Ministério da Educação
Universidade Tecnológica Federal do Paraná
Pró-Reitoria de Pesquisa e Pós-Graduação

RELATÓRIO FINAL DE ATIVIDADES DE INICIAÇÃO CIENTÍFICA

Estudo e Desenvolvimento de Envio Contínuo de Dados de Datasets com pré-processamento utilizando Data Mining

vinculado ao projeto

Estudo e Aplicações de Monitoramento Contínuo de Pessoas por meio de Redes Corporais Sem Fio e Internet das Coisas

Matheus Bueno Faria

Bolsista Modalidade/Agência ou Voluntário(a) Modalidade

Engenharia de Computação

Data de ingresso no programa: 08/2024

Prof. Dr. Rogério Santos Pozza

Área do Conhecimento: 1.03.00.00-7 — Ciência da Computação

CÂMPUS CORNÉLIO PROCÓPIO, 2025

MATHEUS BUENO FARIA
ROGÉRIO SANTOS POZZA

ESTUDO E DESENVOLVIMENTO DE ENVIO CONTÍNUO DE DADOS DE DATASETS COM PRÉ-PROCESSAMENTO UTILIZANDO DATA MINING

Relatório de Pesquisa do Programa de Iniciação Científica da Universidade Tecnológica Federal do Paraná.

CORNÉLIO PROCÓPIO, 2025

SUMÁRIO

INTRODUÇÃO

O estudo de monitoramento de abelhas tem uma grande importância na humanidade, devido ao papel essencial que elas exercem na polinização de culturas agrícolas e ecossistemas naturais. A produção de mel, pólen e própolis geram um grande impacto econômico na sociedade, uma vez que a quantidade e a qualidade desses produtos estão diretamente ligadas ao bem-estar das colmeias. Diante desse contexto, fica evidente a necessidade de um monitoramento eficiente, utilizando tecnologia como sensores, *Internet of things* e inteligência artificial para a detecção de doenças e no controle da aplicação de pesticidas. Esses recursos contribuem de forma eficiente para a preservação das colmeias e na tomada de decisões dos apicultores, maximizando a produção e prolongando o tempo de vida das abelhas.

Dentro do monitoramento de abelhas há uma grande dificuldade em identificar o estado da colmeia sem a intervenção direta de forma física pelo apicultor, dado que métodos tradicionais podem ser considerados inadequados devido ao estresse que pode causar às abelhas. Também há uma falta de soluções automatizadas e que sejam acessíveis aos pequenos e médios apicultores. Sendo assim, uma opção mais adequada seria a utilização de dados extraídos de diferentes colmeias de abelhas para a realização de uma análise criteriosa. Contudo, existem poucos *datasets* públicos relacionados ao tema de colmeia de abelhas, e grande parte deles não está padronizada ou completa, o que pode comprometer a análise. Além disso, existem desafios no que se refere a o que fazer com os dados coletados e de que forma pode-se interpretá-los.

Levando isto em consideração, uma possível solução seria realizar uma padronização de *datasets* por meio de mineração de dados e a aplicação de técnicas de aprendizado de máquina, com o objetivo de prever a presença ou ausência de uma abelha rainha com base nos dados da colmeia. Para isso, foram utilizadas bibliotecas específicas da linguagem *Python* para a aplicação das técnicas anteriormente citadas. Foram utilizados dados reais, tais como temperatura, umidade relativa do ar e pressão, extraídos de um *dataset* real de monitoramento de colmeia para o treinamento dos modelos. Foram utilizados os algoritmos supervisionados *Random Forest Classifier*, *K-Nearest Neighbors*, *Support Vector Machine* e *Gradient Boosting*, separando em duas metodologias de treinamento: uma com validação cruzada usando 5 *folds* e outra apenas com um conjunto de dados teste separado. No final do trabalho, os resultados obtidos foram comparados com base em métricas de desempenho e discutidos quanto à sua eficácia. Além disso, toda a codificação desenvolvida está disponível na plataforma GitHub [1].

Este trabalho de iniciação científica está vinculado ao projeto de pesquisa intitulado *Estudo e Aplicações de Monitoramento Contínuo de Pessoas por meio de Redes Corporais Sem Fio e Internet das Coisas*. Embora com foco seja o monitoramento remoto de colmeias de abelhas, compartilha os mesmos fundamentos tecnológicos do projeto de pesquisa, como o uso de sensores de baixo consumo energético, comunicação sem fio, transmissão de dados em tempo real e aplicação de algoritmos de aprendizado de máquina. Ao aplicar essas tecnologias em um contexto alternativo, este trabalho amplia o estudo de técnicas genéricas de aquisição, envio e análise de dados que podem ser adaptadas tanto ao monitoramento humano quanto a sistemas ambientais inteligentes, e contribui no escopo e na aplicabilidade de sistemas de redes sensoriais sem fio desenvolvidos no âmbito do projeto de pesquisa.

MONITORAMENTO REMOTO DE ABELHAS

Um sistema de monitoramento remoto de abelhas tem como objetivo principal realizar a coleta e envio de dados de colmeias, sem interferir no processo natural de produção de mel. Existem várias formas para realizar a obtenção dos dados, sendo a utilização de sensores o método

mais comum. Estes sensores capturam informações como temperatura, umidade relativa do ar, movimentação das abelhas, peso, pressão atmosférica entre outras. Os sensores são integrados a um sistema de *hardware* para armazenar os dados coletados, ou também para enviar informações a um servidor de banco de dados externo (*cloud*).

Além da escolha adequada dos sensores, fatores como taxa de amostragem, latência na transmissão, sincronização temporal entre sensores e consumo de energia dos dispositivos são determinantes para a eficácia do monitoramento contínuo. Tais parâmetros impactam diretamente na precisão dos dados e na viabilidade de operação em campo, especialmente em sistemas que operam de forma autônoma e com alimentação por bateria, como os descritos nos trabalhos relacionados.

Com o monitoramento remoto de abelhas é possível observar possíveis situações de estresse das abelhas, como descrito em Singh et al. [2]. Portanto, isto pode impactar de forma significativa na vida da colônia de abelhas, na produção de mel, polinização e procriação. Além disso, com os dados coletados podem ser realizadas técnicas de pré-processamento, como foi realizado em Senger et al. [3] para extrair informações a partir dos dados, ou até mesmo aplicar técnicas de *Machine Learning*, como descrito em Chen et al. [4], em que foram comparadas e avaliadas 3 técnicas de aprendizado de máquina.

A coleta e envio de dados em tempo real tem como objetivo caracterizar o estado atual da colônia de abelhas. No trabalho de Singh et al. [2] é realizado um monitoramento em tempo real, em que o apicultor pode visualizar o estado atual da colmeia de abelhas por meio de um aplicativo, em que é possível ver os dados referentes a temperatura e umidade. O sistema também pode alertar ao usuário anomalias que ocorreram dentro da colmeia de abelhas, tais como uma variação inesperada da temperatura ou umidade dentro da colmeia, imprescindíveis no monitoramento de abelhas. Em Jailis et al. [5] também é feito um monitoramento remoto em tempo real, no qual os dados coletados são armazenados na plataforma *ThingSpeak* a cada 30 minutos.

Coleta de Dados.

A coleta de dados de colmeias é realizada por meio de sensores. Na Tabela 1 são apresentadas estimativas para os valores das medidas que foram coletadas dos trabalhos. Em Verma et al. [6] são utilizados dois sensores: HTS221, que é responsável por coletar a temperatura e umidade relativa do ar, e o VEML770, o qual possibilita medir a intensidade luminosa do ambiente interno da colmeia. O *Amazon Web Service* (AWS) *IoT Core*, que é um serviço de comunicação seguro da empresa Amazon, é utilizado para enviar os dados capturados a um servidor de armazenamento da própria Amazon (S3).

Em Singh et al. [2] é utilizado o sensor DHT11 conectado a um microcontrolador ESP8266. O DHT11 coleta informações referentes a temperatura e umidade relativa do ar. As informações são transferidas para o microcontrolador, e então são enviadas ao serviço de nuvem *Blynk Cloud*. Por meio do aplicativo *Blynk App*, os apicultores podem visualizar estes dados por aparelhos eletrônicos (*laptops* e *smartphones*), como mostra a Figura 1.

Em Hadjur, Ammar e Lefevre [7] os dados são coletados por um sensor SHT31, 3 microfones USB e uma câmera. Um dispositivo Raspberry Pi 3b+ é responsável por coletar os dados dos equipamentos e, em seguida, transferi-los para um servidor em nuvem por transmissão Wi-Fi. A Figura 2 mostra o sistema implementado.

Em Jailis et al. [5] é utilizado um sensor DHT22 para monitorar a temperatura, umidade relativa do ar e peso da colmeia. Os dados são coletados a cada 30 minutos em um período de 30 dias e são armazenados em nuvem. Além disso, as condições meteorológicas foram registradas diariamente, indicando se o tempo estava ensolarado ou nublado. Os níveis de temperatura e

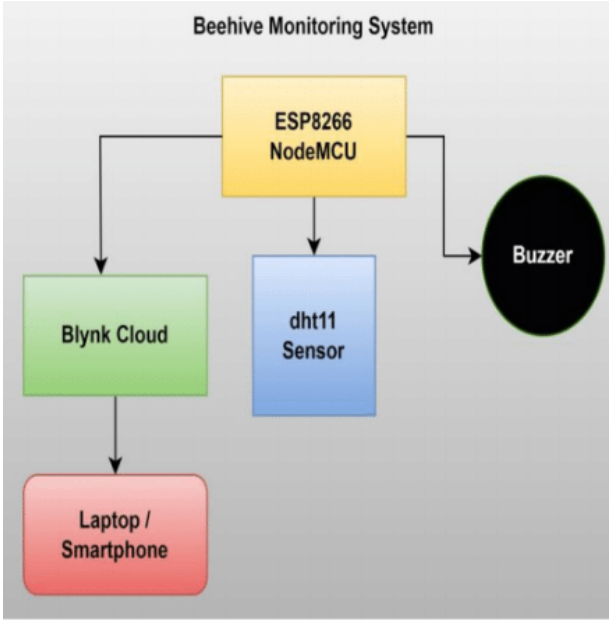

Figura 1. Diagrama de blocos do monitoramento das condições da colmeia do estudo Singh et al. [2], onde são utilizados um sensor DHT11, um módulo embarcado ESP8266, buzzer, um serviço em nuvem e um dispositivo de uso pessoal para o monitoramento.

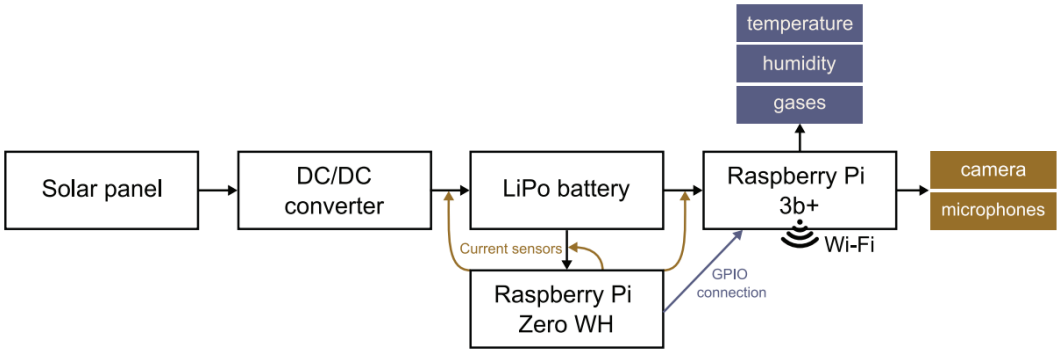

Figura 2. Arquitetura do sistema implantado Hadjur, Ammar e Lefevre [7]. O Raspberry Pi 3b+, que possui uma conexão com os sensores, câmera e microfone, é alimentado por uma bateria LiPo, que é recarregada por painel solar.

umidade relativa do ar são enviados ao banco de dados implementado no *ThingsSpeak*, que consiste em uma plataforma IoT que analisa os dados. O sensor DHT22 é conectado a um Arduíno, como mostra a Figura 3.

Em Chen et al. [4] é feito um monitoramento de áudio para gravar o som produzido pelas abelhas. Para isso, utilizou-se Raspberry Pi 3B+, cartão de memória SD (*Secure Digital*), microfone e uma fonte de alimentação, como mostra a Figura 4. O Raspberry executa algoritmos de classificação e clusterização. Os resultados analisados e classificados são armazenados no cartão SD, e em seguida são enviados a um servidor em nuvem por meio de uma conexão *wireless* empregada no sistema para que os apicultores analisem os resultados.

Em Senger et al. [3] são utilizados 3 sensores: DS18B20, BME280 e Bosche H30/H40. O DS18B20 detecta a temperatura em 5 locais diferentes dentro da colmeia, e também coleta os dados da temperatura externa. O sensor BME280 coleta dados referentes a umidade relativa do ar, pressão e temperatura interna da colmeia. O Bosche H30/H40 coleta o peso da colmeia. Os

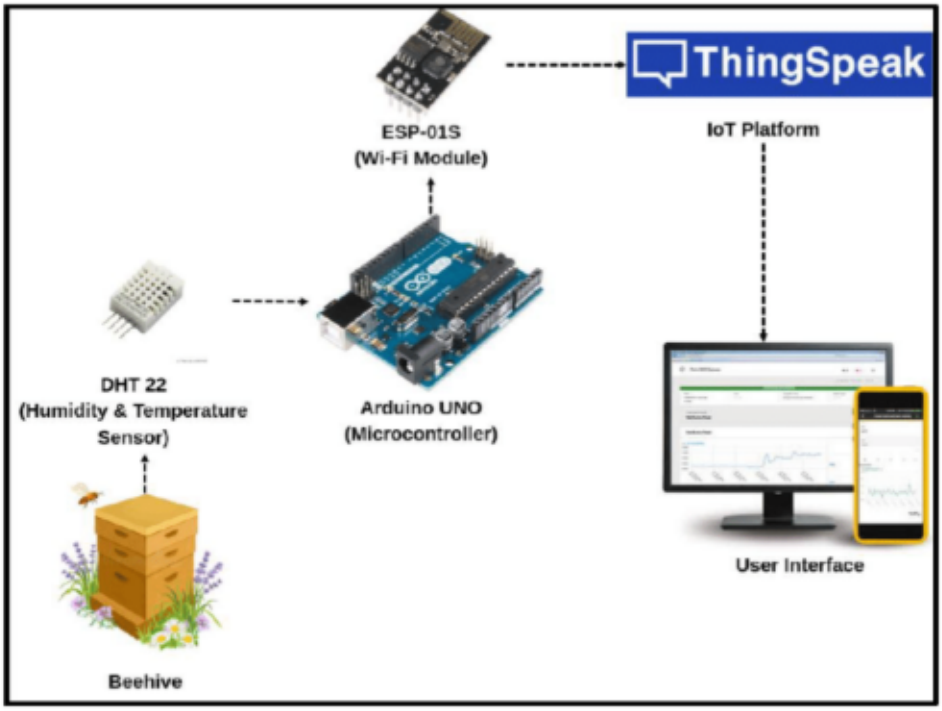

Figura 3. *Framework* do sistema de monitoramento Jailis et al. [5]. É utilizado um sensor DHT 22 para coletar os dados de temperatura e umidade da colmeia. Os dados são transferidos para o arduino UNO, que envia os dados ao ThingSpeak por meio do módulo Wi-Fi ESP-015.

sensores estão conectados em uma placa de circuito com um microcontrolador ESP32. Os dados são transferidos a um servidor central e armazenados em um banco de dados Influx[1].

Medidas e Limites.

Em Alleri et al. [8] é descrito uma revisão sistemática sobre valores das medidas estudadas com base em vários artigos. Ela é de suma relevância, pois os parâmetros externos como temperatura, umidade relativa do ar, velocidade do vento, chuvas e intensidade luminosa podem influenciar na coleta de pólen.

De acordo com a revisão, os valores ideais para a umidade relativa do ar dentro da colmeia estariam na média de 70%, enquanto que para época de procriação (chocagem dos ovos) estariam na faixa de 90% a 95%, e que valores abaixo de 50% impediriam esta fase. Também foi avaliada a umidade relativa na perspectiva de contaminação (como desenvolvimento de parasitas e patógenos) em que níveis entre 55% e 70% favoreceriam a sua procriação, enquanto que valores acima do limite reduziriam sua reprodução.

Os valores ideais de temperatura variam dentre diferentes cenários. Em um cenário em que esteja próxima a época de procriação das abelhas, quando a temperatura ultrapassa os 35°C ou fica abaixo de 20°C, o sistema de um monitoramento de abelhas aciona uma notificação ao apicultor, com o intuito de que seja realizada alguma ação para reduzir este valor.

Na revisão não há valores referentes a uma média ou variação da massa da colmeia. Contudo, foi feita uma observação em uma colmeia de abelhas na China, na qual variações do peso da colmeia foram detectadas de acordo com a saída e entrada de abelhas. Por conta do néctar que trazem, a massa da colmeia aumenta um pouco.

[1] Disponível em: `www.influxdata.com`.

Figura 4. Diagrama esquemático do estudo em [4], em que é mostrado o *hardware* do dispositivo usado, os recursos de extração, algoritmos usados para a análise, serviços em nuvem utilizados e condições da colmeia para o estudo

Em Jailis et al. [5] existe uma breve conclusão acerca das medidas trabalhadas para a abelha sem ferrão. Segundo o estudo, a temperatura interna da colmeia não deve exceder 35°C e o nível da umidade relativa do ar não deve exceder 78% para atingir boas condições de vida para abelhas sem ferrão. Além disso, apesar das variações de temperatura e umidade relativa do ar, o peso da colmeia registrou um aumento de 0.35 Kg durante um período de 30 dias.

Tabela 1. Estimativas de valores ideais para as medidas.

Condição –	Temperatura [°C]	Umidade relativa [%]	Peso [Kg]
Normal	X	Média de 70%	X
Procriação	Entre 20°C e 35°C	Entre 90% e 95%	X
Abelha sem ferrão	Não exceder 35°C	Não exceder 78%	X
Parasitas	X	Não estar entre 55% e 70%	X

Trabalhos Relacionados.

Verma et al. [6] descrevem como são transmitidos os vídeos obtidos pelo monitoramento

de uma colmeia. Nesse estudo são utilizados os protocolos HTTP (*Hypertext Transferer Protocol*) e MQTT (*Message Queuing Telemetry Transport*), nos quais são avaliados a partir da quantidade e tipos de dados transferidos (*frames*, duração dos vídeos em minutos e tamanho dos vídeos).

No trabalho Singh et al. [2] são utilizados sensores e um Arduíno para monitorar a temperatura e umidade das colônias de abelhas. Um microcontrolador ESP8266 e um DHT11 são usados para medir a temperatura e umidade. O artigo demonstra a relação entre a atividade das abelhas e fatores ambientais circundantes.

No trabalho Hadjur, Ammar e Lefevre [7] é abordado o consumo de energia, com foco na análise do consumo de energia do sistema. Apesar de o foco do trabalho ser analisar o consumo de energia gasto do sistema implementado, são coletados dados referentes a temperatura, umidade relativa do ar, gases, amostras de som e imagens. Nele foi possível concluir que, para caso com vários clientes conectados no servidor, seria mais eficiente a adição de mais servidores em nuvem para um consumo menor de energia.

O trabalho Jailis et al. [5] aborda os desafios e aplicações do meio de conexão sem fio utilizando um módulo Wi-Fi. Esse estudo investiga esta aplicação em diferentes áreas, em que utiliza um Arduino Uno ATmega328P e um DHT22 para a realização do monitoramento da temperatura e umidade relativa dentro da colmeia de forma contínua.

O artigo Chen et al. [4] enfatiza a frequência de vibrações da colmeia por meio de um sistema de monitoramento de áudio. De acordo com os resultados, o KNN apresentou melhor acurácia. No estudo, foram diferenciadas colmeias que possuíam uma abelha rainha daquelas que não possuíam, além de distinguir as que apresentavam infecções por vírus ou exposição a pesticidas e, por fim, aquelas em condições normais, sem anomalias.

No trabalho Senger et al. [3] são coletadas informações de 78 colmeias de abelhas na Alemanha, que dentre elas incluem: dados da temperatura externa e interna da colmeia, umidade relativa, peso e pressão. O estudo também categorizou 4 tipos de eventos: onde havia ou não abelhas (enxame), e se houve ou não a morte da colônia de abelhas.

A Tabela 2 mostra os trabalhos relacionados com suas principais características resumidamente.

METODOLOGIA

Fundamentação Teórica.

Os algoritmos de inteligência artificial são amplamente utilizados em várias campos da atualidade, tais como: medicina, tendências do mercado, agricultura, etc. Os algoritmos, especificamente dentro da subárea de *machine learning*, são divididos em duas categorias: supervisionados e não supervisionados. A primeira categoria se caracteriza pelo uso de um rótulo (ou *target*), em que o modelo irá tentar prever uma variável categórica (por exemplo, diagnóstico de doenças como *positivo* ou *negativo*) ou contínua (por exemplo, prever o preço de um produto) a partir de um conjunto de dados. A segunda categoria não possui um rótulo, sendo assim seu objetivo principal está em tentar encontrar padrões nos dados escolhidos para o treinamento. Essa abordagem pode ser usada em agrupamento de dados (*clustering*), como na segmentação de clientes em tendência de vendas, e na redução de dimensionalidade, tal como em técnicas para compressão de dados e visualização.

Os algoritmos *Random Forest Classifier*, *K-Nearest Neighbors*, *Support Vector Machine* e *Gradient Boosting* foram desenvolvidos para a realização de tarefas como classificação e regressão. Enquanto que o *Random Forest Classifier* e *Gradient Boosting* são métodos baseados

Tabela 2. Características dos trabalhos presentes na literatura.

Trabalhos	[6]	[2]	[7]	[5]	[4]	[3]
Comunicação	Protocolos MQQT e HTTP, servidor em nuvem	Sem-fio, servidor em nuvem	Sem-fio, servidor em nuvem	Sem-fio (Módulo Wi-Fi)	Sem-fio vinculado a IoT	Sem-fio LAN
Tipos de dados	Temperatura, umidade, luminosidade e contagem de insetos	Temperatura e umidade	Temperatura, umidade, peso e consumo de energia	Temperatura e umidade	Temperatura, umidade e frequência	Temperatura interna e externa, umidade, peso e pressão
TEC IA	Nenhuma	Nenhuma	Redes neurais convolucionais, *support vector machine*	Nenhuma	KNN, LR, SVM e SVD	Pré-processamento dos dados
Local	Suíça	India	França	Malásia	Taiwan	Alemanha
Dataset	X	X	Existe, mas não disponível	Existe, mas não disponível	Existe, mas não disponível	Existe, os dados são gravados em vários arquivos em formato csv
Abelha	X	X	X	Abelhas sem ferrão (stingless)	X	X
Alvo	X	X	Energia gasta no monitoramento	Coleta de dados de abelhas sem ferrão	Condições da colônia	X

em árvores, os métodos *K-Nearest Neighbors* e *Support Vector Machine* trabalham a partir de instâncias e vetores.

Esses algoritmos foram escolhidos para o trabalho por conta de suas características específicas, tais como: capacidade de lidar com múltiplas variáveis ambientais (*Random Forest Classifier*), processar uma pequena quantidade de dados e não assumir suposições sobre a distribuição dos dados (*K-Nearest Neighbors*), lidar com dados de alta dimensionalidade e não linearmente separáveis (*Support Vector Machine*) e bom funcionamento em dados estruturados e com muitas variáveis, além de ser um modelo altamente preciso e robusto por conta do aprendizado contínuo a partir dos erros anteriores (*Gradient Boosting*).

Algoritmos de Classificação.

Random Forest Classifier é um algoritmo de aprendizado de máquina que pode ser utilizado para modelos de classificação ou regressão linear. Sua categoria é a de supervisionado, ou seja, necessita de um rótulo (*target*) para a sua implementação. Ele utiliza o método de árvores de decisão, que consiste em uma ramificação para o modelo decidir se deve ou não seguir para uma

respectiva classificação. O algoritmo realiza a combinação de várias árvores de decisão, o que ajuda a mitigar o *overfitting* (super ajuste aos dados de treinamento) e sensibilidade a ruídos. Um exemplo de como funciona a árvore está exemplificado na Figura 5, na qual é realizada uma decisão a partir dos dados fornecidos de um veículo.

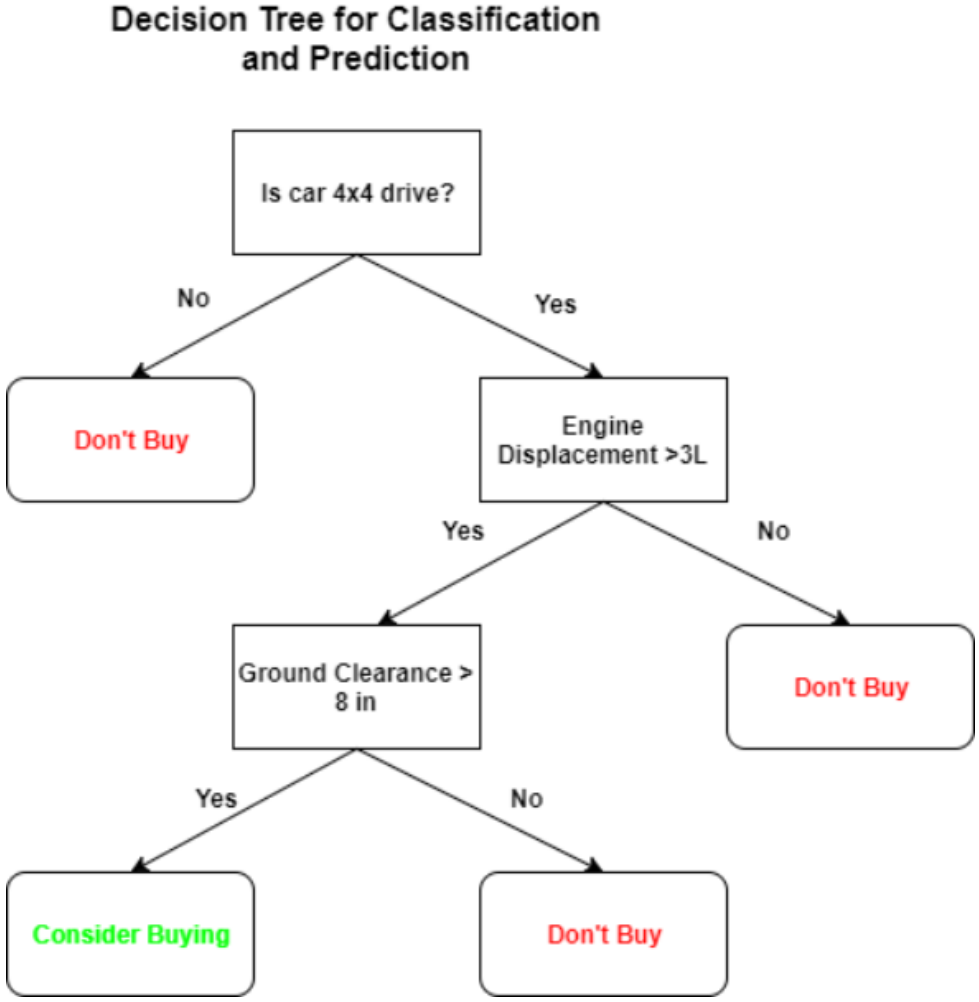

Figura 5. Árvores de decisão de um algoritmo *Random Forest*, em que há uma árvore simples de decisão.

A Figura 6 mostra uma árvore de decisão de maior complexidade, na qual se divide em três ramificações, cada uma treinada com um subconjunto aleatório dos dados.

O *K-Nearest neighbors* é um algoritmo de aprendizado de máquina utilizado também para tarefas de classificação e regressão. Ele é um algoritmo supervisionado, em que seu método de funcionamento consiste na classificação de cada amostra de um conjunto de dados a partir da sua distância em relação aos vizinhos mais próximos. Em termos de classificação, a tarefa do algoritmo é determinar a classe de uma amostra com base nas classes dos seus K vizinhos mais próximos, onde K é um hiperparâmetro definido pelo usuário. A Figura 7 mostra um exemplo, em que há um elemento verde que está prestes a ser classificado pelo algoritmo. Neste caso, existem mais vizinhos vermelhos próximos dele (2) do que vizinhos azuis (1). Sendo assim, ele será classificado como vermelho, de acordo com a Figura 8

O *Support Vector Machine* é um algoritmo de aprendizado de máquina supervisionado utilizado para classificação e regressão linear (embora seja mais comum sua utilização em tarefas de classificação). Sua principal característica é encontrar uma linha de separação, também

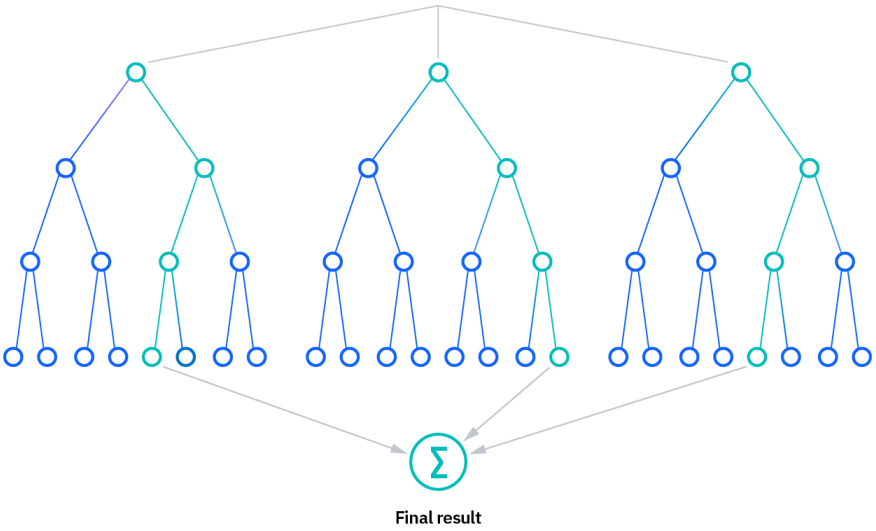

Figura 6. Árvores de decisão de um algoritmo *Random Forest*, em que há múltiplas árvores de decisão.

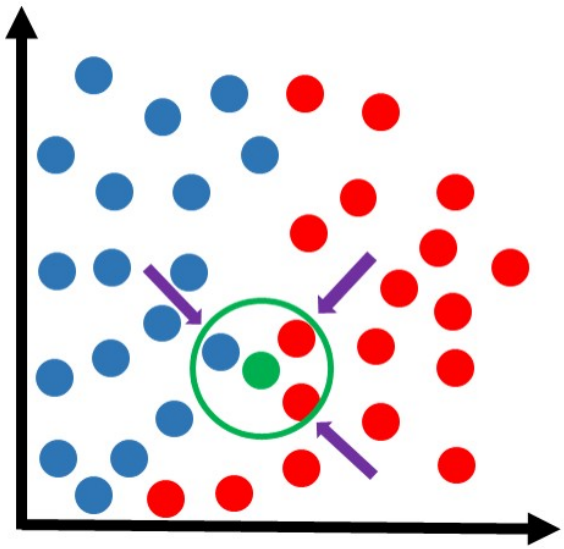

Figura 7. Gráfico representado atributos de um *dataset* genérico, em que o atributo verde está próximo de alguns vizinhos.

chamada de hiperplano, entre dados de duas classes distintas, como mostra a Figura 9. A margem é definida como a distância entre o hiperplano e os pontos mais próximos de cada classe. O objetivo principal do algoritmo é encontrar o hiperplano que maximiza essa distância (margem) entre as duas classes. Ele funciona bem em domínios onde existe uma clara margem de separação. Contudo, não é eficiente em *datasets* com um grande volume de dados ou de ruídos.

O *Gradient Boosting* é um algoritmo de aprendizado de máquina supervisionado, também utilizados para tarefas de regressão linear e classificação. Ele funciona a partir da criação de uma corrente de modelos fracos, em que cada um desses modelos tem como objetivo minimizar o erro do modelo anterior, melhorando a performance gradualmente. A Figura 10 mostra o funcionamento do algoritmo: a cada modelo que é criado e treinado, são produzidos resíduos, que é definido como a distância entre o que foi previsto e o valor real. No próximo modelo a

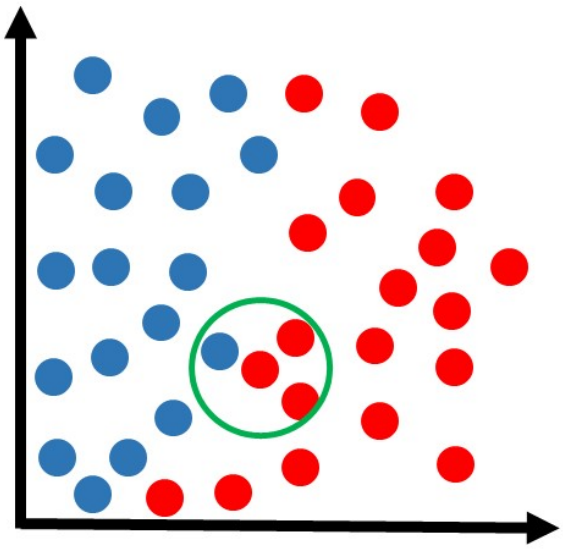

Figura 8. Gráfico representado atributos de um *dataset* genérico, em que o atributo verde foi classificado em vermelho por conta de seus vizinhos próximos.

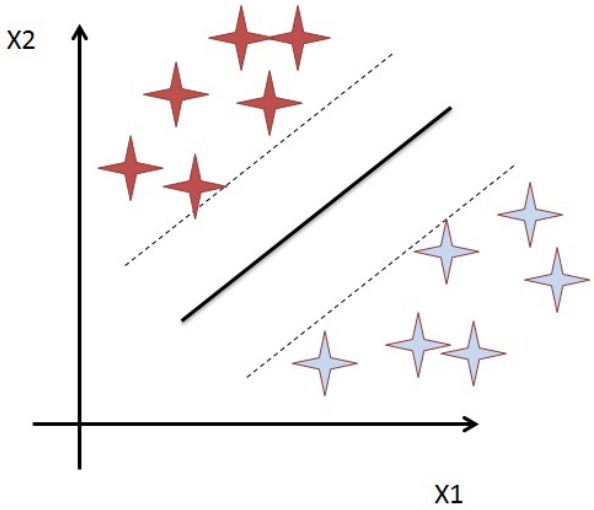

Figura 9. Gráfico representando a margem de um algoritmo SVM, separando duas classes distintas.

ser criado é ajustado com base no resíduo gerado pelo modelo anterior. Em seguida, um novo resíduo é calculado. E assim é feito sucessivamente, até que por fim chegue a um modelo final que é a somatória dos ajustes de todos os modelos fracos.

No trabalho serão analisadas 4 medidas para avaliação dos modelos: acurácia, *F1-score*, *recall* e ROC AUC. A acurácia é responsável por medir a proporção total de previsões corretas feitas pelo algoritmo, incluindo verdadeiros positivos e verdadeiros negativos, em relação ao número total de amostras. Ela é uma métrica considerada geral em termos de desempenho, contudo em conjuntos de dados desbalanceados pode não apresentar uma boa confiabilidade.

O *F1-score* é a média harmônica entre *recall* e precisão, que pode ser bastante útil em casos em que exista um desbalanceamento entre as classes. Um alto desempenho nesta medida indica que o modelo apresenta um bom equilíbrio entre identificar de forma correta os verdadeiros positivos e evitar falsos positivos.

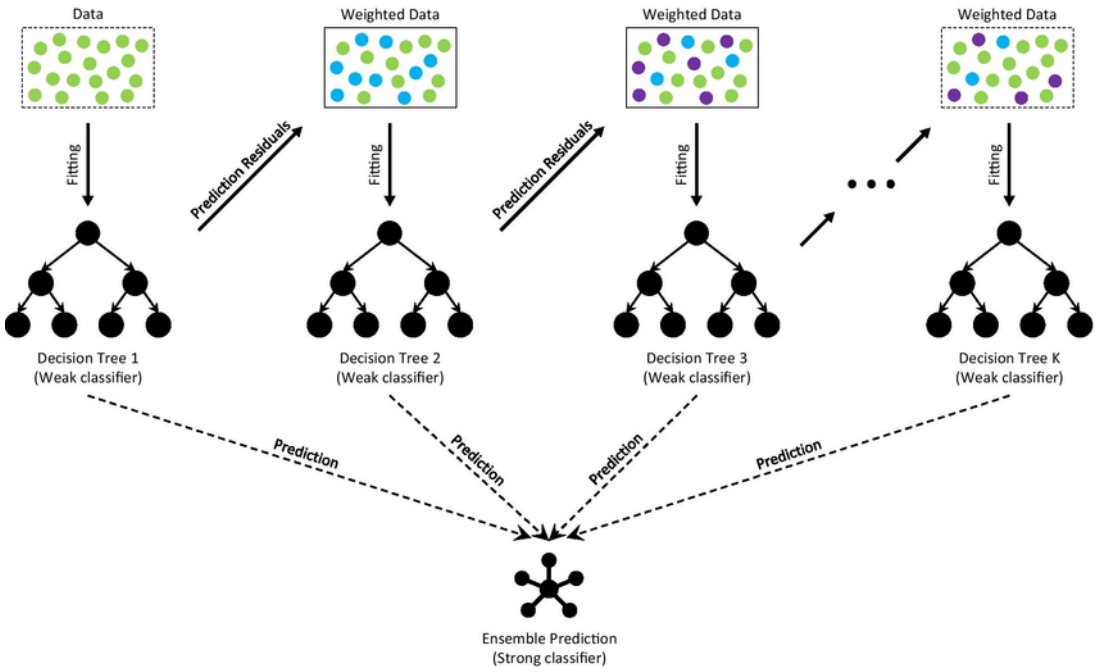

Figura 10. Exemplo de um algoritmo *Gradient Boosting Decision*, no qual o próximo modelo é aprimorado a partir da correção do erro do modelo anterior.

O *recall*, conhecido também pelo termo sensibilidade, indica a proporção de instâncias positivas que foram identificadas de forma correta pelo modelo. Ou seja, se o modelo possuir um alto desempenho nessa medida, indicará que o mesmo consegue detectar com alta eficiência a presença da rainha na colmeia.

O ROC AUC representa a área sob a curva ROC (*Receiver Operating Characteristic*), que tem como objetivo mostrar a relação existente entre a taxa de verdadeiros positivos e a taxa de falsos positivos. Desta forma, um alto valor de ROC AUC indicaria que o modelo possui boa capacidade de discriminação entre as classes, ainda que em cenários em que exista a possibilidade de desbalanceamento entre as classes.

DESENVOLVIMENTO

Limpeza e pré-processamento dos dados.

Para serem construidos os modelos, foi utilizado o *dataset* [9] “all_data_updated.csv”, disponibilizado na plataforma Kaggle. Os dados foram coletados a partir de um dispositivo IoT combinado com um módulo Wi-Fi ESP32, um módulo de microfone INMP441 (usado para coletar o som das abelhas) e um sensor de temperatura e umidade relativa do ar BME280. O processo de mineração de dados foi realizado baseando-se numa etapa sequencial de processos, tal como pode ser visto no diagrama da Figura 11.

O arquivo em formato csv possui 1275 linhas e 23 colunas. Foram realizadas modificações para o pré-processamento e limpeza dos dados, como a inserção de valores com a média aritmética de colunas que possuíam valores nulos. Além disso, valores considerados *outliers* detectados na visualização dos dados foram substituídos pela média aritmética da própria coluna. Feitas essas modificações, o *dataset* foi salvo em um arquivo nomeado “all_data_updated_classifier”. Todas as mudanças foram feitas a partir da linguaguem *Python*, com auxílio das bibliotecas *pandas*,

Figura 11. Diagrama representando o processo de mineração e pré-processamento dos dados.

numpy, *seaborn* e *matplotlib.pyplot*. A biblioteca *pandas* foi usada inicialmente para a realização de leitura. As bibliotecas *seaborn* e *matplotlib.pyplot* foram usadas para a visualização dos dados, bastante úteis para a detecção de *outliers* e valores nulos. Em seguida, a biblioteca *pandas* em conjunto com a biblioteca *numpy*, foram utilizadas para a realização de limpeza e normalização dos dados. E por fim, a biblioteca *pandas* foi usada para a salvar os dados modificados em um novo arquivo csv.

Treinamento dos modelos.

O *dataset* possui várias colunas, dentre as quais se destacam *hive temp*, *hive humidity*, *hive pressure*, *weather temp*,*weather humidity*, *weather pressure*, *wind speed* e *rain*, que correspondem, respectivamente, a temperatura interna da colmeia, umidade interna da colmeia, pressão interna da colmeia, temperatura do ambiente externo, umidade do ambiente externo, pressão do ambiente externo, velocidade do vento e chuva. As colunas podem ser observadas na Figura 12, em que são mostrados alguns exemplos dos dados existentes dentro do *dataset*.

	hive temp	hive humidity	hive pressure	weather temp	weather humidity	weather pressure
0	36.42	30.29	1007.45	26.68	52	1013
1	33.56	33.98	1006.93	25.99	53	1012
2	29.01	42.73	1006.68	24.49	56	1012
3	30.51	36.74	1006.68	22.97	59	1012
4	30.32	35.55	1006.58	21.52	61	1012
...	...	...	...	...	...	...
1270	50.84	11.99	1010.21	23.58	55	1015
1271	49.58	11.60	1009.81	25.60	51	1015
1272	45.83	15.36	1009.80	26.49	49	1015
1273	35.82	23.48	1009.26	27.33	46	1014
1274	31.55	27.90	1008.78	26.90	45	1014

1275 rows × 6 columns

Figura 12. Tabela apresentando uma amostra dos dados do *dataset*.

Inicialmente foram escolhidos estes 8 atributos para o treinamento do modelo. Contudo, verificou-se que o peso da variável *rain* era nulo para o modelo, e que *wind speed* tinha um peso considerado baixo para o modelo. Sendo assim, ambos foram retirados e foram utilizados os 6 atributos restantes como *features* para o treinamento do modelo final.

O objetivo do treinamento desses dados para os diferentes modelos de aprendizado de

máquina era de prever o atributo binário *queen presence*, referente a presença ou não da rainha dentro da colmeia. As *features* relacionadas a temperatura, umidade relativa do ar e pressão foram escolhidas na tentativa de oferecer a melhor predição possível, já que a presença da rainha ou não de uma rainha em uma colmeia de abelhas é afetada por essas variáveis. Para o treinamento foram escolhidos os modelos de classificação linear *Random Forest*, *K-Nearest Neighbors*, *Support Vector Machine* e *Gradient Boosting* para o trabalho. Os dados foram separados em 80% para treinamento e 20% para teste, e para cada modelo foram testados diferentes valores para os hiperparâmetros, com o intuito de obter a maior acurácia possível para maximizar a eficiência do modelo final. Para uma avaliação com maior robustez e generalização do modelo, o conjunto foi dividido em 5 *folds* para treino e teste em rodízio.

O treinamento foi realizado utilizando a linguagem *Python*, com auxílio das bibliotecas *matplotlib.pyplot*, *sklearn.model_selection* (para a importação das funções *train_test_split* e *cross_val_score*), *sklearn.metrics* (para importar as funções *accuracy_score*, *f1_score*, *recall_score* e *roc_auc_score*), *joblib* (responsável por salvar e carregar o modelo), *sklearn.model_selection* (para a importação do *cross_validate*), *sklearn.ensemble* (para a importação dos modelos de classificação linear *Random Forest Classifier* e *Gradient Boosting Classifier*), *sklearn.svm* (para importação do modelo *Support Vector Machine*) e *sklearn.neighbors* (para importação do modelo *K-Nearest Neighbors*). O treinamento dos modelos foi realizado baseando-se numa etapa sequencial de processos, assim como foi feito a mineração de dados, desta vez representado no diagrama da Figura 13.

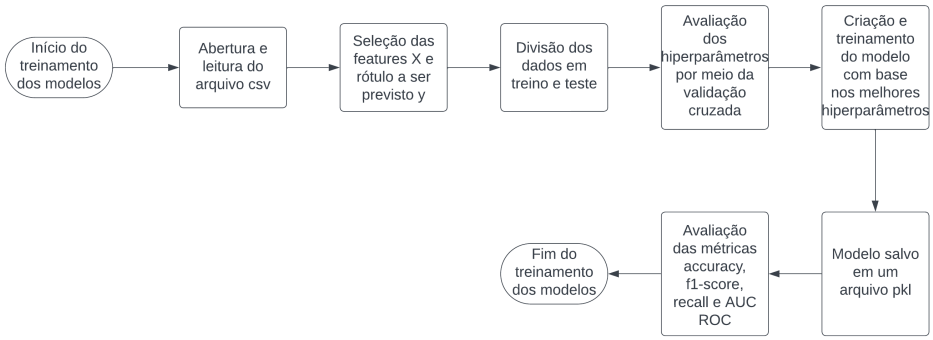

Figura 13. Diagrama representando o processo de treinamento dos modelos.

Os códigos de treinamento para os modelos são bem semelhantes, diferenciando apenas na parte da escolha dos hiperparâmetros para os modelos. Inicialmente são selecionadas as features *hive temp*, *hive humidity*, *hive pressure*, *weather temp*, *weather humidity* e *weather pressure* para a variável *X*, e *queen presence* que é o rótulo a ser previsto é definido para a variável *y*. Em seguida, é feita uma garantia para que os dados de *X* e *y* sejam ambos numéricos, e que além disso *y* contenha apenas valores binários. Posteriormente, os dados são divididos em 80% para treino e 20% para teste, definindo o parâmetro *random_state* como 42. Com a separação pronta, é realizada uma avaliação para decidir quais valores de hiperparâmetros seriam mais ideais para a realização do treinamento dos modelos, a partir da validação cruzada (*5-fold*). No modelo *Random Forest*, foram testados 30 *n_estimators*, variando-se de 20 em 20 no intervalo entre 10 a 600. O melhor resultado obtido foi para o valor 510. No modelo *K-Nearest Neighbors* foram testados 10 *k_estimators*, variando-se de 2 em 22 no intervalo entre 1 a 21, obtendo-se 3 como melhor valor para o estimador k. No modelo *Gradient Boosting*, o hiperparâmetro n foi testando

nos valores do intervalo entre 50 a 1000, variando-se de 50 em 50. Nele o melhor valor obtido foi 800. E por fim no *Support Vector Machine* foi um pouco complexo em se definir variáveis para treinar, pois o modelo estava se ajustando excessivamente aos dados de treino. Sendo assim, os hiperparâmetros *C* e *gamma* foram testados de forma totalmente manual, obtendo-se $C = 2$ e $gamma = 0.08$, utilizando um *kernel* não linear RBF (*radial basis function*). O hiperparâmetro *gamma* controla o *alcance de influência* de um único ponto do dado, o que afeta a forma da curva de decisão do modelo e principalmente o desempenho dele, caso esteja sendo usado um *kernel* não linear. Após a escolha dos melhores hiperparâmetros para os modelos, os modelos são criados e treinados.

A escolha do intervalo dos hiperparâmetros foi feita de forma manual para cada um dos algoritmos, considerando que o ponto de estabilização do desempenho pode variar conforme o valor de *n*, sendo específico para cada modelo.

RESULTADOS E DISCUSSÃO

O objetivo deste trabalho é de realizar uma limpeza e pré-processamento dos dados de um *dataset* com dados referentes a uma colmeia de abelhas e posteriormente aplicar técnicas de aprendizado de máquina para o treinamento de modelos e compará-los, avaliando desta forma o desempenho individual de cada um.

Na tabela 3 encontra-se os resultados dos melhores modelos obtidos de cada técnica aplicada usando a validação cruzada (*5-fold*).

Tabela 3. Comparação das métricas de desempenho entre os modelos usando o melhor hiperparâmetro obtido na validação cruzada (**5-fold**).

Modelo	**Acurácia (%)**	**F1-score (%)**	**Recall (%)**	**ROC AUC (%)**
Random Forest	91.76	95.47	99.10	90.02
K-Nearest Neighbors	90.49	94.68	96.53	80.43
Support Vector Machine	91.27	95.17	98.21	84.24
Gradient Boosting	91.86	95.47	97.98	89.50

O modelo *Random Forest Classifier* obteve excelentes resultados, possivelmente por conta de ser um modelo baseado em árvores de decisão, o que permite ter uma redução considerável em *overfitting* e generalizar bem os dados, dado que são treinadas múltiplas árvores em amostras diferentes de dados, o que reduz também o impacto negativo de *outliers* e ruídos. O modelo se destacou por seu alto *recall* (99.10%) e melhor ROC AUC (90.02%), o que indica que ele possui uma ótima capacidade na detecção de classes positivas (rainha presente) e uma robusta capacidade de generalização na separação de classes, mesmo em cenários em que exista desbalanceamento.

O modelo *Gradient Boosting* obteve resultados promissores; trata-se também de um modelo baseado em árvores de decisão, com a diferença de que ele é aprimorado a partir dos erros dos modelos fracos anteriores. Ele obteve o melhor desempenho da acurácia (91.86%), o que indica um excelente desempenho geral dentre os modelos.

O modelo *K-Nearest Neighbors* obteve o resultado inferior se comparado a todos, contudo ainda assim seu desempenho foi considerado bom. Ele funciona a partir da proximidade com os k-vizinhos. Sendo assim, aparentemente a distribuição dos dados teve padrões locais bem definidos, o que permitiu uma boa classificação por proximidade. Contudo, não foi bom o suficiente para atingir resultados mais promissores, tal como aconteceu com os modelos *Random Forest Classifier* e *Gradient Boosting Classifier*.

O modelo *Support Vector Machine* obteve bons resultados, embora tenha sido o mais problemático para treinar. Foram necessárias várias mudanças nos hiperparâmetro *C* e *gamma* do modelo para que o modelo conseguisse distinguir os dados e possuir um bom desempenho.

Na tabela 4 encontra-se os resultados dos melhores modelos obtidos de cada técnica aplicada usando apenas um conjunto de dados de teste, sem a utilização da validação cruzada.

Tabela 4. Comparação das métricas de desempenho entre os modelos usando o melhor hiperparâmetro usando dados de teste.

Modelo	**Acurácia (%)**	**F1-score (%)**	**Recall (%)**	**ROC AUC (%)**
Random Forest	92.16	95.71	100.00	87.04
K-Nearest Neighbors	91.37	95.18	97.31	76.26
Support Vector Machine	91.76	95.48	99.55	78.90
Gradient Boosting	92.94	96.09	99.10	88.45

O modelo *Random Forest Classifier* no treinamento sem a utilização da validação cruzada obteve uma acurácia de 92.16 % e um *recall* de 100 %, o que infere-se que todas as detecções de rainha presente na colmeia foram identificadas de forma correta pelo modelo.

O modelo *K-Nearest Neighbors* obteve resultados satisfatórios, embora tenham sido os mais inferiores no desempenho geral em comparação aos outros. Ele apresentou uma acurácia de 91.37 % e um *ROC AUC* de 76.26 %, indicando que o modelo pode ter limitações na diferenciação entre classes, apresentando uma baixa generalização do modelo em relação aos demais.

O modelo *Support Vector Machine* em termos gerais apresentou um bom desempenho, com acurácia de 91.76 %, *f1-score* de 95.48 % e *recall* de 99.55 %. Apenas seu *ROC AUC* que ficou um pouco abaixo da média geral, assim como apresentou o modelo *K- Nearest Neighbors*, indicando que ele possui uma maior dificuldade ao diferenciar as classes e generalizar o modelo.

O modelo *Gradient Boosting* obteve o melhor desempenho geral dentre os modelos no treinamento com os dados de teste, com uma acurácia de 92.94 %, *f1-score* de 96.09 %, *recall* de 99.10 % e um ROC AUC de 88.45 %. Sendo assim, ele obteve um equilíbrio entre precisão e a identificação correta de instâncias positivas do modelo, além de obter o melhor valor de *ROC AUC*, o que sugere-se uma excelente capacidade na separação entre classes.

Comparando-se os resultados das Tabelas 3 e 4, os resultados da acurácia, *f1-score* e *recall* obtiveram resultados melhores no conjunto de dados teste separado, com exceção da comparação *recall* no modelo *Support Vector Machine*. Este comportamento é esperado, pois em um cenário que existe apenas uma divisão única entre treino e testes, o modelo tende a sofrer um maior sobreajuste aos dados de treino, fazendo com que haja um desempenho superior. O desempenho de *ROC AUC* obteve os melhores resultados no teste da validação cruzada, o que também é coerente, já que com uma maior quantidade de subconjuntos de dados espera-se que exista uma melhor generalização do modelo, ainda em cenários com classes desbalanceadas. De forma geral, o *cross-validation 5 folds* é um método mais confiável, já que é realizado um treinamento mais robusto com os dados, o que gera uma menor variância dos resultados uma avaliação mais precisa do desempenho real do modelo.

CONCLUSÕES

Este trabalho investigou os algoritmos de aprendizado de máquina, comparando o desempenho obtido pelos modelos treinados usando um *dataset* com dados referentes a uma colmeia de abelhas. Os algoritmos *Random Forest Classifier* e *Gradient Boosting Classifier* se destacaram

por terem obtido os melhores resultados na comparação. Os algoritmos *K-Nearest Neighbors* e *Support Vector Machine* que, apesar de apresentarem um desempenho inferior se comparado aos outros dois, também tiveram bons resultados.

Para trabalhos futuros, poderá ser feito um trabalho com modelos de regressão linear. Durante este trabalho foram realizados treinamentos com modelos de regressão linear, com o intuito de prever *features* faltantes de outros *datasets*, tais como *hive_pressure* e *hive_humidity*, criando-se desta forma dados sintéticos. Também foi realizada uma análise em relação aos pesos que foram atribuídos por cada modelo em seus atributos utilizados no treinamento. Contudo, o trabalho ficaria excessivamente grande para explicar toda a metodologia e técnicas usadas para o modelo de regressão linear e a análise do peso das *features*. Sendo assim, a codificação e resultados desta parte podem ser encontrados no repositório criado para o projeto, disponível na plataforma GitHub [1].

Além disso, poderá ser feita uma coleta de dados mais específicos de diferentes colmeias, com o intuito de explorar mais possibilidades no treinamento de modelos. Não existem muitos *datasets* públicos disponíveis com relação ao tema de colmeia de abelhas na literatura, o que de fato dificultou na criação de diferentes modelos de predição, restringindo-se na predição do atributo binário '*queen presence*. A criação de dados sintéticos, mencionada anteriormente, poderia abrir novas ideias para outros treinamentos, como a presença de um vírus, parasita ou bactéria presente dentro da colmeia, determinar a espécie das abelhas, ou ainda prever variáveis como o peso da colmeia, que é afetado diretamente pela quantidade de abelhas e de mel e pólen armazenados internamente.

REFERÊNCIAS

[1] FARIA, M. B. *Data-Mining-Bee-Colony*. [S.l.: s.n.], 2025. https://github.com/matheusbuenowb/Data-Mining-Bee-Colony. Acessado em: 4 jun. 2025.

[2] SINGH, S.; KUMAR, Y.; ALKHAYYAT, A.; SINGH, D.; ANAND, R.; CHOHAN, J. S. Real-Time Beehive Condition Improving Frequency Selection and Bee Welfare Monitoring and Analysis. In: 2024 IEEE Wireless Antenna and Microwave Symposium (WAMS). [S.l.: s.n.], 2024. P. 1–6. DOI: 10.1109/WAMS59642.2024.10527800.

[3] SENGER, D.; GRUBER, C.; KLUSS, T.; JOHANNSEN, C. Weight, temperature and humidity sensor data of honey bee colonies in Germany, 2019–2022. *Data in Brief*, v. 52, p. 110015, 2024. ISSN 2352-3409. DOI: https://doi.org/10.1016/j.dib.2023.110015. Disponível em: .

[4] CHEN, S.-H.; WANG, J.-C.; LIN, H.-J.; LEE, M.-H.; LIU, A.-C.; WU, Y.-L.; HSU, P.-S.; YANG, E.-C.; JIANG, J.-A. A machine learning-based multiclass classification model for bee colony anomaly identification using an IoT-based audio monitoring system with an edge computing framework. *Expert Systems with Applications*, v. 255, p. 124898, 2024. ISSN 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2024.124898. Disponível em: .

[5] JAILIS, B. A.; KIRING, A.; YEW, H. T.; BARUKANG, L.; FARM, Y. Y.; WONG, F. A Real-Time Web-Based Monitoring System for Stingless Bee Farming. In: 2022 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET). [S.l.: s.n.], 2022. P. 1–5. DOI: 10.1109/IICAIET55139.2022.9936841.

[6] VERMA, R.; RAINA, R.; GARG, V.; GAUTAM, S.; KUMAR, S. An IoT-Based Biodiversity Monitoring System. In: 2024 IEEE International Conference on Omni-layer Intelligent Systems (COINS). [S.l.: s.n.], 2024. P. 1–6. DOI: 10.1109/COINS61597.2024.10622521.

[7] HADJUR, H.; AMMAR, D.; LEFEVRE, L. Services Orchestration at the Edge and in the Cloud for Energy-Aware Precision Beekeeping Systems. In: 2023 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW). [S.l.: s.n.], 2023. P. 769–776. DOI: 10.1109/IPDPSW59300.2023.00129.

[8] ALLERI, M.; AMOROSO, S.; CATANIA, P.; LO VERDE, G.; ORLANDO, S.; RAGUSA, E.; SINACORI, M.; VALLONE, M.; VELLA, A. Recent developments on precision beekeeping: A systematic literature review. *Journal of Agriculture and Food Research*, v. 14, p. 100726, 2023. ISSN 2666-1543. DOI: https://doi.org/10.1016/j.jafr.2023.100726. Disponível em: .

[9] YANG, A. J. *Smart Bee Colony Monitor: Clips of Beehive Sounds*. [S.l.: s.n.], 5 nov. 2022. The GNOME Project. Disponível em: . Acesso em: 23 mai. 2025.