Aplicação de Aprendizado de Máquina na detecção de convulsões epiléptica em Sinais de EEG

Vanessa Machado Silva

Universidade Federal de Uberlândia
Faculdade de Computação
Bacharelado em Sistemas de Informação

Monte Carmelo - MG
2025

Vanessa Machado Silva

Aplicação de Aprendizado de Máquina na detecção de convulsões epiléptica em Sinais de EEG

Trabalho de Conclusão de Curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Sistemas de Informação.

Área de concentração: Sistemas de Informação

Orientador: Professora Daniele Carvalho Oliveira

Monte Carmelo - MG
2025

Agradecimentos

Agradeço à minha mãe, Dagmar, que sempre me incentivou a focar nos estudos e me apoiou ao longo de toda a graduação. Às minhas irmãs, Amanda e Valéria, que são grandes inspirações para mim.

Agradeço à minha orientadora, Prof. Danielle, pela paciência, orientação e apoio durante o desenvolvimento deste trabalho.

Agradeço também à Faculdade de Computação (FACOM) e à Universidade Federal de Uberlândia (UFU) pela oportunidade de cursar Sistemas de Informação e por todas as experiências proporcionadas.

Também sou grato aos meus amigos do curso, que tornaram a jornada mais leve, divertida.

"O computador é incrivelmente rápido, preciso e burro. O ser humano é incrivelmente lento, impreciso e brilhante. Juntos, são poderosos além da imaginação."

(Albert Einstein)

Resumo

A epilepsia é uma doença neurológica que afeta cerca de 50 milhões de pessoas no mundo, sendo caracterizada pela ocorrência de crises epilépticas. O uso de técnicas de inteligência artificial tem-se mostrado uma ferramenta promissora na detecção e classificação de sinais neurais. Neste trabalho, foram utilizados dados de eletroencefalograma (EEG) de 22 pacientes, com aproximadamente 23 canais por registro. Devido à longa duração das gravações e à curta duração das crises, foi necessário balancear as classes por meio do recorte de períodos específicos de cada estado. Os sinais foram filtrados e segmentados em janelas de 15 segundos para extração e seleção de características, com o objetivo de reduzir informações irrelevantes à predição. Na etapa de classificação, foram realizados diversos testes para definir os melhores parâmetros e otimizar o desempenho dos modelos. O objetivo foi identificar os períodos de transição das crises epilépticas, incluindo os estados ictal, interictal e pré-ictal. Foram avaliados algoritmos de aprendizado de máquina, como *Random Forest* (RF), *Decision Tree* (DT), *K-Nearest Neighbors* (KNN), *Support Vector Machine* (SVM) e *Logistic Regression* (LR), além de modelos de aprendizado profundo, como *Convolutional Neural Network* (CNN) e *Long Short-Term Memory* (LSTM). Os melhores resultados foram obtidos na classificação das classes interictal e ictal, com destaque para a CNN, que atingiu 95,27% de acurácia, seguida pela RF (94%) e pela LSTM (93,32%). A LR obteve 91%, a DT 89%, o SVM 88,3% e o KNN 83%. Os resultados evidenciam o bom desempenho dos métodos propostos, especialmente os modelos de aprendizagem profunda, além de permitirem uma comparação clara entre as diferentes abordagens.

Palavras-chave: Epilepsia, EEG, Aprendizado de maquina, Aprendizado profundo, KNN.

Lista de ilustrações

Lista de tabelas

Lista de siglas

CNN Redes Neurais Convolucionais - *Convolutional Neural Network*

DL Apredizagem Profunda - *Deep Learning*

DT Árvores de Decisão - *Decision Trees*

ECG Eletrocardiograma

EEG eletroencefalograma

IA Inteligência Artificial

KNN K-ésimo Vizinho mais Próximo - *K-nearest neighbors*

LR Regressão Logística - *Logistic regression*

LSTM memória de Curto Longo Prazo - *Long short-term memory*

ML Aprendizado de máquina - *Machine learning*

MLP Perceptron multicamadas - *Multilayer Perceptron*

ReLU Função Linear Retificada - *Rectified Linear Unit*

RF Floresta aleatória - *Random forest*

RFE *Recursive Feature Elimination*

RNA Redes Neurais Artificiais

RNN *Rede Neural Recorrente - Recurrent Neural Network*

SVM Máquina de vetores de suporte - *Support Vector Machine*

VNS Estimulação do Nervo Vago

Sumário

Capítulo 1

Introdução

A epilepsia é um distúrbio do sistema nervoso central que afeta aproximadamente 50 milhões de pessoas em todo o mundo, de acordo com a (ONU News, 2022). O diagnóstico dessa condição em pacientes é geralmente realizado por meio da coleta da atividade elétrica cerebral, utilizando ferramentas como o Eletroencefalograma (EEG). Dado o impacto significativo que a epilepsia pode ter na qualidade de vida do paciente, o diagnóstico precoce é crucial para o início do tratamento. No entanto, a análise e classificação desses sinais podem ser um processo demorado e desafiador.

Existem dois tipos principais de epilepsia: a focal, que ocorre devido a uma descarga elétrica focada em uma área específica do cérebro, afetando cerca de 36% dos pacientes; e a epilepsia generalizada, que afeta ambos os lados do cérebro simultaneamente e atinge aproximadamente 62% dos pacientes, conforme relatado por (GASTAUT et al., 1975) (LIMA, 2005). O tratamento é realizado por meio do uso de medicamentos, os quais são capazes de controlar as descargas elétricas anormais, ou por meio de cirurgia, dispositivos ou mudanças alimentares em casos nos quais os medicamentos não conseguem conter as crises, o que ocorre em cerca de 30% dos casos (JOBST et al., 2019).

O diagnóstico da epilepsia é geralmente realizado por meio do EEG, um exame que registra a atividade elétrica espontânea do cérebro. Esse método é amplamente utilizado devido à sua acessibilidade econômica e alta capacidade de gravação, além de ser, em geral, um procedimento não invasivo (DANTAS et al., 2005) (LIMA, 2005). Durante o procedimento, eletrodos são colocados em posições pré-definidas no couro cabeludo para a captura dos sinais, os quais são registrados em gráficos analógicos. A análise dessas gravações permite identificar anormalidades, como picos de ondas focais ou generalizadas, que podem ser indicativas do diagnóstico de epilepsia (ALMEIDA, 2005).

Para identificar a epilepsia, é fundamental realizar uma análise dos dados do EEG, a fim de identificar os sinais característicos da doença. Essa análise requer um conhecimento profundo por parte dos profissionais, pois o EEG pode capturar diversas atividades, como piscar de olhos ou movimentos dos braços, que podem complicar a interpretação dos resultados. Os desafios enfrentados pelos processadores de sinais são significativos, uma vez

que o EEG apresenta uma baixa relação sinal-ruído, não linearidade e não estacionariedade (FREEMAN; QUIROGA, 2012).

O uso de aprendizado de máquina é amplamente difundido na automatização da classificação de dados, incluindo a detecção de crises de epilepsia, o que pode aprimorar tanto o diagnóstico quanto o tratamento dessa condição. Nesse sentido, esta pesquisa propõe o uso da aprendizagem de máquina para realizar a comparação de diferentes algoritmos, com o objetivo de automatizar e detectar os períodos de atividades de epilepsia no cérebro. A aprendizagem de máquina é uma abordagem que permite que os sistemas analisem dados e construam modelos analíticos de forma automatizada, capacitando-os a identificar padrões e tomar decisões com base nesses dados (LIMA, 2005).

1.1 Desafios e Objetivos da Pesquisa

A análise e interpretação de sinais de eletroencefalograma (EEG) para o diagnóstico de epilepsia é um desafio em razão de possuírem dados complexos, volumosos, onde o diagnóstico manual de crises epilépticas através do EEG demanda tempo e conhecimentos especializados. Embora técnicas de aprendizado de máquina tenham sido aplicadas em diversas áreas médicas para automatizar a análise de dados, ainda há uma lacuna em soluções eficazes para a detecção de epilepsia, que sejam acessíveis e adequadas para análise precisa de EEG em contextos clínicos.

Deste modo, este trabalho tem como objetivo investigar e avaliar técnicas de aprendizado profundo e de máquina na detecção autônoma de crises epilépticas. Para isso, serão utilizados modelos de Redes Neurais Convolucionais e métodos de classificação supervisionados, como Árvore de Decisão, k-vizinhos mais próximos, Regressão Logística, Floresta Aleatória e Máquina de Vetores de Suporte.

1.2 Hipótese

Esta pesquisa parte da hipótese de que diferentes algoritmos de aprendizado de máquina apresentam desempenhos distintos na detecção de crises epilépticas a partir de sinais de eletroencefalograma (EEG), e que é possível identificar, por meio de avaliação comparativa, quais modelos oferecem maior precisão e eficácia na classificação desses sinais.

1.3 Organização da Monografia

Esta monografia está organizada em quatro capítulos. O Capítulo 1 apresenta a introdução, os desafios e os objetivos relacionados à detecção de crises epilépticas em sinais de EEG. O Capítulo 2 contém a Fundamentação Teórica, abordando os principais temas

da pesquisa. Inicialmente, contextualiza a relação entre epilepsia e eletroencefalograma (EEG), explicando como o EEG é utilizado para identificar padrões associados à epilepsia. Em seguida, explora a aplicação do aprendizado de máquina, destacando algoritmos de inteligência artificial voltados à análise de EEG para a detecção automática de crises epilépticas. No Capítulo 3, são apresentadas a metodologia adotada, os experimentos realizados e os resultados obtidos. Por fim, o Capítulo 4 traz a conclusão do trabalho, suas principais contribuições, trabalhos futuros, destacando suas contribuições e limitações.

Capítulo 2

Fundamentação Teórica

Neste capítulo, são apresentados os principais conceitos relacionados à epilepsia, incluindo seus tipos, tratamentos e o papel do Eletroencefalograma no auxílio ao diagnóstico da doença. Além disso, são discutidos os fundamentos da aprendizagem de máquina e profunda, abordando suas etapas básicas e os principais algoritmos de classificação, com ênfase em seu funcionamento, aplicações, vantagens e limitações.

2.1 Epilepsia

A epilepsia é uma das doenças neurológicas mais comuns, afetando cerca de 50 milhões de pessoas no mundo (ONU News, 2022). As crises epilépticas são caracterizadas pelo episódio de atividade elétrica anormal no cérebro, onde há ocorrência transitória de sinais decorrentes de atividade anormal, excessiva e síncrona. A crise pode ser causada por doenças neurológicas, como meningite ou traumas cranianos. Portanto, uma única crise epiléptica não é suficiente para diagnosticar epilepsia. A convulsão de epilepsia é marcada pela transição entre estados, assim, durante os eventos epilépticos, há quatro estados (FERREIRA et al., 2023) (MONTENEGRO et al., 2022):

- **Período ictal** é caracterizado por um evento de atividade epileptiforme rítmica, podendo durar de segundos a minutos.
- **Período interictal** considerado o período entre as crises epilépticas, quando o cérebro não está apresentando atividade epiléptica evidente.
- **Período Pré-ictal** refere-se ao período imediatamente anterior à crise epiléptica, onde pode haver mudanças sutis na atividade cerebral, ou até mesmo sintomas como sensações visuais, auditivas, gustativas ou emocionais.
- **Período Pós-ictal** é o período logo após a crise epiléptica. Durante essa fase, o paciente pode experimentar confusão, sonolência, dores de cabeça ou outros sintomas neurológicos transitórios.

As crises podem ser classificadas em dois tipos principais: focais (parciais) e generalizadas. As crises focais ocorrem quando a atividade elétrica anormal começa em uma área específica do cérebro, enquanto as crises generalizadas envolvem atividade elétrica anormal em ambos os hemisférios cerebrais desde o início. As crises focais podem ser subdivididas em categorias simples, complexas e as focais que evoluem para crises secundárias generalizadas, dependendo dos sintomas e do grau de comprometimento da consciência (FERREIRA et al., 2023) (YACUBIAN; CONTRERAS-CAICEDO; RíOS-POHL, 2014):

- **Crises Focais Simples:** A pessoa permanece consciente durante o episódio. Geralmente envolve sensações estranhas, movimentos involuntários, alterações na percepção sensorial ou emocional, entre outros.
- **Crises Focais Complexas:** Há perda temporária da consciência. A pessoa pode realizar movimentos automáticos, parecer confusa ou desorientada, e não se lembrar do episódio depois que ele passa.
- **Crises Focais que evoluem para crises secundária Generalizadas:** Iniciam-se como crises parciais, simples ou complexas, e evoluem para crises generalizadas.

As crises generalizadas também podem ser subdivididas em dois grupos: convulsivas e não convulsivas. As crises não convulsivas ocorrem quando há alteração da consciência, com baixa manifestação de fenômenos motores. Já nas crises convulsivas generalizadas, pode ou não haver alteração da consciência, além de sintomas motores predominantes durante o episódio (YACUBIAN; CONTRERAS-CAICEDO; RíOS-POHL, 2014).

O diagnóstico da epilepsia é feito por meio de diversos exames. O exame clínico é realizado para analisar o histórico do paciente, incluindo dados sobre as características das crises, como a duração e a frequência dos episódios. Além disso, exames complementares, como o EEG, são fundamentais. A análise do EEG permite detectar padrões anormais de atividade cerebral, típicos de crises epilépticas, sendo crucial para a confirmação do diagnóstico. Em alguns casos, exames de neuroimagem, como a ressonância magnética, são também utilizados para investigar possíveis lesões ou anormalidades estruturais no cérebro que possam estar associadas à epilepsia (LIMA, 2005).

2.2 Eletroencefalograma

O EEG foi inventado por Hans Berger em 1924, consiste em um exame que registra a atividade elétrica espontânea do cérebro. Por meio dele, é possível visualizar e monitorar a atividade elétrica cerebral, registrando as diferenças de potencial elétrico entre pontos de escalpe. É um método comumente utilizado para diagnosticar e monitorar pacientes com doenças neurológicas, como a epilepsia, uma vez que pode ser usado para detectar anomalias (ALMEIDA, 2005).

O EEG utiliza eletrodos ao longo do couro cabeludo de forma não invasiva, seguindo o sistema internacional 10-20 mostrado na Figura 1. Esta técnica fornece uma resolução espacial (nível de detalhamento) relativamente baixa, devido ao fato de os eletrodos não conseguirem alcançar as faces internas e inferiores dos hemisférios cerebrais. Por outro lado, o EEG apresenta uma resolução temporal alta, o que permite a medição em milissegundos (FERREIRA et al., 2023).

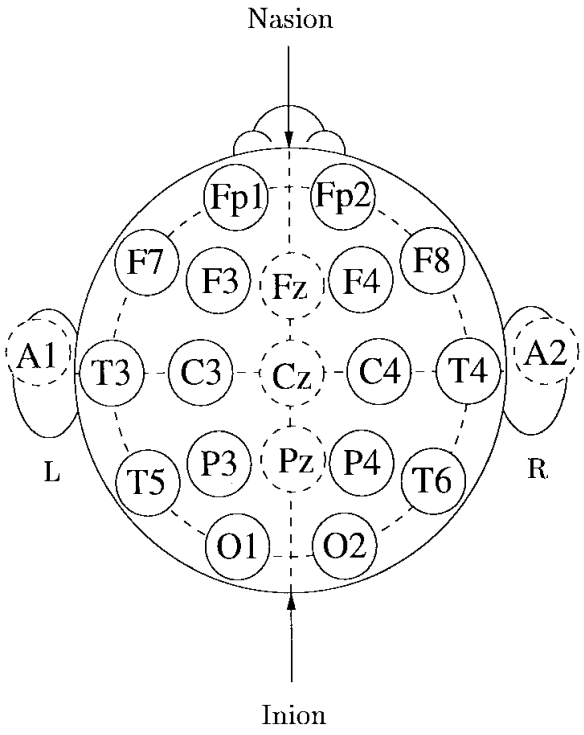

Figura 1 – Sistema Internacional 10-20. Fonte: (WARD et al., 1999).

2.2.1 Eletrodos

O exame de EEG consiste na utilização de eletrodos que são colocados sobre o couro cabeludo na posição correta, com base no sistema Internacional 10-20. As posições devem ser determinadas com base em medidas específicas do crânio, seguindo o particionamento em proporções definidas.

A letra designa a região cerebral coberta por este eletrodo, e o número, sua lateralização. Ou seja: Fp para fronto-polar, F para frontal, C para central, P para parietal, T para temporal e O para occipital. Para diferenciar hemisférios cerebrais, os números pares ficam do lado direito (Fp2, F4, F8, C4, P4, T4, T6 e O2) e os ímpares, no esquerdo (Fp1, F3, F7, C3, P3, T3, T5 e O1). Eletrodos da linha média são identificados pela letra Z, de zero (Fz, Cz e Pz). Eletrodos auriculares são denominados A1 (à esquerda) e A2 (à direita) (FERREIRA et al., 2023).

O procedimento normalmente se inicia com o paciente acordado e relaxado de modo que ele não realize nenhum movimento que possa interferir na coleta da atividade cerebral. O EEG é comumente usado uma vez que se trata de um exame não invasivo que permite o registro espontâneo da atividade neural. Os sinais de EEG mostram as ondas elétricas do cérebro. As oscilações das frequências são correlacionadas com várias funções, como percepção, movimento, processos cognitivos, aprendizagem e memória. Elas são classificadas em tipos de sub-bandas de acordo com a frequência: delta, theta, alfa, beta e gama, conforme descrito por (MONTENEGRO et al., 2022) (BAŞAR et al., 2001):

- Delta (δ): frequência de 0,5-3,5 Hz; são ondas associadas ao estado de sono profundo ou em casos de lesões cerebrais.
- Theta (θ): frequência de 3,5-7,5 Hz; são ondas associadas ao estado de sono leve, inicial do sono ou em meditação;
- Ondas Alfa (α): frequência de 7,5-13 Hz; são ondas associadas ao estado em que a pessoa esta acordada, mas com os olhos fechados em estados de relaxamento;
- Ondas Beta (β): frequência de 13-30 Hz; são ondas associadas a estados de atenção, concentração e atividade mental;
- Ondas Gama (γ): frequência de 30-70 Hz; são ondas associadas relacionadas à atividades cognitivas complexas.

2.3 Eletroencefalograma e epilepsia

O EEG é amplamente utilizado no diagnóstico da epilepsia por ser um exame de baixo custo e de fácil realização. Com ele, é possível definir o tipo de crise e auxiliar na escolha do medicamento para o tratamento. Também é possível identificar as áreas cerebrais envolvidas durante as crises, o que auxilia na definição do foco cirúrgico, quando necessário (FERREIRA et al., 2023). O EEG é uma ferramenta importante no diagnóstico da epilepsia, alcançando níveis de especificidade entre 50% e 92%. Esses níveis podem aumentar com a repetição do exame, registros de sono e o uso de técnicas de ativação (OLIVEIRA; ROSADO, 2004).

Durante as crises, é possível visualizar alterações eletrográficas associadas à epilepsia, que podem ser observadas nos períodos ictal, pré-ictal e interictal. O EEG normalmente não descarta o diagnóstico de epilepsia, nem determina a epilepsia com base em uma anormalidade no exame. Exames normais em pessoas com epilepsia podem ocorrer devido a focos epilépticos distantes dos eletrodos, tempo de registro insuficiente ou uso de medicação que oculta as atividades epileptiformes. O EEG tem um uso bem definido no contexto da doença de episódios epilépticos, tanto convulsivo quanto não convulsivo (MONTENEGRO et al., 2022), (FERREIRA et al., 2023).

O uso de métodos de ativação pode ser indicado em casos de exame normal, como privação de sono, prova de hiperpneia e fotoestimulação intermitente, além de registros prolongados e até mesmo a implantação de eletrodos profundos. O técnico desempenha um papel fundamental na realização do exame, que inclui a coleta de informações do paciente, como medicações em uso, motivo da solicitação do exame, e comportamento do paciente antes, durante e depois das crises. Ele deve monitorar todo o registro e anotar o estado do paciente, como sono, confusão mental, ansiedade e posição da cabeça. Um médico é responsável por interpretar os resultados coletados pelo técnico e traçar o melhor procedimento (FERREIRA et al., 2023).

2.4 Aprendizado de máquina

O Aprendizado de máquina (ML) é um campo dentro da Inteligência Artificial (IA) cujo objetivo é desenvolver métodos computacionais capazes de aprender e operar de forma autônoma. Dessa forma, os algoritmos buscam generalizar ao máximo o aprendizado. A generalização consiste em produzir respostas satisfatórias a partir de conjuntos de dados, relacionando entradas e saídas para realizar previsões com base nos padrões observados. O ML pode ser classificado em três categorias principais (LIMA, 2014) (FACELI et al., 2021) (GABRIEL, 2022) (GéRON, 2021):

- **Aprendizado Supervisionado:** Os algoritmos de aprendizado supervisionado trabalham com dados previamente rotulados, ou seja, cada entrada no conjunto de dados está associada a uma saída conhecida. Nesse contexto, o algoritmo é treinado com esses pares de entrada-saída, permitindo que aprenda a prever corretamente a saída com base nas novas entradas fornecidas. Alguns dos algoritmos comuns utilizados para aprendizado supervisionado incluem a Floresta Aleatória, K-Vizinhos Mais Próximos, Máquina de Vetores de Suporte, Regressão Logística e Árvore de Decisão.

- **Aprendizado não Supervisionada:** Os algoritmos de aprendizagem não supervisionada não possuem um conjunto de dados de saída pré-rotulado, contando apenas com as entradas. Esses algoritmos buscam padrões e similaridades em grandes volumes de dados, permitindo identificar grupos similares entre os dados e/ou similaridades nos grupos existentes. Os algoritmos podem ser divididos em algoritmos de transformação e de agrupamento. Algoritmos de transformação são utilizados para melhorar o aprendizado do algoritmo e facilitar a interpretação humana, criando uma nova e melhor representação dos dados. Algoritmos de agrupamento (*clustering*) agrupam dados com características similares com base em critérios pré-estabelecidos, permitindo encontrar padrões entre os dados. Os métodos de agrupamento podem ser baseados na distância geométrica entre pontos, distribuição

estatística ou densidade de pontos em áreas específicas do conjunto de dados. Por fim, a abordagem semis-supervisionada utiliza uma fração de dados rotulados para melhorar o desempenho.

- **Aprendizado por Reforço:** A aprendizagem por reforço é um processo de tentativa e erro, onde um agente recebe recompensa ou punição de acordo com o resultado de suas ações, sendo o objetivo do agente receber o maior número possível de recompensas. Os agentes precisam observar, interagir e modificar o ambiente ao longo do tempo, de maneira transiente.

2.4.1 Etapas Básicas para Algoritmos de aprendizado de máquina

Os algoritmos de classificação são uma abordagem de aprendizagem supervisionada, na qual os dados de entrada são classificados com base em rótulos previamente conhecidos nos dados de saída. Esses dados podem pertencer a duas classes (classificação binária) ou a mais de duas classes (classificação multiclasse) (FACELI et al., 2021) (LIMA, 2014). Esse método segue algumas etapas, apresentadas na Figura 2 e detalhadas em cada etapa a seguir:

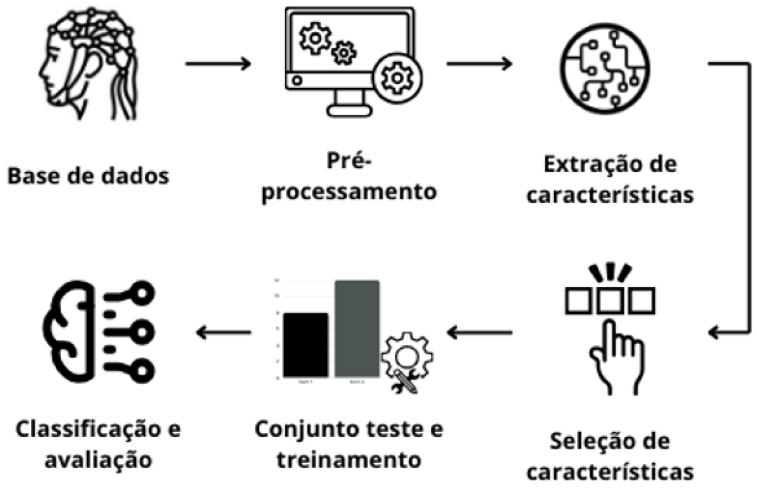

Figura 2 – Etapas para o desenvolvimento de algoritmos de aprendizado de máquina. Fonte: Elaborada pelo autor.

A coleta ou escolha da base de dados e análise são etapas cruciais para o desenvolvimento do modelo de aprendizado de máquina, pois sua quantidade e qualidade determinam quão preditivos ele será, além disso é importante entender os tipos de variáveis que a base de dados possui para serem trabalhadas (qualitativas ou quantitativas), pois a categoria das variáveis permite a realização de escolhas de operações estatísticas. Compreender

os dados pode ajudar a resolver problemas encontrados, uma vez que é possível identificar problemas similares já solucionados para esse tipo de dado (ESCOVEDO; KOSHIYAMA, 2020) (LIMA, 2014) (FACELI et al., 2021).

A preparação ou pré-processamento é uma fase em que pode realizar a aquisição, exploração, limpeza e transformação dos dados brutos. A aquisição de dados é utilizada para determinar quais informações são necessárias para análise. A exploração avalia a qualidade e a distribuição das amostras, ajudando a entender melhor o conjunto de dados e a realizar uma análise mais precisa. A limpeza melhora a qualidade, removendo informações de baixa qualidade ou corrigindo erros. A transformação de dados inclui a formatação, a agregação e o enriquecimento dos dados. O objetivo do pré-processamento é identificar e tratar problemas como artefatos corrompidos ou faltantes que poderiam influenciar na fase de treinamento do modelo de classificação (BATISTA et al., 2003) (ESCOVEDO; KOSHIYAMA, 2020).

A extração e a seleção de características são dois processos que têm como objetivo preservar informações relevantes, além de realizar a redução da dimensionalidade dos dados. Para que o sistema consiga aprender, é necessário que os dados tenham quantidade suficiente e uma boa qualidade, pois, assim o sistema poderá detectar melhor os padrões para realizar suas previsões. O processo de extração de características envolve criar novos dados a partir dos disponíveis, por meio da combinação deles, como média, desvio padrão, moda entre outras medidas estatísticas aplicadas de acordo com as variáveis da base de dados (quantitativas e/ou qualitativas) (GéRON, 2021).

A separação dos dados em conjuntos de teste e treinamento é feita para avaliar o desempenho real do modelo. Caso o treinamento seja realizado com todos os dados disponíveis, isso poderia levar ao *overfitting*[1]. Dessa forma, a divisão em conjuntos é realizada para medir o desempenho real do modelo. A divisão é feita em dois grupos: um de treinamento e outro de teste, geralmente reservando cerca de 20 a 30% dos dados para teste e utilizando o restante para treinamento (GéRON, 2021).

A escolha do modelo deve ser feita de acordo com o problema, por exemplo, caso se pretenda realizar a previsão de valores, o modelo será de regressão; caso se queira classificar categorias, o modelo será de classificação. Os modelos mais utilizados são os de classificação binária, que fazem previsões entre duas opções; e classificação multiclasse, que fazem previsões para múltiplas classes; e modelos de regressão, que são usados para prever um valor numérico.

Para avaliar o desempenho utilizam-se algumas métricas, como a análise do número de predições corretas e erradas. Para o exemplo de classes bínarias podem-se definir os valores de (GéRON, 2021) (FACELI et al., 2021):

- Verdadeiro positivo (TP): Prevê corretamente a classe positiva.

[1] Overfitting refere-se ao ajuste excessivo aos dados de treinamento, onde o modelo fornece previsões precisas para os dados de treinamento, mas não para dados novos.

- ❏ Verdadeiro negativo (TN): Prevê corretamente a classe negativa.
- ❏ Falso positivo (FP): Prevê incorretamente a classe positiva.
- ❏ Falso negativo (FN): Prevê incorretamente a classe negativa.

A soma de todos os valores na matriz de confusão deve ser igual ao número total de amostras utilizadas no conjunto de teste. A Tabela 1 permite visualizar as taxas de erro e acerto das classes reais (GéRON, 2021) (FACELI et al., 2021).

	Valores Previstos	
Valores reais	Positivo (1)	Negativo (0)
Positivo (1)	TP	FP
Negativo (0)	FN	TN

Tabela 1 – Matriz de Confusão. Fonte: Elaborada pelo autor.

A matriz de confusão é avaliada observando os valores da diagonal principal. Quanto maiores os valores na diagonal principal, melhor o desempenho do algoritmo.

Outra medida de desempenho muito utilizada é a acurácia, relaciona o total de acertos com o total de dados pertencentes ao grupo de testes (GéRON, 2021) (FACELI et al., 2021). A fórmula dela é definida como:

$$Acuracia = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

Apesar de a acurácia ser amplamente utilizada, ela nem sempre é a melhor métrica de desempenho, pois pode ser enganosa em conjuntos de dados desbalanceados, onde uma classe pode ser dominante. Para obter uma visão mais completa do desempenho, é possível alinhar a acurácia com outras métricas, como a precisão e o *recall* (GéRON, 2021) (FACELI et al., 2021). Precisão é a fração dos resultados positivos corretos em relação a todos os resultados positivos produzidos:

$$Precisao = \frac{TP}{TP + FP} \tag{2}$$

Essa métrica permite medir a proporção de predições positivas que estavam realmente corretas.

O *recall* é definido como a fração dos resultados positivos corretos em relação a todos os resultados positivos reais:

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

É utilizado para medir quantos dos resultados positivos em um conjunto foram identificados corretamente.

Nesta etapa, é realizado o aprimoramento dos parâmetros. É importante identificar os parâmetros que afetam as métricas de desempenho, como a acurácia, entre outros. A

linha de aprendizagem é analisada com base nesses parâmetros anteriores para ajustar e melhorar o modelo.

Por fim, é realizada a previsão, onde o modelo treinado é capaz de responder às perguntas para as quais foi desenvolvido.

2.4.2 Máquina de Vetores de Suporte

A Máquina de vetores de suporte (SVM) é um algoritmo de aprendizado supervisionado, amplamente utilizado para problemas de classificação, mas que também pode ser aplicado em regressão. Esse método foi desenvolvido por Vapnik e seus colegas e formalizado em (VAPNIK; GOLOWICH; SMOLA, 1996). Ela é eficaz para trabalhar com dados lineares e não lineares, além de ser capaz de lidar bem com dados de alta dimensionalidade. O princípio fundamental da SVM é encontrar o hiperplano que garante a maior margem de separação possível entre as classes (FACELI et al., 2021) (GéRON, 2021).

Em problemas linearmente separáveis, é possível definir uma fronteira clara entre as classes, criando a maior margem possível, evitando assim as violações de margem, entre os pontos mais próximos de cada classe, conhecidos como vetores de suporte (FACELI et al., 2021). Assim, o propósito do algoritmo neste caso é definir o plano de decisão, identificando a posição do hiperplano que melhor separa as amostras. Para isso, o modelo considera os vetores de suporte, que são os pontos mais próximos ao plano de separação, e determina a margem máxima possível entre as classes. O processo de otimização envolve a maximização dessa margem enquanto permite a aplicação de um fator de penalidade, controlado pelo parâmetro C, para lidar com possíveis erros de classificação. Dessa forma, obtém-se o separador ótimo, garantindo a melhor generalização do classificador SVM Linear (GAO; SUN, 2010).

A dimensão do hiperplano é dada por d dimensões: um hiperplano possui $d - 1$ dimensões. Na Figura 3 é apresentada a separação dos hiperplanos. Esse hiperplano maximiza a distância entre os pontos de ambas as classes, garantindo a maior separação possível (FACELI et al., 2021) (GéRON, 2021).

Para dados não lineares, como ilustrado na Figura 4, a SVM utiliza a técnica de kernel, para transformar os dados para um espaço de características de maior dimensão, onde é possível traçar um hiperplano linear para separar eficientemente as classes (FACELI et al., 2021) (GéRON, 2021). Esse mapeamento é realizado por meio de funções que obedecem à condição de Mercer, garantindo que possam ser interpretadas como dados lineares. Assim, usar a função de *kernel*, o modelo otimiza a margem entre as classes no novo espaço. Os *kernels* polinomial, o radial base (RBF) e o sigmoide são os mais utilizados (GAO; SUN, 2010).

A Máquina de Vetores de Suporte (SVM) é amplamente utilizada em diversas áreas, como detecção (KUMAR; SINGH; KUMAR, 2019) e reconhecimento facial (CHEN; HAOYU, 2019), diagnóstico de doenças (VADALI; DEEKSHITULU; MURTHY, 2019), reco-

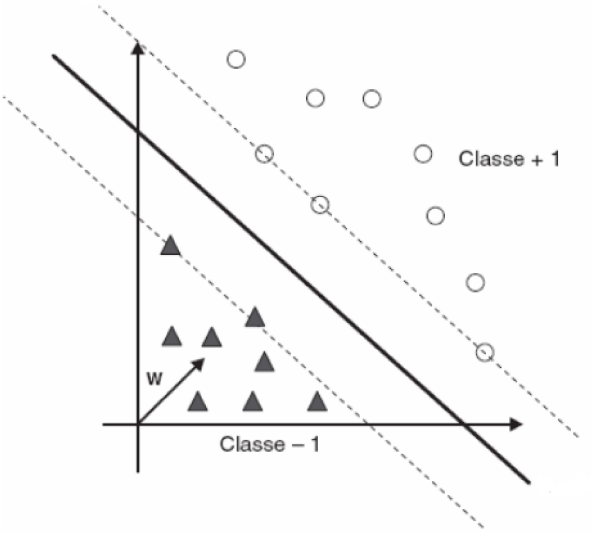

Figura 3 – Ilustração de hiperplanos canônicos e separador. Fonte: (FACELI et al., 2021)

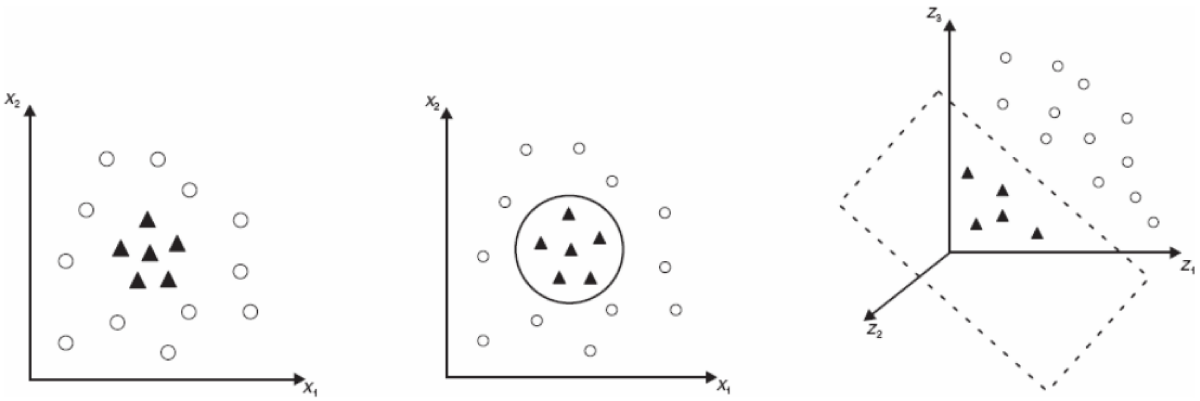

Figura 4 – Transformação de dados não lineares para um espaço de características. Fonte: (FACELI et al., 2021).

nhecimento de texto (ISMAYILOV; MAMMADOV, 2019), análise de sentimentos (ABRO et al., 2020) e detecção de convulsões a partir de sinais de EEG (SAPUTRO et al., 2019), entre outras aplicações. O trabalho de (ABDULLAH; ABDULAZEEZ, 2021) apresenta uma revisão sistemática sobre os diferentes campos que utilizam o classificador SVM, demonstrando que, em alguns casos, ele superou outros métodos de classificação, como o *Naïve Bayes*, Regressão Logística (LR) e Floresta aleatória (RF), especialmente nas áreas de reconhecimento facial e diagnóstico de doenças.

Apesar de sua eficiência, a SVM apresenta algumas limitações. Uma delas é a necessidade de otimização dual durante o treinamento, o que pode aumentar a complexidade computacional, assim para grandes base de dados o treinamento pode se tornar um processo longo e lento. Outro problema seria em relação a classificação de dados desbalanceados a SVM não lida bem com conjuntos de dados desbalanceados, devido a dificuldade de conseguir uma separação do hiperplano neste caso (CERVANTES et al., 2020). Além

disso, a SVM não lida diretamente com dados categóricos, sendo necessário convertê-los em valores numéricos. Outra limitação é que, embora seja ideais para problemas de classificação binária, quando aplicadas a problemas com múltiplas classes, exigem estratégias de decomposição, o que pode aumentar o custo computacional (FACELI et al., 2021).

Mesmo com essas limitações, a SVM tem excelente capacidade de generalização, especialmente em problemas de alta dimensão. O uso de *kernels* permite que ela lide eficientemente com separações não lineares, e sua natureza determinística garante resultados consistentes, independentemente da ordem de apresentação dos dados (FACELI et al., 2021).

2.4.3 K-vizinhos mais próximos

O K-ésimo Vizinho mais Próximo (KNN) supervisionado é um algoritmo usado para classificação e regressão. Ele é amplamente utilizado em tarefas de classificação, devido à sua simplicidade e facilidade de aplicação. O KNN classifica novos exemplos, numéricos ou categóricos, com base na distância deles em relação aos exemplos mais próximos do conjunto de treinamento (FACELI et al., 2021) (Scikit-learn Developers, 2024).

O KNN opera com base na similaridade dos dados, levando em consideração os rótulos dos K vizinhos mais próximos no conjunto de dados de treinamento. Para classificar uma nova amostra, o algoritmo segue os seguintes passos (CADENAS et al., 2018) (FAN et al., 2019):

1. Escolher do valor de K.
2. Calcular as distâncias Euclidianas entre o ponto a ser classificado e cada ponto de dados no conjunto de treinamento.
3. Organizar as distancias em ordem crescente.
4. Encontrar os K vizinhos mais próximos de acordo com o valor de K.
5. Calcular qual classe os k vizinhos mais próximos pertencem.
6. Realizar uma votação majoritária para determinar a classe dos K vizinhos mais próximos.

A escolha do valor de K é crucial para o desempenho do algoritmo, pois define o número de vizinhos considerados. Em geral, recomenda-se que k seja um número ímpar para evitar empates nas classificações. Métodos de validação cruzada podem ajudar a escolher o melhor valor de k, minimizando o risco de *overfitting* ou *underfitting* [2] (FACELI et al., 2021). Se o k for muito grande, ocorre um aumento do erro de aproximação, pois

[2] Underfitting refere-se a um problema que ocorre quando um modelo é muito simples para capturar os padrões dos dados de treinamento

o modelo tende a generalizar demais. Por outro lado, se o k for muito pequeno, o modelo se torna mais sensível a ruídos nos dados, aumentando a variância e a possibilidade de overfitting (FAN et al., 2019).

Para determinar os vizinhos mais próximos, podem ser usadas diferentes métricas de distância, como (FACELI et al., 2021) (CUNNINGHAM; DELANY, 2020):

- **Distância Euclidiana**: Refere-se à distância "reta" entre dois pontos em um espaço n-dimensional. A fórmula é:

$$d(x_i, x_j) = \sqrt{\sum_{l=1}^{d} (xi - x_j)^2} \tag{4}$$

- **Distância de Manhattan**: É a distância entre dois pontos medida ao longo dos eixos em ângulos retos, calculada somando as diferenças absolutas entre as coordenadas.

$$d(x_i, x_j) = \sum_{l=1}^{d} |xi - x_j| \tag{5}$$

- **Distância de Minkowski**: É uma generalização das distâncias Euclidiana e de Manhattan. Quando $p = 1$, temos a fórmula de Manhattan; e quando $p = 2$, obtemos a fórmula Euclidiana.

$$d(x_i, x_j) = \left(\sum_{l=1}^{d} |xi - x_j|^p \right)^{\frac{1}{p}} \tag{6}$$

O algoritmo KNN possui diversas aplicações em diferentes áreas, como a classificação de documentos (SOUCY; MINEAU, 2001), diagnóstico médico e detecção de doenças (XING; BEI, 2020; UDDIN et al., 2022), além da cibersegurança, onde é utilizado para detectar intrusões em sistemas (WAZIRALI, 2020). Além disso, estudos comparativos indicam que o KNN pode apresentar melhor desempenho em termos de precisão quando comparado a outros classificadores, como o SVM (SINHA; SINHA et al., 2015), Regressão Logística e Floresta Aleatória (SHAH et al., 2020).

O KNN é um algoritmo indutivo, ou seja, classifica novos exemplos com base nas semelhanças entre as amostras já rotuladas. Sua simplicidade, escalabilidade e facilidade de aplicação são pontos fortes, além de ser eficaz na regressão e classificação e também mostra um bom desempenho para lidar com dados não lineares (TAUNK et al., 2019). Por outro lado, ele é um algoritmo "preguiçoso", a fase de predição pode ser computacionalmente custosa, especialmente em grandes volumes de dados, já que é necessário calcular as distâncias para todas as amostras do conjunto de treinamento. Além disso, o desempenho do KNN pode ser prejudicado por atributos irrelevantes ou redundantes e pela "maldição da dimensionalidade", em que um número elevado de atributos causa um crescimento exponencial nas dimensões do problema, reduzindo a precisão do algoritmo (FACELI et al., 2021) (TAUNK et al., 2019).

2.4.4 Árvore de decisão

As Árvores de Decisão (DT) são métodos de aprendizado supervisionado não paramétricos, amplamente utilizadas tanto para classificação quanto para regressão. Esses modelos são estruturados de forma hierárquica, onde cada nó interno representa um teste em um atributo dos dados, as ramificações indicam os resultados desses testes, e os nós folha correspondem aos valores previstos ou rótulos de classe. O caminho que conecta a raiz da árvore ao nó folha define uma sequência de regras de decisão que levam à predição final (Scikit-learn Developers, 2024).

A DT opera de maneira recursiva, sendo composta por elementos essenciais como nós, ramificações, divisão, critérios de parada e poda. O processo de divisão dos nós é conduzido pela seleção da variável mais relevante, o que pode ser realizado por diferentes métricas, incluindo entropia, índice de Gini, erro de classificação, ganho de informação e razão de ganho. Para evitar a complexidade excessiva do modelo e reduzir o risco de overfitting, critérios de parada são estabelecidos, como o número mínimo de registros em um nó folha, a quantidade de registros necessária antes da divisão ou a profundidade máxima da árvore. Além disso, a poda é um mecanismo essencial para remover nós que contribuem pouco para a predição, tornando o modelo mais eficiente e generalizável (SONG; LU, 2015).

As aplicações das Árvores de Decisão são amplas e abrangem diversas áreas do conhecimento. Na cibersegurança, por exemplo, elas são utilizadas na detecção de intrusão em dispositivos IoT, permitindo a identificação de acessos suspeitos por meio da análise de padrões de tráfego (FERRAG et al., 2020). No campo da saúde, estudos demonstram que a combinação de DT com modelos baseados em lógica *fuzzy* tem sido aplicada para a predição do câncer de mama, contribuindo para diagnósticos mais precisos (GUPTA et al., 2023).

Dentre as vantagens do uso das Árvores de Decisão, destaca-se a sua facilidade de interpretação, tornando-as acessíveis para usuários sem conhecimento aprofundado em modelagem estatística. Além disso, esses modelos são capazes de lidar tanto com variáveis categóricas quanto numéricas, apresentam resiliência a dados distorcidos e podem ser adaptados para cenários onde há valores ausentes (MONARD; BARANAUSKAS, 2003). No entanto, as DTs também apresentam algumas limitações. A replicação de decisões em diferentes ramos pode resultar em modelos redundantes, comprometendo a eficiência e a acurácia preditiva. O tratamento de valores ausentes pode se tornar um desafio, dificultando a escolha adequada do ramo a ser seguido. Além disso, a presença de atributos contínuos pode aumentar o tempo de processamento, pois exige a ordenação dos dados para determinar pontos de divisão. Modelos baseados em Árvores de Decisão também estão sujeitos a problemas como overfitting, especialmente quando treinados com conjuntos de dados pequenos, e instabilidade, uma vez que pequenas variações nos dados podem resultar em estruturas de árvore significativamente diferentes (FACELI et al.,

2021) (Scikit-learn Developers, 2024) (SONG; LU, 2015).

2.4.5 Floresta aleatória

A RF é um algoritmo supervisionado utilizado tanto para regressão quanto para classificação. Ele funciona criando várias árvores de decisão durante a fase de treinamento, onde cada árvore é construída a partir de um subconjunto aleatório dos dados. Esse algoritmo lida bem com dados não lineares, por conseguir adaptar rapidamente a eles. A ideia é baseada no modelo de criação de árvores de maneira recursiva até o critério de parada (SCHONLAU; ZOU, 2020).

A RF faz uso de múltiplos modelos, combinando-os em um conjunto chamado de *ensambled*, que permite que o modelo lide com problemas de forma colaborativa, aprimorando assim o desempenho preditivo. As principais técnicas de ensemble utilizadas são (Scikit-learn Developers, 2024) (GéRON, 2021):

- ❑ ***Bagging* ou *Bootstrap Aggregation*:** Nesta técnica, os modelos são treinados de forma paralela, onde diferentes subconjuntos de treinamento são criados a partir dos dados originais, utilizando amostragem com reposição. A previsão final é realizada pela média das previsões (no caso de regressão) ou pela votação majoritária (no caso de classificação).

- ❑ ***Boosting*:** Os modelos são criados sequencialmente, com cada novo modelo tentando corrigir os erros dos modelos anteriores. A cada iteração, um peso maior é dado aos dados mal classificados, e a previsão final é obtida pela média ponderada das previsões.

Por ser um algoritmo de *ensambled* [3], a RF consegue obter previsões mais precisas em comparação com modelos individuais. Seu funcionamento pode ser resumido nas seguintes etapas (GéRON, 2021):

1. Construção de um conjunto de árvores de decisão: Cada árvore se especializa em diferentes aspectos dos dados e opera de forma independente, minimizando o risco de o modelo ser influenciado por uma única árvore.

2. Seleção de amostras de dados de treinamento aleatórios: Isso garante a diversidade dos conjuntos de dados, permitindo que as árvores obtenham diferentes recursos.

3. Uso do modelo *Bagging* ou *Bootstrap Aggregation*: Criação de várias amostras *bootstrap* a partir dos dados originais, garantindo a substituição de instâncias. Isso permite a variabilidade no processo de treinamento, pois diferentes subconjuntos de dados são utilizados para cada árvore de decisão.

[3] Refere-se a uma técnica de *machine learning* que combina múltiplos modelos para aumentar a precisão de uma previsão.

4. Realização da votação e decisão final: A previsão final é feita pela média das previsões das árvores (no caso de regressão) ou pelo voto majoritário (no caso de classificação), onde cada árvore de decisão contribui com seu voto.

O algoritmo de RF é amplamente adotado devido à sua eficácia em resolver problemas de classificação e regressão. Ele costuma apresentar melhores resultados do que uma única árvore de decisão, oferecendo um bom desempenho com previsões altamente precisas, mesmo na presença de dados ausentes, além de ser eficiente em grandes bases de dados (GéRON, 2021) (Scikit-learn Developers, 2024).

Uma das principais limitações do algoritmo RF é a interpretabilidade, uma vez que a combinação de múltiplas árvores de decisão dificulta a compreensão dos critérios usados para cada predição, tornando o modelo uma "caixa preta" para muitos usuários. Além disso, o modelo pode se tornar computacionalmente intensivo, especialmente quando se utiliza um grande número de árvores e características, o que exige maior memória e tempo de processamento. Outra desvantagem é a tendência do modelo a superestimar a importância de variáveis altamente correlacionadas, o que pode afetar a análise de interpretação de variáveis (HASTIE; TIBSHIRANI; FRIEDMAN, 2009).

2.4.6 Regressão logística

A LR é um modelo supervisionado utilizado para distinguir entre classes, geralmente binárias, sendo por isso considerado um modelo discriminativo. Seu funcionamento baseia-se na razão de chances, que é a probabilidade de sucesso dividida pela de fracasso, e na transformação logit aplicada a essa razão. A função logit é representada por (GéRON, 2021):

$$\log it(\pi) = \frac{1}{1 + \exp(-\pi)} \tag{7}$$

$$\ln\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 X_1 + \ldots + \beta_k X_k \tag{8}$$

Onde Logit(π) é a variável de resposta, X corresponde às variáveis independentes, e β é o coeficiente a ser estimado por meio da estimativa de máxima verossimilhança (MLE). O valor de β é ajustado através de várias iterações, onde o coeficiente é modificado até se encontrar o melhor ajuste para a razão de chances. Cada iteração gera uma função de verossimilhança logarítmica, e a LR busca maximizar essa função para encontrar a melhor estimativa.

Após o cálculo do modelo, é gerada a probabilidade prevista. Em uma classificação binária, valores menores que 0,5 indicam a classe 0, enquanto valores maiores que 0,5 indicam a classe 1. Finalmente, é realizado o cálculo de adequação para avaliar quão bem o modelo prevê a variável dependente. Existem três tipos principais de análise de LR, diferenciados pela natureza da variável dependente (IBM, 2024):

1. **Regressão logística binária:** É utilizada para problemas de classificação binária, onde os resultados possíveis são 0 ou 1. Esta abordagem é a mais comum na LR.

2. **Regressão logística multinomial:** Aplica-se a problemas com múltiplos resultados finitos possíveis. A LR Multinomial agrupa as saídas em categorias distintas e mapeia os valores para probabilidades entre 0 e 1.

3. **Regressão logística ordinal:** Um tipo de LR multinomial, usada quando a saída precisa ser uma classificação ordenada, em vez de valores reais, com uma ordem definida.

A LR maximiza a função de verossimilhança logarítmica para determinar os coeficientes β do modelo. Quando essa verossimilhança é negativa, ela é considerada como uma perda, e a descida de gradiente é utilizada para encontrar o máximo global. A LR pode ser ajustada em casos de alta dimensionalidade, penalizando coeficientes grandes para evitar o *overfitting* (IBM, 2024).

A LR é um algoritmo simples de implementar, com a capacidade de processar grandes volumes de dados, graças à sua baixa exigência computacional. Além disso, é flexível e pode ser aplicada tanto a problemas com dois resultados quanto a problemas com múltiplos resultados (IBM, 2024). Uma limitação significativa da Regressão Logística é que ela assume uma relação linear entre as variáveis independentes e a função logit (log-odds) do resultado, o que pode limitar sua eficácia em dados que apresentam relações não lineares, tornando-a inadequada para capturar complexidades em conjuntos de dados reais. Além disso, a Regressão Logística pode ser sensível a *outliers*, especialmente quando não se aplicam técnicas de regularização adequadas, como Ridge ou Lasso, o que pode levar a coeficientes instáveis. Esse modelo também é restrito a problemas de classificação binária, sendo necessário recorrer a variações, como Regressão Logística Multinomial, para lidar com problemas multiclasse (BISHOP, 2006).

2.4.7 Aprendizagem Profunda

A DL é uma técnica de IA, derivada da ML, que possui apenas uma camada, enquanto a DL tem múltiplas camadas conectadas entre si. A DL utiliza algoritmos de redes neurais artificiais para criar neurônios artificiais, com o objetivo de resolver problemas mais complexos. O uso da DL tornou-se mais viável recentemente devido a avanços tecnológicos e à superação de limitações práticas, como mencionado em (GABRIEL, 2022). A arquitetura de DL possui múltiplas camadas escondidas; estas camadas têm o objetivo de aprender de forma abstrata com as características extraídas ao longo das camadas a partir da entrada de dados brutos (WANI et al., 2020).

A Redes Neurais Artificiais (RNA) é composta por uma ou mais camadas, nas quais os neurônios são distribuídos. Quando há mais de uma camada, as entradas e saídas podem

ser transmitidas a outros neurônios, passando de uma camada para outra. Na Figura 5, é possível visualizar um exemplo de RNA com três camadas, duas entradas e duas saídas (FACELI et al., 2021).

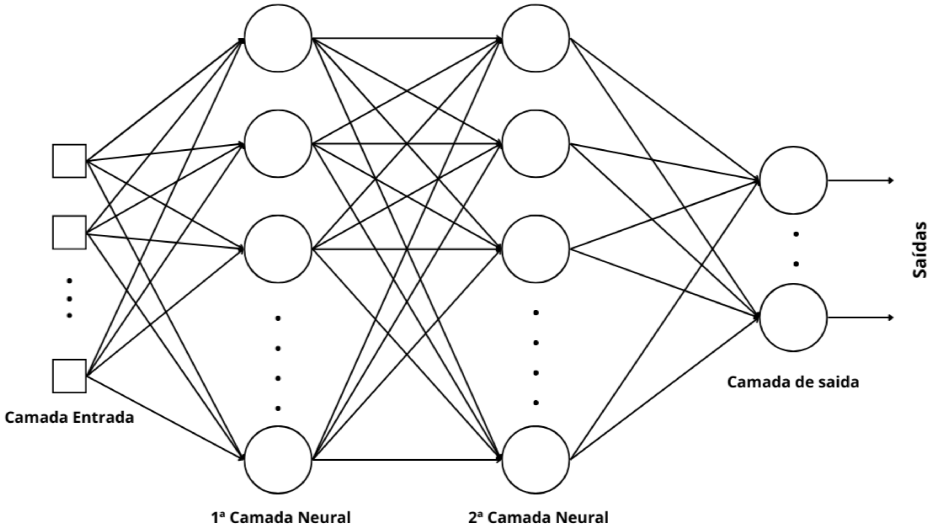

Figura 5 – RNA multicamadas típica. Fonte: Elaborada pelo autor.

A construção de máquinas inteligentes levou ao desenvolvimento da RNA, inspirada na arquitetura do sistema nervoso, com o objetivo de simular o funcionamento do cérebro humano em processos de pensamento e tomada de decisões. O sistema nervoso é composto por um complexo conjunto de neurônios, responsáveis pela transmissão de estímulos a outros neurônios (GABRIEL, 2022).

A Figura 6 apresenta a representação de um modelo matemático simplificado de um neurônio artificial. Esse modelo se inspira no funcionamento dos neurônios biológicos. As entradas x correspondem aos sinais recebidos pelo neurônio, e os pesos sinápticos W determinam a influência de cada entrada na saída do neurônio. Dessa forma, as entradas são multiplicadas pelos pesos, gerando uma soma ponderada. Esse valor resultante é então passado por uma função de ativação, que define a saída final do neurônio (ZHANG et al., 2023).

A função de ativação desempenha um papel fundamental no desempenho de redes neurais. Existem diversas funções de ativação, como a *linear*, *sigmoid*, Função Linear Retificada (ReLU), *Leaky* ReLU, *Tanh* e *Softmax*, representadas na Figura 7 (WANI et al., 2020).

A DL é amplamente utilizada devido ao seu desempenho na aplicação de modelos profundos capazes de treinar dados não lineares com alta eficiência, graças à sua flexibilidade. Além disso, apresenta boa capacidade de generalização para lidar com novos dados e pode ser empregada em diversas aplicações, exigindo apenas pequenas adaptações para se adequar a problemas específicos. Outra vantagem é sua capacidade de resolver problemas complexos. No entanto, o treinamento pode ser demorado e demandar grande poder

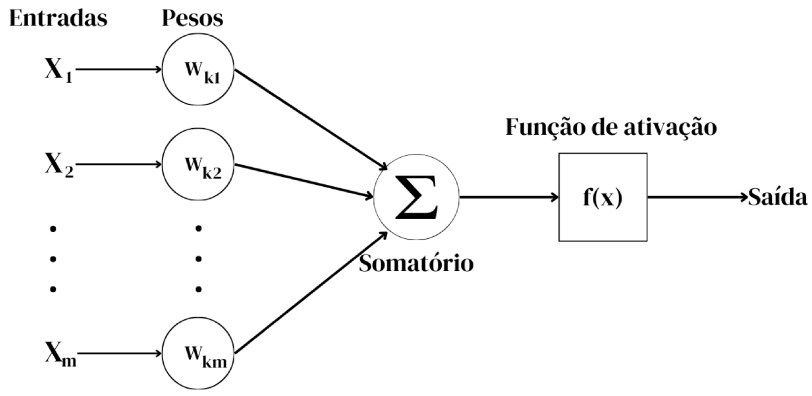

Figura 6 – Neurônio matemático simplificado. Fonte: Elaborada pelo autor.

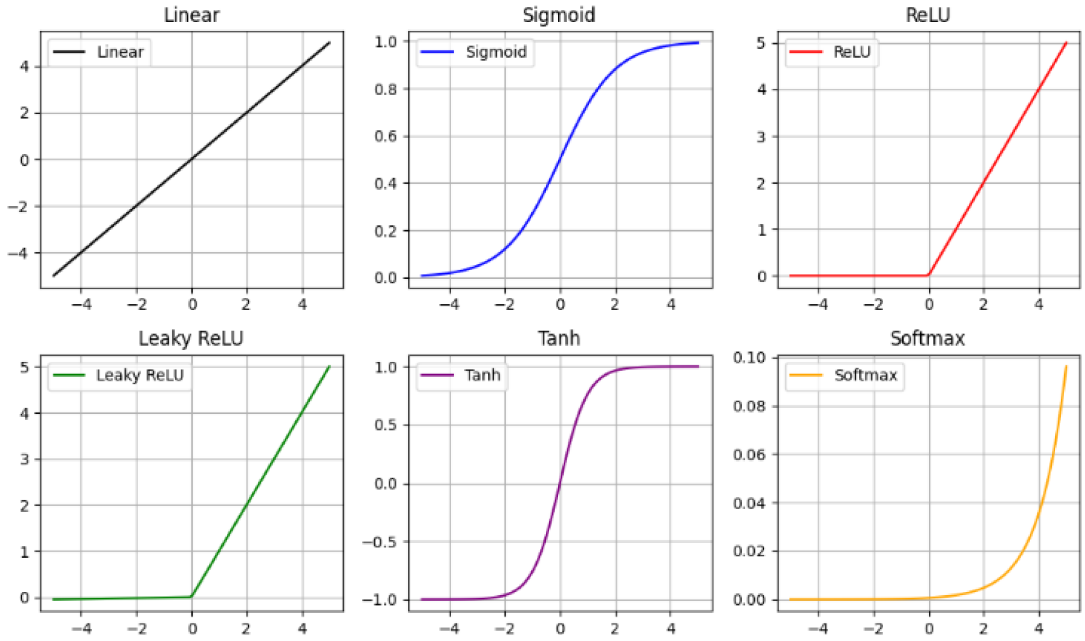

Figura 7 – Funções de ativação. Fonte: Elaborada pelo autor.

computacional para algumas tarefas, devido à complexidade dos problemas, ao número de camadas e aos parâmetros de cada camada (WANI et al., 2020).

A DL é utilizada em diversas tarefas de classificação em diferentes áreas, como medicina, reconhecimento de imagens, entre outras. O trabalho de (AYADI et al., 2021) propôs o uso da CNN para a classificação de tumores cerebrais com base em imagens de ressonância magnética. (BEHERA et al., 2021) utiliza os modelos de CNN e LSTM para realizar a previsão de análise de sentimentos para dados grandes. Nessas aplicações, é possível observar a ampla adoção do modelo CNN específico devido à sua alta *accuracy* na identificação de objetos e na classificação de imagens (SHARIFANI; AMINI, 2023).

2.4.7.1 Rede Neural Convolucional

A CNN é um modelo utilizado principalmente na área de tratamento de imagens e séries temporais. A Figura 8 mostra uma arquitetura básica que contém múltiplas camadas. A primeira camada contém os dados de entrada. A camada *convolution* é responsável por realizar a extração de características, enquanto a camada *Pooling* reduz o tamanho das características, além de diminuir o tempo de processamento e prevenir o *overfitting*. A última camada é a camada totalmente conectada, responsável por calcular as ativações e fornecer uma saída, assim como nas DL (YI et al., 2017).

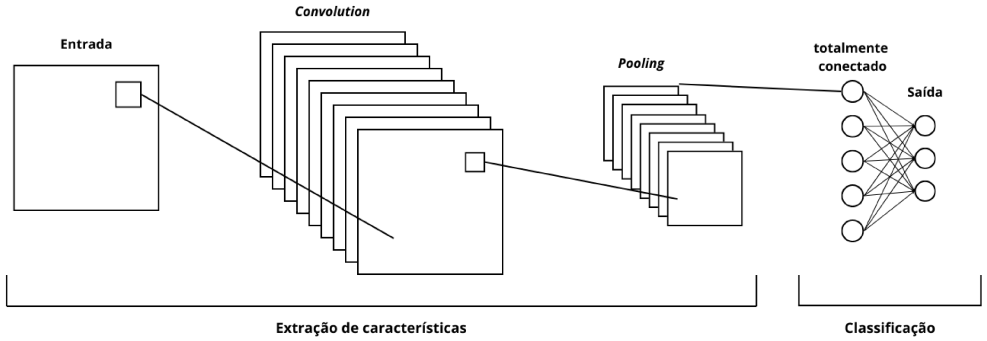

Figura 8 – Arquitetura básica de uma CNN. Fonte: Elaborada pelo autor.

O filtro é responsável por definir o nível de extração, podendo ser grandes ou pequenos. Sistemas que possuem uma rede profunda com funções específicas demonstram maior eficácia em relação a redes menores; porém, o custo computacional aumenta conforme a profundidade. A criação de divisões em seções pode ajudar a mitigar problemas de gradiente que surgem com o aumento da profundidade, reduzindo erros de teste e treinamento.

O mapeamento de características baseado em CNN é realizado para selecionar diferentes atributos com base nos pesos de um *kernel*. A largura da rede tem um impacto significativo na retenção de informações valiosas, pois arquiteturas profundas podem perder dados importantes. A atenção na CNN consiste em hierarquias focadas em características relevantes, o que influencia significativamente a predição. Um modelo que utiliza atenção é a LSTM. A dimensão permite codificar informações de forma separada no espaço; no entanto, é muito custosa em termos de desempenho computacional (BHATT et al., 2021).

As CNN são eficientes para grandes quantidades de dados de séries temporais e imagens, conseguindo extrair informações de problemas de maneira automática. No entanto, apresentam algumas desvantagens, como o fato de serem algoritmos complexos e de difícil compreensão, o que pode dificultar sua verificação. Pequenas mudanças nos hiperparâmetros podem afetar significativamente o desempenho da rede e, dependendo do problema, a CNN pode exigir grandes recursos computacionais (BHATT et al., 2021).

2.4.8 Memória de curto longo prazo

Outro modelo de rede foi proposto em 1997 por (HOCHREITER; SCHMIDHUBER, 1997) conhecido como LSTM, para solucionar problemas de desaparecimento ou perda de gradiente da *Rede Neural Recorrente* (RNN), foi sugerido adicionar o portão de esquecimento. A arquitetura da LSTM utiliza três portões: o de entrada, saída e o do esquecimento como mostrada na Figura 9.

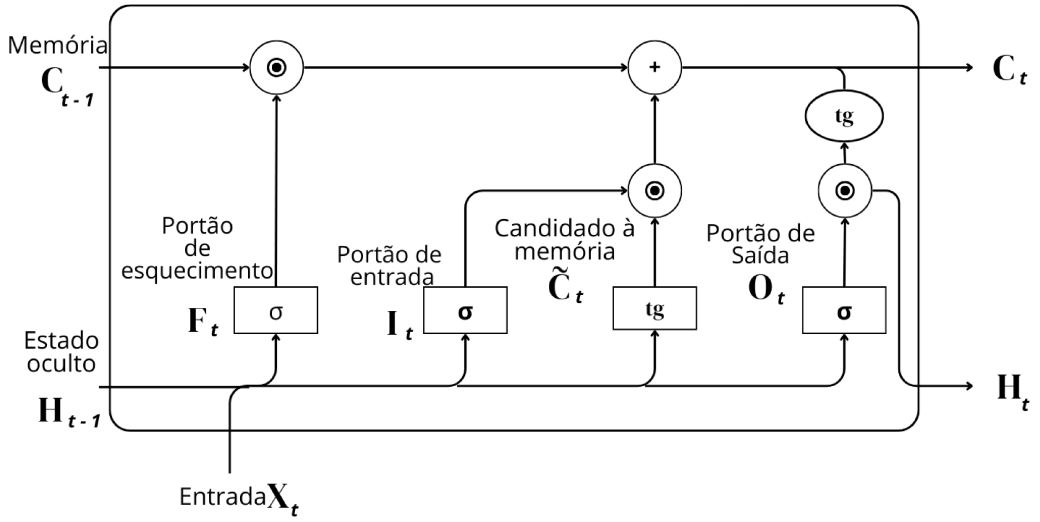

Figura 9 – Arquitetura básica de uma LSTM. Fonte: Elaborada pelo autor.

A arquitetura básica da LSTM consiste em um bloco de memória com portões adaptativos e multiplicativos responsáveis por controlar a entrada e a saída de informações. As células de memória possuem uma unidade recorrente conectada, cujo objetivo é resolver problemas de erro, evitando a perda de informações importantes. O portão de entrada determina quais informações serão armazenadas, enquanto o portão de saída controla quais informações serão enviadas para a próxima camada da rede. Por fim, o portão de esquecimento é responsável por atualizar as informações armazenadas na memória quando os dados são ruins; os portões são calculados pela função *sigmoid* (GERS; SCHMIDHUBER; CUMMINS, 2000).

A LSTM é eficiente em capturar e manter informações de longas séries temporais, evitando a perda de dados importantes para a classificação com alta precisão (DESCOVI et al., 2023). Por outro lado, a LSTM apresenta um alto custo computacional, o que pode tornar o processo de classificação mais lento (SMAGULOVA; JAMES, 2019).

2.5 Trabalhos Correlatos

O trabalho de Messaoud e Chavez (2021) apresentou um modelo para classificar períodos interictais e pré-ictais utilizando o classificador *Random Forest*, Perceptron multi-

camadas (MLP), SVM, KNN e *AdaBoost*. Uma das estratégias adotadas foi a exclusão de épocas extraídas, a fim de diminuir a chance de falsos positivos. O estudo utilizou duas bases de dados, a base de dados de EEG no couro cabeludo de indivíduos com epilepsia descrita por Shoeb (2009) Goldberger et al. (2000) Guttag (2010) e a base intracraniana com EEGs de cachorros, fornecida por bbrinkm e Cukierski (2014). A abordagem com o RF atingiu uma sensibilidade de 82,07%, com uma baixa taxa de falsos positivos. O método proposto envolveu as etapas de aquisição dos sinais, pré-processamento e segmentação, extração e seleção de características, classificação seguidos pela avaliação do modelo. Além disso, o modelo implementou uma técnica de rejeição de artefatos chamada *thresholding*, onde épocas contaminadas por ruídos ou artefatos, que excedem um limite predefinido, são descartadas. Essa técnica foi fundamental para reduzir o número de alarmes falsos, contribuindo para a eficácia do modelo.

Outro trabalho proposto para a detecção de convulsões de epilepsia foi o de Lestari et al. (2020), que utilizou e comparou os classificadores *Naive Bayes*, *Random Tree Forest* e *K-Nearest Neighbor* para classificar os períodos convulsivos e não convulsivos. Os métodos consistiram na aquisição dos sinais da base de dados CHB-MIT Scalp EEG Database Shoeb (2009) Goldberger et al. (2000) Guttag (2010), seguidos pelo pré-processamento, extração de características e o treinamento e validação dos modelos. O pré-processamento consistiu na aplicação de uma filtragem de banda de 0,5-30 Hz. As características extraídas foram temporais e espectrais. As características temporais incluíram a média, raiz quadrada média (RMS) e desvio padrão (STD), enquanto as características espectrais consistiram em picos espectrais e potência espectral nas bandas delta, teta, alfa e beta. O classificador KNN atingiu uma precisão de 92,7%, seguido da *Random Tree Forest*, com 86,6% de precisão, enquanto o *Naive Bayes* obteve uma precisão de 55,6%.

Tuncer e Bolat (2022) propuseram realizar a classificação de crises focais e generalizadas, comparando os métodos tradicionais de *machine learning* e *deep learning*. Os dados foram coletados utilizando o EEG no couro cabeludo com o uso da base de dados do Temple University Hospital, seguido do pré-processamento, extração de características com o método de *discrete wavelet*, seleção de características com base na Correlation-based Feature Selection (CFS) e, por fim, a classificação com os métodos KNN, SVM, RF e LSTM. A classificação obteve 95,92% de acurácia para a LSTM, 95,45% RF, 95,75% o KNN, 91,4% SVM e 98,08% de acurácia para a generalizada da LSTM.

Tran et al. (2022) realizaram um estudo sobre a detecção de epilepsia com o uso de *machine learning*. Utilizaram a base de dados da Universidade de Bonn Andrzejak et al. (2001) para realizar a validação do estudo para os períodos ictal e interictal. Também foi realizada a extração de características da base de dados utilizando o *discrete wavelet transform analysis*. A próxima etapa foi a seleção de características com base na otimização binária dos dados, reduzindo assim a dimensionalidade e o tempo de computação. Para o treinamento, os métodos e as acurácias obtidas foram SVM 99,2%, KNN 98,6%,

DT 97,3% e RF 98,4%.

Savadkoohi, Oladunni e Thompson (2020) utilizaram os algoritmos de aprendizado de máquina SVM e KNN para realizar a classificação de convulsões epilépticas e períodos sem convulsões a partir do sinal de EEG pré-processado. O desempenho foi comparado com base nas métricas de Precisão, Sensibilidade e Especificidade. Antes da classificação, as etapas incluíram a extração de características de cada onda de EEG nos domínios do *frequency and time-frequency domains* utilizando o filtro *Butterworth*, a *Fourier Transform* e a *Wavelet Transform*, respectivamente. Além disso, foi realizada a seleção de características por meio do teste T e da *Sequential Forward Floating Selection* (SFFS). A base de dados utilizada foi a da Universidade de Bonn Andrzejak et al. (2001). Entre os resultados mais relevantes, o SVM utilizando características de *frequency domain* obteve 100% de precisão, sensibilidade e especificidade, enquanto o KNN, ao combinar características de *Transformada Wavelet*, alcançou uma precisão de 99,5%, com sensibilidade de 99% e especificidade de 100%. Esses resultados destacam a eficácia desses algoritmos, com o SVM apresentando um desempenho ligeiramente superior em termos de acurácia global.

Capítulo 3

Experimentos e Análise dos Resultados

Neste capítulo, são detalhadas as etapas realizadas para comparar diferentes técnicas para realizar classificação. São apresentados o banco de dados utilizado, o pré-processamento aplicado, as técnicas de extração e seleção de características, bem como os modelos de classificação e a avaliação dos resultados obtidos.

3.1 Tecnologias

O computador utilizado possui um processador Intel Core i5 e 8 GB de RAM. O ambiente escolhido foi o JupyterLab, devido à sua interface intuitiva e flexível, proporcionando maior interação. A linguagem utilizada foi a linguagem Python, por oferecer uma vasta gama de bibliotecas gratuitas para *machine learning*, além de possuir uma sintaxe simplificada e de fácil compreensão, tornando seu uso acessível e eficiente para esse tipo de aplicação.

3.2 Base de dados

A base de dados utilizada foi a CHB-MIT Scalp EEG Database (SHOEB, 2009), (GOLDBERGER et al., 2000) (GUTTAG, 2010), coletada no *Children's Hospital Boston*. Essa base contém gravações de 22 pacientes pediátricos com epilepsia intratável, mesmo após a retirada de medicamentos. O arquivo 21 contém a gravação do paciente 01 um ano e meio depois da primeira gravação, e o caso 24 foi adicionado à base de dados em 2010. As gravações foram realizadas como parte da avaliação da candidatura para intervenção cirúrgica.

As amostragens foram realizadas com uma frequência de 256 amostras por segundo, utilizando resolução de 16 bits. A maioria dos arquivos contém 23 sinais de EEG, registrados com o sistema internacional de posicionamento de eletrodos 10-20. A Figura 10 apresenta os nomes e as ondas dos sinais de EEG em uma janela com duração de 5 segundos a partir de dados da base de dados gerados no ambiente do Jupyter.

Em alguns registros, há sinais adicionais, como Eletrocardiograma (ECG) ou Estimulação do Nervo Vago (VNS). A base também inclui sinais fictícios que podem ser ignorados no processamento. Os arquivos estão no formato .edf, e o banco de dados registra um total de 197 crises epilépticas. A Tabela 2 contém um modelo com as descrições do arquivo, com o tempo de gravação, quantidade de convulsões e canais em cada caso.

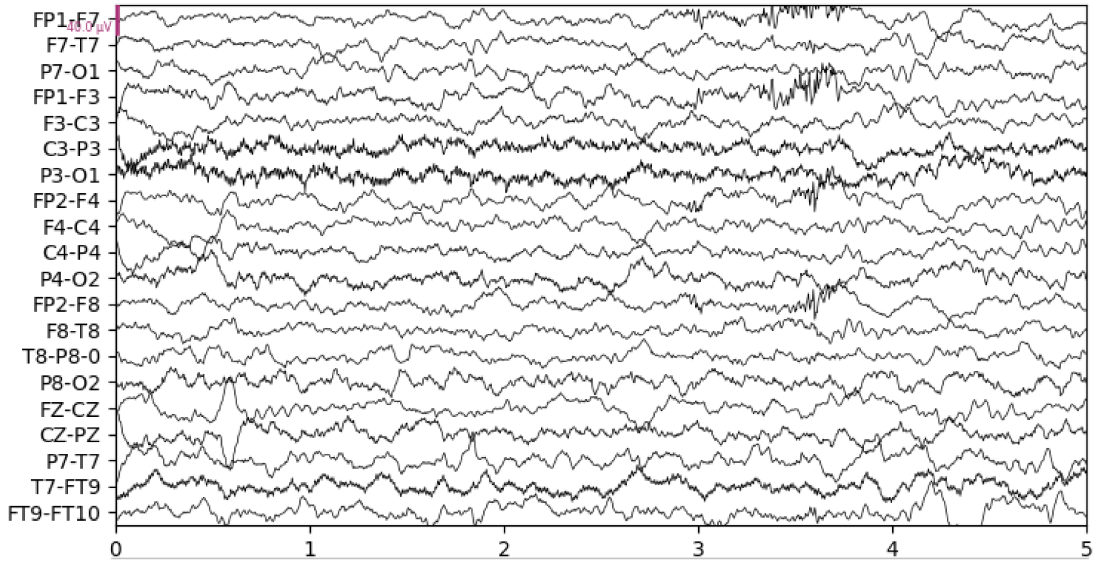

Figura 10 – Gravação do EEG. Fonte: Elaborada pelo autor.

3.3 Pré-processamento

A base de dados utilizada contém mais de 1000 horas de gravação, incluindo um total de 197 crises epilépticas documentadas, cada uma com duração de até 5 minutos. Para reduzir o custo computacional e melhorar o balanceamento entre as classes, foram coletadas pequenas janelas de dados correspondentes aos períodos interictais (período em que não há evidências de atividade anormal no cérebro), ictais (período em que ocorre uma crise epiléptica) e pré-ictais (período que antecede uma crise de epilepsia). Exemplos desses períodos podem ser visualizados nas Figuras 11, 12 e 13, respectivamente, geradas a partir dos sinais da base de dados utilizada.

As janelas foram definidas a partir da realização da classificação com diferentes tempos. O período ictal foi testado com janelas de 5 minutos antes e depois das crises, enquanto para a classificação foram utilizadas janelas de 15 e 10 segundos. Observou-se que, para janelas muito grandes, a distinção entre os períodos pré-ictais e ictais apresentou um desempenho inferior em comparação com janelas menores.

Já o período pré-ictal foi definido com janelas de 20, 15 e 10 minutos antes da ocorrência de uma convulsão. Os resultados mostraram que, quanto maior a distância em relação à crise, maior era a dificuldade na distinção entre os períodos pré-ictal e interictal.

Caso	Tempo de gravação	N° de Convulsões	N° de canais
chb01	21h	7	23
chb02	35h	3	23
chb03	37h	7	23
chb04	141h	4	23
chb05	39h	5	23
chb06	68h	10	23
chb07	81h	3	23
chb08	20h	5	23
chb09	83h	4	24
chb10	48h	7	23
chb11	33h	3	28
chb12	25h	40	28
chb13	33h	12	28
chb14	26h	8	28
chb15	40h	20	38
chb16	19h	10	28
chb17	21h	3	28
chb18	36h	6	28
chb19	29h	2	23
chb20	54h	8	28
chb21	32h	4	28
chb22	28h	3	28
chb23	29h	7	23
chb24	21h	16	23
Total	999h	197	-

Tabela 2 – Descrição da Base de Dados. Fonte: Elaborada pelo autor.

O período ictal foi definido como o intervalo das crises conforme marcado na base de dados. Para o período pré-ictal, foram consideradas janelas de 100 segundos iniciando 10 minutos antes do início de cada crise. Já o período interictal incluiu dados coletados com pelo menos 2 horas de diferença em relação ao período das crises.

Na segunda etapa do pré-processamento, foi realizada a seleção de 21 canais de EEG, excluindo canais fictícios e sinais adicionais ou duplicados. Além disso, foi aplicado um filtro de frequência que permitiu a passagem de sinais com frequências entre 1 Hz e 45 Hz. Esse filtro foi implementado utilizando a biblioteca MNE, com o objetivo de reduzir ruídos nos sinais, garantindo que apenas as componentes de frequência relevantes fossem mantidas.

3.4 Extração e Seleção de Características

Essa etapa consistiu na extração de características das janelas, incluindo média, desvio padrão, curtose, centróide espectral e energia. Foram analisados 21 canais, resultando em

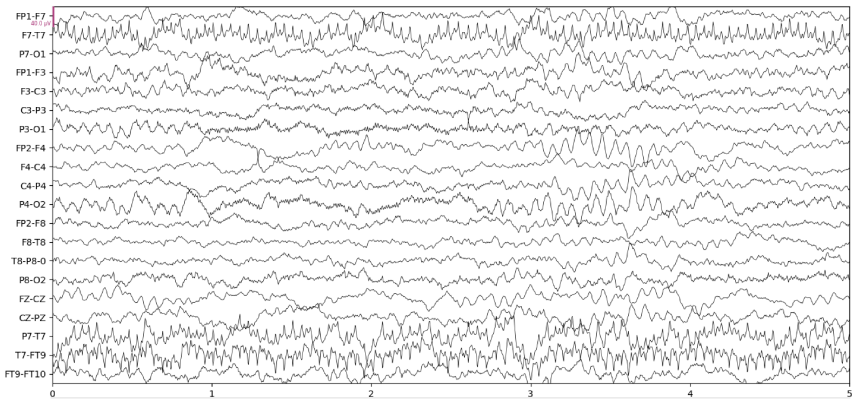

Figura 11 – Sinais de EEG do período interictal com uma janela de 5 segundos Fonte: Elaborada pelo autor.

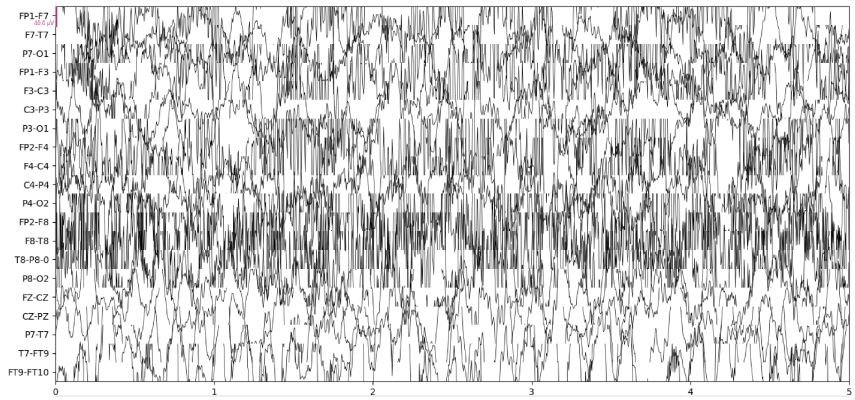

Figura 12 – Sinais de EEG do período ictal com uma janela de 5 segundos Fonte: Elaborada pelo autor.

105 características extraídas por janela.

A seleção de características foi realizada utilizando o método *Recursive Feature Elimination* (RFE) da biblioteca *scikit-learn*. Esse método foi aplicado com um modelo de regressão logística, configurado para identificar as 25 características mais relevantes do conjunto de dados. O RFE funciona removendo iterativamente as características menos importantes com base em sua contribuição para o desempenho do modelo, até atingir o número desejado de características. Assim, foi possível reduzir a dimensionalidade dos dados, mantendo apenas as informações mais significativas para a classificação, o que pode melhorar a eficiência e a precisão do modelo.

3.5 Treinamento e teste

Os dados foram divididos em dois conjuntos: 70% destinados ao treinamento do modelo e 30% para sua validação em testes, uma prática padrão para garantir um equilíbrio entre dados de treinamento e validação. Essa proporção foi escolhida por apresentar um melhor desempenho em comparação com outras divisões padrão testadas, como 75%-25%

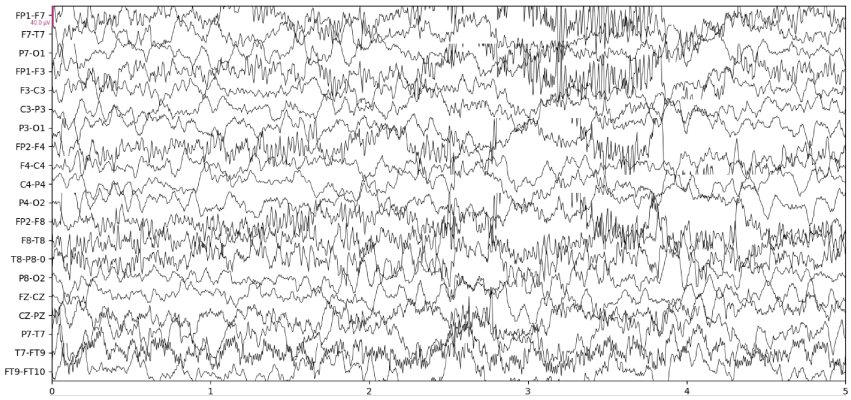

Figura 13 – Sinais de EEG do período pré-ictal com uma janela de 5 segundos Fonte: Elaborada pelo autor.

e 80%-20%.

A separação dos conjuntos foi realizada utilizando a função *train test split* da biblioteca *scikit-learn*, garantindo uma distribuição adequada para avaliar o desempenho do modelo em dados não vistos. Essa divisão permite que o modelo tenha uma quantidade suficiente de dados para aprendizado, ao mesmo tempo que reserva uma parte significativa para testar sua capacidade de generalização.

3.6 Hiperparâmetros

Para realizar as classificações, é necessário encontrar os melhores hiperparâmetros para otimizar o desempenho do modelo. Os primeiros testes foram feitos manualmente, com uma quantidade limitada de parâmetros devido ao tempo de processamento dos algoritmos. A definição de parâmetros de forma manual foi realizada com base em testes feitos manualmente e nos resultados observados ao longo do processo. Na Tabela 3, é possível observar os modelos, as parametrizações testadas e, na última coluna, a combinação escolhida. Esses testes foram realizados de forma limitada, se comparados à definição de parâmetros automática feita com o *Grid Search*, como definido a seguir.

Foi utilizada a técnica de *Grid Search*, uma ferramenta da biblioteca Scikit-learn Developers (2024), que, a partir dos dados e dos hiperparâmetros fornecidos, seleciona automaticamente a melhor configuração. A técnica *Grid Search* permite testar diferentes combinações para encontrar a melhor configuração, conforme descrito em Scikit-learn Developers (2024). Os hiperparâmetros testados para cada modelo estão apresentados na Tabela 4, juntamente com as possíveis combinações avaliadas e o melhor resultado obtido.

O modelo SVM possui o parâmetro *kernel*, que é utilizado para transformar os dados em um formato que possa ser mais facilmente separado. O *kernel rbf*, mostrado na Tabela 3, é adequado para problemas não lineares. Já o *kernel linear*, utilizado na Tabela 4, consegue mapear os dados em um espaço de alta dimensionalidade. O parâmetro *gamma*

Modelo	Parâmetros	Combinação final
SVM	C	200
	kernel	*rbf*
	gamma	*scale*
KNN	*metric*	*Minkowski*
	weights	*uniform*
	n_neighbors	3
RF	*n_estimators*	10
	max_depth	*None*
	min_samples_split	2
	min_samples_leaf	1
DT	*max_depth*	*None*
	min_samples_split	2
	min_samples_leaf	1
LR	C	1
	penalty	L2
	solver	lbfgs

Tabela 3 – Hiperparâmetros obtido utilizando a técnica manual na escolha das combinações de cada modelo. Fonte: Elaborada pelo autor.

escolhido como *scale* ajusta automaticamente o valor com base na variância dos dados. O parâmetro de regularização C controla o equilíbrio entre a busca por uma margem larga e a penalização por erros de classificação, como o escolhido de 200.

O modelo KNN foi configurado com o número de vizinhos igual a 3 (k), pois valores maiores apresentaram um desempenho inferior em comparação com outros testados em ambos os testes. O peso das amostras (*weights*) foi definido como *uniform* para os testes manuais, o que significa que o modelo atribui o mesmo peso para todos os vizinhos. Já para os testes automáticos, foi utilizado o valor *distance*, que atribui pesos maiores para vizinhos mais próximos. Por fim, as métricas de distância (*metric*) utilizadas foram *Minkowski* para o cálculo dos vizinhos mais próximos nos testes manuais, e *manhattan* quando utilizada a técnica do *Grid Search*.

O modelo RF foi testado com diferentes configurações. Nos testes manuais, o número de árvores foi definido como 10, pois valores maiores resultavam em modelos superajustados, levando a problemas de *overfitting* observados. Já com o uso do *Grid Search*, foi utilizado o valor 200. A profundidade máxima das árvores (*max_depth*) não foi definida, permitindo que elas crescessem completamente em ambos os casos. O número mínimo de amostras para dividir um nó (*min_samples_split*) foi mantido em 2, e o número mínimo de amostras em uma folha (*min_samples_leaf*) foi definido como 1 nos testes manuais e como 2 nos testes automáticos.

Já o modelo DT utilizou o critério de impureza *Gini*, sem restrição de profundidade máxima na Tabela 3 e como 10 na Tabela 4. O número mínimo de amostras para dividir um nó (*min_samples_split*) foi mantido em 2 em ambas as tabelas e o número mínimo

Modelo	Parâmetros	Combinações	Combinação final
SVM	C	0.1, 1, 10, 100, 200	200
	kernel	*linear, rbf, poly*	*linear*
	gamma	*scale, auto*	*scale*
KNN	*metric*	*euclidean, manhattan, minkowski*	*manhattan*
	weights	*uniform, distance*	*distance*
	n_neighbors	3, 5, 7, 9, 11	3
RF	*n_estimators*	10,50,100,200	200
	max_depth	*None*, 10,20,30	*None*
	min_samples_split	2,5,10	2
	min_samples_leaf	1,2, 4	2
DT	*max_depth*	*None*, 10, 20 ,30	10
	min_samples_split	2,5,10	2
	min_samples_leaf	1,2,4	2
LR	C	0.01, 0.1, 1, 10, 100	100
	penalty	L1, L2	L1
	solver	*liblinear, saga*	liblinear

Tabela 4 – Hiperparâmetros testados para cada modelo, combinações avaliadas e melhor resultado obtido utilizando a técnica *Grid Search*. Fonte: Elaborada pelo autor.

de amostras em uma folha (*min_samples_leaf*) foi 1, ambos seguindo os valores padrão do algoritmo para os testes manuais e como 2 para os automáticos.

Por fim, o modelo LR foi configurado com regularização do tipo *L2 (Ridge)* nos testes manuais, ou seja, quando a penalidade é aplicada sobre os coeficientes quadráticos. Já na Tabela 4, foi utilizada a regularização *L1*, em que a penalidade é aplicada sobre os valores absolutos dos coeficientes. O coeficiente de regularização foi definido como $C = 1.0$ (regularização moderada) nos testes manuais, e como $C = 100$ nos testes automáticos, o que representa uma penalidade menor sobre os coeficientes. O *solver* utilizado foi o *lbfgs* para os testes manuais, adequado para problemas pequenos e médios, com um máximo de 100 iterações. Nos testes automáticos, foi utilizado o *liblinear*, que é eficiente para problemas de classificação binária e permite a utilização da penalidade *L1*.

3.6.1 Definição da arquitetura e parâmetros para a CNN e LSTM

A construção da CNN foi realizada manualmente por meio de testes empíricos representados na Figura 14. Foi utilizada a camada convolucional 1D, onde o número de filtros foi estabelecido em 60 nas duas primeiras camadas e 100 na última. Essa escolha foi baseada na observação de que um número maior de filtros aumenta o tempo de processamento e pode comprometer o desempenho do treinamento.

O tamanho do *kernel* foi definido como 3, pois apresentou melhor desempenho ao longo das épocas. A função de ativação utilizada foi a ReLU, devido à sua eficácia na propagação

do gradiente e na aceleração do treinamento. Após cada camada convolucional, foi inserida uma camada de *Batch Normalization* para estabilizar a distribuição dos dados e acelerar a convergência do modelo. Em seguida, aplicou-se uma camada de *MaxPooling1D* com fator de redução 2, visando diminuir a dimensionalidade e capturar as características mais relevantes do sinal.

A estrutura da CNN consistiu em três camadas convolucionais seguidas por uma camada *Flatten*, que transforma a matriz de saída em um vetor unidimensional. Em seguida, foram adicionadas três camadas densas com 30, 15 e 2 neurônios, respectivamente. As duas primeiras utilizam a função de ativação ReLU, enquanto a última camada, responsável pela classificação, utiliza a ativação *Softmax*, apropriada para problemas de classificação binária com saída probabilística.

O número de épocas para o treinamento foi definido em 15, pois um valor menor resultava em um desempenho insatisfatório em termos de perda e acurácia. Por outro lado, aumentar excessivamente o número de épocas levou à estagnação do modelo, sem ganhos significativos.

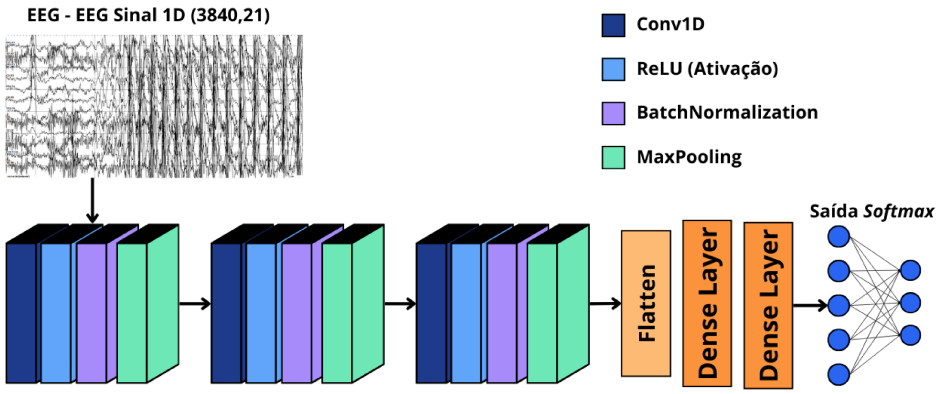

Figura 14 – Arquitetura da CNN usada para o treinamento de dados do EEG. Fonte: Elaborada pelo autor.

Já o modelo de LSTM foi definido manualmente, sendo composto por duas camadas de LSTM. Inicialmente, foram realizados testes com apenas uma camada, mas os melhores resultados foram obtidos com duas. Não foi possível acrescentar mais camadas devido ao alto tempo de processamento exigido por redes LSTM. Além disso, foi adicionada uma camada densa e uma camada de *dropout* para evitar o *overfitting*. Para a camada de saída, utilizou-se a função de ativação *softmax*, conforme ilustrado na Figura 15. O número de épocas foi fixado em 20.

3.6.2 Resultados finais

Os modelos utilizados para avaliar e definir os melhores algoritmos de classificação foram a RF, DT, KNN, SVM e LR. Os modelos foram aplicados utilizando a biblioteca

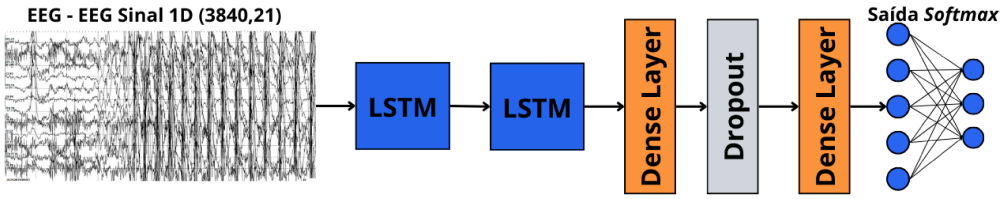

Figura 15 – Arquitetura da LSTM usada para o treinamento de dados do EEG. Fonte: Elaborada pelo autor.

Sckit learning, que contém os algoritmos para treinamento. As métricas avaliadas foram a acurácia, precisão, *Recall* e *F1-score* a partir do conjunto de teste. Além disso, foi considerado dois tipos de classificação: com a escolha dos parâmetros de forma manual e de forma automática.

A classificação foi realizada com o objetivo de identificar os diferentes estágios de uma crise convulsiva, especificamente os períodos ictal, pré-ictal e interictal, por meio de combinações binárias entre essas classes. O período ictal corresponde ao momento em que a crise epiléptica está ocorrendo; o pré-ictal refere-se ao intervalo anterior à crise, durante o qual podem surgir sinais que antecedem o episódio; e o interictal representa o estado entre as crises, quando o cérebro não está passando por uma atividade epiléptica anormal. A transição do estado interictal para o pré-ictal apresenta especial relevância, uma vez que sua detecção pode possibilitar a identificação de uma crise próxima de ocorrer.

Os resultados realizados no primeiro momento com o uso de parâmetros definidos manualmente são mostrados na Tabela 5. Nela, é apresentada a média ponderada, onde é calculada a média ponderada pelo número de exemplos de cada classe para refletir a proporção de exemplos de cada classe.

Os modelos baseados em árvores, RF e DT, apresentaram os melhores resultados, conforme ilustrado na Tabela 5. O RF obteve a maior acurácia em todas as comparações realizadas, demonstrando sua eficácia na classificação dos dados. O DT também apresentou um desempenho elevado, mas ligeiramente inferior ao RF. Por outro lado, os modelos KNN e SVM tiveram desempenhos inferiores, com valores de acurácia mais baixos. A LR, embora tenha superado o KNN em algumas comparações, também não alcançou os mesmos níveis de precisão dos modelos baseados em árvores, o que apresenta uma dificuldade maior para realizar a predição entre as classes.

A métrica precisão, que mede a proporção de previsões corretas em relação ao total de previsões realizadas, seguiu um padrão semelhante ao da acurácia, mantendo a mesma ordem de melhores entre os modelos. O *recall*, que avalia a capacidade do modelo de identificar corretamente todas as instâncias positivas, também apresentou resultados coerentes, evidenciando que os modelos com maior acurácia também tiveram um bom desempenho nessa métrica.

Já o *F1-score*, que combina precisão e *recall* para fornecer uma avaliação mais equili-

Classes	Modelo	Acurácia	Precisão	Recall	F1-score
Interictal e Pré-ictal	**RF**	**0,85**	**0,85**	**0,85**	**0,85**
	DT	0,79	0,79	0,79	0,79
	KNN	0,63	0,65	0,63	0,64
	SVM	0,71	0,70	0,71	0,69
	LR	0,68	0,65	0,68	0,61
Interictal e Ictal	**RF**	**0,92**	**0,91**	**0,91**	**0,91**
	DT	0,90	0,90	0,90	0,90
	KNN	0,76	0,76	0,76	0,76
	SVM	0,82	0,83	0,82	0,82
	LR	0,82	0,82	0,82	0,82
Pré-ictal e Ictal	**RF**	**0,93**	**0,93**	**0,93**	**0,93**
	DT	0,87	0,87	0,87	0,87
	KNN	0,76	0,77	0,76	0,76
	SVM	0,82	0,82	0,82	0,82
	LR	0,84	0,83	0,84	0,83

Tabela 5 – Resultado da classificação com o uso da parametrização manual. Fonte: Elaborada pelo autor.

brada do modelo, reforçou a superioridade dos modelos baseados em árvores. Os modelos RF e DT apresentaram os maiores valores de *F1-score* em todas as comparações, enquanto os modelos KNN, SVM e LR tiveram desempenhos inferiores.

Esses resultados indicam que os modelos baseados em árvores foram mais eficazes na classificação dos diferentes estados dos sinais analisados, enquanto os modelos baseados em vizinhos próximos e regressão logística tiveram dificuldades em alcançar o mesmo nível de desempenho.

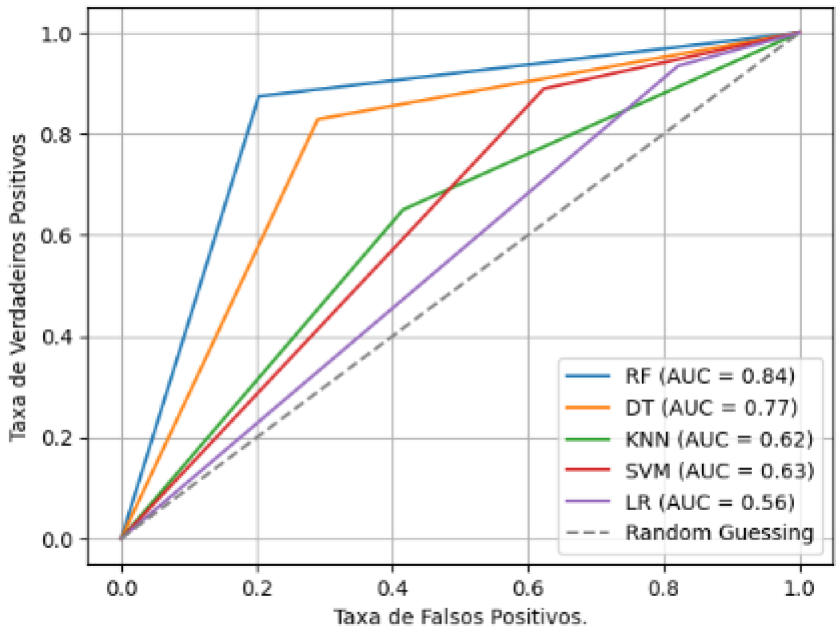

Figura 16 – Curva de ROC das classes pré-ictal e interictal. Fonte: Elaborada pelo autor.

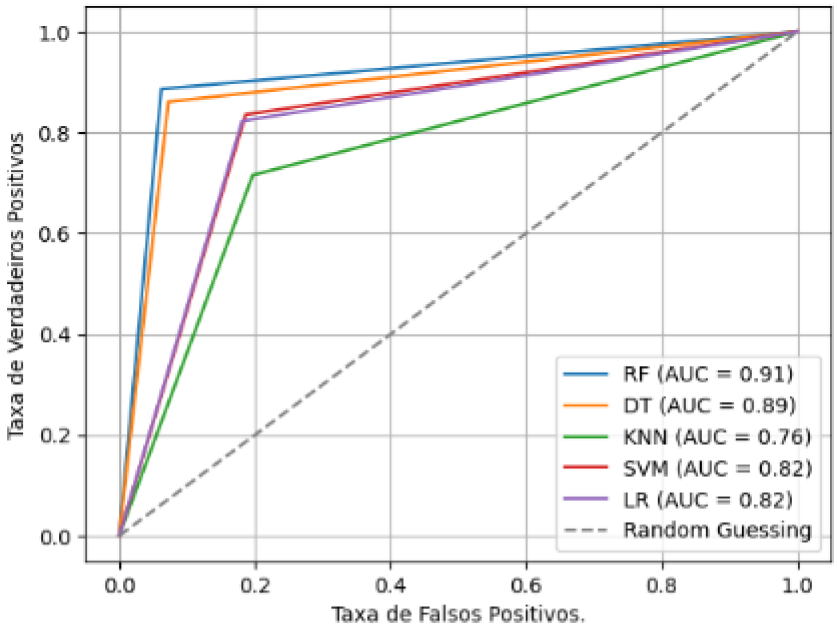

Figura 17 – Curva de ROC das classes ictal e interictal. Fonte: Elaborada pelo autor.

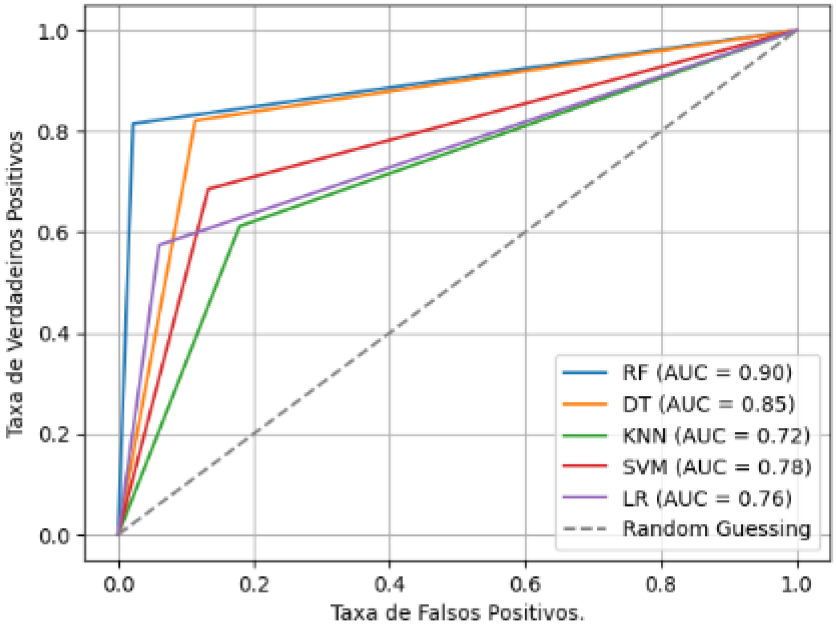

Figura 18 – Curva de ROC das classes ictal e pré-ictal. Fonte: Elaborada pelo autor.

As Figuras 16, 17 e 18 exibem a curva ROC, utilizada para avaliar melhor as métricas. Os pontos mais próximos de (0,1) representam o melhor desempenho do modelo. Dessa forma, os modelos apresentados nas figuras podem ser ordenados da seguinte maneira: o RF em primeiro lugar, seguido do DT, SVM, KNN e, por fim, o LR.

Os modelos apresentaram um desempenho superior na distinção entre as classes do período ictal e interictal. O modelo RF obteve a maior acurácia em todas as comparações, alcançando 0,93 para a distinção entre as classes pré-ictal e ictal. Já os modelos baseados em árvores, RF e DT, destacaram-se com os melhores resultados em todas as classificações.

Por outro lado, a classificação entre as classes interictal e pré-ictal apresentou menor desempenho. Modelos como KNN, SVM e LR foram mais impactados pela possível presença de *outliers*, o que resultou em valores de Acurácia abaixo de 0,70 para algumas comparações.

Classes	Modelo	Acurácia	Precisão	Recall	F1-score
Interictal e Pré-ictal	**RF**	**0,89**	**0,89**	**0,89**	**0,89**
	DT	0,81	0,82	0,81	0,82
	KNN	0,68	0,70	0,68	0,69
	SVM	0,76	0,75	0,76	0,75
	LR	0,75	0,74	0,75	0,74
Interictal e Ictal	**RF**	**0,94**	**0,94**	**0,94**	**0,94**
	DT	0,89	0,89	0,89	0,89
	KNN	0,83	0,83	0,83	0,83
	SVM	0,89	0,89	0,89	0,89
	LR	0,91	0,91	0,91	0,91
Pré-ictal e Ictal	**RF**	**0,94**	**0,94**	**0,94**	**0,93**
	DT	0,88	0,88	0,88	0,88
	KNN	0,83	0,83	0,83	0,83
	SVM	0,91	0,91	0,91	0,91
	LR	0,92	0,92	0,92	0,92

Tabela 6 – Resultado da classificação com o uso do *GrindSearch* para a parametrização. Fonte: Elaborada pelo autor.

A Tabela 6 apresenta os resultados dos modelos de classificação utilizando a parametrização do *Grid Search* do *Python* para encontrar as melhores combinações de parâmetros. Nessa tabela, é possível observar uma melhoria nos avaliadores de desempenho em relação à Tabela 5, além de mudanças nos melhores algoritmos. Após o ajuste dos parâmetros, o desempenho dos classificadores KNN, LR e SVM apresentou uma melhora significativa.

Em relação à ordem de desempenho dos classificadores , o RF continua apresentando os melhores resultados, enquanto o DT oscila entre a segunda posição, junto com os classificadores LR e SVM. Por fim, o KNN mantém o pior desempenho entre os modelos avaliados. O uso da automação na seleção dos conjuntos de parâmetros teve um impacto significativo nos modelos LR, SVM e KNN, promovendo melhorias nas métricas de avaliação — em alguns casos, superiores a 10%. Para os classificadores RF e DT, também houve uma pequena melhora.

As Figuras 19, 20 e 21, que apresentam a curva ROC, também demonstraram uma melhoria no desempenho dos classificadores, com uso automático na escolha dos parâmetros, especialmente para o LR, SVM e KNN.

Este trabalho também explorou o uso de DLs para a classificação, especificamente as arquiteturas CNN e LSTM. Esses modelos possuem camadas capazes de extrair características automaticamente e resolver problemas complexos a partir de dados brutos. Por esse motivo, a etapa de extração e seleção de características, abordada na Seção 3.4, foi

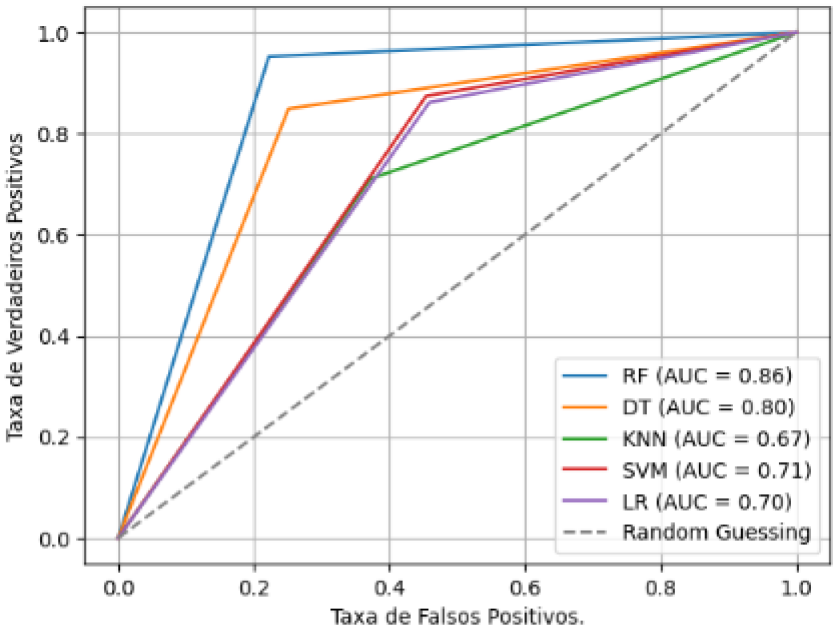

Figura 19 – Curva de ROC das classes pré-ictal e interictal. Fonte: Elaborada pelo autor.

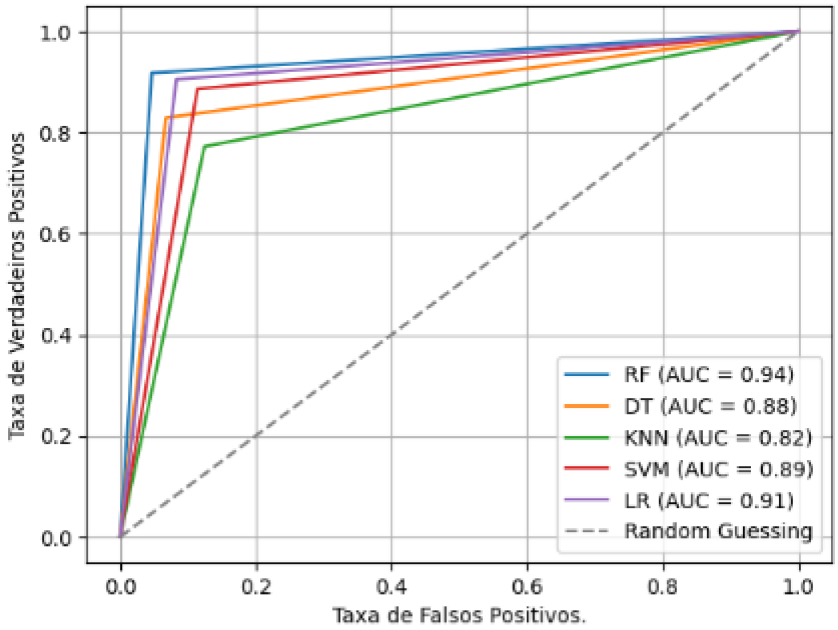

Figura 20 – Curva de ROC das classes ictal e interictal. Fonte: Elaborada pelo autor.

removida do processo, uma vez que os modelos de DL já realizam essa função. Além disso, esses modelos exigem um grande volume de dados para um treinamento eficaz. Na tabela 7 é possível visualizar a acurácia e a perda final dos algoritmos.

Ao observar a Tabela, percebe-se que a CNN apresenta um desempenho superior em comparação aos classificadores de ML, alcançando uma Acurácia próxima ou igual a 100% para todas as combinações de classes. Por outro lado, a LSTM demonstrou dificuldades no treinamento das classes Interictal e Pré-ictal, assim como Ictal e Pré-ictal. No entanto, para as classes Interictal e Ictal, a acurácia foi alta, acompanhada de uma perda baixa.

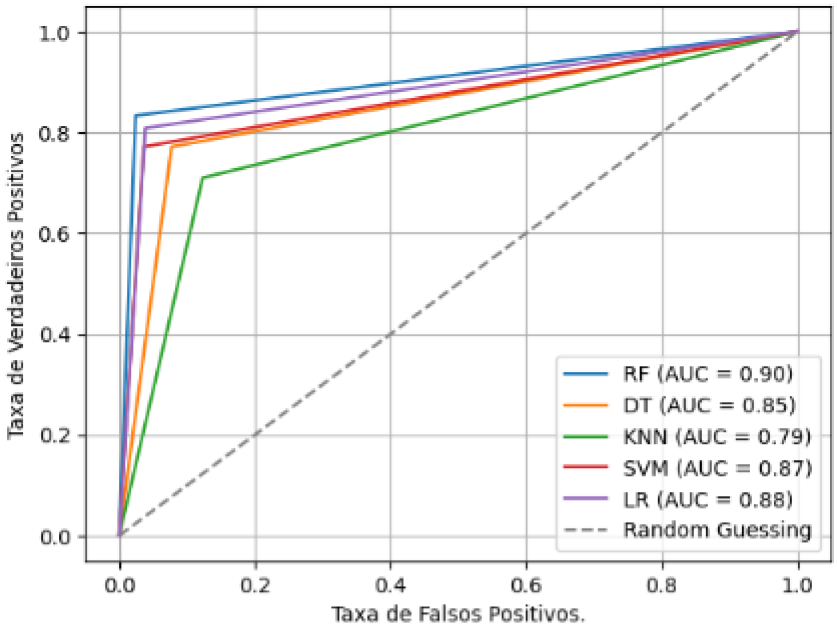

Figura 21 – Curva de ROC das classes ictal e pré-ictal. Fonte: Elaborada pelo autor.

Classes	Modelo	Acurácia	Perda
Interictal e Pré-ictal	**CNN**	**0.9958**	**0.018**
	LSTM	0.7592	0.5636
Interictal e ictal	**CNN**	**1.00**	**0.0006**
	LSTM	0.9332	0.221
ictal e Pré-ictal	**CNN**	**0.9807**	**0.0521**
	LSTM	0.7263	0.5734

Tabela 7 – Resultado da classificação usando os modelos DL CNN e LSTM. Fonte: Elaborada pelo autor.

As Figuras 22, 23 e 24 mostram a performance do treinamento do modelo CNN ao longo de 15 épocas. É possível observar um aprendizado progressivo do modelo, onde a acurácia tende a aumentar e a perda a diminuir, indicando previsões mais precisas.

As Figuras 25, 26 e 27 ilustram a performance do modelo LSTM ao longo de 20 épocas de treinamento. A Figura 25, que representa a predição do período ictal e interictal, apresenta um treinamento progressivo, onde a acurácia tende a aumentar e a perda a diminuir. No entanto, nas Figuras 26 e 27, apesar de inicialmente exibirem um crescimento progressivo, observa-se que, ao final do treinamento, há uma tendência de redução na acurácia e aumento na perda em relação às épocas anteriores. Isso ocorre porque a acurácia e a perda de validação são consideradas ao final da predição, indicando uma dificuldade do modelo em generalizar para novos dados.

A Tabela 8 apresenta as maiores acurácias obtidas durante o treinamento, considerando as diferentes combinações de classes e modelos de classificação. Observa-se que a combinação entre as classes ictal e interictal obteve os melhores resultados em compara-

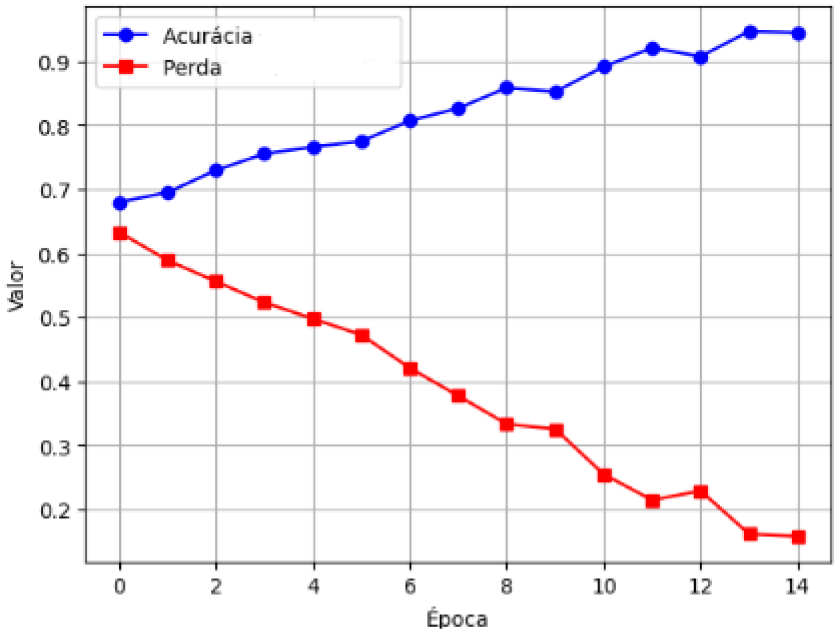

Figura 22 – Performance da CNN ao longo de 15 épocas das classes pré-ictal e interictal. Fonte: Elaborada pelo autor.

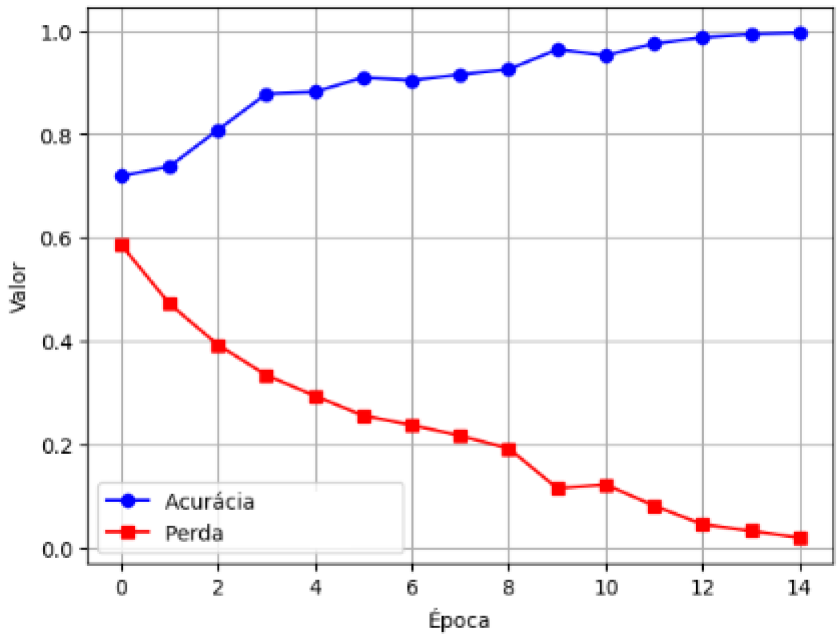

Figura 23 – Performance da CNN ao longo de 15 épocas das classes ictal e pré-ictal. Fonte: Elaborada pelo autor.

ção com a maioria das outras combinações. Esse desempenho superior pode ser atribuído à clara diferenciação entre os sinais característicos dessas duas classes, o que facilita a tarefa dos classificadores.

Além disso, nota-se que a CNN obteve o melhor desempenho em todas as combinações de classes, sendo seguida pelos classificadores RF, DT, LR, SVM e, por último, KNN. A LSTM, por sua vez, demonstrou dificuldades principalmente ao classificar as combinações

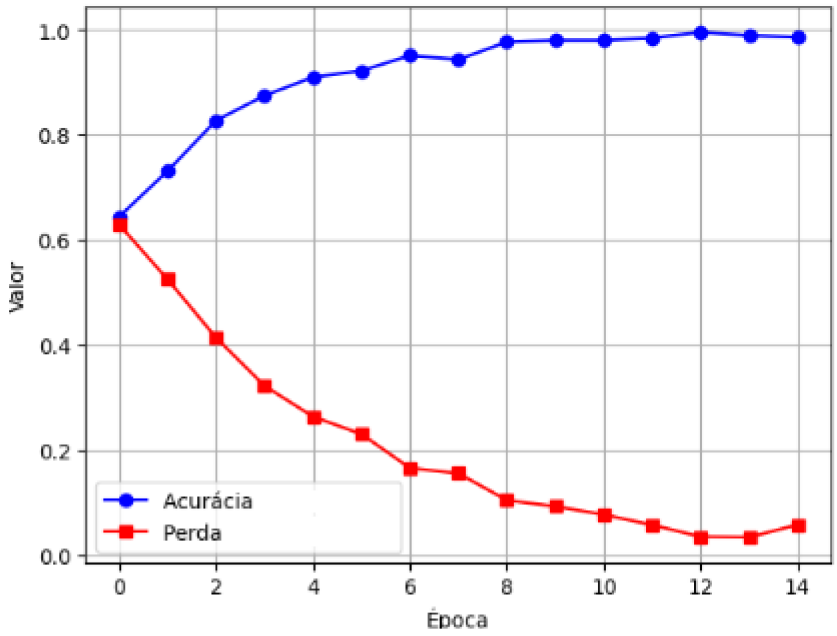

Figura 24 – Performance da CNN ao longo de 15 épocas das classes ictal e interictal. Fonte: Elaborada pelo autor.

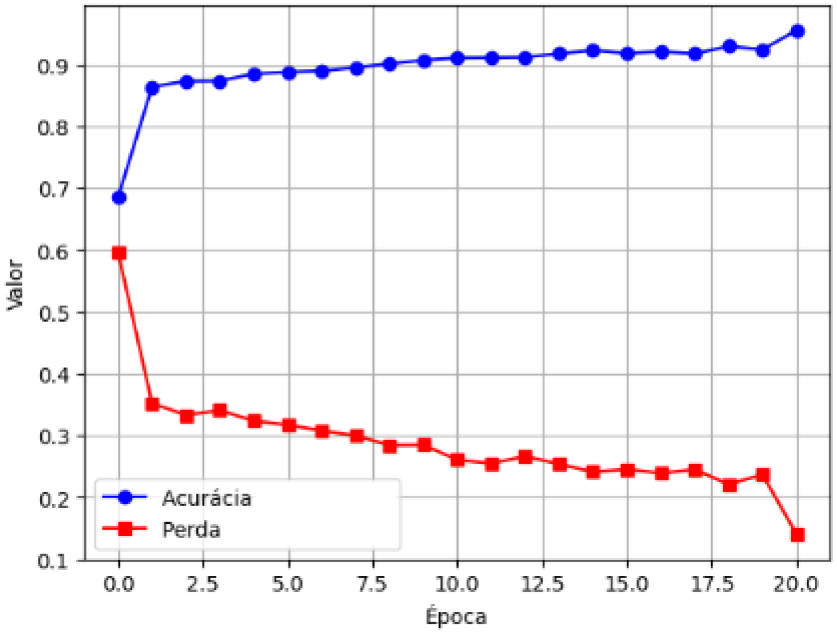

Figura 25 – Performance da LSTM ao longo de 20 épocas das classes ictal e interictal. Fonte: Elaborada pelo autor.

binárias interictal e pré-ictal, bem como pré-ictal e ictal, o que pode indicar a necessidade de ajustes em sua arquitetura ou no pré-processamento dos dados para esses casos.

De maneira geral, a maioria dos classificadores foi capaz de atingir um bom desempenho, com baixas taxas de erro, o que demonstra uma boa capacidade de generalização para novos dados. Esses resultados reforçam a eficácia dos modelos propostos para a tarefa de classificação dos diferentes estados cerebrais associados à epilepsia.

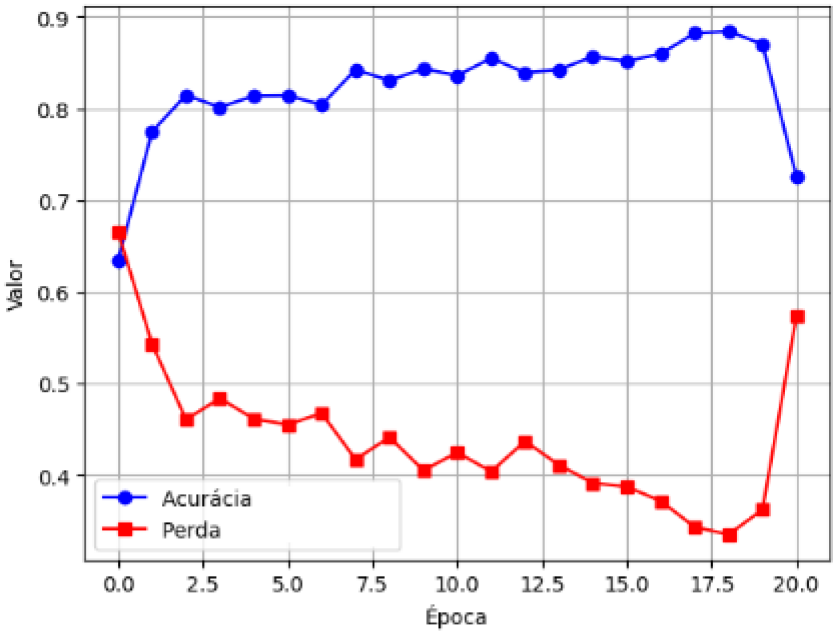

Figura 26 – Performance da LSTM ao longo de 20 épocas das classes ictal e pre-ictal. Fonte: Elaborada pelo autor.

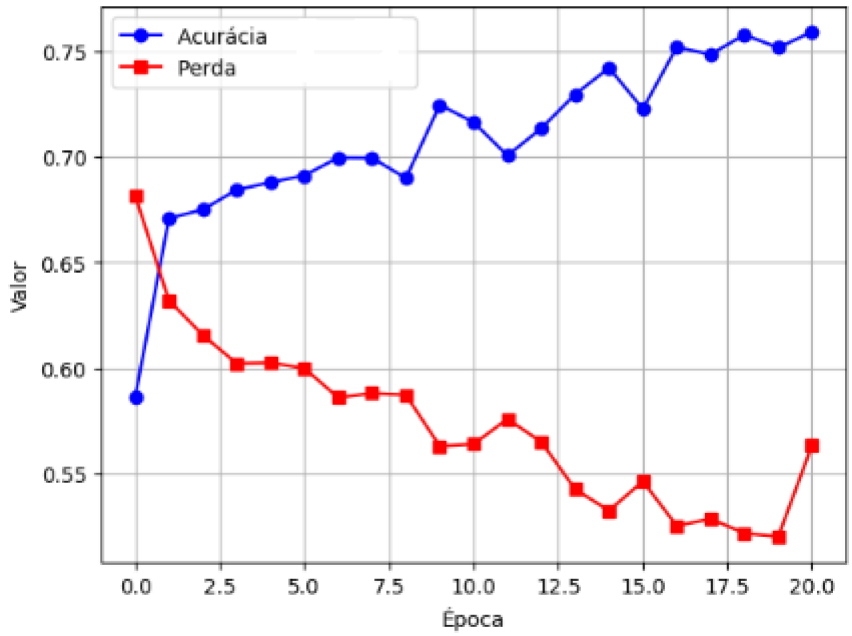

Figura 27 – Performance da LSTM ao longo de 20 épocas das classes pre-ictal e interictal. Fonte: Elaborada pelo autor.

3.7 Avaliação Comparativa com a Literatura

A Tabela 9 apresenta uma comparação entre os resultados obtidos neste trabalho e os estudos apresentados na Seção 2.5, que abordam a classificação da epilepsia por meio da análise de sinais de EEG. Esses estudos utilizaram diferentes bases de dados, métricas e algoritmos de *Machine Learning*, além de variarem quanto às classes avaliadas,

Classes	Modelo	Acurácia
Interictal e Pré-ictal	RF	0,89
	DT	0,81
	KNN	0,68
	SVM	0,76
	LR	0,75
	LSTM	0,7592
	CNN	**0,9958**
Interictal e Ictal	RF	0,94
	DT	0,90
	KNN	0,83
	SVM	0,89
	LR	0,91
	LSTM	0,9332
	CNN	**1,00**
Pré-ictal e Ictal	RF	0,94
	DT	0,88
	KNN	0,83
	SVM	0,91
	LR	0,92
	LSTM	0,7263
	CNN	**0,9807**

Tabela 8 – Acurácia dos modelos por combinação de classes. Fonte: Elaborada pelo autor.

como os períodos ictal, pré-ictal e interictal, ou ainda quanto ao tipo de crise, diferenciando entre crises focais e generalizadas. Na Tabela 9, são apresentados os resultados referentes aos períodos ictal e interictal, que apresentam padrões de sinal mais distintos e, consequentemente, melhor desempenho em relação aos outros períodos analisados neste trabalho.

Os modelos desenvolvidos, com destaque para a CNN e a LSTM, apresentaram ótimo desempenho, igualando ou superando os resultados obtidos em estudos anteriores. Além disso, foram aplicados algoritmos como a Regressão Logística (LR) e a CNN, que não foram explorados por nenhum dos trabalhos comparados.

Referência	Base de dados	Classe	Métrica	Modelo	Resultado
(MESSAOUD; CHAVEZ, 2021)	CHB-MIT Scalp EEG Database	Pré-ictal e interictal	Sensibilidade	RF	0,95
				MLP	0,83
				KNN	0,82
				AdaBoost	0,87
				SVM	0,90
(LESTARI et al., 2020)	CHB-MIT Scalp EEG Database	Ictal e interictal	Precisão	KNN	0,927
				RF	0,866
				Naive Bayes	0,556
(TUNCER; BOLAT, 2022)	Temple University Hospital	Crises focais e generalizadas	Acurácia	RF	0,9545
				SVM	0,914
				KNN	0,9575
				LSTM	0,9592
(TRAN et al., 2022)	Universidade de Bonn	Ictal e interictal	Acurácia	SVM	0,992
				KNN	0,986
				DT	0,973
				RF	0,984
(SAVADKOOHI; OLADUNNI; THOMPSON, 2020)	Universidade de Bonn	Ictal e interictal	Precisão	SVM	1,00
				KNN	0,995
Este trabalho	CHB-MIT Scalp EEG Database	Ictal e Interictal	Acurácia	LR	0,91
				SVM	0,89
				KNN	0,83
				DT	0,90
				RF	0,94
				LSTM	0,9332
				CNN	1,00

Tabela 9 – Comparativo entre este estudo e trabalhos anteriores quanto à base de dados, classes, métricas, modelos e resultados obtidos. Fonte: Elaborada pelo autor.

Capítulo 4

Conclusão

Esta pesquisa teve como objetivo testar, analisar e comparar diferentes métodos de aprendizado de máquina e aprendizado profundo na análise de dados de EEG para a detecção dos estados presentes em convulsões epilépticas.

Após a aplicação das etapas de classificação, que incluíram a seleção da base de dados, pré-processamento, extração e seleção de características relevantes, escolha dos modelos e testes com diferentes parâmetros, foi possível identificar os algoritmos com melhor desempenho. Os modelos que obtiveram as maiores taxas de precisão e métricas de avaliação na classificação do período ictal foram a CNN, RF, LSTM, LR, SVM e KNN.

Entretanto, algumas limitações foram identificadas, principalmente em relação à definição dos períodos interictal e pré-ictal, uma vez que a base de dados utilizada disponibiliza apenas a marcação do período ictal. Além disso, a disponibilidade de repositórios públicos com dados de EEG para epilepsia ainda é limitada, o que pode impactar o desenvolvimento e aprimoramento de novos modelos.

Dessa forma, os resultados obtidos confirmam a hipótese inicial deste trabalho, indicando que os algoritmos de aprendizado de máquina e aprendizado profundo podem, de fato, automatizar com precisão a detecção de crises epilépticas a partir de sinais de EEG. As técnicas de aprendizado profundo apresentaram desempenho superior na classificação dos estados epilépticos, sugerindo que esses métodos são promissores para futuras aplicações na detecção automática de crises.

4.1 Principais Contribuições

Os modelos de classificação apresentaram um bom desempenho em relação à acurácia, precisão, *f1-score* e *recall*. Além disso, os modelos de ML também demonstraram bons resultados em termos de acurácia e perda, permitindo a comparação entre diferentes algoritmos e a definição daqueles mais adequados para a predição dos estágios de uma crise epiléptica.

Dentre os resultados, destaca-se a capacidade dos modelos em identificar o período pré-ictal, possibilitando a detecção antecipada do início de um episódio epiléptico por meio de técnicas de aprendizado de máquina e aprendizado profundo. Além disso, este estudo pode servir como base para pesquisas futuras, contribuindo para o avanço na detecção e previsão de crises epilépticas.

4.2 Trabalhos Futuros

Neste trabalho, os resultados obtidos na Seção 3.6.2 apresentaram um bom desempenho para as combinações binárias das classes ictal, interictal e pré-ictal. Para aprimorar essa classificação, poderia ser aplicada a extração de características mais relevantes.

Para trabalhos futuros, uma boa prática seria utilizar outras bases de dados para a predição, além de explorar combinações que permitam avaliar a classificação em um cenário multiclasse.

Outra abordagem promissora seria o uso de diferentes métodos de classificação, como as *Graph Neural Networks* (GNNs). Essas redes são capazes de capturar as complexas relações espaciais e temporais presentes nos sinais de EEG, levando em consideração tanto a estrutura dos sinais quanto a disposição dos eletrodos.

Além disso, a incorporação de mecanismos de atenção às GNNs representa uma combinação poderosa. A adição de uma camada de atenção não apenas permite a memorização de longas sequências de informações, mas também melhora a captura de relações complexas e de longo alcance entre os nós do grafo, potencialmente elevando a precisão do modelo.

Referências

ABDULLAH, D. M.; ABDULAZEEZ, A. M. Machine learning applications based on svm classification a review. **Qubahan Academic Journal**, v. 1, n. 2, p. 81–90, 2021. Disponível em: <https://doi.org/10.48161/qaj.v1n2a50>. Citado na página 24.

ABRO, S. et al. Aspect based sentimental analysis of hotel reviews: A comparative study. **Sukkur IBA Journal of Computing and Mathematical Sciences**, v. 4, n. 1, p. 11–20, 2020. Citado na página 24.

ALMEIDA, R. A utilidade do eeg. **Revista Portuguesa de Medicina Geral e Familiar**, v. 21, n. 3, p. 301–306, 2005. Citado 2 vezes nas páginas 12 e 16.

ANDRZEJAK, R. et al. Indications of nonlinear deterministic and finite dimensional structures in time series of brain electrical activity: Dependence on recording region and brain state. **Phys. Rev. E**, v. 64, p. 061907, 2001. Disponível em: <https://doi.org/10.1103/PhysRevE.64.061907>. Citado 2 vezes nas páginas 35 e 36.

AYADI, W. et al. Deep cnn for brain tumor classification. **Neural processing letters**, Springer, v. 53, p. 671–700, 2021. Disponível em: <https://doi.org/10.1007/s11063-020-10398-2>. Citado na página 32.

BAŞAR, E. et al. Gamma, alpha, delta, and theta oscillations govern cognitive processes. **International journal of psychophysiology**, Elsevier, v. 39, n. 2-3, p. 241–248, 2001. Disponível em: <https://doi.org/10.1016/S0167-8760(00)00145-8>. Citado na página 18.

BATISTA, G. E. d. A. P. et al. **Pré-processamento de dados em aprendizado de máquina supervisionado**. Tese (Doutorado) — Universidade de São Paulo, 2003. Disponível em: <https://doi.org/10.11606/T.55.2003.tde-06102003-160219>. Citado na página 21.

BBRINKM; CUKIERSKI, W. **American Epilepsy Society Seizure Prediction Challenge**. 2014. Kaggle. Disponível em: <https://kaggle.com/competitions/seizure-prediction>. Citado na página 35.

BEHERA, R. K. et al. Co-lstm: Convolutional lstm model for sentiment analysis in social big data. **Information Processing & Management**, Elsevier, v. 58, n. 1, p. 102435, 2021. Disponível em: <https://doi.org/10.1016/j.ipm.2020.102435>. Citado na página 32.

BHATT, D. et al. Cnn variants for computer vision: History, architecture, application, challenges and future scope. **Electronics**, MDPI, v. 10, n. 20, p. 2470, 2021. Disponível em: <https://doi.org/10.3390/electronics10202470>. Citado na página 33.

BISHOP, C. M. **Pattern Recognition and Machine Learning**. New York: Springer, 2006. Citado na página 30.

CADENAS, J. M. et al. A fuzzy k-nearest neighbor classifier to deal with imperfect data. **Soft Computing**, Springer, v. 22, p. 3313–3330, 2018. Disponível em: <https://doi.org/10.1007/s00500-017-2567-x>. Citado na página 25.

CERVANTES, J. et al. A comprehensive survey on support vector machine classification: Applications, challenges and trends. **Neurocomputing**, Elsevier, v. 408, p. 189–215, 2020. Disponível em: <https://doi.org/10.1016/j.neucom.2019.10.118>. Citado na página 24.

CHEN, H.; HAOYU, C. Face recognition algorithm based on vgg network model and svm. In: IOP PUBLISHING. **Journal of Physics: Conference Series**. 2019. v. 1229, n. 1, p. 012015. Disponível em: <https://doi.org/10.1088/1742-6596/1229/1/012015>. Citado na página 23.

CUNNINGHAM, P.; DELANY, S. J. k-nearest neighbour classifiers: (with python examples). **arXiv preprint arXiv:2004.04523**, 2020. Disponível em: <https://doi.org/10.48550/arXiv.2004.04523>. Citado na página 26.

DANTAS, F. G. et al. Papel do eeg em casos de suspeita ou diagnóstico de epilepsia. **Journal of Epilepsy and Clinical Neurophysiology**, v. 11, p. 77–78, 2005. Disponível em: <https://doi.org/10.1590/S1676-26492005000200002>. Citado na página 12.

DESCOVI, C. S. et al. Utilizing long short-term memory (lstm) networks for river flow prediction in the brazilian pantanal basin. **Holos**, v. 5, n. 39, 2023. Disponível em: <https://doi.org/10.15628/holos.2023.16315>. Citado na página 34.

ESCOVEDO, T.; KOSHIYAMA, A. **Introdução a Data Science: Algoritmos de Machine Learning e métodos de análise**. [S.l.]: Casa do Código, 2020. Citado na página 21.

FACELI, K. et al. **Inteligência Artificial - Uma Abordagem de Aprendizado de Máquina**. Rio de Janeiro: LTC, 2021. E-book. ISBN 9788521637509. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788521637509/>. Citado 11 vezes nas páginas 5, 19, 20, 21, 22, 23, 24, 25, 26, 28 e 31.

FAN, G.-F. et al. Application of the weighted k-nearest neighbor algorithm for short-term load forecasting. **Energies**, MDPI, v. 12, n. 5, p. 916, 2019. Disponível em: <https://doi.org/10.3390/en12050916>. Citado 2 vezes nas páginas 25 e 26.

FERRAG, M. A. et al. Rdtids: Rules and decision tree-based intrusion detection system for internet-of-things networks. **Future internet**, MDPI, v. 12, n. 3, p. 44, 2020. Disponível em: <https://doi.org/10.3390/fi12030044>. Citado na página 27.

FERREIRA, L. S. et al. **Manual do Técnico em EEG**. E-book. Rio de Janeiro: Thieme Revinter, 2023. Acesso em: 06 out. 2024. ISBN 9786555721744. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9786555721744/>. Citado 5 vezes nas páginas 15, 16, 17, 18 e 19.

FREEMAN, W.; QUIROGA, R. **Imaging brain function with EEG: advanced temporal and spatial analysis of electroencephalographic signals**. S.l.: Springer Science & Business Media, 2012. Disponível em: <https://doi.org/10.1007/978-1-4614-4984-3>. Citado na página 13.

GABRIEL, M. **Inteligência Artificial: Do Zero ao Metaverso**. Rio de Janeiro: Atlas, 2022. E-book. ISBN 9786559773336. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9786559773336/>. Citado 3 vezes nas páginas 19, 30 e 31.

GAO, Y.; SUN, S. An empirical evaluation of linear and nonlinear kernels for text classification using support vector machines. In: **2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery**. [s.n.], 2010. v. 4, p. 1502–1505. Disponível em: <https://doi.org/10.1109/FSKD.2010.5569327>. Citado na página 23.

GASTAUT, H. J. G. G. et al. Relative frequency of different types of epilepsy: a study employing the classification of the international league against epilepsy. **Epilepsia**, v. 16, n. 3, p. 457–461, 1975. Disponível em: <https://doi.org/10.1111/j.1528-1157.1975.tb06073.x>. Citado na página 12.

GERS, F. A.; SCHMIDHUBER, J.; CUMMINS, F. Learning to forget: Continual prediction with lstm. **Neural computation**, MIT press, v. 12, n. 10, p. 2451–2471, 2000. Disponível em: <https://doi.org/10.1162/089976600300015015>. Citado na página 34.

GOLDBERGER, A. et al. Physiobank, physiotoolkit, and physionet: Components of a new research resource for complex physiologic signals. **Circulation [Online]**, v. 101, n. 23, p. e215–e220, 2000. Disponível em: <https://doi.org/10.1161/01.CIR.101.23.e215>. Citado 2 vezes nas páginas 35 e 37.

GUPTA, V. et al. A fuzzy rule-based system with decision tree for breast cancer detection. **IET Image Processing**, Wiley Online Library, v. 17, n. 7, p. 2083–2096, 2023. Disponível em: <https://doi.org/10.1049/ipr2.12774>. Citado na página 27.

GUTTAG, J. **CHB-MIT Scalp EEG Database (version 1.0.0)**. PhysioNet, 2010. Disponível em: <https://doi.org/10.13026/C2K01R>. Citado 2 vezes nas páginas 35 e 37.

GéRON, A. **Mãos à Obra: Aprendizado de Máquina com Scikit-Learn, Keras & TensorFlow: Conceitos, Ferramentas e Técnicas para a Construção de Sistemas Inteligentes**. 2nd. ed. Rio de Janeiro: Editora Alta Books, 2021. E-book, p.48. ISBN 9786555208146. Disponível em: <https://integrada.minhabiblioteca.com.br/reader/books/9786555208146/>. Citado 6 vezes nas páginas 19, 21, 22, 23, 28 e 29.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J. **The Elements of Statistical Learning: Data Mining, Inference, and Prediction**. 2nd. ed. New York: Springer,

2009. Disponível em: <https://doi.org/10.1007/978-0-387-84858-7>. Citado na página 29.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. **Neural computation**, MIT press, v. 9, n. 8, p. 1735–1780, 1997. Disponível em: <https://doi.org/10.1162/neco.1997.9.8.1735>. Citado na página 34.

IBM. **O que é Regressão Logística?** [S.l.], 2024. Accessed: 2024-10-21. Disponível em: <https://www.ibm.com/br-pt/topics/logistic-regression>. Citado 2 vezes nas páginas 29 e 30.

ISMAYILOV, E.; MAMMADOV, R. Parallel solution of features subset selection process for hand-printed character recognition. **Azerbaijan Journal of High Performance Computing**, Azerbaijan Journal of High Performance Computing, v. 2, n. 2, p. 170–177, 2019. Disponível em: <https://doi.org/10.32010/26166127.2019.2.2.170.177>. Citado na página 24.

JOBST, B. C. et al. Highlights from the annual meeting of the american epilepsy society 2018. **Epilepsy currents**, v. 19, n. 3, p. 152–158, 2019. Disponível em: <https://doi.org/10.1177/1535759719844486>. Citado na página 12.

KUMAR, S.; SINGH, S.; KUMAR, J. Multiple face detection using hybrid features with svm classifier. In: SPRINGER. **Data and Communication Networks: Proceedings of GUCON 2018**. 2019. p. 253–265. Disponível em: <https://doi.org/10.1007/978-981-13-2254-9_23>. Citado na página 23.

LESTARI, F. P. et al. Epileptic seizure detection in eegs by using random tree forest, naïve bayes and knn classification. In: IOP PUBLISHING. **Journal of Physics: Conference Series**. 2020. v. 1505, n. 1, p. 012055. Disponível em: <https://doi.org/10.1088/1742-6596/1505/1/012055>. Citado 2 vezes nas páginas 35 e 55.

LIMA, I. **Inteligência Artificial**. GEN LTC, 2014. 2 p. E-book. ISBN 9788595152724. Disponível em: <https://integrada.minhabiblioteca.com.br/reader/books/9788595152724/>. Citado 3 vezes nas páginas 19, 20 e 21.

LIMA, J. M. L. Epilepsia-a abordagem clínica. **Revista Portuguesa de Medicina Geral e Familiar**, v. 21, n. 3, p. 291–8, 2005. Citado 3 vezes nas páginas 12, 13 e 16.

MESSAOUD, R. B.; CHAVEZ, M. Random forest classifier for eeg-based seizure prediction. **arXiv preprint arXiv:2106.04510**, 2021. Citado 2 vezes nas páginas 34 e 55.

MONARD, M. C.; BARANAUSKAS, J. A. Indução de regras e árvores de decisão. **Sistemas Inteligentes. Rezende, SO Editora Manole Ltda**, p. 115–140, 2003. Citado na página 27.

MONTENEGRO, M. A. et al. **EEG na Prática Clínica**. 4th. ed. Thieme Revinter, 2022. 273 p. E-book. ISBN 9786555721607. Disponível em: <https://integrada.minhabiblioteca.com.br/reader/books/9786555721607/>. Citado 2 vezes nas páginas 15 e 18.

OLIVEIRA, S. N. D.; ROSADO, P. Electroencefalograma interictal–sensibilidade e especificidade no diagnóstico de epilepsia. **Acta Médica Portuguesa**, v. 17, n. 6, p. 465–470, 2004. Citado na página 18.

ONU News. **OMS divulga série de ações para melhorar a vida das pessoas com epilepsia**. [S.l.], 2022. 13 dez. 2022. Disponível em: <https://news.un.org/pt/story/2022/12/1910302>. Citado 2 vezes nas páginas 12 e 15.

SAPUTRO, I. R. D. et al. Seizure type classification on eeg signal using support vector machine. In: IOP PUBLISHING. **Journal of Physics: Conference Series**. 2019. v. 1201, n. 1, p. 012065. Disponível em: <https://doi.org/10.1088/1742-6596/1201/1/012065>. Citado na página 24.

SAVADKOOHI, M.; OLADUNNI, T.; THOMPSON, L. A machine learning approach to epileptic seizure prediction using electroencephalogram (eeg) signal. **Biocybernetics and Biomedical Engineering**, Elsevier, v. 40, n. 3, p. 1328–1341, 2020. Disponível em: <https://doi.org/10.1016/j.bbe.2020.07.004>. Citado 2 vezes nas páginas 36 e 55.

SCHONLAU, M.; ZOU, R. Y. The random forest algorithm for statistical learning. **The Stata Journal**, SAGE Publications Sage CA: Los Angeles, CA, v. 20, n. 1, p. 3–29, 2020. Citado na página 28.

Scikit-learn Developers. **User Guide: Scikit-learn 1.5.2 Documentation**. [S.l.], 2024. Accessed: 2024-10-21. Disponível em: <https://scikit-learn.org/1.5/user_guide.html>. Citado 5 vezes nas páginas 25, 27, 28, 29 e 41.

SHAH, K. et al. A comparative analysis of logistic regression, random forest and knn models for the text classification. **Augmented Human Research**, Springer, v. 5, n. 1, p. 12, 2020. Disponível em: <https://doi.org/10.1007/s41133-020-00032-0>. Citado na página 26.

SHARIFANI, K.; AMINI, M. Machine learning and deep learning: A review of methods and applications. **World Information Technology and Engineering Journal**, v. 10, n. 07, p. 3897–3904, 2023. Citado na página 32.

SHOEB, A. **Application of Machine Learning to Epileptic Seizure Onset Detection and Treatment**. Tese (PhD Thesis) — Massachusetts Institute of Technology, September 2009. Citado 2 vezes nas páginas 35 e 37.

SINHA, P.; SINHA, P. et al. Comparative study of chronic kidney disease prediction using knn and svm. **International Journal of Engineering Research and Technology**, v. 4, n. 12, p. 608–12, 2015. Citado na página 26.

SMAGULOVA, K.; JAMES, A. P. A survey on lstm memristive neural network architectures and applications. **The European Physical Journal Special Topics**, Springer, v. 228, n. 10, p. 2313–2324, 2019. Disponível em: <https://doi.org/10.1140/epjst/e2019-900046-x>. Citado na página 34.

SONG, Y. Y.; LU, Y. Decision tree methods: applications for classification and prediction. **Shanghai Archives of Psychiatry**, v. 27, n. 2, p. 130–135, April 25 2015. Disponível em: <https://doi.org/10.11919/j.issn.1002-0829.215044>. Citado 2 vezes nas páginas 27 e 28.

SOUCY, P.; MINEAU, G. W. A simple knn algorithm for text categorization. In: IEEE. **Proceedings 2001 IEEE international conference on data mining**. 2001. p. 647–648. Disponível em: <https://doi.org/10.1109/ICDM.2001.989592>. Citado na página 26.

TAUNK, K. et al. A brief review of nearest neighbor algorithm for learning and classification. In: IEEE. **2019 international conference on intelligent computing and control systems (ICCS)**. 2019. p. 1255–1260. Disponível em: <https://doi.org/10.1109/ICCS45141.2019.9065747>. Citado na página 26.

TRAN, L. V. et al. Application of machine learning in epileptic seizure detection. **Diagnostics**, MDPI, v. 12, n. 11, p. 2879, 2022. Disponível em: <https://doi.org/10.3390/diagnostics12112879>. Citado 2 vezes nas páginas 35 e 55.

TUNCER, E.; BOLAT, E. D. Channel based epilepsy seizure type detection from electroencephalography (eeg) signals with machine learning techniques. **Biocybernetics and Biomedical Engineering**, Elsevier, v. 42, n. 2, p. 575–595, 2022. Disponível em: <http://dx.doi.org/10.1016/j.bbe.2022.04.004>. Citado 2 vezes nas páginas 35 e 55.

UDDIN, S. et al. Comparative performance analysis of k-nearest neighbour (knn) algorithm and its different variants for disease prediction. **Scientific Reports**, Nature Publishing Group UK London, v. 12, n. 1, p. 6256, 2022. Disponível em: <https://doi.org/10.1038/s41598-022-10358-x>. Citado na página 26.

VADALI, S.; DEEKSHITULU, G.; MURTHY, J. Analysis of liver cancer using data mining svm algorithm in matlab. In: SPRINGER. **Soft Computing for Problem Solving: SocProS 2017, Volume 1**. 2019. p. 163–175. Disponível em: <https://doi.org/10.1007/978-981-13-1592-3_12>. Citado na página 23.

VAPNIK, V.; GOLOWICH, S.; SMOLA, A. Support vector method for function approximation, regression estimation and signal processing. **Advances in neural information processing systems**, v. 9, 1996. Citado na página 23.

WANI, M. A. et al. **Advances in deep learning**. Springer, 2020. Disponível em: <https://doi.org/10.1007/978-981-13-6794-6>. Citado 3 vezes nas páginas 30, 31 e 32.

WARD, D.-M. et al. Enhancement of deep epileptiform activity in the eeg via 3-d adaptive spatial filtering. **IEEE transactions on biomedical engineering**, IEEE, v. 46, n. 6, p. 707–716, 1999. Disponível em: <https://doi.org/10.1109/10.764947>. Citado 2 vezes nas páginas 5 e 17.

WAZIRALI, R. An improved intrusion detection system based on knn hyperparameter tuning and cross-validation. **Arabian Journal for Science and Engineering**, Springer, v. 45, n. 12, p. 10859–10873, 2020. Disponível em: <https://doi.org/10.1007/s13369-020-04907-7>. Citado na página 26.

XING, W.; BEI, Y. Medical health big data classification based on knn classification algorithm. **IEEE Access**, v. 8, p. 28808–28819, 2020. Disponível em: <https://doi.org/10.1109/ACCESS.2019.2955754>. Citado na página 26.

YACUBIAN, E. M. T.; CONTRERAS-CAICEDO, G.; RíOS-POHL, L. **Tratamento medicamentoso das epilepsias**. São Paulo: Leitura Médica Ltda., 2014. Citado na página 16.

YI, S. et al. Grouped convolutional neural networks for multivariate time series. **arXiv preprint arXiv:1703.09938**, 2017. Disponível em: <https://doi.org/10.48550/arXiv.1703.09938>. Citado na página 33.

ZHANG, A. et al. **Dive into deep learning**. Cambridge University Press, 2023. Disponível em: <https://doi.org/10.48550/arXiv.2106.1134>. Citado na página 31.