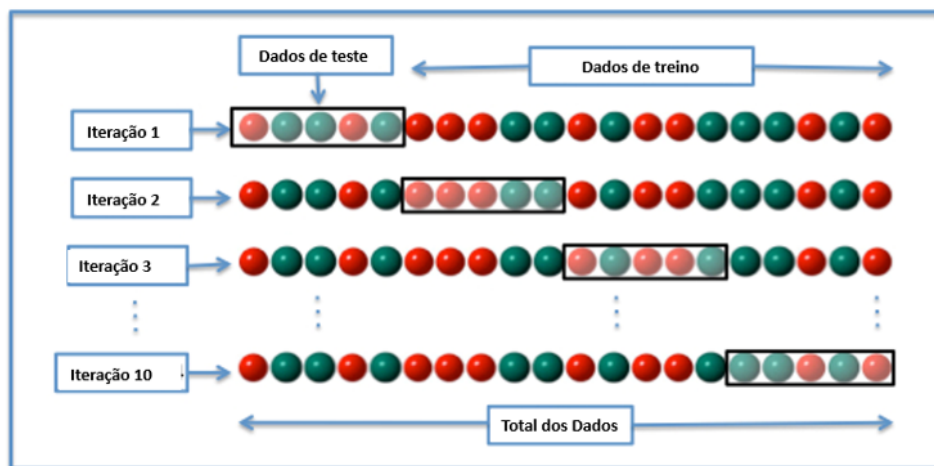
2.2.6 Validação Cruzada

De acordo com Barros (2021), para garantir que as métricas discutidas anteriormente sejam representativas da realidade, é necessário que sejam obtidas fazendo-se uso de um processo de validação robusto, conhecido como validação cruzada.

Uma forma simples de validar um modelo envolve separar uma parte dos dados de treinamento e usá-los para obter previsões do modelo treinado sobre o restante dos dados. Este é o tipo mais simples de técnica de validação cruzada, também conhecida como método *holdout*. O problema com este método é que não é garantido que o *set* de validação separado seja representativo do total da base de dados.

O método mais popular de validação cruzada é o *K-fold*, o qual consiste em particionar o conjunto de dados disponível em *K* subconjuntos e, posteriormente, realizar várias iterações de treinamento e validação do modelo preditivo com os diferentes subconjuntos (BROWNLEE, 2018). O funcionamento do processo de validação cruzada *K-fold* pode ser visto na Figura 7.

Fig. 7 – Processo de Validação Cruzada *K-fold*



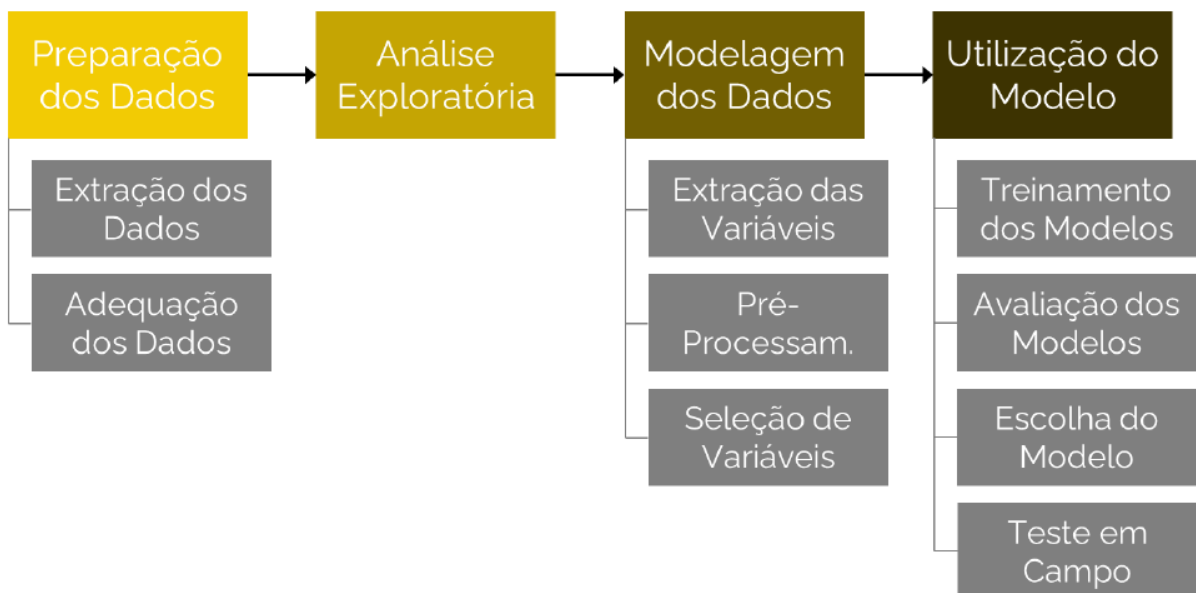
Fonte: Adaptado de Gufosowa (2019)

Além disso, a amostragem utilizada pode ser aleatória ou estratificada. A validação cruzada estratificada é um tipo especial de validação cruzada que cria partições com a mesma distribuição de probabilidade do conjunto de dados maior. Por exemplo, em um conjunto de dados em que 80% dos valores-alvo são “Não” e 20% são “Sim”, cada partição teria aproximadamente 80% de respostas “Não” e 20% de respostas “Sim”. A validação cruzada estratificada costuma ser recomendada quando a variável-alvo está desproporcional.

3 Proposta do Trabalho e Análise das Variáveis

Neste Trabalho, a proposta de identificação de fraudes na distribuição de energia elétrica seguiu uma metodologia semelhante à adotada em [Paulo \(2020\)](#), originalmente proposta por [Pyle \(1999\)](#), guardadas as devidas diferenças em cada etapa, devido às particularidades de cada caso de estudo. Seu fluxograma pode ser visto na Figura 8.

Fig. 8 – Fluxograma da metodologia empregada



Fonte: [Paulo \(2020\)](#)

Inicialmente, foram selecionados e adequados os dados a serem utilizados, a partir da base de dados de consumidores da empresa. De posse de tais dados, foi feita uma análise exploratória, que consiste em verificar o espaço dos dados a fim de identificar e avaliar abordagens apropriadas, definir soluções e estratégias de implementação e produzir resultados mensuráveis ([PYLE, 1999](#)).

Com as informações obtidas, foram escolhidas as variáveis a serem utilizadas no treinamento dos modelos, após passarem pela etapa de pré-processamento. Por fim, tais modelos foram treinados e avaliados. Aqueles que apresentaram melhores resultados nos testes teóricos foram ainda ser utilizados para a realização de testes em campo.

3.1 Preparação dos Dados e Criação das Variáveis

Os sistemas comerciais da distribuidora em questão possuem informações sobre os clientes relacionadas tanto ao serviço prestado, quanto à unidade consumidora. As informações utilizadas neste Trabalho foram classificadas em cinco categorias: inspeções, cadastro, consumo, leitura e medição, serviços e pagamentos. A escolha das informações foi baseada em trabalhos similares e na indicação de colaboradores da própria distribuidora.

Ressalta-se que, para a utilização de técnicas de AM, é necessário a existência de uma variável *target*, que no caso é a existência ou não de fraude. Como isso só pode ser constatado a partir de uma inspeção, e cada unidade consumidora foi inspecionada em uma data diferente, é necessário fazer uma adequação da base considerando uma data de inspeção como a data referência (no caso foi escolhida a inspeção mais recente). Assim, só as informações dessa data, ou de datas anteriores, foram consideradas.

Para considerar a mesma quantidade de pontos para todos os exemplos do banco, assumiu-se uma janela de 36 meses anteriores à data de referência. Essa janela foi aplicada às bases de dados de consumo, irregularidades e serviços. Além disso, foram atualizados os dados de cadastro com as informações válidas naquela data específica. Como o banco de dados da empresa fornece informações de consumo dos últimos cinco anos, foram consideradas apenas unidades inspecionadas nos últimos dois anos.

As variáveis consideradas inicialmente, construídas a partir das informações presentes no banco de dados, podem ser vistas nas Tabelas 2, 3, 4, 5 e 6.

Tab. 2 – Variáveis de Inspeção

Atributo	Descrição
QTD_FRAUDE	Quantidade de fraudes já encontradas na instalação
FAMILIAINSPANT	Família da última inspeção (indica se houve fraude ou não)
INSTPARCEIROFRAUDE	Flag que indica se já houve fraude em alguma instalação do cliente

Tab. 3 – Variáveis de Leitura e Medição

Atributo	Descrição
OCLE_ATUAL	Ocorrência de Leitura no mês da fraude
TENSAOMED	Tensão medida
PERDA_NAO_TECNICA_PRCT	Perda Não-Técnica percentual por unidade consumidora de uma área (obtida a partir de Sensores Inteligentes) ou do alimentador

Tab. 4 – Variáveis de Cadastro

Atributo	Descrição
LATITUDE	Latitude da instalação
LONGITUDE	Longitude da instalação
MOTIVO	Razão pela qual essa instalação foi incluída na base de dados
TIP_INSTALACAO	Tipo de instalação
FAB_MEDIDOR	Fabricante do medidor
MODELO_MEDIDOR	Modelo do medidor
IDADE_ATIVA_MEDIDOR	Idade ativa do medidor
IDADE_INSTALACAO	Idade da instalação
CLASSE_CONTACONTRATO	Classe da conta contrato
SUBCLASSE_CONTACONTRATO	Subclasse da conta contrato
TENSAOFORNEC	Tensão de fornecimento
MODLIGA	Modo de ligação (direta ou indireta)
TIPO_LOCAL	Tipo de local (urbano ou rural)
BAIXARENDA	Flag de clientes de baixa renda

Tab. 5 – Variáveis de Consumo

Atributo	Descrição
CONS_MES_FRAUDE	Consumo do mês anterior à fraude
QUEDA	Quantidade de meses desde a última queda
TX_QUEDA_POS	Taxa de queda pós-queda
TX_QUEDA_POS	Taxa de queda recente
QTD_QUEDA_12M	Quantidade de quedas últimos 12 meses
QTD_QUEDA_24M	Quantidade de quedas últimos 24 meses
QTD_QUEDA_36M	Quantidade de quedas últimos 36 meses
QTD_QUEDA_48M	Quantidade de quedas últimos 48 meses
QTD_QUEDA_60M	Quantidade de quedas últimos 60 meses
COMP_MED12M_MEDVZ12M	Comparação da média de consumo 12 meses com a média de consumo 12 meses do vizinho
COMP_MED24M_MEDVZ24M	Comparação da média de consumo 24 meses com a média de consumo 24 meses do vizinho
COMP_MED36M_MEDVZ36M	Comparação da média de consumo 36 meses com a média de consumo 36 meses do vizinho
COMP_CV12M_CVVZ12M	Comparação do coeficiente de variação do consumo 12 meses com o coeficiente de variação do consumo 12 meses do vizinho
COMP_CV24M_CVVZ24M	Comparação do coeficiente de variação do consumo 24 meses com o coeficiente de variação do consumo 24 meses do vizinho
COMP_CV36M_CVVZ36M	Comparação do coeficiente de variação do consumo 36 meses com o coeficiente de variação do consumo 36 meses do vizinho
COMP_MEDIA12_MED13_24M	Comparação da média de consumo do último ano com a média de consumo do 13º ao 24º mês
COMP_MEDIA12_MED25_36M	Comparação da média de consumo do último ano com a média de consumo do 25º ao 36º mês

Tab. 6 – Variáveis de Serviços e Pagamentos

Atributo	Descrição
FATPAGVENC_12M	Quantidade de faturas pagas até o vencimento nos últimos 12 meses
MEDIA_ATRASO_12M	Diferença média de dias de atraso entre o vencimento e o pagamento das 12 últimas faturas
MAXIMO_VENCIMENTO	Máximo de dias para compensação de uma fatura em atraso nos últimos 12 meses
MINIMA_VENCIMENTO	Mínimo de dias para a compensação de uma fatura em atraso nos últimos 12 meses
CORTEEFETIVO	Quantidade de cortes efetivos
CORTEEXECUTADO	Quantidade de cortes executados
RECORTEEFETIVO	Quantidade de recortes efetivos
RECORTEEXECUTADO	Quantidade de recortes executados
RELIGACAORECORTE	Quantidade de relações recorte
RELIGACAOEFETIVO	Quantidade de relações efetivas
RELIGACAOCORTE	Quantidade de relações corte

3.2 Análise Exploratória

Como as variáveis foram selecionadas com base em trabalhos similares e indicações de colaboradores, nessa etapa do processo foi realizado o cálculo da correlação entre elas, utilizando o coeficiente de Pearson. Ele é dado por 3.1, onde \bar{x} e \bar{y} são as médias aritméticas e x_1, x_2, \dots, x_n e y_1, y_2, \dots, y_n são os valores que compõem os dois conjuntos.

$$r = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum (x_i - \bar{x})^2} \cdot \sqrt{\sum (y_i - \bar{y})^2}} \quad (3.1)$$

É interessante que seja selecionado o melhor subconjunto dos atributos originais preservando toda ou a maior parte da informação dos dados, eliminando aqueles que são irrelevantes ou redundantes.

Mesmo que, em geral, uma redução na dimensão do vetor de entrada signifique uma redução de informação, em aplicações reais, como a quantidade de dados é limitada, a maldição da dimensionalidade leva a dados esparsos e pode reduzir a performance de sistemas de classificação (BISHOP, 1995).

O coeficiente r varia entre -1 e +1 e uma correlação é considerada forte quando esses coeficientes são maiores que 0,8 ou menores que -0,8 (DEVORE, 2010). Assim, utilizando a abordagem seguida por Paulo (2020), foram eliminadas as variáveis numéricas que possuíam correlação forte com alguma outra, de forma a reduzir a redundância. A matriz de correlação pode ser vista na Figura 9, e as variáveis eliminadas foram:

- MAXIMO_VENCIMENTO
- QTD_QUEDA_24M
- QTD_QUEDA_36M
- QTD_QUEDA_48M
- COMP_CV24M_CVVZ24M
- CORTEEXECUTADO
- RELIGACAORECORTE
- RELIGACAOEFETIVO
- RELIGACAOCORTE

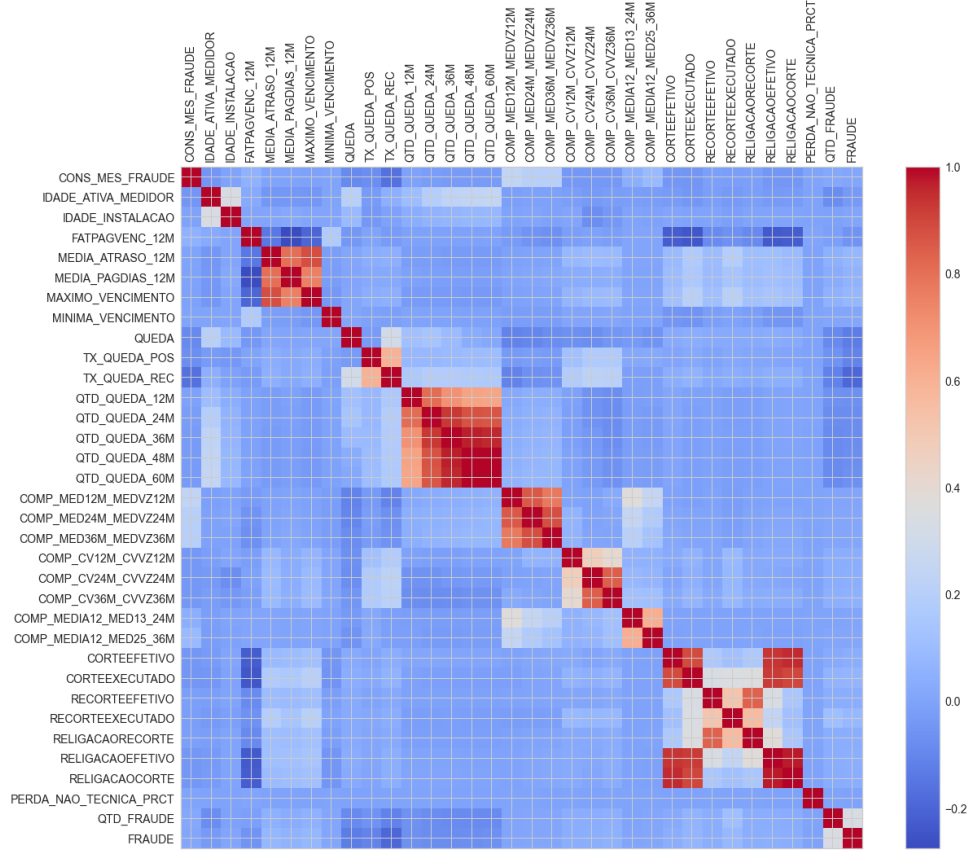
Outra técnica utilizada passível de ser utilizada é a Análise de Componentes Principais (ACP). Tal técnica tem como objetivo produzir combinações lineares de variáveis que capturem o máximo possível a variância das variáveis observadas, sendo em geral utilizada também para fins de redução de dados (FILHO; JUNIOR, 2015).

Para a determinação dos componentes principais, a matriz de covariância dos dados amostrais é considerada, a partir da qual a matriz da Transformada de Hotelling é determinada. A covariância é a variância medida entre duas variáveis e sua fórmula é dada pela Equação (3.2), onde \bar{x} e \bar{y} são as médias aritméticas, x_1, x_2, \dots, x_n e y_1, y_2, \dots, y_n são os valores que compõem os dois conjuntos e n a quantidade de dados da amostra.

$$cov(x, y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n} \quad (3.2)$$

Na matriz da Transformada de Hotelling, as linhas são formadas a partir dos autovetores da matriz de covariância, exemplificada na Equação (3.3), arranjados de modo que a primeira linha seja o autovetor correspondente ao maior autovalor, e assim

Fig. 9 – Matriz de Correlação de Pearson



Fonte: Elaborado pelo autor.

sucessivamente até que a última linha corresponda ao menor autovalor. O autovetor com o maior autovalor associado corresponde à componente principal do conjunto de dados.

$$Matriz_{cov} = \begin{bmatrix} cov(x, x) & cov(x, y) & cov(x, z) \\ cov(y, x) & cov(y, y) & cov(y, z) \\ cov(z, x) & cov(z, y) & cov(z, z) \end{bmatrix} \quad (3.3)$$

Uma das questões associada à [ACP](#) é a determinação do número de componentes (fatores) a ser considerado. Cada componente principal é uma combinação linear das variáveis originais, sendo ortogonais entre si. Quanto mais componentes forem extraídos, menor é o grau de parcimônia, no entanto, maior é a quantidade total de variância representada pelos componentes. Por outro lado, quanto menos componentes forem extraídos, maior é o grau de parcimônia, todavia, menor a quantidade total de variância representada pelos componentes. ([ARAÚJO, 2016](#))

Normalmente apenas uma das técnicas de redução do número de variáveis é utilizada. Neste Trabalho, como o número de variáveis já havia sido reduzido utilizando o coeficiente

de correlação de Pearson, a [ACP](#) não trouxe resultados muito pertinentes. Para manter um desempenho similar dos modelos, avaliado posteriormente, seria necessário manter quase todas as componentes. Dessa forma, tal técnica foi desconsiderada.

3.3 Tratamento dos Dados

Como citado na secção anterior, foram eliminadas algumas variáveis do conjunto selecionado inicialmente, resultando num subconjunto final de 45 variáveis. Muitas delas, entretanto, ainda precisaram passar por algum tratamento antes do treinamento dos modelos.

Inicialmente foram tratadas aquelas variáveis que possuíam ocorrências com valores nulos ou mesmo impossíveis. Para a variável `PERDA_NAO_TECNICA_PRCT`, os valores nulos e menores do que zero foram substituídos pela média aritmética dos valores positivos. Ressalta-se que, na prática, é impossível que haja uma perda percentual negativa, e isso normalmente ocorre devido a erros de cadastro. Para as variáveis `LATITUDE` e `LONGITUDE`, os valores nulos foram substituídos pela latitude e longitude do município onde a instalação está localizada e, quando nem isso foi possível, pela média aritmética de todos os valores.

Em seguida, foi feita a codificação das variáveis categóricas em variáveis numéricas, ou seja, cada ocorrência diferente foi representada por um número. Isso é feito automaticamente com o auxílio da função `LabelEncoder()` da biblioteca *Scikit-Learn*. Por fim, todas as linhas do arquivo de dados que ainda possuíam algum valor nulo foram descartadas, resultando num conjunto final de 46.977 instalações, de um total de 60.432 da base de dados extraída e 290.391 do nicho de clientes considerado.

4 Concepção e Teste dos Modelos

4.1 Criação dos modelos

Para a implementação dos modelos, foi escolhida a linguagem Python. Trata-se de uma linguagem de programação orientada a objetos, fácil escrever em grande escala e com código robusto e, embora não tenha um conjunto de pacotes e bibliotecas tão abrangente como os disponíveis para outras linguagens (linguagem R, por exemplo), a combinação de Python com algumas delas tornam-na uma das principais escolhas entre os Cientistas de Dados (MATOS, 2018). As bibliotecas utilizadas, bem como suas finalidades, foram:

- **Scikit-Learn:** ferramental para trabalhar com grande parte das técnicas de AM, trazendo, além dos classificadores, funções de pré-processamento dos dados e métricas de avaliação dos modelos.
- **OPFython:** classificador OPF.
- **Yellowbrick:** visualização dos dados (como em matrizes de confusão, por exemplo).
- **Pickle:** exportar modelos na forma de um arquivo .sav.
- **Pandas:** manipulação e análise dos dados.
- **NumPy:** funções matemáticas.
- **Datetime:** classes para manipulação de datas e horas.

Foram construídos 5 modelos utilizando técnicas diferentes, a fim de verificar qual se adaptaria melhor ao problema em questão. As técnicas escolhidas foram as mais populares para esse tipo de aplicação, discutidas no Capítulo 2. São elas: SVMs, RNAs (mais especificamente Perceptron Multicamadas), *Random Forest*, *Gradient Boosting* e OPFs.

Em todos os casos, o valor inicial dos parâmetros dos modelos foram escolhidos com base em estudos semelhantes e na experiência própria do autor. Em seguida, eles foram sendo ajustados empiricamente para fornecer um melhor resultado e, no final, foram utilizados os valores da Tabela 7. Os parâmetros que não foram citados assumiram os valores padrão da biblioteca utilizada (OPFython para OPF e Scikit-Learn para as demais técnicas).

Ressalta-se ainda que, para o treinamento dos modelos, a base de dados disponível foi dividida de forma aleatória entre as bases de treinamento (75%) e teste (25%) (*holdout*).

Tab. 7 – Parâmetros dos Modelos

Técnica	Parâmetros Utilizados
Floresta de Caminhos Ótimos	<code>distance = "log_squared_euclidean"</code> <code>pre_computed_distance = None</code> <code>n_iterations = 5</code>
<i>Gradient Boosting</i>	<code>n_estimators = 10</code> <code>learning_rate = 1</code>
Máquina de Vetores de Suporte	<code>kernel = 'linear'</code> <code>C = 1.0</code>
<i>Random Forest</i>	<code>criterion = 'gini'</code> <code>n_estimators = 60</code>
Rede Neural Artificial (Perceptron Multicamadas)	<code>early_stopping = True</code> <code>max_iter = 1000</code> <code>tol = 0.0000010</code> <code>hidden_layer_sizes = (32,)</code> <code>n_iter_no_change = 100</code>

4.2 Testes Teóricos

Para todos os modelos foi gerada a matriz de confusão dos resultados, a partir da qual podem ser calculadas todas as métricas de avaliação descritas no Capítulo 2. Com o auxílio das bibliotecas, contudo, tal cálculo é feito automaticamente com as funções `score()` (Scikit-Learn) e `metrics()` (OPFython). As matrizes de confusão podem ser vistas nas Figuras 10, 11, 12 e 13 e as métricas nas Tabelas 8, 10, 10 e 9.

Fig. 10 – Matriz de Confusão - *Gradient Boosting*

		Classe Preditiva	
		0	1
Classe Real	0	10863 (VN)	43 (FP)
	1	642 (FN)	197 (VP)

Fonte: Elaborado pelo autor.

Tab. 8 – Métricas de Avaliação - *Gradient Boosting*

	Precisão	Sensibilidade	F1-Score	Suporte
0	0,94	1,00	0,97	10906
1	0,82	0,23	0,37	839
Acurácia			0,94	11745
Média Simples	0,88	0,62	0,67	11745
Média Ponderada	0,94	0,94	0,93	11745

Fig. 11 – Matriz de Confusão - Floresta de Caminhos Ótimos

		Classe Preditiva	
		0	1
Classe Real	0	11247 (VN)	396 (FP)
	1	0 (FN)	102 (VP)

Fonte: Elaborado pelo autor.

Tab. 9 – Métricas de Avaliação - Floresta de Caminhos Ótimos

	Precisão	Sensibilidade	F1-Score	Suporte
0	1,00	0,97	0,98	11643
1	0,20	1,00	0,34	102
Acurácia			0,97	11745
Média Simples	0,60	0,98	0,66	11745
Média Ponderada	0,99	0,97	0,98	11745

Fig. 12 – Matriz de Confusão - *Random Forest*

		Classe Preditiva	
		0	1
Classe Real	0	10897 (VN)	9 (FP)
	1	641 (FN)	198 (VP)

Fonte: Elaborado pelo autor.

Tab. 10 – Métricas de Avaliação - *Random Forest*

	Precisão	Sensibilidade	F1-Score	Suporte
0	0,94	1,00	0,97	10906
1	0,96	0,24	0,38	839
Acurácia			0,94	11745
Média Simples	0,95	0,62	0,67	11745
Média Ponderada	0,95	0,94	0,93	11745

Fig. 13 – Matriz de Confusão - *Rede Neural Artificial*

		Classe Preditiva	
		0	1
Classe Real	0	10826 (VN)	80 (FP)
	1	671 (FN)	168 (VP)

Fonte: Elaborado pelo autor.

Tab. 11 – Métricas de Avaliação - *Rede Neural Artificial*

	Precisão	Sensibilidade	F1-Score	Suporte
0	0,94	0,99	0,97	10906
1	0,68	0,20	0,31	839
Acurácia			0,94	11745
Média Simples	0,81	0,60	0,64	11745
Média Ponderada	0,92	0,94	0,92	11745

Nota-se que os resultados para o modelo construído utilizando **SVM** não foram apresentados. Na realidade, a construção de um modelo fazendo uso dessa técnica mostrou-se ser extremamente custosa computacionalmente, demorando mais de um dia para a realização do treinamento e seleção. Assim, a utilização dessa técnica foi considerada inviável no contexto deste Trabalho.

É interessante citar ainda o custo computacional para treinamento e previsão utilizando as outras técnicas. Isso não foi feito de forma analítica, visto que depende muito da máquina que está sendo utilizada, mas observou-se que, enquanto os modelos construídos com **RNA**, *Random Forest* e *Gradient Boosting* demoravam segundos ou poucos minutos para completar o processo, aqueles que utilizavam **OPF** demoravam horas.

No cenário proposto, a previsão de valores 1 (fraude) é a mais importante, visto que as classes são desbalanceadas e a existência de valores 0 (sem fraude) é muito maior. Além disso, como citado no Capítulo 2, as métricas mais importantes nesse contexto são precisão e sensibilidade, resumidas em forma de média harmônica no *F1-score*. Uma comparação de tais métricas, para todos os modelos, pode ser vista na Tabela 12.

Tab. 12 – Comparação dos Modelos

	Precisão	Sensibilidade	F1-Score
Floresta de Caminhos Ótimos	0,20	1,00	0,34
<i>Gradient Boosting</i>	0,82	0,23	0,37
Máquina de Vetores de Suporte	-	-	-
<i>Random Forest</i>	0,96	0,24	0,38
Rede Neural Artificial	0,68	0,20	0,31

Tomando o *F1-score* como base de comparação, vê-se que, teoricamente, o melhor modelo é o que utiliza a técnica de *Random Forest*, seguido de perto pelo modelo com *Gradient Boosting*.

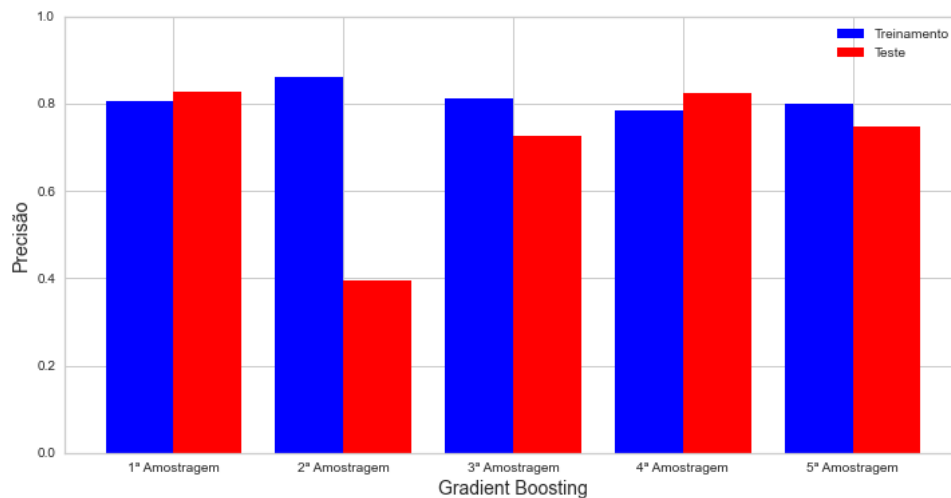
4.2.1 Validação dos Resultados e Testes em Campo

Ao final dos testes teóricos, foi possível selecionar 50 instalações para serem inspecionadas pela distribuidora, de forma a testar a performance em campo. Tal amostra pode não ser tão significativa quando considerado o universo estatístico, mas é razoável para um teste preliminar, sem que o faturamento da distribuidora seja afetado de forma considerável.

Usando o *F1-score* como base, escolheu-se, a princípio, o modelo que utiliza a técnica de *Random Forest* para selecionar as instalações, a partir de uma base de dados com 200.600 elegíveis para inspeção, fornecida também pela distribuidora. Neste caso, 2.986 instalações foram selecionadas. Foi testado ainda o modelo com *Gradient Boosting*, que apresentou um *F1-score* muito parecido, resultando em 5.761 instalações selecionadas. Considerando que a meta de quantidade de inspeções da distribuidora é relativamente elevada, esse segundo foi considerado mais interessante, já que fornece mais opções de instalações a serem inspecionadas.

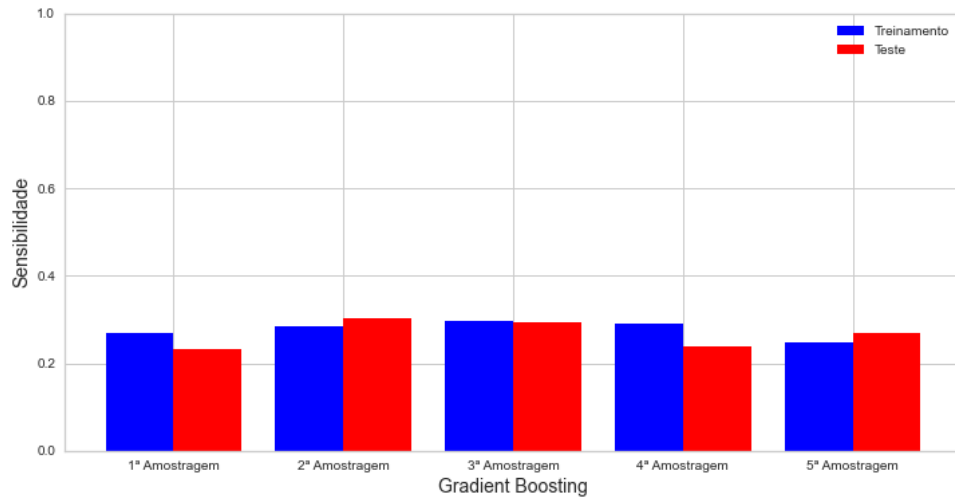
Considerando o tamanho da base de dados, fazendo uso do *holdout* com amostragem aleatória, os resultados de treinamento devem ser razoavelmente precisos. Não obstante, antes do envio a campo, é interessante validar o modelo escolhido utilizando uma técnica mais precisa, tendo sido escolhido o *K-fold* com $K = 5$ e amostragem estratificada. Isso resultou em médias de precisão, sensibilidade e *F1-Score* de, respectivamente, 0,70, 0,26 e 0,37. Os resultados de cada teste podem ser vistos nas Figuras 14, 15 e 16. Nota-se que os valores obtidos são similares utilizando as duas técnicas, sendo o *F1-Score* inclusive igual.

Fig. 14 – *K-fold* estratificado - Precisão

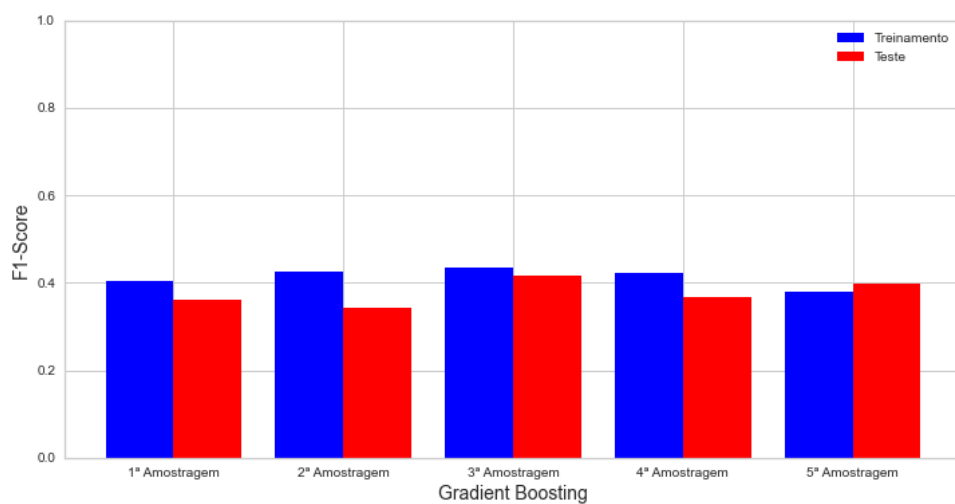


Fonte: Elaborado pelo autor.

Para fins de testes teóricos, as instalações a serem inspecionadas deveriam ser escolhidas aleatoriamente. Para a distribuidora, contudo, o que importa é a quantidade de energia que será recuperada, e na base de dados fornecida existe uma coluna com a previsão de recuperação. Dessa forma, entre as instalações selecionadas pelo modelo, foram escolhidas as 50 com maior previsão de recuperação para serem inspecionadas.

Fig. 15 – *K-fold* estratificado - Sensibilidade

Fonte: Elaborado pelo autor.

Fig. 16 – *K-fold* estratificado - F1-Score

Fonte: Elaborado pelo autor.

4.3 Análise e Adequação dos Modelos

No total, 32 instalações puderam ser inspecionadas, sendo que apenas uma delas detectou perdas. Isso resulta num acerto de 3,13%, inferior ao do modelo já utilizado pela distribuidora, que é de 6%. Vê-se, portanto, que o modelo construído provavelmente não é adequado para as ações de campo, ou pelo menos não para as instalações com maior previsão de recuperação.

Nesse contexto, foi necessário analisar novamente os parâmetros e dados utilizados na construção, já que, como os resultados teóricos foram bons, pode ter havido *overfitting*, quando um modelo estatístico se ajusta muito bem ao conjunto de dados anteriormente observado, mas se mostra ineficaz para prever novos resultados, ou *leakage*, quando há o uso de informações no processo de treinamento do modelo que não deveriam estar disponíveis no momento da predição.

Utilizando o atributo *feature_importances* da biblioteca Scikit-Learn é possível determinar a importância que cada variável tem no modelo final. Fazendo uso dele, foi possível observar que a variável INSTPARCEIROFRAUDE possuía uma importância muito maior do que todas as outras. Essa pode ser de fato ser uma variável importante, mas sua presença não deveria ser tão influente nos resultados como é, levando a acreditar que exista um problema com os dados da base fornecida pela distribuidora. Essa coluna pode estar considerando a última inspeção, o que não deveria ocorrer, consistindo num *leakage*.

Dessa forma, o modelo foi treinado novamente, dessa vez sem fazer uso dessa variável, resultando na matriz de confusão da Figura 17 e nas métricas da Tabela 13. Nota-se que os valores das métricas de avaliação diminuíram bastante. O F1-score, por exemplo, foi de 0,37 para 0,23. Isso mostra como a variável INSTPARCEIROFRAUDE estava influenciando o modelo. Dessa vez, o resultado teórico, apesar de pior, foi mais condizente com a realidade e com outros trabalhos desenvolvidos sobre o tema.

Tab. 13 – Métricas de Avaliação - *Gradient Boosting* sem INSTPARCEIROFRAUDE

	Precisão	Sensibilidade	F1-Score	Suporte
0	0,94	0,99	0,96	10906
1	0,49	0,15	0,23	839
Acurácia			0,93	11745
Média Simples	0,71	0,57	0,60	11745
Média Ponderada	0,91	0,93	0,91	11745

Fig. 17 – Matriz de Confusão - *Gradient Boosting* sem INSTPARCEIROFRAUDE

		Classe Preditiva	
		0	1
Classe Real	0	10772 (VN)	134 (FP)
	1	710 (FN)	129 (VP)

Fonte: Elaborado pelo autor.

Aplicando o modelo à mesma base de dados utilizada anteriormente para selecionar as instalações a serem inspecionadas, foram retornadas 18.735 instalações. Como esse é um valor consideravelmente elevado, decidiu-se por utilizar outra metodologia: o modelo serviu como uma pré-seleção e, dentre as instalações selecionadas por ele, foram escolhidas 50 com base nos critérios utilizados habitualmente pela distribuidora.

Acredita-se que esse processo pode ser interessante, já que a análise manual torna-se muito mais rápida que o normal, uma vez que menos instalações são analisadas, e pode contribuir para aumentar o acerto das inspeções enviadas. Isso foi verificado na prática, já que das 21 instalações que puderam ser inspecionadas, 2 delas detectaram perdas, resultando num acerto de 9,52%.

4.4 Discussão dos Resultados

Nos testes em campo, apesar do modelo construído ter apresentado um resultado superior à metodologia tradicional da distribuidora, como citado na Seção anterior, ainda houve uma divergência considerável entre os resultados dos testes teóricos e de campo. Isso pode ser devido a uma série de fatores, os quais serão discutidos nesta Seção.

Em primeiro lugar é possível citar a qualidade das bases de dados fornecida pela distribuidora. Assim como a INSTPARCEIROFRAUDE, é possível que haja outra variável com incorreções que não pôde ser detectada, enviesando assim os resultados do modelo. Para contornar esse problema seria necessário ter acesso aos códigos de criação da base de

dados e aos dados mãe, que não foram disponibilizados.

Da parte da distribuidora, é possível citar ainda a qualidade do serviço de campo. Se as inspeções não forem feitas minuciosamente, é possível que não sejam detectadas as fraudes mesmo que existam. Isso influencia de forma clara o resultado dos testes em campo, já que, se as fraudes não forem encontradas, o acerto será reduzido. Tal problema, contudo, pode influenciar ainda o treinamento, afinal, se os dados de uma instalação fraudadora forem fornecidos com indicativo que não há fraude na variável *target*, o modelo tomará tal padrão como característico de uma instalação não fraudadora.

Outra questão que pode ter afetado os resultados é a desconsideração da evolução temporal dos dados. Como citado no Capítulo 3, foi feita uma adequação na base de dados considerando a data da última inspeção na instalação como data de referência para a obtenção dos dados (o que é feito nos trabalhos utilizados como referência). Dessa forma, é como se todas as inspeções tivessem ocorrido na mesma data, o que não é verdade na prática. Isso pode ser tornar um problema visto que determinado padrão pode ser uma característica de fraude em determinada época, mas não em outra, afetando o treinamento do modelo. Além disso, em produção, o modelo é aplicado numa base de dados que possui as informações mais atuais das instalações, o que não ocorre no teste. Ressalta-se ainda a influência da sazonalidade nas inspeções (existem fraudes que são mais comuns no verão do que no inverno, por exemplo), o que contribui para uma pior performance do modelo, que não leva isso em consideração.

Esse raciocínio pode ser demonstrado na teoria, dividindo as bases de teste e treinamento de acordo com a data de inspeção. Para isso, foi tomado o ano de 2021 como base, de forma a ter também a influência da sazonalidade, e mantida a proporção de 75% dos dados para treinamento e 25% para teste. Os resultados podem ser vistos na Figura 18 e na Tabela 14. Vê-se que a performance foi reduzida consideravelmente, tendo o *F1-score* baixado de 0,23 para 0,12, e a precisão, que representaria o acerto, de 0,49 para 0,18, o que está mais de acordo com o resultado dos testes em campo.

Por fim, como citado anteriormente, o número de instalações inspecionadas é muito pequeno quando considerado o universo estatístico, sendo impossível a realização de um processo de inferência. A distribuidora normalmente inspeciona cerca de 2.500 instalações desse nicho por mês. Dessa forma, é normal que o resultado dos testes em campo não coincidam com os teóricos. Além disso, a própria sazonalidade pode influenciar nos testes em campo: o acerto da distribuidora varia ao longo do ano, havendo épocas mais comuns de se encontrar fraudes. Assim, idealmente, os testes deveriam ter sido realizados ao longo do período de um ano. Por questões de cronograma, contudo, isso não foi possível.

Fig. 18 – Matriz de Confusão - *Gradient Boosting* considerando evolução temporal

		Classe Preditiva	
		0	1
Classe Real	0	3607 (VN)	84 (FP)
	1	181 (FN)	18 (VP)

Fonte: Elaborado pelo autor.

Tab. 14 – Métricas de Avaliação - *Gradient Boosting* considerando evolução temporal

	Precisão	Sensibilidade	F1-Score	Suporte
0	0,95	0,98	0,96	3691
1	0,18	0,09	0,12	199
Acurácia			0,93	3890
Média Simples	0,56	0,53	0,54	3890
Média Ponderada	0,91	0,93	0,92	3890

5 Conclusão

Neste Trabalho foi abordado o problema das PNTs na rede de distribuição e apresentada uma possível forma de combatê-las fazendo uso de técnicas de AM. Como evidenciado pela revisão bibliográfica, esse é um tema bastante estudado, e a utilização de novas técnicas pode trazer um acréscimo considerável no faturamento das empresas distribuidoras. Além disso, é importante estar sempre inovando, tendo em vista que os consumidores fraudadores também evoluem no sentido de esquivar-se das ações da distribuidora.

A fundamentação teórica trouxe um resumo dos principais conceitos utilizados no desenvolvimento do Trabalho. Na parte de distribuição de energia, foram trazidas informações sobre concessionária, consumidor, medição, faturamento e perdas, enquanto que na parte de AM foram apresentados os algoritmos utilizados, as métricas de avaliação de desempenho e o conceito de validação cruzada.

Os modelos construídos foram desenvolvidos com base em metodologias já utilizadas em outros trabalhos, sobretudo em Paulo (2020), mas esse processo foi realizado após o período da pandemia de Covid-19, na qual houve alteração drástica dos padrões de consumo devido ao isolamento social, e trouxe dados diferentes para sua construção, resultando em uma abordagem diferente e contribuindo para a literatura sobre o tema. Além disso, diferentemente da grande maioria dos trabalhos, foi possível apresentar o resultado de testes em campo, evidenciando uma discrepância entre a prática e a teoria e revelando alguns dos problemas da metodologia em questão.

Os algoritmos utilizados foram SVM, RNA, *Random Forest*, *Gradient Boosting* e OPF. Os que ofereceram melhores resultados teóricos em termos de F1-score (e também de custo computacional) foram os baseados em Árvores de Decisão, ou seja, *Random Forest* e *Gradient Boosting*. Entre eles, o segundo foi escolhido para a realização dos testes em campo por fornecer um número maior de alvos.

O acerto final das inspeções realizadas em campo foi de 9,52%, representando um aumento razoável em relação ao modelo tradicional da distribuidora, que traz um acerto de 6%. Tal acerto foi, contudo, consideravelmente menor do que era esperado a partir da teoria (49%). Isso pode ser devido a uma série de fatores, como qualidade das bases de dados e serviços de campo, universo amostral reduzido e desconsideração de evolução temporal e sazonalidade.

A qualidade das bases de dados e serviços é de responsabilidade da distribuidora, sendo assim um fator externo, mais difícil de ser alterado. Também não seria possível tomar uma amostra de tamanho considerável sem afetar o faturamento da empresa. Dessa

forma, acredita-se que seria interessante para trabalhos futuros a investigação uma forma de considerar a evolução temporal no treinamento do modelo, o que poderia ser feito, por exemplo, tomando apenas dados dos meses imediatamente anteriores ou então do mesmo período do ano anterior (de forma a levar em conta a sazonalidade). A eficiência de tais métodos, entretanto, pode variar de consumidor para consumidor, além de ser afetada pela redução da quantidade de dados de treinamento, sendo necessário um estudo mais aprofundado.

Referências

- ANEEL. *Perdas de Energia*. 2015. Disponível em: <https://www.aneel.gov.br/metodologia-distribuicao/-/asset_publisher/e2INtBH4EC4e/content/perdas/654800>. Acessado em 14/12/2021. Citado na página 23.
- ANEEL. *Módulo 7 - Cálculo de Perdas na Distribuição*. 2018. Disponível em: <<https://www.aneel.gov.br/modulo-7>>. Acessado em 18/12/2021. Citado na página 31.
- ANEEL. *Perdas de Energia Elétrica na Distribuição*. 2021. Disponível em: <https://www.aneel.gov.br/documents/654800/18766993/Relat%C3%B3rio+Perdas+de+Energia__+Edi%C3%A7%C3%A3o+1-2021.pdf/143904c4-3e1d-a4d6-c6f0-94af77bac02a>. Citado 3 vezes nas páginas 23, 24 e 33.
- ANEEL. *Regras de Prestação do Serviço Público de Distribuição de Energia Elétrica*. 2021. Disponível em: <<https://www.in.gov.br/en/web/dou/-/resolucao-normativa-aneel-n-1.000-de-7-de-dezembro-de-2021-368359651>>. Acessado em 18/05/2022. Citado 5 vezes nas páginas 24, 29, 30, 31 e 34.
- ARAÚJO, R. V. de. *Melhoria de Efetividade de Ações de Cobrança Através de Métodos Quantitativos*. Dissertação (Mestrado), 2016. Citado na página 49.
- BARROS, R. M. R. *Advanced Analytics Aplicado à Gestão da Perda Não Técnica em Energia em Sistemas Elétricos de Distribuição*. Tese (Doutorado), 2021. Citado 6 vezes nas páginas 25, 28, 34, 39, 40 e 41.
- BISHOP, C. *Neural networks for pattern recognition*. [S.l.]: Oxford University Press, USA, 1995. Citado 2 vezes nas páginas 37 e 48.
- BREIMAN, L. Random forests. *Machine Learning*, v. 45, p. 5–32, 10 2001. Citado na página 35.
- BROWNLEE, J. A gentle introduction to k-fold cross-validation. *Machine learning mastery*, v. 2019, 2018. Citado na página 42.
- DANTAS, P. R. P. et al. Avaliação de perdas de energia elétrica não-técnicas metodologia aplicada no município de salvador. Universidade Salvador, 2006. Citado na página 24.
- DEVORE, J. *Probabilidade e estatística para engenharia e ciências*. Cengage Learning Edições Ltda., 2010. ISBN 9788522109241. Disponível em: <<https://books.google.com.br/books?id=0dojnQAACAAJ>>. Citado na página 48.
- FACELI, K. et al. *Inteligencia Artificial: Uma Abordagem de Aprendizagem de Máquina*. [S.l.]: LTC, 2011. Citado na página 34.
- FARIA, L. T. d. Estimção espaço-temporal das perdas não técnicas no sistema de distribuição de energia elétrica. Universidade Estadual Paulista (UNESP), 2016. Citado na página 24.
- FERREIRA, H. M. et al. Uso de ferramentas de aprendizado de máquina para prospecção de perdas comerciais em distribuição de energia elétrica. [sn], 2008. Citado na página 33.

- FILHO, D. B. F.; JUNIOR, J. A. S. Visão além do alcance: uma introdução à análise fatorial. *Opinião Pública*, v. 16, n. 1, p. 160–185, out. 2015. Disponível em: <<https://periodicos.sbu.unicamp.br/ojs/index.php/op/article/view/8641349>>. Citado na página 48.
- GOODFELLOW, I. J. et al. *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. <<http://www.deeplearningbook.org>>. Citado na página 37.
- GUFOSOWA. *K-fold cross validation*. 2019. Disponível em: <https://commons.wikimedia.org/wiki/File:K-fold_cross_validation_EN.svg>. Acessado em 24/08/2022. Citado na página 42.
- KELLEHER, J. D.; NAMEE, B. M.; D'ARCY, A. *Fundamentals of machine learning for predictive data analytics: algorithms, worked examples, and case studies*. [S.l.]: MIT press, 2020. Citado na página 39.
- MATOS, D. *R ou Python para Análise de Dados?* 2018. Disponível em: <<https://www.cienciaedados.com/r-ou-python-para-analise-de-dados/>>. Acessado em 10/06/2022. Citado na página 51.
- MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, n. 4, p. 115–133, 1943. Citado na página 36.
- OLIVEIRA, D. R. M. *Utilização de Sensores Inteligentes na detecção e compare às perdas não técnicas na distribuição de energia*. Trabalho de Conclusão de Curso, 2021. Citado na página 27.
- ORTIZ, P. F. *Operações de combate as perdas nao técnicas de energia elétrica utilizando algoritmos de aprendizado de máquina*. Tese (Doutorado) — Universidade Federal do Rio de Janeiro, 2017. Citado na página 33.
- PAPA, J. P. et al. Design of robust pattern classifiers based on optimum-path forests. In: *8th International Symposium on Mathematical Morphology Rio de Janeiro Brazil Oct.* [S.l.: s.n.], 2007. p. 337–348. Citado na página 38.
- PAULO, F. R. *Detecção de fraude em unidades consumidoras não telemedidas com uso de técnicas de aprendizado de máquina*. Dissertação (Mestrado), 2020. Citado 10 vezes nas páginas 25, 27, 28, 34, 35, 37, 38, 43, 48 e 65.
- PYLE, D. *Data Preparation for Data Mining*. [S.l.]: Elsevier Science, 1999. (ITPro collection). ISBN 9781558605299. Citado na página 43.
- RAMACHANDRAN, P.; ZOPH, B.; LE, Q. V. *Searching for Activation Functions*. 2017. Citado na página 37.
- RAMOS, C. C. O. *Desenvolvimento de Ferramentas Computacionais Inteligentes para Identificação de Perdas Comerciais em sistemas de Energia*. Dissertação (Mestrado), 2010. Citado 4 vezes nas páginas 25, 28, 34 e 39.
- ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958. Citado na página 36.

SAEED, M. S. et al. Detection of non-technical losses in power utilities—a comprehensive systematic review. *Energies*, Multidisciplinary Digital Publishing Institute, v. 13, n. 18, p. 4727, 2020. Citado na página [28](#).

SIMON, P. *Too Big to Ignore: The Bussines Case for Big Data*. [S.l.]: John Wiley and Sons, 2013. Citado na página [34](#).

VIEGAS, J. L. et al. Solutions for detection of non-technical losses in the electricity grid: A review. *Renewable and Sustainable Energy Reviews*, Elsevier, v. 80, p. 1256–1268, 2017. Citado 3 vezes nas páginas [27](#), [28](#) e [29](#).

APÊNDICE A – Códigos em Python

Neste Apêndice constam os códigos utilizados para a criação e treinamento dos modelos e para a seleção das instalações. Ressalta-se que, no primeiro, os blocos de código referentes a 4 das 5 técnicas utilizadas estão na forma de *string*, sendo criado o modelo apenas para uma delas.

A.1 Treinamento

```
import pandas as pd
import pickle
import datetime
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import classification_report
from sklearn import preprocessing
from sklearn import model_selection
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.neural_network import MLPClassifier
from opfython.models import SupervisedOPF
from yellowbrick.classifier import ConfusionMatrix
from sklearn import metrics
from sklearn.utils import shuffle
from sklearn.model_selection import cross_validate

now = datetime.datetime.now() # current date and time
date_time = now.strftime("%Y%m%d_%H%M%S")

#Importar base de dados

data_ref = datetime.datetime(2022, 8, 12)
df = pd.read_csv("data.csv",sep=',', error_bad_lines=False)
```

```

df = df.replace(r'\\','.', regex=True)

df['DT_ULT_EXEC'] = pd.to_datetime(df['DT_ULT_EXEC'])
df['PERDA_NAO_TECNICA_PRCT'] = pd.to_numeric(df['PERDA_NAO_TECNICA_PRCT'])
df['LATITUDE'] = pd.to_numeric(df['LATITUDE'])
df['LONGITUDE'] = pd.to_numeric(df['LONGITUDE'])

#TRATAMENTO VALORES IMPOSSÍVEL E NULOS-----

df['CORTEEFETIVO'][pd.isna(df['CORTEEFETIVO'])] = 0
df['CORTEEXECUTADO'][pd.isna(df['CORTEEXECUTADO'])] = 0
df['RECORTEEFETIVO'][pd.isna(df['RECORTEEFETIVO'])] = 0
df['RECORTEEXECUTADO'][pd.isna(df['RECORTEEXECUTADO'])] = 0
df['RELIGACAORECORTE'][pd.isna(df['RELIGACAORECORTE'])] = 0
df['RELIGACAOEFETIVO'][pd.isna(df['RELIGACAOEFETIVO'])] = 0
df['RELIGACAOCORTE'][pd.isna(df['RELIGACAOCORTE'])] = 0
df['PERDA_NAO_TECNICA_PRCT'][df['PERDA_NAO_TECNICA_PRCT']<=0 ] = \
    df['PERDA_NAO_TECNICA_PRCT'][df['PERDA_NAO_TECNICA_PRCT']>=0].mean()
df['PERDA_NAO_TECNICA_PRCT'][pd.isna(df['PERDA_NAO_TECNICA_PRCT'])] = \
    df['PERDA_NAO_TECNICA_PRCT'][df['PERDA_NAO_TECNICA_PRCT']>=0].mean()
df['QTD_FRAUDE'][pd.isna(df['QTD_FRAUDE'])] = 0
df['LATITUDE'][pd.isna(df['LATITUDE'])] = df['LATITUDE'].mean()
df['LONGITUDE'][pd.isna(df['LONGITUDE'])] = df['LONGITUDE'].mean()

#-----

#CODIFICAÇÃO DE VARIÁVEIS CATEGÓRICAS-----
le_motivo = preprocessing.LabelEncoder()
le_tip_instalacao = preprocessing.LabelEncoder()
le_fab_medidor = preprocessing.LabelEncoder()
le_modelo_medidor = preprocessing.LabelEncoder()
le_classe_contacontrato = preprocessing.LabelEncoder()
le_subclasse_contacontrato = preprocessing.LabelEncoder()
le_modliga = preprocessing.LabelEncoder()
le_ocle_atual = preprocessing.LabelEncoder()

```

```

le_familiainspant = preprocessing.LabelEncoder()
le_tipo_local = preprocessing.LabelEncoder()

df['MOTIVO'] = le_motivo.fit_transform(df['MOTIVO'])
df['TIP_INSTALACAO'] = le_tip_instalacao.fit_transform(df['TIP_INSTALACAO'])
df['FAB_MEDIDOR'] = le_fab_medidor.fit_transform(df['FAB_MEDIDOR'])
df['MODELO_MEDIDOR'] = le_modelo_medidor.fit_transform(df['MODELO_MEDIDOR'])
df['CLASSE_CONTAcontrato'] = \
    le_classe_contacontrato.fit_transform(df['CLASSE_CONTAcontrato'])
df['SUBCLASSE_CONTAcontrato'] = \
    le_subclasse_contacontrato.fit_transform(df['SUBCLASSE_CONTAcontrato'])
df['MODLIGA'] = le_modliga.fit_transform(df['MODLIGA'])
df['OCLE_ATUAL'] = le_ocle_atual.fit_transform(df['OCLE_ATUAL'])
df['FAMILIAINSPANT'] = le_familiainspant.fit_transform(df['FAMILIAINSPANT'])
df['TIPO_LOCAL'] = le_tipo_local.fit_transform(df['TIPO_LOCAL'])

#-----

# DATAFRAME DE CONSUMO-----
dfcons = df.loc[:, 'MES-01_CONS': 'MES-60_CONS']
dfcons['DT_ULT_EXEC'] = df['DT_ULT_EXEC']
dfcons.set_index('DT_ULT_EXEC', inplace=True)

datas_exec = dfcons.index.tolist()
for i in range(0, len(df)):
    data_exec = datas_exec[i]
    difmes = (data_ref.year - data_exec.year) * 12 + \
        (data_ref.month - data_exec.month)
    nova_row = dfcons.iloc[i].shift(periods=-(difmes+1))
    dfcons.iloc[i] = nova_row

dfcons.reset_index(inplace=True)

dfcons.drop('DT_ULT_EXEC', axis=1, inplace=True)
dfcons.drop(dfcons.loc[:, 'MES-37_CONS': 'MES-60_CONS'], \
            inplace = True, axis = 1)

```

```

dfcons = dfcons.loc[:, 'MES-01_CONS': 'MES-36_CONS'].dropna()

#-----

#DATAFRAME DO MODELO-----

dfmod = pd.DataFrame()

dfmod['CONS_MES_FRAUDE'] = dfcons['MES-01_CONS']
dfmod['LATITUDE'] = df['LATITUDE']
dfmod['LONGITUDE'] = df['LONGITUDE']
dfmod['MOTIVO'] = df['MOTIVO']
dfmod['TIP_INSTALACAO'] = df['TIP_INSTALACAO']
dfmod['FAB_MEDIDOR'] = df['FAB_MEDIDOR']
dfmod['MODELO_MEDIDOR'] = df['MODELO_MEDIDOR']
dfmod['IDADE_ATIVA_MEDIDOR'] = df['IDADE_ATIVA_MEDIDOR']
dfmod['IDADE_INSTALACAO'] = df['IDADE_INSTALACAO']
dfmod['CLASSE_CONTAcontrato'] = df['CLASSE_CONTAcontrato']
dfmod['SUBCLASSE_CONTAcontrato'] = df['SUBCLASSE_CONTAcontrato']
dfmod['TENSAOFORNEC'] = df['TENSAOFORNEC']
dfmod['TENSAOMED'] = df['TENSAOMED']
dfmod['MODLIGA'] = df['MODLIGA']
dfmod['OCLE_ATUAL'] = df['OCLE_ATUAL']
dfmod['FAMILIAINSPANT'] = df['FAMILIAINSPANT']
dfmod['TIPO_LOCAL'] = df['TIPO_LOCAL']
dfmod['BAIXARENDA'] = df['BAIXARENDA']
dfmod['INSTPARCEIROFRAUDE'] = df['INSTPARCEIROFRAUDE']
dfmod['FATPAGVENC_12M'] = df['FATPAGVENC_12M']
dfmod['MEDIA_ATRASO_12M'] = df['MEDIA_ATRASO_12M']
dfmod['MEDIA_PAGDIAS_12M'] = df['MEDIA_PAGDIAS_12M']
dfmod['MAXIMO_VENCIMENTO'] = df['MAXIMO_VENCIMENTO']
dfmod['MINIMA_VENCIMENTO'] = df['MINIMA_VENCIMENTO']
dfmod['QUEDA'] = df['QUEDA']
dfmod['TX_QUEDA_POS'] = df['TX_QUEDA_POS']
dfmod['TX_QUEDA_REC'] = df['TX_QUEDA_REC']
dfmod['QTD_QUEDA_12M'] = df['QTD_QUEDA_12M']
dfmod['QTD_QUEDA_24M'] = df['QTD_QUEDA_24M']
dfmod['QTD_QUEDA_36M'] = df['QTD_QUEDA_36M']

```

```

dfmod['QTD_QUEDA_48M'] = df['QTD_QUEDA_48M']
dfmod['QTD_QUEDA_60M'] = df['QTD_QUEDA_60M']
dfmod['COMP_MED12M_MEDVZ12M'] = df['COMP_MED12M_MEDVZ12M']
dfmod['COMP_MED24M_MEDVZ24M'] = df['COMP_MED24M_MEDVZ24M']
dfmod['COMP_MED36M_MEDVZ36M'] = df['COMP_MED36M_MEDVZ36M']
dfmod['COMP_CV12M_CVVZ12M'] = df['COMP_CV12M_CVVZ12M']
dfmod['COMP_CV24M_CVVZ24M'] = df['COMP_CV24M_CVVZ24M']
dfmod['COMP_CV36M_CVVZ36M'] = df['COMP_CV36M_CVVZ36M']
dfmod['COMP_MEDIA12_MED13_24M'] = df['COMP_MEDIA12_MED13_24M']
dfmod['COMP_MEDIA12_MED25_36M'] = df['COMP_MEDIA12_MED25_36M']
dfmod['CORTEEFETIVO'] = df['CORTEEFETIVO']
dfmod['CORTEEXECUTADO'] = df['CORTEEXECUTADO']
dfmod['RECORTEEFETIVO'] = df['RECORTEEFETIVO']
dfmod['RECORTEEXECUTADO'] = df['RECORTEEXECUTADO']
dfmod['RELIGACAORECORTE'] = df['RELIGACAORECORTE']
dfmod['RELIGACAOEFETIVO'] = df['RELIGACAOEFETIVO']
dfmod['RELIGACAOCORTE'] = df['RELIGACAOCORTE']
dfmod['PERDA_NAO_TECNICA_PRCT'] = df['PERDA_NAO_TECNICA_PRCT']
dfmod['QTD_FRAUDE'] = df['QTD_FRAUDE']
dfmod['FRAUDE'] = df['FRAUDE']

```

```
dfmod = dfmod.dropna()
```

```
#-----
```

```
#MATRIZ DE CORRELAÇÃO-----
```

```
dfmat = pd.DataFrame() #Não entram variáveis categóricas
```

```

dfmat['CONS_MES_FRAUDE'] = dfcons['MES-01_CONS']
dfmat['IDADE_ATIVA_MEDIDOR'] = dfmod['IDADE_ATIVA_MEDIDOR']
dfmat['IDADE_INSTALACAO'] = dfmod['IDADE_INSTALACAO']
dfmat['FATPAGVENC_12M'] = dfmod['FATPAGVENC_12M']
dfmat['MEDIA_ATRASO_12M'] = dfmod['MEDIA_ATRASO_12M']
dfmat['MEDIA_PAGDIAS_12M'] = dfmod['MEDIA_PAGDIAS_12M']
dfmat['MAXIMO_VENCIMENTO'] = dfmod['MAXIMO_VENCIMENTO']
dfmat['MINIMA_VENCIMENTO'] = dfmod['MINIMA_VENCIMENTO']
dfmat['QUEDA'] = dfmod['QUEDA']
dfmat['TX_QUEDA_POS'] = dfmod['TX_QUEDA_POS']

```

```

dfmat['TX_QUEDA_REC'] = dfmod['TX_QUEDA_REC']
dfmat['QTD_QUEDA_12M'] = dfmod['QTD_QUEDA_12M']
dfmat['QTD_QUEDA_24M'] = dfmod['QTD_QUEDA_24M']
dfmat['QTD_QUEDA_36M'] = dfmod['QTD_QUEDA_36M']
dfmat['QTD_QUEDA_48M'] = dfmod['QTD_QUEDA_48M']
dfmat['QTD_QUEDA_60M'] = dfmod['QTD_QUEDA_60M']
dfmat['COMP_MED12M_MEDVZ12M'] = dfmod['COMP_MED12M_MEDVZ12M']
dfmat['COMP_MED24M_MEDVZ24M'] = dfmod['COMP_MED24M_MEDVZ24M']
dfmat['COMP_MED36M_MEDVZ36M'] = dfmod['COMP_MED36M_MEDVZ36M']
dfmat['COMP_CV12M_CVVZ12M'] = dfmod['COMP_CV12M_CVVZ12M']
dfmat['COMP_CV24M_CVVZ24M'] = dfmod['COMP_CV24M_CVVZ24M']
dfmat['COMP_CV36M_CVVZ36M'] = dfmod['COMP_CV36M_CVVZ36M']
dfmat['COMP_MEDIA12_MED13_24M'] = dfmod['COMP_MEDIA12_MED13_24M']
dfmat['COMP_MEDIA12_MED25_36M'] = dfmod['COMP_MEDIA12_MED25_36M']
dfmat['CORTEEFETIVO'] = dfmod['CORTEEFETIVO']
dfmat['CORTEEXECUTADO'] = dfmod['CORTEEXECUTADO']
dfmat['RECORTEEFETIVO'] = dfmod['RECORTEEFETIVO']
dfmat['RECORTEEXECUTADO'] = dfmod['RECORTEEXECUTADO']
dfmat['RELIGACAORECORTE'] = dfmod['RELIGACAORECORTE']
dfmat['RELIGACAOEFETIVO'] = dfmod['RELIGACAOEFETIVO']
dfmat['RELIGACAOCORTE'] = dfmod['RELIGACAOCORTE']
dfmat['PERDA_NAO_TECNICA_PRCT'] = dfmod['PERDA_NAO_TECNICA_PRCT']
dfmat['QTD_FRAUDE'] = dfmod['QTD_FRAUDE']
dfmat['FRAUDE'] = dfmod['FRAUDE']

correlacoes = dfmat.corr()

f1 = plt.figure(figsize=(19, 15))
plt.matshow(dfmat.corr(), fignum=f1.number, cmap='coolwarm')
plt.xticks(range(dfmat.shape[1]), dfmat.columns, fontsize=14, rotation=90)
plt.yticks(range(dfmat.shape[1]), dfmat.columns, fontsize=14)
cb = plt.colorbar( )
cb.ax.tick_params(labelsize=14)
plt.title('Matriz de Correlação', fontsize=16);

#-----

#REMOÇÃO DE VARIÁVEIS CORRELATAS
dfmod = dfmod.drop('MAXIMO_VENCIMENTO', axis=1)

```

```

dfmod = dfmod.drop('QTD_QUEDA_24M', axis=1)
dfmod = dfmod.drop('QTD_QUEDA_36M', axis=1)
dfmod = dfmod.drop('QTD_QUEDA_48M', axis=1)
dfmod = dfmod.drop('COMP_CV24M_CVVZ24M', axis=1)
dfmod = dfmod.drop('CORTEEXECUTADO', axis=1)
dfmod = dfmod.drop('RELIGACAORECORTE', axis=1)
dfmod = dfmod.drop('RELIGACAOEFETIVO', axis=1)
dfmod = dfmod.drop('RELIGACAOCORTE', axis=1)

```

```

dfmat = dfmat.drop('MAXIMO_VENCIMENTO', axis=1)
dfmat = dfmat.drop('QTD_QUEDA_24M', axis=1)
dfmat = dfmat.drop('QTD_QUEDA_36M', axis=1)
dfmat = dfmat.drop('QTD_QUEDA_48M', axis=1)
dfmat = dfmat.drop('COMP_CV24M_CVVZ24M', axis=1)
dfmat = dfmat.drop('CORTEEXECUTADO', axis=1)
dfmat = dfmat.drop('RELIGACAORECORTE', axis=1)
dfmat = dfmat.drop('RELIGACAOEFETIVO', axis=1)
dfmat = dfmat.drop('RELIGACAOCORTE', axis=1)

```

```

f2 = plt.figure(figsize=(19, 15))
plt.matshow(dfmat.corr(), fignum=f2.number, cmap='coolwarm')
plt.xticks(range(dfmat.shape[1]), dfmat.columns, fontsize=14, rotation=90)
plt.yticks(range(dfmat.shape[1]), dfmat.columns, fontsize=14)
cb = plt.colorbar( )
cb.ax.tick_params(labelsize=14)
plt.title('Matriz de Correlação Reduzida', fontsize=16);
#-----

```

```

#MACHINE LEARNING-----

```

```

X_mod = dfmod.iloc[:,0:(len(dfmod.columns)-1)].values
y_mod = dfmod.iloc[:,(len(dfmod.columns)-1)].values

X_mod_train, X_mod_test, y_mod_train, y_mod_test = \
    model_selection.train_test_split(X_mod,y_mod,

```

```

    test_size = 0.25, random_state = 1, shuffle=True)

#-----

#SVM-----
'''

svm = SVC(kernel='linear', random_state = 1, C = 5)
svm.fit(X_mod_train, y_mod_train)

pickle.dump(svm, open('../MODELOS/SVM/svm_'+date_time+'.sav', 'wb'))

previsoes = svm.predict(X_mod_test)

f3 = plt.figure(figsize=(19, 15))
cm = ConfusionMatrix(svm)
cm.fit(X_mod_train, y_mod_train)
cm.score(X_mod_test, y_mod_test)
plt.title('Matriz de Confusão', fontsize=16);

print(classification_report(y_mod_test, previsoes))

'''
#-----

#Random Forest-----
'''

#SEM IPF
#rfc = RandomForestClassifier(n_estimators=60, criterion='gini', \
#    random_state = 1)
#COM IPF
rfc = RandomForestClassifier(n_estimators=60, criterion='gini', \

```



```
        random_state = 1)
rfc.fit(X_mod_train, y_mod_train)

pickle.dump(rfc, open('../MODELOS/RFC/rfc_'+date_time+'.sav', 'wb'))

previsoes = rfc.predict(X_mod_test)

f3 = plt.figure(figsize=(19, 15))
cm = ConfusionMatrix(rfc)
cm.fit(X_mod_train, y_mod_train)
cm.score(X_mod_test, y_mod_test)
plt.title('Matriz de Confusão', fontsize=16);

print(classification_report(y_mod_test, previsoes))

'''
#-----

#Gradient Boosting-----

#SEM IPF
#gbc = GradientBoostingClassifier(n_estimators=100, learning_rate=0.5,\
#    random_state = 1)
#COM IPF
gbc = GradientBoostingClassifier(n_estimators=10, learning_rate=1, \
                                random_state = 1)
gbc.fit(X_mod_train, y_mod_train)

pickle.dump(gbc, open('../MODELOS/GBC/gbc_'+date_time+'.sav', 'wb'))

previsoes = gbc.predict(X_mod_test)
```

```

f3 = plt.figure(figsize=(19, 15))
cm = ConfusionMatrix(gbc)
cm.fit(X_mod_train, y_mod_train)
cm.score(X_mod_test, y_mod_test)
plt.title('Matriz de Confusão', fontsize=16);

print(classification_report(y_mod_test, previsoes))

#-----

#Rede Neural-----
,,,
#SEM IPF
#rna = MLPClassifier(verbose=True, early_stopping= True, max_iter=400, \
#    learning_rate_init = 0.001, tol = 0.0000010, hidden_layer_sizes = (32,), \
#    n_iter_no_change = 1000, random_state=1)
#COM IPF
rna = MLPClassifier(verbose=True, early_stopping= True, max_iter=1000,\
    learning_rate_init = 0.001, tol = 0.0000010, hidden_layer_sizes = (32,), \
    n_iter_no_change = 100, random_state=1)

rna.fit(X_mod_train, y_mod_train)

pickle.dump(rna, open('../MODELOS/RNA/rna_'+date_time+'.sav', 'wb'))

previsoes = rna.predict(X_mod_test)

f3 = plt.figure(figsize=(19, 15))
cm = ConfusionMatrix(rna)
cm.fit(X_mod_train, y_mod_train)
cm.score(X_mod_test, y_mod_test)
plt.title('Matriz de Confusão', fontsize=16);

```

```

print(classification_report(y_mod_test, previsoes))

f4 = plt.figure(figsize=(19, 15))
plt.plot(rna.loss_curve_)
plt.plot(rna.validation_scores_)
'''
#-----

#Optimum Path Forest-----

'''
opf = SupervisedOPF(distance="log_squared_euclidean", \
                    pre_computed_distance=None)
opf.learn(X_mod_train, y_mod_train, X_mod_test ,y_mod_test, n_iterations=5)

pickle.dump(opf, open('../MODELOS/OPF/opf_'+date_time+'.sav', 'wb'))

previsoes = opf.predict(X_mod_test)

f3 = plt.figure(figsize=(19, 15))
cm = metrics.confusion_matrix(y_mod_test, previsoes)
cm_display = metrics.ConfusionMatrixDisplay(confusion_matrix = cm, \
                                             display_labels = [False, True])

cm_display.plot()
plt.title('Matriz de Confusão', fontsize=16);

print(classification_report(y_mod_test, previsoes))
'''
#-----

#Testes-----

'''
with open(r"../MODELOS/SVM/modelo_20220706_185711.sav", "rb") as input_file:
    modelo = pickle.load(input_file)

previsoes = modelo.predict(X_mod_test)

```

```

f3 = plt.figure(figsize=(19, 15))
cm = ConfusionMatrix(svm)
cm.fit(X_mod_train, y_mod_train)
cm.score(X_mod_test, y_mod_test)
plt.title('Matriz de Confusão', fontsize=16);

print(classification_report(y_mod_test, previsoes))
'''
#-----

#Validação cruzada-----

#Autor das funções: Iniabasi Affiah
#Fonte: https://www.section.io/engineering-education/how-to-implement-k-fold-\\
#    cross-validation/

def cross_validation(model, _X, _y, _cv=5):

    '''Function to perform 5 Folds Cross-Validation
    Parameters
    -----
    model: Python Class, default=None
        This is the machine learning algorithm to be used for training.
    _X: array
        This is the matrix of features.
    _y: array
        This is the target variable.
    _cv: int, default=5
        Determines the number of folds for cross-validation.
    Returns
    -----
    The function returns a dictionary containing the metrics 'accuracy', \\
        'precision',
        'recall', 'f1' for both training set and validation set.
    '''
    _scoring = ['accuracy', 'precision', 'recall', 'f1']
    results = cross_validate(estimator=model,

```

```

        X=_X,
        y=_y,
        cv=_cv,
        scoring=_scoring,
        return_train_score=True)

return {"Training Accuracy scores": results['train_accuracy'],
        "Mean Training Accuracy": results['train_accuracy'].mean()*100,
        "Training Precision scores": results['train_precision'],
        "Mean Training Precision": results['train_precision'].mean(),
        "Training Recall scores": results['train_recall'],
        "Mean Training Recall": results['train_recall'].mean(),
        "Training F1 scores": results['train_f1'],
        "Mean Training F1 Score": results['train_f1'].mean(),
        "Validation Accuracy scores": results['test_accuracy'],
        "Mean Validation Accuracy": results['test_accuracy'].mean()*100,
        "Validation Precision scores": results['test_precision'],
        "Mean Validation Precision": results['test_precision'].mean(),
        "Validation Recall scores": results['test_recall'],
        "Mean Validation Recall": results['test_recall'].mean(),
        "Validation F1 scores": results['test_f1'],
        "Mean Validation F1 Score": results['test_f1'].mean()
}

def plot_result(x_label, y_label, plot_title, train_data, val_data):
    '''Function to plot a grouped bar chart showing the training and
    validation results of the ML model in each fold after applying K-fold
    cross-validation.

    Parameters
    -----
    x_label: str,
        Name of the algorithm used for training e.g 'Decision Tree'

    y_label: str,
        Name of metric being visualized e.g 'Accuracy'

```

```

plot_title: str,
    This is the title of the plot e.g 'Accuracy Plot'

train_result: list, array
    This is the list containing either training precision, accuracy,
    or f1 score.

val_result: list, array
    This is the list containing either validation precision, accuracy,
    or f1 score.
Returns
-----
The function returns a Grouped Barchart showing the training and
validation result in each fold.
'''

```

```

# Set size of plot
plt.figure(figsize=(12,6))
labels = ["1ª Amostragem", "2ª Amostragem", "3ª Amostragem", \
          "4ª Amostragem", "5ª Amostragem"]
X_axis = np.arange(len(labels))
ax = plt.gca()
plt.ylim(0, 1)
plt.bar(X_axis-0.2, train_data, 0.4, color='blue', label='Treinamento')
plt.bar(X_axis+0.2, val_data, 0.4, color='red', label='Teste')
plt.title(plot_title, fontsize=30)
plt.xticks(X_axis, labels)
plt.xlabel(x_label, fontsize=14)
plt.ylabel(y_label, fontsize=14)
plt.legend()
plt.grid(True)
plt.show()

```

```

resultcrossval = cross_validation(gbc, X_mod, y_mod, 5)

```

```

model_name = "Gradient Boosting"
plot_result(model_name,
            "Precisão",

```

```

        "",
        resultcrossval["Training Precision scores"],
        resultcrossval["Validation Precision scores"])

plot_result(model_name,
            "Sensibilidade",
            "",
            resultcrossval["Training Recall scores"],
            resultcrossval["Validation Recall scores"])

plot_result(model_name,
            "F1-Score",
            "",
            resultcrossval["Training F1 scores"],
            resultcrossval["Validation F1 scores"])

print("precisão = "+str(resultcrossval["Mean Validation Precision"]))
print("sensibilidade = "+str(+resultcrossval["Mean Validation Recall"]))
print("F1-score = "+str(resultcrossval["Mean Validation F1 Score"]))

```

A.2 Seleção

```

import pandas as pd
import pickle
import datetime
import matplotlib.pyplot as plt
import numpy as np
from sklearn.metrics import classification_report
from sklearn import preprocessing
from sklearn import model_selection
from sklearn.svm import SVC
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.neural_network import MLPClassifier
from opfython.models import SupervisedOPF
from yellowbrick.classifier import ConfusionMatrix
from sklearn import metrics
from sklearn.utils import shuffle

```

```

now = datetime.datetime.now() # current date and time
date_time = now.strftime("%Y%m%d_%H%M%S")

#Importar base de dados

dfprod = pd.read_csv("data_prod.csv",sep=',', error_bad_lines=False)
dfprod = dfprod.replace(r'\\',',', regex=True)
dfprod = dfprod.set_index('ZCGINSTAL')

dfprod= dfprod.drop('INSTPARCEIROFRAUDE', axis=1)

with open(r"../MODELOS/RNA/rna_20220714_104726.sav", "rb") as input_file:
    modelo = pickle.load(input_file)

#TRATAMENTO VALORES IMPOSSÍVEL E NULOS-----

dfprod['CORTEEFETIVO'][pd.isna(dfprod['CORTEEFETIVO'])] = 0
dfprod['RECORTEEFETIVO'][pd.isna(dfprod['RECORTEEFETIVO'])] = 0
dfprod['RECORTEEXECUTADO'][pd.isna(dfprod['RECORTEEXECUTADO'])] = 0
dfprod['PERDA_NAO_TECNICA_PRCT'][dfprod['PERDA_NAO_TECNICA_PRCT']<=0 ] = \
    dfprod['PERDA_NAO_TECNICA_PRCT'][dfprod['PERDA_NAO_TECNICA_PRCT']>=0].mean()
dfprod['PERDA_NAO_TECNICA_PRCT'][pd.isna(dfprod['PERDA_NAO_TECNICA_PRCT'])] =\
    dfprod['PERDA_NAO_TECNICA_PRCT'][dfprod['PERDA_NAO_TECNICA_PRCT']>=0].mean()
dfprod['QTD_FRAUDE'][pd.isna(dfprod['QTD_FRAUDE'])] = 0
dfprod['LATITUDE'][pd.isna(dfprod['LATITUDE'])] = dfprod['LATITUDE'].mean()
dfprod['LONGITUDE'][pd.isna(dfprod['LONGITUDE'])] = dfprod['LONGITUDE'].mean()

#-----

#CODIFICAÇÃO DE VARIÁVEIS CATEGÓRICAS-----

le_motivo = preprocessing.LabelEncoder()
le_tip_instalacao = preprocessing.LabelEncoder()
le_fab_medidor = preprocessing.LabelEncoder()
le_modelo_medidor = preprocessing.LabelEncoder()
le_classe_contacontrato = preprocessing.LabelEncoder()
le_subclasse_contacontrato = preprocessing.LabelEncoder()

```



```

le_modliga = preprocessing.LabelEncoder()
le_ocle_atual = preprocessing.LabelEncoder()
le_familiainspant = preprocessing.LabelEncoder()
le_tipo_local = preprocessing.LabelEncoder()

dfprod['MOTIVO'] = le_motivo.fit_transform(dfprod['MOTIVO'])
dfprod['TIP_INSTALACAO'] = \
    le_tip_instalacao.fit_transform(dfprod['TIP_INSTALACAO'])
dfprod['FAB_MEDIDOR'] = le_fab_medidor.fit_transform(dfprod['FAB_MEDIDOR'])
dfprod['MODELO_MEDIDOR'] = \
    le_modelo_medidor.fit_transform(dfprod['MODELO_MEDIDOR'])
dfprod['CLASSE_CONTAcontrato'] = \
    le_classe_contacontrato.fit_transform(dfprod['CLASSE_CONTAcontrato'])
dfprod['SUBCLASSE_CONTAcontrato'] = \
    le_subclasse_contacontrato.fit_transform(dfprod['SUBCLASSE_CONTAcontrato'])
dfprod['MODLIGA'] = le_modliga.fit_transform(dfprod['MODLIGA'])
dfprod['OCLE_ATUAL'] = le_ocle_atual.fit_transform(dfprod['OCLE_ATUAL'])
dfprod['FAMILIAINSPANT'] = \
    le_familiainspant.fit_transform(dfprod['FAMILIAINSPANT'])
dfprod['TIPO_LOCAL'] = le_tipo_local.fit_transform(dfprod['TIPO_LOCAL'])

#-----

dfprod = dfprod.dropna()

previsoes_prod = modelo.predict_proba(dfprod)
df_result = pd.DataFrame(previsoes_prod, index = dfprod.index)

#previsoes_prod = modelo.predict(dfprod)
#df_result = pd.DataFrame(previsoes_prod, columns = ['FRAUDE'],\
    index = dfprod.index)

df_result = shuffle(df_result)

df_result.to_csv('../RESULTADOS/resultado_'+str(modelo)+'_'+date_time+\
    '.csv', sep=';')

```