UNIVERSIDADE DE CAXIAS DO SUL
ÁREA DO CONHECIMENTO DE CIÊNCIAS EXATAS E ENGENHARIAS

LUCAS MASSIGNANI COELHO DA SILVA

INTERFACE PREDITIVA APLICADA À ÁREA DA SAÚDE

CAXIAS DO SUL

2022

LUCAS MASSIGNANI COELHO DA SILVA

INTERFACE PREDITIVA APLICADA À ÁREA DA SAÚDE

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Ciência da Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul.

Orientador: Profa. Dra. Carine G. Webber

CAXIAS DO SUL

2022

LUCAS MASSIGNANI COELHO DA SILVA

INTERFACE PREDITIVA APLICADA À ÁREA DA SAÚDE

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Ciência da Computação na Área do Conhecimento de Ciências Exatas e Engenharias da Universidade de Caxias do Sul.

Aprovado(a) em 00/07/2021

BANCA EXAMINADORA

Profa. Dra. Carine G. Webber
Universidade de Caxias do Sul - UCS

Profa. Dra. Helena Grazziotin Ribeiro
Universidade de Caxias do Sul - UCS

Profa. Dra. Elisa Boff
Universidade de Caxias do Sul - UCS

Este trabalho é dedicado aos meus pais por tornarem esse sonho possível e a minha mulher por todo seu apoio.

AGRADECIMENTOS

Agradeço aos meus pais Alaor Coelho da Silva e Ana Maria Massignani, que não mediram esforços para me ajudar nessa etapa tão importante da minha vida.

A minha noiva Laíz Cristina Follmann de Fraga que sempre me ajudou e me fortaleceu nos momentos difíceis.

Agradeço a professora Carine Webber, responsável pela orientação deste trabalho.

"Há apenas uma maneira de evitar críticas: não falar, não fazer e não ser nada."

Aristóteles

RESUMO

As doenças cardiovasculares (DCV) são as principais causas de morte em todo o mundo. Neste contexto, ferramentas de triagem automáticas podem auxiliar a identificar essas doenças. Como exemplo, o eletrocardiograma (ECG) é um dos principais métodos de triagem, por ser eficiente e não invasivo. Por conta dessas características, ele é amplamente utilizado para identificar DCVs. O infarto do miocárdio (IM) ou ataque cardíaco é uma DCV que ocorre devido ao bloqueio parcial ou completo do fluxo sanguíneo para os músculos cardíacos. Ele pode levar a danos irreversíveis ao coração, ou até mesmo a morte, se não for identificado precocemente. Neste sentido, existe um conceito chamado *golden hour*, ou seja, a hora de ouro, o que significa que o restabelecimento da circulação sanguínea deve ser feito o quanto antes. Pode-se evitar, assim, a morte do músculo cardíaco, reduzindo a taxa de mortalidade. Nesse contexto, o objetivo desse trabalho consiste em, a partir dos sinais vindos de um ECG, empregar métodos de Aprendizado de Máquina para fins de predição de infarto e implementar uma interface de programação de aplicações (API), para disponibilizar o acesso ao modelo de aprendizado pelos dispositivos capazes de enviar sinais. Para iniciar, esse projeto partiu de um processo de revisão sistemática da literatura, a fim de mapear o estado da arte na área. Foram selecionados trabalhos relacionados ao estudo, aplicando-se filtros e refinamentos sucessivos. A partir da análise desses trabalhos, foi identificado o melhor método de classificação para dar continuidade ao estudo. Além disso, foi identificado um padrão entre os trabalhos analisados, sendo esse a utilização do dataset PTB, ou PTB-XL, esteve presente na maioria dos projetos. Selecionou-se para este projeto o dataset PTB-XL por conter um número maior de instâncias. Por fim, foram utilizados os algorítimos *Random Forest* e Árvore de Decisão para fazer a implementação dos modelos, além da criação de uma Interface de Programação de Aplicação (API) e sua disponibilização via Heroku.

Palavras-chave: Aprendizado de Máquina. Eletrocardiograma. Infarto do Miocárdio. Ataque Cardíaco.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ALGORITMOS

LISTA DE ABREVIATURAS E SIGLAS

IA.	Inteligência artificial
OMS.	Organização Mundial da Saúde
ECG.	Eletrocardiograma
API.	Interface de programação de aplicações
DCV.	Doença cardiovascular
IM.	Infarto do miocárdio
AD.	Árvore de Decisão
VP.	Verdadeiro positivo
VN.	Verdadeiro negativo
FP.	Falso positivo
FN.	Falso negativo
MPCA.	Análise de componentes principais multilinear
DT.	Árvore de decisão
KNN.	*k-nearest neighbors*
CNN.	*Convolutional neural network*
SVM.	*Support vector machine*
DWPT.	*Discrete wavelet packet transform*

SUMÁRIO

1 INTRODUÇÃO

A inteligência exibida por máquinas é chamada de inteligência artificial (IA). Dessa forma, na ciência da computação, um dispositivo que perceba seu ambiente e seja capaz de realizar ações que maximizam sua chance de sucesso em algum objetivo, pode ser classificado como um "agente inteligente"(ONGSULEE, 2017). Além disso, os objetivos da pesquisa em IA incluem raciocínio, conhecimento, planejamento, aprendizado, processamento de linguagem natural, percepção e a capacidade de mover e manipular objetos (LUGER, 2013). Partindo da IA limitada (figura 1), cujos aplicativos são direcionadas para uma única área e função, a inteligência geral está entre os objetivos de longo prazo para este campo. Ela apresentará sistemas com capacidade humana de resolução de problemas. Mais adiante, a IA superior irá superar a capacidade humana. Ademais, as abordagens utilizadas para a conquista destes objetivos são: métodos estatísticos, inteligência computacional e IA simbólica (ONGSULEE, 2017).

Figura 1 – Níveis para avaliar produtos de Inteligência Artificial

Estágio	IA Limitada	IA Geral	Super IA
Momento	Hoje	2040	2050
Implicações	Capaz de executar tarefas como conduzir um veículo, diagnosticar algumas doenças e recomendação de itens.	Toma decisões em áreas ambientais, econômicas, sociais e tecnológicas necessárias para a sobrevivência humana na terra.	Supera as habilidades humanas em lidar com os problemas da sociedade e do planeta para manutenção da vida.

Fonte: O Autor

O Aprendizado de Máquina é uma área da inteligência artificial, e ela dá a capacidade ao computador de aprender sobre um domínio sem explicitamente ser programado (ONGSULEE, 2017). Dessa forma, um sistema de aprendizado pode ser entendido como um software capaz de tomar decisões utilizando como base a experiência acumulada de outras soluções bem sucedidas de problemas anteriores. Ademais, mesmo que o Aprendizado de Máquina seja um recurso poderoso para a obtenção automática de conhecimento, não existe somente um algoritmo para resolver todos os problemas. Sendo assim, é crucial analisar cada situação para escolher o melhor algoritmo para cada caso específico. Além disso, os métodos mais amplamente utilizados de Aprendizado de Máquina são os supervisionados (dados previamente classificados) e os não-

supervisionados (dados não rotulados). A maior parte do aprendizado é do tipo supervisionado sendo cerca de 70%, em seguida o não-supervisionado que possui entre 10 a 20 por cento e por fim, o restante são aprendizado semi-supervisionado e por reforço (ONGSULEE, 2017).

As doenças cardiovasculares (DCV) são um dos maiores contribuintes para o número de mortes relatadas globalmente. Estima-se que até o ano de 2030 cerca de 44% da população dos Estados Unidos tenham pelo menos um tipo de DCV. Além disso, estatísticas disponibilizadas pela Organização Mundial da Saúde (OMS) mostram que 17,9 milhões de pessoas morrem todos os anos devido a DCVs (JAHMUNAH *et al.*, 2021). Assim, o infarto é uma condição cardíaca que surge devido ao bloqueio parcial ou completo do fluxo sanguíneo para o coração. Sua causa primária é a doença arterial coronariana que se resume no acúmulo de gordura (placa) nas artérias coronarianas, dessa forma reduzindo o fluxo sanguíneo para o coração. Seus fatores de risco incluem obesidade, diabetes, falta de exercício, colesterol alto, estresse, tabagismo, consumo excessivo de álcool e má alimentação (FATIMAH *et al.*, 2021).

Ademais, o infarto do miocárdio pode levar a danos irreversíveis no coração ou mesmo à morte, seus sintomas incluem dor no peito, dificuldade para respirar e batimentos cardíacos irregulares, é fundamental rastrear e tratar a doença em estágio inicial (FATIMAH *et al.*, 2021). A primeira hora de um ataque cardíaco, também conhecida como hora de ouro, é a mais importante, pois nessa hora se os médicos identificarem o infarto os índices de mortalidade são reduzidos. Por isso, é tão importante sua detecção antecipada (XIONG *et al.*, 2021).

Dentre essas técnicas para se detectar ataques cardíacos, o Eletrocardiograma (ECG) se destaca por ser um método não invasivo, econômico e amplamente utilizado na detecção de IM (LIU *et al.*, 2020). Além disso, ele também pode mostrar os locais do coração que o infarto está acontecendo. Existem vários tipos de ECG e com um número variável de derivações. Sendo que os modelos de uma derivação podem ser gerados a partir de alguns aparelhos como *smartwatches* ou *smartphones* que possuam os sensores necessários.

1.1 OBJETIVO GERAL E ESPECÍFICOS

O objetivo geral desse trabalho consiste em, a partir dos sinais vindos de um ECG, empregar métodos de Aprendizado de Máquina para fins de predição de infarto e implementar uma interface de programação de aplicações (API), para disponibilizar o acesso ao modelo de aprendizado pelos dispositivos capazes de enviar sinais.

Para atingir o objetivo da proposta tem-se alguns objetivos específicos:

- Identificar os métodos computacionais utilizados para detecção de ataques cardíacos e selecionar o mais apropriado;
- Identificar o *dataset* mais utilizado para a classificação de ataques cardíacos;

- Implementar, treinar e testar os métodos para fins de detecção de infartos;
- Implementar uma API que consiga receber sinais de um ECG e retornar uma classificação de infarto ou não.

1.2 ORGANIZAÇÃO DO DOCUMENTO

O presente trabalho está organizado da seguinte forma:

- No Capítulo 2 é feita a fundamentação teórica do trabalho caracterizando tópicos importantes na detecção de infartos. São detalhados os conceitos de Aprendizado de Máquina, algoritmos e métricas de avaliação de modelos. Por fim, é conduzida uma revisão sistemática da literatura.
- O Capítulo 3 contém o processo de desenvolvimento da aplicação, apresentando a abordagem escolhida para a condução do projeto, as etapas e as tecnologias que foram utilizadas.
- No Capítulo 4 é feita uma síntese do trabalho, as contribuições e os trabalhos futuros.

2 APRENDIZADO DE MÁQUINA

Nesse capítulo apresenta-se o embasamento teórico do trabalho. Em primeiro lugar, se observa o Aprendizado de Máquina. Após, são apresentados os algoritmos de aprendizagem, Árvore de decisão e *Random Forest*, cuja relevância foi percebida na etapa de revisão. Em seguida, são elencadas as principais métricas de avaliação. Por fim, é feita uma revisão sistemática da literatura.

2.1 O QUE É APRENDIZADO DE MÁQUINA

O Aprendizado de Máquina produz generalizações a partir de um conjunto particular de exemplos (MONARD; BARANAUSKAS, 2003). O processo de aprendizado ocorre por meio de um algoritmo indutivo, que a partir de dados constrói um modelo geral. A indução (processo de aprendizado) é criada a partir de um conceito específico que depois é generalizado. Dessa forma, por meio de inferências indutivas sobre exemplos apresentados, as hipóteses geradas podem ou não preservar a verdade.

Apesar da indução ser um recurso amplamente usado, ela deve ser utilizada com cautela, devendo sempre tomar alguns cuidados. Portanto, se a quantidade de dados for insuficiente ou se os exemplos não forem bem escolhidos, as hipóteses obtidas podem ter um valor abaixo do esperado ou até mesmo não terem um valor real (MONARD; BARANAUSKAS, 2003).

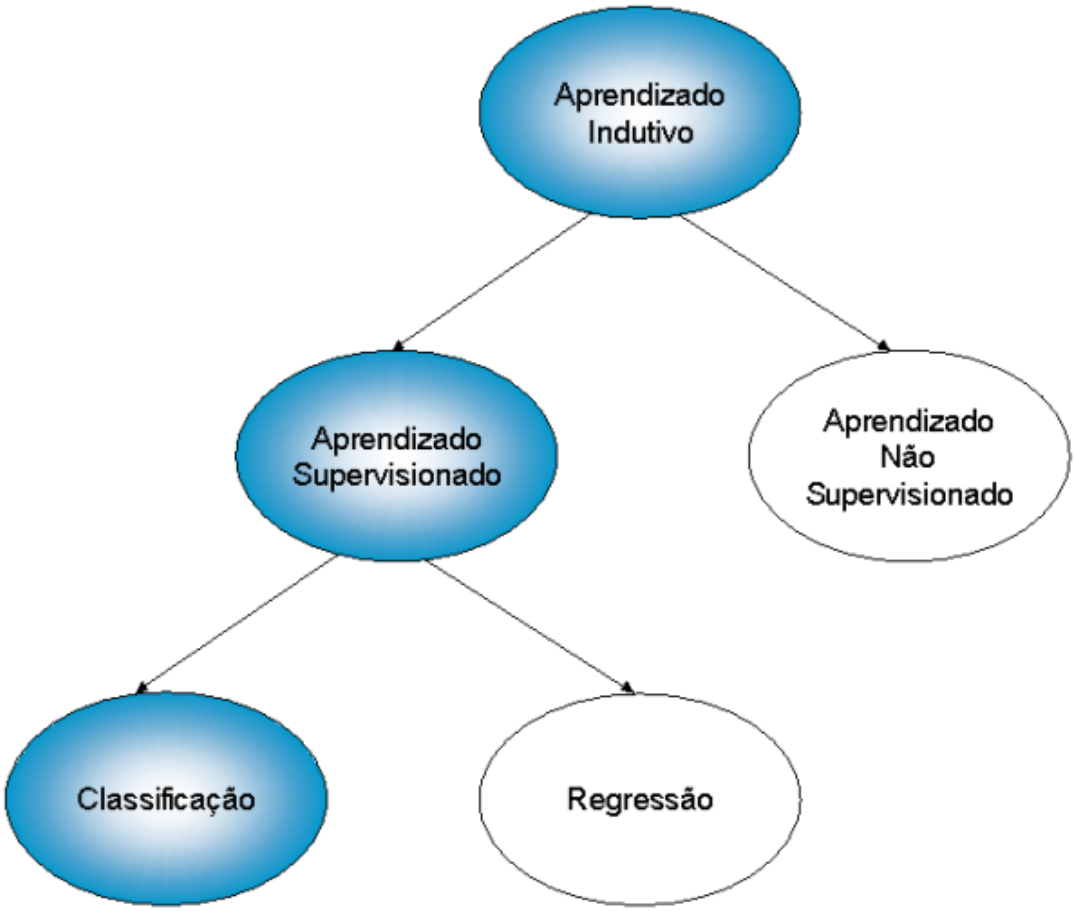
Além disso, o aprendizado indutivo possui uma hierarquia, como pode ser visto na figura 2 e ela é dividida em duas principais categorias: supervisionado e não-supervisionado. A principal diferença é que no aprendizado supervisionado é fornecido um conjunto de exemplos de treinamento, no qual o rótulo das classes já é conhecido (MONARD; BARANAUSKAS, 2003).

No aprendizado supervisionado, os exemplos, denominados instâncias, são armazenados em *datasets* compostos por uma matriz de valores de características e um rótulo ou classe associada (MONARD; BARANAUSKAS, 2003). A tarefa do algoritmo indutivo é a construção de um classificador que tenha a capacidade de determinar corretamente a classe de novos exemplos, sem antes serem rotulados. Os problemas com classe discreta são chamados de classificação e os de valores contínuos de regressão.

No aprendizado não-supervisionado, não se conhece previamente os rótulos dos dados (MONARD; BARANAUSKAS, 2003). Nesse modelo, os algoritmos tentam analisar os exemplos e agrupá-los por similaridade ou proximidade espacial, formando agrupamentos ou *clusters*. Por fim, normalmente é necessária uma análise dos grupos formados para determinar seu significado dentro do contexto do problema analisado.

Aprendizado de Máquina apesar de ainda ser uma área desafiadora, possui uma impor-

Figura 2 – Hierarquia do aprendizado indutivo



Fonte: Monard e Baranauskas (2003)

tância intangível (LUGER, 2013). Ela possibilita a criação de sistemas baseados em conhecimento que podem executar cálculos extensivos e custosos para resolver um problema de classificação, por exemplo. Por outro lado, uma de suas limitações, comparada com seres humanos, é de que ao se deparar com um problema igual ou semelhante ele acaba tendo que realizar todos os cálculos novamente, pois registra soluções anteriores. A solução mais aparente para o problema é deixar com que o sistema aprenda sozinho.

2.2 PARADIGMAS DO APRENDIZADO DE MÁQUINA

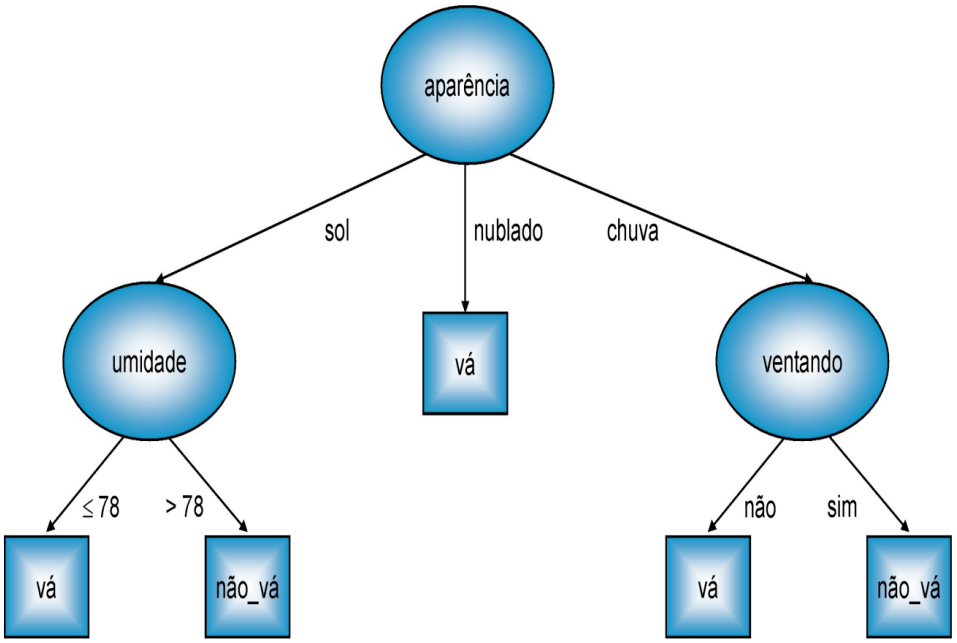
Os modelos mais relevantes, segundo Luger (2013), são os seguintes: simbólico, probabilístico, conexionista e genético.

2.2.1 Simbólico

Os sistemas de aprendizado simbólico utilizam-se de análises e contra-análises de exemplos para aprender como construir representações simbólicas de um conceito. Geralmente as representações simbólicas estão como forma de uma expressão lógica, árvores de decisão ou rede semântica (MONARD; BARANAUSKAS, 2003). Além disso, principalmente os sistemas simbólicos são utilizados para situações onde o que foi aprendido pelo modelo possa ser interpretado por humanos (PRATI JOSé AUGUSTO BARANAUSKAS, 2001).

Um dos métodos simbólicos mais conhecidos é a Árvore de Decisão (AD). A figura 3 representa uma AD, os círculos representam os nós de decisão, as setas são os possíveis resultados e os quadrados são os nós folhas. Uma AD pode ser utilizada para classificação de exemplos novos, começando a partir da raiz da árvore e descendo pelos nós de decisão até encontrar um nó folha (PRATI; BARANAUSKAS; MONARD, 2002).

Figura 3 – Representação Gráfica da Árvore de Decisão



Fonte: Prati, Baranauskas e Monard (2002)

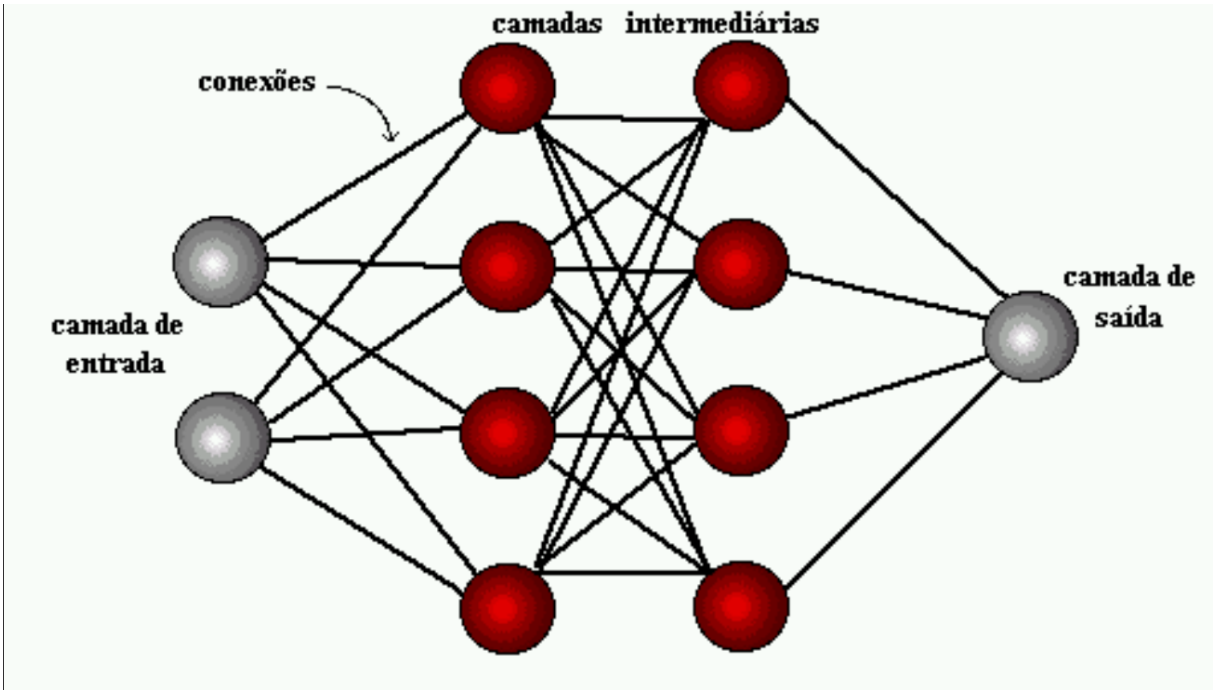
2.2.2 Conexionista

Também conhecido como Processamento paralelo distribuído, os sistemas conexionistas utilizam-se do significado dos símbolos para resolver problemas (LUGER, 2013). Ademais, as Redes Neurais são uma forma inspirada e simplificada da construção matemática do modelo biológico do sistema nervoso. A representação da rede neural envolve unidades altamente interconectadas e devido a este motivo que se dá o nome conexionismo para descrever esta área de

estudo. Além disso, sua semelhança com a biologia tem levado muitos pesquisadores a acreditar que Redes Neurais possam ter uma ampla capacidade de resolver problemas que requerem intenso processamento sensorial humano, como visão e reconhecimento de voz (MONARD; BARANAUSKAS, 2003).

A maior parte dos modelos de redes neurais possui uma regra de treinamento, na qual os pesos das suas conexões são alterados de acordo com os padrões aprendidos através de exemplos (CARVALHO, 2001). Na figura 4 é possível ver um exemplo de rede neural a qual é dividida em três grupos. O primeiro grupo, a camada de entrada, na qual os padrões são apresentados a rede, o segundo, camadas intermediárias ou escondidas, na qual o processamento é feito através de conexões que fazem a extração das características e por fim o terceiro grupo, a camada de saída a qual conclui e apresenta o resultado final (CARVALHO, 2001).

Figura 4 – Camadas da rede neural



Fonte: Carvalho (2001)

2.2.3 Genético

Um classificador genético se resume a uma população de elementos classificatórios que disputam para fazer a predição. Os componentes que possuem um desempenho fraco são descartados, enquanto os com desempenho forte se proliferam, e produzem variações similares. Todavia, este paradigma se faz jus a Teoria Darwiniana, na qual sobrevivem os mais bem adaptados ao meio (MONARD; BARANAUSKAS, 2003).

Os algoritmos genéticos atuam sobre uma população de indivíduos e ele se baseia no fato que indivíduos com boas características genéticas têm uma probabilidade maior de sobre-

viver e por consequência gerarem indivíduos cada vez mais aptos, por outro lado, os indivíduos menos aptos tendem a desaparecer, cada indivíduo da população é chamado de cromossomo e ele corresponde a uma possível solução de um dado problema. Além disso, é aplicado sobre a população um mecanismo de reprodução, baseado no processo evolutivo, com o objetivo de explorar o espaço de busca e encontrar a melhor solução para um dado problema (SOUTO *et al.*, 2003).

O funcionamento de um algoritmo genético pode ser visto na figura 5, iniciando com a população inicial é aplicada a função de avaliação (*fitness*), essa função é responsável por apresentar os requisitos para uma população se adaptar e assim avançar para a próxima geração. Em seguida é aplicado um método de seleção, esse é o responsável pela escolha dos indivíduos que formarão os descendentes para a próxima geração. Agora, com os indivíduos selecionados, dois operadores genéticos são aplicados: cruzamento e mutação. O cruzamento, como o nome sugere, combina dois indivíduos (pais) para gerar indivíduos descendentes (filhos). Por outro lado, a mutação acontece alterando de forma aleatória alguma característica genética de alguns indivíduos pais, selecionados por um critério probabilístico. Após a aplicação desses métodos é obtida a nova geração que logo é avaliada pelo critério de parada, esse critério pode ser definido quando se encontra um padrão ótimo em relação aos indivíduos da população, não exigindo mais a necessidade de execução do algoritmo ou quando se percebe que não é possível identificar um padrão ótimo para os indivíduos. Por fim, se a condição de parada for atingida é retornada a melhor solução encontrada para o problema. Entretanto, se não for atingida o processo retorna para a etapa inicial de *fitness*, agora utilizando os novos conhecimentos adquiridos pela população anterior (KATO; PAIVA; IZIDORO, 2021).

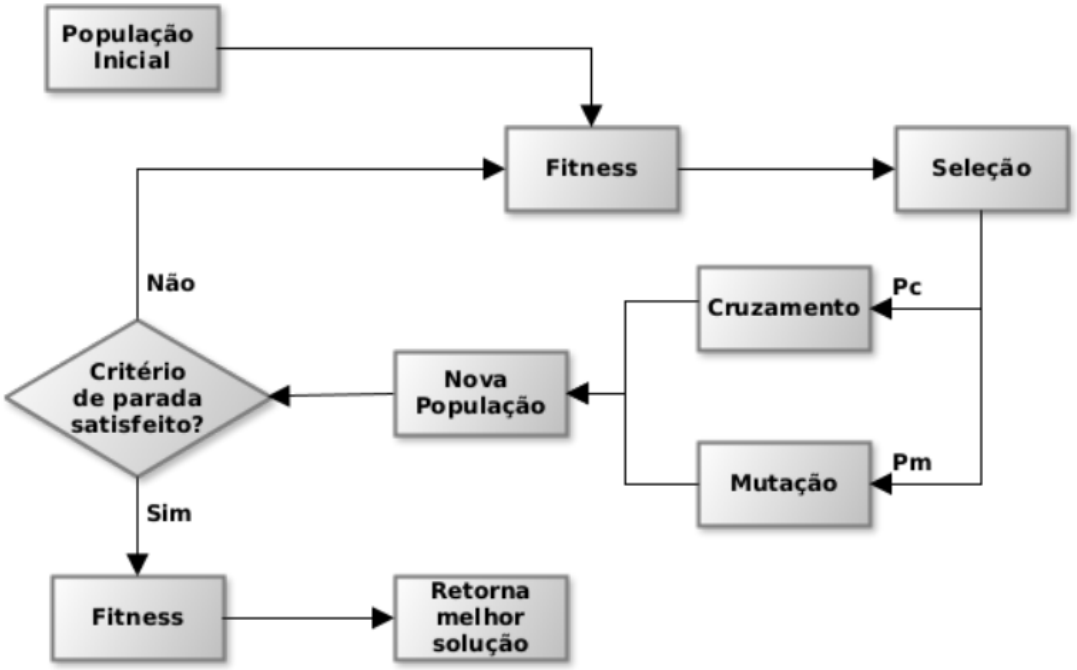
2.2.4 Probabilístico

A comunidade de Aprendizado de Máquina tem desenvolvido diversos métodos de classificação, muitos deles semelhantes aos métodos criados por pesquisadores. O conceito geral corresponde a utilizar modelos estatísticos que consigam ter uma boa aproximação do conceito induzido. Esses métodos são paramétricos e assumem alguma forma de modelo utilizando-se dos dados para obter parâmetros do modelo e assim encontrar valores apropriados (MONARD; BARANAUSKAS, 2003).

Assim, por exemplo, um classificador linear entende que as classes podem se apresentar como combinação linear dos valores dos atributos, portanto busca uma combinação linear que se aproxima da melhor maneira sobre o conjunto de dados. Dentre os principais métodos estatísticos se salienta o aprendizado Bayesiano, que se emprega do modelo probabilístico baseado no conhecimento preliminar do problema, o qual determina a probabilidade final de uma hipótese combinando os exemplos de treinamento (MONARD; BARANAUSKAS, 2003).

Dessa forma, as vantagens dos métodos estáticos dentre eles o Bayesiano são: a possibilidade de encaixar nas probabilidades calculadas o conhecimento de domínio caso o tenha e

Figura 5 – Funcionamento de um Algoritmos Genético padrão.



Fonte: Kato, Paiva e Izidoro (2021)

a competência das classificações feitas pelo algoritmo de máquina se basearem em evidências fornecidas que podem aumentar ou diminuir as probabilidades dos métodos a serem vistos em uma nova instância. Portanto, as desvantagens do método estático são o alto custo computacional devido às muitas probabilidades que devem ser calculadas (MONARD; BARANAUSKAS, 2003). Ademais, um exemplo de classificador Bayesiano é o *Naive Bayes*, ele assume que o efeito de um valor de atributo em uma determinada classe é independente dos valores de outros atributos, esse comportamento é denominado de independência condicional de classe. Sendo assim, essa medida é tomada para simplificar a computação envolvida, e por conta disso o nome desse método possui a palavra "*Naive*"(Ingênuo) (LEUNG, 2007).

2.3 ALGORITMOS

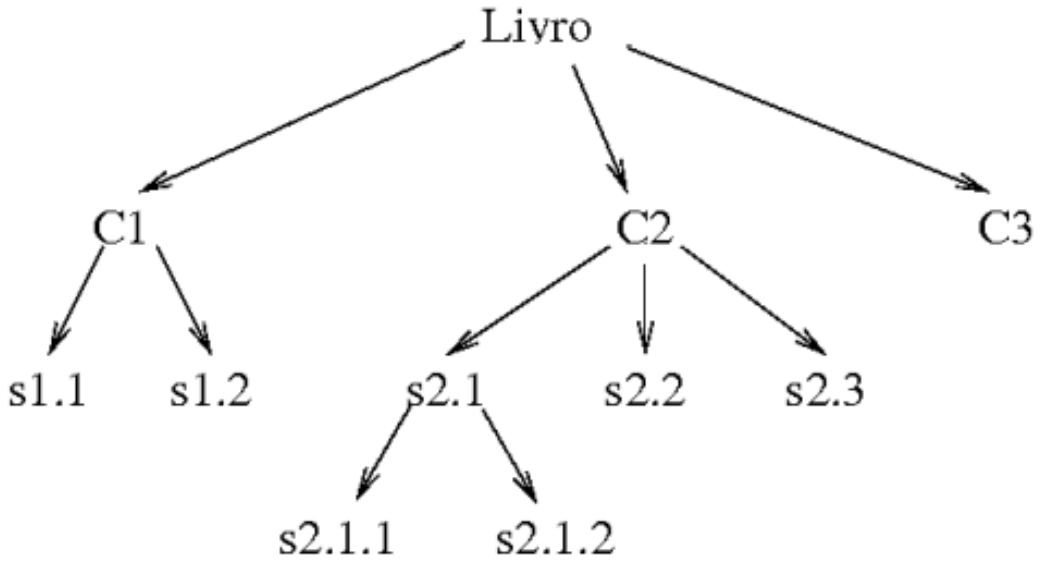
Essa seção trata de algoritmos de Aprendizado de Máquina. Por meio do estudo realizado na revisão sistemática, diversos métodos foram reconhecidos e um deles, apresentou o melhor resultado de acurácia: a Árvore de Decisão. As seções seguintes descrevem o funcionamento deste método e apresentam o método *Random Forest* que utiliza várias Árvores de Decisão para fazer sua predição.

2.3.1 Árvore de decisão

Uma árvore de decisão se baseia no conceito computacional de árvore. Pode-se definir uma árvore como sendo um conjunto de elementos chamados de nós, sendo que um deles é chamado de raiz que define o início ou ponto de partida da árvore. Esta raiz cria uma relação de paternidade com os outros nós criando, dessa forma, uma hierarquia (LAURETTO, 2010).

Um exemplo claro que se pode usar para descrever a árvore é o sumário de um livro, o qual possui diversos nós, são eles: os capítulos e os sub-capítulos. Como visualizado na figura 6, a raiz da árvore é o nó Livro, que possui outros nós como filho (C1, C2, C3), que por sua vez podem ou não ter mais nós como filhos. Além disso, os nós que não possuem filhos são denominados nós terminais ou folhas, um exemplo deste nó é o C3. Em contrapartida os que possuem filhos são chamados de nós não-terminais ou nós internos, como exemplo: o nó C2 (LAURETTO, 2010).

Figura 6 – Exemplo de uma árvore



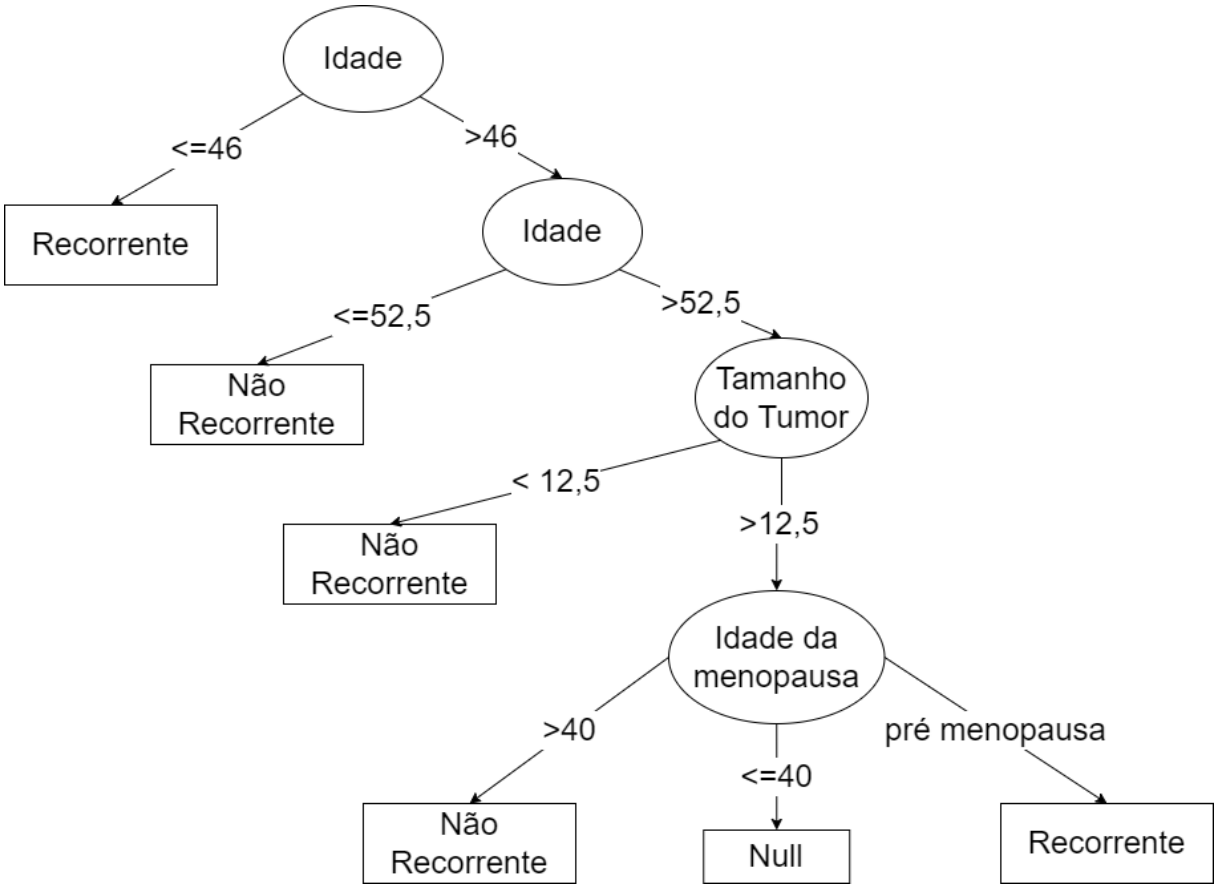
Fonte: Adaptado de Lauretto (2010)

Então, define-se uma árvore de decisão (AD) como sendo composta de nós terminais que contêm o nome de uma classe ou um símbolo nulo. Assim, entende-se que o símbolo nulo indica a incapacidade da AD de atribuir uma classe válida para aquele exemplo. Ela também é formada pelos nós não terminais, que contêm o nome de um atributo, no qual cada possível valor desse atributo corresponderá a um ramo para uma outra árvore de decisão (LAURETTO, 2010).

Além disso, a estratégia utilizada pela AD para classificação é denominada dividir para conquistar, onde um problema complexo é dividido em subproblemas mais simples e recursivamente é aplicada a mesma estratégia de resolução (FACELI *et al.*, 2021). A solução desses subproblemas combinados geram uma árvore que, por fim, resolve o problema complexo.

A forma mais comum de árvore de decisão possui uma estrutura característica (LAURETTO, 2010). Ademais, os nós internos são rotulados com atributos, as folhas rotuladas com classes e os ramos são rotulados com valores específicos ou intervalos. Na figura 7 se analisa uma AD para determinar a recorrência de câncer de mama. Assim, a idade é o nó raiz, e seus ramos levam a diferentes conclusões. Por exemplo, se a idade for menor ou igual a 46 chega-se a uma folha que fará a classificação como recorrente. Porém, se a idade for maior que 46 o ramo levará para um nó interno e a classificação continuará daquele ponto.

Figura 7 – Predição de recorrência de câncer de mama



Fonte: Adaptado de Lauretto (2010)

A criação de um algoritmo que gera as árvores de decisão pode ser feita de forma simples (FACELI *et al.*, 2021). Os principais passos para a criação da árvore de decisão estão no algoritmo 1. A função principal, GeraArvore, recebe como entrada um conjunto de dados **D**. Na primeira condição o algoritmo faz a validação do critério de parada. Caso o critério de parada não seja verdadeiro, mais divisões são necessárias, então é escolhido o atributo que maximiza algumas medidas de impureza. Por fim, a função GeraArvore é chamada recursivamente para

cada partição do conjunto de dados **D**.

Algoritmo 1: Gera árvore de decisão

Entrada: Um conjunto de treinamentos **D**
Saída: Árvore de Decisão
início
  /* Função GeraArvore(D) */
  se *CriterioDeParada(D) = Verdadeiro* **então**
    retorna *um nó folha rotulado com a constante que minimiza a função perda;*;
  fim
  Escolha o atributo que maximiza o critério de divisão **D**;
  para *para cada partição dos exemplos **Di** baseado nos valores do atributo escolhido*
  faça
    Induz uma subárvore Árvore = GeraArvore(**Di**)
  fim
  retorna *Árvore contendo um nó de decisão baseado no atributo escolhido*;
fim

Fonte: Adaptado de Faceli *et al.* (2021)

2.3.2 *Random Forest*

Random Forest é um grupo de árvores de decisão, que são construídas através de um subconjunto aleatório do conjunto de treinamento. Esse grupo de árvores formam um classificador agregado que pode ser usado para predição de novos objetos por meio de um sistema de votação (LAURETTO, 2010).

Dessa forma, cada árvore do *Random Forest* é construída utilizando uma amostra *bootstrap* dos dados originais fazendo com que se use cerca de dois terços dos exemplos originais, estes são utilizados na construção da árvore. Assim, o conjunto não utilizado para treinamento é usado como conjunto de testes.

Ainda, o *bootstrap* é um método de estatística de propósito geral (LAURETTO, 2010). A ideia por trás dele é que ao receber uma amostra de tamanho N, são selecionadas as chamadas amostras *bootstrap*, elas nada mais são do que amostras sorteadas de forma aleatória do conjunto original e para cada uma dessas amostras *bootstrap* é feita uma estimativa dos parâmetros de interesse.

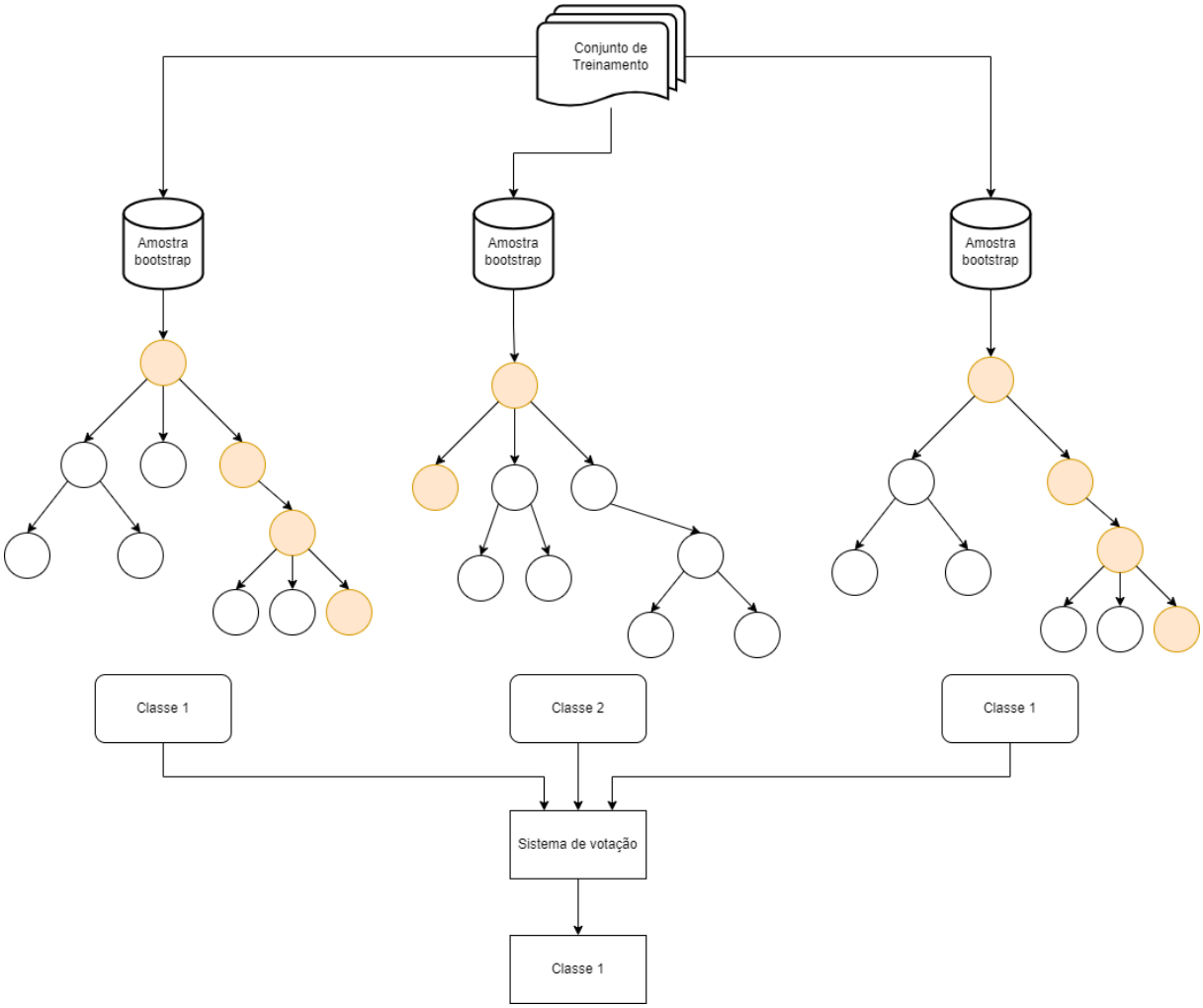
Portanto, a construção de uma *Random Forest* pode ser definida nesses principais passos:

- Os dados são divididos em conjunto de teste e treinamento, normalmente sendo 2/3 para treinamento e 1/3 para teste.

- Utilizando o conjunto de treinamento a árvore de decisão é criada e sua taxa de erros de classificação são estimadas pelo dados de teste.
- A árvore evolui com base nas amostras *bootstrap* extraídas do conjunto de treinamento. Logo, em seguida, o conjunto original de treinamento é utilizado para selecionar a melhor sub-árvore gerada. Essa etapa pode ser repetida K vezes.

Na figura 8 se observa um exemplo de *Random Forest*. No início, o conjunto de treinamento é separado em diversas amostras *bootstrap*, cada uma delas gerando sua própria árvore de decisão. Logo, quando um novo valor precisa ser classificado, cada árvore classifica esse valor independentemente, como se observa nos nós em laranja. Em seguida, cada resposta gerada é enviada para um sistema de votação majoritário e por fim, esse sistema irá retornar à classe mais recorrente nos resultados das árvores.

Figura 8 – Exemplo de *Random Forest*



Fonte: O autor

2.4 MÉTRICAS DE AVALIAÇÃO

Para avaliar um algoritmo de Aprendizado de Máquina, depois de treinado são enviados novos objetos, os quais o modelo ainda não conhece e assim, com os resultados das predições são montadas as métricas. Para problemas de duas classes, como o de detecção de infartos, comumente é definida uma classe como negativa e outra como positiva. Dessa forma é possível se obter a matriz de confusão (FACELI *et al.*, 2021).

Utilizando-se de exemplo qualquer modelo preditivo, que seu objetivo seja a classificação de cogumelos como venenosos ou não. Ao submeter o modelo a 100 cogumelos diferentes se obtém a matriz de confusão da figura 9, em que:

- Verdadeiro positivo (VP), corresponde ao número de exemplos da classe positiva (venenoso) classificadas corretamente.
- Verdadeiro negativo (VN), corresponde ao número de exemplos da classe negativa (não venenoso) classificadas corretamente.
- Falso positivo (FP), corresponde ao número de exemplos que sua classe verdadeira é a negativa, porém foram incorretamente classificados como positiva.
- Falso negativo (FN), corresponde ao número de exemplos que sua classe verdadeira é a positiva, porém foram incorretamente classificados como negativa.

Com a matriz de confusão completa é possível utilizar as métricas de VP, VN, FP e FN para a criação de outras métricas de avaliação de desempenho do classificador.

A taxa de erro total, representada pela equação 2.1, é calculada com a soma da diagonal secundária da matriz de confusão, dividido pelo valor total de elementos da matriz.

$$err(\widehat{f}) = \frac{FP + FN}{n} \tag{2.1}$$

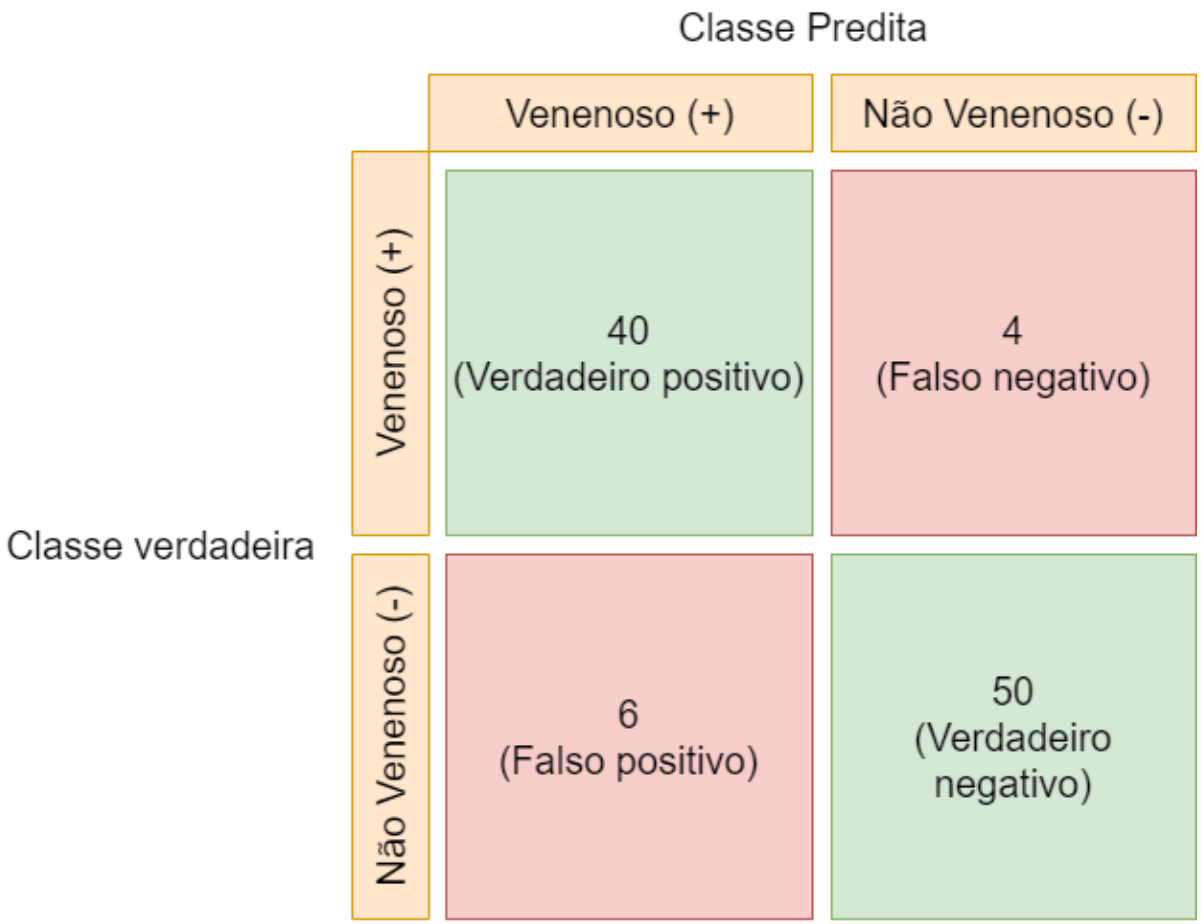
A taxa de acerto total ou acurácia, representada pela equação 2.2, semelhante a equação anterior, com a diferença que ela é calculada com a soma da diagonal principal da matriz de confusão, dividido pelo valor total de elementos da matriz.

$$ac(\widehat{f}) = \frac{VP + VN}{n} \tag{2.2}$$

A sensibilidade, representada pela equação 2.3, avalia a taxa de acertos na classe positiva. sendo assim, a quantidade de VP é dividido pela soma entre VP e FN.

$$sens(\widehat{f}) = \frac{VP}{VP + FN} \tag{2.3}$$

Figura 9 – Matriz de confusão



Fonte: O autor

A especificidade, representada pela equação 2.4, ao contrário da sensibilidade, a especificidade mede a taxa de acertos na classe negativa. Para isso, é usado o valor de VN e é feita a divisão pela soma de VN + FP.

$$esp(\widehat{f}) = \frac{VN}{VN + FP} \tag{2.4}$$

Agora utilizando essas métricas no exemplo de cogumelos da figura 9 obtemos os seguintes valores: a taxa de erro total (equação 2.5), que representa quantos valores foram classificados de forma errada, neste caso 1%. A acurácia (equação 2.6), indicando que 90% dos valores foram classificados corretamente. A sensibilidade (equação 2.7), apontando que o classificador acertou 90,9% das classificações para cogumelos venenosos. E por fim a especificidade (equação 2.8), estabelecendo que o 89,2% dos cogumelos não venenosos foram classificados corretamente.

$$err(\widehat{f}) = \frac{FP + FN}{n} = \frac{6 + 4}{100} = 0,1 \tag{2.5}$$

$$ac(\widehat{f}) = \frac{VP + VN}{n} = \frac{40 + 50}{100} = 0,9 \tag{2.6}$$

$$sens(\widehat{f}) = \frac{VP}{VP + FN} = \frac{40}{40 + 4} = 0,909 \tag{2.7}$$

$$esp(\widehat{f}) = \frac{VN}{VN + FP} = \frac{50}{50 + 6} = 0,892 \tag{2.8}$$

2.5 REVISÃO SISTEMÁTICA

Para dar início ao projeto foi realizada a revisão sistemática (SAMPAIO; MANCINI, 2007). Assim, o processo começou pela definição de um conjunto de palavras-chaves, esse é utilizado para a pesquisa bibliográfica. Assim, as palavras selecionadas para o projeto foram: "predict", "myocardial", "infarction", "neural"e "network". Também, para o portal de pesquisa foi utilizado o ScienceDirect. Além disso, foram utilizados os filtros disponibilizados pelo site, dessa forma, somente foram selecionados trabalhos publicados nos anos de 2019 até 2022. Então, os títulos de publicação filtrados foram: *Computer Methods and Programs in Biomedicine*, *Computers in Biology and Medicine*, *Biomedical Signal Processing and Control*, *Artificial intelligence in Medicine*, *Heart Rhythm, Machine Learning in Cardiovascular Medicine* e *Cardiovascular Digital Health Journal*. Ademais, as áreas de estudo também foram filtradas, sendo selecionadas apenas a *Medicine and Dentistry* e *Computer Science*, todos esses filtros foram aplicados para afunilar mais os resultados e ter como foco estudos mais próximos com a área de pesquisa escolhida.

Portanto, como consequência da busca foram obtidos 157 artigos, que foram selecionados com base nos seus títulos, tendo-se excluídos 68 publicações consideradas fora de contexto da pesquisa. Logo em seguida, foi realizada a leitura dos resumos. Com a leitura concluiu-se que apenas 40 publicações estavam relacionadas ao tema da pesquisa. Enfim, após a leitura das contribuições selecionadas, 8 artigos se destacaram sendo os mais relevantes. A partir do aprofundamento, identificou-se que os principais elementos a serem analisados eram os seguintes: método, algoritmos, *dataset*, ano de publicação e resultados.

No estudo desenvolvido por Liu *et al.* (2020) foi proposta uma maneira automatizada de identificar e localizar ataques cardíacos utilizando um algoritmo de redução de ruído. Assim, a partir do *dataset* PTB foram utilizadas as seguintes classes, as quais identificam a localização do ataque cardíaco: anterior (AMI), anterior-lateral (ALMI), inferior (IMI), anterior-septal (ASMI), inferior-lateral (ILMI), inferior-posterior lateral (IPLMI), além do grupo de pessoas saudáveis (H). Na etapa de pré-processamento, os dados foram filtrados utilizando o método Dual-Q TQWT, esse algoritmo de redução de ruído decompõem os sinais baseados em ressonância, com esse método um sinal pode ser decomposto na soma de um componente de alta

ressonância e um componente de baixa ressonância. Além disso, se fez uma extração de características usando o *Discrete wavelet packet transform* (DWPT) e por fim, utilizaram a técnica de análise de componentes principais multilinear (MPCA) para reduzir a dimensionalidade dos dados.

Na etapa de classificação foi utilizado como classificador o *Treebagger*, que por sua vez para realizar a classificação usufrui de diversas Árvores de decisão usando o resultado mais frequente como resultado final. Enfim, a performance do classificador foi medida pela sensibilidade, especificidade e acurácia. Para detecção de ataques cardíacos foi obtido uma acurácia de 99.98%, a sensitividade de 100%, e a especificidade de 99.90%. Para a tarefa de localização de ataques cardíacos foi obtida uma acurácia de 99.87%.

O objetivo do estudo realizado por Khan e Pachori (2021) é a automatização na detecção de infarto do miocárdio posterior (PMI), utilizando vetor cardiograma (VCG). O *dataset* utilizado para o estudo foi o PTBD, no qual, os dados de ECG foram convertidos para VCG. Na etapa de pré-processamento foi aplicada uma estratégia de remoção de ruído e logo em seguida o método FBSE-EWT foi aplicado para decompor os sinais do VCG. Para finalizar o pré-processamento ainda foi utilizado o PCA para redução de dimensionalidade.

No processo de classificação foram testados os classificadores, KNN, DT e SVM. A eficiência de cada classificador foi medida pela sensitividade, especificidade, pela acurácia, valor positivo preditivo, valor negativo preditivo, F1-score e a área sob a curva. O melhor classificador dentre eles foi o KNN, obtendo valores de sensitividade, especificidade, valor positivo preditivo, valor negativo preditivo, acurácia, F1-score e a área sob a curva respectivamente de 96.63%, 98.50%, 96.63%, 98.50%, 97.92%, 96.6% e 0.98.

Foi proposto neste estudo realizado por Jahmunah *et al.* (2021), a detecção da Doença arterial coronariana (CAD), sendo possível prevenir a ocorrência de outros problemas como o ataque cardíaco (IM). As classes selecionadas para classificação foram Normais (N), CAD, IM e insuficiência cardíaca (CHF). Durante o pré-processamento foi realizado um balanceamento do *dataset*.

Foram feitos testes com dois classificadores: o CNN e o Gabor CNN. A diferenciação do Gabor CNN é a utilização de filtros Gabor. Esses filtros são definidos por uma onda plana senoidal com frequências específicas. Apesar da aplicação do Gabor CNN, o classificador com melhores resultados foi o próprio CNN, cujo qual, teve uma acurácia, especificidade e sensibilidade respectivamente de 99,55%, 99,67% e 99,27%.

Como proposta do trabalho feito pelo Dai, Hwang e Tseng (2021), fez-se um classificador CNN para identificar cinco doenças cardiovasculares, sendo assim, as classes selecionadas foram: normais, ataque cardíaco, bloqueio de ramo, doença cardiovascular, cardiopatia CMH e cardiopatia CMD. Para esse projeto foi utilizado o *dataset* do PTB. Os dados do *dataset* foram segmentados em três diferentes intervalos: um segundo, dois segundos e três segundos.

Para pré-processamento, os intervalos segmentados foram expostos ao método de normalização min-max. Para resolver o problema de desbalanceamento de dados foi utilizado o método perda focal (FL).

Como dito no parágrafo acima, o classificador utilizado foi o CNN. O modelo que foi desenvolvido foi uma variação do ResNet, sendo uma das principais características desse modelo a introdução de conexões de atalhos entre as camadas convolucionais consecutivas além do fluxo original dos dados. Os melhores resultados foram obtidos com a segmentação de três segundos e como métricas de desempenho foram utilizadas a acurácia, sensibilidade e especificidade que obtiveram valores respectivamente de 99.84%, 99.52% e 99.95%.

O Fatimah *et al.* (2021), propôs nesse estudo o desenvolvimento de dois algoritmos para detecção de ataques cardíacos, sendo que a única diferenciação entre os dois se encontra no pré-processamento. No primeiro algoritmo foi utilizado o método FDM para cada batimento obtido no ECG. Por sua vez, o segundo algoritmo utiliza o FDM apenas uma vez. O FDM é um algoritmo que auxilia na remoção de dois ruídos gerados da etapa de coleta dos dados em ambiente clínico, o desvio da linha de base (BW) e a interferência da linha de energia (PLI). O BW é um ruído de baixa frequência, resultado de qualquer movimentação dos eletrodos, podendo ser movimentos musculares ou de respiração. Já o PLI são ruídos de 50 ou 60 Hz. Por ser um algoritmo eficiente ele pode ser utilizado de forma efetiva na eliminação de ruídos em tempo real.

Na classificação foram testados alguns algoritmos, são eles: *ensemble* subespaço k-vizinhos mais próximos (ESkNN), kNN, SVM, EBT. O ESkNN, é um conjunto de classificadores kNN, ele consiste em selecionar classificadores com base no seu desempenho, após isso o ESkNN utiliza os classificadores escolhidos para classificar os dados recebidos, seguindo um sistema de votação majoritária. O EBT é outro método de *ensemble*, ele combina várias árvores de decisão para melhorar o desempenho do classificador, reduzindo a variância de uma única árvore. O melhor classificador com base nas métricas de acurácia, sensibilidade e seletividade foi o kNN. Quando utilizado com o primeiro algoritmo seus resultados foram 99.96% de acurácia, 99.96% de sensibilidade e 99.95% de seletividade. Por outro lado, o segundo algoritmo obteve resultados piores, em troca do ganho de desempenho, sendo 99.65% de acurácia, 99.61% de sensibilidade e 99.73% de seletividade.

A ideia proposta pelo Hao *et al.* (2020) foi de criar um *framework* de fusão multi-ramal para detecção de ataques cardíacos, utilizando imagens de eletrocardiogramas com 12 derivações. Para obter melhores resultados foi projetada a rede multi-ramal composta por doze agentes independentes, cada um deles usando o mesmo método de extração de recursos, cada agente sendo uma das derivações do ECG.

Para a classificação foram testados os seguintes classificadores: ResNet, DenseNet e a Rede neural rasa. Após a classificação é implementado um método de fusão para reagrupar os valores antes divididos pelo multi-ramal. Os métodos podem ser a fusão de profundidade ou

a fusão de comprimento. Na fusão de profundidade os resultados são concatenados na dimensão da profundidade. Já na fusão de comprimento os resultados são concatenados na dimensão de comprimento. O melhor resultado alcançado pelo estudo foi a utilização do DenseNet juntamente com a fusão de profundidade. Os dois métodos juntos alcançaram uma acurácia de 94.73%, uma sensibilidade de 96.41%, uma especificidade de 95.95% e um pontuação F1 de 93.79%.

No estudo feito por Xiong *et al.* (2021), foi proposta uma abordagem multidimensional para localização de ataques cardíacos se baseando na rede convolucional densamente conectada (DenseNet). Utilizando o *dataset* PTB foram selecionadas as seguintes classes: controle saudável (HC), anterior (A), anterolateral (AL), anteroseptal (AS), ântero-septal lateral (ASL), inferior (I), inferolateral (IL), inferoposterior (IP), inferoposterior lateral (IPL), lateral (L), posterior (P) e posterolateral (PL). No pré-processamento foi aplicado um algoritmo de redução de ruído e os dados foram segmentados com base nos batimentos cardíacos.

A rede DenseNet foi a escolhida no estudo para construção do modelo de localização de ataque cardíaco e o Softmax para a classificação. Os vetores de características retornados pelo DenseNet são submetidos ao classificador Softmax para treinamento. No treinamento ainda foi utilizado o algoritmo Gradiente descendente para otimizar os parâmetros da rede. As métricas para medir o desempenho do programa foram a acurácia, sensibilidade e a especificidade, os melhores resultados obtidos foram e 99,87%, 99,84% e 99,98% respectivamente.

O objetivo do estudo realizado por He *et al.* (2021) é criar um algoritmo para diagnóstico de infartos através da onda de batimentos cardíacos e recursos de relacionamento de derivação para diagnósticos de MI multi categoria. Os dados utilizados para realização do projeto são *dataset* PTB. Foram selecionadas as categorias para classificação de pessoas com ataques cardíacos AMI, ASMI, ALMI, IMI e ILMI. Também, foi selecionada a categoria de pessoas saudáveis HC. Os valores do eletrocardiograma foram reduzidos de 1000 Hz para 250 Hz assim deixando o custo do cálculo menor e um algoritmo de redução de ruído foi aplicado para cada batimento cardíaco.

A arquitetura utilizada foi a MFB-LANN e ela se baseia em três características: especificidade, semelhança e integridade. Essas características vem de como se deve interpretar as derivações do eletrocardiograma. Por causa da especificidade e da similaridade das derivações, os valores pré-processados do eletrocardiograma de doze derivações são colocados em doze ramos de convolução separados. Então, são atribuídos pesos para cada ramificação, dessa forma, fortalecendo os recursos essenciais e enfraquecendo as demais. Por fim, todos os ramos são fundidos para a classificação com base na integridade do ECG. Enfim os todos os vetores são integrados pelo *fully connected* e enviados para o softmax para classificação. As métricas para medir a eficiência foram acurácia, sensibilidade, especificidade, previsibilidade positiva e F1. Os melhores resultados obtidos foram acurácia 99.63%, sensibilidade 99.61%, especificidade 99.93%, previsibilidade positiva 99.56% e F1 99.59%.

2.6 ANÁLISE DA REVISÃO SISTEMÁTICA

Como se observa na revisão sistemática, a maioria dos estudos utiliza o *dataset* PTB quando se deseja fazer classificadores baseados em dados numéricos, tornando evidente sua predominância no estado da arte, tanto para casos de ECG de uma ou doze dimensões. Por outro lado, no caso do estudo Hao *et al.* (2020), que foi voltado para um classificador baseado em imagens, foi solicitado para o hospital Zhejiang, segundo o Hospital Popular da China, as imagens de Eletrocardiograma.

Além disso, uma quantidade diversa de algoritmos foram testados, sendo que os que mais se destacaram foram: KNN, CNN e Árvores de decisão. Com acurácias que variam de 94,73% até 99,98%. Porém, o modelo com maior acurácia foi a Árvore de decisão, obtendo a acurácia de 99.98%.

Por fim, os resultados da revisão sistemática (Tabela 1) mostram que os padrões de coleta dos ECG foram baseados em doze ou uma única dimensão. As tecnologias portáteis para coleta de ECG costumam ser de apenas uma dimensão, como ocorre quando um paciente utiliza um *smartwatch* ou *smartphone*. Mas, como analisado no estado da arte, independente de ser 12 ou apenas 1 dimensão os algoritmos já possuem acurácia elevada em ambos os casos. Sendo assim, possibilitando a criação de uma API que retorne o estado do paciente, dessa forma tornaria viável, em um segundo momento, a criação de diversos clientes para inúmeros dispositivos como *smarthphones*, *smarthwatchs* ou qualquer outro dispositivo capaz de coletar sinais de ECG e se conectar a API.

Tabela 1 – Resultados da Área Científica

Referência	**Algoritmos**	***Dataset***	**Tipo**	**Métricas**	**Acurácia**
(LIU *et al.*, 2020)	Treeberger e Árvores de decisão	PTB	Numéricos	Sensibilidade, especificidade e acurácia	99,98%
(KHAN; PACHORI, 2021)	KNN, DT e SVM	PTB	Numéricos	Sensibilidade, especificidade, valor positivo preditivo, valor negativo preditivo, acurácia e F1-score	97,92%
(JAHMUNAH *et al.*, 2021)	CNN e o Gabor CNN	PTB	Numéricos	Acurácia, especificidade e sensibilidade	99,55%
(DAI; HWANG; TSENG, 2021)	CNN	PTB	Numéricos	Acurácia, sensibilidade e especificidade	99,84%
(FATIMAH *et al.*, 2021)	ESkNN, kNN, SVM, EBT	PTB	Numéricos	Acurácia, sensibilidade e seletividade	99,96%
(HAO *et al.*, 2020)	ResNet, DenseNet e a Rede neural rasa	Zhejiang Segundo Hospital Popular da China	Imagem	Acurácia, sensibilidade, especificidade e pontuação F1	94,73%
(XIONG *et al.*, 2021)	DenseNet	PTB	Numéricos	Acurácia, sensibilidade e especificidade	99,87%
(HE *et al.*, 2021)	MFB-LANN	PTB e PTB-XL	Numéricos	Acurácia, sensibilidade, especificidade, previsibilidade positiva e F1	99,63%

Fonte: O Autor.

3 MODELO DE CLASSIFICAÇÃO DE INFARTO

Esse capítulo detalha o desenvolvimento da implementação proposta neste trabalho, mostrando as tecnologias e etapas que foram empregadas para a construção de um modelo de classificação de infarto do miocárdio. Esse modelo foi desenvolvido a partir de técnicas de Aprendizado de Máquina.

3.1 CAMINHO METODOLÓGICO

O trabalho foi uma pesquisa de natureza exploratória, a qual visou investigar, compreender e aplicar técnicas de Aprendizado de Máquina no contexto da classificação de problemas cardíacos, com o intuito de construir um modelo a ser aplicado na área da saúde e acessível via API.

Para isso, o trabalho foi desenvolvido em sete etapas:

- Etapa 1: identificação dos dados (*construção do dataset*)
- Etapa 2: pré-processamento dos dados
- Etapa 3: transformação dos dados
- Etapa 4: aplicação do algoritmo
- Etapa 5: comparação e avaliação dos modelos
- Etapa 6: seleção do melhor modelo
- Etapa 7: criação de API para acesso e uso do modelo

3.1.1 Etapa 1: identificação dos dados (*construção do dataset*)

Uma das maneiras mais consolidadas para se diagnosticar ataques cardíacos é o eletrocardiograma. Sendo assim, faz-se necessário uma base de informações contendo sinais gerados pelos sensores do ECG para vários pacientes diferentes. Portanto, foi feita uma pesquisa e constatou-se que o *dataset* mais completo para a realização do projeto é o PTB-XL (WAGNER *et al.*, 2020a). Essa escolha foi feita a partir da revisão sistemática, a qual identificou o PTB como o *dataset* mais utilizado entre os estudos.

A base de dados escolhida foi um grande conjunto de dados de 21.837 eletrocardiogramas clínicos de 12 derivações, de 18.885 pacientes com duração de 10 segundos, alguns desses ECG podem conter mais de uma classe. Além disso, os sinais de ECG, forma de onda, foram anotados por cardiologistas, que atribuíram suas declarações para cada registro. Desses dados,

52% são do sexo masculino e 48% são do sexo feminino, as idades variam entre 0 a 95 anos e sua mediana é de 62 anos. Também, possuem cinco classes majoritárias como podemos ver na tabela 2, sendo elas: ECG normal, Infarto do miocárdio, Mudança de ST/T, Distúrbio de condução e Hipertrofia (WAGNER *et al.*, 2020a). Ademais, o *dataset* é dividido em dois tipos de arquivos: CSV e WFDB. Os arquivos WFDB são as ondas do ECG, cada um possuindo o equivalente a 10 segundos de informações vindas do dispositivo de coleta. O restante das informações estão presentes no formato CSV, os dados de cada paciente ficam armazenadas no arquivo ptbxl_database.csv que contém o total de 28 colunas, cada uma delas especificada na tabela 3.

Tabela 2 – Registros do *dataset* PTB-XL

#Instâncias	**Classe**	**Descrição**
9528	NORMA	ECG normal
5486	MI	Infarto do miocárdio
5250	STTC	Mudança de ST/T
4907	CD	Distúrbio de condução
2655	HYP	Hipertrofia

Fonte: Adaptado de Wagner *et al.* (2020a).

Além disso, cada paciente possui dois arquivos de eletrocardiograma, sendo um deles de 100Hz e outro de 500Hz. Cada um deles possui doze dimensões e sua principal diferença é a quantidade de valores. Cada um desses arquivos representa 10 segundos de informação, porém, o de 500Hz possui muito mais valores do que o de 100Hz. Sendo o corpo da matriz de 100Hz [1000, 12] e a de 500Hz [5000, 12]. Visualmente quase não existe diferença entre os dois valores como podemos ver na figura 10 de um ECG de 100Hz e a figura 11 de um ECG de 500Hz. Porém, mesmo com uma sutil diferença podemos notá-la quando damos um zoom como na figura 12, o ECG de 500Hz a direita é muito mais denso do que seu equivalente de 100Hz a esquerda.

3.1.2 Etapa 2: pré-processamento dos dados

O desempenho de um algoritmo de Aprendizado de Máquina está diretamente relacionado à qualidade do conjunto de dados. Os conjuntos de dados possuem diferentes características, dimensões e formatos. Eles ainda podem ser dados limpos ou conter ruídos, valores incorretos, inconsistentes, duplicados ou ausentes. A etapa de pré-processamento é responsável por realizar a limpeza dos dados e melhorar a sua qualidade, eliminando ou minimizando os problemas citados (FACELI *et al.*, 2021).

O *dataset* escolhido possui arquivos em dois tipos de formato: CSV e WFDB. O *dataset*

Tabela 3 – Colunas do *dataset*

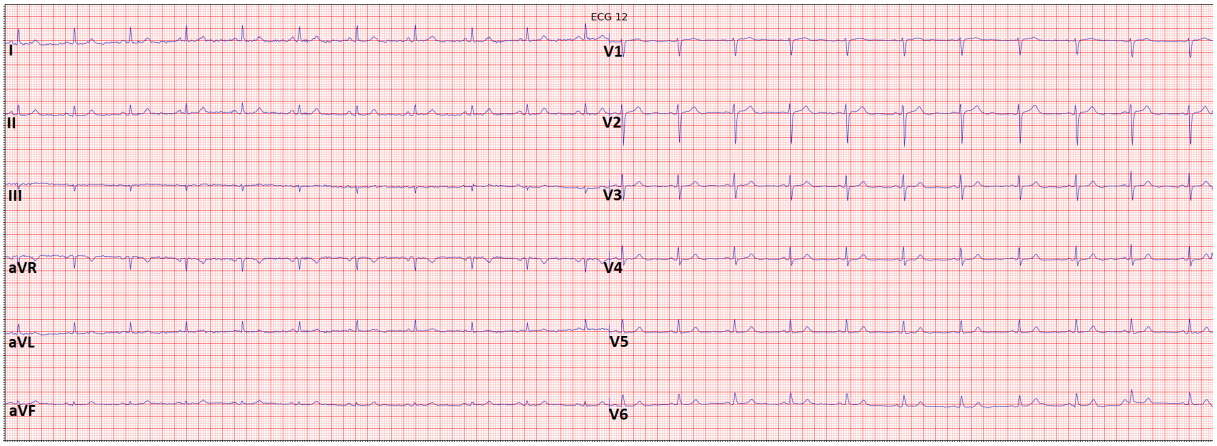
Coluna	Tipo	Descrição
ecg_id	integer	Identificador único do ECG
patient_id	integer	Identificador único do paciente
filename_lr	string	Caminho para o WFDB (100Hz)
filename_hr	string	Caminho para o WFDB (500Hz)
age	integer	Idade em anos
sex	categorical	Sexo (masculino 0, feminino 1)
height	integer	Altura em centímetros
weight	integer	Peso em quilogramas
nurse	categorical	Enfermeira envolvida
site	categorical	Local de gravação
device	categorical	Dispositivo de gravação
recording_date	datetime	Data e hora da gravação do ECG
report	string	Relatório de ECG do cardiologista de diagnóstico
scp_codes	dictionary	Declarações SCP ECG
heart_axis	categorical	Eixo elétrico do coração
infarction_stadium1	categorical	Fase do infarto
infarction_stadium2	categorical	Segunda fase do infarto
validated_by	categorical	Cardiologista validador
second_opinion	boolean	Sinalizado como segunda opinião
initial_autogenerated_report	boolean	Relatório inicial gerado automaticamente pelo dispositivo de ECG
validated_by_human	boolean	Validado por humanos
baseline_drift	string	Desvio de linha de base ou salto presente
static_noise	string	Zumbido elétrico/ruído estático presente
burst_noise	string	Ruído de ruptura
electrodes_problems	string	Problemas de eletrodos
extra_beats	string	Batidas extras
pacemaker	string	Marcapasso
strat_fold	integer	Dobras estratificadas sugeridas

Fonte: Adaptado de Wagner *et al.* (2020b).

possui diversas classes, nessa etapa foi feita a remoção das classes não utilizadas (STTC, CD e HYP), dessa forma reduzindo consideravelmente a quantidade de dados e por consequência transformando a solução em um problema binário sem ter que classificar de forma errada as classes não utilizadas. Portanto, foram conservadas somente as classes NORMA e MI.

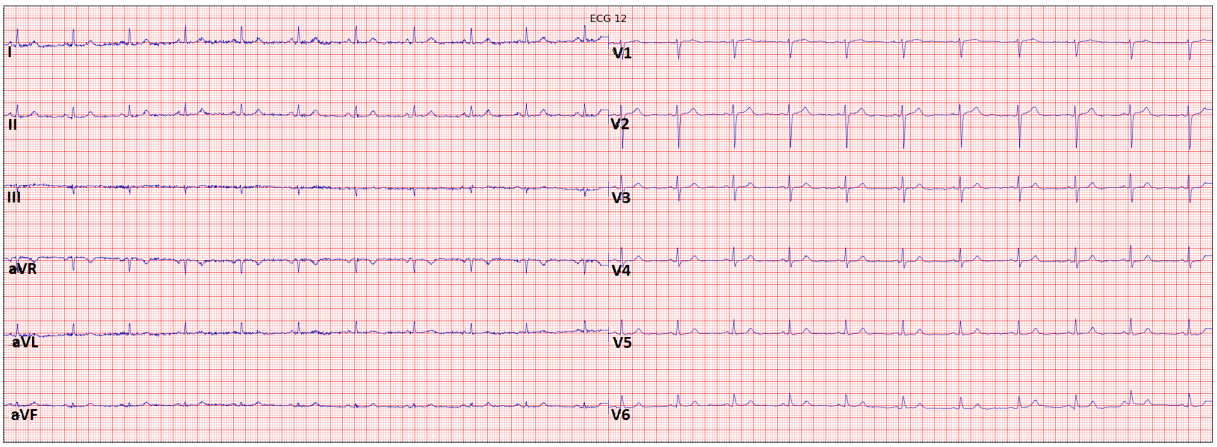
Ademais, existem muitas colunas com valores nulos. A figura 13 mostra uma visão geral de colunas preenchidas. Cada entrada corresponde a uma linha na tabela. *Pixels* pretos indicam valores existentes; valores ausentes permanecem em branco. Sendo assim, as colunas

Figura 10 – Eletrocardiograma 100Hz



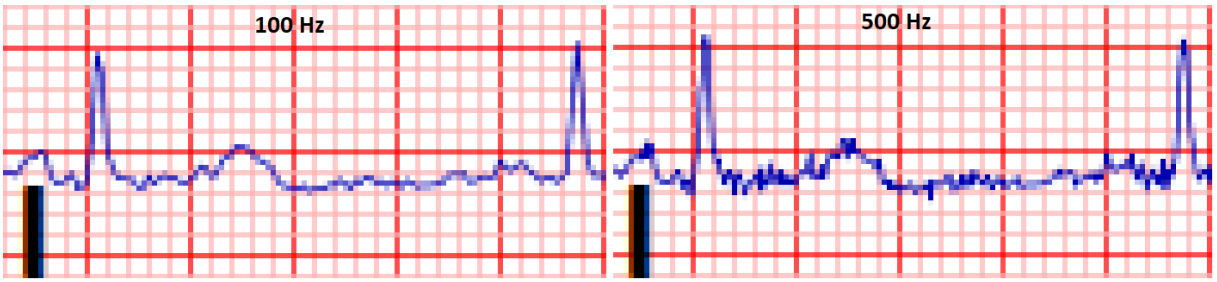
Fonte: O Autor

Figura 11 – Eletrocardiograma 500Hz



Fonte: O Autor

Figura 12 – Eletrocardiograma 100Hz x Eletrocardiograma 500Hz

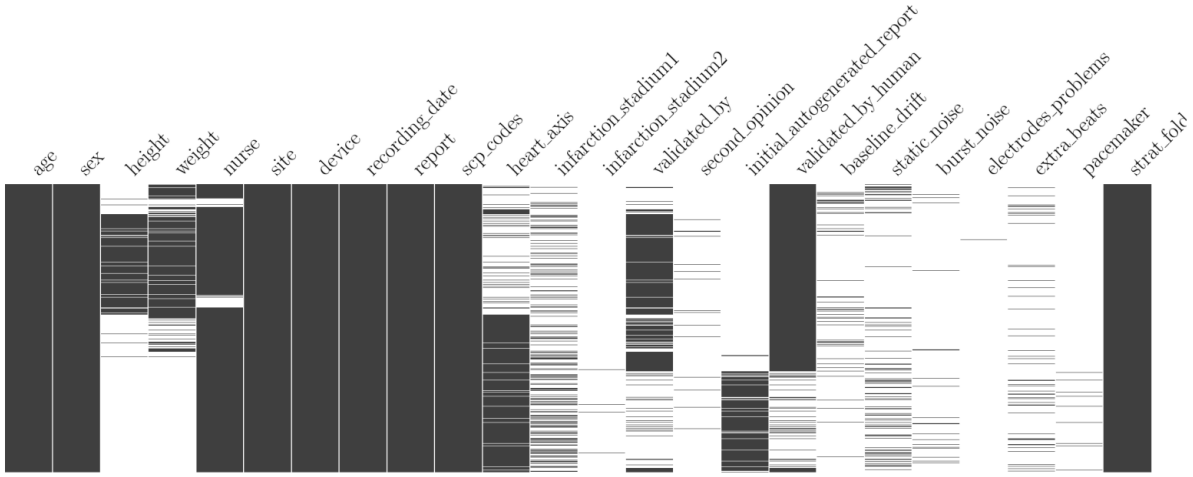


Fonte: O Autor

com muitos dados faltando também devem ser removidas. Por fim, das colunas restantes, podem ser utilizadas apenas as relevantes para a detecção de infartos (LONGO *et al.*, 2013), sendo elas: idade, sexo e os sinais de ECG.

Além disso, teve que ser feita a escolha entre os arquivos de ECG de 100Hz e 500Hz. Os arquivos de 500Hz por possuírem mais dados auxiliam a obter resultados melhores para

Figura 13 – Dados nulos



Fonte: Wagner *et al.* (2020b)

o aprendizado dos modelos, porém o custo computacional seria muito mais elevado visto que os arquivos de 500Hz possuem cinco vezes mais dados do que os de 100Hz. Por esse projeto implementar uma API que retorna dados em tempo real foi escolhido os arquivos de 100Hz para preservar a velocidade de previsão do modelo.

Também, nessa etapa, dividiu-se os valores entre treino e teste, para isso foi utilizada a própria sugestão do *dataset* para realizar essa divisão. Com o auxílio da coluna *strat_fold* com valores no intervalo de 1 até 10, foi possível testar diversas combinações de dados de teste e treino, todas possuindo um valor aproximado de 10% para teste e 90% para treino.

3.1.3 Etapa 3: transformação dos dados

Diversas técnicas de Aprendizado de Máquina são limitadas à manipulação de valores de determinados tipos, por exemplo, apenas valores numéricos ou simbólicos. Algumas outras técnicas possuem uma perda no desempenho devido a variação dos valores numéricos (FACELI *et al.*, 2021). Por isso, essa etapa foi responsável por normalizar os dados e prepará-los para a aplicação no modelo. A primeira parte desse processo foi adquirir os valores dos arquivos WFDB, utilizando a biblioteca do Python wfdb, assim, foi possível que os arquivos fossem convertidos para uma matriz [1000, 12]. A figura 14 mostra como foi esse processo, na linha 15 foi escolhido os arquivos de 100hz, logo em seguida o *dataset* foi carregado. Então, foi necessário trazer os valores de ECG, para isso na linha 20 foi chamada a função *load_raw_data*, nela foi feita um *for* que passa por todos os pacientes lendo seus arquivos de ECG com o auxílio da biblioteca wfdb e adicionando em uma lista. Além disso, como a maioria dos modelos não aceitam matriz, foi preciso realizar a redução de dimensionalidade dessa matriz, assim tornando ela um array. Ademais, foi criado mais um array para adicionar as informações de idade e sexo, dessa forma todos os dados necessários foram apresentados ao modelo de aprendizado.

Figura 14 – Convertendo dados para matriz

```python
import pandas as pd
import numpy as np
import wfdb
import ast

def load_raw_data(df, sampling_rate, path):
    if sampling_rate == 100:
        data = [wfdb.rdsamp(path+f) for f in df.filename_lr]
    else:
        data = [wfdb.rdsamp(path+f) for f in df.filename_hr]
    data = np.array([signal for signal, meta in data])
    return data

path = './ptb-xl/'
sampling_rate=100

Y = pd.read_csv(path+'ptbxl_database.csv', index_col='ecg_id')
Y.scp_codes = Y.scp_codes.apply(lambda x: ast.literal_eval(x))

X = load_raw_data(Y, sampling_rate, path)
```

Fonte: O Autor

Por fim, foi feita a conversão das classes NORMA e MI para dados numéricos para que elas se encaixem aos modelos de Aprendizado de Máquina, sendo NORMA representado por 0 e MI representado por 1. Na figura 15, é mostrado como esse processo foi feito, a função *convert_class_to_number* faz um *for* por todos os valores encontrando as classes NORM e MI e adicionando o número referente de cada classe em uma nova lista.

3.1.4 Etapa 4: aplicação dos algoritmos

Nessa etapa foram implementados e testados os algoritmos de classificação. Como visto na análise da revisão sistemática (Seção 2.6), o algoritmo mais relevante foi a Árvore de Decisão, pois obteve a maior acurácia entre todos os outros modelos analisados. Sendo assim, o algoritmo utilizado foi a Árvore de Decisão (Seção 2.3.1) e também foi testada a técnica de *ensemble Random Forest* (Seção 2.3.2). Os códigos da Árvore de Decisão podem ser vistos na figura 16 e o código para o *Random Forest* na figura 17. Em ambas as figuras os modelos são instanciados com parâmetros que ainda não foram otimizados, logo em seguida foi feito o treino desses modelos e sua aplicação para tentar classificar as instâncias de teste.

Figura 15 – Convertendo classe para número

```python
def convert_class_to_number(Y):
    data = []
    for classification in Y:
        if 'NORM' in classification:
            data.append(0)
        elif 'MI' in classification:
            data.append(1)
    return data

y_train_num = convert_class_to_number(y_train)
y_test_num = convert_class_to_number(y_test)
```

Fonte: O Autor

3.1.5 Etapa 5: comparação e avaliação dos modelos

O objetivo dessa etapa foi a criação de diversos classificadores *Random Forest* e Árvore de Decisão com parâmetros diferentes para se ter uma grande variedade de modelos e posteriormente a seleção do melhor modelo. Para a otimização dos hiper-parâmetros foi aplicado o método de *Grid Search*. A técnica se baseia no agrupamento de um conjunto de valores para hiper-parâmetros distintos. Assim, operar os modelos para cada combinação possível dos elementos desses conjuntos. Contudo, foram descartadas as combinações que geraram modelos mais caros, caso o custo computacional dos conjuntos escolhidos exija muito tempo de processamento.

Os hiper-parâmetros para o *Random Forest* são os seguintes:

- *n_estimators*: a quantidade de árvores que o *Random Forest* vai possuir
- *min_samples_split*: o mínimo de amostras para realizar a divisão de um nó interno
- *min_samples_leaf*: o mínimo de amostras para se chegar a um nó folha
- *max_leaf_nodes*: o máximo de nós folha
- *max_depth*: a profundidade máxima das árvores

O intervalo de valores testados para o *Random Forest* podem ser analisados na tabela 4.

Os hiper-parâmetros para o Árvore de Decisão são os seguintes:

Figura 16 – Árvore de Decisão

```python
from sklearn.tree import DecisionTreeClassifier

params = {
    'max_depth': 10,
    'max_leaf_nodes': 30,
    'min_samples_leaf': 1,
    'min_samples_split': 2
}

clf = DecisionTreeClassifier(
    max_depth=params['max_depth'],
    max_leaf_nodes=params['max_leaf_nodes'],
    min_samples_leaf=params['min_samples_leaf'],
    min_samples_split=params['min_samples_split']
)
clf.fit(X_train_arr, y_train_num)

predict = clf.predict(X_test_arr)
```

Fonte: O Autor

Tabela 4 – Intervalo de hiper-parâmetros do *Random Forest*

Parâmetro	**Intervalo**
n_estimators	10 até 1500
min_samples_split	1 até 10
min_samples_leaf	1 até 10
max_leaf_nodes	Sem limite e de 1 até 50
max_depth	10 até 150

Fonte: O Autor.

- *min_samples_split*: o mínimo de amostras para realizar a divisão de um nó interno
- *min_samples_leaf*: o mínimo de amostras para se chegar a um nó folha
- *max_leaf_nodes*: o máximo de nós folha
- *max_depth*: a profundidade máxima das árvores

Figura 17 – Random Forest

```python
from sklearn.ensemble import RandomForestClassifier

params = {
    'bootstrap': False,
    'max_depth': 110,
    'min_samples_leaf': 1,
    'min_samples_split': 6,
    'n_estimators': 1200,
    'max_leaf_nodes': None
}

clf = RandomForestClassifier(
    bootstrap=params['bootstrap'],
    max_depth=params['max_depth'],
    min_samples_leaf=params['min_samples_leaf'],
    min_samples_split=params['min_samples_split'],
    n_estimators=params['n_estimators'],
    max_leaf_nodes=params['max_leaf_nodes'],
)
clf.fit(X_train_arr, y_train_num)

predict = clf.predict(X_test_arr)
```

Fonte: O Autor

O intervalo de valores testados para a Árvore de Decisão podem ser analisados na tabela 5.

3.1.6 Etapa 6: seleção do melhor modelo

Para avaliação das técnicas implementadas foram empregados os critérios regularmente utilizados na literatura, conforme localizado na Seção 2.4. Em obras que envolveram Aprendizado de Máquina utilizaram-se métricas quantitativas para julgar os modelos. Sendo assim, a avaliação acompanhou os padrões fixados na área, baseados em critérios numéricos de acurácia do modelo, sendo eles: taxa de erro total, acurácia, sensibilidade e especificidade. Desta forma, foram avaliados os melhores modelos para *Random Forest* e Árvore de Decisão.

Dos modelos testados com *Random Forest* o melhor deles obteve a matriz de confusão mostrada na figura 18. Essa figura apresenta o total de 1463 valores, sendo 861 valores ECG

Tabela 5 – Intervalo de hiper-parâmetros da Árvore de Decisão

Parâmetro	**Intervalo**
min_samples_split	1 até 10
min_samples_leaf	1 até 10
max_leaf_nodes	Sem limite e de 1 até 50
max_depth	10 até 150

Fonte: O Autor.

normais classificados corretamente, 51 ECG normais classificados incorretamente, 315 infartos classificados corretamente e 236 infartos classificados incorretamente. Agora utilizando essas métricas para o *Random Forest* obtemos os seguintes valores: a taxa de erro total (equação 3.1), que representa quantos valores foram classificados de forma errada, neste caso 19,61%. A acurácia (equação 3.2), mostrando que 80,38% dos valores foram classificados corretamente. A sensibilidade (equação 3.3), apontando que o classificador acertou 94,40% das classificações para ECG normais. E por fim a especificidade (equação 3.4), indicando que o 57,16% dos infartos foram classificados corretamente.

$$err(\widehat{f}) = \frac{FP + FN}{n} = \frac{51 + 236}{1463} = 0,1961 \tag{3.1}$$

$$ac(\widehat{f}) = \frac{VP + VN}{n} = \frac{861 + 315}{1463} = 0,8038 \tag{3.2}$$

$$sens(\widehat{f}) = \frac{VP}{VP + FN} = \frac{861}{861 + 51} = 0,9440 \tag{3.3}$$

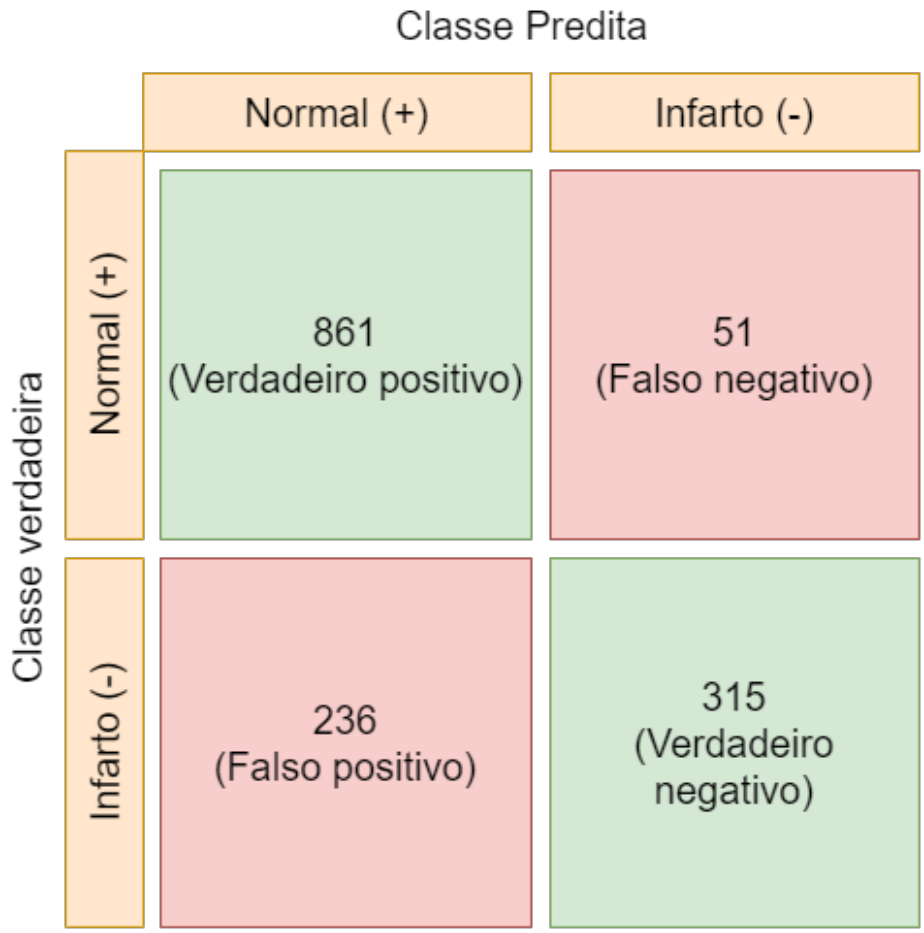
$$esp(\widehat{f}) = \frac{VN}{VN + FP} = \frac{315}{315 + 236} = 0,5716 \tag{3.4}$$

Agora dos modelos testados com Árvore de Decisão a figura 18 apresenta a melhor matriz de confusão adquirida para esse algoritmo. Essa figura também apresenta o total de 1463 valores, sendo 719 valores ECG normais classificados corretamente, 193 ECG normais classificados incorretamente, 306 infartos classificados corretamente e 245 infartos classificados incorretamente. Agora utilizando essas métricas para Árvore de Decisão obtemos os seguintes valores: a taxa de erro total (equação 3.1) 29,93%. A acurácia (equação 3.2) 70,06%. A sensibilidade (equação 3.3) 78,83%. E por fim a especificidade (equação 3.4) 55,53%.

$$err(\widehat{f}) = \frac{FP + FN}{n} = \frac{193 + 245}{1463} = 0,2993 \tag{3.5}$$

$$ac(\widehat{f}) = \frac{VP + VN}{n} = \frac{719 + 306}{1463} = 0,7006 \tag{3.6}$$

Figura 18 – Matriz de Confusão *Random Forest*



Fonte: O Autor

$$sens(\widehat{f}) = \frac{VP}{VP + FN} = \frac{719}{719 + 193} = 0,7883 \tag{3.7}$$

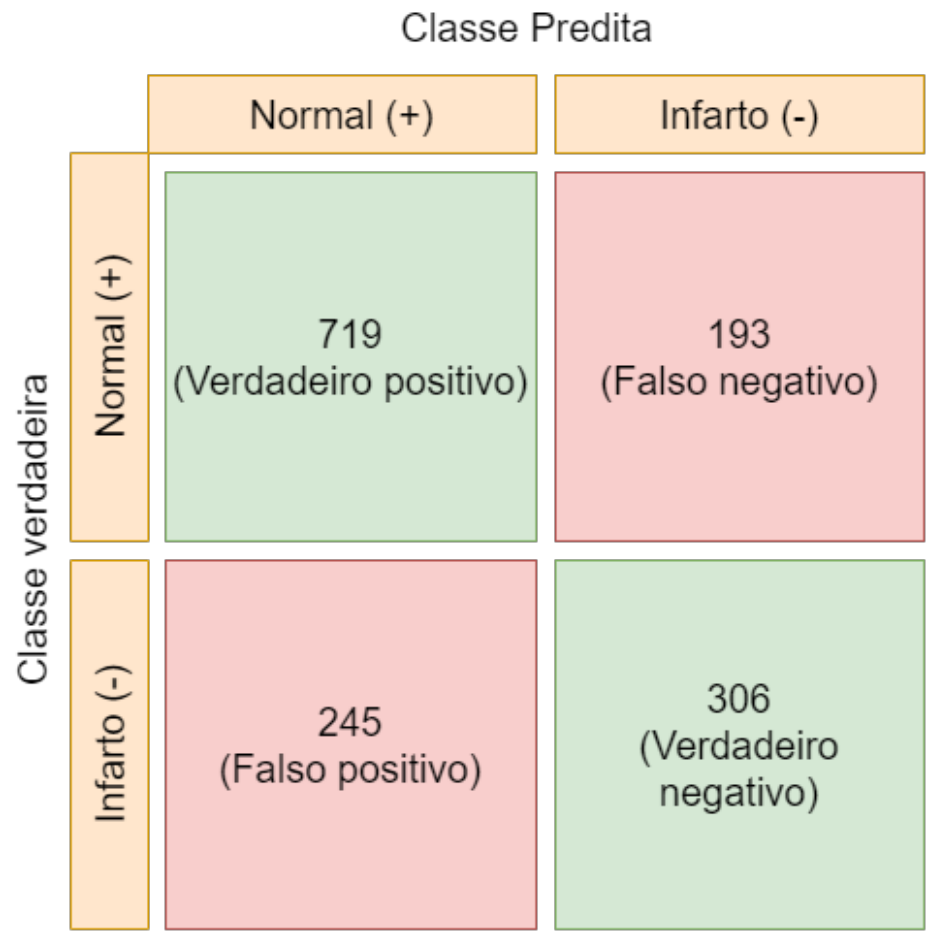
$$esp(\widehat{f}) = \frac{VN}{VN + FP} = \frac{315}{315 + 51} = 0,5553 \tag{3.8}$$

O *Random Forest* possuiu um desempenho melhor do que a Árvore de Decisão, possuindo valores melhores em todas as métricas avaliadas, principalmente na sensibilidade, que obteve uma diferença de 24 pontos. Por conta disso, o método selecionado para dar continuidade na construção da API foi o *Random Forest*. Por fim, os melhores hiper-parâmetros encontrados para o *Random Forest* são os mostrados na tabela 6.

3.1.7 Etapa 7: criação de API para acesso e uso do modelo

Dentre os trabalhos estudados, a maior parte deles se preocupou exclusivamente em desenvolver um método de Aprendizado de Máquina. Contudo, somente o modelo de aprendizado não é o suficiente, necessitando de uma plataforma para que estes modelos possam ser

Figura 19 – Matriz de Confusão Árvore de Decisão



Fonte: O Autor

Tabela 6 – Melhores hiper-parâmetros do *Random Forest*

Parâmetro	**Intervalo**
n_estimators	1300
min_samples_split	6
min_samples_leaf	1
max_leaf_nodes	Sem limite
max_depth	110

Fonte: O Autor.

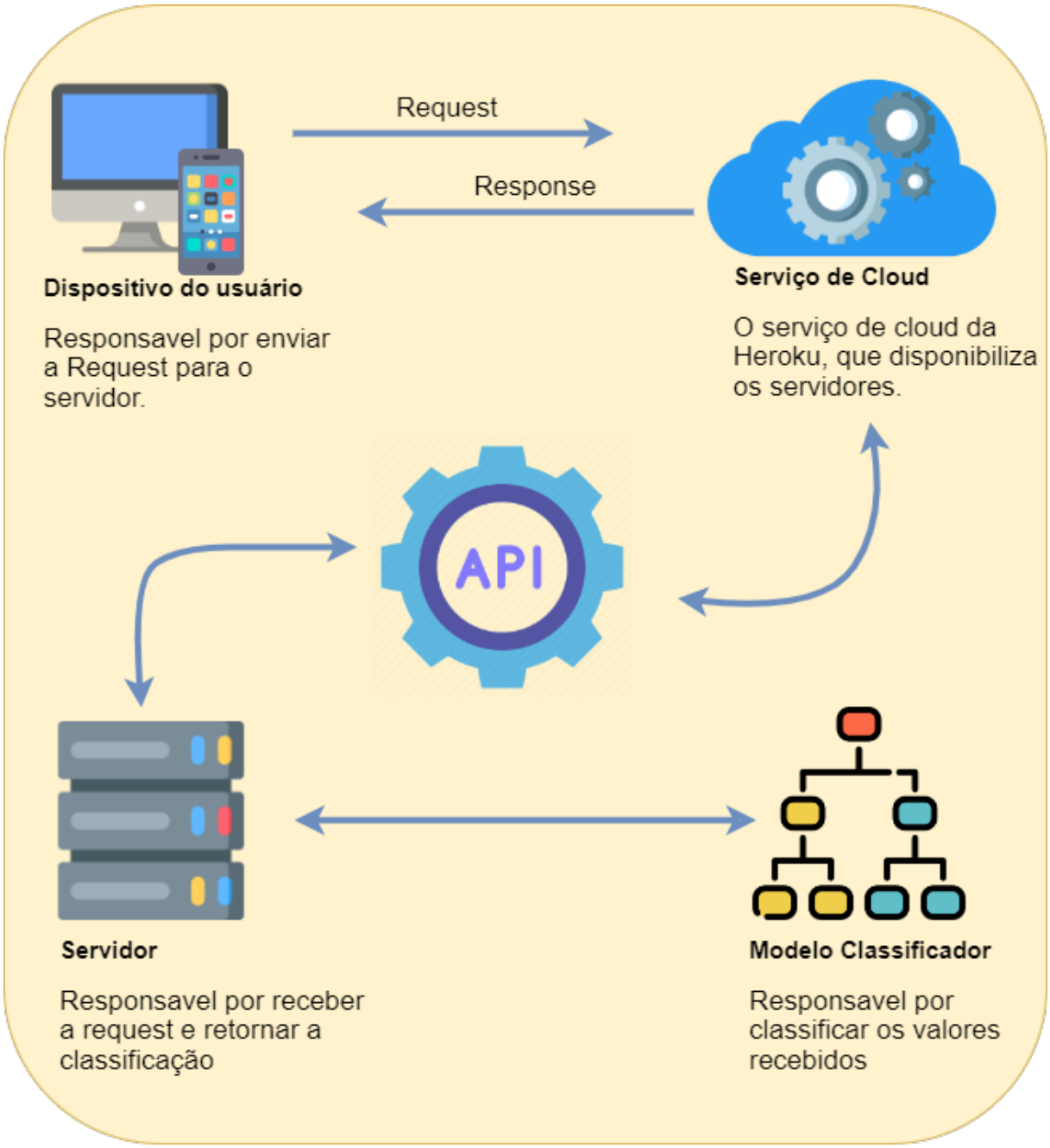
utilizados. A solução para esse problema foi a criação de uma interface para que programadores possam utilizá-la em suas aplicações sem a necessidade de implementar o modelo de Aprendizado de Máquina.

A figura 20 apresenta diagrama da API, o qual foi desenvolvido o modelo *Random Forest* obtido a partir do treinamento com o *dataset* PTB-XL e uma API para acesso externo via clientes (cujo o desenvolvimento não está incluído dentro deste TCC), foi desenvolvido um

cliente para teste e acesso a API.

A figura representa os possíveis clientes da API Rest, eles enviam os sinais de ECG para o servidor que através do método *Random Forest* classifica os sinais recebidos e retorna a classificação final para o cliente.

Figura 20 – Diagrama da API Rest



Fonte: O Autor

Para implementação dessa arquitetura foram utilizadas as plataformas Python [1], Scikit

[1] https://www.python.org/

Learn [2] e React Native [3] para definição de uma aplicação cliente simples. Como IDE de desenvolvimento foi utilizada a plataforma Jupyter. Para fins de hospedagem da API foi utilizado os servidores da Heroku [4].

Dessa forma, para o desenvolvimento da API foi utilizada a biblioteca de Python *FastAPI*, esta possibilita a criação de rotas de acesso a API de forma simples e prática. Na figura 21 observamos a implementação dessa API, na linha 6 se inicia a aplicação, logo em seguida foi carregado o modelo de classificação *Random Forest*. Assim, na linha 10 foi criado a classe *Pacient* que representa a interface que será validada na função *mi_classification*. Também, na linha 21 foi instanciada a rota *classification* que quando acessada recebe os valores de idade, sexo e sinais de ECG e utiliza o modelo anteriormente carregado para classificar como infarto ou normal.

No desenvolvimento do aplicativo obteve-se três estados, o sem classificação, o normal e o infarto. Os três estados são bem semelhantes, possuindo três botões na parte central da tela, sendo eles: Enviar ECG Normal, Enviar ECG de Infarto e *RESET*. Ao iniciar o aplicativo nos deparamos com o primeiro estado o "sem classificação"(figura 22), ele possui um fundo preto e apresenta os botões para interação com a API. Assim, quando é clicado em "Enviar ECG Normal"é enviado uma *request* para a API com a idade, sexo e o ECG de um paciente normal e com a resposta recebida o estado é alterado para "normal"e o fundo é alterado para verde (figura 23). Já quando se clica em "Enviar ECG de Infarto"é escolhido outro paciente, porém, dessa vez com um ECG de infarto e é feita mais uma *request* para a API, e com a resposta recebida o estado é alterado para "infarto"e o fundo é alterado para vermelho (figura 24). Agora, se clicado em *Reset*, o estado é alterado diretamente para "sem classificação".

Ainda, com auxílio dos simuladores do Android Studio [5], foi possível testar o aplicativo e a funcionalidade da API, enviando os valores de sexo, idade e os sinais de ECG para interface e recebendo a classificação como resposta, dessa forma analisando o comportamento do aplicativo nas trocas de estado.

3.2 CONSIDERAÇÕES FINAIS

Nesse capítulo se partiu de um conjunto de dados para desenvolver um modelo de Aprendizado de Máquina e a API para acesso ao modelo. Foram utilizados os métodos de Árvore de Decisão e *Random Forest*. O melhor resultado obtido pelos experimentos foram do *Random Forest*, com uma acurácia de 80,38% a sensibilidade de 94,40% e especificidade de 57,16%. Estes resultados foram abaixo do que os encontrados no estado da arte, provavelmente relacionado aos ruídos nos sinais de eletrocardiograma que normalmente ocorrem nas máqui-

[2] https://scikit-learn.org/
[3] https://reactnative.dev/
[4] http://heroku.com
[5] https://developer.android.com/studio

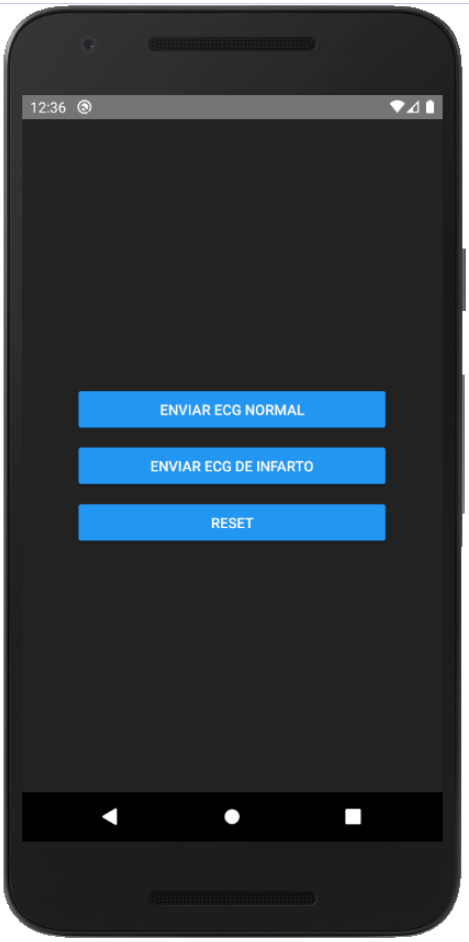
Figura 21 – Código

```python
from fastapi import FastAPI
import joblib
from pydantic import BaseModel
from model.mi import mi

app = FastAPI()

model = joblib.load("./random_forest.joblib")

class Pacient(BaseModel):
  data: list

  class Config:
    schema_extra = {
      "example": {
        "data": mi
      }
    }

@app.post("/classification")
async def mi_classification(pacient: Pacient):
    data = list(map(float, pacient.data))

    prediction = model.predict([data])

    return {"classification": str(prediction[0])}
```

Fonte: O Autor

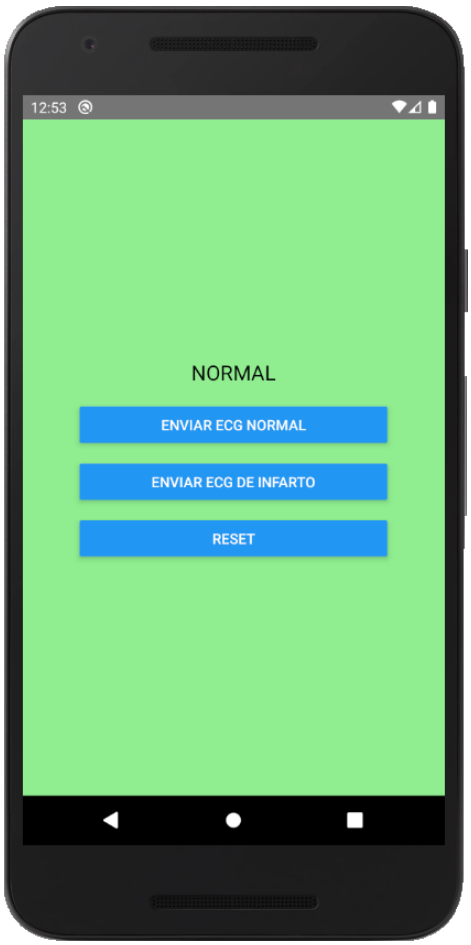
nas que coletam esses sinais. A solução para esse problema é a implementação de um redutor de ruído, como foi feito em outros estudos como o Liu *et al.* (2020) que usufruiu de algoritmos de redução de ruídos para obter melhores resultados. Além disso, a API foi implementada utilizando o modelo *Random Forest* e foi testada com o auxílio do aplicativo criado de experimentação que envia os dados para a API e recebe a classificação. Por fim, mesmo não tendo obtido o melhor modelo de aprendizado, a API possibilita a troca desse modelo, assim tornando possível aprimorar os métodos e substituir posteriormente na API.

Figura 22 – Estado Sem Classificação



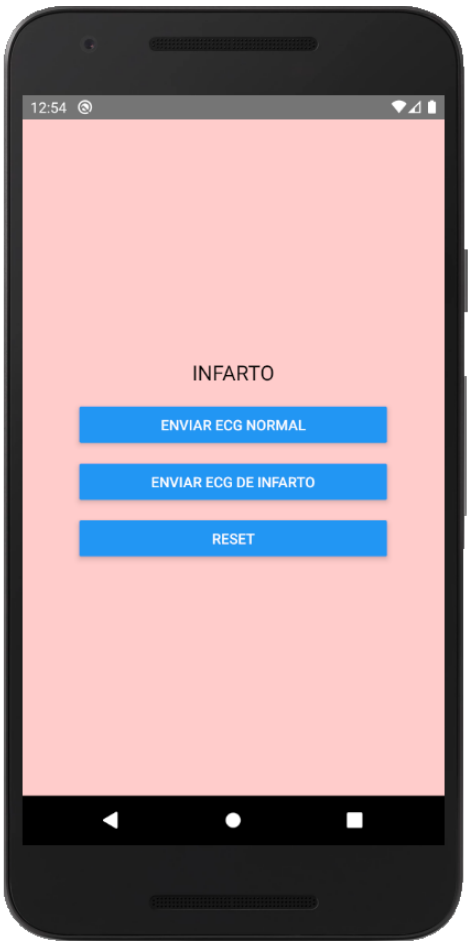
Fonte: O Autor

Figura 23 – Estado Normal



Fonte: O Autor

Figura 24 – Estado Infarto



Fonte: O Autor

4 CONCLUSÕES

Neste trabalho partiu-se da hipótese que seria possível construir uma API para apoiar o desenvolvimento de sistemas de detecção de infarto, dessa forma tornando a API pública e disponibilizando para que qualquer pessoa com o interesse de utilizá-la no desenvolvimento de um aplicativo de celular, smartwatch ou outros dispositivos consiga, e assim usufrua de um classificador de infartos.

Dessa forma, detectar antecipadamente ataques cardíacos é uma tarefa que pode salvar muitas vidas, visto que a descoberta do infarto do miocárdio dentro da primeira hora, conhecida como hora de ouro, é um dos fatores que diminui a mortalidade desse ocorrido. Sistemas preditivos são uma boa solução para a detecção infartos do miocárdio, visto que já possuem uma acurácia elevada. Sendo assim, uma pessoa que não tem a habilidade de avaliar um ECG, pode com o auxílio de um sistema preditivo procurar ajuda médica antes que os sintomas de um ataque cardíaco comecem a se manifestar.

Esse estudo partiu de um processo de revisão sistemática da literatura referente a sistemas preditivos na área de detecção de infartos e doenças cardiovasculares. Através desse estudo, concluiu-se que o método de Árvore de Decisão apresentava ótimos resultados na detecção de ataques cardíacos. Além disso, também se percebeu a vasta utilização do *dataset* PTB ou PTB-XL, que foi o selecionado para dar continuidade ao projeto. E também, foi feita uma apresentação sobre os conceitos de Aprendizado de Máquina e sobre os algoritmos envolvidos. Ademais, foram apresentadas as principais métricas de avaliação de algoritmos de Aprendizado de Máquina.

Vários objetivos específicos foram alcançados ao longo do trabalho em favor de um objetivo geral. O primeiro, foi a identificação dos métodos mais utilizados para a detecção de infartos e a conclusão de que a Árvore de Decisão possui os melhores resultados entre os estudos do estado da arte. Por segundo a busca pelo *dataset* mais utilizado e completo, ficando claro que o PTB-XL foi o mais utilizado entre todos os trabalhos analisados. Em terceiro, foi implementado os métodos de *Random Forest* e Árvore de Decisão, que apesar de possuírem resultados inferiores ao estado da arte ainda foram resultados satisfatórios para a continuidade do projeto e a criação da API. E o quarto e último objetivo também foi alcançado, o qual foi criar e disponibilizar publicamente a API utilizando o modelo do *Random Forest*.

Por conta disso, como todos os objetivos específicos foram alcançados, o objetivo geral que era a utilização de métodos de Aprendizado de Máquina para fins de predição de infarto e a criação da API foram alcançados com sucesso visto que no fim do projeto foram criados vários modelos de Aprendizado de Máquina e selecionado o *Random Forest* para ser utilizado na API que foi disponibilizada na plataforma do Heroku.

4.1 CONTRIBUIÇÕES

Na área da saúde existem muitos trabalhos que desenvolvem modelos preditivos, mas poucos colocam em prática esses modelos, a maioria foca na comparação entre estudos e modelos para atingir os melhores resultados. Neste trabalho o foco foi realmente tornar realidade uma API gratuita que possibilita a classificação de eletrocardiogramas em infarto ou normal. Dessa forma, qualquer um com o interesse em criar um aplicativo preditivo de infartos conseguirá sem ter que desenvolver um novo modelo de classificação, sendo somente necessária a consulta a API. Por conta disso, a contribuição feita para a área foram os modelos desenvolvidos e a interface de acesso a ele, para a criação de produtos, aplicativos e outros estudos.

4.2 TRABALHOS FUTUROS

A área da saúde é extremamente promissora para o desenvolvimento de sistemas a partir de modelos de aprendizado de máquina. Em se tratando de sistemas preditivos, além do diagnóstico antecipado, o próprio estudo das enfermidades pode se beneficiar dos processos baseados em Aprendizado de Máquina.

Para trabalhos futuros é possível criar mais modelos de Aprendizado de Máquina que consigam fazer a predição de um infarto com números diferentes de dimensões, sendo as mais comuns 1 dimensão, 3 dimensões e 12 dimensões.

Como os modelos baseados nas 12 dimensões do eletrocardiograma já foram criados e estudados, o próximo passo seria aprimorar estes modelos, com o objetivo de melhorar sua sensibilidade e especificidade. Além disso, um trabalho futuro relevante seria propor avanços e melhorias na API proposta a partir de modelos gerados com 1 ou 3 dimensões.

Entende-se que esse seria um caminho para tornar os dispositivos que coletam dados em 1 ou 3 dimensões aptos a usarem a API proposta. Outro aspecto importante trata do fato que neste trabalho foi desenvolvido um aplicativo simples para testes. No futuro poderá ser feita a criação de um aplicativo real, que se comunique com a API, e ofereça um benefício específico para os usuários.

REFERÊNCIAS

CARVALHO, A. P. de Leon F. de. **Redes Neurais Artificiais**. 2001. Acessado em: 10/06/2022. Disponível em: <https://sites.icmc.usp.br/andre/research/neural>.

DAI, H.; HWANG, H.-G.; TSENG, V. S. Convolutional neural network based automatic screening tool for cardiovascular diseases using different intervals of ecg signals. **Computer Methods and Programs in Biomedicine**, v. 203, p. 106035, 2021. ISSN 0169-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169260721001103>.

FACELI, K. *et al.* **Inteligência artificial: uma abordagem de aprendizado de máquina**. [S.l.]: LTC, 2021.

FATIMAH, B. *et al.* Efficient detection of myocardial infarction from single lead ecg signal. **Biomedical Signal Processing and Control**, v. 68, p. 102678, 2021. ISSN 1746-8094. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1746809421002755>.

HAO, P. *et al.* Multi-branch fusion network for myocardial infarction screening from 12-lead ecg images. **Computer Methods and Programs in Biomedicine**, v. 184, p. 105286, 2020. ISSN 0169-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169260719314920>.

HE, Z. *et al.* Mfb-lann: A lightweight and updatable myocardial infarction diagnosis system based on convolutional neural networks and active learning. **Computer Methods and Programs in Biomedicine**, v. 210, p. 106379, 2021. ISSN 0169-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169260721004533>.

JAHMUNAH, V. *et al.* Automated detection of coronary artery disease, myocardial infarction and congestive heart failure using gaborcnn model with ecg signals. **Computers in Biology and Medicine**, v. 134, p. 104457, 2021. ISSN 0010-4825. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0010482521002511>.

KATO, R.; PAIVA, V.; IZIDORO, S. Algoritmos genéticos. In: **BIOINFO - Revista Brasileira de Bioinformática e Biologia Computacional**. Alfahelix, 2021. Disponível em: <https://doi.org/10.51780/978-6-599-275326-13>.

KHAN, S. I.; PACHORI, R. B. Derived vectorcardiogram based automated detection of posterior myocardial infarction using fbse-ewt technique. **Biomedical Signal Processing and Control**, v. 70, p. 103051, 2021. ISSN 1746-8094. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1746809421006480>.

LAURETTO, M. S. **Árvores de Decisão**. [S.l.]: EACH-USP, 2010.

LEUNG, K. M. Naive bayesian classifier. **Polytechnic University Department of Computer Science/Finance and Risk Engineering**, v. 2007, p. 123–156, 2007.

LIU, J. *et al.* Automated detection and localization system of myocardial infarction in single-beat ecg using dual-q tqwt and wavelet packet tensor decomposition. **Computer Methods and Programs in Biomedicine**, v. 184, p. 105120, 2020. ISSN 0169-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169260719313574>.

LONGO, D. L. *et al.* In: **Medicina interna de Harrison**. [S.l.: s.n.], 2013.

LUGER, G. Inteligência artificial. Pearson Universidades, v. 1, n. 1, p. 632, 2013.

MONARD, M. C.; BARANAUSKAS, J. A. Conceitos sobre aprendizado de máquina. **Sistemas inteligentes-Fundamentos e aplicações**, Manole, 2003.

ONGSULEE, P. Artificial intelligence, machine learning and deep learning. In: **2017 15th International Conference on ICT and Knowledge Engineering (ICT&KE)**. [S.l.: s.n.], 2017. p. 1–6.

PRATI JOSé AUGUSTO BARANAUSKAS, M. C. M. R. C. Extração de informações padronizadas para a avaliação de regras induzidas por algoritmos de aprendizado de máquina simbóllico. 2001. Disponível em: <https://web.icmc.usp.br/SCATUSU/RT/BIBLIOTECA_113_RT_145.pdf>.

PRATI, R. C.; BARANAUSKAS, J. A.; MONARD, M. C. Padronizaçao da sintaxe e informaçoes sobre regras induzidas a partir de algoritmos de aprendizado de máquina simbólico. **Revista Eletrônica de Iniciaçao Cientıfica**, v. 2, n. 3, p. 21, 2002.

SAMPAIO, R. F.; MANCINI, M. C. Systematic review studies: a guide for careful synthesis of the scientific evidence. **Brazilian Journal of Physical Therapy**, SciELO Brasil, v. 11, n. 1, p. 83–89, 2007.

SOUTO, M. D. *et al.* Técnicas de aprendizado de máquina para problemas de biologia molecular. **Sociedade Brasileira de Computaçao**, v. 1, n. 2, 2003.

WAGNER, P. *et al.* **PTB-XL, a large publicly available electrocardiography dataset**. PhysioNet, 2020. Disponível em: <https://physionet.org/content/ptb-xl/1.0.1/>.

____. Ptb-xl, a large publicly available electrocardiography dataset. 2020. Disponível em: <https://iphome.hhi.de/samek/pdf/WagSciData20.pdf>.

XIONG, P. *et al.* Localization of myocardial infarction with multi-lead ecg based on densenet. **Computer Methods and Programs in Biomedicine**, v. 203, p. 106024, 2021. ISSN 0169-2607. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0169260721000997>.