UNIVERSIDADE ESTADUAL DA PARAÍBA
CAMPUS I – CAMPINA GRANDE
CENTRO DE CIÊNCIAS E TECNOLOGIA
DEPARTAMENTO DE COMPUTAÇÃO
CURSO DE GRADUAÇÃO EM BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

ARTHUR LINCOLN DA PAZ CRISTOVÃO

USO DE TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL PARA IDENTIFICAÇÃO DE CHAMADAS PROCEDENTES E IMPROCEDENTES

CAMPINA GRANDE – PB

2024

ARTHUR LINCOLN DA PAZ CRISTOVÃO

USO DE TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL PARA IDENTIFICAÇÃO DE CHAMADAS PROCEDENTES E IMPROCEDENTES

Trabalho de Conclusão de Curso apresentado ao Departamento do Curso de Computação da Universidade Estadual da Paraíba, como requisito parcial à obtenção do título de Bacharel em Ciência da Computação.

Área de concentração: Inteligência Artificial

Orientador(a): Profª. Drª. Kézia de Vasconcelos Oliveira Dantas

Coorientador(a): Prof. Me. Diogo Florêncio de Lima

CAMPINA GRANDE – PB

2024

C933u Cristovão, Arthur Lincoln da Paz.

Uso de técnicas de Inteligência Artificial para identificação de chamadas procedentes e improcedentes [manuscrito] / Arthur Lincoln da Paz Cristovão. - 2024.

54 f. : il. color.

Digitado.

Trabalho de Conclusão de Curso (Graduação em Ciência da computação) - Universidade Estadual da Paraíba, Centro de Ciências e Tecnologia, 2024.

"Orientação : Prof. Dra. Kézia de Vasconcelos Oliveira Dantas, Departamento de Computação - CCT".

1. Machine learning. 2. Classificação de chamadas. 3. Atendimento ao cliente. I. Título

21. ed. CDD 006.35

Elaborada por Lêda Cristina Diniz Andrade - CRB - 15/1032 BC

ARTHUR LINCOLN DA PAZ CRISTOVAO

USO DE TÉCNICAS DE INTELIGÊNCIA ARTIFICIAL PARA IDENTIFICAÇÃO DE CHAMADAS PROCEDENTES E IMPROCEDENTES

Monografia apresentado à Coordenação do Curso de Ciência da Computação da Universidade Estadual da Paraíba, como requisito parcial à obtenção do título de Bacharel em Computação

Aprovada em: 31/10/2024.

BANCA EXAMINADORA

Documento assinado eletronicamente por:

- **Diogo Florêncio de Lima** (***.660.428-**), em **22/11/2024 14:48:15** com chave **f3919f9ca8f911ef9e6306adb0a3afce**.
- **Kézia de Vasconcelos Oliveira Dantas** (***.714.244-**), em **22/11/2024 14:21:56** com chave **4610297ca8f611ef86a02618257239a1**.
- **Daniella Lima de Carvalho** (***.215.374-**), em **22/11/2024 14:48:16** com chave **f3c7b6a4a8f911ef83a32618257239a1**.
- **Sabrina de Figueirêdo Souto** (***.047.964-**), em **24/11/2024 07:59:21** com chave **29102328aa5311efbc421a1c3150b54b**.

Documento emitido pelo SUAP. Para comprovar sua autenticidade, faça a leitura do QrCode ao lado ou acesse https://suap.uepb.edu.br/comum/autenticar_documento/ e informe os dados a seguir.
Tipo de Documento: Folha de Aprovação do Projeto Final
Data da Emissão: 05/02/2025
Código de Autenticação: 196ff0

À minha família e amigos pelo apoio,
compreensão e amor incondicional.
DEDICO

AGRADECIMENTOS

A Deus, por minha vida, família e amigos, e por sempre ser meu alicerce nos momentos em que mais preciso. Agradeço por todas as bençãos que Ele vem proporcionando em toda minha jornada. Ele é minha fé, meu refúgio e minha fonte infinita de inspiração. Sou imensamente grato pelo amor incondicional, por todas as oportunidades que Ele tem me dado e por se fazer presente em cada momento da minha vida.

Agradeço a minha mãe Lucélia, meu maior exemplo de amor e resiliência. Sou grato por todo o amor, cuidado, comprometimento, apoio e companheirismo. Tudo o que sou hoje devo a você. És o melhor exemplo de cuidado que uma pessoa pode demonstrar.

Ao meu pai Edvan por sempre se fazer presente em minha vida, demonstrando amor, cumplicidade, cuidado e preocupação. Cada vivência ao seu lado foi essencial para percorrer essa trajetória.

Às minhas queridas avó Nereide e bisavó Terezinha que sempre estiveram ao meu lado em qualquer tipo de situação. Obrigado por tudo.

À minha querida madrinha Elânia por todos os ensinamentos e por ser um exemplo de pessoa que sempre esteve ao meu lado para me motivar, acompanhar e aconselhar, e a minha adorável prima Sophi por esse imenso e sincero amor.

É com muita admiração e respeito que venho agradecer a Kézia, minha orientadora de TCC, projeto e estágio. Sou extremamente grato por todas as oportunidades, bem como pelo tempo, orientação e paciência dedicados a mim. Obrigado por acreditar no meu potencial.

À minha Tia Lucinha por fazer um papel de mãe quando necessário, ao seu marido Jefferson por apoiar toda nossa família, e aos meus primos Luiz e João Lucas por todo companheirismo.

Aos meus avôs Socorro e Elpídio, minha Tia Rosário, também minhas primas Dandara, Jennifer e Valentina por todo apoio ao longo da minha vida.

Aos meus Tios Junior e Edgley, e a todos familiares por sempre estarem na torcida por mim. O apoio e presença de vocês significam muito para mim.

Venho agradecer aos meus amigos, tanto aqueles que estão comigo desde o início da minha vida, como também os que conheci durante o curso e em especial Philipe, Neto, Noberto, Luana e Marinho por termos construído uma verdadeira amizade.

À minha família de EJC por cada encontro e oração feitas a fim de renovar nosso compromisso com Deus.

RESUMO

No setor de energia elétrica, a eficiência e a qualidade do atendimento ao cliente são fundamentais para o sucesso de uma concessionária desse ramo. A identificação da legitimidade das chamadas de serviço, separando entre chamadas improcedentes e procedentes, é um desafio recorrente para as empresas que fornecem esse serviço. Diante desse cenário, este trabalho propõe o uso da inteligência artificial para desenvolver um modelo de *machine learning* capaz de identificar a procedência das chamadas recebidas pelas centrais de atendimento ao cliente da concessionária de energia, visando reduzir os custos operacionais causados por deslocamentos improcedentes. O desenvolvimento do modelo de classificação de chamadas envolveu técnicas de pré-processamento, treinamento e avaliação dos modelos de aprendizado de máquina. Os resultados mostraram que o modelo *WeightedEnsemble_L2* superou o desempenho de modelos base com a combinação de diversos algoritmos, fazendo a captura de padrões nos dados, reduzindo o *overfitting* e melhorando o desempenho do modelo para a previsão de novas chamadas. O modelo obteve uma acurácia de 89% e uma pontuação *AUC* de 0.855 para as chamadas improcedentes, indicando a alta capacidade de distinguir entre quais são as chamadas improcedentes e procedentes. A implementação deste modelo de aprendizado de máquina demonstrou ser uma solução viável para a aumentar a eficiência no atendimento aos clientes da empresa, reduzindo custos operacionais e melhorando a satisfação do cliente.

Palavras-chave: machine learning; classificação de chamadas; atendimento ao cliente; energia elétrica.

ABSTRACT

In the electricity sector, efficiency and quality of customer service are fundamental to the success of utility companies. Identifying the legitimacy of service calls and distinguishing between unfounded and well-founded calls is a recurring challenge for companies providing these services. In light of this, the present study proposes the use of artificial intelligence to develop a machine learning model capable of identifying the origin of calls received by the energy utility's customer service centers, with the aim of reducing operational costs caused by unfounded calls. The development of the call classification model involved data preprocessing techniques, as well as training and evaluating machine learning models. The results showed that the *WeightedEnsemble_L2* model outperformed the base models by combining various algorithms, capturing patterns in the data, reducing overfitting, and improving the model's performance in predicting new calls. The model achieved an accuracy of 89% and an *AUC* score of 0.855 for unfounded calls, demonstrating a high ability to distinguish between unfounded and well-founded calls. The implementation of this machine learning model proved to be a viable solution for increasing the efficiency of the company's customer service, reducing operational costs, and improving customer satisfaction.

Keywords: machine learning; call classification; customer service; electricity.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ANEEL	Agência Nacional de Energia Elétrica
AUC	Area Under the Curve
AutoML	Aprendizado de Máquina Automatizado
FP	False Positive
FPR	False Positive Rate
FN	False Negative
GDIS	Geração Distruibuída
IA	Inteligência Artificial
LightGBM	Light Gradient Boosting Machine
ML	Machine Learning
NN	Neural Network
PCA	Principal Component Analysis
ROC	Receiver Operating Characteristic
sklearn	Scikit-Learn
SVM	Support Vector Machine
TN	True Negative
TP	True Positive
TPR	True Positive Rate
t-SNE	t-Distrubuted Stochastic Neighbor Embedding
XGBoost	Extreme Gradiente Boost

LISTA DE EQUAÇÕES E FÓRMULAS

SUMÁRIO

1 INTRODUÇÃO

A interrupção no fornecimento de energia elétrica é um dos principais critérios utilizados pela Agência Nacional de Energia Elétrica (ANEEL) para avaliar a qualidade do serviço prestado pelas concessionárias de energia (ANEEL, 2020). Nesse contexto, encontra-se o setor de atendimento aos clientes, o qual é responsável por receber as chamadas de emergência, relacionadas a tais interrupções, e determinar a prioridade do atendimento, visando, dentre outros fatores, minimizar o tempo e os custos envolvidos. A determinação do roteiro de atendimento das ordens de serviço emergenciais é uma tarefa custosa, haja vista a natureza dinâmica das ocorrências emergenciais, cujo surgimento não pode ser previsto (SCHMITZ et al., 2016). Além disso, cerca de 25% dos serviços atendidos pelas equipes de campo das empresas distribuidoras de energia no país são de caráter improcedente ou reincidente. Este número poderia ser minimizado se o sistema de atendimento ao cliente fosse capaz de identificar e classificar tais tipos de chamadas como procedente ou improcedente com base no perfil e histórico de chamadas dos seus clientes.

Para realizar a classificação das chamadas, torna-se necessário o uso de técnicas de inteligência artificial para analisar o grande volume de dados advindos das reclamações e solicitações dos clientes. Em conjunto a isso, nos últimos anos, a inteligência artificial (IA) tem se mostrado como uma ótima solução para solucionar os problemas de classificação, por exemplo com a utilização de algoritmos de aprendizado de máquina, do inglês *Machine Learning* (*ML*).

Técnicas de *ML*, em particular, têm sido bastante utilizadas para automatizar e melhorar vários processos, desde a previsão de demanda até a classificação de chamadas em sistemas de energia elétrica (RUSSEL; NORVIG, 2016; GOODFELLOW; BENGIO; COURVILLE, 2016). A aplicação de algoritmos de classificação para encontrar padrões nos dados é um meio de melhorar a identificação prévia das chamadas improcedentes, levando a uma melhoria na eficiência operacional da empresa, reduzindo custo e aumentando a satisfação do cliente com o atendimento fornecido (ZHANG; CHEN, 2020).

Diversos algoritmos de *ML* foram avaliados para encontrar a melhor abordagem para solução deste problema, levando em consideração a precisão, processamento e a capacidade do modelo sobre novos dados (NGUYEN et al., 2019). Os resultados

demonstram que a utilização dessas técnicas traz melhorias no gerenciamento dos atendimentos. Em meio as técnicas de aprendizado de máquina, a técnica de *ensemble learning* se destaca por combinar os resultados de outros modelos para obtenção de um modelo final. O modelo final deste trabalho é obtido a partir da técnica do *ensemble*, mostrando um resultado melhor quando comparado com outros algoritmos de classificação.

Theissler et al. (2021) demonstraram que a utilização de técnicas de *ensemble* pode melhorar significativamente a precisão e eficácia dos modelos de *ML* aplicados à manutenção preditiva na indústria automotiva. Ao combinar modelos, o *ensemble* se mostrou eficaz na detecção precoce de falhas em componentes automotivos, aumentando a confiabilidade dos sistemas e reduzindo os custos com manutenções não planejadas.

Korhan et al. (2020) revisaram as técnicas de *ML* aplicadas para a manufatura na Indústria 4.0, enfatizando o papel do *ensemble* na otimização da gestão da saúde dos equipamentos industriais. Eles observaram que essas técnicas permitem uma análise mais abrangente e precisa, crucial para o monitoramento em tempo real e prevenção de falhas em linhas de produção.

Texeira et al. (2019) propuseram um classificador de procedência de chamadas usando um algoritmo de aprendizado de máquina supervisionado, denominado *Extreme Gradient Boosting* (*XGBoost*), o qual é baseado em árvores de decisão e determina a probabilidade de cada chamada resultar em um deslocamento improcedente. Ainda, uma técnica de Modelo Preditivo de Controle (MPC) é empregada para calcular o risco associado com o possível custo de não atender uma chamada, especialmente no caso desta ser procedente. Para validação da metodologia, foram usados dados extraídos do banco de dados do *call center* da CPFL Energia e uma acurácia de 84% foi obtida. Observa-se que os autores não fizeram uso de técnicas mais avançadas de IA para análise e classificação das chamadas, o que poderia aumentar a acurácia dos resultados obtidos por meio de uma avaliação mais aprofundada da descrição do problema relatado pelo cliente, levando em consideração a análise de características semânticas. Ainda, os autores não realizaram a análise de eventos relacionados a ocorrência de falhas no sistema elétrico. A metodologia descrita faz uso apenas de uma flag para indicar a interrupção

no fornecimento de energia. A partir da identificação e localização de falhas no sistema, seria possível determinar com uma maior precisão os possíveis clientes afetados e com isso, realizar uma classificação mais confiável das chamadas. Ainda, seria possível realizar um melhor direcionamento das equipes de campo, a fim de que os problemas na rede fossem solucionados de maneira mais eficiente.

1.1 Objetivos

Com o intuito de preencher as lacunas existentes na literatura, busca-se o desenvolvimento de um modelo de classificação para a identificação de chamadas improcedentes e procedentes. A correta distinção entre esses tipos de chamadas é crucial para otimizar os recursos operacionais e reduzir custos associados a deslocamentos desnecessários. Para alcançar o objetivo do trabalho a fim de desenvolver o modelo, serão aplicadas técnicas avançadas de *ML*, explorando diferentes algoritmos e abordagens de *ensemble*. O modelo será avaliado com base em métricas de performance, como acurácia, matriz de confusão e curva *AUC_ROC*, visando garantir que o modelo seja eficiente na classificação das chamadas, com foco na detecção de chamadas improcedentes, que representam maior impacto para a concessionárias.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção, são apresentados os fundamentos teóricos que agregam ao desenvolvimento deste trabalho. A fundamentação teórica aborda os conceitos e pesquisas relevantes que basearam a realização do trabalho.

2.1 Inteligência Artificial

A IA é uma área da ciência da computação que é direcionada ao estudo e desenvolvimento de sistemas capazes de realizar atividades normalmente atribuídas aos humanos e sua capacidade intelectual. Essas atividades incluem aprendizado, raciocínio, detecção de padrões, tomada de decisões e compressão de linguagem natural.

De acordo com Haugeland (1985), o objetivo da inteligência não é apenas simular o pensamento cognitivo humano, mas construir máquinas que realmente possuam uma forma de mente, envolvendo não apenas a capacidade de processamento de informações de maneira avançada, mas também a capacidade de entendimento, aprendizado e tomada de decisões de uma forma semelhante à atribuída à mente humana.

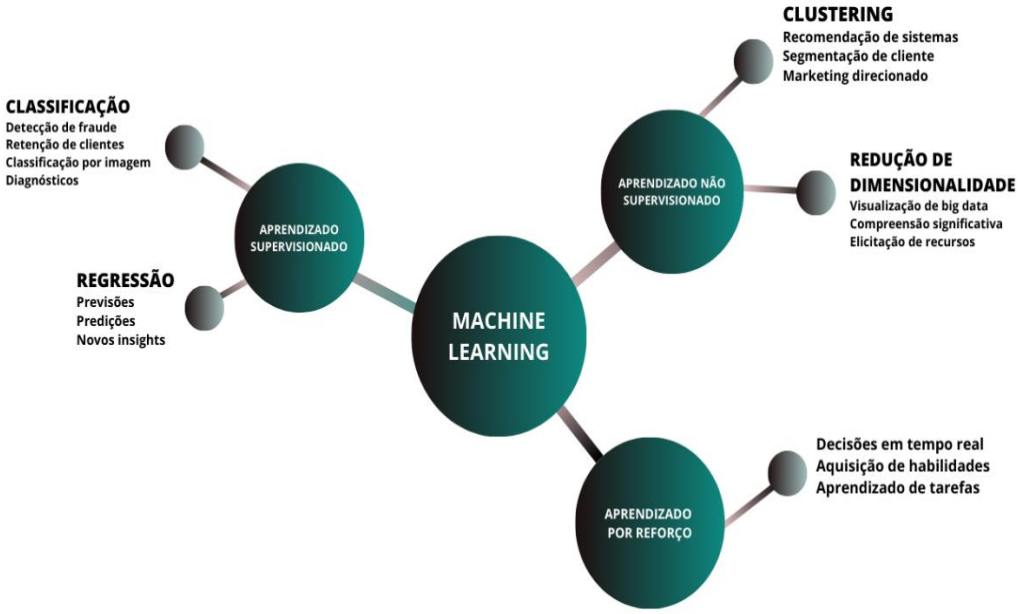
Em termos de aplicação prática, a inteligência artificial já demonstra impacto significativo em setores como finanças e saúde. Segundo Wang et al. (2018), os sistemas de IA são cada vez mais utilizados em instituições financeiras para a otimização de ações e previsões de mercado, melhorando a precisão das decisões tomadas. Essas tendências mostram o potencial da inteligência artificial, destacando a necessidade de sua utilização, garantindo que o desenvolvimento de sistemas de IA sejam benéficos em cada setor utilizado.

2.2 Aprendizado de Máquina

O aprendizado de máquina é um subcampo da inteligência artificial que compreende abordagens e técnicas para instruir sistemas a aprenderem com a experiência do aprendizado.

Existem três principais abordagens para o aprendizado de máquina, cujas abordagens se diferem pelo tipo de dados que utilizam para o aprendizado. Dependendo do objetivo e dados, as técnicas de aprendizado de máquina podem ser classificadas nos seguintes tipos: aprendizado supervisionado, aprendizado não supervisionado e aprendizagem por reforço. Os tipos de aprendizados e algumas aplicações de cada tipo estão apresentadas na Figura 1.

Figura 1 – Tipos de aprendizado de máquina e algumas aplicações



Fonte: Elaborado pelo autor, 2024.

Como apresentado na Figura 1, as duas principais aplicações relacionadas ao aprendizado supervisionado, utilizando dados rotulados, são classificação e regressão.

2.2.1 Aprendizado Supervisionado

O aprendizado supervisionado é um método onde o modelo é treinado utilizando um conjunto de dados rotulados, onde cada exemplo possui uma entrada e uma saída esperada, com o objetivo de aprender a realizar o mapeamento das entradas para as saídas corretas (ALPAYDIN, 2014). Esse método é bastante utilizado no contexto de classificação, na qual o modelo deve categorizar entradas em classes

pré-definidas, e de regressão, onde o objetivo do aprendizado é obter a previsão de valores contínuos.

Existem algoritmos já comuns para a resolução de problemas que envolvem classificação, alguns exemplos são as árvores de decisão que são estruturas de árvore que dividem os dados em subconjuntos com base em questões sobre as principais características de cada dado, com o intuito de se ter uma interpretação mais simples (QUINLAN, 1986). *Support Vector Machine* (*SVM*) é um método de encontrar a melhor linha de separação entre classes maximizando a margem entre os pontos mais próximos de diferentes classes (CORTES; VAPNIK, 1995).

Alguns dos algoritmos de regressão são, o de regressão linear que é um método estatístico que modela a relação entre uma variável dependente contínua e uma ou mais variáveis dependentes (MURPHY, 2012) e a regressão logística é utilizada para modelar a probabilidade de ocorrência de um evento utilizando uma função logística, comumente aplicada em problemas de classificação binária (HOSMER Jr; LEMESHOW, 2004).

2.2.2 Aprendizado Não Supervisionado

No aprendizado não supervisionado, é fornecido dados de entrada ao algoritmo sem nenhum desses sendo rotulado, com intuito de identificar padrões ou estrutura para saída de cada um desses dados. Ao contrário do aprendizado supervisionado em que um conjunto de dados rotulados é utilizado para o treinamento, na aprendizagem não supervisionada, o modelo tem como finalidade identificar padrões nos dados.

Alguns tipos de técnicas de aprendizado não supervisionado são:

- ***Clustering*:** É uma maneira de organizar os dados via agrupamento, sendo realizado a partir da maior similaridade existente entre os dados de um conjunto (JAIN; MURTY; FLYNN, 1999).
- **Regras de associação:** Tem o objetivo de encontrar relacionamentos ou padrões frequentes entre conjunto de dados (AGRAWAL; IMIELIŃSKI; SWAMI, 1993).

- **Redução de dimensionalidade:** Tem o objetivo de reduzir o número de variáveis aleatórias que serão inseridas em modelo para treino. Uma das principais técnicas de redução de dimensionalidade é a *Principal Component Analysis* (*PCA*) (JOLLIFE, 2002).

2.2.3 Aprendizado Por Reforço

O aprendizado por reforço realiza o treinamento do software a fim de tomar decisões em busca dos melhores resultados e desempenho através da tentativa e erro (SUTTON; BARTO, 2018). Ele é orientado por uma meta de otimização contínua, sendo diferente do aprendizado supervisionado que treina um conjunto de dados rotulado fixos, e do aprendizado não supervisionado, onde o objetivo é a identificação de padrões nos dados não rotulados.

Esse tipo de aprendizado é aplicado em situações onde as interações com o ambiente são dinâmicas e o resultados das ações do agente levam a outras experiências até o seu desempenho ser otimizado. Alguns exemplos da aplicação desse tipo de aprendizagem são jogos e aplicações robóticas.

2.3 Ensemble Learning

Ensemble é uma técnica de aprendizado de máquina que combina múltiplos modelos, onde são treinados para resolução de um mesmo problema e suas previsões são combinadas para obter um resultado final mais preciso e robusto (YANG; LV; CHEN, 2022). O principal objetivo do *ensemble learning* é melhorar a performance do modelo final, reduzindo o *overfitting* e o *underfitting*. As principais técnicas de *ensemble learning* são *Bagging*, *Boosting* e *Stacking*, cada uma com suas próprias características.

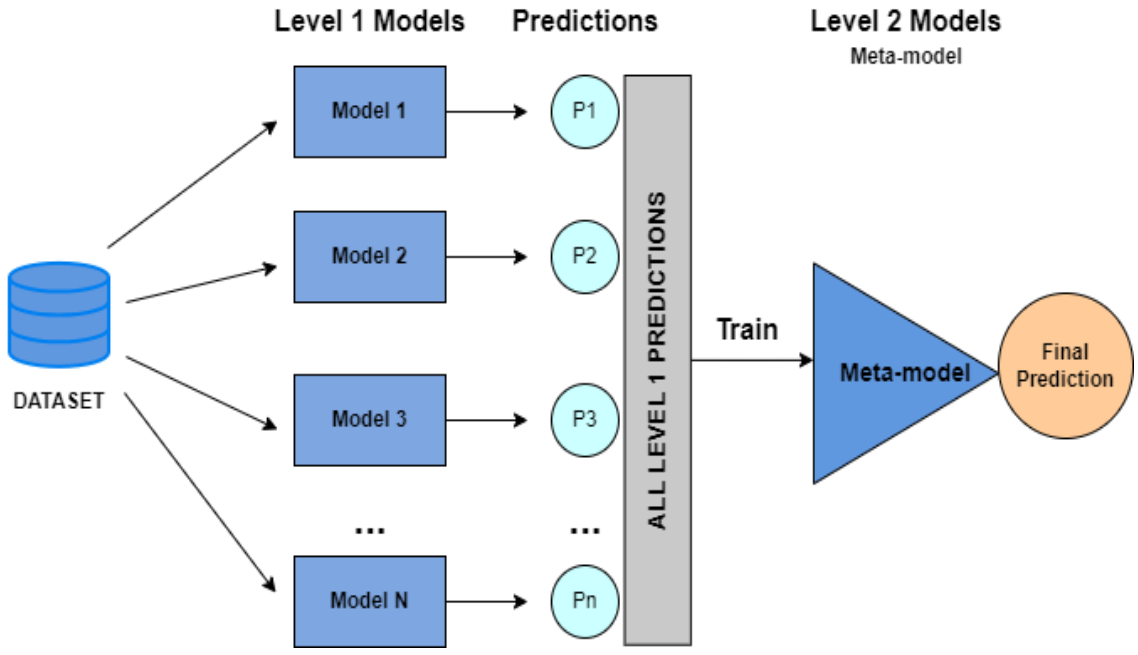
Bagging uma técnica que treina vários modelos independentes usando diferentes subconjuntos do conjunto de dados de treinamento, criados a partir de amostragem com reposição (*bootstrap*). As previsões dos modelos são combinadas por média, quando o cenário é de regressão ou por votação majoritária, quando for para classificação (BREIMAN, 1996).

Boosting é uma técnica sequencial onde os modelos são treinados de modo iterativo, e cada modelo foca em corrigir os erros dos modelos treinados anteriormente. As previsões são combinadas de forma ponderada para aumentar a precisão final (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

2.4 Stacking

É uma técnica do ensemble learning diferente do *bagging* e do *boosting* que combinam previsões de múltiplos modelos através de média ou votação. No método de *stacking*, é utilizado meta-modelo para aprender a melhor maneira de combinar as previsões feitas (WOLPERT, 1992). A Figura 2 apresenta como se dá a metodologia do *stacking*.

Figura 2 – Metodologia do *stacking*



Fonte: Elaborado pelo autor, 2024.

No primeiro nível são treinados os modelos base nos dados de treinamento, com a previsões resultantes dessa etapa sendo utilizadas posteriormente para geração novas entras para o meta-modelo. Após isso, no segundo nível o meta-modelo é treinado nas previsões dos modelos base, aprendendo a melhor maneira de combinar as previsões obtidas na primeira etapa para a previsão final.

2.5 Modelos de *ML*

Nesta seção, são apresentados os modelos de *ML* que foram utilizados como base para a desenvolvimento do trabalho. Cada um dos algoritmos descritos possui características específicas que os tornam adequados para as tarefas que serão realizadas.

2.5.1 XGBoost

O *XGBoost* é uma implementação avançada e otimizada do algoritmo de *boosting* de gradiente amplamente utilizada para resolver problemas de classificação e regressão, sendo projetado para ser altamente eficiente em termos de tempo e recursos computacionais (CHEN; GUESTRIN, 2016).

O diferencial do *XGBoost* em relação a outras implementações de *boosting* é sua eficiência e capacidade de regularização, o que melhora o desempenho e robustez do modelo. Ele incorpora técnicas avançadas, como a regularização L1 e L2, que ajudam a evitar o *overfitting,* além de ser otimizado para oferecer alta eficiência computacional em termos de tempo de execução e uso de memória. O *XGBoost* permite paralelização durante o treinamento, o que torna especialmente útil em grandes conjuntos de dados, além de ser facilmente distribuído em múltiplas máquinas, tornando-o escalável para projetos de larga escala.

Uma das grandes vantagens do *XGBoost* é a capacidade de ajustar os pesos das observações mal preditas em cada iteração, aumentando o impacto das previsões difíceis e, assim, melhorando a performance do algoritmo. A flexibilidade do *XGBoost* também é um ponto de destaque, permitindo o ajuste de uma série de hiperparâmetros para otimizar o desempenho.

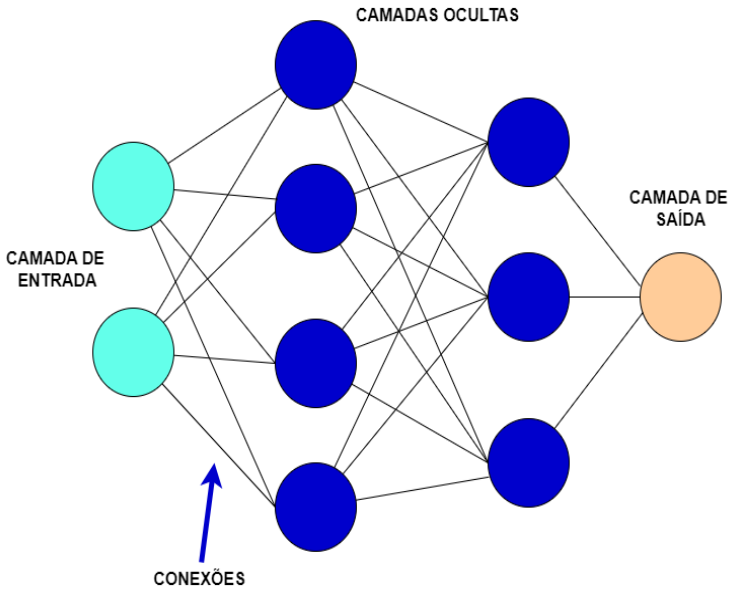
2.5.2 Neural Network

As Redes Neurais (*Neural Networks – NN*), são modelos computacionais inspirados no funcionamento do cérebro humano, capazes de aprender padrões complexos entre múltiplas camadas de neurônios artificiais. Elas se tornaram uma peça fundamental na IA e *ML*, especialmente em problemas que envolvem

processamento de imagem, reconhecimento de fala, processamento de linguagem natural e previsão de séries temporais (HAYKIN, 2009).

A arquitetura de uma *NN* inclui três tipos de camadas principais: a camada de entrada, que recebe os dados; as camadas ocultas, que processam a informações por meio de funções de ativação; e a camada de saída, que fornece o resultado final da *NN*, como uma previsão de classificação ou valor numérico. As conexões entre os neurônios de uma camada para outra são ajustadas por pesos, que são o elemento essencial do aprendizado. Na Figura 3, mostra o funcionamento das camadas principais da arquitetura de uma *NN*.

Figura 3 – Arquitetura de uma *neural network*



Fonte: Elaborado pelo autor, 2024.

O treinamento de uma *NN* envolve o ajuste contínuo dos pesos, com o intuito de minimizar o erro entre as previsões e os resultados reais, usando algoritmos como *backpropagation* e gradiente descendente. O processo é realizado em várias etapas chamadas épocas, onde os dados são analisados. Funções de custo, como erro quadrático médio ou a entropia cruzada, são utilizadas para medir a precisão das previsões e orientar os ajustes dos pesos. As *NN* utilizam funções de ativação para introduzir não linearidade, permitindo a captura de padrões complexos. Algumas funções comuns são a *ReLU*, *sigmoid* e a *tanh*. A *ReLU* é popular por sua simplicidade e eficácia, enquanto as outras mapeiam valores de entrada em intervalos específicos.

2.5.3 LightGBM

Light Gradient Boosting Machine (*LightGBM*) é um algoritmo que utiliza a técnica de *boosting* para melhorar a precisão nas tarefas na aprendizagem supervisionada. Ele é especialmente eficiente em grandes volumes de dados e alta dimensionalidade, apresentando um desempenho superior em comparação com outros algoritmos de *boosting* (KE et al., 2017).

Uma das características do *LightGBM* é o uso de histogramas, agrupando os valores de entrada em *bins,* reduzindo o tempo de processamento e a complexidade computacional, permitindo um treinamento mais rápido das árvores. O *LightGBM* também emprega uma técnica chamada *Gradient- Base One-Side Sampling* (*GOSS*), que seleciona amostra de dados com maior gradiente, mantendo as informações mais relevantes e acelerando a convergência do modelo.

Além disso, o *LightGBM* constrói suas árvores de forma *leaf-wise.* Em vez de expandir todas as camadas da árvore ao mesmo tempo, ele escolhe a folha com o maior ganho de informação em cada iteração. Essa abordagem resulta em árvores mais profundas que capturam melhor a complexidade dos dados, embora possa acarretar em *overfitting*.

2.5.4 Random Forest

Random Forest é um algoritmo que combina várias árvores de decisão, com o intuito de aprimorar a precisão das previsões. Esse algoritmo se destaca por ser um método de *ensemble learning*, visando minimizar problemas como variância e o *overfitting,* comuns em modelos que usam apenas uma única árvore.

O algoritmo funciona por meio do *bagging,* criando múltiplas amostras aleatórias dos dados de treinamento com reposição. Para cada amostra, é treinada uma árvore de decisão, utilizando um subconjunto aleatório de características em cada nó. Essa abordagem ajuda a diversificar as árvores, tornando o modelo mais robusto e menos suscetível a ruídos nos dados (BREIMAN, 1996). Para realizar as previsões, o algoritmo agrega os resultados de todas as árvores criadas. Em problemas de classificação, é utilizado a média das previsões. Com a combinação dos resultados, o algoritmo tende a ter uma melhor generalização para novos dados.

Uma das vantagens do *Random Forest* é a capacidade de lidar com dados em alta dimensionalidade e variáveis categóricas, o que o torna útil em várias aplicações. O modelo também fornece uma medida de importância de variáveis, que ajuda a identificar quais as características que mais influenciam nas previsões.

2.6 PCA

A *PCA* é uma técnica estatística bastante utilizada na redução de dimensionalidade de conjunto de dados, transformando um conjunto de variáveis em outro conjunto de variáveis da mesma dimensão denominadas de componentes principais (JOLLIFFE; CADIMA, 2016). Essa técnica é importante na etapa de pré-processamento dos dados, visualização e aumento da eficiência dos algoritmos de aprendizado de máquina.

No cenário de pré-processamento de dados, a *PCA* reduz o número de variáveis de um conjunto de dados, mantendo a maior quantidade de informação possível, simplificando os modelos e diminuindo o tempo de processamento. Para visualização, projeta os dados de alta dimensionalidade em duas ou três dimensões, facilitando a detecção de padrões. Na etapa de pré-processamento, remove o ruído dos dados, contribuindo para a eficiência dos modelos.

O objetivo principal da *PCA* é identificar a direção em que os dados variam mais. Esses vetores de direção são chamados de componentes principais. O primeiro componente principal explica a maior variância dos dados, o segundo explica a segunda maior e assim por diante.

2.7 T-SNE

O *t-Distrubuted Stochastic Neighbor Embedding* (*t-SNE*) é uma técnica de redução de dimensionalidade amplamente utilizada para visualização de dados em alta dimensão (VAN DER MAATEN; HINTON, 2008). A principal função do *t-SNE* é mapear dados de alta dimensão para um espaço de menor dimensão, preservando a estrutura das vizinhanças locais dos pontos, facilitando a visualização de

agrupamentos, padrões e relações entre os dados, que seriam difíceis de identificar em um espaço com muitas dimensões.

O *t-SNE* funciona calculando as probabilidades de similaridade entre os pontos no espaço de alta dimensão e tenta garantir que essas mesmas probabilidades sejam preservadas no espaço de baixa dimensão. Ele utiliza uma distribuição *t* de *Student* para calcular as distâncias no espaço reduzido, o que ajuda a evitar que pontos distantes sejam erroneamente agrupados.

O primeiro passo do *t-SNE* é calcular a similaridade entre dos pontos x_i e x_j, no espaço de alta dimensão usando uma distribuição gaussiana em cada ponto. A probabilidade condicional $p_{j|i}$ de x_j ser o vizinho de x_i é dada pela fórmula a seguir:

$$p_{j|i} = \frac{exp\left(-\frac{\left||xi - xj|\right|^2}{2\sigma_i^2}\right)}{\sum k \neq i \; exp\left(-\frac{\left||xi - xk|\right|^2}{2\sigma_i^2}\right)} \quad (1)$$

onde $\left||xi - xj|\right|$ é a distância euclidiana entre x_i e x_j, e σ_i é o desvio padrão associado ao ponto x_i, que controla a distribuição ao redor de x_i.

As probabilidades simétricas são definidas pela fórmula:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N} \quad (2)$$

onde *N* é o número total de pontos.

No espaço de baixa dimensionalidade, em vez de uma distribuição gaussiana, o *t-SNE* utiliza a distribuição de t de *Student* de uma única variável, que tem caudas mais longas. A probabilidade q_{ij} de dois pontos y_i e y_j no espaço de baixa dimensão serem vizinhos é dado pela seguinte fórmula:

$$q_{ij} = \frac{(1 + \left||yi - yj|\right|^2)^{-1}}{\sum k \neq 1 \; (1 + \left||yk - yi|\right|^2)^{-1}} \quad (3)$$

Com y_i e y_j sendo as projeções de x_i e x_j no espaço de baixa dimensionalidade.

O *t-SNE* minimiza a divergência de Kullback-Leibler entre as distribuições p_{ij} e q_{ij}, que mede o quanto uma distribuição difere da outra. A função de custo que o *t-SNE* minimiza é dada pela fórmula:

$$C = \sum_{i \neq j} pij \log \frac{p_{ij}}{qij} \tag{4}$$

Minimizar essa divergência é garantir com que as relações locais no espaço de alta dimensionalidade sejam preservados no espaço de baixa dimensionalidade.

Para minimizar a função de custo C, o *t-SNE* utiliza um método de gradiente descendente. O gradiente descendente da função de custo com respeito a cada y_i é calculado:

$$\frac{\partial C}{\partial yi} = 4 \sum_{J} (pij - qij)(yi - yj)(1 + \|yi - yj\|^2)^{-1} \tag{5}$$

Esse gradiente é utilizado para atualizar os valores de y_i iterativamente, de modo a minimizar C e ajustar as projeções no espaço de baixa dimensionalidade.

2.8 Métricas de Avaliação

A avaliação de modelos de aprendizado de máquina é uma etapa essencial para determinar a eficácia e precisão dos modelos. Entre as métricas utilizadas, a matriz de confusão, a acurácia e a curva *ROC* são bastante utilizadas por conta da sua simplicidade e capacidade de fornecer uma avaliação abrangente sobre o desempenho dos modelos.

2.8.1 Matriz de Confusão

A matriz de confusão é uma métrica fundamental para avaliação dos modelos de classificação, oferecendo uma abordagem clara sobre a performance do modelo ao fazer a descrição das previsões realizadas pelo modelo e os valores reais. Dessa forma, permite identificar não só o número de previsões corretas e incorretas, mas

também o tipo de erro cometido, podendo ser um falso positivo (FP) ou falso negativo (FN).

A matriz é uma tabela $n\ X\ n$ (onde n é o número de classes), resumindo as previsões do modelo de classificação. Para uma situação de classificação binária, a matriz possui a seguinte estrutura:

Tabela 1 – Representação da matriz de confusão

Confusion Matrix		Predict class	
		True	False
Original class	True	*TP*	*FN*
	False	*FP*	*TN*

Fonte: Elaborada pelo autor, 2024.

- *True Positives* (*TP*) são os verdadeiros positivos: casos em que o modelo previu corretamente a classe positiva.
- *True Negatives* (*TN*) são os verdadeiros negativos: casos em que o modelo previu corretamente a classe negativa.
- *False Positives* (*FP*) são os falsos positivos: caos em que o modelo previu incorretamente a classe positiva.
- *False Negatives* (*FN*) são os falsos negativos: casos em que o modelo previu incorretamente a classe negativa.

A utilização da matriz de confusão pode auxiliar na comparação de desempenho dos modelos de classificação, contribuindo na escolha do modelo mais adequada para determinada aplicação. Ao analisar a matriz de confusão, é possível identificar padrões de erro e realizar ajustes no modelo para maximizar sua performance.

2.8.2 Acurácia

A acurácia é uma das métricas mais simples e intuitivas para avaliar modelos de classificação, avaliando o percentual de acertos. Ela representa a proporção de previsões corretas feitas pelo modelo em relação ao número total de previsões. É

especialmente útil no cenário em que as classes estão balanceadas, fornecendo uma medida sobre a eficácia do modelo (POWERS, 2011). Para a obtenção do valor da acurácia é dada a partir de:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

sendo obtida a partir da razão de todas as previsões corretas (tanto positivas quanto negativas) em relação ao total de casos.

2.8.3 Curva ROC e AUC

A Curva *Receiver Operating Characteristic* (*ROC*) e *Area Under the Curve* (*AUC*) são meios de visualização e métricas que fornecem uma visão detalhada do desempenho de um modelo de classificação binária. A curva *ROC* traça o *True Positive Rate* (*TPR*) contra o *False Positive Rate* (*FPR*) em diferentes limiares de decisão, permitindo a visualização do balanço entre sensibilidade e especificidade (FAWCETT, 2006). A *AUC* é útil em problemas de classificação desbalanceada, fornecendo uma avaliação que não depende dos limiares de decisão, com valor resultante variando de 0 a 1. Quanto maior o valor obtido para a *AUC*, melhor a capacidade de distinção do modelo.

A curva *ROC* é gerada ao variar o limiar de decisão (*thresholds*) de um modelo. O limiar de decisão é um valor que determina a partir de qual ponto a previsão do modelo é considerada positiva ou negativa. O ajuste do limiar para um valor mais alto ou mais baixo pode aumentar sensibilidade ou a especificidade do modelo, dependendo das necessidades do modelo. Por exemplo, um limiar de 0.5 pode significar que se a probabilidade prevista de um evento for maior ou igual a 0.5, o modelo classifica o evento como positivo.

Para obtermos o *TPR* ou sensibilidade, calculamos a razão de *TP* em relação ao total de positivos reais. A sensibilidade é essencial quando o objetivo é reduzir o número de *FN*, sendo definida por:

$$TPR\ (Recall) = \frac{TP}{TP + FN} \quad (7)$$

Já para o cálculo do *FPR* utilizamos a seguinte fórmula:

$$FPR = \frac{FP}{FP + TN} \tag{8}$$

medindo a proporção de *FP* (previsões incorretas de resultados positivos) em relação ao total de negativos. O cálculo do *FPR* é importante para minimizar o número de *FP*.

2.9 Tecnologias Utilizadas

A implementação de algoritmos de aprendizado de máquina e a análise de dados necessitam do uso de diversas tecnologias e bibliotecas. Cada uma das tecnologias que serão abordadas a seguir desempenham um papel fundamental no fluxo de trabalho de ciência de dados, desde a coleta e preparação de dados até a modelagem, visualização e avaliação de modelos.

Figura 4 – Tecnologias utilizadas

Fonte: Elaborado pelo autor, 2024.

2.9.1 Python

Python é uma linguagem de programação de alto nível muito utilizada em desenvolvimento de software, ciência dados, aprendizado de máquina e aplicações *Web*, conhecida por sua sintaxe simples, o que facilita o desenvolvimento rápido de aplicações complexas (LUTZ, 2013). Além disso, a facilidade de integração do *Python*

com quaisquer tipos de sistema agiliza o desenvolvimento e aumenta a versatilidade (VAN ROSSUM; DRAKE, 2003).

Os cientistas de dados utilizam as bibliotecas *Python* de para treinar modelos de *ML* e criar classificadores que categorizam dados com precisão. Para a manipulação e análise de dados, bibliotecas como *Pandas* e *NumPy* são fundamentais, pois oferecem estrutura de dados eficientes e ferramentas para processamento de big data (MCKINNEY, 2010; OLIPHANT, 2006).

A simplicidade e legibilidade do *Python* permite aos desenvolvedores escrevem um código que é fácil de ser lido e manter. Isso é fundamental em projetos de IA e ciência de dados, onde a complexidade dos algoritmos pode tornar o código difícil de gerenciar (LUTZ, 2013).

2.9.2 Pandas

Pandas é uma biblioteca de software de código aberto para o *Python* voltada para manipulação e análise de dados. Ela oferece estruturas de dados rápidas e efetivas, como os *DataFrames*, que permitem a manipulação de grandes volumes de dados.

É amplamente utilizada para limpeza, tratamento e transformação dos dados, proporcionando operações como agrupamento, filtragem e agregação. Suas funcionalidades permitem realizar análises exploratórias de dados de modo eficiente e essencial para preparação dos dados (MCKINNEY, 2010).

2.9.3 Numpy

A biblioteca *NumPy* é direcionada para a computação científica, permitindo o entendimento e a resolução de problemas complexos (OLIPHANT, 2006). Ela oferece suporte para matrizes e *arrays* multidimensionais, junto a uma gama de função matemáticas de alto nível para operações rápidas sobre esses *arrays*.

NumPy é a base de outras bibliotecas para ciência de dados e aprendizado de máquina, fornecendo a estrutura necessária para cálculos eficientes e manipulação

de grandes conjuntos de dados. A integração com outras bibliotecas como *Pandas* e *Scikit-Learn* a torna essencial para tarefas de análise e processamento de dados.

2.9.4 Matplotlib e Plotly

Matplotlib e *Plotly* são duas das mais diversas bibliotecas para a visualização de dados em *Python*, cada uma com suas vantagens e casos de uso específicos. Ambas são bastante utilizadas para ciência de dados para a criação de visualizações que ajudam a entender os insights dos dados (HUNTER, 2007; PLOTLY TECHNOLOGIES INC., 2015).

O *Matplotlib* é uma biblioteca de plotagem 2D que oferece uma interface simples e versátil, oferecendo flexibilidade e uma grande de variedade de gráficos. Já o *Plotly* oferece suporte a gráficos interativos e publicações baseadas na *web*, permitindo aos usuários criarem visualizações interativas que podem ser exploradas de forma dinâmica.

2.9.5 Scikit-Learn

A biblioteca *Scikit-Learn* (sklearn) é uma das ferramentas de aprendizado de máquina mais populares, oferecendo um conjunto abrangente de recursos para modelagem estatística, que inclui classificação, regressão, *clustering* e redução de dimensionalidade. Desenvolvida para fornecer implementações eficientes e reutilizáveis de algoritmos de aprendizado de máquina, o *sklearn* também conta com ferramentas robustas para a avaliação e validação de modelos (PEDREGOSA et al., 2011).

2.9.6 AutoGLuon

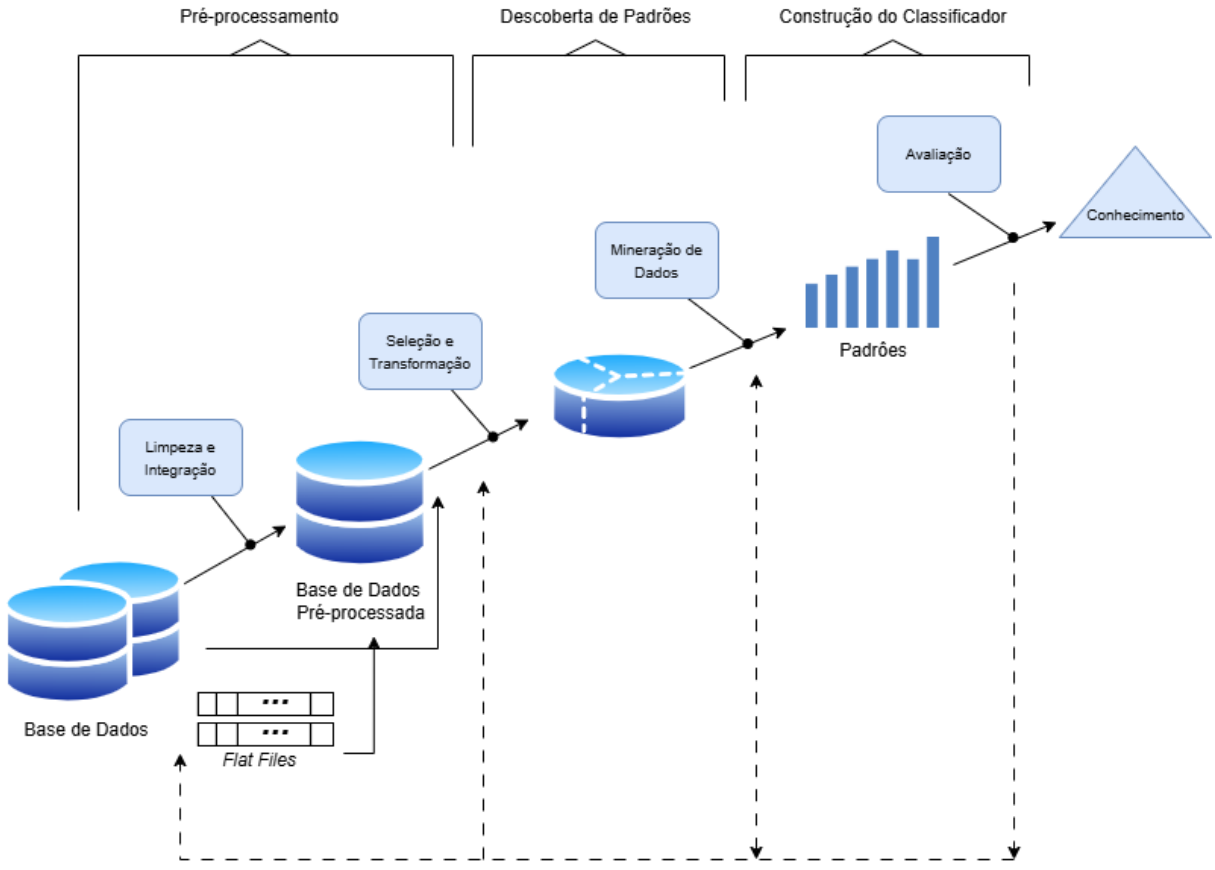
O *AutoGluon* é uma biblioteca de aprendizado máquina automatizado (AutoML) projetada para facilitar a criação de modelos preditivos de alta qualidade com o mínimo de codificação (ERICKSON et al., 2020). Ela é flexível e eficaz, pois permite a automatização de tarefas como seleção de modelos, ajuste de hiperparâmetros e combinação de previsões *(ensemble learning*).

AutoGluon é particularmente útil para desenvolvedores maximizarem a performance dos modelos sem ter que passar por cada etapa do processo de modelagem.

3 METODOLOGIA

Os procedimentos metodológicos deste trabalho consistem em várias etapas fundamentais para o desenvolvimento de modelos de aprendizado de máquina. As etapas incluem desde pré-processamento dos dados, descoberta de padrões até a avaliação dos resultados. Na Figura 5 é apresentado o fluxograma da metodologia utilizada para desenvolvimento do trabalho destacando cada uma das etapas mencionadas.

Figura 5 – Fluxograma da metodologia utilizada



Fonte: Elaborada pelo autor, 2024.

Para a etapa de pré-processamento dos dados foi realizada limpeza e integração com a utilização das bibliotecas *Pandas* e *NumPy*, a remoção de valores ausentes e correção de inconsistências, onde o objetivo é fazer com a base de dados fique preparada para as etapas posteriores. Além disso, feita a seleção e transformação com *sklearn*, separando as características mais importantes para a realização do trabalho e a transformação para tornar os dados apropriados para a

modelagem, onde pode ser feita a normalização, padronização e codificação das variáveis categóricas.

Após o pré-processamento, e etapa de descoberta de padrões foi realizada a partir da mineração dos dados, onde é aplicado algoritmo de aprendizado de máquina para identificar padrões, relações e *insights* nos dados. Nessa etapa, foi utilizado o *sklearn* e o *AutoGluon* para aplicação das técnicas de *ML* e automação do processo de modelagem.

Para a avaliação dos resultados, foi feita aplicação de métricas para avaliar a performance dos modelos. Isso inclui a geração de gráfico, como curvas *ROC* e matriz de confusão para visualizar a eficácia do modelo.

A seguir, será apresentado a base de dados utilizada no trabalho, com as informações e os campos.

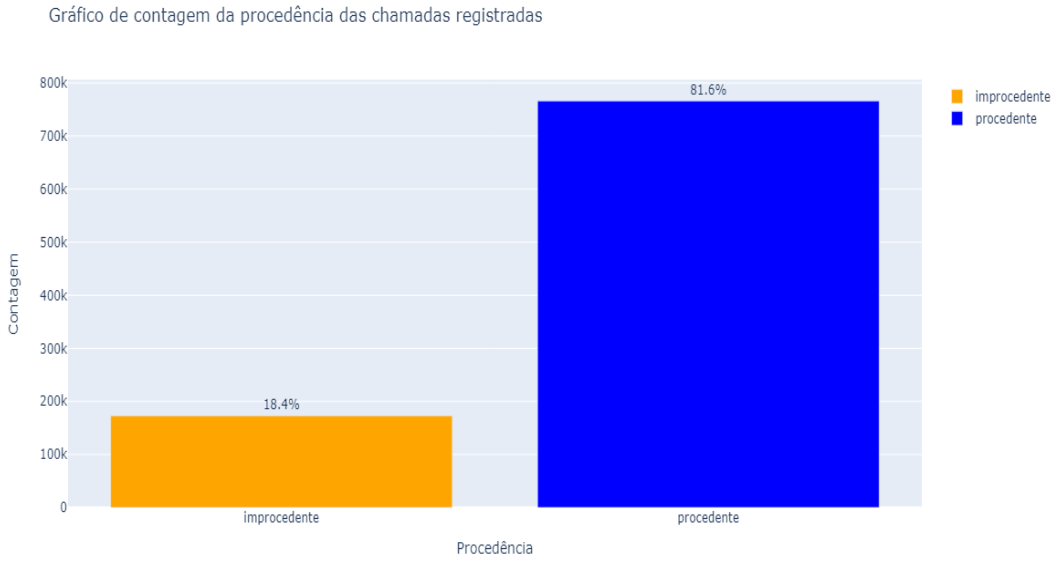
3.1 Base de Dados do GDIS

Para a realização deste trabalho foi utilizado da base de Geração Distribuída (GDIS) fornecida pela concessionária de energia, os dados são referentes ao período entre janeiro e maio de 2023, contendo um total de 938.561 mil chamadas registradas durante esse intervalo de tempo. GDIS é o sistema que registra todas as solicitações de ordens de serviço dos clientes.

As chamadas são classificadas em procedentes e improcedentes, são consideradas procedentes quando a solicitação de um serviço é confirmada como necessária no momento do atendimento. As improcedentes são aquelas que o cliente faz a requisição de um serviço e é necessário outro tipo de atividade para solucionar o problema da chamada.

Nos registros da base de dados GDIS, foram identificadas 169.681 chamadas improcedentes e 768.880 procedentes, numa proporção de 18.4% e 81.6%, respectivamente. Na Figura 6 é ilustrada a distribuição dessas chamadas.

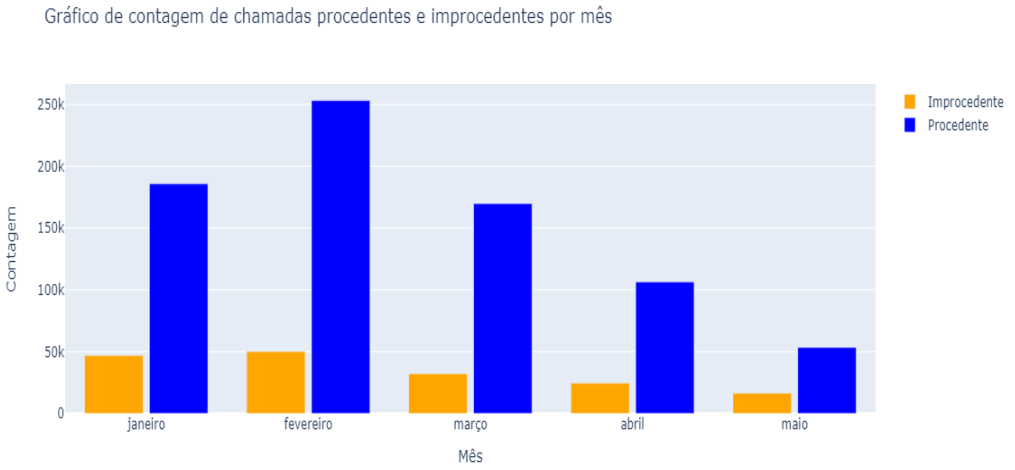
Figura 6 – Gráfico de contagem da procedência das chamadas registradas



Fonte: Elaborado pelo autor, 2024.

O gráfico na Figura 7 mostra a distribuição de chamadas nos meses em que as chamadas foram registradas, sendo possível visualizar que o número de registros foram caindo após o mês de fevereiro, mas mantendo sempre a mesma proporção na distribuição de chamadas improcedentes e procedentes.

Figura 7 – Gráfico de contagem da procedência de chamadas por mês



Fonte: Elaborado pelo autor, 2024.

A base de dados contém uma variedade de informações relacionadas às chamadas registradas no sistema de atendimento da concessionária de energia. Entre os dados mais relevantes, estão as informações pessoais dos clientes, como cadastro de pessoa física (CPF) ou cadastro nacional de pessoa jurídica (CNPJ), número de telefone e endereço. Também são armazenados dados de geolocalização, indicando o local de onde a chamada foi realizada, além dos registros do dia e hora em que o atendimento foi solicitado e concluído. Informações operacionais, como a causa e o fechamento da ocorrência, além do tipo de serviço e reincidência de chamadas, também estão presentes, fornecendo um panorama detalhado do atendimento e da sua resolução. Essas informações são essenciais para o desenvolvimento dos modelos preditivos que buscam diferenciam as chamadas improcedentes das procedentes.

Com a visão geral dos dados disponíveis, na próxima seção será abordado como foi realizado o trabalho da etapa de pré-processamento dos dados, explanando detalhadamente os processos de preparação e transformação dos dados para que eles possam ser utilizados de forma eficaz pelos algoritmos de IA.

3.2 Pré-processamento de Dados

Esta etapa envolve a limpeza e transformação dos dados, eliminando informações inconsistentes, dados ausentes ou anômalos, preparando todos os elementos necessários da base de dados para os modelos.

O primeiro passo foi realizar um mapeamento detalhado dos dados, com o objetivo de identificar possíveis inconsistências, erros ou valores ausentes. Durante essa análise, foram detectados problemas nos campos de datas de atendimento das chamadas, que estavam armazenados num formato inadequado, utilizando notação científica, o que comprometeria a qualidade das previsões. Para o ajuste, foi necessário realizar a conversão dos valores para o formato adequado, utilizando um padrão de data e hora no formato de 24 horas. Esse processo garantiu a consistência e integridade dos dados temporais, essenciais para uma análise precisa e confiável nas etapas seguintes.

Além da remoção das inconsistências nos campos de datas, foi necessário lidar com valores ausentes em outras *features.* Para o campo que representa a causa da ocorrência, o valor mais frequente foi utilizado para preencher os dados faltantes, assegurando que a feature mantenha um comportamento desejado.

Dentre os campos ajustados, a matrícula do consumidor, que é um identificador essencial para associar as ocorrências aos clientes, sendo uma informação bastante importante no processo de análise, quando ausente, foi preenchida com zero para sinalizar que a *feature* não estava preenchida. Já o campo de identificação das normas da ANEEL foi preenchido com o valor mais frequente, mantendo a coerência dos dados.

Com o preenchimento dos dados ausentes e a padronização de campos críticos, asseguramos que os dados estejam consistentes e prontos para alimentar os modelos de *ML*. Essas correções são fundamentais para garantir a qualidade dos dados, evitando que falhas nos dados comprometam os resultados finais do trabalho.

Com os dados preparados, na próxima seção será abordado a etapa de descoberta de padrões, aplicando técnicas para identificar relações e padrões nos dados.

3.3 Descoberta de Padrões

Após a etapa de pré-processamento, na etapa de descoberta de padrões foi realizado a identificação da procedência das chamadas, utilizando as ferramentas *Pandas*, *NumPy,* e *sklearn* para identificação dos padrões, como o *Matplotlib* e o *Plotly* para visualização de dados.

O *Pandas* fornece funções que realizam análise geral dos campos da base de dados GDIS. Dessa maneira, foi realizada a análise de cada *feature* presente nos dados, obtendo a distribuição das chamadas procedente e improcedentes a partir de cada elemento.

Com *sklearn* foi possível identificar quais as *features*, tem a maior importância para o cenário de classificação através do treinamento dos dados com o modelo do

Random Forest, visto que esse modelo tem uma funcionalidade de medir a importância das *features*.

Para as identificações realizadas a partir de um código de fechamento associado à chamada quando ela é finalizada, sendo passada uma lista dos códigos de fechamento que identificam a procedência das chamadas. O *NumPy*.foi utilizado para criar a variável-alvo do trabalho, sendo essa variável denominada como “procedente”, sendo utilizada na etapa de construção do classificador. O código de fechamento é um valor numérico relacionado a natureza do problema da chamada. A partir desses códigos é inferido se as chamadas são de caráter improcedente ou procedente.

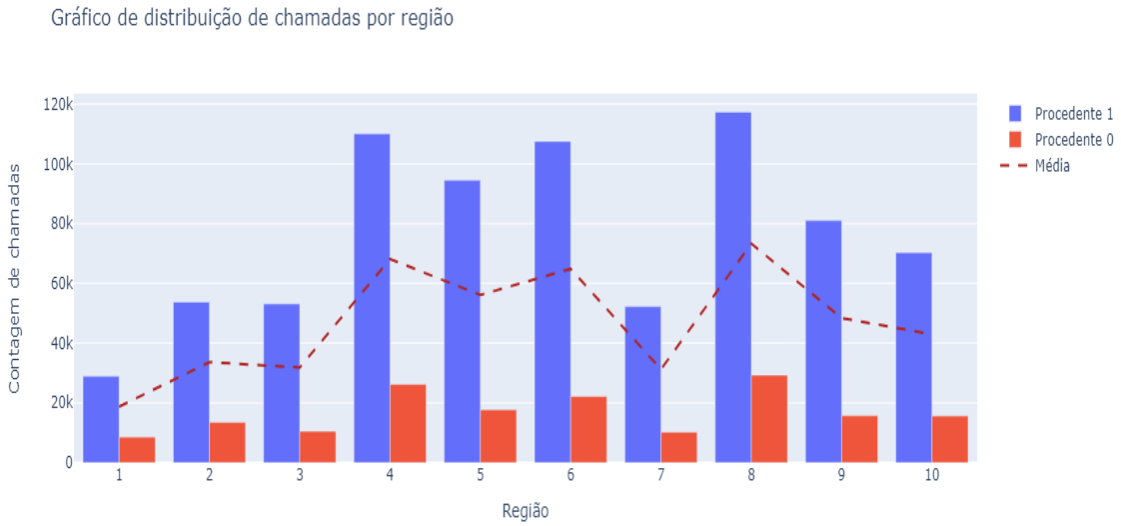
Na Figura 8 é apresentada a distribuição dos códigos de fechamento para as chamadas improcedentes. A partir dessa distribuição, observa-se que a maioria das solicitações improcedentes estão vinculadas a um conjunto específico de problemas, sugerindo que há padrões recorrentes nas reclamações.

Figura 8 – Distribuição dos códigos de fechamentos das chamadas improcedentes

Fonte: Elaborado pelo autor, 2024.

Além da identificação das chamadas improcedentes, foi possível identificar qual a região em que essas solicitações têm maior tendência a ocorrer. Na Figura 9, o gráfico mostra a distribuição de chamadas por região, assim como a média de registros de cada região representada pela linha no gráfico.

Figura 9 – Gráfico de distribuição de chamadas por região



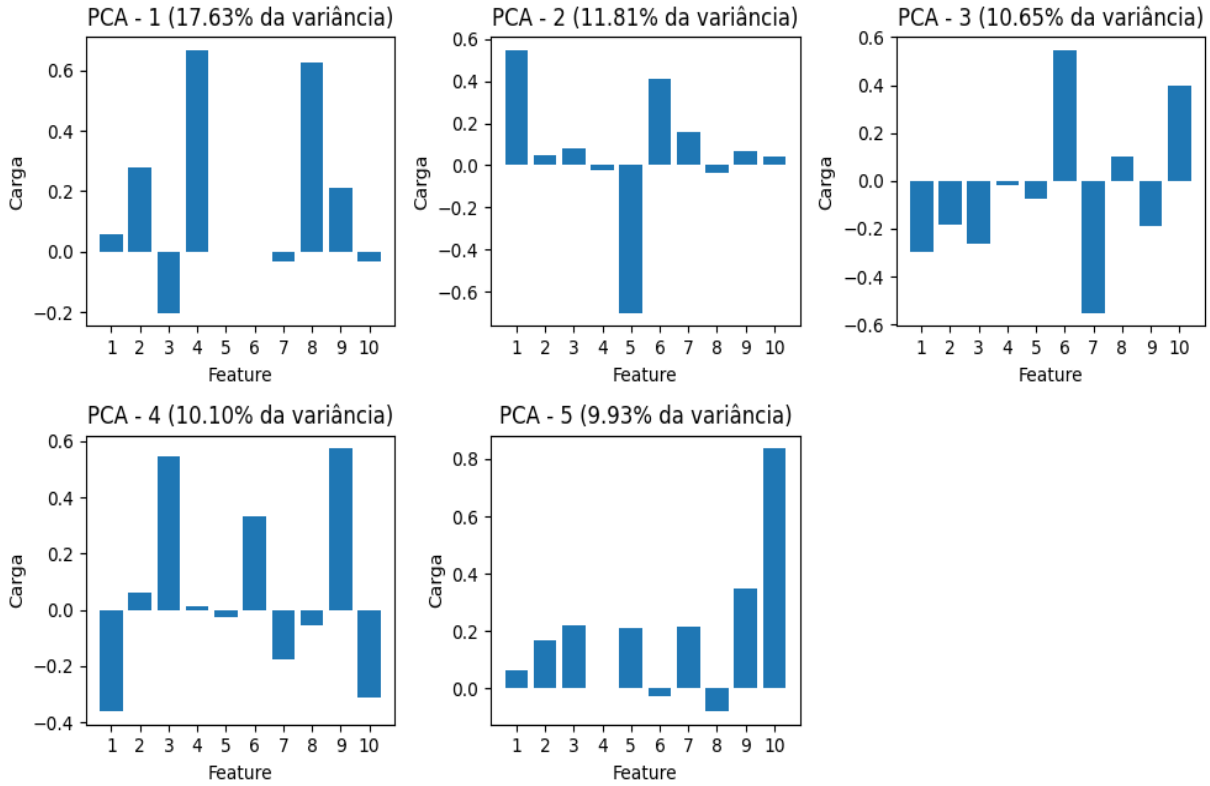
Fonte: Elaborado pelo autor, 2024.

A partir da análise dos padrões identificados, a próxima seção é avaliar os resultados obtidos a partir de técnicas de redução de dimensionalidade utilizadas, bem como os algoritmos de *ML* aplicados para o desenvolvimento do modelo de classificação de chamadas.

3.4 Construção do Classificador

. Na etapa de construção do classificador, para identificar a direção em que os dados variam, foi aplicado a *PCA* para explicar a maior variância dos dados. Na Figura 10, os gráficos mostram como cada uma das variáveis originais contribui para os componentes principais, e a direção das suas influências (positiva ou negativa). Esses componentes representam novas dimensões que capturam a maior variação possível nos dados, com o intuito de reduzir a dimensionalidade da base original sem perder a maior quantidade possível de informações.

Figura 10 – Variâncias dos componentes principais

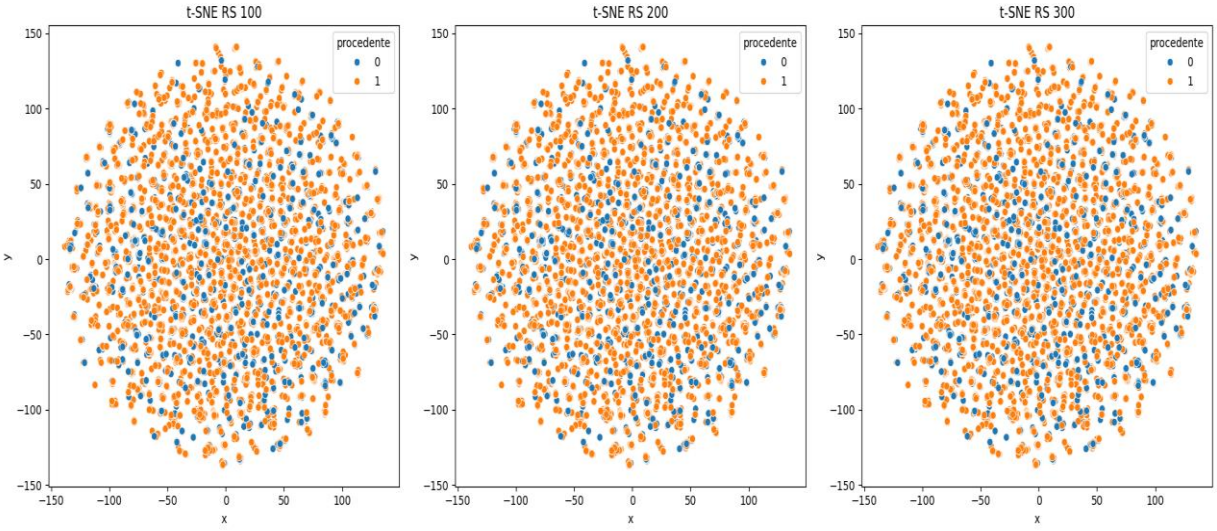


Fonte: Elaborado pelo autor, 2024.

Como visualizado na Figura 10, o componente principal 1 representa 17,63% da variância dos dados, com destaque para as features 4 e 8 apresentando uma carga alta e positiva, indicando que essas *features* desempenham um papel importante na explicação da maior variância dos dados. Isso permite que os modelos de *ML* possam operar de maneira mais eficiente e com menor dimensionalidade.

Afim de conseguir realizar uma separação entre as chamadas procedentes e improcedentes em um espaço, foi aplicada a redução de dimensionalidade fazendo uso do método do *t-SNE*, para que pudesse ser feita a separação entre as classes das chamadas procedentes e improcedentes. Na Figura 11 é ilustrado as projeções da aplicação do *t-SNE* definindo 3 valores diferentes de *random_state*. Entretanto, não foi possível obter uma separação entre as classes, indicando que o padrão de procedência das chamadas pode depender de fatores mais complexos.

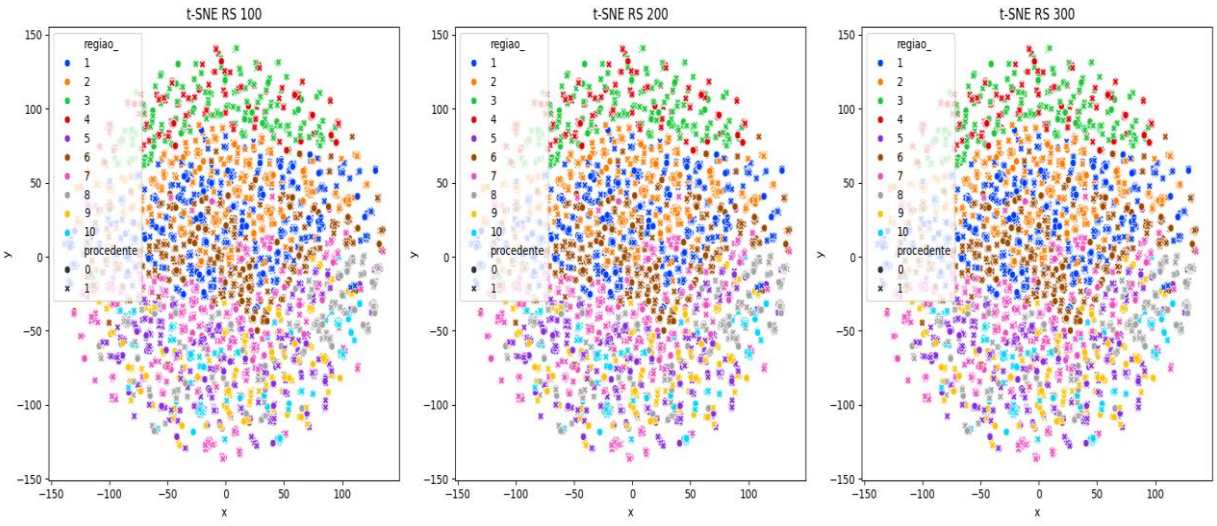
Figura 11 – Aplicação do *t-SNE* sobre as chamadas



Fonte: Elaborado pelo autor, 2024.

Como não foi possível obter uma separação entre as chamadas, foi realizada a aplicação do método do *t-SNE* na chamadas novamente, dessa vez na tentativa de realizar a separação entre as classes pela região do atendimento. Na Figura 12 são apresentadas as projeções do *t-SNE* na tentativa de realizar a segmentação das classes a partir da região. Contudo, mesmo apresentando uma pequena diferença comparado a aplicação voltadas apenas nas chamadas procedentes e improcedentes, os resultados da *t-SNE* não foram capazes de realizar uma separação entre as classes.

Figura 12 – Aplicação do *t-SNE* sobre as chamadas por região



Fonte: Elaborado pelo autor, 2024.

Assim, dado que a aplicação do *t-SNE* não conseguiu produzir uma separação das classes, a utilização da *PCA* se mostrou ser uma alternativa mais adequada para compreender as direções de maior variância nos dados. Isso permitiu uma análise mais direta das variáveis que contribuem para a classificação das chamadas.

Após a aplicação das técnicas de redução de dimensionalidade, foi realizada a divisão da base de dados em conjuntos de treino e teste. A separação seguiu a estratégia clássica de divisão, onde 70% dos dados foram reservados para o treinamento dos modelos, enquanto 30% foram alocados para a validação dos resultados. Essa divisão é fundamental para garantir que os modelos de *ML* possam ser avaliados de forma mais assertiva, garantindo que o desempenho obtido nos dados de treino sejam generalizáveis para novos dados.

Para a construção do modelo baseado em *ensemble* utilizando a técnica de *stacking*, foram utilizados os modelos *XGBoost*, *NN*, *LightGBM* e *Random Forest*, foi aplicado diferente pesos para que suas predições sejam combinadas. A ideia é explorar os pontos fortes de cada algoritmo, atribuindo maior peso para os modelos que melhor capturam os padrões nos dados.

O *XGBoost* devido ao seu desempenho superior em capturar a relação entre *features* e a variável-alvo, recebeu um peso de 0.66. O algoritmo de *NN* com a capacidade de generalização onde outros algoritmos não capturam a complexidade adequada dos dados, recebeu um peso de 0.302. O *LightGBM* eficiente para trabalhar com grandes volumes de dados, recebeu um peso de 0.019. O *Random Forest* com critério *Gini* para medir a qualidade das previsões, recebeu também o peso de 0.019.

Após a construção do modelo, foi realizado a avaliação do desempenho utilizando as métricas de classificação para assegurar que o modelo estivesse funcionando de maneira eficaz. As principais métricas utilizadas incluem a matriz de confusão, a acurácia e curva *ROC_AUC*. Cada uma delas oferecem *insights* sobre a qualidade das predições e são importantes para uma análise abrangente do modelo.

4 RESULTADOS

Nesta seção serão apresentadas as especificações da máquina e ferramentas utilizadas para o desenvolvimento e treinamento do modelo de *ML*, como também a avaliação do classificador de chamadas obtido no trabalho.

4.1 Ambiente de Execução

As especificações do ambiente de execução são fundamentais para garantir a reprodução dos experimentos e a consistência dos resultados. Todos os experimentos foram realizados em um ambiente local, utilizando um computador com as especificações descritas na Tabela 2.

Tabela 2 – Especificações do computador.

Processador	Intel Core i5-1135G7
RAM instalada	20,0GB
Velocidade do processador	2.40GHz - 2.42 GHz
Armazenamento	SSD 256GB
Sistema operacional	Windows 11 23H2
Tipo de sistema	Sistema operacional de 64 bits, processador baseado em x64

Fonte: Elaborado pelo autor, 2024.

As ferramentas foram selecionadas de acordo com as necessidades para lidar com o volume e a complexidade dos dados do projeto. A utilização delas permitiu o desenvolvimento eficiente e a avaliação do modelo, e suas versões estão descritas na Tabela 3.

Tabela 3 – Versões das ferramentas

Python	3.10.11
Pandas	2.2.1
NumPy	1.26.4
Matplotlib	3.8.4
Plotly	5.19.0
Scikit-Learn	1.4.2
AutoGluon	1.0.0

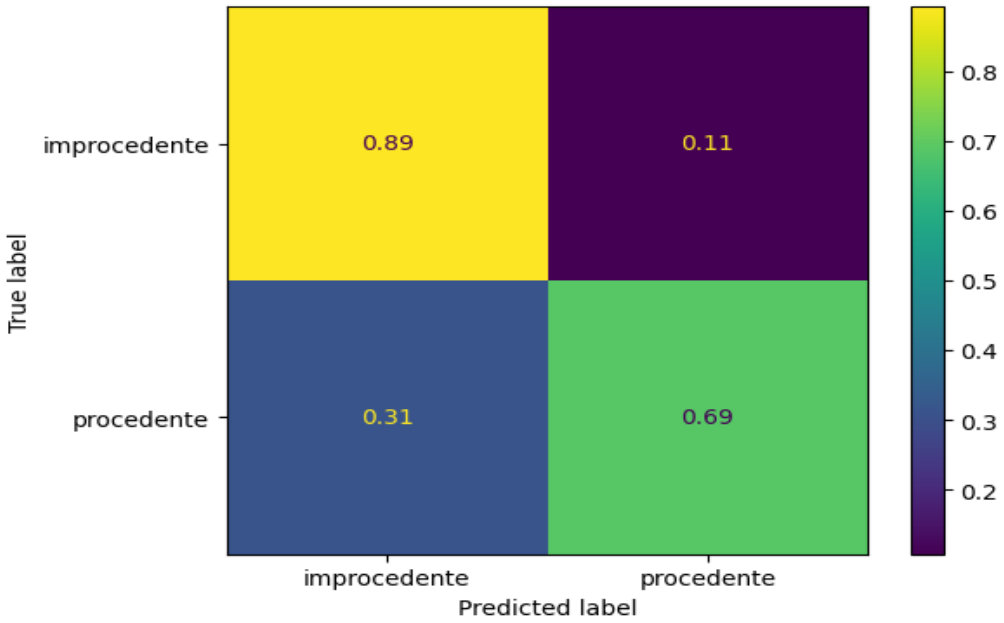
Fonte: Elaborado pelo autor, 2024.

4.2 Avaliação do Classificador de Chamadas

Para a avaliação dos resultados do modelo foi gerada a matriz de confusão para ter uma visualização da porcentagem das chamadas improcedentes que foram classificadas corretamente, assim como o percentual de chamadas procedentes que foram corretamente identificadas.

O modelo classificou corretamente 89% das chamadas improcedentes e teve um total 69% de acerto na classificação das chamadas procedentes. Isso mostra que o modelo tem um desempenho significativamente melhor para o cenário de detecção das chamadas de caráter improcedente, dado que essa detecção é o objetivo principal do modelo, reduzindo custos com deslocamentos improcedentes. Na Figura 13 é apresentada a matriz de confusão obtida a partir do desempenho do modelo.

Figura 13 – Matriz de confusão do *WeightedEnsemble_L2*



Fonte: Elaborada pelo autor, 2024.

A performance e tempo de execução são aspectos de suma importância a serem considerados, com o *WeightedEnsemble_L2* alcançando uma acurácia de 0.802 no conjunto de teste e 0.814 no conjunto de validação, indicando um desempenho consistente que possui uma boa capacidade de classificar corretamente novas chamadas, mostrando que o modelo não apresenta *overfitting*. Além disso, o modelo requer um custo de tempo de ajuste elevado, comum em *ensembles*, dado

que fazem a combinação de múltiplos modelos. O L2 no nome do modelo indicando que está no nível de empilhamento 2, combinando as previsões dos outros modelos de nível 1. A Figura 14 apresenta a performance e tempo de execução dos modelos de classificação.

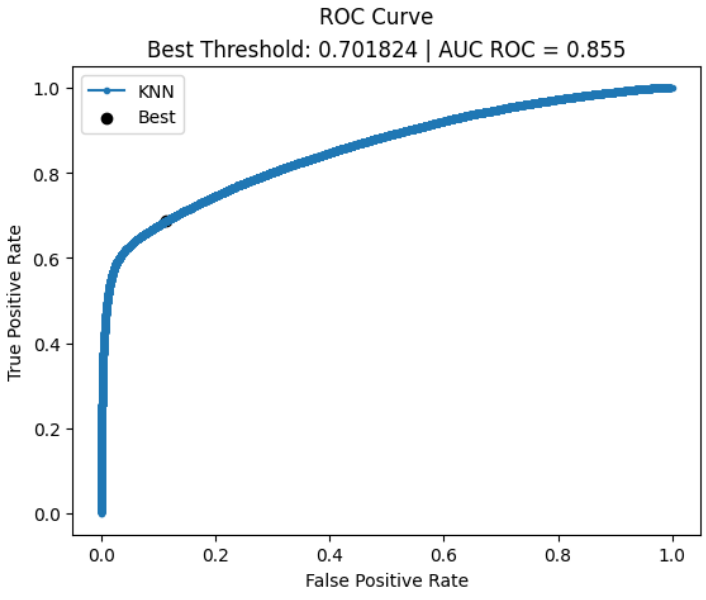
Figura 14 – Performance dos modelos de classificação

model	score_test	score_val	pred_time_test	pred_time_val	fit_time	time_test_marginal	time_val_marginal	fit_time_marginal	stack_level
WeightedEnsemble_L2	0,802761	0,814	12,994834	0,475002	658,535195	0,016495	0,004274	0,963269	2
LightGBMXT	0,800347	0,8004	1,755925	0,049532	15,293179	1,755925	0,049532	15,293179	1
XGBoost	0,799668	0,8056	2,641662	0,087234	22,533326	2,641662	0,087234	22,533326	1
NeuralNetTorch	0,795238	0,8032	0,731248	0,055095	264,939293	0,731248	0,055095	264,939293	1
LightGBM	0,794772	0,7924	1,328315	0,049936	10,812596	1,328315	0,049936	10,812596	1
NeuralNetFastAI	0,794268	0,7972	1,424957	0,044438	206,978299	1,424957	0,044438	206,978299	1
CatBoost	0,7942	0,792	1,013224	0,112339	112,102665	1,013224	0,112339	112,102665	1
RandomForestGini	0,792252	0,7892	7,739299	0,257154	418,675332	7,739299	0,257154	418,675332	1
RandomForestEntr	0,791447	0,7916	8,277114	0,278463	359,286711	8,277114	0,278463	359,286711	1
ExtraTreesEntr	0,784408	0,7832	11,293407	0,33141	491,40213	11,293407	0,33141	491,40213	1
ExtraTreesGini	0,782692	0,7844	10,217129	0,311112	432,298842	10,217129	0,311112	432,298842	1
LightGBMLarge	0,771863	0,7724	1,215022	0,036438	8,277801	1,215022	0,036438	8,277801	1

Fonte: Elaborado pelo autor, 2024.

A curva *AUC_ROC* foi utilizada para representar o equilíbrio entre sensibilidade e especificidade em diferentes limiares de classificação, facilitando a compreensão da performance do modelo. Isso acontece de acordo com objetivo estabelecido. Na Figura 15 é apresentada a curva *ROC* o resultado da *AUC* e do melhor limiar.

Figura 15 – Curva *ROC e AUC*



Fonte: Elaborada pelo autor, 2024.

O resultado da *AUC* foi calculado em 0.855, o que indica que o modelo tem uma capacidade muito boa de diferenciar as classes positiva e negativa. Esse resultado é consistente com a alta acurácia observada anteriormente. O melhor limiar foi identificado como 0.701, onde o ajuste entre sensibilidade e especificidade é otimizada.

4.3 Riscos à Validade

Embora os bons resultados com o modelo de classificação de chamadas para identificar a procedência das solicitações de serviço na concessionária de energia, é importante destacar alguns riscos à validade desses resultados quando consideramos a aplicação do modelo em outros cenários. A seguir, é discutido alguns pontos para assegurar que o modelo possa ser generalizado e adaptado a diferentes empresas.

4.3.1 *Dependência de Dados Específicos*

O modelo foi treinado utilizando dados específicos da concessionária de energia provenientes da base de dados GDIS, o que significa que o modelo aprendeu padrões e características únicas desse conjunto de dados. Sendo assim, o comportamento e o tipo de reclamações dos clientes podem diferir em empresas diferentes, visto que cada empresa possui uma estrutura de atendimento e um conjunto de necessidades específicas para a classificação de chamadas.

Por isso, ao aplicar o modelo em outra empresa ou setor, existe o risco de que ele não consiga identificar de forma adequada os padrões de procedência das chamadas, resultando em um desempenho inferior. Isso ressalta a necessidade da realização do treinamento do modelo sendo ajustado para os dados específicos de cada empresa em que o classificador for implementado.

4.3.2. *Mudanças ao Longo do Tempo*

Outro risco relevante à validade do modelo é o “*drift* de dados”, que ocorre quando as características dos dados mudam ao longo do tempo. Mudanças nas regulamentações, variações econômicas, novas práticas de atendimento e alterações

no comportamento das chamadas podem afetar diretamente a natureza das chamadas recebidas. Dessa maneira, em um cenário onde mudanças são frequentes, é necessário monitorar continuamente o modelo para assegurar que ele continue a fornecer previsões precisas

Em casos de *drift* de dados, o modelo pode precisar ser ajustado com os dados mais recentes para que se adapte aos novos padrões de atendimento e demandas dos clientes.

5 CONCLUSÃO

Há uma dificuldade por parte da concessionária de energia elétrica para identificar a procedência das chamadas, pois não há uma forma consolidada para detectar e separar quais são as chamadas improcedentes e procedentes. Diante disso, o desenvolvimento de um modelo de classificação provou ser uma solução para esse problema. Com isso, foi iniciado o desenvolvimento para a construção de um modelo de ML que possua a capacidade de identificar a procedência das ordens de serviço, com a finalidade de melhorar o atendimento ao consumidor no setor de energia elétrica.

A aplicação das técnicas de IA mostrou um resultado significativo na capacidade preditiva com a implementação do *WeightedEnsemble_L2.* Isso se dá devido a combinação ponderada de modelos diversos, permitindo capturar diferentes padrões e aspectos diferentes dos dados, reduzindo o risco de *overfitting* e aumentando a capacidade do modelo prever novas entradas de dados. A utilização do *AutoGluon* facilitou bastante as etapas para o desenvolvimento e obtenção dos resultados do modelo, dado que a ferramenta automatiza essas etapas.

Os resultados indicaram que o modelo foi eficaz em identificar os padrões de improcedência das chamadas, com 89% de detecção para previsão de chamadas improcedentes, fornecendo uma previsão que pode ser utilizada para aprimorar o processo de atendimento ao cliente. Embora os resultados apontem uma boa capacidade de previsão, as técnicas de *ensemble* requerem um processamento computacional mais intenso, exigindo mais tempo e recursos para o treinamento do modelo.

Para trabalhos futuros é sugerido o estudo e aplicação de novas técnicas que demandam menos recursos computacional para verificar se o desempenho pode ser aprimorado. Além disso, a obtenção de novos dados contribui no aperfeiçoamento das técnicas de *ML*, dado que a qualidade dos dados influencia diretamente no desempenho dos modelos de aprendizado de máquina. Dessa maneira, seria possível obter um modelo com melhor avaliação e desempenho, com a concessionária de energia elétrica reduzindo os gastos e melhorando o atendimento ao cliente.

REFERÊNCIAS

AGÊNCIA NACIONAL DE ENERGIA ELÉTRICA (ANEEL). **Manual de procedimentos do setor elétrico**. Brasília, 2020.

AGRAWAL, R.; IMIELIŃSKI, T.; SWAMI, A**. Mining association rules between sets of items in large databases**. *ACM SIGMOD Record*, v. 22, n. 2, p. 207-216, 1993.

AIN, A. K.; MURTY, M. N.; FLYNN, P. J. **Data clustering: a review**. *ACM Computing Surveys (CSUR)*, v. 31, n. 3, p. 264-323, 1999.

ALPAYDIN, E. ***Introduction to machine learning***. 3. ed. Cambridge, MA: MIT Press, 2014.

BREIMAN, L. **Bagging predictors**. *Machine Learning*, v. 24, n. 2, p. 123-140, 1996.

CHAPPELLE, O.; SCHOLKOPF, B.; ZIEN, **A. Semi-supervised learning**. Cambridge, MA: MIT Press, 2006.

CORTES, C.; VAPNIK, V**. Support-vector networks**. *Machine Learning*, v. 20, n. 3, p. 273-297, 1995.

ERICKSON, N. et al. **AutoGluon-Tabular: robust and accurate AutoML for structured data. Advances in Neural Information Processing Systems**, v. 33, 2020.

FAWCETT, T. **An introduction to ROC analysis**. *Pattern Recognition Letters*, v. 27, n. 8, p. 861-874, 2006.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. ***Deep learning***. Cambridge: MIT Press, 2016.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. ***The elements of statistical learning: data mining, inference, and prediction***. 2. ed. Springer, 2009.

HAUGELAND, J. ***Artificial intelligence: the very idea***. Cambridge, MA: MIT Press, 1985.

HAYKIN, S. ***Neural networks and learning machines***. 3. ed. Upper Saddle River: Pearson, 2009.

HOSMER Jr, D. W.; LEMESHOW, S. ***Applied logistic regression***. 2. ed. New York: Wiley, 2004.

HUNTER, J. D. **Matplotlib: a 2D graphics environment**. *Computing in Science & Engineering*, v. 9, n. 3, p. 90-95, 2007.

JOLLIFFE, I. T. ***Principal component analysis***. Springer Series in Statistics, 2002.

JOLLIFFE, I. T.; CADIMA, J. **Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences***, v. 374, n. 2065, p. 20150202, 2016.

JONES, M.; et al. P**rivacy concerns in health data analysis using artificial intelligence**. *Journal of Medical Ethics*, v. 46, n. 12, p. 841-847, 2020.

KE, G. et al. **LightGBM: a highly efficient gradient boosting decision tree**. In: *Proceedings of the 31st Conference on Neural Information Processing Systems (NeurIPS)*, 2017. p. 3146-3154.

KORHAN, O.; ASMAEL, M.; SAFAEI, B. **Machine learning in predictive maintenance towards sustainable smart manufacturing in industry 4.0**. *Sustainability*, v. 12, n. 19, p. 8211, 2020.

LUTZ, M. ***Learning Python***. 5. ed. O'Reilly Media, 2013.

McKINNEY, W. **Data structures for statistical computing in Python**. *Proceedings of the 9th Python in Science Conference*, 2010.

MURPHY, K. P. ***Machine learning: a probabilistic perspective***. Cambridge, MA: MIT Press, 2012.

NGUYEN, G. et al. **Machine learning and deep learning frameworks and libraries for large-scale data mining: a survey**. *Artificial Intelligence Review*, v. 52, n. 1, p. 77-124, 2019.

OLIPHANT, T. E**.** ***Guide to NumPy***. Trelgol Publishing, 2006.

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

PLOTLY TECHNOLOGIES INC. **Collaborative data science**. Plotly Technologies Inc., Montreal, QC, 2015.

POWERS, D. M. **Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation**. *Journal of Machine Learning Technologies*, v. 2, n. 1, p. 37-63, 2011.

QUINLAN, J. R. **Induction of decision trees**. *Machine Learning*, v. 1, n. 1, p. 81-106, 1986.

RUSSELL, S.; NORVIG, P. ***Artificial intelligence: a modern approach***. 3rd ed. Upper Saddle River: Pearson, 2016.

SCHMITZ, M.; BERNARDON, D.; GARCIA, V.; MILBRADT, R.; SCHMITZ, W.; SILVA, G. **Análise multicritério no atendimento de ordens emergenciais em redes de distribuição de energia elétrica**, 2016.

SUTTON, R. S.; BARTO, A. G. ***Reinforcement learning: an introduction***. 2nd ed. Cambridge, MA: MIT Press, 2018.

TEIXEIRA, W. W.; FREITAS, J. P. F. S.; FERREIRA, R. S.; MELO, A. L. M.; SIMIONI, T. **Aplicação de inteligência artificial no processo de triagem de chamados emergenciais da distribuidora visando redução de deslocamentos improcedentes no serviço de campo**. *XXV SNPTEE Seminário Nacional de Produção e Transmissão de Energia Elétrica*, 2019.

THEISSLER, A.; PÉREZ-VELÁZQUEZ, J.; KETTELGERDES, M.; ELGER, G. **Predictive maintenance enabled by machine learning: use cases and challenges in the automotive industry**. *Reliability Engineering & System Safety*, v. 215, p. 107864, 2021.

VAN DER MAATEN, L.; HINTON, G. **Visualizing data using t-SNE**. *Journal of Machine Learning Research*, v. 9, p. 2579-2605, 2008.

VAN ROSSUM, G.; DRAKE, F. L. ***The Python language reference manual***. Network Theory Ltd., 2003.

WANG, S.; et al. **Artificial intelligence in finance: state of the art and future prospects**. *Financial Innovation*, v. 4, n. 1, p. 1-14, 2018.

WOLPERT, D. H. **Stacked generalization**. *Neural Networks*, v. 5, n. 2, p. 241-259, 1992.

YANG, Y.; LV, H.; CHEN, N. **A survey on ensemble learning under the era of deep learning**. *Neural Computing and Applications*, v. 56, n. 6, p. 1-20, 2022.

ZHANG, J.; CHEN, Y. **Explainable AI for improving customer satisfaction in call centers**. *Journal of Service Research*, v. 23, n. 4, p. 559-574, 2020.

ZHANG, L.; et al. **Internet of things (IoT) and artificial intelligence (AI): a survey of applications and challenges**. *IEEE Internet of Things Journal*, v. 6, n. 3, p. 2214-2228, 2019.