UNIVERSIDADE FEDERAL DO AMAZONAS

INSTITUTO DE COMPUTAÇÃO

PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Jacó Miranda dos Santos

Otimização do Processo de Vacinação por meio de *Machine Learning* e Dispositivos IoT: Monitoramento de Doses Aplicadas e Recomendações Baseadas em Dados.

Manaus

2024

Jacó Miranda dos Santos

Otimização do Processo de Vacinação por meio de *Machine Learning* e Dispositivos IoT: Monitoramento de Doses Aplicadas e Recomendações Baseadas em Dados.

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Informática do Instituto de Computação da Universidade Federal do Amazonas como requisito parcial para obtenção do grau de Mestre em Informática.

Orientador: Prof. Vicente Ferreira de Lucena Junior

Manaus

2024

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo(a) autor(a).

Santos, Jacó Miranda dos
S237o Otimização do processo de vacinação por meio de machine learning e dispositivos IoT : monitoramento de doses aplicadas e recomendações baseadas em dados / Jacó Miranda dos Santos . 2024
73 f.: il. color; 31 cm.

Orientador: Vicente Ferreira de Lucena Junior
Dissertação (Mestrado em Informática) - Universidade Federal do Amazonas.

1. Machine learning. 2. Gridsearchcv. 3. Rede de frio. 4. IOT-Internet of Things. 5. LoRa-Long Range. I. Lucena Junior, Vicente Ferreira de. II. Universidade Federal do Amazonas III. Título

Agradecimentos

Primeiramente, ao único a Quem defino o que sou e o que serei, ao Deus do impossível que durante a pandemia do covid-19 concedeu-me a oportunidade de ser um dos que chegou em 2022 com vida, diante muitas perdas, onde colegas, amigos e parentes se foram. Hoje, em 2024, por ter me dado forças e me guiado durante esta pesquisa, eu a Ele rendo graças! Agradeço minha esposa Dayana Leão e meus filhos (Jayana, Jordana e Jariel) pela compreensão durante essa caminhada. Ao meu orientador, Professor Vicente por me conduzir à reta final deste trabalho. Ao amigo Elson Vasques, motivador e profissional de trabalho, minha gratidão por compartilhar sua experiência nesta conquista. Agradeço ao parceiro de sala, Isaias Saraiva, pelo apoio e trocas de ideias relevantes, nossa amizade nos levou ao apoio recíproco e conquista do título.
E mais uma vez, a TI Senhor. Conceda-nos mais dias de PAZ!

RESUMO

A campanha de vacinação desempenha um papel crucial na proteção da saúde pública, especialmente em períodos de surtos epidemiológicos e pandemias. No entanto, a coleta e gestão de dados durante esse processo, em particular o controle de quantidades de frascos, doses e temperatura, continuam a enfrentar desafios significativos. Algumas das atividades de registro de dados, como o monitoramento da quantidade de doses e da temperatura, são realizadas manualmente, o que pode resultar em atraso no atendimento, em erros de registros e falta de eficiência. Relatórios analíticos que sintetizam dados oriundos de registros por dispositivos de sensoriamento remoto são geralmente utilizado para visualização e tomada de decisão. Nesse contexto, este trabalho apresenta um método para otimizar o processo de monitoramento que utiliza um modelo gerado por *Machine Learning* para sugerir recomendações com base na meta diária e dados climáticos da *openweather API*. Os dados foram simulados utilizando-se dispositivos de Internet das Coisas (*IoT*) com comunicação *LoRa* e tecnologia *RFID*. Sete modelos foram treinados por meio do *GridSearchCV* para otimizar o processo de ajuste dos melhores hiperparâmetros e seleção do melhor modelo de classificação. O algoritmo *Gradient Boosting* obteve o melhor desempenho com a acurácia de aproximadamente 96,3% e resultado de *Best Parameters*: *'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 200*. Os dados foram recuperados e apresentados em um *dashboard* do *Power BI*. A utilização do *Machine Learning* e tecnologia de sensoriamento remoto na gestão de dados durante o processo vacinal representa um avanço significativo para a saúde pública. A automatização da coleta, da geração de informação e a previsão de demandas são fundamentais para garantir uma resposta rápida e precisa em situações de emergência, como surtos de doenças contagiosas. Além disso, a tecnologia empregada pode ser adaptada e expandida para outros contextos de saúde, proporcionando maior eficiência e eficácia em diversas áreas da medicina como transplante de órgão por exemplo.

Palavras-chave— *machine learning, gridsearchcv*, *rfid*, *lora*, *iot*, rede de frio.

ABSTRACT

The vaccination campaign plays a crucial role in protecting public health, especially during periods of epidemiological outbreaks and pandemics. However, data collection and management during this process, in particular the control of vial quantities, doses and temperature, continue to face significant challenges. Some of the data recording activities, such as monitoring the quantity of doses and temperature, are carried out manually, which can result in delays in service, recording errors and lack of efficiency. Analytical reports that synthesize data from records by remote sensing devices are generally used for visualization and decision making. In this context, this work presents an approach to optimize the monitoring process through a method that uses a model generated by Machine Learning to suggest recommendations based on the daily target and weather data from the OpenWeather API. The data was simulated using Internet of Things (IoT) devices with LoRa communication and RFID technology. Seven models were trained using GridSearchCV to optimize the process of tuning the best hyperparameters and selecting the best classification model. The Gradient Boosting algorithm achieved the best performance with an accuracy of approximately 96.3% and Best Parameters result: 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 200. The data was retrieved and presented in a dashboard of Power BI. The use of Machine Learning and remote sensing technology in data management during the vaccination process represents a significant advance for public health. Automating collection, generating information and forecasting demands are fundamental to ensuring a quick and accurate response in emergency situations, such as outbreaks of contagious diseases. Furthermore, the technology used can be adapted and expanded to other healthcare contexts, providing greater efficiency and effectiveness in various areas of medicine, such as organ transplantation, for example.

Keywords*: machine learning, gridsearchcv, rfid, lora, iot, cold chain.*

Lista de Figuras

Lista de Tabelas

Sumário

Capítulo 1 – Introdução

Neste capítulo são apresentadas a contextualização e a descrição do problema, bem como a motivação e as questões de pesquisas; os objetivos e a organização dos capítulos seguintes.

1.1. Contextualização

De acordo com o (Ministério da Saúde, 2022), por meio do Manual da Rede de Frio – MRF, dados essenciais para a identificação da vacina e suas características incluem data de fabricação, número do lote, validade e outras variáveis, como temperatura e umidade, que podem impactar na qualidade do produto. A coleta manual desses dados ocorre frequentemente no ambiente de estoque, geralmente nas unidades de saúde ou durante as próprias campanhas de vacinação.

O MRF traz o referencial teórico e operacional adequado para promover o mínimo de unidade de procedimentos fundamentais para garantir os aspectos importantes na qualidade e segurança dos imunobiológicos disponibilizados pelo Programa Nacional de Imunização (PNI).

As informações relacionadas ao gerenciamento de estoques de vacinas são registradas em papel e posteriormente digitadas em meio digital de informação, como Sistema de Informação do PNI, o SI-PNI.

O emprego da tecnologia digital (TD) no campo da saúde tem se mostrado cada vez mais relevante, especialmente no monitoramento, em previsões de riscos e geração de informações e recomendações para tomada de decisão. Essa relevância é corroborada por estudos como os de (Hu *et al.*, 2023) e (Kim *et al.*, 2021), que enfatizam a importância do uso de técnicas de *Machine Learning* (*ML*) na área da saúde, bem como dispositivos para coleta de dados envolvendo a *Internet* das Coisa (*IoT- Internet of Things)*. Não menos importante, outros estudos envolvendo temas diversos na área da saúde foram realizados por Moreira; Bernardino e Vieira (2022) e Ndiaye; Tendeng e Seck, (2020).

Os modelos de aprendizado de máquina geralmente aplicados na área da saúde são o *RNN (Recurrent Neural Network)*, *LSTM (Long Short-Term Memory), DT (Decision Tree)* e *GRU(Recurrent Unit.* A aplicação da TD trazem diversos benefícios na gestão da Rede de Frios como afirmado no próprio MRF onde constam

informações sobre os sistemas de informações de conservação, estoque, logística, procedimentos, descarte e requerimento de imunizantes.

Contudo, ainda se percebe, no contexto da saúde e, em especial, durante as campanhas de vacinação, o uso de registros de procedimentos em papel, assim como situações onde a aplicação de artefatos computacionais poderia trazer diversos benefícios. Assim, este trabalho apresenta uma problemática ainda recorrente no âmbito da saúde pública, onde se investigou a utilização de recursos computacionais com o objetivo de encontrar uma solução que minimizasse os erros de coleta de dados e facilitasse a análise dos dados coletados durante as campanhas vacinais.

1.2. Problemática

O monitoramento da Rede de Frio, conforme observado por Hasant et al. (2020), Enriko et al. (2021) e Hu et al. (2023), é aplicado em algumas instâncias ao longo da Rede, especificamente durante o transporte. Além disso, a partir desses estudos, observa-se que a análise de dados por meio da aplicação de *Machine Learning* é realizada em bancos de dados contendo informações de campanhas anteriores e não diretamente na instância local (sala de vacina).

Nas campanhas de vacinação, segundo o MRF, sejam elas em unidades básicas de saúde (UBS) ou em outros estabelecimentos, o transporte de vacinas é realizado em caixas térmicas. Nessas caixas, Figura 1, são utilizados termômetros analógicos ou digitais para medição das temperaturas, garantindo a observância dos padrões de conservação dos imunizantes.

Figura 1 – Condicionamento em caixas térmicas.

(Ministério da Saúde, 2022)

Além disso, são utilizados insumos, como bobinas reutilizáveis, para manter as vacinas na temperatura adequada. As bobinas são compostas de água ou gel à base de celulose vegetal e são refrigeradas a certas temperaturas. Depois, são colocadas nas caixas térmicas para conservação das vacinas no processo de armazenamento e transporte. Durante essas atividades, é imperativo monitorar constantemente as

vacinas, devido aos limites de temperatura estabelecidos para assegurar a qualidade. É crucial registrar os frascos utilizados e a quantidade de doses aplicadas.

Os frascos podem ser de dose única ou multidoses; por exemplo, um frasco pode conter 5 ou 10 doses. Se o frasco for aberto e apenas uma dose for utilizada, as demais doses correm o risco de serem perdidas caso não sejam aplicadas. Além disso, é fundamental que as doses sejam administradas antes do término da vida útil do imunizante.

Durante uma campanha vacinal, diversos processos são fundamentais na distribuição e aplicação das vacinas. Por exemplo, na vacinação contra a COVID-19, foram considerados grupos prioritários que são categorizados de acordo com critérios como idade, área profissional e comorbidades (doenças que podem agravar a ação do vírus), (Lana *et al.*, 2021).

Algumas ocorrências relacionadas à variação de temperatura, validade vencida, problemas logísticos e fatores como a incerteza da procura pelo imunizante no dia da campanha podem gerar situações adversas, gargalos e a necessidade de redistribuição (Hu *et al.*, 2023). Dessa forma, infere-se que:

Se houver uma procura ou demanda alta, podem surgir gargalos e falta do imunizante; e se a demanda for baixa, pode haver perdas, uma vez que frascos abertos e doses aplicadas representam doses perdidas.

Dessa forma pode-se elencar algumas situações durante esses cenários:

a) Relacionados a conservação ou estoque:
    - Alteração da temperatura, produto com valiade vencida, manuseio incorreto, falta de energia.
b) Relacionados a demanda:
    - Demanda acima do esperado, baixa demanda, remanejamento.

Uma questão observável durante o período de vacinação, é que a validade da vacina poderá finalizar no dia da campanha e, mesmo alcançada a meta para uma determinada demanda, muitos frascos com imunizantes poderão ser inutilizados por não ter direcionamento prévio ou mesmo dificuldades de logísticas e/ou tempo hábil para utilização de doses que sobram ao final do dia.

Exemplo de cenário comuns: o não comparecimento da categoria desejada ao estabelecimento vacinal por motivos adverso; a má distribuição do imunizante devido ao não cadastramento individual em sistema disponível para tal.

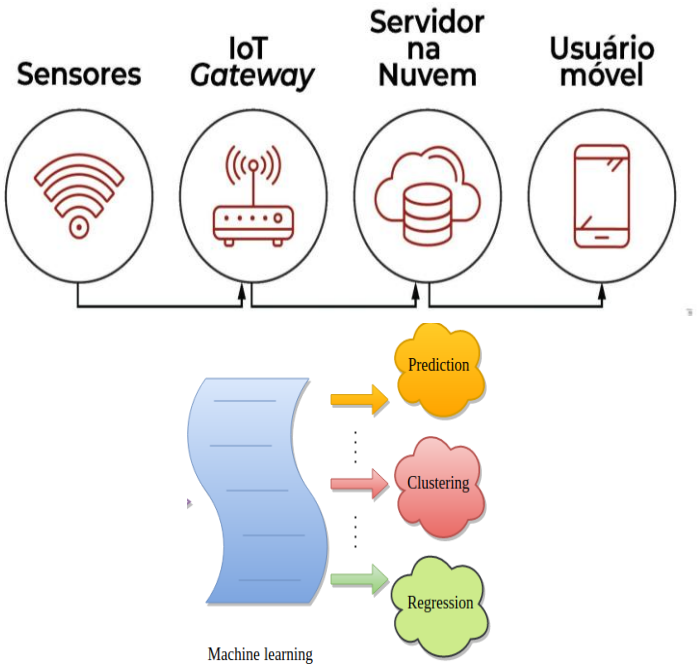
1.3. Motivação e Justificativa

A utilização de algoritmos, dispositivos inteligentes de *IoT*, conforme Akhbarifar *et al.* (2023) e Júnior; Vieira; Xavier, (2021), vem possibilitando a coleta de dados e geração de informação de forma remota, proporcionando melhoria em sistemas de saúde.

Com isso, surgem novas aplicações e estudos que viabilizam a criação de bases de dados propícias à análise e aplicação de algoritmos que apoiam ações estratégicas, tais como classificações de risco, automação de procedimentos e aumento de confiança nos diagnósticos em saúde e etc.

Dessa forma, mediante as estruturas tecnológicas disponíveis, conforme mostrado na Figura 2, que abrange a computação em nuvem, o processamento de dados por meio de plataformas de *hardware* e *softwares*, a utilização de sensores inteligentes e *frameworks* para aplicação de Inteligência Artificial, este trabalho procura contribuir com o estudo e aplicação dessas tecnologias no campo da saúde e, especificamente, na coleta e análise de dados durante a vacinação.

Figura 2 - Sensoriamento *IoT*, *cloud*, análise e aplicação.



Fonte: o autor, adaptada. https://pt-static.z-dn.net/files/d05/bc30d191b73405709348de3288489179.png

1.4. Hipótese

Existem benefícios consideráveis em melhorar o monitoramento da temperatura e contagem da vacina durante as campanhas de vacinação por meio da integração de métodos de coleta e análise de dados, aproveitando as vantagens do *Machine Learning* e da *Internet of Things*?

1.5. Objetivos

Nesta seção, são descritos o objetivo geral e específicos.

1.6. Objetivo Geral

O objetivo geral desta pesquisa é investigar a viabilidade de aprimorar o monitoramento da temperatura e das doses aplicadas e fornecer suporte à tomada de decisão.

1.7. Objetivos Específicos

I. Realizar uma revisão detalhada das tecnologias de sensoriamento remoto aplicadas à Rede de Frio e das técnicas de *Machine Learning* relevantes para análise de dados.

II. Implementar um sistema de sensoriamento remoto capaz de coletar dados relacionados à temperatura e detecção de frascos.

III. Utilizar técnicas de *Machine Learning*, processar os dados coletados ou simulados, treinar e avaliar modelos.

IV. Desenvolver uma aplicação para exibição das recomendações geradas pelo modelo de *Machine Learning*.

1.8. Metodologia

O Trabalho, quanto a natureza, foi, (i) uma pesquisa aplicada, segundo Paranhos (2014), a pesquisa aplicada ou tecnológica, objetiva gerar conhecimento para aplicação prática direcionado à resolução de problemas específicos.

Empregou-se, (ii) o método de abordagem hipotético-dedutivo na análise dos dados, o qual, conforme (Alegria *et al.*, 2011), a pesquisa parte de uma premissa geral para específica, visando descobrir as relações presentes na diversidade dos dados.

E, (iii) foram utilizados os procedimentos técnicos de pesquisa bibliográfica e experimental, e procedimento monográfico em sua elaboração.

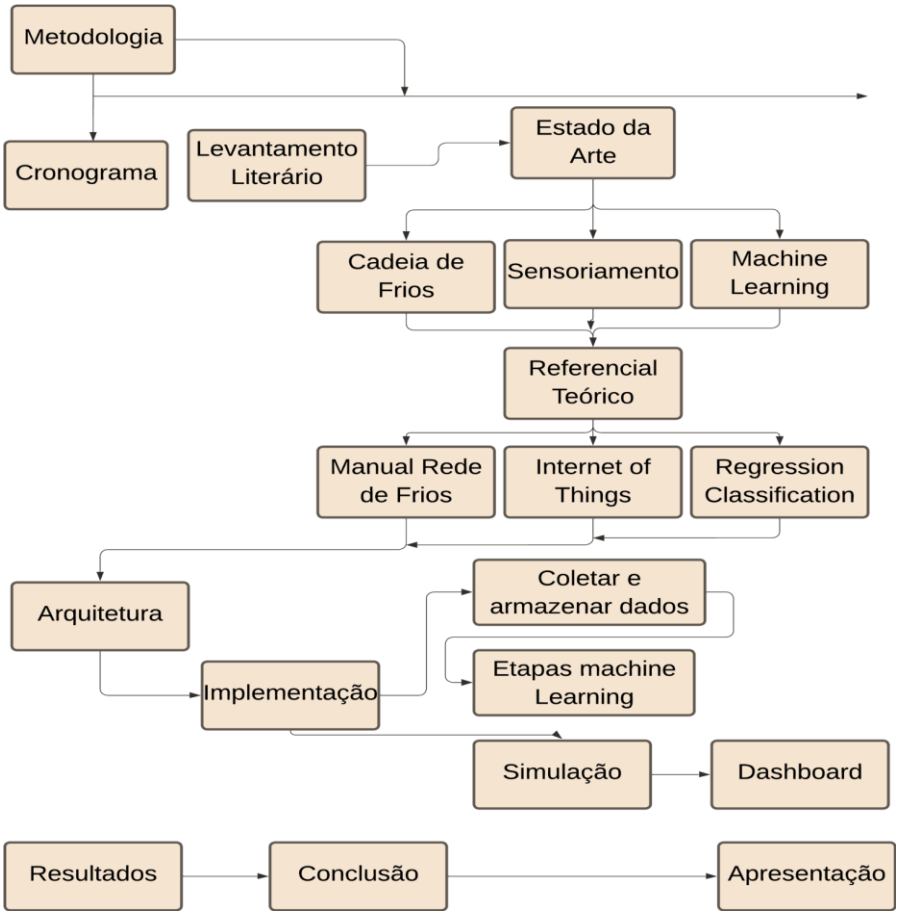
Na coleta de dados utilizou-se (iv) a documentação indireta, onde "...o pesquisador procura o levantamento que outros já fizeram". (Rampazzo, 2005)

Simulou-se em laboratório e/ou local próprio o comportamento de uma sala de vacinação.

Para fins de embasamento científico, as subseções 1.9 e 1.10, trazem o estado da arte e os assuntos pesquisados relacionado ao tema proposto respectivamente.

A Figura 3 presenta um diagrama ilustrativo que delineia os passos a serem seguidos para a pesquisa do material, bem como sequência estrutural de ações empregadas para finalização deste trabalho.

Figura 3 - Metodologia aplicada



Fonte: o autor.

1.9. Estado da Arte

Para proporcionar uma base comparativa e orientar a delimitação do presente estudo, foram realizadas pesquisas sobre assuntos semelhantes ao tema proposto. Os trabalhos relacionados foram selecionados durante o levantamento literário e são comentados a seguir. Ao todo, foram identificados dez trabalhos que contribuem para o entendimento do contexto:

1- Hu *et al.* (2023) afirma que o desequilíbrio entre o fornecimento e a demanda de vacinas, especialmente durante a pandemia de COVID-19, foi um desafio significativo na Cadeia de Suprimentos de Vacinas (CSV). Segundo Hu *et al*. (2023), a estimativa incorreta da demanda por vacinas frequentemente resulta na expiração das mesmas e na utilização inadequada de recursos. Propõe um sistema de dados em tempo real sobre o uso de vacinas nos centros de vacinação, os quais são inseridos no blockchain para gerar dados sobre a demanda por vacinas. Técnicas de aprendizado de máquina

são empregadas para treinar esses dados, sendo modelos de séries temporais como *RNN*, *LSTM* e *GRU* comumente utilizados. Em nosso estudo, comparamos o desempenho desses modelos para previsão da demanda por vacinas utilizando um conjunto de dados específico. O conjunto de dados consiste nos volumes de vacinação contra influenza nos *EUA* (em milhões de doses) de 1980 a 2020, sendo os dados de 1980 a 2011 usados para treinamento e os de 2012 a 2020 usados para teste. Os modelos, incluindo *RNN*, *LSTM* e *GRU*, são projetados para prever a demanda por vacinas contra influenza com base nas doses de vacinação dos sete anos anteriores. Por meio dessa análise, buscou aprimorar a precisão da previsão da demanda por vacinas e mitigar os riscos associados à expiração de vacinas e à má alocação de recursos na CSV.

2- Conforme Enriko *et al.* (2021) o rastreamento e monitoramento na cadeia de frios de imunizantes e distribuição de vacinas podem correr implantando um sistema rastreador baseado em *LoRa* (Sistema embarcado com diversos recursos como *WI-FI*, sensores diversos capaz de monitorar e manter controle, além de conectasse a *Internet*). A distribuição da vacina é uma tarefa desafiadora segundo os autores, especialmente para um país do arquipélago como a Indonésia. A distribuição da vacina, Covid-19, é um sistema baseado em *offline*, usando relatórios de monitoramento em papel, sistema de monitoramento individual da cadeia de frio e nenhum alerta para prevenir condições não ideais da vacina. A pesquisa dedica-se a melhorar o sistema de distribuição de vacinas da cadeia de frio existente na Indonésia, fazendo uso da *IoT*. O uso de sensores nos caminhões de distribuição para monitorar as condições da vacina proporciona alertas de notificações em tempo real gerando relatórios digitais diminuindo a utilização dos relatórios de papel.

3 - Bhatia; Malhotra (2021) corroboram que a mineração de dados, junto com o aprendizado de máquina, desempenha um papel importante na previsão de doenças. Com a pandemia do Covid-19 novos cenários surgiram com restrições sanitárias no mundo todo. O Covid-19 é uma doença transmissível e leva de 12 a 24 horas para receber os laudos de diagnóstico. Em várias áreas remotas e de grande altitude e devido ao crescimento exponencial do Covid-19 em várias partes do mundo, não é viável realizar o teste em massa populacional. A pesquisa descreve uma nova técnica para diagnosticar coronavírus usando classificador *NBC* e conseguir dá um grande passo para prever o Covid-19. Calculou-se todas as probabilidades individuais possíveis aplicadas no atributo alvo do coronavírus contendo todas probabilidades de atributo de coronavírus. As técnicas de mineração de dados podem ser implementadas em associação com o algoritmo classificador *Naive Bayes*.

4- Suguna; Sathishkumar; Deepa (2020) propõem um método utilizando o algoritmo *Naive Bayes* para melhorar a exatidão das recomendações de produtos na *Internet*, reduzindo os problemas de escalabilidade da abordagem de filtragem colaborativa convencional, agrupando os usuários com base em suas informações fornecidas em seus perfis. O conjunto de dados é retirado da base de dados *MovieLens* e *IMDb*, e

a precisão do sistema de recomendações é medida com base na métrica principal *f-measure*. O resultado do experimento proporcionou um melhor aprimoramento com novos atributos agregados. A precisão da recomendação do sistema é estendida para 87% no *IMDb* e 88% no conjunto de dados *MovieLens*. Foi utilizado a aprendizagem não supervisionada para o algoritmo descobrir a semelhança dos usuários. O Autor sugere para trabalhos futuros a recomendação baseada em itens personalizados usando técnicas de aprendizagem para melhorar a precisão das recomendações.

5 - Hasant *et al.* (2020) afirmam que a falta de gerenciamento adequado ocasiona o pouco alcance das vacinas em lugares remotos. Na cadeia de frios, um sistema pode ser extremamente benéfico quando utilizado para rastrear e monitorar os processos de transferência de vacinas em áreas remotas nos países em desenvolvimento e subdesenvolvidos. Um sistema concentrado em dados adquiridos em tempo real para monitoramento contínuo do processo de distribuição e transporte de vacina é proposto seguido do uso de várias tecnologias.

A utilização de sensores de temperatura, microcontroladores, comunicação *GSM*, *GPS-GPRS* module, podem ser eficazes no aumento da cobertura vacinal em regiões remotas. O monitoramento remoto consiste em um aplicativo para e gerenciar viagens individuais no processo de vacinas, juntamente com a supervisão regular de temperatura e umidade do ambiente transportador através do sensoriamento e transmissão de dados via *GSM* e localização por *GPS* e apresentação dos dados por *interface web*. Os autores sugerem uma melhoria do aplicativo para torna-lo mais amigável para os trabalhadores de campo não treinados e adicionar recursos conforme as necessidades futuras.

6 – Conforme Rawat *et al.* (2020), a tecnologia *LoRa* e o protocolo *LoRaWAN* têm se destacado como soluções eficazes para aplicações de IoT, especialmente durante a pandemia de COVID-19. A capacidade da *LoRa* de operar com baixo consumo de energia e longo alcance torna-a ideal para dispositivos de *IoT* que exigem vida útil prolongada da bateria e cobertura ampla, tanto em áreas urbanas densas quanto em regiões rurais. O artigo descreve como dispositivos *LoRa* e o protocolo *LoRaWAN* têm sido implementados para garantir a segurança pública, auxiliando no rastreamento de contatos, conformidade com regulamentos de saúde no ambiente de trabalho e atendimento às necessidades dos profissionais de saúde. Exemplos específicos incluem sensores de temperatura corporal baseados em *LoRa* desenvolvidos pela *Polysense Technologies*, que permitem o monitoramento de indivíduos com febre, proporcionando dados precisos e em tempo real para autoridades de saúde. A tecnologia *LoRa* utiliza a modulação de espectro espalhado por *chirp* (*CSS*), o que proporciona uma comunicação robusta e de longa distância com baixa taxa de transferência de dados, comparada a outras tecnologias como *WI-FI* e *Bluetooth*, que são mais adequadas para comunicações de curto alcance.

7 - Para Haviluddin *et al.* (2018), um estudo aprofundado sobre técnicas acadêmico de avaliação de alunos usando *Naive Bayes Classifier* (*NBC*). O conjunto de dados

com parâmetros específicos como idade, local de nascimento, sexo, status de escola secundária (pública ou privado), departamento no ensino médio, atividade organizacional, idade no início do ensino médio, e progresso do programa de graduação do 1º ao 4º semestre com três critérios de graduação (rápido, ligado e atrasado vezes) foram descritos e analisados. O experimental os resultados indicaram que o algoritmo de precisão (AC - *Accuracy algorithm*) de 76,79% com taxa positiva verdadeira (*TP – True Positive*) de 44,62% usando dados de treinamento de qualidade de 80% e 90% têm um bom valor de precisão de desempenho. O método utilizou a matriz de confusão - *confusion matrix*.

8 - Devido ao aumento da sensibilidade das vacinas mais modernas e outros produtos da área farmacêutica, deve-se garantir a qualidade do produto e detalhes sobre casos de falha de equipamentos que levam a deterioração do produto ou mesmo algum evento do status da entrega deve ser reportado ao fabricante.

Segundo Mohsin; Yellampalli (2017) problemas de rastreamento e de localização, monitoramento de ambiente e lacunas de infraestruturas têm prejudicado o setor da logística da cadeia de frio. Os Autores propõem a redução da presença humana ao longo da cadeia por meio de artefatos da *IoT*. A solução seria a utilização de um módulo *GPS* e uma plataforma com sensores *IoT* para localização e controle do meio envolvido. Os dados de sensores são enviados via *WiFitransmitter* para um centro de recebimento principal de monitoramento que recebe as atualizações regulares sobre os parâmetros do ambiente controlado permitindo uma tomada de decisão em casos de emergência ou fatores relacionados.

9 - Palacio *et al.* (2017) mostram uma nova arquitetura para monitorar cadeias de frio utilizando o paradigma da Computação Ubíqua, por meio de Computadores de Placa Única. O sistema inclui instrumentação, processamento embarcado com *Single Board Computers*, banco de dados em tempo real, *Human Computer Interfaces*, gerenciamento remoto e suporte para implantação de uma solução completa. Uma gama de artefatos tecnológicos como: *Wireless Sensor Networks (WSN)*, *Single Board Computer (SBC), Microcontrolador, temperature sensor (TT01) and GPS (ZT01), Bluetooth LE, GSM, GPS-GPRS module*, *mobile application*. A mudança de temperatura pode causar danos nas propriedades específicas dos produtos. Monitorar constantemente a temperatura e registrá-la para oferecer rastreabilidade. Além disso, se houver necessidade de transporte de produtos, as coordenadas de posição também devem ser levadas em consideração, devido à possibilidade de erros no pessoal de logística, percorrer rotas não ótimas para chegar ao destino e aumentar o tempo de transporte. Assim, os gerentes de logística precisam de ferramentas para medir, salvar e analisar a temperatura e a posição em tempo real para tomar as decisões necessárias. A Computação Ubíqua pode cumprir o desafio, pois as informações geradas poderão ser lidas, modificadas e armazenadas em qualquer lugar e a qualquer hora. Alertas podem ser enviados sobre anomalias quanto à mudança de temperaturas ou coordenadas, adicionando ao sistema.

Para uma visão resumida e detalhada sobre quais tipos de comunicação, hardware e softwares utilizados pelos autores, bem como as variáveis monitoradas, elaborou-se a Tabela 1, conforme apresentada a seguir.

Tabela 1 - Síntese dos temas relacionados

Autores	Comunicação	Hardware/Software	Variáveis Monitoradas
Hu *et al*. (2023)	*Wireless, RFID*	*Plataforma IoT, Blockchain, Machine Learning (RNN, LSTM e GRU)*	Localização. Temperatura, satisfação do cliente
Enriko, *et al*. (2021)	*LoRa, WI-FI*	*ESP32*	Localização, temperatura
Bhatia & Malhotra (2021)	Previsão de doenças usando o algoritmo *Naive Bayes Classifier* (NBC).	*Machine Learning*	-
Suguna *et al*. (2020)	Aprimorar a precisão de um sistema de recomendação de produtos.	*Machine Learning*	
Hasant, *et al*. (2020)	*GSM, GPS-GPRS*	*WEB*	Temperatura, umidade localização
Rawat *et al*. (2020)	*Wireless, LoRa, LoRaWAN*	*Plataforma IoT, LoRaWAN, Semtech's LoRa devices*	Localização, Temperatura, Dados de saúde pública
Haviluddin *et al*. (2018)	Avaliação de alunos usando o algoritmo *Naive Bayes Classifier (NBC).*	*Machine Learning*	
Mohsin; Yellampalli, (2017)	*GPS, WI-FI*	plataforma *IoT*	Localização
Palacio, *et al*. (2017)	*GSM, GPS-GPRS*	*Mobile Application*	Temperatura, localização
Este Trabalho (2023)	*LoRa, WI-FI, GPS, RFID*	*ESP32, LoRa, Cloud, Machine Learning, Power BI*	Temperatura, contagem de frascos, condições climáticas

Fonte: o autor.

A maioria dos trabalhos utiliza sensores de temperatura para monitorar a temperatura do produto em tempo real. Alguns trabalhos também monitoram a localização do produto, a umidade do ambiente, a vibração e o choque. O algoritmo *NBC* é utilizado em alguns trabalhos para previsão e classificação. A Computação Ubíqua é uma nova abordagem para monitoramento da cadeia de frio.

1.10. Assuntos pesquisados

Em síntese, o material bibliográfico pesquisado é descrito a seguir:

a) Sensoriamento — Pesquisas quais as tecnologias de sensores que envolvem coleta de dados de temperatura, humidade, rastreamento, contagem de objetos, bem como formas transmissão e armazenamento de dados localmente ou em nuvem.

b) *Machine Learning* — Pesquisar quais modelos de algoritmos são empregados na predição de dados, quais *frameworks* mais utilizados e formas e linguagem de programação utilizadas na implementação desses modelos.

c) Cadeia de Frios — Pesquisar o funcionamento da Rede de Frio principalmente como se dá o armazenamento das vacinas durante as campanhas de vacinação, que diretrizes gerenciam esse processo e quais os principais fatores críticos na conservação e atualização de estoques.

1.11. Organização do documento

Este trabalho está organizado em capítulos que abordam diferentes aspectos da pesquisa. O primeiro capítulo, "Introdução", fornece contextualização, identifica a problemática, formula hipóteses, estabelece objetivos gerais e específicos, apresenta a metodologia, revisa o estado da arte e lista os temas a serem pesquisados.

O segundo capítulo, "Referencial Teórico", explora conceitos e tecnologias relevantes, desde *Machine Learning* até tecnologias de sensoriamento e contagem de objetos.

O terceiro capítulo, "Método Proposto", apresenta a arquitetura, bibliotecas utilizadas e detalhes da implementação do protótipo. O quarto capítulo aborda a escolha do modelo e sua implementação no *Power BI*. O quinto capítulo discute os resultados obtidos.

Por fim, o sexto e sétimo capítulos apresentam a conclusão e as considerações finais respectivamente. O documento encerra-se com as referências bibliográficas e anexo.

Capítulo 2 – Referencial Teórico

Nesse capítulo é apresentado os principais assuntos pesquisados na literatura sobre aprendizado de máquina, sensoriamento, *API* para condições climáticas e cadeia de frios. O assunto abordado já tem um direcionamento para as tecnologias que poderão ser utilizadas no método proposto.

2.1. *Machine Learning - ML*

Os termos mais frequentes na literatura que caracterizam o *Machine Learning* são quatro entre os noves encontrado: computador, predição, automação, aprendizado, inteligência, método, programação, processo e conjunto. Em seguida são apresentados os quatros grupos.

"Computador", algumas vezes definido como máquina por Mahesh e Files (2000), é a coisa-recipiente, onde tudo acontece. É a máquina na qual se armazena os dados; é o objeto programável onde a ciência experimenta os cálculos, onde se combina algoritmos e se gera a inteligência artificial O grupo "Predição", aparece como segundo grupo, com a mesma frequência do grupo computador e, representa os termos: produção, mudança, tempo, adaptação, futuro, previsão e melhoria.

O termo previsão assume o valor máximo da função do aprendizado de máquina onde fica evidente o esforço de se melhorar o algoritmo para se adaptar as mudanças, encontrar padrões, relacionar-se ao tempo, armazenar e analisar dados passados, e produzir resultados não descobertos (Isakova, 2021), (Marechal; DAHL, 2016).

"Automação" é o terceiro grupo semântico dos atributos. Conforme (Mahesh; Batta, 2020), a máquina utiliza métodos para detecção automática de padrões obtidos dos dados, e os usa para prever informações futuras. Além de fazer previsões, o computador seria capaz de tomar decisões, constituir autonomia, definir e modificar regras de acordo com os padrões encontrados (Chaurasia, 2020).

Segundo (Abdullah; ALI; abdulghafor, 2020) a programação de computadores, através de métodos supervisionados ou não, pode levar a máquina a aprender com os dados e com experiências passadas. O tipo supervisionado sugere que o computador terá uma intervenção humana em algum processo do aprendizado de máquina, como combinações diferentes de métodos caso o resultado esperado tenha algum parâmetro de comparação e etc. Roderus; Larson e Pihl (2021) afirmam que o

conhecimento pode ser extraído de um conjunto de dados por meio de um computador, construindo de forma artificial uma inteligência capaz de imitar certas características do pensamento lógico humano.

2.2. Tipos de Sistemas de *Machine Learning*

Este subtópico, baseado em uma pré-leitura, dará ênfase ao Aprendizado supervisionado objetivando convergir para solução da problemática desse trabalho.

Para (Géron, 2019), é útil classificar o *ML* em categorias com base em alguns contextos condicionais, conforme tabela abaixo, e conceitua as classificações descrito nos parágrafos seguintes.

Tabela 2- Categorias do Aprendizado de Máquina

Condição:	Categoria
Se forem ou não treinados com Supervisão humana.	Supervisionado, não supervisionado, semissupervisionado e aprendizado por reforço.
Se podem ou não apreender rapidamente de forma incremental.	Aprendizado online versus aprendizado por lotes.
Se funcionam simplesmente comparando novos pontos de dados. Se detectam padrões em dados de treinamento e criam um modelo preditivo, como os cientista.	Aprendizado baseado em instâncias versus aprendizado baseado em modelo.

Fonte: Adaptada (Géron, 2019)

2.3. Aprendizado supervisionado

Gérson (2019) ainda afirma que o *ML* pode ser classificado de acordo com a quantidade e o tipo de supervisão que os dados recebem durante a etapa de treinamento. Dessa forma, no aprendizado supervisionado, tarefa típica do *ML*, as soluções procuradas (rótulos) são incluídas juntamente com os dados utilizados para treinamento. Gérson toma como exemplo o filtro de spam que no algoritmo treinado são colocados muitos e-mails junto às classes de spam e não-spam para que o algoritmo apreenda e classifique os novos e-mails.

Alguns *algoritmos* importantes no aprendizado supervisionado são: *k-Nearest Neighbors*, *Linear Regression, Logistic Regression*, *Support Vector Machine* (SVM), *Decision Tree*, *Random Forests* e *Neural Networks*.

Para (Duarte de Britto Lira *et al.*, 2019), as técnicas de mineração de dados são fortemente utilizadas para a classificação de dados e a classificação consiste no processo de encontrar, através de aprendizado de máquina, um modelo ou função que descreva diferentes classes de dados.

A classificação tem como objetivo rotular de forma automática novas instâncias da base de dados com uma determinada classe aplicando-se um modelo predefinido.

Estes modelos ou função “aprendidos” é baseado no valor dos atributos das instâncias de treinamento.

2.4. Aprendizado não supervisionado

Nesta classificação, o *algoritmo* tenta aprender sozinho, ou seja, os dados de treinamento não são rotulados. A detecção de anomalias, por exemplo é uma tarefa importante não supervisionada. O aprendizado não-supervisionado pode ser utilizado para detecção de transações incomuns em cartões de crédito, entre outros, no intuito de evitar fraudes.

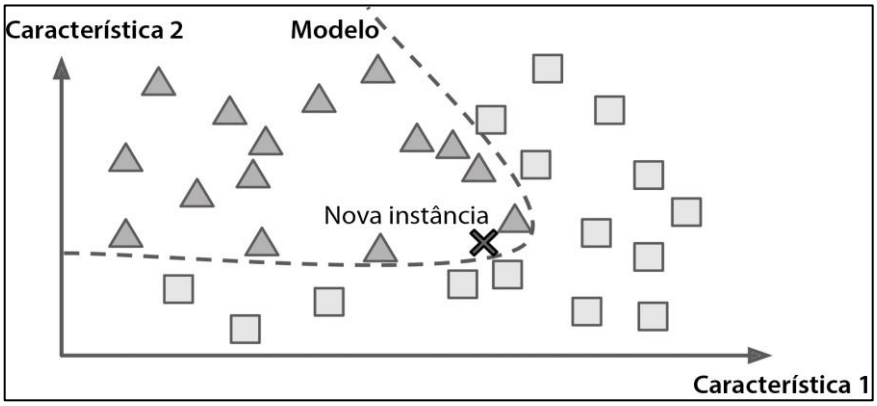
O conjunto de treinamento inclui as instâncias normais, sendo que, ao detectar uma nova instância, poderá dizer se ela parece normal ou não, apontando uma provável anomalia. Tem-se ainda algumas categorias: Semissupervisionado, por Esforço, Rede Neurais e etc., porém, o Aprendizado baseado em modelo será apresentado pois deseja-se descrever e entender melhor o seu uso nos dias atuais.

2.5. Aprendizado baseado em modelo

“...uma forma de categorizar os sistemas de Aprendizado de Máquina é por meio da generalização. A maioria das tarefas de Aprendizado de máquina faz previsões. [...] dada uma série de exemplos de treinamento, o sistema precisa ser capaz de generalizar em exemplos que nunca viu antes.” (Géron, 2019)

O aprendizado baseado em modelo surge da construção de um modelo a partir de um conjunto de exemplos, onde esse modelo poderá ser utilizado para fazer predições. Na Figura 4, o modelo mediante exemplos treinados agrupa os dados segundo suas as características 1 ou característica 2.

Figura 4 - Aprendizado baseado em modelo



Fonte: (Géron, 2019)

Se um modelo é uma função matemática, este pode ter parâmetros os quais podem ser ajustados com a finalidade de encontrar ou representar o melhor funcionamento possível deste modelo. Ou seja, o modelo que melhor se ajusta aos dados treinados. Existem bibliotecas e *frameworks* de *Machine Learning* gratuitos que

permitem o uso de diversos modelos. Alguns exemplos serão apresentados nos tópicos seguintes.

2.6. *Scikit-Learn*

Scikit-Learn é um módulo *Python* que integra uma ampla gama de algoritmos de aprendizado de máquina de última geração para problemas de escala média, tanto supervisionados quanto não supervisionados.

Este pacote se concentra em levar o aprendizado de máquina para não especialistas usando uma linguagem de alto nível de uso geral. A ênfase é colocada na facilidade de uso, desempenho, documentação e consistência da API.

No contexto do aprendizado supervisionado, existem diversos modelos, sendo possível descrever, pelo menos, cinco modelos e seus respectivos métodos nas

Tabela 3 - *Scikit-Learn* - Modelos de *Machine Learning*

Model	Main Method	Preprocessing	Metrics	Model hyperparameters
Linear Regression	Quadratic Error Minimization	Normalization	Mean Squared Error (MSE), Mean Absolute Error (MAE)	Number of terms, Regularization
Decision Tree	Data Cutting	Data Cleaning	Precision, Recall, F1-score	Maximum tree depth, Number of nodes, Splitting algorithm
Random Forest	Data Cutting	Data Cleaning	Precision, Recall, F1-score	Number of trees, Maximum tree depth, Number of nodes, Splitting algorithm
Gradient Boosting	Data Cutting	Data Cleaning	Precision, Recall, F1-score	Number of trees, Maximum tree depth, Number of nodes, Splitting algorithm
MLP Classifier	Backpropagation	Normalization	Precision, Recall, F1-score	Number of layers, Number of neurons per layer, Activation functions, Optimization algorithm

Fonte: (Pedregosa *et al.*, 2013)

O *Scikit-Learn* fornece uma variedade de métricas para avaliar o desempenho de modelos de *Machine Learning*, alguns dos quais são apresentados a seguir.

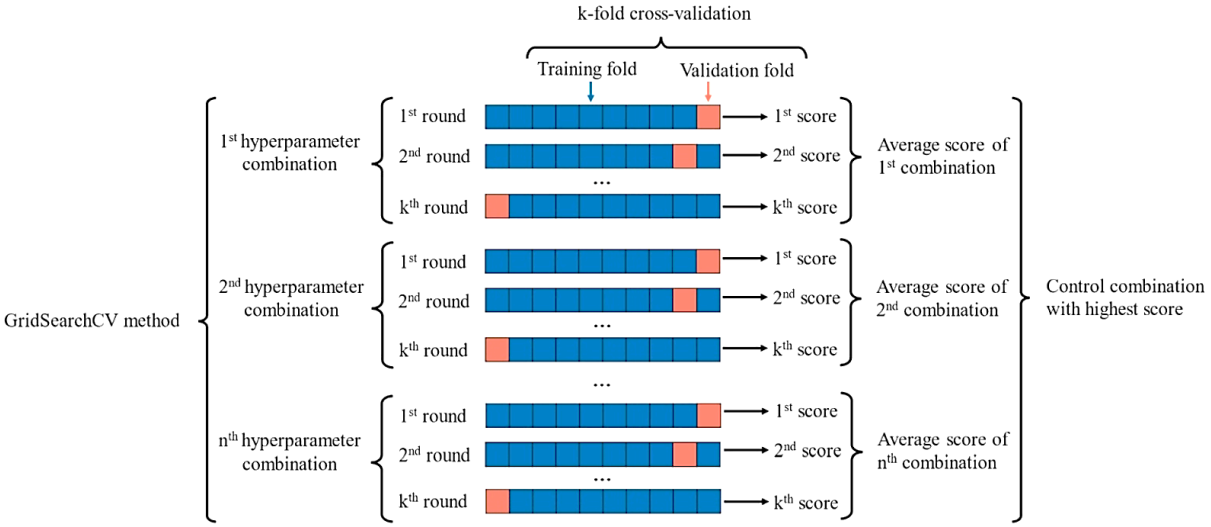
2.6.1. *GridSearchCV e* Validação Cruzada

O *GridSearchCV,* pesquisa em grade em português, é um método útil para descoberta da coleção ideal de hiperparâmetros para um modelo específico (Gill; Gupta, 2023). Também pode ser usada para determinar metodologias para descobrir a combinação ideal de hiperparâmetros (Ahmad *et al.*, 2022).

A Shatnawi *et al.* (2022) explora duas técnicas importantes para melhorar o desempenho de modelos de aprendizado de máquina: validação cruzada e ajuste de hiperparâmetros. A validação cruzada aborda o desafio dos dados limitados, dividindo o conjunto de dados em dobras para treinamento e validação. A técnica procura evitar *overfitting* e garantir que o modelo generalize bem para dados não vistos. O ajuste de hiperparâmetros envolve encontrar as configurações ideais para os hiperparâmetros de um modelo, que são cruciais para sua precisão.

Essa técnica automatizada explora todos os valores possíveis dentro de um intervalo definido para cada hiperparâmetro. Enfaticamente, a validação cruzada e a pesquisa em grade avalia diferentes combinações de hiperparâmetros e seleciona a que apresenta melhor desempenho.

Figura 5 – Método de *GridSearchCV*



Fonte: (Shatnawi *et al.*, 2022)

Os principais detalhes sobre a Figura 5 são apresentados a seguir:

Na validação cruzada *k-fold*, o conjunto de dados é dividido aleatoriamente em *k* dobras de tamanhos iguais para a primeira combinação até *n* combinação de hiperparâmetros. Para cada dobra, uma dobra é usada como conjunto de validação(*validation fold)*, enquanto as *k-1* dobras restantes são usadas como conjunto de treinamento. Este processo é repetido *k* vezes, garantindo que cada dobra seja usada para validação exatamente uma vez.

O desempenho do modelo, *score* 1 a *score k*, é então calculado em média em todas as *k* dobras, fornecendo uma estimativa mais robusta de sua generalização para dados não vistos em comparação com a avaliação do modelo em uma única divisão de treinamento-validação.

As escolhas comuns para *k* incluem 5, 10 e 20. O valor ideal de *k* depende do conjunto de dados específico e o modelo que está sendo usado.

2.6.2. *Mean Squared Error (MSE) e Mean Absolute Error (MAE).*

MSE (Mean Squared Error): esta métrica calcula a média dos quadrados das diferenças entre os valores previstos e os valores reais. É comumente utilizada para avaliar modelos de regressão, pois penaliza erros maiores mais severamente do que erros menores.

MAE (Mean Absolute Error): esta métrica calcula a média dos valores absolutos das diferenças entre os valores previstos e os valores reais. É menos sensível a *outliers* que o *MSE* e pode ser mais adequada para avaliar modelos de regressão com

distribuições de erro não gaussiana. É possível fazer a implementação no *Scikit-Learn* com os métodos seguintes:

MSE: sklearn.metrics.mean_squared_error(y_true, y_pred).

MAE: sklearn.metrics.mean_absolute_error(y_true, y_pred).

2.6.3. *Accuracy*.

A *Accuracy* é a métrica que calcula a proporção de previsões corretas feitas pelo modelo. É uma métrica simples e fácil de interpretar, mas pode ser enganosa em casos de classes desbalanceadas. O método para Implementação no *Scikit-Learn* é:

Accuracy: sklearn.metrics.accuracy_score(y_true, y_pred)

2.6.4. *Precision, Recall, F1-score*.

Precision: esta métrica calcula a proporção de previsões positivas que são realmente positivas. Pode-se avalia a exatidão do modelo quando os falsos positivos são mais custosos do que os falsos negativos.

Recall: esta métrica calcula a proporção de amostras positivas que são corretamente classificadas pelo modelo. Pode-se avaliar a exatidão do modelo quando os falsos negativos são mais custosos do que os falsos positivos.

F1-score: esta métrica é a média harmônica entre a precisão e o recall. Avalia o desempenho geral do modelo quando ambos os falsos positivos e falsos negativos são indesejáveis. O método para Implementação no *Scikit-Learn* é:

Precision: sklearn.metrics.precision_score(y_true, y_pred)

Recall: sklearn.metrics.recall_score(y_true, y_pred)

F1-score: sklearn.metrics.f1_score(y_true, y_pred)

A escolha da métrica de desempenho adequada depende do tipo de problema e dos custos associados a diferentes tipos de erros.

Geralmente, *MSE* e *MAE* são métricas para avaliar modelos de regressão. Accuracy pode ser enganosa em casos de classes desbalanceadas. *Precision, recall* e *F1-score* são métricas para avaliar modelos de classificação, especialmente quando ambos os falsos positivos e falsos negativos são indesejáveis. (Scikit Learn, 2023)

O Modelos linear, por exemplo, o *Linear Regression*, é um conjunto de métodos destinados a regressão em que se espera que o valor de destino seja uma combinação linear dos recursos ou características dos dados anteriores. Pedregosa, *et al*. (2013), afirma que a notação matemática para tais modelo pode ser ilustrada conforme apresenta-se na Equação 1, onde o vetor $w = (w_1, \ldots, w_p)$ é designado como *coef_* e w_0 como *intercept_.*

Equação 1 — $\hat{y}(\boldsymbol{w}, \boldsymbol{x}) = \boldsymbol{w_0} + \boldsymbol{w_1 x_1} + \ldots + \boldsymbol{w_p x_p}$

2.6.5. Pré-processamento dos dados (*Preprocessing data*).

Segundo Werner de Vargas *et al.* (2023), o pré-processamento de dados desempenha um papel crucial na obtenção de resultados precisos e confiáveis nas análises de dados. Ao limpar, transformar e organizar os dados de forma adequada, o pré-processamento elimina ruídos, inconsistências e redundâncias que podem comprometer a qualidade dos resultados.

Essa etapa é fundamental por várias razões: primeiro, ao eliminar erros e inconsistências, garantindo a confiabilidade dos dados para análise. Em segundo lugar, a normalização dos dados para uma escala uniforme permite comparações equitativas, garantindo que os dados sejam comparáveis e consistentes. Ao preparar os dados para modelos de *ML*, o pré-processamento estabelece as bases para modelos precisos e eficazes, reduzindo o viés natural e melhorando a interpretabilidade e o desempenho dos modelos.

O *StandardScaler*, pode ser usado para padronizar tanto características numéricas quanto categóricas. Para características categóricas, o *StandardScaler* usa o *LabelEncoder* para converter as categorias em valores numéricos. Se alguns valores discrepantes estiverem presentes no conjunto, escalonadores robustos ou outros transformadores podem ser mais apropriados. Alguns exemplos codificados podem ser encontrado no ANEXO A.

2.6.6. Métodos e funções utilizadas por modelos no *Scikit-Learn*.

Existe uma variedade de exemplos em linguagem *Python* para implementação de modelos e parâmetros. Os modelos possuem métodos úteis que são usados durante o aprendizado de máquina. A Tabela 4 apresenta alguns métodos utilizados.

Tabela 4 - Métodos utilizados pelos modelos Scikit-learn

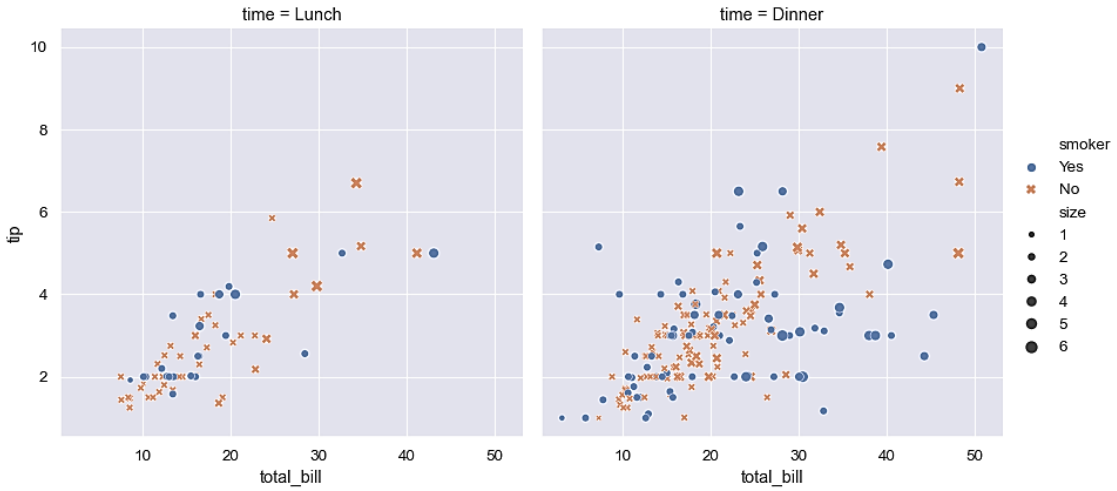
Métodos	**Descrição da funcionalidade**
fit(X, y)	Ajusta o modelo à matriz de dados X e ao(s) alvo(s) y. Treina o modelo.
get_metadata_routing()	Obtém o roteamento de metadados deste objeto.
get_params([profundo])	Obtém parâmetros para este estimador.
partial_fit(X, y[, classes])	Atualiza o modelo com uma única iteração sobre os dados fornecidos.
predict(X)	Prever usando o classificador perceptron multicamadas.
predict_log_proba(X)	Retorna o log das estimativas de probabilidade.
predict_proba(X)	Estima a probabilidade da predição.
score(X, y[, peso_amostra])	Retorna a precisão média dos dados de teste e rótulos fornecidos.
set_params(**parâmetros)	Define parâmetros para o estimador.
set_partial_fit_request(*[, Aulas])	Configura metadados passados para o partial_fit método.
set_score_request(*[, peso_amostra])	Configura metadados passados para o score método.

Fonte: o autor.

Também é possível fazer a representação gráfica utilizando-se de bibliotecas *Python*, como o *Matplotlib*. Uma delas é o *Seaborn*, que é uma biblioteca de visualização de dados baseada no *Matplotlib*.

O *Seaborn* fornece uma *interface* com estilos de gráficos predefinidos. Outra biblioteca que pode ser usada para plotar gráficos é o *Plotly*, que é uma biblioteca de visualização de dados interativa. O *Plotly* permite criar gráficos que podem ser visualizados no navegador, o que pode ser útil para compartilhar gráficos com outras pessoas, como mostrado na Figura 6.

Figura 6 – Visualização gráfica com *Seaborn.*



Fonte: https://seaborn.pydata.org/tutorial/introduction.html

2.7. *TensorFlow* (*TF*)

Descrito no site *tensorflow.org/federated*, o *TensorFlow* é uma biblioteca de código aberto para *Machine Learning*, também utilizado em outros cálculos com dados descentralizados. Entre as bibliotecas disponíveis algumas são citadas na Tabela 5.

Tabela 5 – Algumas bibliotecas do TensorFlow

Biblioteca	Definição	Recursos
TensorBoard	*É um kit de visualização e as ferramentas necessárias para experimentos*	1. Acompanhamento e visualização de métricas como perda e precisão; 2. Visualização do grafo do modelo (ops e camadas); 3. Visualização de histogramas de pesos, viés ou outros tensores conforme eles mudam ao longo do tempo; 4. Projeção de *embeddings* em um espaço dimensional inferior; 5. Exibição de dados de imagem, texto e áudio; 6. Criação de perfis de programas do *TensorFlow*.
TensorFlow Hub	*É um repositório de modelos reutilizáveis de Machine Learning treinados que podem ser ajustados e implantados em outros projetos.*	1. (*BERT*) *Bidirectional Encoder Representations from Transformers* – fornece representações vetoriais densas para linguagem natural usando uma rede neural profunda e pré-treinada com a arquitetura *transformer*. 2. *R-CNN* – Modelo para detecção de objetos, treinado com conjunto de imagem (dados coco 2017) com tamanho de lote 64, imagens dimensionadas para resolução 640x640.

Fonte: https://www.tensorflow.org/tensorboard, 2022 (adaptado)

O *TensorFlow* expressa os algoritmos em uma interface capaz de fazer a implementação do aprendizado de máquina desses algoritmos. Conforme (Abadi *et*

al., 2015), o *TensorFlow* consiste em um sistema flexível e pode ser usado para expressar uma ampla variedade de algoritmos, incluindo os de treinamento e inferência para modelos de redes neurais profundas.

Na área da ciência da computação e em outros campos, possui ampla aceitação na condução de pesquisas e implementação de sistemas de aprendizado de máquina, que incluem o reconhecimento de fala, visão computacional, processamento de linguagem natural extração de informação geográfica e etc.

É possível executar os modelos do *TF* diretamente no *Google Colabore,* ambiente desenvolvido pela Google que possibilita a execução de código *python* por meio do navegador utilizando-se dos recursos gratuitos de *GPUs – Graphics Processing Unit*.

Isso possibilita a importação de um conjunto de dados, treiná-lo e avaliar o modelo utilizando se as bibliotecas que proporciona essa implementação com poucas linhas de código.

2.8. Comparativo entre características do *Scikit-Learn* e *TensorFlow*

A Tabela 6 resume as informações encontradas sobre os *frameworks TensorFlow* e *Scikit-Learn*. As informações foram organizadas para facilitar a comparação entre os dois frameworks.

Tabela 6 - Comparação entre *Scikit-Learn* e *TensorFlow*

Característica	***TensorFlow***	***Scikit-Learn***
Nível de abstração	Baixo	Alto
Tipo de aprendizado de máquina	*Deep Learning*	Tradicional
Algoritmos suportados	Redes neurais, programação dinâmica, aprendizado por reforço	Classificação, regressão, agrupamento, redução de dimensionalidade
Popularidade	Mais popular	Menos popular
Flexibilidade	Menos flexível	Mais flexível
Utilização	Grandes conjuntos de dados, problemas complexos	Conjuntos de dados médios, problemas simples
Implementação	Classe base	Estimador base
Exemplos de uso	Deteção de objetos, reconhecimento de voz, tradução automática	Classificação de imagens, regressão linear, agrupamento de dados

Fonte: Adaptada de https://www.simplilearn.com/scikit-learn-vs-tensorflow-article

2.9. Tecnologia de Sensoriamento e Contagem de Objetos

O uso de sensores com diferentes funções como: sensores de térmicos, de presença, contagem e etc. têm crescido bastante, e o emprego estes com outras tecnologias tem proporcionado outros tipos de aplicações, como por exemplo, sensores de temperatura dentro de *tags RFID*.

“A identificação por radiofrequência (*RFID*) usa etiquetas eletrônicas para identificar objetos por meio de tecnologia sem fio (*Wireless*) em distâncias curtas. Ela

promete substituir outras tecnologias de identificação existentes, tais como o código de barras.” (Baltzan, 2016, p. 233)

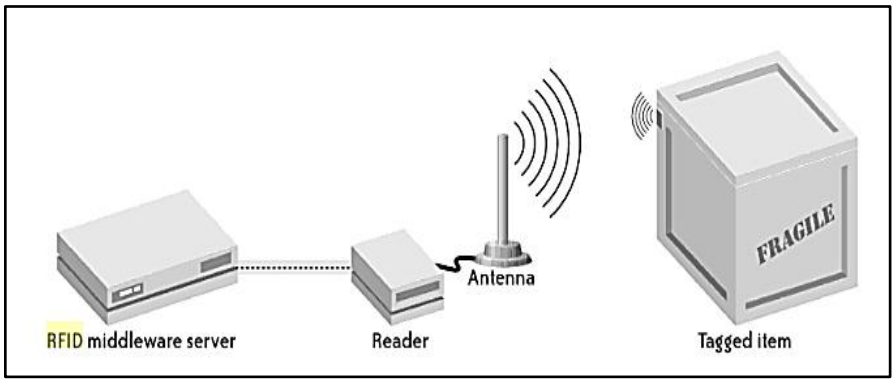
A partir da Figura 7, mostra-se um *Tagged Item* (item com uma etiqueta acoplada) onde, a antena localizada na etiqueta eletrônica, recebe os sinais eletromagnéticos emitidos por um leitor (*Reader*), e transmite as informações armazenadas (para o *RFID middleware server*).

“Ao contrário dos códigos de barras convencionais, que exigem um contato direto entre o código impresso e o leitor (scanner), o sistema *RFID* envia sinais eletromagnéticos e possibilita a identificação à distância. ” (SChinoff; Luz; Aguiar, 2019, p. 113).

As etiquetas podem ser ativas ou passivas, sendo que a ativa possui uma bateria cuja a energia permite que a etiqueta se comunique com o receptor; já a passiva obtém energia por meio do campo eletromagnético criado pela antena.

Com o tamanho reduzido podem ser embutidas em cartões com espessuras de um cartão magnético ou etiquetas de roupas (Gonçalves, 2013).

Figura 7 - Componentes de um sistema RFID



Fonte: (Glover; Bhatt, 2006).

Alguns tipos e aplicações das etiquetas podem ser classificados conforme Gonçalves (2013) descreve a seguir.

Read only (somente leitura) – os dados são gravados na etiqueta uma única vez, na hora da produção e a memória existente no chip não permite novas gravações;

Write-once-read-only (*worm*) – os dados são escritos uma única vez pelo usuário através de equipamento específico;

Read-write – permite gravação e regravação, e com quantidade de memória maior que as outras;

Read-write on board sensor – contém sensor especial que permite gravar dados como temperatura e pressão a que o produto foi submetido;

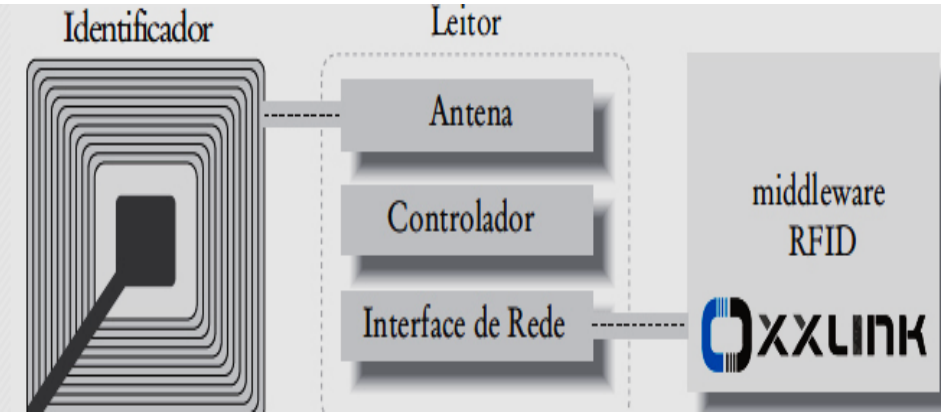
Read-write with integrated transmitters – opera como miniatura de dispositivo de rádio e podem se comunicar com outras etiquetas.

Middleware, ilustrado na Figura 8, é um programa que faz a ponte entre outros softwares. Possibilita integração de algum hardware com aplicações desenvolvidas em diversas linguagens de programação permitindo a independência relativa com dispositivos que fornecem os dados.

Composto por conjunto de processos ou objetos em grupo de computadores que integram entre si de forma a implementar comunicação e oferecer suporte para compartilhamento de recursos a aplicativos distribuídos.

Geralmente utiliza linguagem própria de baixo nível e segue regra de negócio para sua correta integração com banco de dados, isso dificulta algumas vezes o uso da tecnologia *RFID*. (Oxxcode, 2021)

Figura 8 - Middleware na integração de dispositivos *RFID*



Fonte: (Oxxcode, 2021)

Sistema de monitoramento monitora o comportamento e, avisa sobre eventuais falhas em uma rotina executada continuamente, devendo ser tratadas as intercorrências fora dos limites esperados.

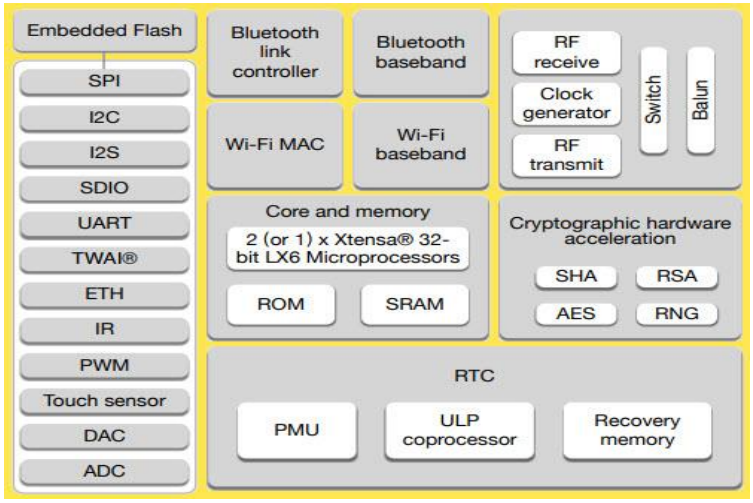
Sistema de predição é utilizado para projeção de dados futuros, de acordo com uma base de dados de acontecimentos passados e coleta de dados do presente (Barbieri, 2017).

2.9.1. Microcontrolador

Com uso frequente em *IoT*, o *ESP32* é um microcontrolador *System on Chip (SoC)* de baixo custo da *Espressif Systems*, desenvolvedores do ESP8266 *SoC*. Este é sucessor do ESP8266 *SoC* e vem em variações single-core e dual-core do microprocessador *Xtensa LX6* de 32 bits da Tensilica com *WI-FI* e *Bluetooth* integrados. (Teja, 2021)

A Figura 9 mostra uma ilustração da composição deste dispositivo. Com o avanço da tecnologia, novas ideias de projetos e implementações específicas surgiram dando início a *Internet* das Coisas ou *IoT*, uma plataforma conectada, onde várias “coisas” ou dispositivos são conectados pela internet para troca de informações.

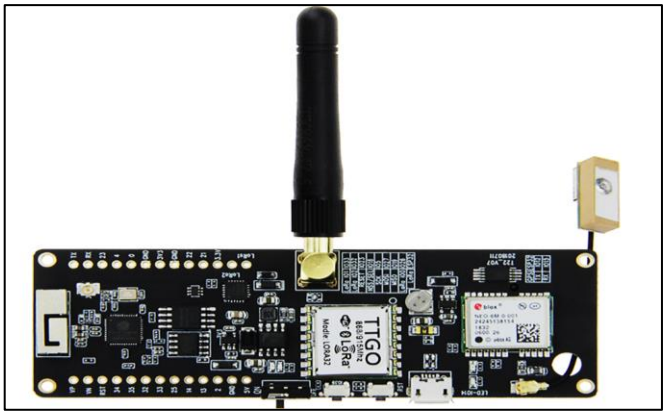
Figura 9 - ESP32S : diagrama de componentes



Fonte: (Teja, 2021)

Os projetos *IoT* são focados principalmente em aplicativos de automação residencial e casa inteligente, e os projetos *IoT* comerciais e industriais têm implementações mais complexas, como aprendizado de máquina, inteligência artificial, redes de sensores sem fio, etc. As implementações que necessitam de conectividade *WI-FI* e com a Internet conseguem facilmente cumprir esse requisito utilizando-se de módulos que agregam a capacidade de sensoriamento, controle, atuação e monitoramento.

Figura 10 - Módulo ESP32 LILYGO



Fonte: http://www.lilygo.cn/prod_view.aspx?TypeId=50060&Id=1237&FId=t3:50060:3

O *ESP32 LILYGO TTGO T- Beam*, Figura 10, traz componentes de Rádio Frequência integrados, amplificador de potência, amplificador de recepção de baixo ruído, interruptor de antena, tecnologia *LoRa* e *WI-FI*. A plataforma possibilita a implementação de projeto de sensoriamento e monitoramento mais eficiente, pois são necessários poucos componentes externos.

2.9.2. *Wireless Sensor Networks Technology – WSNT*

Na Tecnologia de Redes de Sensores sem fio (*WSNT*), os sensores inteligentes não apenas observam os processos e o ambiente, mas também processam os dados medidos e transferem os resultados obtidos.

Isso permite o monitoramento e / ou controle de sistemas complexos concentrados em áreas específicas, que podem servir como um sistema de alerta precoce. (Moller, 2016)

Um exemplo de tal aplicação é a detecção inteligente de vibração em turbinas eólicas em sistemas de energia regenerativa que desligam o sistema se as vibrações da torre atingirem altas amplitudes perigosas. Com o uso de comunicação sem fio, sensores inteligentes podem ser conectados diretamente a redes de sensores, por exemplo, para monitorar as turbinas eólicas do sistema de energia regenerativa.

Consequentemente, os sensores interconectados sem fio têm uma grande vantagem em condições ambientais difíceis, onde as conexões do sensor com fio podem ser evitadas.

Segundo Moller (2016), os *WSNs – Wireless sensor network* – são sensores autônomos espacialmente distribuídos que monitoram as condições físicas e/ ou ambientais para enviar cooperativamente os dados medidos através da rede para um local principal.

Sensores desse tipo mais avançados também permitem o controle de sua atividade, por isso são conhecidos como sensores bidirecionais. Pela característica de acessar informações e tarefas de qualquer lugar e a qualquer hora, essa rede pode ser apresentada como rede de comunicação ubíqua seguindo o paradigma da *IoT*.

Os domínios de aplicação das redes de sensores sem fio são múltiplos, conforme mencionado. Alguns exemplos são:

- Monitoramento inteligente de estradas: permite que todos os carros interajam uns com os outros, o que pode ajudar a prevenir: acidentes de carro, ficar preso no congestionamento do trânsito, dirigir muito rápido em zonas de escola ou hospital.
- Monitoramento de eventos esportivos: permite: monitoramento de parâmetros vitais de corredores, supervisionar uma linha de destino, rastreamento de bola em jogos de futebol.
- Monitoramento da gestão de resíduos: monitorar o processo de decomposição de um depósito de resíduos em relação à temperatura interna, reações químicas, etc., para reagir caso seja atingido um nível crítico. As aplicações de monitoramento mais gerais incluem:
    a. Monitoramento ambiental interno / externo
    b. Monitoramento de saúde e bem-estar

c. Monitoramento de energia
d. Monitoramento de localização de estoque
e. Monitoramento de automação de fábrica e processo
f. Monitoramento de eventos sísmicos e estruturais

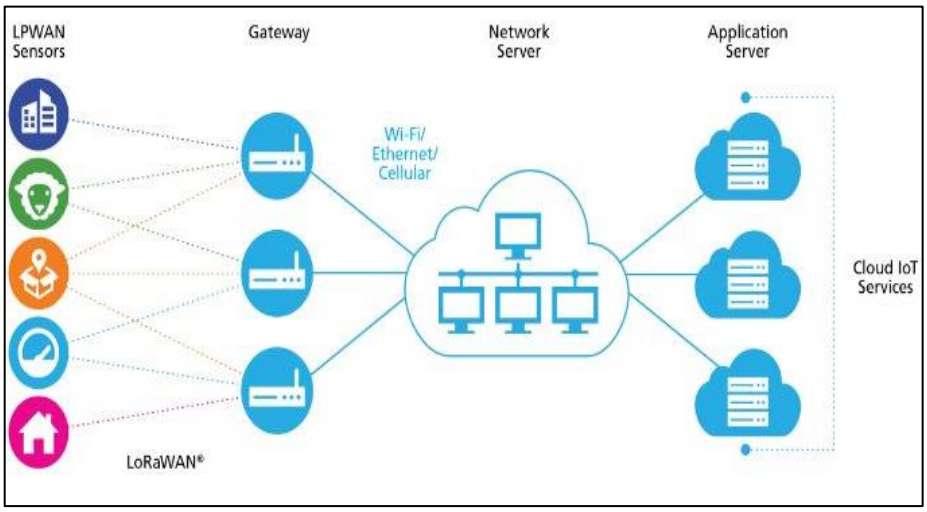
Além de monitorar aplicativos de rastreamento em redes de sensores sem fio, o rastreamento também pode incluir: animais, humanos, objetos e veículos.

2.9.3. Tecnologia *LoRa* (*Short for Long Range*) e *LoRaWAN*

LoRa é uma técnica de modulação sem fio para fornecer interface de transmissão de dados de longa distância.

Segundo o site da *Semtech*, multinacional fabricante de semicondutores -- *semtech.com*, mantenedora dessa tecnologia, o *LoRa* é uma plataforma para a *Internet das coisas* (*IoT*). Um modelo de arquitetura *LoRaWAN* é apresentado na Figura 11 onde uma rede de dispositivos *IoT* passa por *gateway*s que encaminham dados através da *Internet* aos servidores.

Figura 11 - Arquitetura *LoRaWAN*



Fonte: https://LoRa-developers.semtech.com/uploads/documents/images/Semtech-LoRaWAN-Diagram-NetworkArchitecture-Horiz-01.jpg

O padrão *LoRaWAN*, conforme informa o site *teleco.com.br/ipwap.asp*, foi desenvolvido pela *Semtech* e os proprietários da tecnologia do chip são a *IBM Research* e a *Actility*, fazendo parte da *LoRa* a *Cisco, KPN, Orange, ZTE* entre outros.

O dispositivo com esta tecnologia utiliza-se do protocolo *LoRaWAN*, padrão de rede de longa distância de baixa potência (*LWPAN*). O circuito integrado (CI) possibilita a eliminação dos repetidores, prolonga vida útil da bateria, reduz custos e melhora a capacidade da rede.

Este são usados internacionalmente para aplicações em residências inteligentes, edifícios, cidades, cadeias de suprimentos e logísticas, medição em serviços públicos, agricultura e entre outros.

A partir dos servidores são disponibilizados para armazenamento, tratamento e análise, entre diversas outras aplicações.

Os componentes atrelados a essa tecnologia incluem: *gateway*s, transceptores, receptores e transmissores que cobrem o espectro de radiofrequência da banda Industrial, Científica e Médica (*ISM - Industrial, Scientific and medical band*) de *sub-GHz* até 2,4 GHz.

Os transceptores *FSK* (*Frequency Shift Keying*), isto é, chaveamento por desvio de frequência, podem operar na banda ISM não licenciada de 433, 868 e 915 MHz.O *LoRaWAN* atende aos principais requisitos da *IoT*, como comunicação bidirecional, segurança de ponta a ponta, mobilidade e serviços de geolocalização. Os operadores de redes que implantam redes públicas geralmente usam uma abordagem hibrida combinando *LoRaWAN* com tecnologias celulares parra atender às necessidades de diferentes casos de uso.

Conforme informado o site da *Semtech*, o ecossistema *LoRaWAN* inclui 170 operadores de rede em vários continentes capaz de conectar vários dispositivos tornando a interoperabilidades entres eles mais fácil de implantação imediata e soluções eficientes. (Semtech Corporation, 2023)

Alguns dispositivos, podem funcionar na configuração mestre-escravo, sendo que vários dispositivos de mesmo modelo podem ser configurados como nós que transmitem entre si os dados dos sensores até chegar a um nó principal (*gateway*) conectado à Internet conforme.

2.10. *API* de previsão climática

As *APIs* de condições climáticas são ferramentas essenciais para aplicações que exigem dados meteorológicos precisos e atualizados. Essas *APIs* coletam e processam dados meteorológicos de diversas fontes, como modelos meteorológicos globais e locais, satélites, radares e uma vasta rede de estações meteorológicas. Os dados estão disponíveis em formato *JSON, XML ou HTML*, para que possam ser usados em qualquer aplicação. O acesso pode depender de um cadastro no site e geração de uma chave (*key*) usada para verificação e autenticação do usuário cadastrado. Algumas *APIs* encontrada durante a pesquisa são apresentadas a seguir:

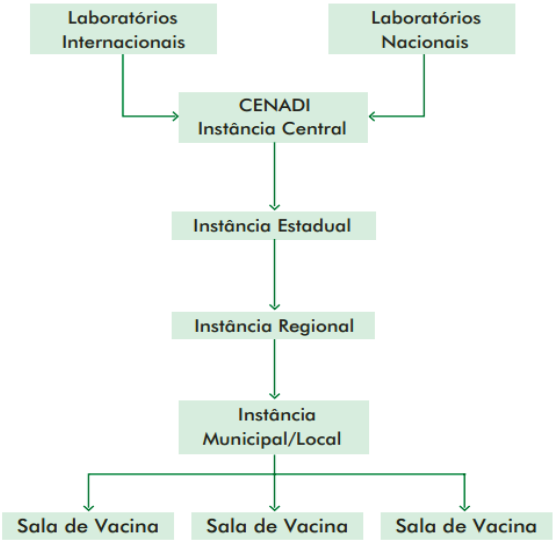
- *API* do INMET - É uma *API* desenvolvida pelo Instituto Nacional de Meteorologia (INMET) que fornece dados meteorológicos para todo o território brasileiro. Os dados incluem temperatura, umidade, precipitação, vento e outros. A *API* do INMET é gratuita para uso pessoal ou educacional. (INMET, 2023)

- *API* do CPTEC/INPE - É uma *API* desenvolvida pelo Centro de Previsão de Tempo e Estudos Climáticos (CPTEC) do Instituto Nacional de Pesquisas Espaciais (INPE) que fornece dados meteorológicos e climáticos para todo o território brasileiro. Os dados incluem temperatura, umidade, precipitação, vento, ondas, marés e outros. A *API* do CPTEC/INPE é gratuita para uso pessoal ou educacional. (CPTEC/INPE, 2023)
- *OpenWeather*, é um conjunto de *APIs* amplamente reconhecíveis. Alimentado por soluções de aprendizado de máquina convolucional. Ela fornece informações meteorológicas em tempo real e históricos para qualquer lugar do mundo. Os dados incluem temperatura, umidade, precipitação, vento e outros. (OpenWeatherMap, 2023)

2.11. Cadeia de Frios do PNI

Conforme Ministério da Saúde (2013), em o manual de Rede de Frio do Programa Nacional de Imunização - PNI, cadeia de frio, Figura 12 , é um processo que exige cuidados e que considera alta a sensibilidade dos imunizantes (vacinas) às alterações de temperatura de conservação somados a possíveis desvios de fluxo das atividades de armazenamento, distribuição, e aos aspectos de impacto direto na segurança e qualidade dos produtos destinados às ações de vacinação.

Figura 12 - Cadeia de Frios



Fonte: (Ministério da Saúde, 2022)

O manual da Rede de Frio considera os termômetros como os instrumentos de medição de temperatura mais utilizados na Rede de Frio, além desses, outros modelos (digitais, infravermelho, de registros, etc.) com diferentes princípios de funcionamento são utilizados para medir e monitorar as variações desta grandeza nos ambientes de armazenamento, nos equipamentos frigoríficos e nas caixas térmicas.

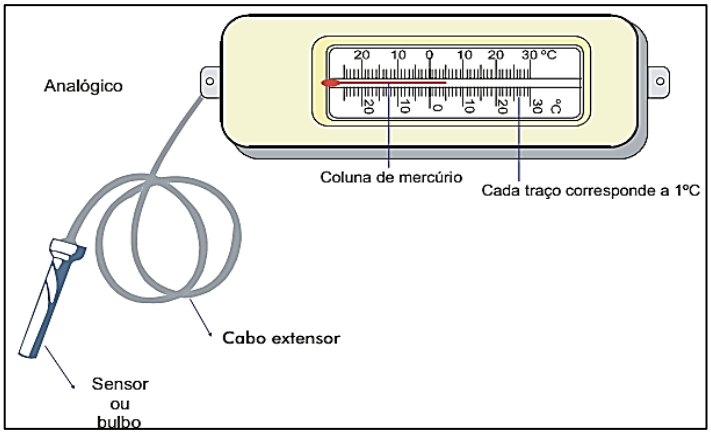
A Instância Local (Sala de Vacina) da rede faz parte do fluxo de distribuição de imunobiológicos que se estende desde a instância Nacional, Estadual, Regional e Municipal.

A sala de vacinação destina-se às atividades operacionais de vacinação, onde acontece o contato direto com o usuário final para os eventos de campanhas vacinais, bloqueios e intensificações.

O MS recomenda diversas formas de refrigeração para minimizar os riscos indesejados de alterações de temperatura nas salas de vacina.

Consta do Manual da Rede de Frio, a divisão da cadeia, nos seguintes elementos: 1-Equipe Técnica, 2-Equipamentos, 3-Instâncias de armazenamento, 4-Transporte entre instâncias, 5-Controle de Temperatura, 6-Financiamento.

Figura 13 - Termômetro analógico de cabo extensor utilizados em caixas térmicas



Fonte: (Ministério da Saúde, 2022)

A instância de armazenamento local é onde os imunobiológicos são utilizados, isto é, na sala de vacinação da unidade de saúde ou em local devidamente apropriado. Nela são usados os termômetros para medição de temperatura, como no exemplo mostrado na Figura 13.

Para o MS os imunobiológicos devem ser mantidos em temperatura de +2ºC, utilizando-se geladeiras domésticas com capacidade mínima de 280 litros. A temperatura da geladeira deverá ser regulada a +2°C, devendo, porém, oscilar normalmente entre +2ºC e +4ºC. A ocorrência de oscilação entre +2ºC e +8ºC quando esporadicamente, não compromete a qualidade do imunobiológicos.

2.12. Ambiente de Desenvolvimento

Para construção de programas, aplicações *Desktop* ou *Web*, é comum o uso de diversas ferramentas de *softwares* em um ambiente de desenvolvimento.

“Um ambiente de desenvolvimento integrado (*IDE*) é uma aplicação de software que ajuda os programadores a desenvolver código de *software* de maneira eficiente. “ (AWS, 2023)

O objetivo da *IDEs* é aumentar a produtividade do desenvolvedor. É facilitar a combinação de recursos como edição, compilação, teste e empacotamento de software de maneira prática e dinâmica em um só ambiente. Entre as *IDEs* estão:

Visual Studio desenvolvido pela *Microsoft,* oferece suporte a uma ampla gama de linguagens como *C++, C#, Java, Python, PHP, Go, .NET, JavaScript, TypeScript* entre outras. É multiplataforma. (Microsoft, 2022)

IntelliJ IDEA IDE *da JetBrains* para desenvolvimento de *software Java* e desenvolver aplicativos para *Android* e *iOS.* (IntelliJ IDEA 2023.3, 2023)

Eclipse é um *IDE open source* para desenvolvimento de *software*. É usado para desenvolver aplicativos para *Windows, macOS, Linux e Android*. O Eclipse oferece suporte a uma ampla gama de linguagens de programação, incluindo *Java, C++, C#, PHP, Python e JavaScript.* (Eclipse, 2023)

Apache NetBeans é um *IDE* open source para desenvolvimento de *software*. É usado para desenvolver aplicativos para *Windows, macOS, Linux e Android*. O *NetBeans* oferece suporte a uma ampla gama de linguagens de programação, incluindo *Java*, *C++, C#, PHP, Python e JavaScript*. (Netbeans, 2023)

Arduino IDE 2 é um editor de código com muitos recursos. Por meio desta *IDE* é possível instalar bibliotecas de controle de *hardware*, configurar portas de entrada e saídas de dados, além de embarcar código para teste e uso em automação. (ArduinoIDE2, 2023)

Power BI é um serviço da *Microsoft* que permite a criação de painel personalizáveis, relatórios, acesso a bancos de dados e *APIs* em tempo real. Pode ser acessado por aplicativos móveis e desktop. Pode gerar *links* online de *dashboard* para acesso sincronizado. (Microsoft, 2023)

JupyterLab é um ambiente de desenvolvimento interativo baseado na *web* para *notebooks*, código e dados. Permite configurar e organizar fluxos de trabalho em ciência de dados, computação científica, jornalismo computacional e aprendizado de máquina.

Jupyter Notebook é um aplicativo simplificado de criação de *notebook* e faz parte do Projeto *Jupyte*r, centrado no objetivo de fornecer ferramentas e padrões para computação interativa com notebooks computacionais. (Jupyter, 2023)

2.13. *On-premises data gateway - Power BI*

O *gateway* de dados local atua como uma ponte aprovisionando transferência de dados rápida e segura entre dados locais e vários serviços em nuvem da *Microsoft*. É necessário ter uma conta na *Microsoft* para o uso do serviço online.

Além do *Power BI*, outros serviços como *Power Apps*, *Power Automate*, *Azure Analysis Services* e *Azure Logic Apps* são oferecidos. (Microsoft, 2023)

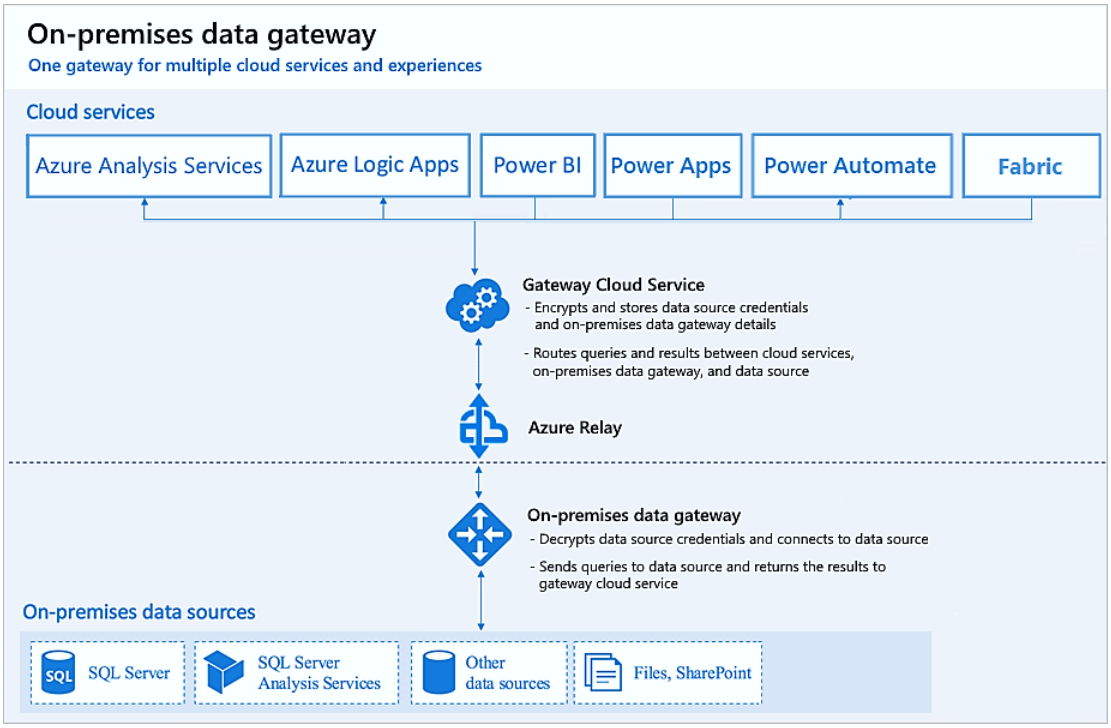
Por meio do *gateway*, as organizações podem manter bancos de dados e outras fontes de dados em suas redes locais enquanto usam esses dados locais com segurança em serviços em nuvem.

A arquitetura o apresenta, Figura 14, mostra os serviços em *On-premises data sources* que é possível ser implementado pelo *On-premises data gateway* ao *cloud services* da *Microsoft.* Um dos serviços possíveis é a sincronização com *Power BI*.

Para fazer a implantação ou instalação do *gateway* localmente é necessário baixar a ferramenta seguindo as recomendações na documentação disponível no site da *Microsoft*.

O *Power BI* oferece área administrativa online onde é possível administrar configurações, baixar ferramentas para instalação local, monitorar *gateway*s, criar, atualizar, editar painéis e dashboards. Tudo isso com a finalidade de gerenciar e disponibilizar relatórios dinâmicos.

Figura 14 - *On-premises data gateway - Power BI*



Fonte: (Microsoft, 2023)

Capítulo 3 – Método Proposto

O objetivo deste capítulo é apresentar o método proposto para o monitoramento de uma caixa térmica na sala de vacina.

O método consiste na criação de um banco de dados que armazena as seguintes informações:

1. Temperatura interna da caixa
2. Geolocalização da caixa
3. Condições climáticas do momento da coleta dos dados
4. Meta diária de vacinados
5. Recomendações relacionadas à meta alcançada

O banco de dados será armazenado em uma nuvem pública, para que possa ser acessado de qualquer lugar.

As informações armazenadas no banco de dados serão utilizadas para:

- Monitorar a temperatura interna da caixa;
- Monitorar a quantidade de frascos coletados;
- Monitorar a geolocalização da caixa;
- Monitorar condições climáticas como temperatura média do dia; se está chovendo ou fazendo sol, etc.
- Gerar gráficos e recomendações da meta alcançada considerando horários específicos de 7 às 17H;

Um diagrama em bloco é mostrado na Figura 15 representando a arquitetura da solução proposta.

A solução utiliza-se de artefatos *IoT* com capacidade de comunicação sem fio entre os dispositivos e comunicação com bancos de dados através da *Interne*t.

3.1. Estudo comparativo das tecnologias

Nesta subseção, apresentamos tabelas comparativas de algumas tecnologias consideradas para a solução proposta.

3.1.1. Comparação entre dispositivos

Na Tabela 7, juntamente com suas especificações e preços são apresentados alguns dispositivos que possuem comunicação sem fio e função de gateway.

Tabela 7 – Dispositivos Gateway especificações

Nome do produto e marca	Tipo de comunicação	Frequência *LoRA*	Tipo de dispositivo	Outros detalhes
Placa de Desenvolvimento *ESP32 LoRa*	*WiFi, Bluetooth, USB, LoRA*	868-915MHz	*Gateway*	Suporte a bateria
TTGO T-Beam ESP32 LoRa	*WiFi, Bluetooth, USB, LoRA*	868MHz-915MHz	*Gateway*	Suporte a bateria, *GPS*
Waveshare SX1303 868M LoRaWAN Gateway HAT	*WiFi, Bluetooth, USB, LoRa*	868MHz	*Gateway*	Suporte a bateria, *GPS*
EchoFlow GW019158CH	*LoRa, WI-FI, LTE, Ethernet*	915MHz	*Gateway*	Até 5 horas de trabalho autônomo com bateria
TAD208NH8-915	*LoRa, WI-FI, Ethernet, 3G, 4G*	915 MHz	*Gateway*	Capacidade:10.000 dispositivos nós

Fonte: o autor

Na Tabela 8 são apresentados alguns dispositivos para leituras ou detecção de objeto por rádio frequência (*RFID)*.

Tabela 8 – Dispositivos com tecnologia *RFID*

Nome do produto e marca	Modelo	Frequência	Distância de leitura	Fonte de alimentação	Tensão de funcionamento	Temperatura de funcionamento
Leitor *RFID Mfrc522* Arduino	*MFR-C522*	13,56 MHz	0-60 mm	5V	3,3 V	-20°C ~ +70℃
Leitor com ID de alta frequência	*R65D*	125 kHz	0-8 cm	0,5 W	5 V 10 mA	-20°C ~ +70℃
ACR122U	*ACR122U*	13,56 MHz	0-50 mm	5V	5V DC, 50 a 200mA	0°C ~ +50℃
UHF RFID Reader	*TIMMY*	860-960 MHz	10-15 metros	12V	-	-20°C ~ +70℃

Fonte: o autor

As variáveis de principal interesse para a criação de uma implementação de baixo custo são a frequência, distância de leitura, tensão de funcionamento e preço.

Para atender a esses requisitos, foram escolhidos dispositivos com baixo consumo de potência e preços acessíveis, com capacidade de transmissão sem fio de dados e à distância de pelo menos 500 metros. Além disso, os dispositivos devem ter *GPS* e *WI-FI*.

Outras especificações também são apresentadas, como na Tabela 9, onde descreve-se uma composição básica dos dispositivos para um sistema.

Tabela 9 – Outras especificações de dispositivo necessário para solução

Tipo	Modelo	Frequência	Distância	Fonte de alimentação
Leitor RFID	MFR-C522	13,56 MHz	0-60 mm	5V
Tag RFID	Mifare 1 S50	13,56 MHz	0-10 cm	-
GPS	TinyGPS++	1,5 GHz	20 km	-
WI-FI	ESP32	2,4 GHz	100 m	3,3 V
LoRaWAN Gateway	*LoRaWAN* Gateway	915 MHz	10 km	5V
Sensor Temperatura	DS18B20, Precisão: ±0.5°C	-	-	3,3 - 5V

Fonte: o autor

Na Tabela 10 mostra-se os dispositivos escolhidos para implementação da solução.

Tabela 10 - Conjunto de principais dispositivos escolhidos para compor a solução

Tipo	Modelo	Frequência	Distância	Fonte de alimentação
Leitor RFID	*MFR-C522*	13,56 MHz	0-60 mm	5V
Tag RFID	*Mifare 1 S50*	13,56 MHz	0-10 cm	Passiva
LoRa Gateway	*TTGO T-Beam*	915 MHz	10 km	3,3 -5V
Sensor Temperatura	DS18B20, Precisão: ±0.5°C	-	-	3,3 - 5V

Fonte: o autor

3.1.2. Comparação entre os *frameworks* de *Machine Learning*

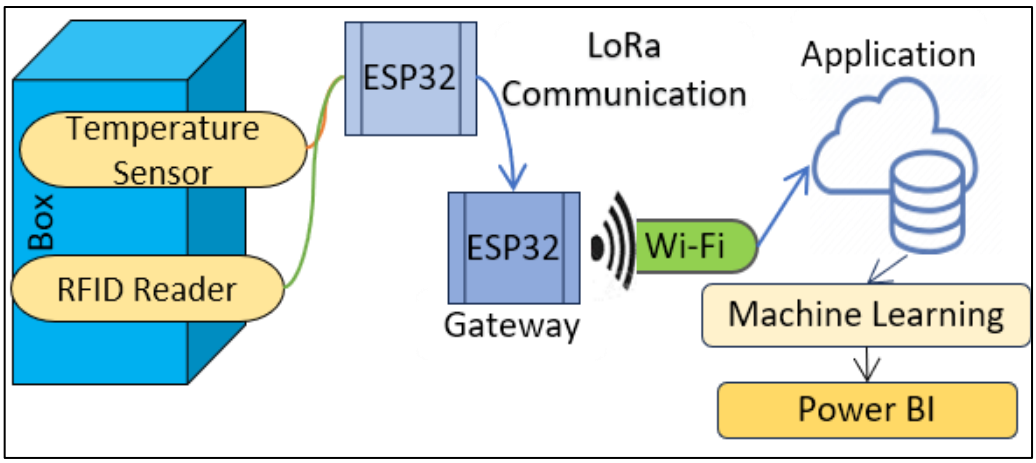
Conforme o comparativo em tabela apresentado na subseção 2.8, observar-se que os dois *frameworks* possuem algumas particularidades que os direcionam para determinadas aplicações.

Considerando que a base de dados utilizadas neste experimento é menor que 1200 registros e no máximo 10 colunas e que, trata-se do treinamento de modelos para classificação de rótulos (*labels*), será utilizado o *framework Scikit-Learn* para implementação e avaliação dos modelos e consequentemente a escolha de um.

3.2. Arquitetura

A primeira parte do diagrama é responsável pela coleta e armazenamento de dados (temperatura, geolocalização e valor da tag *RFID*). A segunda parte é responsável pela recuperação dos dados e iniciação do processo de *ML* (fases de exploração, análise, previsão e exibição de informações).

Figura 15 - Arquitetura proposta



Fonte: o autor

3.3. Bibliotecas e Materiais utilizados

Alguns componentes mencionados aqui podem ser vistos na Figura 16. Um sensor térmico (*DS18B20* - faixa de medição: -55 a +125 °C) foi usado para monitorar a temperatura.

Para a detecção de etiquetas, foi usado o leitor *RFID-RC522*, em que, de acordo com o padrão *ISO/IEC* 14443, o circuito integrado (IC) é capaz de ler e gravar em "cartões de proximidade" em uma frequência de comunicação entre 13,56 (MHz) e mais ou menos 7 (kHz). As etiquetas representam frascos de vacina.

Também foi usado o *ESP32*, *LILYGO TTGO T-Beam*, que apresenta vários componentes integrados como amplificador de potência, amplificador de recepção de baixo ruído, chave de antena, tecnologia *GPS*, *LoRa* e *WI-FI*.

A *LoRa* atende aos principais requisitos da *IoT*, como comunicação bidirecional, segurança de ponta a ponta, mobilidade e serviços de geolocalização.

As operadoras de rede que implantam redes públicas geralmente usam uma abordagem híbrida que combina *LoRa* com tecnologias celulares para atender às necessidades de diferentes casos de uso. (Semtech Corporation 2022)

Para programar o *ESP32*, foram usados o *Arduino IDE*, as bibliotecas *LoRa*, *Wire*, *SPI*, *WI-FI*, *MySQL_Connection*, *OneWire* e *DallasTemperature* (*Sensor-DS18B20*), *MFRC522* (Leitor *RFID*).

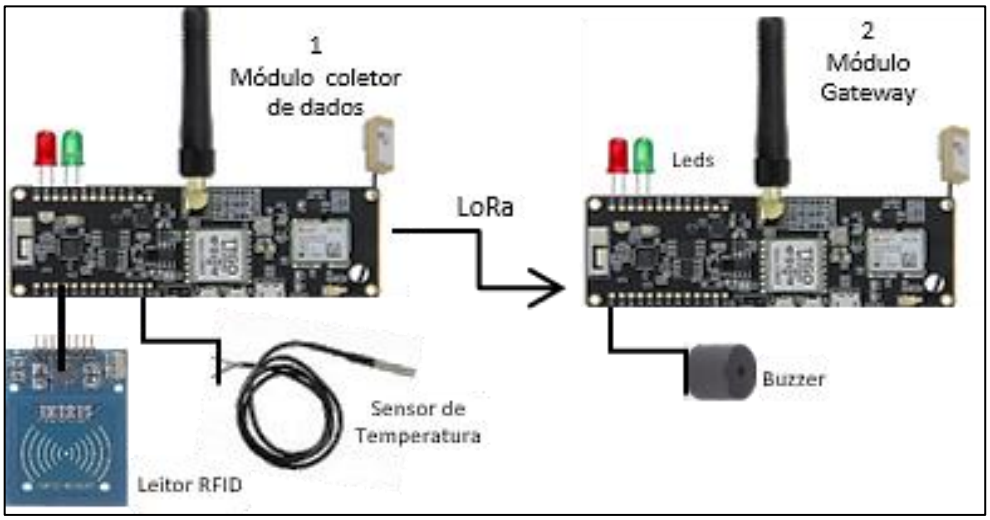
Para o Ambiente de desenvolvimento utilizou-se a extensão do *Jupyter Notebook* dentro do *VS. Code, Power BI On-premises data gateway* entre outros e Python para recuperação do modelo de *ML* treinado e exibição de um dashboard.

3.4. Implementação e funcionamento do protótipo

Cada módulo foi acomodado em caixa plástica para evitar danos após a montagem dos componentes externos.

O módulo coletor de dados, conforme apresentados na Figura 16, é composto por um microcontrolador, um sensor de temperatura e um sensor de detecção de objetos, dois leds. O módulo *gateway*, foi acrescentado apenas os dois *leds* e um *buzzer* (atuador sonoro). O *ESP32 TTGO* já vem com os Módulo GPS e Módulo *LoRa* na sua composição.

Figura 16 - Módulos de aquisição e transmissão de dados



Fonte: o Autor

Criou-se um cenário simulando-se uma campanha de vacinação. Foram utilizados dois dispositivos microcontroladores *ESP32*, um sensor de temperatura, os módulos integrados de Sistema de Posicionamento Global (GPS), um sensor de detecção de objetos, quatro leds e um atuador sonoro.

3.4.1. Procedimento Simulado Manualmente

Quando uma *tag* é aproximada do leitor *RFID*, o microcontrolador-1 a detecta e lê os dados da *tag*. Em seguida, o microcontrolador-1 envia um comando via *LoRa* para o microcontrolador-2. O microcontrolador-2, que já está conectado a uma rede *WI-FI*, recebe os dados da tag e os envia para um banco remoto (em nuvem).

Dessa forma, conforme fluxo ilustrado na Figura 17, após o método "*RFID_check()*" receber um "*true*" confirmando a detecção da *tag*, *Card present* =Y, o método "*readTag()*" ler os dados da etiqueta, enquanto o "*temperature()*" registra a

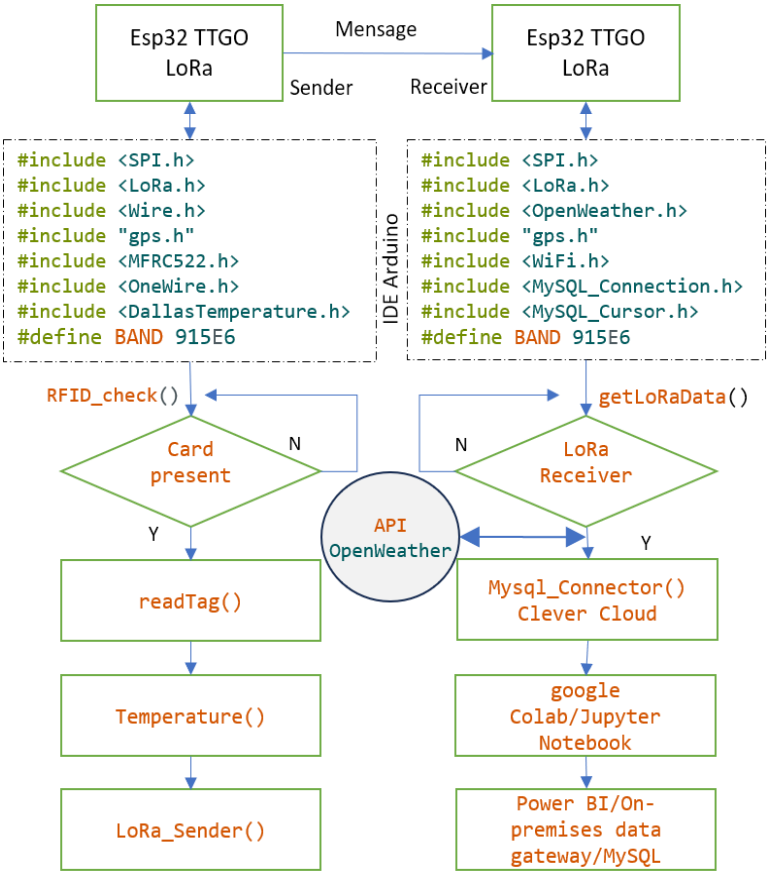
temperatura do interior da caixa. Em seguida o método *LoRa_Sender()* inicia o envio dos dados via *LoRa*.

No *gateway*, o método *getLoRaData()* fica aguardando até que uma nova mensagem seja detectada. A partir dessa detecção, os dados são armazenados em variáveis e enviados ao *Clever Cloud* via *Mysql_Connector*.

Os dados enviados ao banco são: *cx_id, tag_id, temperature, lat, lon, rssi* e dados climáticos que correspondem a: identificador da caixa, identificador da *tag*, temperatura em graus *Celsius*, latitude e longitude, *rssi, main_weather, temp_weather* e *humidity_weather* respectivamente.

Foram realizadas várias inserções de dados passando as etiquetas *RFID* no leitor, entre o período de 07 às 17H. Esse procedimento é necessário para criar uma 'base de dados que servirá de modelo para elaboração de novas linhas no banco.

Figura 17 - Fluxo dos métodos no código fonte durante envio de mensagens e etapas subsequentes de armazenamento.



Fonte: o autor

Por meio da latitude e longitude é feito a busca das condições climática à *API OpenWeather*. Assim, cada linha inserida no banco de dados, significa que um frasco foi esvaziado e que as condições climáticas são obtidas no instante da inserção.

3.4.2. Recuperação e criação do modelo de *ML*

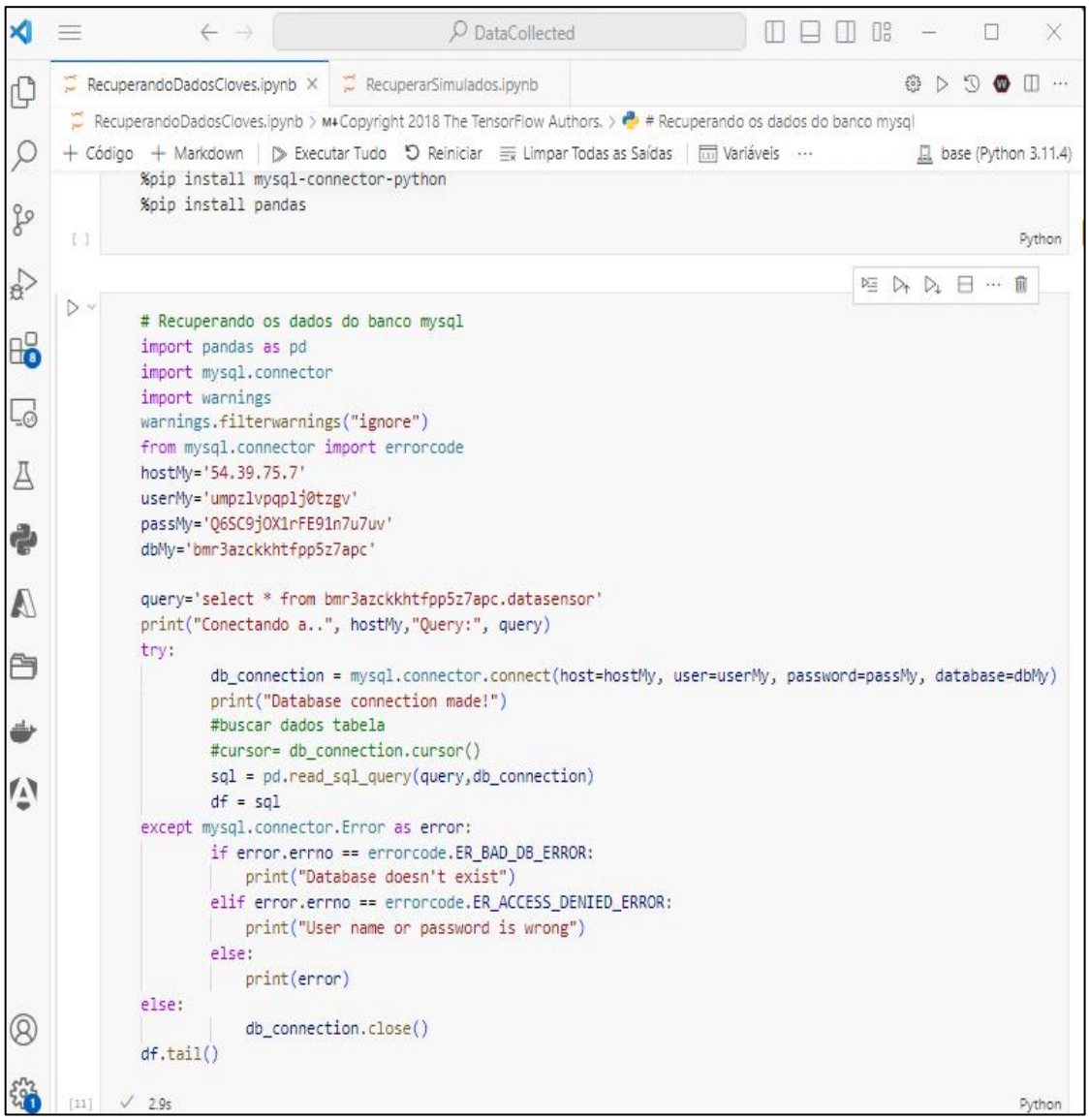
Os dados foram recuperados por meio do ambiente do *VS. Code*, onde foi possível usar a biblioteca *mysql-connector-python.*

Para os procedimentos iniciais de recuperação e exploração dos dados utilizou-se a biblioteca *pandas*.

Todos os procedimentos foram desenvolvidos na plataforma *Windows*. Foram instaladas no *VS. Code* as extensões do *Jupyter Notebook*, *Jupyter Notebook Renderers, Python Extension Pack,* entre outras.

Na Figura 18, é apresentado o código em *python* por meio do qual iniciou-se a recuperação dos dados.

Figura 18 – Ambiente de desenvolvimento *VS. Code*



Fonte: O autor

Na Figura 19, um *dataframe* (tabela) gerado pela biblioteca *pandas.* O *dataframe* foi convertido para o formato *CSV (comma-separated-values)* para ser utilizado na criação do modelo de *Machine Learning.*

Figura 19 - Dados recuperado do Banco *MySQL* (*Clever Cloud*)

	id	cx_id	temperature	tag_id	dateCollection	lat	lon	rssi	main_weather	temp_weather
1412	1413	1	6.56	4325C418	2023-10-09 16:44:30	-3.024054	-59.999485	-54	storm	25.39
1413	1414	1	6.54	4325C418	2023-10-09 16:48:22	-3.024054	-59.999485	-46	storm	20.74
1414	1415	1	6.46	4325C418	2023-10-09 16:52:09	-3.024054	-59.999485	-60	fog	31.28
1415	1416	1	7.15	4325C418	2023-10-09 16:55:59	-3.024054	-59.999485	-56	clear	28.86
1416	1417	1	6.72	4325C418	2023-10-09 16:59:45	-3.024054	-59.999485	-57	cloudy	30.25

```
df_db = df.to_csv(f'collection.csv')
df_db
```

Python

Fonte: O autor

3.4.3. Exploração e separação dos dados

Antes de treinar o modelo foi necessário submeter o *dataframe* recuperado a filtros e remoção de algumas colunas, além do aumento de registro com base nos dados já existentes.

Este procedimento foi necessário para deixar a *df* mais robusto e com informações bem mais parecidas com o desenrolar do atendimento em uma sala de vacinação. Os seguintes procedimentos são apresentados em seguida.

Figura 20 - Filtros realizados na base de dados

```
df = pd.read_csv('csv\collection.csv',).set_index('dateCollection')
df.index = pd.to_datetime(df.index)
df = df.drop(['Unnamed: 0'],axis=1)
df['id_gateway']=1
df.head(8)
```

Python

```
#Copiando o dataframe para b1 apenas com os campos relevantes para o modelo

b1 = df[['main_weather', 'temp_weather', 'humidity_weather']].copy()

b1= b1[(b1.index < '2023-10-01 07:12:09') | (b1.index > '2023-10-01 07:45:00')]
b1= b1[(b1.index < '2023-10-01 10:23:09') | (b1.index > '2023-10-01 10:44:00')]
b1= b1[(b1.index < '2023-10-01 11:03:09') | (b1.index > '2023-10-01 11:33:00')]
b1= b1[(b1.index < '2023-10-01 15:12:09') | (b1.index > '2023-10-01 15:39:00')]
b1= b1[(b1.index < '2023-10-02 07:15:00') | (b1.index > '2023-10-02 07:45:00')]
b1= b1[(b1.index < '2023-10-02 08:20:00') | (b1.index > '2023-10-02 08:50:00')]
b1= b1[(b1.index < '2023-10-02 09:09:00') | (b1.index > '2023-10-02 10:30:00')]
b1= b1[(b1.index < '2023-10-02 11:01:00') | (b1.index > '2023-10-02 11:00:00')]
b1= b1[(b1.index < '2023-10-02 11:56:00') | (b1.index > '2023-10-02 12:30:00')]
```

Fonte: O autor

Figura 21 - Criando a lógica para coluna "redirecionar"

```python
#Dias com redirecionamento "sim" '2023-10-06' 2,4,5,8
#Dia com redirecionamento "não" '2023-10-06' 1,3,6,7,9
margem_aceitacao = 0.968  # 97,2% para padrão
# Calcular a meta alvo para cada período
limite_ate_11 = (40)# Limite de aceitação para 11H dia
limite_ate_12 = (50)# Limite de aceitação para 12H dia
limite_ate_13 = (60)  # Limite de aceitação para 13H dia
limite_ate_14 = (70)  # Limite de aceitação para 14H dia
limite_ate_15 = (80)  # Limite de aceitação para 15H dia
limite_ate_16 = (85)  # Limite de aceitação para 16H dia
limite_ate_17 = (95)  # Limite de aceitação para 17H dia

# Função para determinar o status 'redirecionar' com base na 'meta_alcanc'
def determinar_status(meta_alcanc, margem_aceitacao, limite_aceitacao):
    if meta_alcanc < (margem_aceitacao * limite_aceitacao) and 14 <= idx.hour <= 17:
        return 'sim'
    elif meta_alcanc < (margem_aceitacao * limite_aceitacao):
        return 'atenção'
    elif meta_alcanc >= (margem_aceitacao * limite_aceitacao) and meta_alcanc <= limite_aceitacao:
        return 'aceitável'
    else:
        return 'não'

# Verificação do status 'redirecionar' para diferentes horários
for idx in b2.index:
    if 7 <= idx.hour < 11:
        b2.loc[idx, 'redirecionar'] = 'não'
    elif 11 <= idx.hour < 12:
        b2.loc[idx, 'redirecionar'] = determinar_status(b2.loc[idx, 'meta_alcanc'], margem_aceitacao, limite_ate_11)
    elif 12 <= idx.hour < 13:
        b2.loc[idx, 'redirecionar'] = determinar_status(b2.loc[idx, 'meta_alcanc'], margem_aceitacao, limite_ate_12)
    elif 13 <= idx.hour < 14:
        b2.loc[idx, 'redirecionar'] = determinar_status(b2.loc[idx, 'meta_alcanc'], margem_aceitacao, limite_ate_13)
    elif 14 <= idx.hour < 15:
        b2.loc[idx, 'redirecionar'] = determinar_status(b2.loc[idx, 'meta_alcanc'], margem_aceitacao, limite_ate_14)
    elif 15 <= idx.hour < 16:
        b2.loc[idx, 'redirecionar'] = determinar_status(b2.loc[idx, 'meta_alcanc'], margem_aceitacao, limite_ate_15)
    elif 15 <= idx.hour < 16:
        b2.loc[idx, 'redirecionar'] = determinar_status(b2.loc[idx, 'meta_alcanc'], margem_aceitacao, limite_ate_16)
    elif 16 <= idx.hour < 17:
        b2.loc[idx, 'redirecionar'] = determinar_status(b2.loc[idx, 'meta_alcanc'], margem_aceitacao, limite_ate_17)
    else:
        b2.loc[idx, 'redirecionar'] = 'não'  # ou qualquer outra ação necessária
```

✓ 0.0s Python

Fonte: O autor

Na Figura 20 consta o código de recuperação do *dataframe* “*collection.csv*”, onde o mesmo é atribuído a variável *df* e ainda, na mesma figura, é feito uma cópia para um novo *dataframe* ‘b1’ que recebe apenas as colunas *‘dateCollection’*, *'main_weather', 'temp_weather', 'humidity_weather', ‘meta_alcanc’,* e que posteriormente recebe a coluna *‘redirecionar’* com os valores de ‘*aceitável*’, ‘*atenção’*, ’*não*’ e ‘*sim*’, conforme visto nas Figura 21 e, na Figura 22.

O novo dataframe salvo, ‘*dataset.csv’,* contém os dados para criação do modelo. Sua estrutura é apresentada na Figura 23.

Figura 22 - Acrescentado coluna meta_alcanc, dose e acumulado

```
# Converta o índice do DataFrame para o tipo de data
meta= 300
# Copie o DataFrame para b1
b1 = df[['main_weather', 'temp_weather', 'humidity_weather']].copy()
b1['dose'] = 5

# Use groupby por dia e calcule o valor acumulado e o percentual
# da meta atingida para cada dia
b1['acumulado'] = b1.groupby(b1.index.date)['dose'].cumsum()
b1['meta_alcanc'] = round((b1['acumulado'] / meta) * 100, 2)
```

✓ 0.1s Python

dateCollection	main_weather	temp_weather	humidity_weather	dose	acumulado	meta_alcanc
2023-10-01 07:03:03	clear	28.48	79.83	5	5	1.67
2023-10-01 07:05:08	clear	28.09	77.62	5	10	3.33
2023-10-01 07:10:51	clear	30.35	67.84	5	15	5.00
2023-10-01 07:15:00	clear	35.30	80.67	5	20	6.67
2023-10-01 07:18:34	clear	32.54	52.87	5	25	8.33
2023-10-01 07:22:02	clear	29.61	61.76	5	30	10.00

Fonte: O autor

Figura 23 – *Dataframe* para treinamento do modelo de *Machine Learning*

dateCollection	main_weather	temp_weather	humidity_weather	meta_alcanc	redirecionar
2023-10-04 16:03:06	smoke	30.48	73.40	80.83	sim
2023-10-04 16:07:03	smoke	28.95	80.76	81.67	sim
2023-10-04 16:33:05	smoke	30.33	75.33	82.50	sim
2023-10-04 16:37:23	smoke	30.31	77.82	83.33	sim
2023-10-04 16:41:01	rain	22.36	64.21	84.17	sim
2023-10-04 16:45:04	cloudy	30.58	82.05	85.00	sim
2023-10-04 16:48:37	fog	28.07	72.80	85.83	sim
2023-10-04 16:52:30	cloudy	29.14	79.17	86.67	sim
2023-10-04 16:55:53	storm	27.37	62.79	87.50	sim
2023-10-04 16:59:48	clear	32.06	67.46	88.33	sim

Fonte: O autor

3.4.4. Funções de Pré-processamento dos dados

As funções criadas têm o objetivo de preparar os dados para a fase de treinamento e teste.

Figura 24 - Funções de conversões de tipos de dados utilizando o *LabelEncoder*.

```
#Funções de conversões de tipo
lb = LabelEncoder()
variaveis_numericas = []
variaveis_categoricas = []
variaveis_datas = []
nu = []
cat= []
def numericas(dataframe1):

        for i in dataframe1.columns[0:len(dataframe1)].tolist():
        # print(df.dtypes[i])
            if dataframe1.dtypes[i] == 'int64' or dataframe1.dtypes[i] == 'float64':
                variaveis_numericas.append(i)
            variaveis_numericas
        for var in variaveis_numericas:
            dataframe1[var] = lb.fit_transform(dataframe1[var])
            nu.append(lb.classes_)

def categoricas(dataframe2):

        for i in dataframe2.columns[0:len(dataframe2)].tolist():
            if dataframe2.dtypes[i] == 'object' or dataframe2.dtypes[i] == 'category':
                variaveis_categoricas.append(i)
            variaveis_categoricas
        for var in variaveis_categoricas:
            dataframe2[var] = lb.fit_transform(dataframe2[var])
            cat.append(lb.classes_)

def datas(dataframe3):

        for i in dataframe3.columns[0:len(dataframe3)].tolist():
            if dataframe3.dtypes[i] == 'datetime64':
                variaveis_datas.append(i)
            variaveis_datas
        for var in variaveis_datas:
            dataframe3[var] = lb.fit_transform(dataframe3[var])
            dataframe3.astype(float)
# Cria o encoder e aplica OneHotEncoder
```

Fonte: O autor

O *LabelEncoder*, Figura 24, do pacote do *sklearn.preprocessing* e foi utilizado para padronizar os dados transformando variáveis categóricas em numéricas.

Por exemplo, a função categórica(dataframe2) foi utilizada para codificar a coluna ‘redirecionar’ obtendo-se o resultado {'aceitável': 0, 'atenção': 1, 'não': 2, 'sim': 3}.

3.4.5. Separação dos dados em treino e teste

Utilizando a função *train_test_split* separou-se os dados em dois grupos: *train* (treino) e *test* (teste). Nesta fase também é necessário fazer a separação das variáveis preditoras da variável ‘alvo’ (*target*).

Figura 25 - Criação dos conjuntos de treino e teste

```
#Separando os dados em treino e teste
from sklearn.model_selection import train_test_split, GridSearchCV
df = pd.read_csv('csv\df_main_ok.csv')# dataset.csv
df= df.drop(['humidity_weather'], axis=1)
# Converte a coluna 'dateCollection' em timestamp()
# Convertendo a coluna 'dateCollection' para o formato datetime
df['dateCollection'] = pd.to_datetime(df['dateCollection'])
# Calculando o timestamp em segundos
df['dateCollection'] = (df['dateCollection'] - pd.Timestamp('1970-01-01')) // pd.Timedelta('1s')
numericas(df)
categoricas(df)
# Separando variáveis independentes (features) e variável target
X = df.drop('redirecionar', axis=1)
y = df['redirecionar']
# Separando os dados em conjuntos de treino e teste
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Fonte: O autor

A coluna *humidity_weather* foi retirada do *dataset* para que, no treinamento do modelo, seja considerado apenas a temperatura média, condições do tempo, meta alcançada e a data da coleta.

Nas últimas linha do código, na Figura 25, as variáveis preditoras foram atribuídas a X e a variável alvo atribuída a y. Os dados foram divididos em *X_train, X_test, y_train, y_test.*

3.4.6. Treinamento e teste dos modelos

Antes de iniciar o treinamento e teste utilizou-se a função *GridSearchCV* para encontrar os melhores parâmetros de configuração de modelos com base nos dados de *X_train* e *y_train*. Para isso, a variável *grid_search*, na função (1), armazena os dados de configuração para ser usado em *grid_search.fit.*

Grid_search = GridSearchCV (mlp, *param_grid*, cv=5, scoring='accuracy') (1).

Na Figura 26 é apresentado parte do código para encontrar o conjunto de melhores parâmetros para o modelo *MLPClassifier.*

Esse procedimento também foi executado para os modelos *RandomForestClassifier, GradientBoostingClassifier, SVC (Support Vector Classification).*

.

Figura 26 - Encontrando parâmetros por meio do *GridSearchCV*

```python
#Descobrindo os melhores hiperparâmetros para o MLPClassifier
mlp = MLPClassifier(random_state=42)

# Definição dos hiperparâmetros a serem testados
param_grid = {
    'hidden_layer_sizes': [(100,), (200,), (300,)], # varia o número de neurônios
    'activation': ['logistic', 'tanh', 'relu'],  # Varia as funções de ativação
    'solver': ['adam', 'sgd'],  # Varia os otimizadores
}

# Inicialização da pesquisa de grade
 # Ajuste 'scoring' conforme a métrica desejada
grid_search = GridSearchCV(mlp, param_grid, cv=5, scoring='accuracy')

# Realização da busca de grade
grid_search.fit(X_train, y_train)

# Melhores parâmetros encontrados
print("Melhores parâmetros:", grid_search.best_params_)

# Melhor pontuação encontrada durante a busca de grade
print("Melhor pontuação:", grid_search.best_score_)
```

43.2s Python

Fonte: O autor

Assim, encontrou-se os melhores parâmetros para os modelos conforme apresentado na Figura 27.

Figura 27 - Melhores parâmetros para o treinamento dos modelos.

```
RandomForestClassifier(
    n_estimators=300,
    max_depth= None,
    min_samples_split=5,
    random_state=42)

GradientBoostingClassifier(
    random_state=42,
    learning_rate= 0.11,
    max_depth= 3,
    n_estimators= 250)
```

```
SVC(
    random_state=42,
    C=1, gamma=0.01)

MLPClassifier(
    hidden_layer_sizes=(200),
    activation='logistic',
    solver='adam',
    alpha=0.0002,
    max_iter=400,
    random_state=42)
```

Fonte: O autor

Com os parâmetros encontrados, cada modelo foi configurado e treinado, respectivamente, utilizando-se as funções *fit* e *score* para treinar o modelo e gerar a acurácia, empregando os conjuntos de testes (*X_test, y_test)*. Na Figura 28, consta o código que gerou a acurácia durante o treinamento e teste dos modelos.

Figura 28 - Treinamento, teste e acurácia

```python
# Inicialização configuração de parâmetros e treinamento do modelo
random_forest = RandomForestClassifier(n_estimators=300, max_depth= None,
                                       min_samples_split=5,random_state=42)
gradient_boosting = GradientBoostingClassifier(random_state=42,learning_rate= 0.11,
                                               max_depth= 3, n_estimators= 250)
svm = SVC(random_state=42, C=1, gamma=0.01)
mlp_classifier = MLPClassifier(hidden_layer_sizes=(200), activation='logistic',
                               solver='adam', alpha=0.0002, max_iter=400, random_state=42)

# Treinamento dos modelos
random_forest.fit(X_train, y_train)
gradient_boosting.fit(X_train, y_train)
svm.fit(X_train, y_train)
mlp_classifier.fit(X_train, y_train)

# Avaliação dos modelos
rf_accuracy = random_forest.score(X_test, y_test)
gb_accuracy = gradient_boosting.score(X_test, y_test)
svm_accuracy = svm.score(X_test, y_test)
mlp_accuracy = mlp_classifier.score(X_test, y_test)

# Impressão das métricas
print("Random Forest Accuracy:", rf_accuracy)
print("Gradient Boosting Accuracy:", gb_accuracy)
print("SVM Accuracy:", svm_accuracy)
print("MPC Accuracy:", mlp_accuracy)
```

Python

```
Random Forest Accuracy: 0.9263157894736842
Gradient Boosting Accuracy: 0.9631578947368421
SVM Accuracy: 0.6789473684210526
MPC Accuracy: 0.7473684210526316
```

Fonte: O autor

Os modelos após treinado foram salvos por meio do módulo *pickle*, tal módulo *Python* implementa protocolos binários de serializar e desserializar uma estrutura de objeto *Python*. Veja Figura 29 a seguir.

Figura 29 – Código utilizado para salvar modelos treinados.

```python
#salva os modelos
import pickle as pickle
caminho = r'C:\Users\J\Documents\\'

# Salvando os modelos com pickle5
with open(f'{caminho}random_forest_model.pkl', 'wb') as file:
    pickle.dump(random_forest, file)

with open(f'{caminho}gradient_boosting_model.pkl', 'wb') as file:
    pickle.dump(gradient_boosting, file)

with open(f'{caminho}svm_model.pkcls', 'wb') as file:
    pickle.dump(svm, file)

with open(f'{caminho}mlp_classifier_model.pkl', 'wb') as file:
    pickle.dump(mlp_classifier, file)
```

Fonte: O autor

Capítulo 4 – Escolha do modelo e Implementação no *Power BI*

O modelo *Gradient Boosting* apresentou acurácia de aproximadamente 96%, conforme Tabela 11. Assim, este foi o modelo escolhido para ser utilizado durante a implementação.

Tabela 11 - Resultado da acurácia durante o treinamento

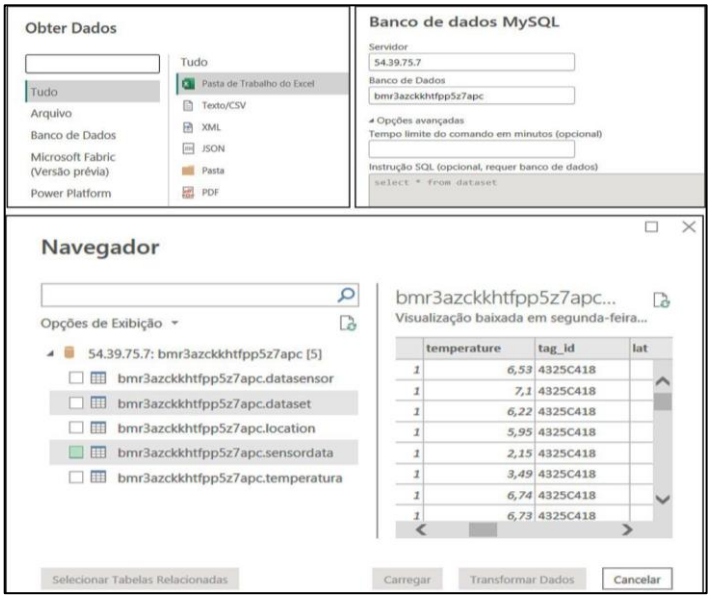
Model	Accuracy
Gradient Boosting	96,30%
Random Forest	92,60%
MPC	74,70%
SVM	67,90%

Fonte: o autor.

Uso da versão grátis, 2.122.1066.0 64-bit (outubro de 2023), do *Power BI* para implementação do *dashboard* foi aceitável pois os recursos disponíveis eram suficientes para a exibição e sincronização necessária dos dados; nele foi possível fazer conexões a bancos remotos.

Dessa forma, criou-se uma conexão ao *Clever Cloud* por meio do menu “Obter dados”, pesquisando por ‘*mysql’*, escolheu-se a opção ‘banco de dados *MySQL’*.

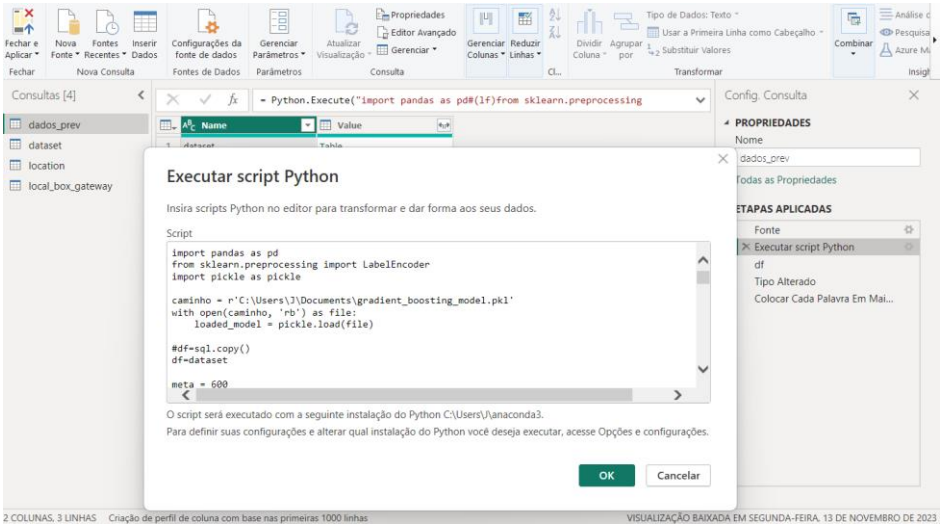
Figura 30 - Acessando banco remoto no *Power BI*



Fonte: O autor

Após configurar o acesso ao banco, Figura 30, acessou-se a tabela '*dataset'*. Utilizou-se da ferramenta 'Transformar dados', que se encontra no menu inicial, e do editor '*Script Python'* para carregar o modelo treinado e o código de pré-processamento dos dados do banco, conforme Figura 31. O editor acessa a tabela por meio da variável '*dataset'*. O código completo é apresentado nas figuras seguintes. Na Figura 32, faz-se o importe das bibliotecas necessárias ao pré-processamento dos dados.

Figura 31 - Ferramentas 'Transformar dados', '*Script Python', 'Power BI'.*



Fonte: O autor

O modelo salvo foi recuperado de uma pasta local por meio do módulo *pickle*, além de definir a meta diária de 600, o frasco contendo 5 doses e a criação das colunas necessárias, assim como foi realizado na fase de criação do modelo.

Figura 32 – Parte 1: código executado no Editor Script Python - *Power BI*

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
import pickle as pickle

caminho = r'C:\Users\J\Documents\gradient_boosting_model.pkl'
with open(caminho, 'rb') as file:
    loaded_model = pickle.load(file)

df=sql.copy()
#df=dataset
meta = 600
# Converte 'dateCollection' para datetime, se necessário
df['dateCollection'] = pd.to_datetime(df['dateCollection'])
# Define 'dateCollection' como o índice do DataFrame
df.set_index('dateCollection', inplace=True)
# Cria uma coluna 'dose' com valor 5
df['dose'] = 5
# Use groupby por dia para calcular o valor acumulado
# e o percentual da meta atingida para cada dia
df['acumulado'] = df.groupby(df.index.date)['dose'].cumsum()
df['meta_alcanc'] = round((df['acumulado'] / meta) * 100, 2)
df['redirecionar'] = ''
df.reset_index(inplace=True)

# Criação das listas para armazenar os LabelEncoders
label_encoders_categoricos = []
label_encoders_numericos = []
variaveis_numericas = []
variaveis_categoricas = []

```

Fonte: O autor

Também, apresenta-se o restante do código, Figura 33, onde constam as funções *preprocess_and_predict*, responsáveis por fazer o pré-processamento e realizar as previsões.

Na linha 57 da Figura 34, a variável *predicted_values* recebe os valores de predições processados pelo método *loaded_model.predict* para construção da coluna 'redirecionar' e atribui-la ao *dataframe df* que será utilizado para construção do *dashboard.*

Figura 33 - Parte 2: função de pré-processamento e predição – *Power BI*

```
31 def preprocess_and_predict(df):
32     le = LabelEncoder()  # Definindo o LabelEncoder aqui
33 
34     # Pré-processamento para a dataCollection
35     df['dateCollection'] = (df['dateCollection'] - pd.Timestamp('1970-01-01')) // pd.Timedelta('1s')
36 
37     # Identificação das variáveis categóricas e numéricas
38     variaveis_categoricas = df.select_dtypes(include=['object', 'category']).columns.tolist()
39     variaveis_numericas = df.select_dtypes(include=['int64', 'float64']).columns.tolist()
40 
41     # Inicialização do LabelEncoder para variáveis categóricas
42     for var in variaveis_categoricas:
43         le = LabelEncoder()
44         df[var] = le.fit_transform(df[var])
45         label_encoders_categoricos.append(le)
46 
47     # Inicialização do LabelEncoder para variáveis numéricas
48     for var in variaveis_numericas:
49         le = LabelEncoder()
50         df[var] = le.fit_transform(df[var])
51         label_encoders_numericos.append(le)
52 
53     # Seleção das colunas para previsão
54     df_prev = df[['dateCollection', 'main_weather', 'temp_weather', 'meta_alcanc']]
55 
56     # Realização da previsão
57     predicted_values = loaded_model.predict(df_prev)
58 
59     # Reverte os valores preditos para a coluna "redirecionar"
60     mapeamento_inverso = {0: 'aceitável', 1: 'atenção', 2: 'não', 3: 'sim'}
61     predicted_classes = [mapeamento_inverso[pred] for pred in predicted_values]
62     df['redirecionar'] = predicted_classes
63     #print(predicted_classes)
64     return predicted_classes
```

Fonte: O autor

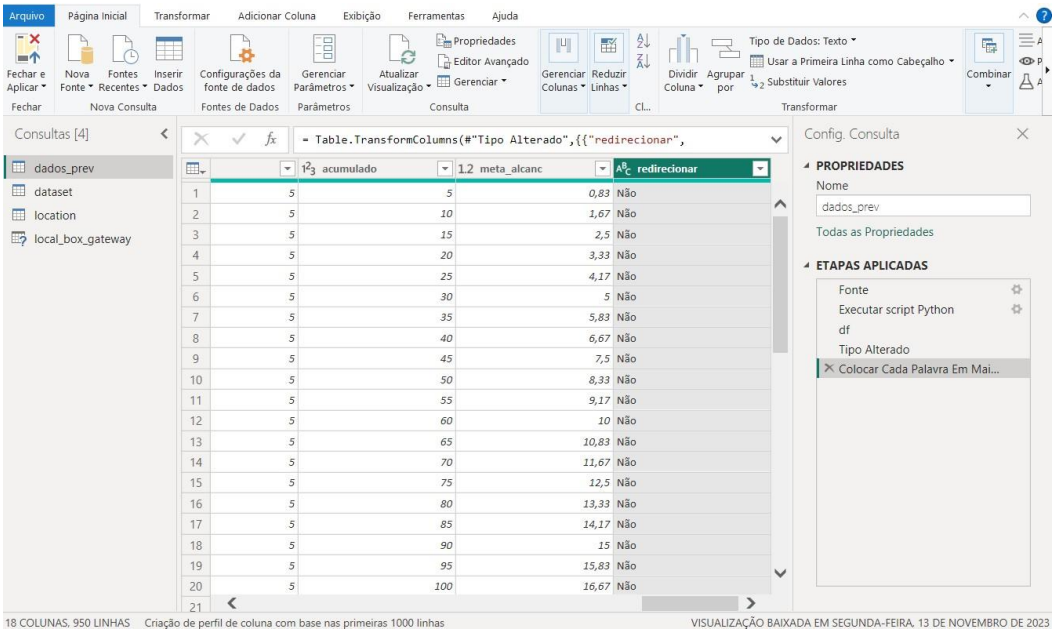
Figura 34 - Realizando o processo de predição - *Power BI*

```
65 
66 # realizar a predição e a inversão do LabelEncoder após a predição
67 if not df.empty:
68     df_prev = df[['dateCollection', 'main_weather', 'temp_weather', 'meta_alcanc']]
69 
70     # Supondo que 'redirecionar' é a coluna que será prevista
71     predicted_values = preprocess_and_predict(df_prev)
72 
73     # Reverter o 'LabelEncoder' após a predição
74 
75     # Código para reverter a transformação para variáveis categóricas
76     for var, encoder in zip(variaveis_categoricas, label_encoders_categoricos):
77         predicted_values[var] = encoder.inverse_transform(predicted_values[var])
78 
79     # Código para reverter a transformação para variáveis numéricas
80     for var, encoder in zip(variaveis_numericas, label_encoders_numericos):
81         predicted_values[var] = encoder.inverse_transform(predicted_values[var])
82 
83     # Adicionar as previsões ao DataFrame original
84     df['redirecionar'] = predicted_values
```

Fonte: O autor

Na Figura 35 é visto o *dataset* com dados recuperados. É possível observar a coluna ‘redirecionar’ com as previsões incluídas e demais colunas com dados referentes a meta, dose, valores acumulados e etc.

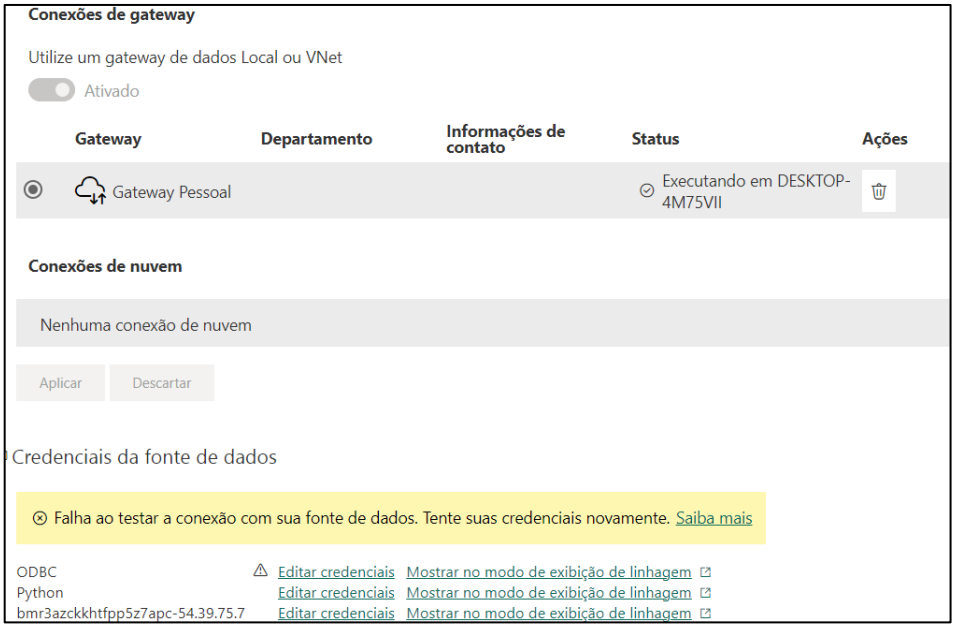
Figura 35 - Dados recuperados do dataset para criação do dashboard



Fonte: O autor

Na área de administração *online* do *Power BI* é possível agendar horários de atualização, isto é, sincronizar com servidor local (*on-premises gateway)*. Após fazer a publicação com *Power BI Desktop,* é possível acessar as configurações, nos ‘Modelos semânticos’, Figura 36, e fazer os ajustes necessários para acessar.

Figura 36 - Configuração Área Administração *Power BI*

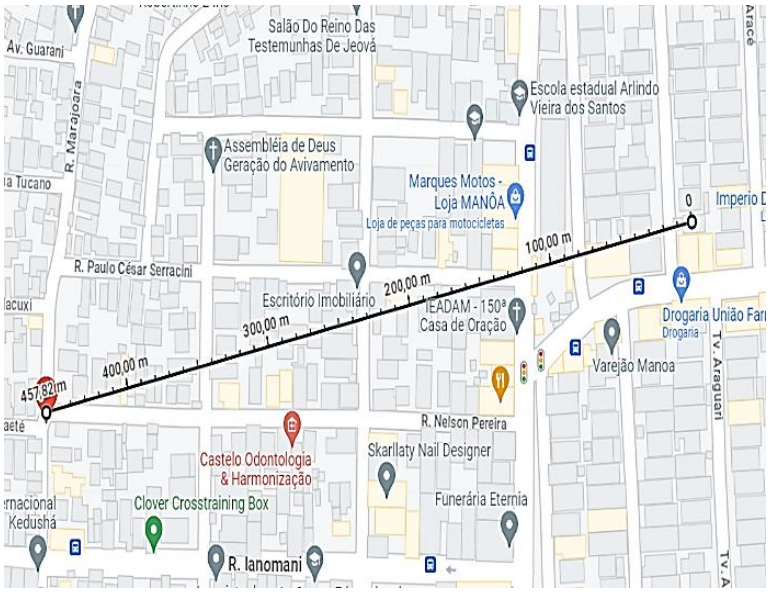


Fonte: o autor.

Capítulo 5 – Resultados Obtidos

Conforme visto na Figura 37, a distância máxima alcançada foi de aproximadamente 458 metros. Essa medida foi obtida por meio da função de "Medir distância" no *Google Maps*, utilizando as coordenadas capturadas pelo *ESP32* e anteriormente armazenadas no banco de dados.

Figura 37 - Máxima medida alcançada pelo *LoRa* simples



Fonte: O Autor

A partir da implementação do protótipo, das fases de exploração, pré-processamento, análise e previsão de dados, com base nos dados salvo inicialmente no banco, construiu-se um método próprio para proporção da meta esperada por hora, conforme mostrado na Figura 38.

Figura 38 - Meta esperada por hora (Método próprio).

	11H	12H	15H	16H	17H
Meta Esperada	40%	50%	80%	90%	100%

Fonte: O Autor.

A partir desses critérios, o algoritmo foi treinado para entender o padrão e generalizar o modelo para outros casos. Caso a meta alcançada se distancie da meta esperada, o algoritmo sugere uma recomendação.

Em resumo, os frascos vazios foram computados e multiplicados por suas respectivas doses, resultando na coluna “acumulado”.

A coluna “meta_alcanc” é o resultado da divisão do valor acumulado pela “meta diária”, ou seja, para cada linha calcula-se quanto da meta diária foi alcançada.

A meta diária utilizada para previsão no *Power BI* foi de 600 e considerou-se que cada frasco contia 5 doses. O conjunto de dados teve um total de 950 linhas.

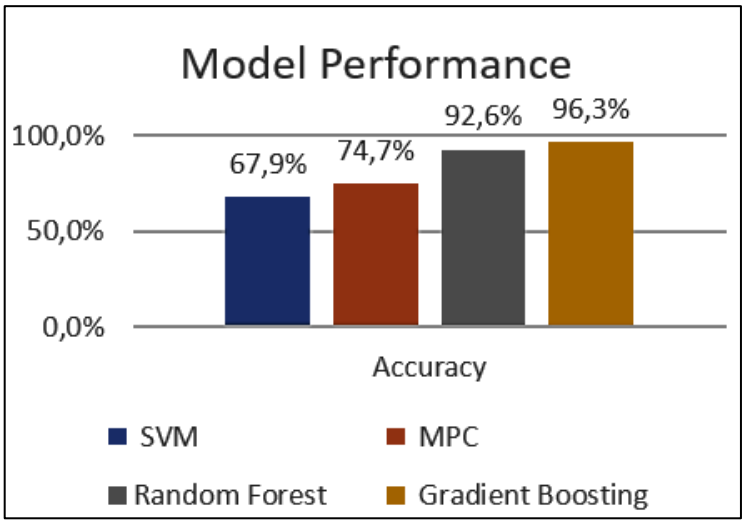
Utilizou-se a acurácia para medição e comparação do desempenho.

Quando a acurácia se aproxima de 1, isso indica que o modelo está classificando corretamente a maioria das instâncias no conjunto de dados. Se a acurácia for próxima de 0, significa que o modelo está tendo dificuldades em realizar classificações corretas.

O modelo *Gradient Boosting* (*Gb*) obteve o melhor desempenho.

Na Figura 39 é possível observar o desempenho dos modelos pela comparação da acurácia durante os testes de classificação.

Figura 39 - Gráfico da acurácia (*Accuracy*)



Fonte: O Autor

À acurácia apresentada foi obtida em relação a *X_test* e *y_test*, onde, respectivamente correspondem aos 20% das variáveis preditoras e 20% da variável resposta separada pela bibl*iot*eca *train_test_split* do pacote *Scikit-Learn* durante o processo de treinamento dos modelos.

Outras métricas foram observadas, entre estas o *f1-score* (0,97), Figura 40. O *f1-score* é uma medida de precisão e revocação, e um valor mais próximo de 1 indica um equilíbrio eficiente entre as duas.

Na Figura 41 é possível notar uma clara diferença entre os modelos, com gráficos correspondentes a acertos e erros de cada modelo.

Figura 40 - Relatório de classificação gerado pela função *classification_report* do Scikit-Learn

Gb Report:

	precision	recall	f1-score	support
0	1.00	0.58	0.74	12
1	0.96	1.00	0.98	22
2	0.95	1.00	0.98	124
3	1.00	0.94	0.97	32
accuracy			0.96	190
macro avg	0.98	0.88	0.91	190
weighted avg	0.96	0.96	0.96	190

Mlp Report:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	12
1	0.59	0.45	0.51	22
2	0.79	0.93	0.85	124
3	0.65	0.53	0.59	32
accuracy			0.75	190
macro avg	0.51	0.48	0.49	190
weighted avg	0.69	0.75	0.71	190

Rf Report:

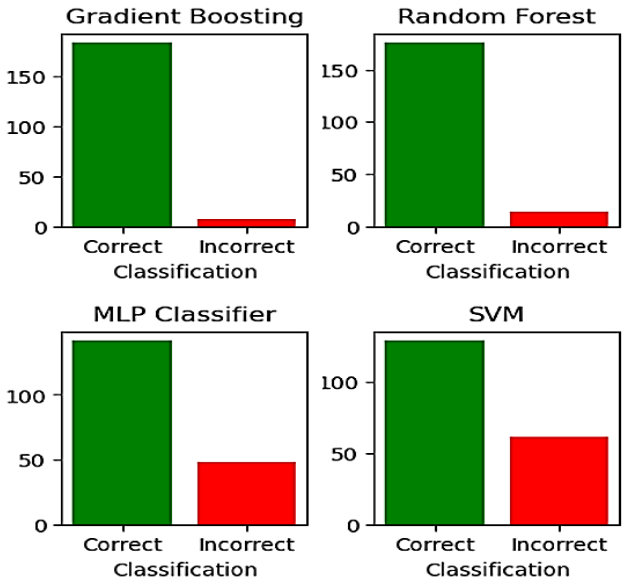
	precision	recall	f1-score	support
0	1.00	0.25	0.40	12
1	0.84	0.95	0.89	22
2	0.95	0.99	0.97	124
3	0.88	0.91	0.89	32
accuracy			0.93	190
macro avg	0.92	0.78	0.79	190
weighted avg	0.93	0.93	0.91	190

Svm Report:

	precision	recall	f1-score	support
0	0.00	0.00	0.00	12
1	1.00	0.09	0.17	22
2	0.67	1.00	0.80	124
3	1.00	0.09	0.17	32
accuracy			0.68	190
macro avg	0.67	0.30	0.29	190
weighted avg	0.72	0.68	0.57	190

Fonte: o autor:

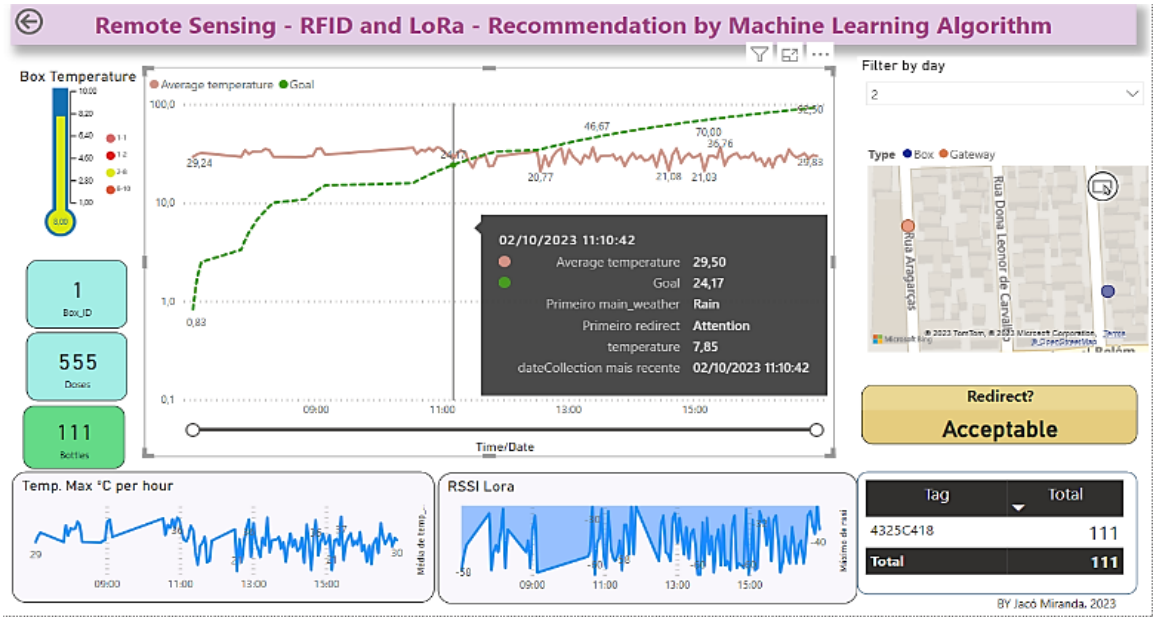
Figura 41 - Comparativo dos erros e acertos de cada modelo.



Fonte: O Autor.

As variáveis mais importantes do monitoramento foram reunidas em um dashboard no *Power BI* por meio de *link* e *Mobile,* Figura 42, e *Mobile*, Figura 43, incluindo temperatura da caixa, número de frascos/doses, meta alcançada, informações do clima e recomendações de redirecionamento de vacinas.

Figura 42 - *Dashboard* de visualizações da meta alcançada.



Fonte: Link: https://app.powerbi.com/view?r=eyJrIjoiM2QwOTA3MjMtYWMzNS00ODYxLTljNzktMTViYjBjOTk0YmYyIiwidCI6ImNkYTA4NGM2LWU4ZDMtNDQyMC05Y2I4LTE5MmZiYzgyODExNyJ9

No *dashboard* contém filtros, por exemplo filtro por dia (*Filter by day*) que permite, a partir da escolher o dia/data, que toda a interface dinamicamente seja atualizada com os dados referente ao dia escolhido.

Figura 43 - Acesso pelo *Power BI Mobile*

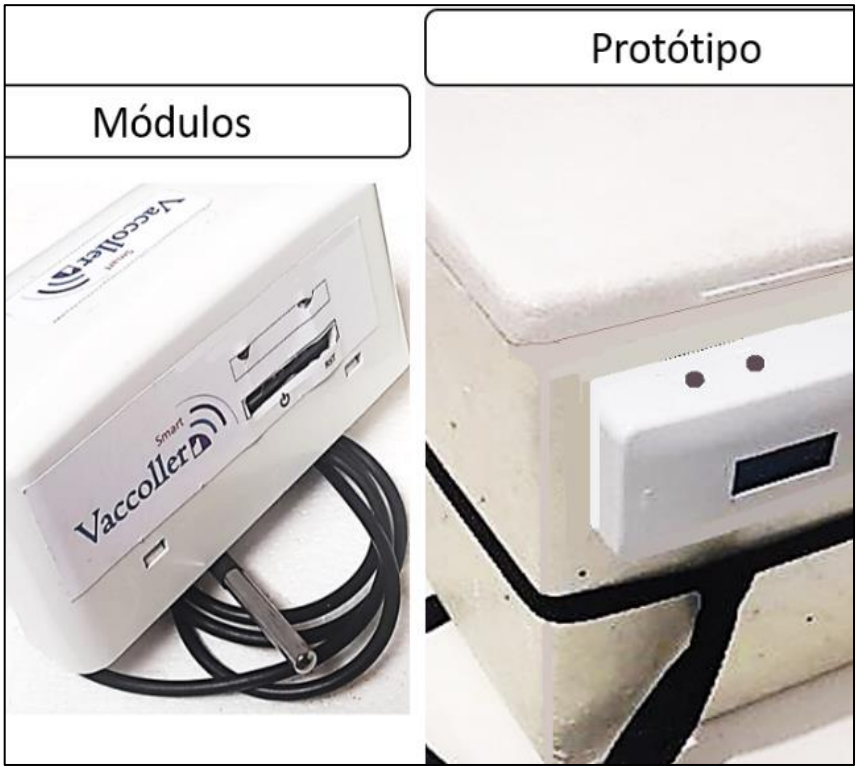
Fonte: o autor. Moldura adaptada. Esta Foto está licenciada em CC BY-NC.

Também, é apresentado um gráfico com as informações da temperatura e meta alcançada. Ao passar o mouse sobre a meta, linha verde (*Goal*) do gráfico central do dashboard. Outras informações são apresentadas como a data e hora do dia, meta alcançada, direcionamento recomendado (*Redirect*?), intensidade do sinal *LoRa (RSSI)* entre o dispositivo coletor e o *gateway*, quantidade e nº identificador da etiqueta (*tag*), mapa com a geolocalização da caixa térmica e do *gateway* (*Type Box* e *Gateway*, respectivamente).

O protótipo final, Figura 44, foi acondicionado dentro de duas caixas plásticas (usando quadros para disjuntores PVC do tipo 2, Disjuntor *Din Steck*) e modificado para a adaptação dos componentes.

Cada caixa plástica foi equipada com um módulo, que foi acoplado na lateral das caixas térmicas para conferir um *design* semelhante às caixas utilizadas nas campanhas de vacinação. Por fim, o protótipo foi configurado com um módulo coletor, que contém um sensor de temperatura, um leitor *RFID* na parte superior e *LEDs* na parte da frente para sinalizar a transmissão de dados. O segundo módulo (*gateway*) ficou responsável pela recepção dos dados e transmissão ao banco como já descrito anteriormente.

Figura 44 - Protótipo montado em caixa de isopor.



Fonte: O autor

Capítulo 6 – CONCLUSÃO

Este estudo apresentou uma proposta que explora a aplicação da tecnologia computacional, em particular o uso de *Machine Learning* e tecnologia *IoT* como *RFID* e *LoRa*, no contexto de campanhas de vacinação. A abordagem incluiu o monitoramento da temperatura, a contagem de frascos, a consideração das condições climáticas e recomendações baseadas em um modelo treinado, todos integrados em uma interface de suporte à tomada de decisão.

Conforme evidenciado nos trabalhos relacionados, a aplicação de modelos de *Inteligência Artificial* ao longo da Rede de Frio é sustentada, porém é notável a escassez de sua utilização em cenários das campanhas de vacinação, consideradas a última instância da rede. Identificaram-se oportunidades onde a coleta de dados poderia ser otimizada mediante a redução da interferência humana no processo, viabilizando a aplicação de análise de dados para a geração de informações pertinentes.

Comparativamente aos trabalhos relacionados, que utilizam *ML* acima da instância local e para o monitoramento do dado, frequentemente se apoiam em tecnologias como *GPS*, *SMS*, redes de sensores sem fio e *RFID*, esta pesquisa se diferencia em três aspectos principais. Primeiramente, pela sua aplicação na última instância da Rede de Frio, direcionada às campanhas de vacinação. Em segundo lugar, pela adoção da tecnologia *LoRa* para a comunicação entre dispositivos, o que permite a conectividade à internet através de um único dispositivo, resultando em redução de custos com roteadores e consumo de energia. Por fim, pela aplicação de aprendizado de máquina na geração de recomendações sobre a meta alcançada, possibilitando decisões oportunas e a redução de desperdícios.

6.1. Observações acerca das primeiras ideias

A primeira ideia para solução da problemática era utilizar a tecnologia *RFID*, com antenas de detecção à distância acima de 5 metros e etiquetas ativas que já pudessem medir a temperatura do frasco. Assim, dentro de uma sala de vacinação, os frascos/etiquetas poderiam ser detectados automaticamente, possibilitando a contagem dos frascos e, consequentemente, a quantidade de doses. Para isso, seria necessário ter um banco de dados com as informações das etiquetas, como a data de validade, a dose por frasco, o tipo de vacina e a temperatura padrão de armazenamento. Algumas considerações são descritas a seguir:

- Considerando o alto valor das antenas e middlewares, e que, mesmo implementando o sistema de detecção, o frasco seria detectado cheio ou vazio e para detectá-lo somente vazio teria que sair da zona de alcance da antena, essa solução não seria viável para o contexto da vacinação.
- A utilização do protocolo *LoRaWAN* também foi pensado, porém os poucos dispositivos proprietários também possuíam valores altos e com custo para uso nas plataformas de fabricantes como a Semtech Corporation que atualmente criou o *LoRa Cloud* e *The Things Network (TTN).*

6.2. Dificuldades encontradas durante o uso das tecnologias

- O *ESP32 TTGO T-BEAM* V1.1 reúne as tecnologias principais para uso do *LoRa*, incluindo o protocolo *LoRaWAN*, além de módulo *WI-FI*, *GPS* e uso de bateria. No entanto, várias tentativas para conectar à rede *TTN* por meio da biblioteca disponibilizada em *https://github.com/kizniche/ttgo-tbeam-ttn-tracker* não foram bem-sucedidas. Após várias pesquisas em sites de discussão sobre o assunto, foram encontrados vários relatos de dificuldade de acesso à rede pelo dispositivo desde meados de 2021. Apesar de o site da *Semtech* informar cobertura alta da rede *LoRa* em Manaus, é provável que o dispositivo esteja fora do alcance dos *gateways*. Não foi encontrado mapa da localização exata dos *gateways* para confirmar a suspeita.
- A ativação do módulo *GPS* também apresentou problemas relativos às bibliotecas comumente utilizadas, como por exemplo, a *TinyGPS++*. Isso ocorreu devido à versão utilizada apresentar pinagem diferente da utilizada pela biblioteca. Para resolver o problema, foi necessário fazer reset do chip por meio das bibliotecas encontradas em *https://github.com/eriktheV-king/TTGO_T-beam_GPS-reset/tree/master.*

A proposta utilizando *Power BI* para publicação dos dados coletados por contagem de frascos e recomendação com base na meta alcançada e condições climática configurou a melhor opção considerando que o *ESP32* utilizado faz comunicação por *LoRa* Simples e por *WI-FI*.

No contexto do aprendizado no mestrado em informática, a jornada desde a coleta de dados até a disponibilização online dessas informações representa um valioso exercício prático. A criação de um sistema, envolvendo todos os aspectos do processo, contribui para o desenvolvimento técnico e acadêmico do aluno, proporcionando uma compreensão profunda das complexidades envolvidas na implementação de soluções *IoT* e de aprendizado de máquina em cenários do mundo real.

Embora os resultados preliminares indiquem a viabilidade do monitoramento automático da temperatura e contagem de frascos, ressalta-se a necessidade de mais

dados reais de campanhas de vacinação para aprimorar a precisão das previsões. O progresso na aplicação dessas tecnologias nas instâncias finais da Rede de Frio poderá beneficiar tanto os profissionais de saúde quanto o público em geral, destacando o papel fundamental do avanço tecnológico na melhoria da eficácia e da tomada de decisões assertivas em serviços essenciais.

6.3. Observações em relação aos objetivos específicos

Para solução de monitoramento inteligente na instância local da Rede de Frio, foram alcançados os seguintes objetivos específicos:

- Revisão detalhada das tecnologias de Sensoriamento Remoto e técnicas de *Machine Learning* relevantes para a monitorização da Rede de Frio. Essa revisão permitiu identificar as tecnologias e técnicas mais adequadas para o cenário da Rede de Frio, considerando fatores como a necessidade de coleta de dados contínua e em tempo real, a complexidade do processamento dos dados e a necessidade de gerar *insights* e recomendações úteis para a tomada de decisões.
- Identificação de artefatos de sensoriamento e modelos de *Machine Learning* aplicáveis ao cenário da Rede de Frio. Essa identificação foi realizada por meio de uma análise comparativa dos diferentes frameworks utilizado em *Machine Learning* e dispositivos IoT.
- Desenvolvimento e implementação de um sistema de Sensoriamento Remoto que colete dados relacionados à temperatura e outras variáveis importantes no monitoramento. O sistema desenvolvido utiliza sensores de temperatura, de detecção de objetos, além de outros sensores que podem ser adicionados conforme a necessidade. Os dados coletados são transmitidos para um servidor central, onde são processados por um modelo de *Machine Learning* da biblioteca *Scikit-Learn*.
- Proposição de estratégias e recomendações baseadas nas análises realizadas pelo sistema de *Machine Learning* para otimizar a distribuição e o uso das vacinas. As estratégias e recomendações propostas são baseadas nos *insights* gerados pelo modelo de *Machine Learning*. Essas estratégias podem ajudar a melhorar a eficiência da distribuição e do uso das vacinas, reduzindo o risco de perda ou ineficácia das vacinas.
- Validação do sistema de monitoramento em situações simuladas, por meio de simulação manual de inserção de dados. Os resultados apresentados em *Power BI* demonstram que é possível a implementação da solução proposta. A escalabilidade depende da implementação de mais recursos ao projeto.

Capítulo 7 – CONSIDERAÇÕES FINAIS E TRABALHO FUTUROS

O progresso na aplicação dessas tecnologias nas instâncias finais da Rede de Frio poderá beneficiar tanto os profissionais de saúde quanto o público em geral, destacando o papel fundamental do avanço tecnológico na melhoria da eficácia e da tomada de decisões assertivas em serviços essenciais.

A saúde pública tem diversas áreas para aplicação da Inteligência Artificial (*IA*). Além do contexto utilizado neste trabalho, o uso em outros contextos pode proporcionar otimização das tarefas, como por exemplo:

- Diagnóstico de exames por imagens ou texto: a *IA* pode ser usada para automatizar o diagnóstico de doenças, aumentando a precisão e a eficiência.
- Previsão de diagnóstico com base em dados históricos de pacientes: a *IA* pode ser usada para prever o diagnóstico de pacientes com base em seus dados históricos, ajudando os médicos a tomar decisões mais rápidas e precisas.
- Previsão de surtos de alerta epidemiológico: a *IA* pode ser usada para prever surtos de doenças, ajudando as autoridades de saúde a tomar medidas preventivas.
- Previsão de alta transmissão de arboviroses transmitidas pelo mosquito Aedes aegypti (Dengue, Chikungunya e Zika) no estado do Amazonas: a *IA* pode ser usada para prever períodos de alta transmissão de arboviroses, ajudando as autoridades de saúde a tomar medidas de prevenção e controle.

7.1. Algumas sugestões de trabalhos futuros:

- Utilização das bibliotecas *Plotly, Boken, Dash*, e etc., para implementações gráficas de *dashboard* em vez do *Power BI*, por ter licença proprietária.
- Incluir um modelo de previsão de demanda: para isso, é necessário coletar mais dados reais de campanhas de vacinação, incluindo informações sobre a população-alvo, o número de doses aplicadas, as condições climáticas e de trânsito.

- Expandir o alcance da solução: A solução apresentada pode ser expandida para incluir outros dispositivos, como geladeiras e freezers, para monitorar a temperatura de toda a Rede de Frio.
- Incorporar outras funcionalidades: A solução pode ser ainda mais aprimorada com a inclusão de outras funcionalidades, como o monitoramento da umidade e a detecção de violação.
- Verificar a viabilidade de sistema de informação usando plataforma *IoT* e protocolo *LoRaWAN*;
- Integração com base de dados do SI-PNI, Sistema de informação do programa nacional de imunização;
- Incorporação de sistema de alertas que dispara mensagens instantâneas para pessoas cadastradas com pendencia vacinal, as quais se encontram na região de campanha, sobretudo quando houver recomendações de redirecionamento de vacinas por causa da baixa procura.

Portanto, este trabalho lança as bases para futuras pesquisas e experimentações, visando aperfeiçoar e ampliar a aplicação na área da saúde pública, especialmente considerando que as instituições de saúde públicas possuem seus próprios *datacenters* com dados clínicos, bancos de imagens e recurso diversos que podem ser direcionados para o desenvolvimento tecnológico em saúde.

REFERÊNCIAS BIBLIOGRÁFICAS

ABADI, Martín *et al.* TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Disponivel em: https://www.tensorflow.org/.

ABDULLAH, Talal AA ; ALI, Waleed ; ABDULGHAFOR, Rawad. Empirical study on intelligent Android malware detection based on supervised machine learning. **International Journal of Advanced Computer Science and Applications**, v. 11, n. 4, 2020.

AHMAD, Ghulab Nabi *et al.* Efficient Medical Diagnosis of Human Heart Diseases Using Machine Learning Techniques with and Without GridSearchCV, 10, 2022.

AKHBARIFAR, Samira *et al.* A secure remote health monitoring model for early disease diagnosis in cloud-based IoT environment. **Personal and Ubiquitous Computing**, v. 27, n. 3, p. 697-713, jun. 2023. ISSN 16174917.

ALEGRIA, Rosa *et al.* **Teoria e Prática da Pesquisa Aplicada**. [S.l.]: Elsevier Brasil, 2011. 9 p. Disponivel em: https://books.google.com.br/books?id=Oot9S0RLWjwC.

ANDERSON, Richard *et al.* Supporting immunization programs with improved vaccine cold chain information. **IEEE Global Humanitarian Technology Conference (GHTC 2014)**, Seattle, Washington, USA, 2014. 8. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/6970284/. Acesso em: 24 Setembro 2021.

ARDUINOIDE2. **Programas**, 2023. Disponivel em: https://docs.arduino.cc/software/ide-v2.

AWS. **O que é IDE (Ambiente de desenvolvimento integrado)?**, 2023. Disponivel em: https://aws.amazon.com/pt/what-is/ide/.

BALTZAN, Paige. **Tecnologias Orientadas para Gestão**. 6ª. ed. Porto Alegre: AMGH, 2016.

BARBIERI, G.G.R. **Sistema de informação**. Porto Alegre: SAGAH, 2017. Disponivel em: https://integrada.minhabiblioteca.com.br/#/books/9788595022270. Acesso em: 10 Janeiro 2021.

BARROSO RODRIGUES, Samuel *et al.* Uso do Sistema de Informação de Imunização do Brasil: qual a realidade? TT - Using the Brazilian Immunization Information System: What is real? **rev.cuid. (Bucaramanga.2010)**, v. 13, n. 1, p. 1-13, 2022. ISSN 2216-0973. Disponivel em: https://revistas.udes.edu.co/cuidarte/article/view/2138/2411.

BHAMIDIPATI, Sai Jaya Sasanka ; , et al.. **Classification of Positive and Negative Stimuli Using EEG Data Functional Connectivity and Machine Learning**. [S.l.]: University Of Houston-Clear Lake, 2018.

BHATIA, Sugandh ; MALHOTRA, Jyoteesh. Naïve Bayes Classifier for Predicting the Novel Coronavirus. **2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)**, Amritsar - India, 2021. 4. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/9388410/. Acesso em: 06 Outubro 2021.

CHAURASIA, Pawan Kumar. Paradigms of Machine Learning and Data Analytics. [S.l.]: [s.n.], 2020. p. 156-174.

CPTEC/INPE. Previsão de Tempo em XML - CPTEC/INPE:Dados da Previsão de Tempo, IUV e Ondas do CPTEC/INPE no formato XML puro, 10 dez. 2023. Disponivel em: http://servicos.cptec.inpe.br/XML/. Acesso em: 10 dez. 2023.

DUARTE DE BRITTO LIRA, Lariisa *et al.* Congresso Nacional de Pesquisa e Ensino em Ciências-CONAPESC. **Análise Do Algoritmo Naive Bayes Na Classificação de Amostras do Banco de Dados Hepatite**, 2019. Disponivel em: http://www.editorarealize.com.br/editora/anais/conapesc/2019/TRABALHO_EV126_MD1_SA10_ID2806_12082019105434.pdf. Acesso em: 30 set. 2021.

ECLIPSE. **Eclipse documentation - Current Release**, 2023. Disponivel em: https://help.eclipse.org/2023-12/index.jsp.

ENRIKO, I Ketut Agung *et al.* COVID-19 Vaccine Distribution Tracking and Monitoring Using IoT. **2021 International Conference on Green Energy, Computing and Sustainable Technology (GECOST)**, Jakarta, Indonesia, 2021. 5. Disponivel em: 2021 International Conference on Green Energy, Computing and Sustainable Technology (GECOST). Acesso em: 22 Setembro 2021.

GÉRON, Aurélien. **Mãos à Obra:** Aprendizado de Máquina com Scikit-Learn & TensorFlow. Brasil: Alta Books, 2019. Brasil: Alta Books, 2019.

GILL, Kanwarpartap Singh ; GUPTA, Rupesh. Chronic Kidney Disease Detection Using GridSearchCV Cross Validation Method, 2023. 318-322.

GLOVER, Bill ; BHATT, Himanshu. **RFID Essentials**. 1ª. ed. Sebastopol: [s.n.], 2006.

GONÇALVES, Paulo Sérgio. **Logística e Cadeia de Suprimentos:** o essencial. 1ª. ed. Barueri: Manole, 2013. Disponivel em: https://integrada.minhabiblioteca.com.br/#/books/9788520448915/. Acesso em: 10 Janeiro 2021.

GUIA de Conetvidade de IoT: LPWA. **https:** //www.teleco.com.br/, 2021.

HASANT, Raisa Tahseen *et al.* An IoT based Real-time Data-centric Monitoring System for Vaccine Cold Chain. **2020 IEEE East-West Design & Test Symposium (EWDTS)**, Dhaka, Bangladesh, 2020. 5. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/9225047/. Acesso em: 22 Setembro 2021.

HAVILUDDIN *et al.* Student Academic Evaluation using Naïve Bayes Classifier Algorithm. **2018 2nd East Indonesia Conference on Computer and Information Technology (EIConCIT)**, Samarinda, Indonesia, 2018. 4. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/8878626/. Acesso em: 06 Outubro 2021.

HU, Hui *et al.* Vaccine supply chain management: An intelligent system utilizing blockchain, IoT and machine learning. **Journal of Business Research**, v. 156, p. 113480, 2023. Disponivel em: https://doi.org/10.1016/j.jbusres.2022.113480.

INMET. MANUAL DE USO DA APIS DO INMET. **Instituto Nacional de Meteorologia:** Ministério da Agricultura e Pecuária, 10 dez. 2023. Disponivel em: https://portal.inmet.gov.br/manual.

INTELLIJ IDEA 2023.3. **Accessibility**, 2023. Disponivel em: https://www.jetbrains.com/help/idea/accessibility.html.

ISAKOVA, Olga. Application of machine learning algorithms for classification and regression problems for mobile game monetization. Computer - University of Applied Sciences. ed. [S.l.]: Mittweida, 2021.

JÚNIOR, Ronaldo A. M. ; VIEIRA, Vinícius F. ; XAVIER, Carolina R.. Estudo de critérios de distribuição de vacinas com doses limitadas - uma abordagem baseada em redes complexas. **Anais do Brazilian Workshop on Social Network Analysis and Mining (BraSNAM)**, p. 103-114, jul. 2021. ISSN 2595-6094. Disponivel em: https://sol.sbc.org.br/index.php/brasnam/article/view/16129.

JUPYTER. **Free software, open standards, and web services for interactive computing across all programming languages**, 2023. Disponivel em: https://jupyter.org/.

KIM, Yujeong *et al.* Machine Learning Approach for Active Vaccine Safety Monitoring. **Journal of Korean Medical Science**, v. 36, n. 31, p. 1-13, jul. 2021. ISSN 1598-6357. Disponivel em: https://doi.org/10.3346/jkms.2021.36.e198.

LANA, Raquel Martins *et al.* Identificação de grupos prioritários para a vacinação contra COVID-19 no Brasil. **Cadernos de Saude Publica**, v. 37, n. 10, p. undefined-undefined, 2021. ISSN 16784464. Disponivel em: https://www.mendeley.com/catalogue/b80d39aa-4c7f-363e-b121-f12a374d1769/.

LOUREIRO, A.F *et al.* Rede de Sensores Sem Fio. **Departamento de Ciência da Computacão, Universidade Federal de Minas Gerais, Belo Horizonte**, Departamento de Ciˆencia da Computac¸ao. 48. Disponivel em: https://homepages.dcc.ufmg.br/~loureiro/cm/docs/sbrc03.pdf. Acesso em: 22 Outubro 2021.

MAGRANI, Eduardo. **Entre dados e robôs:** ética e privacidade na era da hiperconectividade. Ebook. ed. [S.l.]: [s.n.], v. 2, 2019.

MAHESH, Batta ; BATTA, Mahesh. **Machine learning algorithms-a review**. [S.l.]: International Journal of Science and Research (IJSR), 2020.

MARECHAL, J. ; DAHL, T.. Machine Learning Algorithm for Nao robot. *In:* **Advances in Communications, Electronics, Networks, Robotics and Security Volume 13**. [S.l.]: [s.n.], 2016. p. 53.

MICROSOFT. **GitHub Copilot e Visual Studio 2022**, 11 ago. 2022. Disponivel em: https://visualstudio.microsoft.com/pt-br/.

MICROSOFT. **Power BI:** Recursos do Power BI, 2023. Disponivel em: https://www.microsoft.com/pt-br/power-platform/products/power-bi.

MICROSOFT. **Documentação:** What is an on-premises data gateway?, 2023. Disponivel em: https://learn.microsoft.com/en-us/data-integration/gateway/service-gateway-onprem-indepth.

MINISTÉRIO DA SAÚDE. Manual de Rede de Frio do Programa Nacional de Imunizações. 5. ed. **Gov.BR**, 15 Agosto 2022. Disponivel em: https://www.gov.br/saude/pt-br/vacinacao/rede-de-frio/publicacoes/manual-de-rede-de-frio-do-programa-nacional-de-imunizacoes-5-ed/view. Acesso em: 23 Dezembro 2020.

MOHSIN, Afreen ; YELLAMPALLI, Siva S. IoT based cold chain logistics monitoring. **2017 IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI)**, Bangalore, India, 2017. 4. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/9225047/. Acesso em: 22 Setembro 2021.

MOLLER, Dietmar P.F. **Guide to Computing Fundamentals in cyber-Physical Systems:** Concepts, Design Methods, and Applications. ilustrada. ed. [S.l.]: Springer, 2016. 422 p.

MONTELEONE, Sergio ; SAMPAIO, Mauro ; MAIA, Rodrigo Filev. Conference on Information Systems and Technologies (CISTI). **A novel deployment of smart Cold Chain system using 2G-RFID-Sys temperature monitoring in medicine Cold Chain based on Internet of Things**, Medellin, Colombia, 2017. 8. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/7975685/. Acesso em: 22 Setembro 2021.

MOREIRA, Jorge R. H. ; BERNARDINO, Heder S. ; VIEIRA, Alex B.. Predição de Óbito Neonatal usando Dados dos Sistemas de Informação do SUS e de Censo Demográfico. **Anais do Simpósio Brasileiro de Computação Aplicada à Saúde (SBCAS)**, p. 234-245, jun. 2022. ISSN 2763-8952. Disponivel em: https://sol.sbc.org.br/index.php/sbcas/article/view/21635.

NDIAYE, Babacar Mbaye ; TENDENG, Lena ; SECK, Diaraf. Comparative prediction of confirmed cases with COVID-19 pandemic by machine learning, deterministic and stochastic SIR models, abr. 2020. Disponivel em: http://arxiv.org/abs/2004.13489.

NETBEANS. **Apache NetBeans,Fits the Pieces Together:** Development Environment, Tooling Platform and Application Framework, 2023. Disponivel em: https://netbeans.apache.org//front/main/.

OPENWEATHERMAP. OpenWeatherMap API, 10 dez. 2023. Disponivel em: https://openweathermap.org/api. Acesso em: 10 dez. 2023.

OXXCODE. OxxxCode Mobility Solutions. **Middleware RFID**, 2021. Disponivel em: http://www.oxxcode.com.br/middleware-rfid/. Acesso em: 10 Janeiro 2021.

PALACIO, Mauricio González *et al.* A novel ubiquitous system to monitor medicinal cold chains in transportation. **2017 12th Iberian Conference on Information Systems and Technologies (CISTI)**, Medellín, Colombia, 2017. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/7975685/. Acesso em: 22 Setembro 2021.

PEDREGOSA, F. *et al.* Scikit-learn: Machine Learning in Python. **Journal of Machine Learning Research**, 2013. 2825-2830.

PERLAZA, Claudia Lorena *et al.* Factors of abandonment of tuberculosis treatment in the public health network. **Revista de Saude Publica**, v. 57, 2023. ISSN 00348910.

RAMPAZZO, Lino. **Metodologia Científica:** Para alunos dos cursos de graduação e pós-graduação. 3ª. ed. São Paulo: Loyola, 2005. 51 p.

RAWAT, Arvind Singh *et al.* LORA (Long Range) and LORAWAN technology for IoT applications in Covid-19 pandemic. **Proceedings - 2020 International Conference on Advances in Computing, Communication and Materials, ICACCM 2020**, 21 ago. 2020. 419-422. Disponivel em: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9213067. Acesso em: 04 ago. 2022.

ROBERT, C.P. A Defense of the Bayesian Choice. **The Bayesian Choice**, New York, NY, 1994. pp-369-380. Disponivel em: https://link-springer-com.ez2.periodicos.capes.gov.br/chapter/10.1007/978-1-4757-4314-2_10#citeas. Acesso em: 30 set. 2021.

RODERUS, Jens ; LARSON, Simon ; PIHL, Eric. **Hadoop scalability evaluation for machine learning algorithms on physical machines:** Parallel machine learning on computing clusters. [S.l.]: Springer, 2021.

RUSDAH ; AGNI BREGASTANTYO, Brian ; EVERHARD RIWUROHI, Jan. Prognosis Model of The Treatment Period of Tuberculosis Patients with Medication Compliance Parameters using The Gradient Boosting Algorithm, dez. 2023. 220-225. Disponivel em: https://ieeexplore.ieee.org/document/10467254/.

SCHINOFF, Roberto Amaral ; LUZ, Charlene Bitencourt Soster ; AGUIAR, Fernanda Rocha de Aguiar. **Gestão de tecnologia e informação em logística**. São Paulo: SAGAH, 2019. Disponivel em: https://integrada.minhabiblioteca.com.br/#/books/9788595028487/. Acesso em: 10 Janeiro 2021.

SCIKIT Learn. **Machine Learning in Python**, 2023. Disponivel em: https://scikit-learn.org/.

SEMTECH CORPORATION. Building a LoRa Device End-to-End with Arduino. **LoRa - Developer Portal**, 2023. Disponivel em: https://lora-developers.semtech.com/documentation/tech-papers-and-guides/building-a-lora-based-device-end-to-end-with-arduino/introduction/. Acesso em: 07 ago. 2022.

SHATNAWI, Amjed *et al.* Shear Strength Prediction of Slender Steel Fiber Reinforced Concrete Beams Using a Gradient Boosting Regression Tree Method, 12, n. 5, maio 2022. Disponivel em: https://www.researchgate.net/publication/360187776_Shear_Strength_Prediction_of_Slender_Steel_Fiber_Reinforced_Concrete_Beams_Using_a_Gradient_Boosting_Regression_Tree_Method.

SUGUNA, R. ; SATHISHKUMAR, P ; DEEPA, S. User Location and Collaborative based Recommender System using Naive Bayes Classifier and UIR Matrix. **2020 4th International Conference on Electronics, Communication and Aerospace Technology (ICECA)**, Tamilnadu, India, 5, 2020. 8. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/9297589/. Acesso em: 06 Outubro 2021.

TEBALDI GOMES, Pedro César. Datageeks. **Algoritimo de Machine Learnig:** classificação com Naive Bayesdatageeks, 2019. Disponivel em: https://www.datageeks.com.br/naive-bayes/. Acesso em: 06 Outrubro 2021.

TEJA, Ravi. Eletcronics Hub. **Getting Started with ESP32 | Introduction to ESP32**, 17 fev. 2021. Disponivel em: https://www.electronicshub.org/getting-started-with-esp32/. Acesso em: 10 jul. 2021.

TEMPOLA, Firman ; MUHAMMAD, Miftah ; KHAIRAN, Amal. Naive Bayes Classifier for Prediction of Volcanic Status in Indonesia. **2018 5th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)**, Ternate, Indonesia, 2018. 5. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/6970284/. Acesso em: 06 Outubro 2021.

VEENA KUMARI, H. M. ; SURESH, D. S. ; DHANANJAYA, P. E.. Clinical Data Analysis and Multilabel Classification for Prediction of Dengue Fever by Tuning Hyperparameter using GridsearchCV, 2022. 302-307.

WERNER DE VARGAS, Vitor *et al.* Imbalanced data preprocessing techniques for machine learning: a systematic mapping study. **Knowledge and Information Systems**, v. 65, n. 1, 2023. ISSN 02193116.

YANG, Feng-Jen. An Implementation of Naive Bayes Classifier. **2018 International Conference on Computational Science and Computational Intelligence (CSCI)**, 2018. pp. 301-306. Disponivel em: https://ieeexplore-ieee-org.ez2.periodicos.capes.gov.br/document/8947658. Acesso em: 01 out. 2021.

ANEXO A - CÓDIGO EXEMPLO EM PYTHON

Nos algoritmos de aprendizagem, conforme descrito em Scikit Learn (2023), os modelos se beneficiam da padronização do conjunto de dados. Isso ocorre porque a padronização torna as características do conjunto de dados mais homogêneas, o que facilita o aprendizado do modelo. A padronização é feita subtraindo a média de cada característica e dividindo-a pelo desvio padrão. Um código exemplo é mostrado na Figura 45.

Figura 45 – Exemplo de *Preprocessing - LabelEncoder - Scikit-learn.*

```
>>> le = LabelEncoder()
>>> le.fit(["paris", "paris", "tokyo", "amsterdam"])
LabelEncoder()
>>> list(le.classes_)
['amsterdam', 'paris', 'tokyo']
>>> le.transform(["tokyo", "tokyo", "paris"])
array([2, 2, 1]...)
>>> list(le.inverse_transform([2, 2, 1]))
['tokyo', 'tokyo', 'paris']
```

Fonte: (Scikit Learn, 2023).

Na Figura 46 apresenta-se um exemplo para implementação do modelo *Multi-layer Perceptron Classifier* (*MLPClassifier),* Classificador Perceptron multicamadas que faz parte do pacote Python *sklearn.neural_network.MLPClassifier*, otimiza a função de perda logarítmica usando *LBFGS* ou descida gradiente estocástica.

Figura 46 – Exemplo *Multi-layer Perceptron Classifier*.

```
>>> from sklearn.neural_network import MLPClassifier
>>> from sklearn.datasets import make_classification
>>> from sklearn.model_selection import train_test_split
>>> X, y = make_classification(n_samples=100, random_state=1)
>>> X_train, X_test, y_train, y_test = train_test_split(X, y, stratify=y,
...                                                     random_state=1)
>>> clf = MLPClassifier(random_state=1, max_iter=300).fit(X_train, y_train)
>>> clf.predict_proba(X_test[:1])
array([[0.038..., 0.961...]])
>>> clf.predict(X_test[:5, :])
array([1, 0, 1, 0, 1])
>>> clf.score(X_test, y_test)
0.8...
```

Fonte: (Scikit Learn, 2023).

Outra codificação pode ser observada na Figura 47. Trata-se do modelo *GradientBoostingClassifier*.

Nela percebe-se a divisão de um conjunto de dados para treino e teste (X_train e X_test) juntamente com a divisão da variável alvo ou Target, y. Neste caso, (y_train e y_test).

Figura 47 - Exemplo de implementação *GradientBoostingClassifier*.

```
>>> from sklearn.datasets import make_hastie_10_2
>>> from sklearn.ensemble import GradientBoostingClassifier

>>> X, y = make_hastie_10_2(random_state=0)
>>> X_train, X_test = X[:2000], X[2000:]
>>> y_train, y_test = y[:2000], y[2000:]

>>> clf = GradientBoostingClassifier(n_estimators=100, learning_rate=1.0,
...     max_depth=1, random_state=0).fit(X_train, y_train)
>>> clf.score(X_test, y_test)
0.913...
```

Fonte: (Scikit Learn, 2023).