UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Karl Vandesman de Matos Sousa

DESENVOLVIMENTO DE BASE DE DADOS E ALGORITMO DE CLASSIFICAÇÃO ANTECIPADA DE EXAMES RT-qPCR PARA COVID-19

Recife

2022

Karl Vandesman de Matos Sousa

DESENVOLVIMENTO DE BASE DE DADOS E ALGORITMO DE CLASSIFICAÇÃO ANTECIPADA DE EXAMES RT-qPCR PARA COVID-19

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Área de Concentração: Inteligência Computacional

Orientador: Adenilton José da Silva

Recife

2022

Catalogação na fonte
Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

S725d Sousa, Karl Vandesman de Matos
Desenvolvimento de base de dados e algoritmo de classificação antecipada de exames RT-qPCR para Covid-19 / Karl Vandesman de Matos Sousa. – 2022.
89 f.: il., fig., tab.

Orientador: Adenilton José da Silva.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2022.

Inclui referências e apêndice.

1. Inteligência computacional. 2. Aprendizagem de máquina. I. Silva, Adenilton José da (orientador). II. Título.

006.31 CDD (23. ed.) UFPE - CCEN 2022-160

Karl Vandesman de Matos Sousa

“Desenvolvimento de base de dados e algoritmo de classificação antecipada de exames RT-qPCR para COVID-19”

Dissertação de Mestrado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Ciência da Computação. Área de Concentração: Inteligência Computacional

Aprovado em: 02/06/2022.

BANCA EXAMINADORA

_____

Prof. Dr. Adenilton José da Silva
Centro de Informática/UFPE
(**Orientador**)

_____

Prof. Dr. George Gomes Cabral
Departamento de Computação / UFRPE

_____

Prof. Dr. André Câmara Alves do Nascimento
Departamento de Computação / UFRPE

AGRADECIMENTOS

Meus agradecimentos vão primeiramente à minha família, com destaque aos meus pais, Carlos e Vanda, pelo apoio e suporte desde sempre, sou grato a eles pelas conquistas que tive até hoje.

Ao meu orientador, Adenilton, pelas diversas discussões e investigações para chegarmos à este tema de pesquisa, e acreditar nesse grande desafio, pois não existia nada literatura científica sendo feito com essa perspectiva que abordamos.

Aos amigos de mais longa data: Clara Macedo, Pedro Ishimaru, Said Cândido, João Victor, Diogo Moury, Elys Carvalho, Ana Cristina, Catarina Cataldi, Brenda Alencar e Brena Rocha, que não necessariamente me ajudaram diretamente no trabalho desta pesquisa, mas fizeram e ainda fazem minha vida ser melhor.

Às profissionais de saúde Karoline Lima e Kamilla Cardeal que me acompanharam durante essa jornada acadêmica, cujos auxílios foram primordiais para um melhor equilíbrio da vida profissional e pessoal.

Aos colegas da pós-graduação Elisa Andrade, Túlio Pedrosa, Edmar Soares, Jéssica Feliciano, Jorge Delgado, Amanda Feitosa, Maria Clara Gê, Isis Vieira e Erika Marcella (que não é da pós, mas estava inserida no contexto), agradeço o partilha dos diversos momentos de conversas, risos e frustrações da vida acadêmica.

Agradeço pelo incentivo e fomento à pesquisa pela CAPES e à UFPE como um todo por toda infraestrutura de pesquisa e ensino que fizeram parte de toda minha formação acadêmica de graduação e mestrado.

Finalmente agradeço a mim mesmo por acreditar e persistir. Definitivamente este trabalho só foi concebido pela contribuição e influência direta de todos os citados, e ainda de vários outros que participaram da minha jornada de forma indireta. A todos a minha gratidão.

RESUMO

A doença denominada COVID-19, causada pelo vírus SARS-CoV-2 foi a responsável por gerar uma pandemia, preocupando diversos órgãos e instituições ao redor do mundo de forma a combatê-la e reduzir o seu impacto. Um dos fatores importantes para esse combate é o diagnóstico da doença, que deve considerar a assertividade e o tempo levado para sua conclusão. Este trabalho atuou nos dados quantitativos gerados pelo exame RT-qPCR, obtidos pela disponibilização feita pelo Núcleo de Pesquisa em Inovação Terapêutica (NUPIT) da Universidade Federal de Pernambuco. Os arquivos de saída do exame foram ajustados de forma a se ter uma base de dados disponível para treinamento de algoritmos de aprendizagem supervisionada, totalizando 75.716 linhas com 45 características temporais e um alvo classificado como positivo ou negativo. Esses dados formam uma série temporal, sendo cada característica um valor de fluorescência do processo da reação química do RT-qPCR, para detectar genes do SARS-CoV-2. Foram elencados diferentes algoritmos da literatura, desde mais genéricos a mais complexos, com comitês, e considerando ordenação temporal. Primeiramente, esses algoritmos foram treinados com a totalidade dos ciclos, para se ter uma referência do desempenho que poderia ser obtido. Posteriormente, foram realizados treinamentos com uma redução desse número de ciclos, com o intuito de antecipar o exame e consequentemente diminuir o tempo necessário para sua conclusão. Para comparar os desempenhos, foram analisadas as métricas AUROC (do inglês, *Area Under the Receiver Operating Characteristic*), acurácia, especificidade e sensibilidade, com média e desvio padrão calculadas em cima de cem reamostragens da base de dados. Para o cenário de redução de 30 ciclos, foi realizada a otimização de hiperparâmetros dos três algoritmos que se destacaram na etapa anterior: um baseado em redes neurais, MLP (*Multilayer Perceptron*) e dois comitês de classificadores, o XGBoost e o *Time Series Forest* (TSF), sendo que este último considera a relação temporal das características. Testes estatísticos realizados indicaram um maior desempenho do TSF (AUROC $98,98 \pm 0,07\%$) e do MLP ($98,94 \pm 0,19\%$) para 30 ciclos, uma melhoria de desempenho graças à otimização, mas ainda com desempenho inferior ao algoritmo treinado com 35 ciclos com valores padrões de hiperparâmetros. Com isso, este trabalho fornece respaldo para a redução do tempo do RT-qPCR aplicado para COVID-19, por meio de algoritmos de aprendizagem de máquina.

Palavras-chaves: aprendizagem de máquina; classificação antecipada de séries temporais; otimização de hiperparâmetros; RT-qPCR; COVID-19.

ABSTRACT

The disease named COVID-19, caused by the SARS-CoV-2 virus, has led to a pandemic, challenging several bodies and institutions around the world in order to combat it and reduce its impact. One important element for this combat is the disease diagnosis, that should consider both the assertiveness and the time spent to its conclusion. This work deals with quantitative data generated by RT-qPCR exam, provided by the Núcleo de Pesquisa em Inovação Terapêutica (NUPIT) of Universidade Federal de Pernambuco. The output files from this exam were filtered and adjusted in a way that they can be used in supervised learning algorithms, forming a dataset with a total of 75.716 rows and 45 temporal features (machine cycles), and a target to be classified as positive or negative. These data form a temporal series, each feature being a fluorescence value of a chemical reaction process from the RT-qPCR, that detects the genes of SARS-Cov-2. Different algorithms from the literature were chosen, from generic ones to others more complex, that use ensembles and consider temporal relation between features. At first, these algorithms were trained with all machine cycles, in order to have a benchmark. Then, the training was made reducing the number of cycles, to anticipate the exam and consequently decrease the time needed to conclude it. To compare the performances, the metrics used were AUROC (Area Under the Receiver Operating Characteristic), accuracy, specificity and sensitivity, with mean and standard deviation calculated on one hundred resamplings of the dataset. In the 30-cycle scenario, the hyperparameter optimization was made for the top three algorithms in the previous stage: one based on neural networks, MLP (Multilayer Percepton), and two ensemble classifiers, XGBoost and Time Series Forest (TSF), with the last one considering the relationship between features. Statistical tests performed indicated a higher performance of TSF (AUROC $98,98 \pm 0,07\%$) and MLP ($98,94 \pm 0,19\%$) using 30 cycles and an improvement made by the optimization, but still underperforming the algorithm trained with 35 cycles using default hyperparameter values. Therefore, this work provides support for the reduction of cycles of RT-qPCR applied to COVID-19, through machine learning algorithms.

Keywords: machine learning; early classification of time series; hyperparameter optimization; RT-qPCR; COVID-19.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Esta dissertação descreve o desenvolvimento de uma base de dados contendo exames de RT-qPCR para o COVID-19, indo posteriormente para a modelagem preditiva do problema, estruturado como uma classificação de série temporal, e realizando finalmente a análise e comparação dos resultados. Inicialmente, neste capítulo, será abordada a motivação para se investigar esse problema, depois descritos os objetivos gerais e específicos, e por final será exposta a organização dos capítulos seguintes.

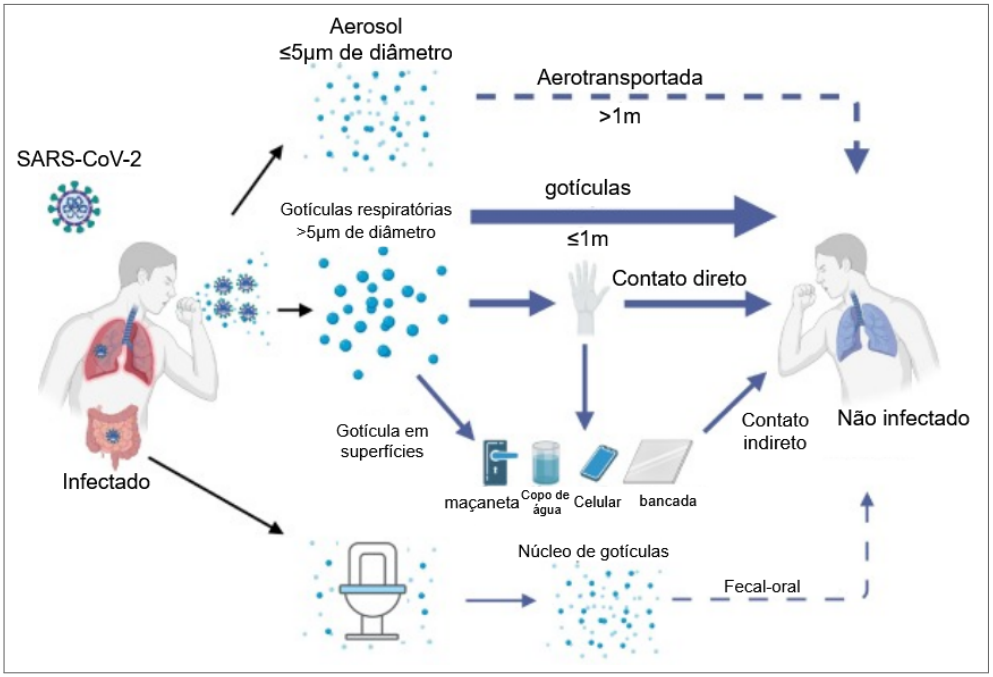
1.1 MOTIVAÇÃO

No final do ano de 2019 surge uma doença causada pelo vírus SARS-CoV-2, denominada COVID-19, em que logo em março de 2020 foi declarada a situação de pandemia global pela Organização Mundial de Saúde (OMS) (GHEBREYESUS, 2020). A COVID-19 pode ser transmitida por diferentes mecanismos, como apresentado no esquema de transmissões da Figura 1. Dentre esses mecanismos, pode-se citar principalmente as gotículas respiratórias e o contato direto com superfícies contaminadas (HARRISON; LIN; WANG, 2020), sendo essas formas simples e diretas de transmissão grandes facilitantes da disseminação da doença, aumentando significativamente as possibilidades de surtos, com a evolução para níveis de alerta epidemiológicos como epidemia, e posteriormente, em escala global, uma pandemia (BUTANTAN, 2021), como ocorreu.

Nesse contexto de transmissão acelerada de uma doença, torna-se fundamental dispor de exames que a detectem de forma rápida, e ainda com níveis satisfatórios de desempenho, para que se tenha uma maior rastreabilidade da doença, tornando essa ferramenta parte de uma estratégia coordenada de combate epidemiológico (MANABE; SHARFSTEIN; ARMSTRONG, 2020). Apesar desse grande papel da testagem, é importante destacar que ela por si só não é suficiente para prevenção de casos de surtos, devendo se integrar como parte de uma planejamento de saúde pública mais amplo, envolvendo também o distanciamento social, uso de máscaras, diminuição de aglomerações (MANABE; SHARFSTEIN; ARMSTRONG, 2020), e vacinação (ZIENELDIEN et al., 2021), quando já disponível.

O exame de diagnóstico padrão de referência para o COVID-19 é o RT-PCR (do inglês, *Reverse Transcriptase - Polymerase Chain Reaction*), realizado em laboratório, mas com amos-

Figura 1 – Mecanismos de transmissão do COVID-19.

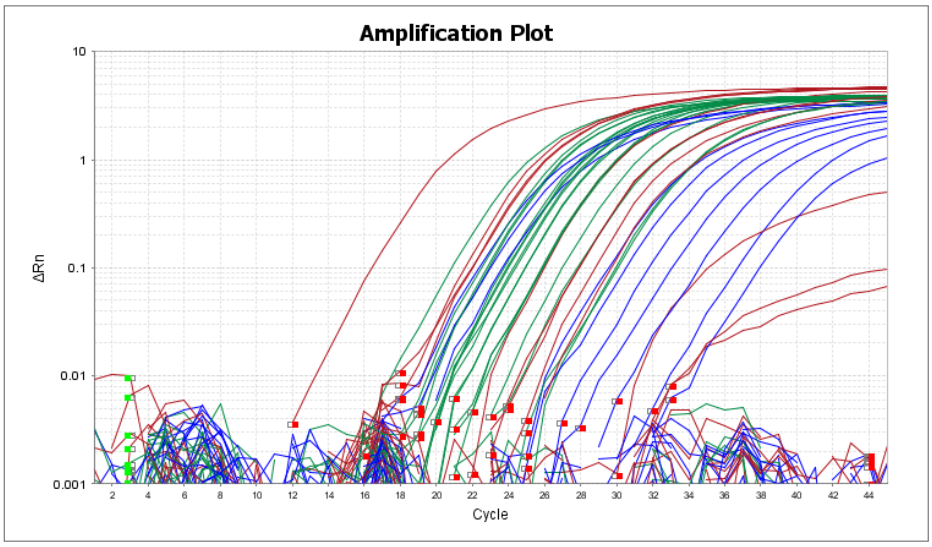


Fonte: Adaptado de HARRISON; LIN; WANG (2020)

tras coletadas por *swab* nasofaríngeo (CHENG et al., 2020). Esse exame possui uma versão que pode ser acompanhada quantitativamente em tempo real, o RT-qPCR, como mostra a Figura 2, com gráficos para diferentes amostras. Esse gráfico acompanha o processo de fluorescência da reação química que ocorre no meio preparado com as amostras. Essa fluorescência está relacionada com a presença dos genes a serem detectados do vírus, e é registrada em intervalos regulares de tempo (chamado de ciclos de máquina). Logo, esses dados formam uma série temporal que pode ser classificada como positiva (contendo a presença do vírus) ou negativa. Dessa forma, analisando por uma perspectiva de aprendizagem supervisionada, um conjunto de exames de RT-qPCR, com a devida estruturação dos dados para uma visão de modelagem, podem ser utilizados para treinar algoritmos de Aprendizagem de Máquina (AM), que serão utilizados como preditores de exames. Dado que se tem estruturado em uma tabela os dados de amplificação por ciclo, com a respectiva resposta, pode-se avaliar cenários de utilização de um número menor de ciclos do exame para se atingir um mesmo desempenho, por meio de algoritmos mais complexos, diminuindo eventualmente o tempo necessário para conclusão do exame (XING; PEI; YU, 2011).

Na Figura 3 é apresentado o processo geral de desenvolvimento do trabalho. Como primeira etapa, foi feita a estruturação dos arquivos gerados pela máquina de RT-qPCR, e montagem da visão de modelagem com as variáveis que poderiam ser utilizadas em um modelo preditivo,

Figura 2 – Curva de monitoramento de fluorescência (ΔRn) gerada para vários exames de RT-qPCR.

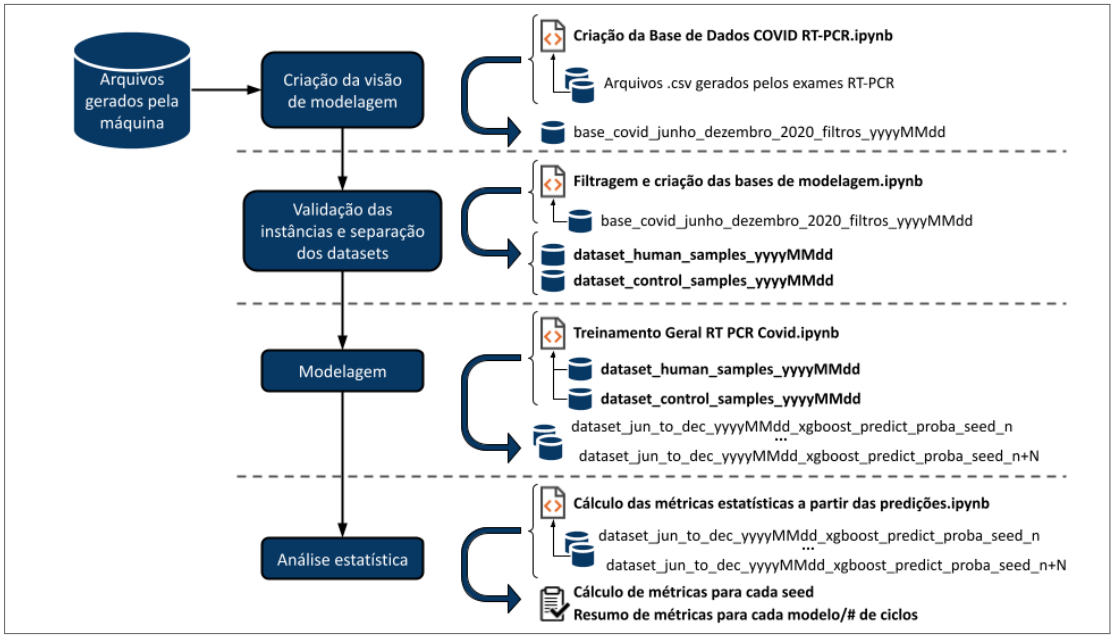


Fonte: Elaborada pelo autor (2022)

em que para cada instância (exame realizado para uma pessoa), se tem as variáveis elencadas. Na segunda etapa, passou-se pela validação das instâncias e exclusão de casos inválidos, agrupando os casos de amostras humanas, e de controle de qualidade. Com a base de dados pronta para modelagem, foram então elencados os algoritmos mais adequados para o problema de classificação de uma série temporal. Finalmente, tem-se como último passo análises estatísticas, utilizando métricas de classificação, e na comparação direta dos algoritmos, por meio de testes estatísticos que indicam se, na base de dados considerada, o desempenho nas métricas de um algoritmo é melhor que a de outro.

Várias ferramentas tecnológicas e computacionais vêm sendo aplicadas nas diferentes frentes de combate ao COVID-19 (ALIMADADI et al., 2020). No entanto, pouco vem se sendo desenvolvido especificamente para o exame que é utilizado como referência de diagnóstico (TAHAMTAN; ARDEBILI, 2020), o RT-PCR. Em um desses trabalhos (GUNAY; GOCERI; BALASUBRAMANIYAN, 2016), desenvolvido antes mesmo do surto da COVID-19, dado que o método PCR já é aplicado há vários anos na detecção de diferentes patógenos, foca no cálculo de um valor ótimo de ciclo limite, C_t (do inglês *Threshold Cycle*), número de ciclo no qual se atinge o limite de amplificação, e que está relacionado com a quantidade do gene alvo na amostra coletada. Outra importante pesquisa, (CORDARO et al., 2021) investiga o impacto das configurações do PCR, com o objetivo de aumentar a taxa de sucesso na realização do exame, identificando 15 características com potenciais de otimização. Apesar de apontar desafios na execução do PCR, a pesquisa está restrita à uma forma mais tradicional do método, não sendo

Figura 3 – Ilustração do processo geral de desenvolvimento do problema de pesquisa.



Fonte: Elaborada pelo autor (2022)

utilizado para o RT-qPCR, mas indicando o número de ciclos como a segunda característica com melhor poder de predição.

Em (ALOUANI et al., 2021) é realizado um trabalho mais próximo ao desta pesquisa, aplicando AM para COVID-19 considerando especificamente os dados de fluorescência do exame de RT-qPCR. Mais recentemente, durante a escrita deste trabalho, foi visto em (LEE et al., 2022) a abordagem de redução do número máximo de ciclos de máquina, com o mesmo contexto de dados de RT-qPCR. Apesar dos resultados indicando a possibilidade de antecipação do exame, uma volumetria de amostras baixa foi utilizada, com 5.810 amostras, sendo 181 positivas. Um primeiro ponto que merece destaque, na diferenciação ao trabalho de (LEE et al., 2022), é a análise estatística mais aprofundada, tanto pela maior volumetria de amostras, 75.716 casos, como também pela metodologia praticada para avaliar os desempenhos com 100 reamostragens da base de teste e aplicação de testes estatísticos. Em segundo lugar, está o detalhamento das amostras utilizadas, e diferenciação entre amostras de genes e controle negativo. Essa separação é importante pois o comportamento de fluorescência das amostras de controle são mais estáveis. Por fim, a complexidade do problema é vista a partir da comparação de diferentes algoritmos de AM, indo desde mais básicos e genéricos de classificação, para outros de comitês de classificadores e que consideram a ordenação de características do problema.

1.2 OBJETIVOS DO TRABALHO

1.2.1 Objetivo geral

Neste trabalho tem-se como objetivo geral o desenvolvimento de um modelo de AM para predição da presença do vírus SARS-CoV-2 em humanos, por meio dos dados de fluorescência gerados pelo exame RT-qPCR, considerando um cenário de redução do número de ciclos do exame.

1.2.2 Objetivos específicos

- Construção de uma base de dados com visão de modelagem preditiva a partir de arquivos gerados pela máquina que realiza o exame RT-qPCR para COVID-19;
- Realizar o treinamento de algoritmos de AM de classificação em séries temporais com a base de dados desenvolvida, e comparar o desempenho em cenários de diminuição do número de ciclos do exame de RT-qPCR;
- Realizar a otimização de hiperparâmetros dos algoritmos treinados com um número menor de ciclos, de forma a se aproximar ou manter o desempenho, em comparação com algoritmos treinados com a totalidade de ciclos do exame;

1.3 ORGANIZAÇÃO DO TRABALHO

Neste capítulo foi apresentado o contexto geral da pesquisa, e a motivação que gerou a realização desse trabalho, no contexto de pandemia do COVID-19, e importância da testagem. Contando com a construção de uma base de dados em que torna possível a modelagem preditiva de exames de RT-qPCR, abre-se possibilidades para aplicação de algoritmos de AM.

No Capítulo 2 é dada uma visão geral do exame de RT-qPCR e o contexto de sua utilização, dado que é a base para a construção das bases de dados para treinamento dos algoritmos. Além disso, são apresentados os conceitos básicos dos algoritmos usados posteriormente para comparação de desempenho na classificação dos exames de PCR.

Posteriormente, no Capítulo 3 são apresentados os trabalhos relacionados a esta pesquisa. Para isso, são consideradas as diferentes formas de diagnóstico utilizadas para o COVID-

19, a utilização das ferramentas que a área de Inteligência Artificial dispõe no combate à doença, e mais especificamente aos algoritmos que podem ser utilizados para problemas de classificação. Nesse último caso, é comparado de forma geral a diferença em considerar a relação de ordenação de características, com um algoritmo genérico de classificação.

A construção da bases de dado utilizada para treinamento dos algoritmos é apresentada no Capítulo 4. Foram reunidos milhares de arquivos advindos do exame de PCR, que ao final, após as devidas filtragens de casos segundo os protocolos, geraram pouco mais de 76.000 instâncias, com 80% sendo amostras humanas, e o restante de amostras de controle. Esse desenvolvimento de base de dados é uma importante etapa da pesquisa, pois deixa-o mais reprodutível e com possibilidades de fácil acesso para trabalhos futuros.

No capítulo 5 são apresentados os experimentos realizados no treinamento dos algoritmos para a predição do exame de PCR. Os experimentos iniciais envolvem avaliar o potencial de desempenho com a totalidade de 45 ciclos do exame, para diferentes classes de algoritmos, desde classificadores tidos como genéricos, passando por baseados em árvore e *boosting*, e que levam em consideração a ordenação das características (série temporal). Com essa referência de valores iniciais, é feita a redução do número de ciclos e consequente treinamento e avaliação, para análise do impacto da antecipação da classificação no exame. Para avaliação dos algoritmos, são consideradas métricas usuais no contexto médico de diagnóstico, como sensibilidade e especificidade, e outra mais geral para medir desempenho de algoritmos de classificação, AUROC (BRADLEY, 1997). Com esses valores de métricas obtidos para várias reamostragens, torna possível também a comparação por meio de testes estatísticos, comprovando com maior rigor o desempenho de um algoritmo, no contexto considerado.

Por fim é feita uma discussão dos resultados obtidos. Também são mencionadas as contribuições realizadas por este trabalho, e o impacto que possa a ser gerado a partir da utilização dos algoritmos desenvolvidos.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo reúne os conceitos básicos que fundamentam o trabalho. Na Seção 2.1 são introduzidos os conceitos gerais da área de Aprendizagem de Máquina (AM), seguido dos algoritmos aqui elencados para classificação. Na definição e descrição dos algoritmos, são exploradas suas características, possíveis vantagens e desvantagens nas suas utilizações, e também pontuando sobre seus hiperparâmetros que podem ser modificados. Aproveitando essas possíveis modificações, na Seção 2.2 é apresentado o conceito de Otimização de Hiperparâmetros (OHP) (YANG; SHAMI, 2020), estratégias para sua utilização, e por fim ferramentas que podem ser utilizadas para suporte à construção de algoritmos ajustados a um problema específico.

2.1 ALGORITMOS DE APRENDIZAGEM DE MÁQUINA

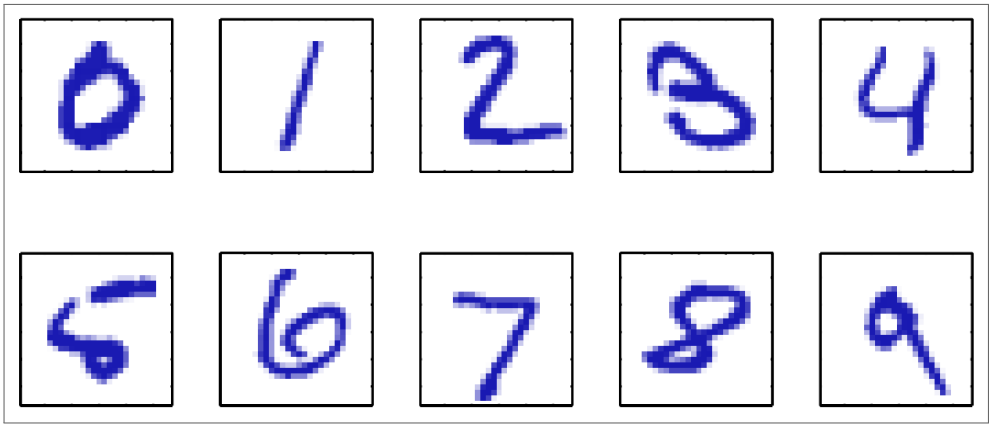
A grande área de Inteligência Artificial (IA) (RUSSELL, 2016) tem origens e forte influência nos trabalhos de Alan Turing, na década de 1950 (TURING, 1950), introduzindo o Teste de Turing além de conceitos importantes como Aprendizagem de Máquina (AM), algoritmos genéticos e aprendizagem por reforço. Sendo assim, a AM trata-se de um dos campos da Inteligência Artificial, sendo interdisciplinar, pois é desenvolvida a partir de resultados da computação, de probabilidade, estatística, neurobiologia, entre outros. À medida que foi sendo aplicada com sucesso em diferentes áreas, como Visão Computacional (KRIZHEVSKY; SUTSKEVER; HINTON, 2017), Processamento de Linguagem Natural (VASWANI et al., 2017) e em diferentes campos da medicina (RAJKOMAR; DEAN; KOHANE, 2019), foi ganhando cada vez maior relevância e atenção da comunidade acadêmica e do mercado.

Para a área de AM, pode-se identificar três principais ramificações, a depender do objetivo e dos dados disponíveis. Primeiramente, a AM supervisionada, que como o nome sugere, necessita que os dados à disposição possuam um rótulo correspondente, que será utilizado como parte fundamental no processo de treinamento do algoritmo. Caso a saída desejada seja do tipo contínua, o problema trata-se de uma regressão, e caso contrário, em que se atribui para a instância um número finito de diferentes categorias, tem-se um problema de classificação (BISHOP, 2011). Em contrapartida, na aprendizagem não-supervisionada, retira-se a necessidade da rotulagem das instâncias, e o objetivo está em aprender sobre propriedades e comportamentos de interesse no conjunto de treinamento. Um exemplo de aplicação seria

na tarefa de agrupamento, ou *clustering*, em que se divide os dados por grupos similares entre si. Por último, um caso especial que se distingue dos dois anteriores, é o de aprendizagem por reforço (SUTTON; BARTO, 2018). Além de se ter um conjunto de dados, é especificado para o problema uma interação com um ambiente, de forma que exista uma retroalimentação entre o algoritmo que está aprendendo e suas experiências (GOODFELLOW; BENGIO; COURVILLE, 2016).

Um caso que pode ilustrar o avanço na abordagem com AM supervisionada, é o de reconhecimento de dígitos manuscritos, com exemplos na Figura 4. A construção de um sistema que tenha como entrada os píxeis de escrita do dígito, e como saída o resultado do reconhecimento, um dentre os dez dígitos possíveis (0, . . ., 9), seria um problema complexo, devido à enorme variabilidade da escrita dos dígitos. Uma abordagem mais direta, seria pela utilização de heurísticas e regras que levem em consideração os formatos de cada dígito. No entanto, na prática, isso traria um incontável número de regras e exceções, podendo mesmo assim não obter o desempenho desejado (BISHOP, 2011). A abordagem com AM está no desenvolvimento de tal sistema, nesse caso, um classificador, que reúne um conjunto de dados para treiná-lo, consistindo de vários exemplos de imagens de dígitos, e o correspondente rótulo do dígito de resposta, que se sabe antecipadamente. Dessa maneira, durante a etapa de treinamento do algoritmo de AM, utilizando esse conjunto de treinamento construído, é definida a forma de uma função, que será utilizada para a predição de novas imagens.

Figura 4 – Exemplos de dígitos manuscritos, que podem ser utilizados para treinamento de algoritmos de AM.



Fonte: Adaptado de BISHOP (2011)

A partir do conhecimento do caso apresentado anteriormente, e partindo para um conceito geral, segundo (DUDA; HART; STORK, 2000) é aplicada uma aprendizagem em quaisquer métodos que utilizam dados de um conjunto de treinamento e são aplicados na construção de um classificador. O processo de aprendizagem implica na redução de uma métrica de erro no

decorrer da iteração dos dados de treinamento e checagem de seus rótulos.

Depois de apresentada a área de forma geral, agora são formalizados alguns conceitos básicos da AM supervisionada, considerando o contexto específico de aplicação desse trabalho. Como discutido anteriormente, para um problema de classificação, é necessário dispor de um conjunto de dados, com quantidade representada por N instâncias, sendo $\mathbf{x}_i$ a i-ésima instância, cada uma caracterizada por p atributos, $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$. Cada uma dessas instâncias pertence a uma categoria ou rótulo $y \in \{y_0, y_1\}$. A saída do algoritmo é expressa por uma pontuação, que pode ser vista como uma probabilidade de pertencer a um rótulo, sendo portanto um valor contínuo, $0 \leq y_{pred} \leq 1$.

O tipo de dado (imagem, texto, dados tabulares, ...) e contexto considerado na classificação binária também é importante, dado que há algoritmos que focam em características específicas. Por isso, vale conceituar também uma série temporal e como seriam os dados específicos desse contexto. Uma série temporal consiste de medições de valores reais ordenados, amostrados em um intervalo regular de tempo (MITSA, 2010). Uma série temporal de $\mathbf{x}_i = (x_{i1}, \ldots, x_{ip})$ para $t = \{t_1, t_2, \ldots, t_p\}$, representa os atributos, para cada marcação de intervalo de tempo. Nesse caso, tem-se uma série temporal univariada, pois para cada t, existe somente um valor sendo registrado para a respectiva instância, x_{i1}.

Diversos algoritmos podem ser utilizados para a aprendizagem no contexto classificação de séries temporais. Primeiramente, os algoritmos mais simples: K-vizinhos mais próximos (K-NN, do inglês K-*Nearest Neighbors*) (COVER; HART, 1967), Bayes ingênuo (*naive* Bayes em inglês, NB) (LEWIS, 1998), o Máquina de Vetores de Suporte (SVM, do inglês *Support Vector Machines*) (BOSER; GUYON; VAPNIK, 1992) e MLP (BRAGA, 2007). Logo depois, estão classificados os algoritmos baseados em árvore de decisão, que a partir desse conceito mais básico de árvore podem ser realizadas diversas melhorias, como o conceito de *boosting*, utilizado no XGBoost (CHEN; GUESTRIN, 2016). Por fim, será apresentado também o algoritmo *Time Series Forest* (TSF) que, em sua formulação, já considera a relação de ordem entre as características da base de dados, que nesse caso trata-se de uma relação temporal.

2.1.1 Bayes Ingênuo

O algoritmo classificador NB utiliza como princípio fundamental o teorema de Bayes, definido como

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}. \tag{2.1}$$

Esse teorema pode então ser utilizado para derivar a probabilidade de uma hipótese (classificação) considerando dados observados (RANGANATHAN; NAKAI; SCHONBACH, 2018):

$$P(hipotese|dados) = \frac{P(dados|hipotese)P(hipotese)}{P(dados)}, \tag{2.2}$$

em que $P(dados|hipotese)$ é a probabilidade de observação dos dados, dada a condição verdadeira da hipótese, $P(hipotese)$ é a probabilidade *a priori* da hipótese e $P(dados)$ é a probabilidade de observação dos dados, independente da hipótese. Aqui, o termo hipótese pode ser entendido como a predição de classe do modelo, e portanto, tem-se para o caso binário duas hipóteses, com $P(H_1) = 1 - P(H_2)$.

Dado esse contexto, pode-se agora evidenciar que o processo de aprendizagem do classificador NB se dá pela construção de um modelo probabilístico Bayesiano, que atribui uma probabilidade a posteriori de uma classe para uma instância. Revisitando a Equação 2.2, e adaptando para as notações que representam classe, y, e o vetor de atributos de uma instância $\mathbf{x}_i$:

$$P(y|\mathbf{x}_i) = \frac{P(\mathbf{x}_i|y)P(y)}{P(\mathbf{x}_i)}. \tag{2.3}$$

O termo *naive* (em português, ingênuo), se refere ao fato de se assumir que as características do problema são condicionalmente independentes em relação ao alvo, ou seja, que não existe correlação entre essas características. Embora na prática isso seja muito difícil de ocorrer, há cenários em que mesmo modelos mais complexos de classificadores de Bayes não apresentam melhoria significativa em relação à essa versão mais simples. (KUPERVASSER, 2014).

As diferentes formas de implementação desse algoritmo se diferenciam principalmente pela distribuição que é assumida para $P(x_i|y)$. Uma dessas é a gaussiana,

$$P(x_i|y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right), \tag{2.4}$$

cujos parâmetros μ_y e σ_y representam, respectivamente, média e desvio padrão da distribuição da classe, e são estimados durante o treinamento usando o método da máxima verossimilhança (ROSSI, 2018).

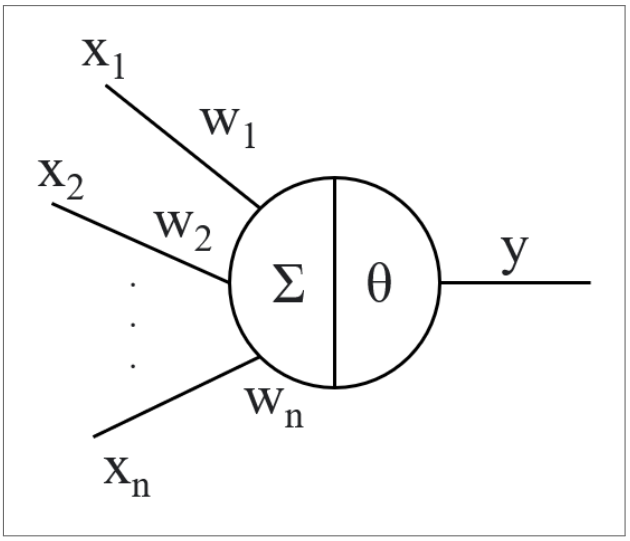
2.1.2 Redes Neurais Artificiais

Redes Neurais Artificiais (RNAs) são sistemas matemáticos utilizados em AM com inspiração biológica, capazes de mapear relações de entrada e saída de dados, atuando como mapeadores universais de funções multivariáveis (BRAGA, 2007). Sua estrutura consiste na interligação de unidades similares aos neurônios por meio de pesos (análogo às sinapses), que indicam a relevância da conexão, e são arranjados em camadas. Esse neurônio pode ser descrito como um modelo matemático, como no caso do *perceptron* (ROSENBLATT, 1958), sendo uma unidade de processamento que realiza a soma dos sinais de entrada, ponderados pelos pesos das conexões, como pode ser observado na Figura 5. Logo após o somatório, é aplicada uma função de ativação, de forma a introduzir não-linearidade, chegando-se à seguinte equação:

$$y_i = \theta\left(w_0 + \sum_{i=1}^{n} x_i w_i\right), \tag{2.5}$$

sendo n o número de entradas, $\theta()$ a função de ativação e w_i o peso associado à entrada x_i.

Figura 5 – Representação Matemática do modelo de Neurônio de McCulloch e Pitts.

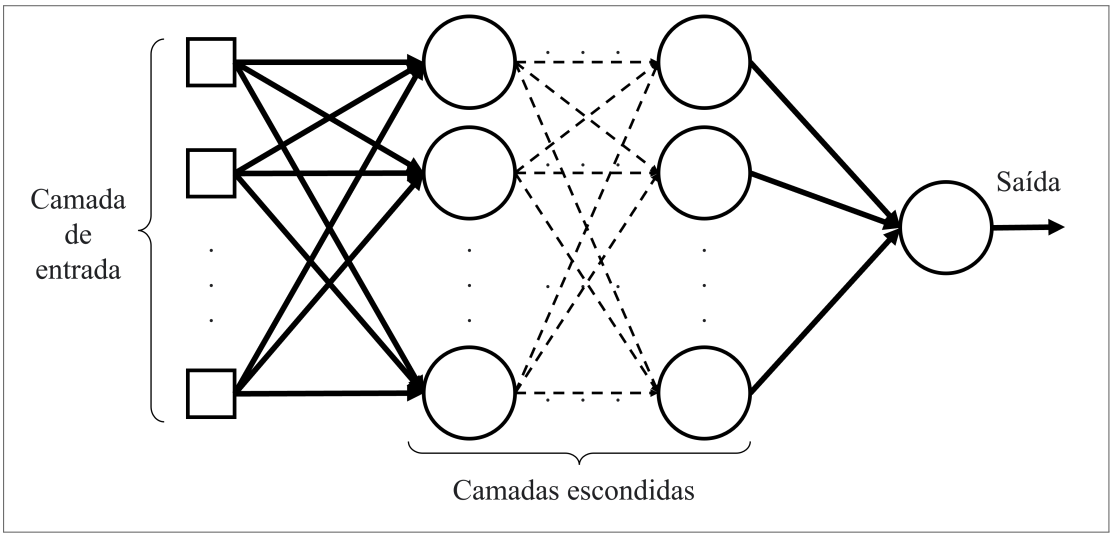


Fonte: Adaptado de BRAGA (2007)

Uma configuração específica de RNAs define as conexões dos neurônios para propagação dos sinais no sentido de ligar as camadas de entrada para as camadas seguintes, sendo chamada então *Feedfoward Neural Networks* (FNNs), em que os sinais seguem a direção unidirecional de entrada pra saída, sem retroalimentação. Essa arquitetura pode ser totalmente caracterizada com uma matriz (que por suas dimensões, indica o número de neurônios em cada camada), e o tipo de função de ativação aplicada em cada neurônio. A formação dessa

estrutura por meio da combinação de *perceptrons*, é denominada de *perceptrons* de múltiplas camadas (*Multilayer Perceptron*, MLP), ilustrada na Figura 6, para o caso de classificação binária. Em uma MLP, a camada de entrada está diretamente relacionada com a quantidade de características consideradas para o problema. Já a última camada, chamada de camada de saída, tem uma correspondência com o número de classes. No caso de classificação binária, é necessário somente um neurônio, pois a probabilidade de classificação de uma classe, será o complementar da outra. Outro conceito importante para a MLP, é o de camada intermediária, também denominada de camada escondida, cuja adição à sua estrutura permite que a rede neural possa implementar qualquer função contínua (CYBENKO, 1989).

Figura 6 – Grafo de uma rede neural do tipo MLP, com generalização da quantidade de neurônios e camadas, para classificação binária.



Fonte: Adaptado de HAYKIN (2001)

Um dos marcos no avanço de desempenho das redes neurais foi o algoritmo de retropropagação de erro (RUMELHART; HINTON; WILLIAMS, 1986). Nele, é fornecido um método computacional eficiente para o treinamento da MLP, aplicando um procedimento de aprendizagem para treinamento e atualização dos parâmetros da rede neural (HAYKIN, 2001). Esse processo é definido basicamente por duas etapas:

- Propagação: Por meio de uma instância de treinamento aplicado à camada de entrada $(\mathbf{x}_i, y_i)$, o sinal é propagado pelo cálculo de resposta do conjunto de *perceptrons* das camadas seguintes, mantendo-se os pesos sinápticos inalterados.

- Retropropagação: Essa etapa se inicia na camada de saída, comparando-se o sinal de saída com o rótulo real da instância passada, gerando então um erro. Esse sinal de erro

é passado para a esquerda através da rede, recursivamente calculando-se os gradientes locais de cada *perceptron*. Nesse processo, os pesos sinápticos são otimizados para adaptar o sinal de saída à resposta real desejada, num processo estatístico considerando o conjunto de dados de treinamento.

2.1.3 Máquina de Vetores de Suporte

O algoritmo denominado Máquina de Vetores de Suporte foi proposto inicialmente por (BOSER; GUYON; VAPNIK, 1992). A ideia por trás desse método se baseia em simplificar problemas ao caso onde os dados possuam classes linearmente separáveis. Isso é feito por meio de um mapeamento não-linear do espaço de entrada para um espaço de características em que seja possível separar linearmente o conjunto de dados por suas classes (VAPNIK, 2010). Utilizando teorias de otimização, atua-se na minimização de erro por meio da maximização da margem do hiperplano de separação entre os dados de diferentes classes.

Primeiramente, para explicar seu funcionamento, será considerado o problema mais básico em que os dados já apresentam uma separação linear. De forma similar à equação de saída de um *perceptron* apresentada anteriormente na Equação 2.5, sem a aplicação de uma função de ativação, tem-se a saída de uma função linear, seguida da forma vetorial,

$$y = \sum_{i=1}^{n} w_i x_i + b, \tag{2.6}$$

$$y = \mathbf{w}^T \mathbf{x} + b, \tag{2.7}$$

em que **x** é o vetor de entrada, **w** o vetor de pesos a serem ajustados e b o *bias* (viés). Em termos de classificação linear, pode-se definir que caso $y \geq 0$, atribui-se para a instância uma classe positiva, caso contrário, uma classe negativa. A equação que representa a superfície de decisão que realiza a separação entre o conjunto de dados é a seguinte

$$\mathbf{w}^T \mathbf{x} + b = 0, \tag{2.8}$$

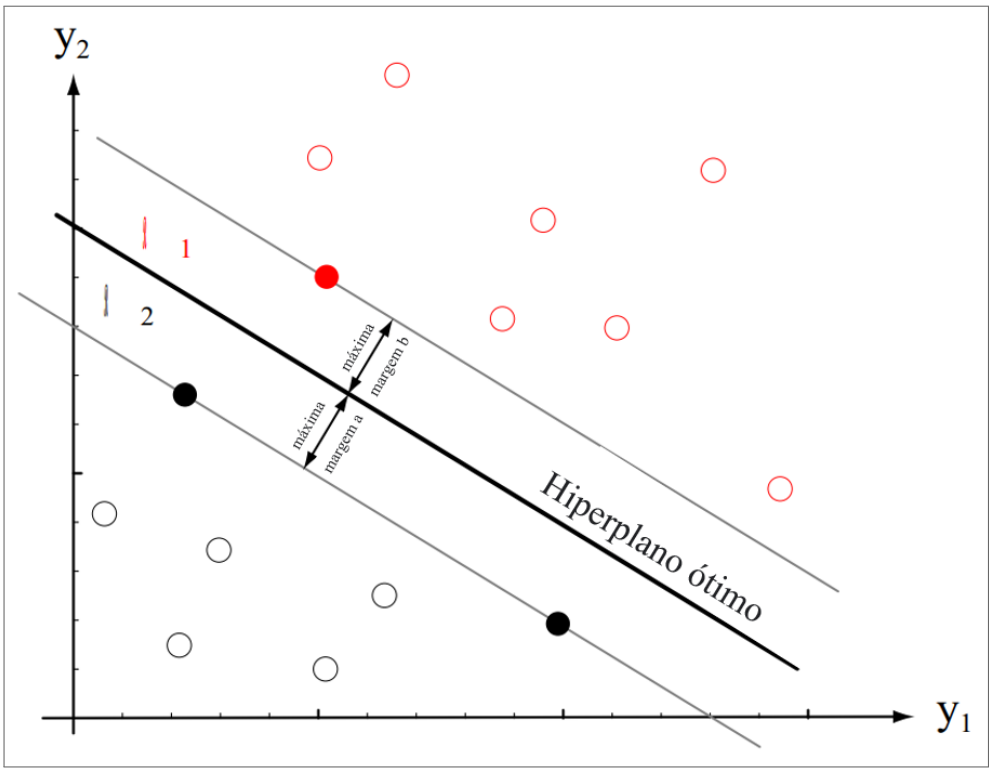
cuja representação simplificada em duas dimensões pode ser vista na Figura 7. O vetor de pesos **w** define a direção perpendicular ao hiperplano (nesse caso, uma reta), enquanto que o valor de b move o hiperplano no sentido paralelo a ele mesmo. Considerando então $\mathbf{w}_o$ e b_o os valores ótimos, o objetivo da otimização é encontrar tais valores que façam a margem

de separação entre instâncias de classes opostas mais próximos, a maior possível. Isso é feito a partir do cálculo da distância do hiperplano aos chamados vetores de suporte, que são os vetores formados pelos dados que se encontram mais próximos da superfície de decisão, também representados na Figura 7. A soma das distâncias algébricas dos vetores ao hiperplano, é dada por

$$d = \frac{2}{\|\mathbf{w}_o\|}, \tag{2.9}$$

indicando que maximizar a margem de separação implica em minimizar a norma euclidiana do vetor de pesos **w** (HAYKIN, 2001).

Figura 7 – Construção de um hiperplano ótimo no treinamento de um algoritmo de SVM.



Fonte: Adaptado de DUDA; HART; STORK (2000)

Considerando problemas reais de classificação, é bastante usual encontrar dados não-separáveis em seu espaço de entrada original. Para a resolução de problemas com essa configuração, a ideia é transformar os dados do seu espaço original de características ($\mathbf{x}_i$) para outro espaço, de maior dimensão ($\mathbf{z}_i = \phi(\mathbf{x}_i)$), onde pode então ser aplicado o caso base de hiperplano de separação ótima. Essa transformação pode ser entendida geometricamente por meio da Figura 8, em que na representação da esquerda, tem-se um espaço bidimensional onde não se consegue separar os dados e suas classes por meio de uma reta, mas na represen-

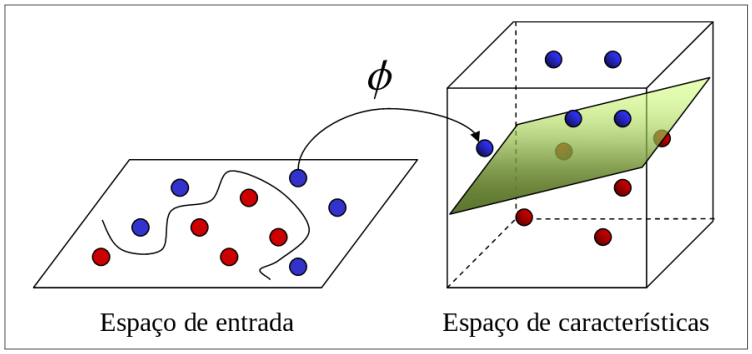
tação tridimensional à direita, torna-se possível. Apesar dessa representação para um melhor entendimento geométrico, a ideia das máquinas de *kernel* é substituir o produto interno a ser realizado pelas funções base, $\phi(\mathbf{x})$ e $\phi(\mathbf{x}')$ por uma função *kernel*, $K(\mathbf{x}, \mathbf{x}')$, entre os dados no espaço original de entrada. Ou seja, dada a equivalência,

$$K(\mathbf{x}, \mathbf{x}') = \langle \phi(\mathbf{x}), \phi(\mathbf{x}') \rangle, \tag{2.10}$$

é mais fácil calcular $K(\mathbf{x}, \mathbf{x}')$ do que calcular as transformações $\phi(\mathbf{x})$, $\phi(\mathbf{x}')$ e realizar o produto escalar. Exemplos típicos utilizados como funções *kernel* são os seguintes:

- Linear: $K(\mathbf{x}, \mathbf{x}') = \mathbf{x}^T\mathbf{x}'$
- Polinomial (dimensão d): $K(\mathbf{x}, \mathbf{x}') = (x^T x' + c)^d$
- Gaussiano ou RBF (do inglês *Radial Basis Function*): $K(\mathbf{x}, \mathbf{x}') = e^{-\frac{||\mathbf{x}-\mathbf{x}'||^2}{\sigma^2}}$

Figura 8 – Transformação do espaço de entrada para o espaço de características.



Fonte: Adaptado de WILIMITIS (2018)

Após apresentados os principais conceitos sobre o SVM, será formalizado o problema para o caso genérico não separável (HAYKIN, 2001). Tem-se que para um conjunto de treinamento $(\mathbf{x}_i, y_i)$, com $i = 1, 2, \ldots, N$, a restrição é dada por

$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 - \xi_i, \text{ para } i = 1, 2, \ldots, N \tag{2.11}$$

$$\xi \geq 0 \text{ para todo } i. \tag{2.12}$$

O vetor de peso $\mathbf{w}$ e as chamadas variáveis soltas ξ_i devem minimizar a seguinte funcional de custo:

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i, \tag{2.13}$$

em que C é um hiperparâmetro definido que está relacionado com a regularização do modelo, controlando sua complexidade.

2.1.4 K-Vizinhos mais próximos

Algoritmo proposto por (FIX; HODGES, 1951), e aperfeiçoado posteriormente por (COVER; HART, 1967), o K-vizinhos mais próximos representa uma aprendizagem com paradigma diferente dos classificadores apresentados anteriormente. O K-NN pertence à uma classe de algoritmos cujo treinamento é dito "preguiçoso" (*lazy learning*), devido à necessidade de armazenamento dos dados de treino, que serão utilizados diretamente a cada nova predição, não sendo criado um modelo ou equação a partir dos dados que se tem inicialmente. Outra forma de se referir a essa classe de classificador, é aprendizagem baseada por instância.

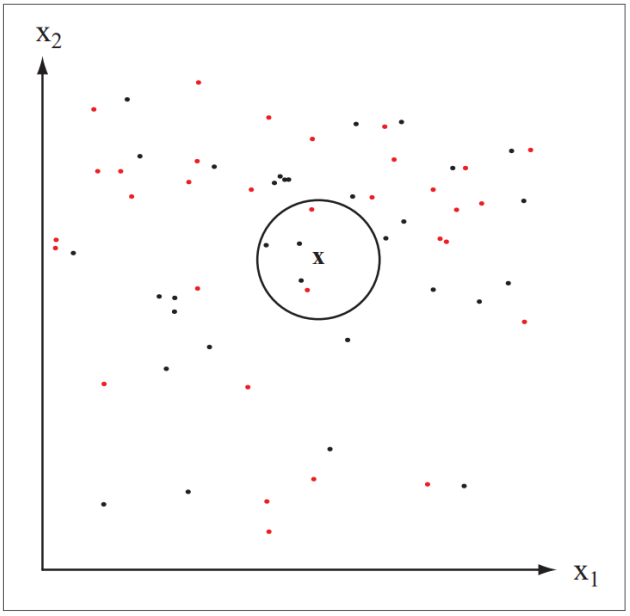
Esse algoritmo é dividido nas partes de treinamento e classificação de uma nova instância, e pode ser resumida da seguinte forma:

- **Etapa de treino**: Armazenar as instâncias de treino (valores de atributos e respectivas classes).
- **Etapa de classificação**:
    1. Calcular as distâncias da instância a ser classificada (teste) com todos as de treino;
    2. Ordenar as instâncias baseado na distância calculada, mantendo um número K de vizinhos, representando os mais próximos, na métrica de distância considerada;
    3. Contagem da classe de cada um dos K vizinhos;
    4. Atribuição da classe majoritária para a instância de teste (para evitar um empate, é preferível escolher K ímpar para uma classificação binária).

Na Figura 9 está representada a região criada em volta do dado de teste, em que serão contabilizadas as classes dos dados ali presentes, para uma instância com apenas duas características, x_1 e x_2.

Um problema associado ao K-NN está no custo computacional. A complexidade considerando d dimensões, ou seja, a quantidade de características do problema, e n a quantidade

Figura 9 – Representação básica de funcionamento do K-NN. A partir desse dado de teste, é criada uma região que contenha os K exemplos mais próximos, nesse caso, $K = 5$.



Fonte: DUDA; HART; STORK (2000)

de instâncias do treinamento, pode ser reduzida de $O(dn^2)$ (abordagem mais direta) para $O(d^3 n^{\lfloor d/2 \rfloor} \ln n)$ (DUDA; HART; STORK, 2000), tendo ainda alta dependência da dimensionalidade. Dessa forma, o custo para classificar uma nova instância pode ser alto, e praticamente todo ele está concentrado no momento da classificação, em vez de ser feito durante o treinamento, como nos outros algoritmos apresentados anteriormente.

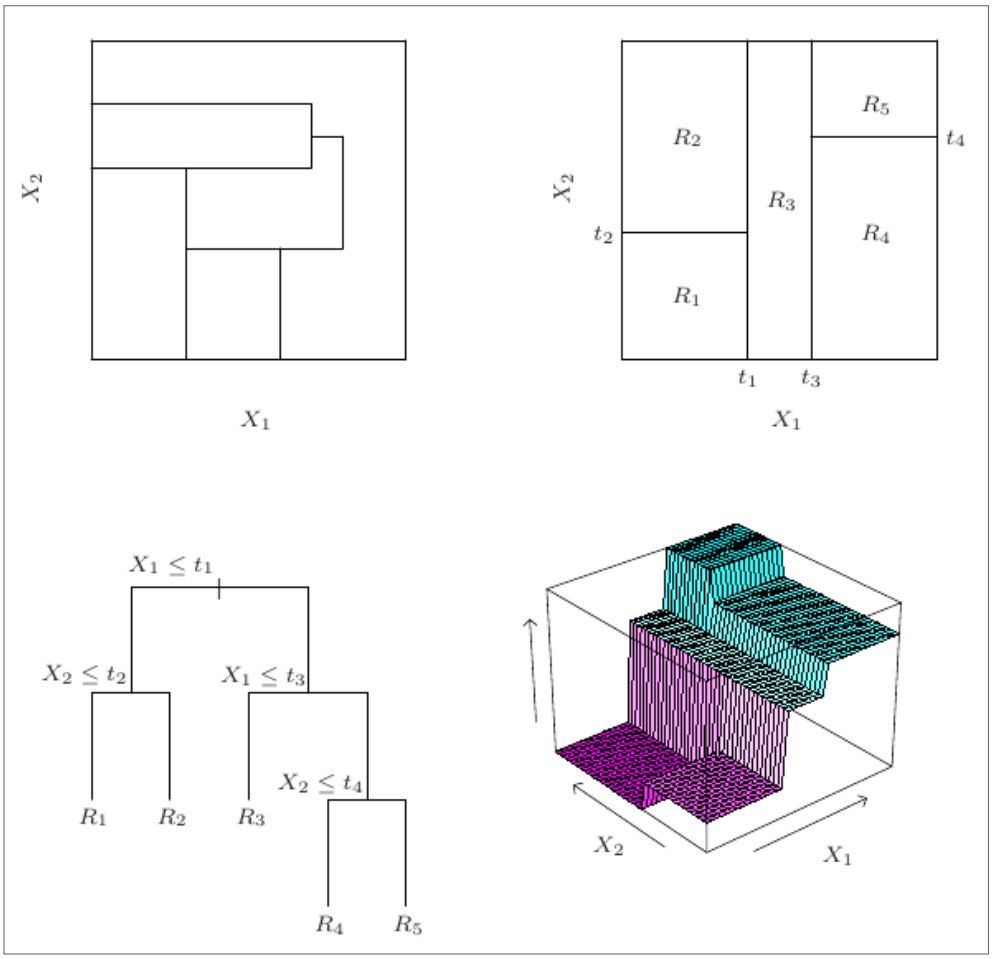
Os hiperparâmetros associados à este algoritmo são principalmente o número de vizinhos a serem considerados, K, e a métrica de distância. O valor de K está associado com um certo grau de suavização da decisão, dado que com um valor maior, são considerados mais instâncias para se tomar a decisão. A distância é outro fator extremamente importante, pois a partir dela, pode-se obter desempenhos bastantes distintos entre si (ALFEILAT et al., 2019).

Outro ponto que impacta na decisão sobre a distância a ser considerada, é que tipicamente, dentre diversas métricas, como Manhattan, Minkowski, Chebychev e a Euclidiana (ALFEILAT et al., 2019), são considerados todos os atributos para a decisão. Em contrapartida, a métrica DTW (do inglês, *Dynamic Time Warping*) (BERNDT; CLIFFORD, 1994) contorna esse possível problema, sendo destaque na área de séries temporais (GUPTA et al., 2020).

2.1.5 Algoritmos baseados em Árvore

Árvores de decisão são uma classe de algoritmos que apesar de possuírem um conceito simples, apresentam grande poder preditivo e consequentemente grande relevância para a área de AM (HASTIE, 2009). Essa classe realiza o mapeamento de variáveis de entrada, a partir de pontos de corte, em partições, como apresentado no cenário simples de apenas duas variáveis na Figura 10. Cada região criada está associada com um conjunto específico de regras, que podem ser resumidas em conjuntos de instruções do tipo "se-então". Para a implementação da árvore de decisão, existem diferentes métodos, como o ID3 (QUINLAN, 1986), CART (do inglês, *Classification and Regression Trees*) (BREIMAN et al., 1984), C4.5 (QUINLAN, 2014), entre outros.

Figura 10 – Partição do mapa a partir do algoritmo de árvore de decisão.



Fonte: Adaptado de HASTIE (2009)

Considerando o algoritmo CART para construção da árvore de decisão (RUTKOWSKI et al., 2014), ele se inicia pelo nó chamado de raiz L_0, em que estão disponíveis todos os dados

de treinamento. De acordo com uma medida de separação, é eleito um atributo que melhor separe esse público disponível em L_0. Durante o processo de aprendizagem, vão sendo criados subconjuntos de dados, S_q, definidos pela regra de decisão do nó L_q. Assim, é escolhido um atributo especificamente para esse subconjunto que melhor o segregue. Quando todos os elementos de um subconjunto S_q pertencem a uma mesma classe, o processo de divisão nesse ramo é encerrado e o nó é marcado como uma folha.

O critério para o ponto de corte nas variáveis é definido a partir da impuridade calculada, ou seja, a capacidade da variável distinguir a população pelo alvo. O coeficiente de Gini (Equação 2.14) e a entropia (Equação 2.15) são medidores de impureza bastante utilizados:

$$\sum_{i=1}^{K} \hat{p_k}(1 - \hat{p_k}), \tag{2.14}$$

$$\sum_{k=1}^{K} \hat{p_k} \log \hat{p_k}, \tag{2.15}$$

sendo $\hat{p_k}$ a proporção de instâncias pertencentes à classe k, e K o número de classes distintas. Para a classificação binária, $K = 2$, o índice de Gini se reduz à:

$$Gini_{ramo} = 1 - \hat{p_0}^2 - \hat{p_1}^2. \tag{2.16}$$

Após isso, é feita uma média ponderada do coeficiente calculado em cada ramo do nó, considerando as respectivas quantidades de dados da divisão (p_1 e p_2):

$$Gini = Gini_1 \left(\frac{p_1}{p_1 + p_2} \right) + Gini_2 \left(\frac{p_2}{p_1 + p_2} \right). \tag{2.17}$$

A variável, com seu respectivo ponto de corte, que obtiver menor impureza, ou seja, que melhor dividir o subconjunto, é então escolhida para esse nó.

Outro ponto importante é decidir quando se encerra o processo de divisão para aquele ramo. Caso, após a decisão com uma variável, se obtiver uma impuridade maior do que o cenário sem aquele ponto de decisão, então termina-se naquele ponto mesmo, fazendo dele uma folha (ponto final da árvore, naquela direção). Vale destacar, no entanto, que uma impuridade muito baixa, e até mesmo igual a zero, não necessariamente corresponde a um caso ideal, pois pode ser um indicativo de sobreajuste (em inglês, *overfitting*).

Ao final da construção da árvore, a saída gerada pela predição de uma nova instância pode tanto ser feita pela atribuição da classe majoritária da folha, ou por uma probabilidade. Este

último caso deixa a possibilidade de se atribuir um ponto de corte, e realizar o controle de métricas conflitantes entre si, como sensibilidade e especificidade, por exemplo.

Sobre os tipos de dados considerados para o algoritmo de árvore de decisão, é importante destacar que se lida mais facilmente com variáveis categóricas, quando se tem valores discretizados e com pouca quantidade. Isso porque os pontos de decisão nos nós estarão diretamente disponíveis. Quanto às variáveis numéricas, deve-se proceder de maneira distinta:

- **Variável numérica contínua**: Primeiramente ordena-se os valores distintos disponíveis, depois calcula-se a média entre esses valores adjacentes. Usa-se então esses valores de média como pontos de corte para o cálculo da impuridade. O valor de média com menor impuridade, representa o melhor valor de ponto de corte a ser escolhido.

- **Variável numérica discreta ordenada**: Para esse caso tem-se valores limitados a um conjunto, então calcula-se a impuridade para cada cenário. Os pontos de corte considerados são $\leq x, \leq x+1, \ldots, \leq x+N-1$, em que x representa o menor valor, e $x+n-1$ o segundo maior valor para essa variável. Não se considera o maior valor, pois a desigualdade incluiria toda as amostras.

Apesar de ser um algoritmo de implementação bastante simples, mas ainda muito versátil, problemas reais podem se tornar muito complexos, e nesses casos, a árvore de decisão pode apresentar o que se chama de subajuste (*underfitting*). Por isso, vários outros algoritmos utilizam a árvore de decisão como elemento mais básico em um comitê de classificadores, adicionando estratégias de métodos estatísticos como *boostrap*, *bagging* e *boosting* (WAN; YANG, 2013).

2.1.6 Comitês de classificadores

Nas seções anteriores, foram apresentados diferentes algoritmos de AM, com princípios de funcionamento diferentes entre si. Esses classificadores, apesar de terem suas vantagens e desempenhos atuando sozinhos, podem ser melhorados por métodos de comitês de classificadores (DIETTERICH, 2000), em que é construído um conjunto de classificadores que classificam um novo dado a partir do voto ponderado de suas predições.

A aplicação de comitês pode variar bastante, indo desde casos mais básicos como a ponderação por uma média, até outros mais complexos, como a construção de um algoritmo

prévio, que irá predizer qual modelo preditivo será escolhido para aquele espaço de atributos de entrada. No entanto, duas condições são consideradas necessárias e suficientes para que um comitê tenha melhor desempenho do que seus classificadores individuais (HANSEN; SALAMON, 1990). São elas: 1) Que cada um dos classificadores tenham desempenho melhor do que uma predição aleatória, e 2) sejam diversos, possuindo diferentes erros em novos dados. Portanto, as variações dos comitês exploram essas características para se sobressaírem em melhor desempenho.

Essa diversidade nos classificadores pode ser obtida treinando o algoritmo em diferentes conjuntos de dados. Uma forma direta de se manipular os dados de treinamento é por meio do método chamado *bagging* (BREIMAN, 1996), termo que vem de *bootstrap aggregation*. Para cada rodada de treinamento, é apresentado ao algoritmo de aprendizagem um conjunto contendo m instâncias, escolhidas aleatoriamente, com substituição, do conjunto de treinamento original de m itens. Dessa forma, cada conjunto escolhido contém cerca de 63,2% dos dados iniciais, dado que as instâncias podem aparecer várias vezes (DIETTERICH, 2000).

Outro importante método para implementar o comitê de classificadores é por meio do *boosting*, que é utilizado em algoritmos apresentados a seguir em 2.1.6.2 e 2.1.6.3. O princípio básico utilizado é o treinamento de vários modelos de classificadores, de forma sequencial, aproveitando-se de alguma métrica de erro calculada ao fim de cada treinamento, e utilizando como melhoria para o próximo.

2.1.6.1 Random Forests

Uma modificação que demonstrou grande impacto utilizando como base a árvore de decisão foi feita em (HO, 1995) e (BREIMAN, 2001), dando origem ao *Random Forests*. Diferentemente do método de seleção *bagging*, o *Random Forests* usa como o algoritmo base a árvore de decisão não podada.

Seu funcionamento básico consiste no treinamento de árvores individuais, usadas como base, a partir de uma amostragem do mesmo tamanho do conjunto de treinamento inicial, mas com substituição, e apresentando uma mesma distribuição. Em cada nó criado no decorrer do treinamento, é feita uma seleção aleatória dos atributos considerados para a divisão, garantindo maior robustez contra ruídos, comparado ao uso do Adaboost (FREUND; SCHAPIRE et al., 1996).

Uma característica importante é a convergência do erro de generalização (erro em dados não vistos no treinamento), com o aumento do número de árvores. Por meio da lei dos grandes

números, teorema da teoria da probabilidade, é provada essa convergência (BREIMAN, 2001), que implica que, com a adição de mais árvores no comitê, não ocorre o problema de sobreajuste.

Outra grande vantagem frente a outros métodos de formação de comitês, está na facilidade de treinamento, dado que elas podem ser treinadas de forma independente. Logo, sua velocidade de processamento pode ser favorecida ao aplicar computação paralela.

2.1.6.2 Gradient Boosting

A aplicação do método de *boosting* apresentou maior sucesso por meio do Adaboost (*Adaptive Boost*) (FREUND; SCHAPIRE, 1997). De forma similar ao *bagging*, no *boosting* também se tem um comitê de classificadores formados pelo treinamento de reamostragem dos dados, que posteriormente são combinados para formar o classificador dito "forte". No entanto, a estratégia de criação dessas reamostragens é feita de forma sequencial, pois a cada treinamento são geradas novas informações para se ter uma melhoria na classificação. Essas informações são obtidas pelos erros dos classificadores anteriores, então o algoritmo varia o peso dos dados do treinamento, para que os classificadores subsequentes foquem mais em determinadas instâncias.

O funcionamento básico do *boosting* pode ser ilustrado pela formação de três classificadores a cada iteração do algoritmo. O primeiro classificador, $C1$, é formado pelo treinamento de um conjunto aleatório dos dados iniciais. Já $C2$ é treinado com dados formados por 50% das instâncias classificados corretamente por $C1$, e os outros 50%, são dos casos incorretos, sendo portanto, um subconjunto que agrega mais informação. Para os dados utilizados para o classificador $C3$, são reunidas as instâncias classificados erroneamente por $C1$e $C2$ (WAN; YANG, 2013).

Em (SCHAPIRE, 1990) é mostrado que o erro de classificação desse algoritmo possui uma relação de erro a partir do algoritmo base utilizado para criar $C1$, $C2$ e $C3$. Considerando que o elemento base do comitê possua um erro ϵ, então o erro do classificador formado pelo voto majoritário de $C1$, $C2$ e $C3$ é dado por:

$$f(\epsilon) = 3\epsilon^2 - 2\epsilon^3, \tag{2.18}$$

que será $f(\epsilon) \leq \epsilon$ para a desigualdade $\epsilon < 0,5$. Esse valor limite para o algoritmo base é justamente uma das condições elencadas anteriormente para a formação de um comitê com

desempenho superior à de seus classificadores individuais: que seu erro seja melhor do que uma predição aleatória.

A ideia por trás do Adaboost, de ser adaptativo e utilizar reponderação da amostra de forma sequencial, é também utilizado em outros algoritmos, dando origem ao termo *arcing* (do inglês, *adaptive reweighting and combining algorithms*), e sendo estudados em (BREIMAN, 1999). Posteriormente em (FRIEDMAN, 2001), é desenvolvida e apresentada uma estrutura estatística mais geral, tendo como objetivo a minimização da perda do modelo pela adição de classificadores fracos utilizando o método de gradiente descendente. Devido à junção desses dois métodos (gradiente descendente e *boosting*), veio o nome de *Gradient Boosting Machine* (GBM), ou simplesmente *Gradient Boosting*.

A generalização desenvolvida pelo *Gradient Boosting* permite a utilização de quaisquer funções de perdas, sendo exigido apenas a sua diferenciabilidade. Dessa forma, pode ser facilmente aplicado não apenas em problemas de classificação binária, mas também em multi-classe e regressão.

2.1.6.3 XGBoost

Uma forma específica de implementação do GBM que ganhou bastante popularidade, é o XGBoost (CHEN; GUESTRIN, 2016) (do inglês, *eXtreme Gradient Boosting*). Seu foco de melhoria está na velocidade e eficiência computacional. As principais otimizações realizadas na implementação são:

- O método de cálculo de gradiente de segunda ordem desenvolvido originalmente por (FRIEDMAN; HASTIE; TIBSHIRANI, 2000) é utilizado com algumas modificações;
- Normalização dos pesos por um fator η, após cada etapa do *boosting* das árvores, similar à taxa de aprendizagem em algoritmos como Redes Neurais. Técnica introduzida por (FRIEDMAN, 2002);
- Subamostragem dos atributos, técnica que é utilizada no *Random Forests*, apresentado em 2.1.6.1.

Para discutir um pouco sobre essas modificações, serão formalizados a seguir alguns conceitos. Primeiramente, a saída do XGBoost para uma determinada instância $\mathbf{x}_i$ é dada pela adição de K funções de valores contínuos:

$$\hat{y}_i = \sum_{k=1}^{K} f_k(\mathbf{x}_i), \tag{2.19}$$

em que f_k é formado pelo espaço de árvores de regressões (CART), contendo T folhas, onde cada folha tem associado um valor contínuo, diferentemente de uma árvore de decisão tradicional. O aprendizado do algoritmo se dá pela minimização da seguinte função objetivo regularizada:

$$\mathcal{L}(\phi) = \sum_i l(\hat{y}_i, y_i) + \sum_k \Omega(f_k), \tag{2.20}$$

sendo $\Omega(f_k)$ a regularização aplicada, que é dada por

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda ||w||^2. \tag{2.21}$$

Na Equação 2.20 é realizado um somatório na função de perda $l(\hat{y}_i, y_i)$, que deve ser uma função convexa diferenciável, que mede a diferença entre a predição $\hat{y}_i$ e o rótulo real dessa instância, y_i. Quando o parâmetro de regularização é definido como zero, reduz-se para o caso tradicional do GBM.

Observando a função de perda da Equação 2.20 no processo de iteração, $\hat{y}_i^t$ é a predição do algoritmo para a i-ésima instância na t-ésima iteração. De forma correspondente, para a função objetivo regularizada da Equação 2.20, tem-se:

$$\mathcal{L}^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(\mathbf{x}_i)) + \Omega(f_t), \tag{2.22}$$

e aplicando a aproximação de segunda ordem, o XGBoost calcula essa perda de forma otimizada da seguinte forma:

$$\tilde{\mathcal{L}}^{(t)} \approx \sum_{i=1}^{n} [g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i] + \Omega(f_t). \tag{2.23}$$

Partindo agora para a avaliação da divisão de uma árvore e criação de seus ramos, divide-se entre os ramos da esquerda, I_E e da direita, I_D. A redução da perda para cada opção avaliada de divisão pode ser calculada por:

$$\mathcal{L}_{divisao} = \frac{1}{2}\left[\frac{(\sum_{i\in I_E} g_i)^2}{\sum_{i\in I_E} h_i + \lambda} + \frac{(\sum_{i\in I_D} g_i)^2}{\sum_{i\in I_D} h_i + \lambda} - \frac{(\sum_{i\in I} g_i)^2}{\sum_{i\in I} h_i + \lambda}\right] - \gamma. \tag{2.24}$$

Além das questões citadas anteriormente de otimização de funções, sua vantagem de implementação está na possibilidade de paralelização, entre outros fatores computacionais. Entre

eles, (CHEN; GUESTRIN, 2016) são apresentadas formas de acesso à padrões de cache, compressão e fragmentação de dados, de forma a possibilitar a escalabilidade para o treinamento de bilhões de instâncias com menos recursos.

2.1.6.4 Time Series Forest

Apesar de ser perfeitamente possível a utilização de cada atributo em seus respectivos pontos no tempo com um classificador tradicional, este pode ter sensibilidade a possíveis distorções da base de dados quanto ao eixo temporal. Para contornar esse possível problema, um algoritmo também baseado em árvore de decisão, mas que considera a relação de ordem entre as características, é o *Time Series Forest* (DENG et al., 2013).

A utilização de atributos intervalares, que são calculados sobre intervalos de uma série temporal, demonstrou ser bastante efetiva na utilização em classificação de séries temporais (RODRÍGUEZ; ALONSO; BOSTROM, 2001). No entanto, as possibilidades de atributos a serem criadas dessa forma são inúmeras, sendo custoso para a avaliação em cada nó de decisão de uma árvore, $\mathcal{O}(M^2)$, sendo M o número de atributos. A estratégia utilizada para redução desse espaço de características é pela subamostragem de atributos, técnica do *Random Forests*, reduzindo a complexidade para $\mathcal{O}(M)$ em cada nó.

Além dessa estratégia, o TSF também emprega uma nova medida chamada *Entrance* (do inglês, *Entropy* e *Distance*). Ela surge no contexto do critério de separação a ser adotado, que para o caso de uma árvore de série temporal, componente básico de uma floresta de série temporal (*Time Series Forest*), uma separação S a ser testada é dada por:

$$f_k(t_1, t_2) \leq \tau, \tag{2.25}$$

em que $f_k^n(t_1, t_2)$ é o valor de um atributo intervalar, como média e desvio padrão, calculada no intervalo entre t_1 e t_2. Para o limite τ, as instâncias que satisfazem a desigualdade irão para o ramo da esquerda, e os outros para o da direita. Dada essa condição, primeiramente tem-se a entropia definida da mesma forma anteriormente na Equação 2.15. O ganho de entropia, $\Delta Entropia$, para uma divisão em um nó será a diferença entre a soma ponderada da entropia em um nó filho e a entropia do nó que lhe deu origem. É considerada, em adição à essa medida de separação, uma denominada de *Margem*, que calcula a distância entre um limite considerado τ e seu valor de atributo mais próximo:

$$Margem = \min_{n=1,2,...,N} |f_k^n(t_1, t_2) - \tau|, \tag{2.26}$$

em que $f_k^n(t_1, t_2)$ é o valor de um atributo intervalar, calculada no intervalo entre t_1 e t_2 para a k-ésima instância. A combinação dessas duas medidas dá origem à *Entrance*:

$$E = \Delta Entropia + \alpha Margem, \tag{2.27}$$

em que o fator α tem como função evitar problemas cuja origem está valor excessivo de ganho de entropia.

2.2 OTIMIZAÇÃO DE HIPERPARÂMETROS

A construção de modelos preditivos de AM para resolução de problemas específicos é uma tarefa que envolve determinar não somente o algoritmo apropriado, mas também seus valores de hiperparâmetros (ELSHAWI; MAHER; SAKR, 2019). Essa segunda parte pode assumir uma grande parte do tempo, e é especialmente importante, dado que hiperparâmetros ajustados possuem desempenho melhor do que valores padrões fornecidos pelas bibliotecas de AM (OLSON et al., 2018). De forma geral, os parâmetros dos algoritmos preditivos podem ser divididos em dois grandes grupos (YANG; SHAMI, 2020):

- Aqueles que possuem uma inicialização, e são atualizados conforme o processo de aprendizagem. Um exemplo típico são os pesos das redes neurais, que são calculados pelo algoritmo de *backpropagation*;
- Os chamados hiperparâmetros, que apesar de também definidos no início do processo, não são atualizados, e constituem a configuração do algoritmo. O número de vizinhos do K-NN, a taxa de aprendizagem utilizada em diferentes algoritmos, a arquitetura de uma rede neural (número de neurônios e de camadas escondidas), são exemplos de hiperparâmetros.

A abordagem mais fácil e direta para ajuste de um algoritmo seria a aplicação de diferentes valores para o conjunto de hiperparâmetro que se deseja avaliar. No entanto, em aplicações reais isso seria inviável, por um lado pelo enorme espaço de busca, e por outro, pelo tempo consumido no processo de treinamento dos algoritmos. Isso se torna o caso especialmente para

algoritmos baseados em árvore e redes neurais, que tipicamente possuem diversos hiperparâmetros a serem otimizados (HUTTER, 2019). Devido à essas dificuldades, existem técnicas focadas para automatizar esse processo, denominadas de forma genérica de técnicas de otimização de hiperparâmetros (OHP).

Formalizando a OHP (EGGENSPERGER et al., 2015), considera-se $\lambda_1, \lambda_2, \ldots, \lambda_n$ os hiperparâmetros de um algoritmo elencados para a otimização, e $\Lambda_1, \Lambda_2, \ldots, \Lambda_n$ seus respectivos domínios. O espaço de hiperparâmetros total é definido como

$$\boldsymbol{\Lambda} = \Lambda_1 \times \cdots \times \Lambda_n. \tag{2.28}$$

A métrica utilizada para medir o erro em um conjunto de validação, com treinamento em $\mathcal{D}_{treino}$ e hiperparâmetros $\boldsymbol{\lambda} \in \boldsymbol{\Lambda}$ é dada por $\mathcal{L}(\boldsymbol{\lambda}, \mathcal{D}_{treino}, \mathcal{D}_{validacao})$. Considerando uma validação cruzada com k partes, o problema de otimização será dado pela minimização da seguinte equação:

$$f(\lambda) = \frac{1}{k} \sum_{i=1}^{k} \mathcal{L}(\boldsymbol{\lambda}, \mathcal{D}^i_{treino}, \mathcal{D}^i_{validacao}). \tag{2.29}$$

O processo de OHP (BERGSTRA; BENGIO, 2012) varia bastante para cada algoritmo de AM, pois cada um possui suas próprias características, como abordado na Seção 2.1. Uma das complexidade da OHP está justamente nesse espaço de busca a ser considerado, podendo existir alguma condicionalidade na existência de algum hiperparâmetro, que poder ser definido somente se outro assumir determinado valor. Isso acontece, por exemplo, na definição da arquitetura de uma rede neural, em que só pode ser definido o número de neurônios da i-ésima camada, se ela existir, ou seja, se o número de camadas for pelo menos i. Além disso, existe também a variedade quanto ao tipo de informação do hiperparâmetro, que pode assumir valores categóricos, discretos ou contínuos. No caso de métodos tradicionais de otimização, como o gradiente descendente, valores contínuos e diferenciáveis são exigências para sua aplicação, limitando-o bastante para aplicações gerais.

Algumas meta-heurísticas como os algoritmos evolucionários e genéticos podem ser utilizadas para OHP (YANG; SHAMI, 2020), mesmo para problemas que não sejam convexos e contínuos, e têm como principais representantes os métodos baseados em populações. A característica comum entre eles é a modificação das chamadas populações, conjunto de configurações para um algoritmo de AM, por melhorias ao serem aplicadas perturbações ou mutações, gerando combinações dos seus indivíduos para dar fruto a uma nova geração.

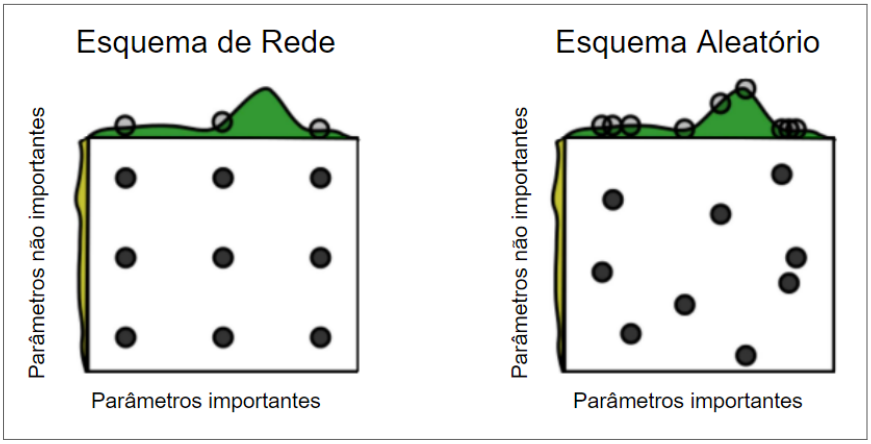
Os algoritmos genéticos (HOLLAND, 1992) possuem inspiração na teoria da evolução natural, e se utilizam da ideia de que no decorrer de gerações, os indivíduos que se adaptam melhor ao ambiente que vivem, são capazes de gerar descendentes, se tornando cada vez mais representativos. Dessa forma, levando para o contexto de AM, cada "indivíduo" representa uma solução de valores para os hiperparâmetros, dentro de uma população de diferentes soluções. A competição entre eles se dá por meio de uma métrica estabelecida, e os sobreviventes, que possuírem melhor desempenho, são combinados por meio de mutações gerando uma próxima geração de soluções. Um problema a ser enfrentado é a adição de novos hiperparâmetros no processo, como o tamanho da população, taxas de mutação e cruzamento, e o tipo de função de aptidão (*fitness*).

Outros métodos de otimização, como o *Grid Search* e *Random Search*, focam em estratégias para o espaço de busca. A partir da definição de um conjunto finito de valores para cada hiperparâmetro, $\Lambda_1, \Lambda_2, \ldots, \Lambda_n$, a estratégia de busca do algoritmo *Grid Search* considera o produto cartesiano desses conjuntos, como na Equação 2.28. Por essa combinação de possibilidades, o aumento nas avaliações cresce exponencialmente com a quantidade de hiperparâmetros considerados, incorrendo em um problema de dimensionalidade. Uma forma de se contornar isso é pelo algoritmo *Random Search* (BERGSTRA; BENGIO, 2012) que gera amostras aleatórias a partir do espaço de busca definido. Essa estratégia funciona melhor quando há uma diferença de importância entre os hiperparâmetros, pois para um número fixo de B casos a serem avaliados, enquanto o *Grid Search* consegue avaliar $B^{1/N}$ opções para cada um dos N hiperparâmetros, o *Random Search* amplia essas possibilidades para B diferentes valores (HUTTER, 2019). Esse benefício de explorar as diferentes importâncias pode ser ilustrado na Figura 11.

Tanto no *Random Search* como no *Grid Search*, os conjuntos de valores são avaliados de forma independente, inclusive sendo um fator que os beneficia pela possibilidade de paralelização. A abordagem por otimização Bayesiana (EGGENSPERGER, 2013), no entanto, é um processo que considera o resultado de uma avaliação anterior para guiar a estratégia de busca para o próximo conjunto de hiperparâmetros a serem testados. Dessa forma, ele se beneficia do histórico formado, e pode explorar a proximidade de valores que estão obtendo maior desempenho.

Os dois principais componentes da Otimização Bayesiana são a função de aquisição, que mede o valor da próxima iteração, e as funções ou modelos substitutos (*surrogate functions*), utilizadas para diminuir o custo de processamento para avaliar as configurações (BERGSTRA et

Figura 11 – Representação de tentativas para as técnicas *Grid Search* e *Random Search*.



Fonte: Adaptado de BERGSTRA; BENGIO (2012)

al., 2011). Modelos substitutos comumente utilizados são os processos Gaussianos, *Random Forests* e o Estimador de Árvore Parzen (EAP) (YANG; SHAMI, 2020). O ponto que maximiza a função substituta se torna o valor em que a função verdadeira deve ser avaliada.

De forma geral, a otimização Bayesiana utiliza um modelo probabilístico $\mathcal{M}$ para simular o desempenho $f(\boldsymbol{\lambda})$ obtido por meio da configuração de hiperparâmetros $\boldsymbol{\Lambda}$. O processo iterativo de ajuste desse modelo é feito usando dados prévios para selecionar novos valores de hiperparâmetros, $\boldsymbol{\lambda}$.

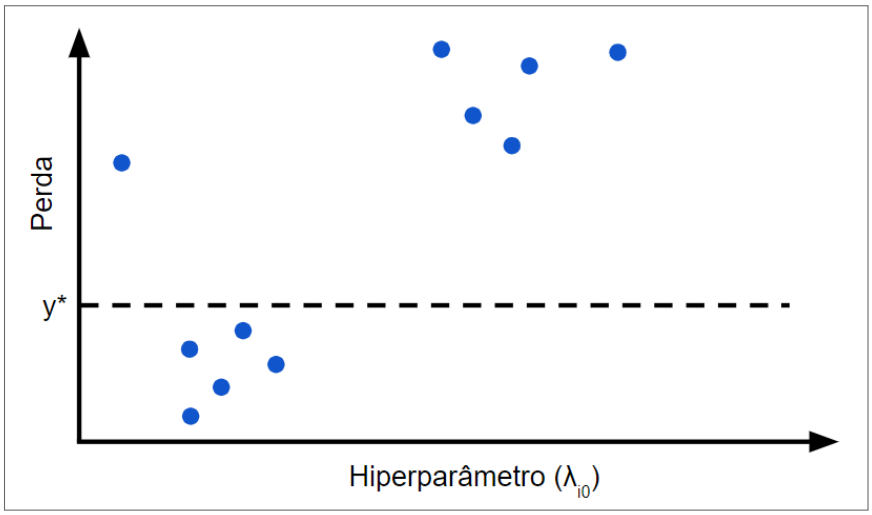
Em se tratando da estratégia EAP (BERGSTRA et al., 2011), é realizada a modelagem de $p(y|x)$, usando o teorema de Bayes visto anteriormente na Equação 2.3, a partir de de $p(x|y)$ e $p(y)$. A distribuição de $p(x|y)$ é definida como:

$$p(x|y, D) = \begin{cases} l(x), \text{ se } y < y^* \\ g(x), \text{ se } y \geq y^*, \end{cases} \tag{2.30}$$

em que $l(x)$ é a densidade formada pelas instâncias que a função de perda $f(\mathbf{x}_i)$ sejam menores que o valor de limiar y^*, e $g(x)$ os demais casos, como ilustrado na Figura 12. A escolha de y^* pelo EAP é feita de forma que seja maior do que o melhor valor observado de $f(x)$, tal que $y < y^*$, e portanto encontrar conjuntos de hiperparâmetros que estejam na distribuição $l(x)$ para que a perda seja menor que y^*.

Dada essa visão geral sobre conceitos e definições sobre OHP, vale destacar também ferramentas que implementam tais algoritmos, e também facilitam seu uso em termos práticos de resolução de problemas. A forma de amostragem de valores para os hiperparâmetros varia para cada caso, podendo uma ferramenta implementar mais de uma estratégia, deixando ao

Figura 12 – Considerando o hiperparêmtro λ_{i0}, os valores abaixo do ponto de corte y^* seguem a distribuição $l(x)$, e acima, $g(x)$.



Fonte: Adaptado de AGRAWAL (2021)

usuário a opção de escolha. Em algumas mais recentes, também são implementados métodos de poda, que monitora resultados intermediários das tentativas, e descarta aquelas que não são promissoras. Na lista a seguir são apresentados algumas principais, e suas respectivas referências:

- SMAC (HUTTER; HOOS; LEYTON-BROWN, 2011);
- Hyperopt (BERGSTRA et al., 2015);
- Google Vizier (GOLOVIN et al., 2017);
- Autotune (KOCH et al., 2018);
- Optuna (AKIBA et al., 2019).

Para esse trabalho optou-se pelo Optuna, dada sua facilidade de implementação dos espaços de busca, e tendo desempenho em estado da arte (AKIBA et al., 2019). No Optuna se tem disponível diferentes algoritmos de amostragem de hiperparâmetros e pode ser utilizada estratégias de poda. Para o algoritmo de amostragem de hiperparâmetros, que seleciona um conjunto de valores para cada iteração, se tem como padrão o Estimador de Árvore de Parzen, selecionado para a OHP dos algoritmos.

3 REVISÃO DA LITERATURA

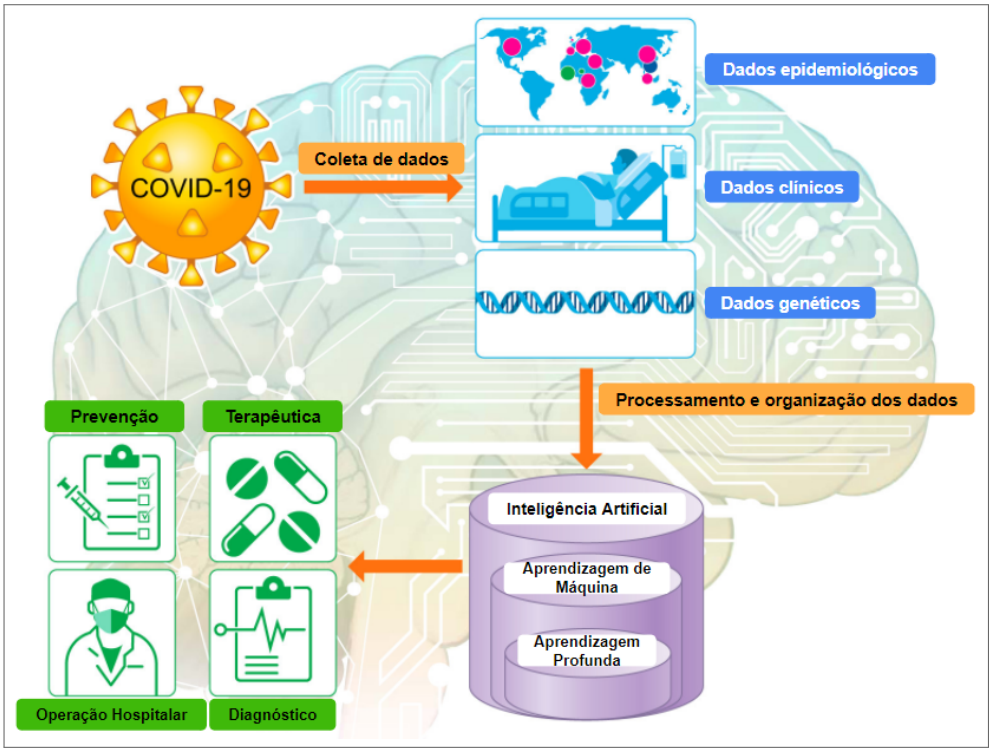
O surgimento da COVID-19 é bastante recente, datando pouco mais de dois anos. Ainda assim, muito se produziu cientificamente e dada a situação global, em uma velocidade bastante rápida. A literatura científica considerada para bibliografia abrangeu, primeiramente na Seção 3.1, aplicações gerais de tecnologia voltadas ao diagnóstico de COVID-19 e pesquisas envolvendo a análise matemática e estatística do método PCR. Uma diversidade de exames foram e estão sendo utilizados para se diagnosticar a doença, e como consequência, uma abrangência de tecnologias computacionais estão assistindo à esse desafio. De forma mais específica, na Seção 3.2 são apresentados trabalhos que lidaram também, assim como este trabalho, com dados da leitura de fluorescência do exame de RT-qPCR. Dada essa maior similaridade nesses trabalhos mais recentes, são destacados os pontos de diferença e as lacunas desses trabalhos que foram preenchidas aqui.

3.1 APLICAÇÃO DE TECNOLOGIAS PARA DIAGNÓSTICO DE COVID-19

Logo após a declaração de que a COVID-19 tinha se tornado uma pandemia (GHEBREYESUS, 2020), gerou-se uma mobilização global para criação de estratégias de enfrentamento e minimização dos danos. Isso pode ser constatado pela solicitação emitida pelos Estados Unidos, com a colaboração de institutos de pesquisa e empresas de tecnologia, para o desenvolvimento de novas técnicas de mineração de dados para dar suporte à pesquisa de temas relacionados à COVID (ALIMADADI et al., 2020). Esse esforço em conjunto foi responsável pelo desenvolvimento do CORD-19 (do inglês, COVID-19 *Open Research Dataset*) (WANG et al., 2020a), que atualmente agrega mais de 500.000 artigos científicos sobre o tema. O editorial (ALIMADADI et al., 2020), lançado no início da pandemia, apresenta e discute ferramentas de Inteligência Artificial que podem ser utilizadas no combate à COVID-19, como mostra o esquema da Figura 13. Dentre elas, estão inclusos a classificação taxonômica de genes, predição de sobrevivência em pacientes graves e descobrimento de drogas em potenciais para o combate à doença. Outro conjunto de pesquisas citadas envolve o desenvolvimento de novas técnicas para o diagnóstico, baseado no padrão respiratório (WANG et al., 2020b), imagens de tomografia computadorizada do tórax (GOZES et al., 2020) e sistema de detecção baseado na tecnologia CRISPR (METSKY et al., 2020). Por fim, além de citar o potencial e o avanço

dessas pesquisas, é destacada a importância e o desafio na geração de bases de dados que dão suporte à essas pesquisas.

Figura 13 – Aplicações de Inteligência Artificial no combate à COVID-19.



Fonte: Adaptado de ALIMADADI et al. (2020)

Considerando que o método de PCR se encontra bastante disseminado na área ciências biomédicas e análise de expressão gênica (YUAN et al., 2006), não se limitando à detecção de COVID-19, naturalmente existem trabalhos de antes mesmo da pandemia que nele realizam análises estatísticas e aplicação de ferramentas de AM. Em (YUAN et al., 2006), por exemplo, são avaliadas diferentes análises estatísticas nos dados de PCR, comparando a eficiência da amplificação para diferentes amostras, propondo ao final um modelo de controle de qualidade. Essa avaliação se faz importante pois modelos matemáticos e estatísticos para processamentos dos dados são incorporados aos *softwares* das máquinas que realizam o exame.

Com o objetivo de melhorar a taxa de sucesso na realização do exame PCR convencional (1ª geração do método), (CORDARO et al., 2021) coleta dados de configurações e parâmetros de diferentes laboratórios usados para realização dos exames. Para a amplificação padronizada por PCR (desconsiderando outros tipos de PCR, como o qPCR), esses parâmetros precisam ser otimizados para cada novo experimento, sendo de responsabilidade dos pesquisadores que o conduzem defini-los, com base na combinação entre conhecimento científico, experiências prévias, e tentativa e erro. Enquanto a taxa de sucesso do PCR projetado por humanos está em

torno de 55 a 63%, os autores, por meio de técnicas de AM otimizam os parâmetros necessários para conduzir o experimento, indicam uma melhoria de 17 a 26% na taxa de sucesso.

A testagem em massa se tornou necessária para combater a disseminação de uma doença facilmente contagiosa (MANABE; SHARFSTEIN; ARMSTRONG, 2020). Dado esse contexto, uma variedade de testes de diagnóstico vão sendo desenvolvidos e continuamente melhorados, incluso os classificados como teste diagnóstico no ponto de atendimento (*Point-of-Care Testing*, PoCT). Há uma tentativa de, idealmente, se alcançar o nível de desempenho de um teste como o PCR, mas a partir de outras fontes de dados. Em (ZOABI; DERI-ROZOV; SHOMRON, 2021) é proposto um modelo de AM que prediz se há a infecção pelo SARS-CoV-2 em um exame de PCR a partir de oito perguntas básicas. O objetivo do modelo é atuar em uma triagem da população e encaminhamento de pessoas que precisam ser priorizadas, dada a predição do modelo. As perguntas do questionário utilizado, que vêm a se tornar variáveis do modelo, são basicamente indicativos da presença de sintomas: tosse, febre, dor de cabeça, dor de garganta e falta de ar, e também de características fisiológicas: gênero, se a idade é maior que 60 anos, e sobre interação com infectados. Em estudo semelhante, (GOODMAN-MEZA et al., 2020) tem o objetivo de reduzir e otimizar o tempo e utilização de recursos de exames de PCR e imagens de raio-X, que demandam maior tempo e custo. A ferramenta de triagem desenvolvida utiliza dados demográficos e laboratoriais básicos, que são usados para o treinamento dos algoritmos.

Uma revisão da aplicação de AM para diagnóstico de COVID-19 utilizando dados clínicos e/ou dados laboratoriais de rotina é realizada em (ALJAME et al., 2020). Nessa revisão, é apresentada de forma sumarizada uma tabela contendo a fonte da base de dados (em sua maioria, proprietárias e de pequeno volume), o número de características consideradas, o modelo preditivo utilizado, e as métricas de desempenho acurácia, sensibilidade e especificidade. As classes de algoritmos preditivos que se destacam são o *Random Forest*, Regressão Logística, Máquina de Vetores de Suporte e Redes Neurais Artificiais. Em um dos estudos considerados, é apresentado um estudo de viabilidade (BRINATI et al., 2020), em que são considerados valores hematológicos advindos de exames de sangue de rotina. Neste estudo, foram desenvolvidos dois modelos classificadores de AM para detecção da doença. A base de dados utilizada é composta de 279 casos, extraídos de forma aleatória de pacientes, em que para cada caso foram coletados os dados sobre idade, gênero, valores extraídos de exames de sangue de rotina, e o resultado do teste de RT-PCR, obtido por *swab* em nasofaringe. Os algoritmos considerados para o treinamento dos dados foram os seguintes:

- Árvore de decisão;
- *Random Forest*;
- *Extremely Randomized Tree*;
- K-Vizinhos Mais Próximos;
- Regressão Logística;
- Bayes ingênuo;
- Máquinas de Vetores de Suporte.

A avaliação dos modelos foi feita a partir de diferentes métricas de classificação, mas a partir da discussão com médicos envolvidos no estudo, foi decidido por priorizar acurácia e sensibilidade, dado que um paciente que está com COVID-19, mas foi classificado erroneamente como negativo, vem a ser mais nocivo à sociedade.

Por fim, ainda utilizando dados clínicos e/ou laboratoriais, em (ALJAME et al., 2021) o autor introduz uma nova abordagem de técnica de AM para esse contexto. É utilizado o método *Deep Forest*, proposto em (ZHOU; FENG, 2018), em que múltiplos classificadores são usados em múltiplas camadas. Essa estratégia, assim como em outros comitês de classificadores, procura promover a diversidade de preditores para melhoria do desempenho. Os resultados experimentais foram obtidos a partir de duas bases de dados públicas, atingindo 99,5% de acurácia, 95,28% de sensibilidade e 99,96% de especificidade.

3.2 PREDIÇÃO DE DIAGNÓSTICO POR MEIO DE DADOS DO RT-QPCR

Além do explorado na Seção 3.1 de utilização de variadas técnicas de IA voltadas para o combate ao COVID-19, aqui serão abordados os trabalhos específicos que consideram o RT-qPCR, no objetivo de melhorar ou antecipar a sua predição por meio dos dados de fluorescência.

Primeiramente, em um trabalho (GUNAY; GOCERI; BALASUBRAMANIYAN, 2016) anterior à pandemia de COVID-19, são considerados os dados de fluorescência do RT-qPCR para detecção de patógenos em geral, usando um conjunto de dados de referência. Nele, é proposto um método dividido em duas etapas, a primeira, para detecção se a amostra é positiva, e em caso afirmativo, um algoritmo de regressão é utilizado para o cálculo de C_t, ciclo limite (do inglês, *Threshold Cycle*), ponto no qual se cruza o limite de amplificação da fluorescência.

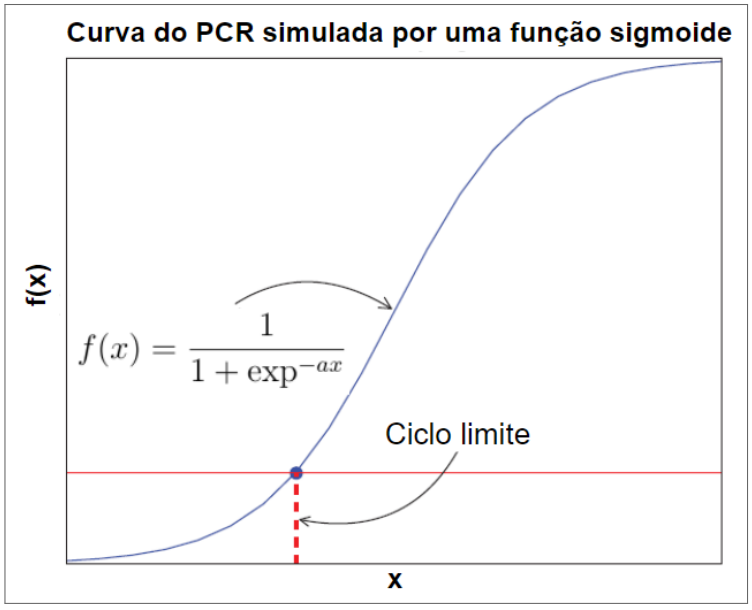
A metodologia utilizada é pela identificação matemática da forma da curva de amplificação, com um algoritmo de ajuste de curva, utilizando como base uma função sigmoide, expressa por

$$f(x) = \frac{1}{1 + \exp^{-ax}}, \tag{3.1}$$

que pode ser vista na Figura 14, e logo depois se aplica o algoritmo de agrupamento *K-Means*, para cálculo do ponto de corte.

O treinamento para cálculo dos pontos de corte são feitos um conjunto de 38 amostras. Depois de calculados os parâmetros, o algoritmo geral de cálculo para o C_t é testado frente a 400 amostras, e comparado com um *software* de padrão ouro denominado *PCRMiner*.

Figura 14 – Função sigmoide simulando a curva de fluorescência do PCR, com marcação do ciclo limite.



Fonte: Adaptado de GUNAY; GOCERI; BALASUBRAMANIYAN (2016)

Apesar de existirem diferentes métodos para análise e ajuste dos dados fluorescência, como o visto anteriormente, eles não são robustos para lidar com variações estocásticas do sinal (ruído) (ALOUANI et al., 2021). Para a determinação do ciclo limite (C_t), tipicamente uma linha de base é calculada de forma antecipada, e os ruídos nesse período podem se tornar impeditivos para uma correta interpretação dos resultados. Principalmente no cenário de pandemia em que um número grande desse exame é realizado, a presença do ruído nos dados pode se tornar mais frequente, e consequentemente o erro na classificação.

Em (ALOUANI et al., 2021) os autores seguem por um método em que não se utiliza o C_t para interpretar os dados do exame e retornar um diagnóstico. É treinado um algoritmo de

redes neurais convolucionais, denominado *qPCRdeepNet*, baseado nas leituras de fluorescência obtidas durante o RT-qPCR, sendo a mesma abordagem para obtenção dos dados para diagnóstico do COVID-19 utilizada neste trabalho. Esse tipo de arquitetura para redes neurais é bastante utilizado em tarefas de classificação de imagens (LECUN; BENGIO; HINTON, 2015), e para se adaptar a isso, os dados de fluorescência são vistos como uma imagem. Os dados utilizados vieram das leituras de fluorescência (valores R_n) para até 40 ciclos, extraídos de arquivos EDS, normalizados pelo método de *z-score* e plotado em uma imagem de 299x299 píxeis. Além disso, uma linha tracejada no valor zero do *z-score* foi adicionada em todas as imagens, de forma à fornecer um referencial de escala para o algoritmo.

A base de dados para treinamento e validação do estudo feito em (ALOUANI et al., 2021) foi realizado em um ensaio CDC 2019-nCov, contendo 7.763 amostras, em que cada uma possui um alvo ($N1$, $N2$ e controle) com suas próprias curvas de fluorescência, totalizando 20.961 de treino e 2.328 de teste, ambos com aproximadamente 43% de casos positivos. A marcação entre positivos e negativos foi feita com base nos valores de C_t, em que se tem um caso positivo com $C_t \leq 40$, e negativo para $C_t > 40$. Esses valores são calculados pelo *software* versão v.2.3 do sistema *Applied Biosystems 7500 Fast PCR* da ThermoFisher.

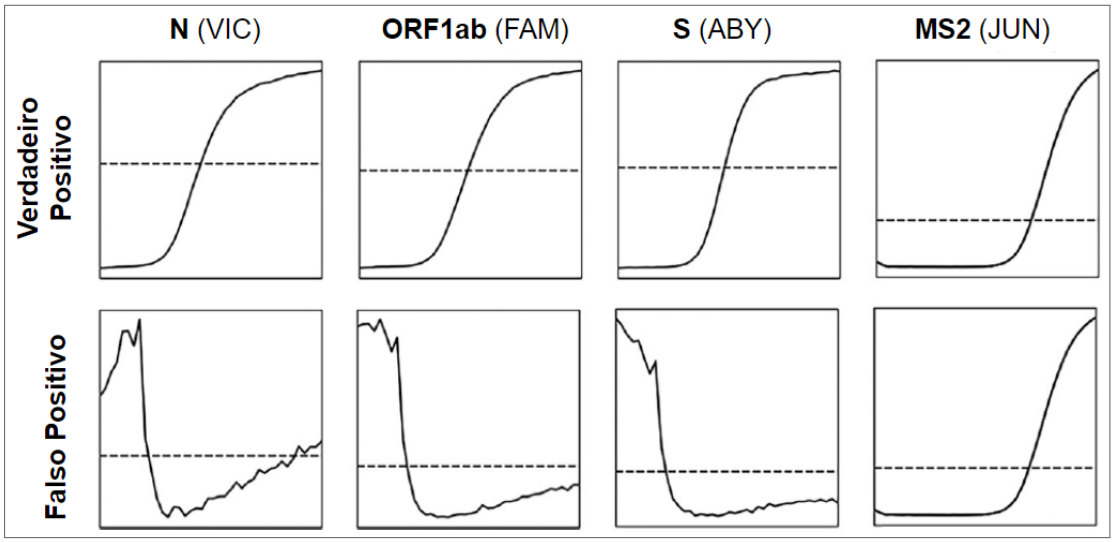
Primeiramente foram avaliados para teste amostras advindas do ensaio TaqPath COVID-19 Combo Kit, analisadas pelo ABI COVID-19 *Interpretative Software* 1.3, contendo 50.146 amostras clínicas para quatro canais sendo 3 deles os genes ORF1ab, N e S, e outro de controle interno, o bacteriófago MS2, gerando portanto 200.584 imagens, com ilustração na Figura 15. Cada amostra foi classificada como detectado, não detectado e inconclusivo, por meio do *Applied Biosystems* (ABI) COVID-19 *Interpretive Software* v1.3 Outros testes foram realizados, sem retreinamento, para demonstrar a aplicabilidade do algoritmo em três diferentes ensaios de RT-qPCR:

- TaqMan SARS-CoV-2, Flu A, Flu B multiplex - 950 amostras;
- TaqMan SARS-CoV-2, Flu A/B, multiplex RSV - 1.306 amostras;
- TaqMan detecção de mutação JAK2 V617F Cast-PCR - 87 amostras.

Os principais resultados de desempenhos estariam disponíveis em material suplementar, que não foi possível ser encontrado pelo artigo online. No entanto, o artigo tem como conclusão um resultado favorável para a abordagem alternativa ao método tradicional de interpretação dos

resultados do RT-qPCR pelo C_t, sendo o ponto principal a redução de casos falso-positivos, e baixando a taxa de casos inconclusivos, que seriam repetidos.

Figura 15 – Gráficos das fluorescências normalizadas por *z-score* para os genes N, ORF1ab, S e o controle, MS2. Representam casos de curvas tanto positivas, como falso positivas, com comportamento que pode ser mal interpretado por algoritmos.



Fonte: Adaptado de ALOUANI et al. (2021)

Durante a escrita desse trabalho, tendo os experimentos já sido realizados, foi encontrada uma pesquisa publicada (LEE et al., 2022) com mesma abordagem, com o intuito de redução do tempo do RT-qPCR para diagnóstico de COVID-19 com dados de fluorescência obtidos no decorrer do exame. Nela é enfatizada, assim como neste trabalho de dissertação, a necessidade de redução do tempo de diagnóstico da doença, que até então não se tinha alguma alternativa para se acelerar o processo.

Também com a utilização dos dados brutos de fluorescência de cada ciclo do exame, o algoritmo é treinado para predizer a classificação do exame antes que se conclua o tempo previamente estabelecido, que nesse caso totalizam 40 ciclos. O algoritmo de AM utilizado é o LSTM (*Long short-term memory*), uma arquitetura de aprendizagem profunda de redes neurais recorrentes, que pode trabalhar diretamente com os dados tabulares que se dispõe. Foram treinados vários modelos, cada um com uma quantidade máxima de ciclos a serem utilizados. O primeiro modelo utilizou somente o primeiro ciclo para ser treinado, e assim predizer o resultado do exame. O segundo, os dois primeiros ciclos, e assim por diante, tendo o quadragésimo modelo sido treinado com os 40 ciclos.

A base de dados utilizada no desenvolvimento possui 5.810 instâncias, cada uma com 40 características. Desse total, apenas 181 casos, equivalente a 3% da base, são positivos. Com

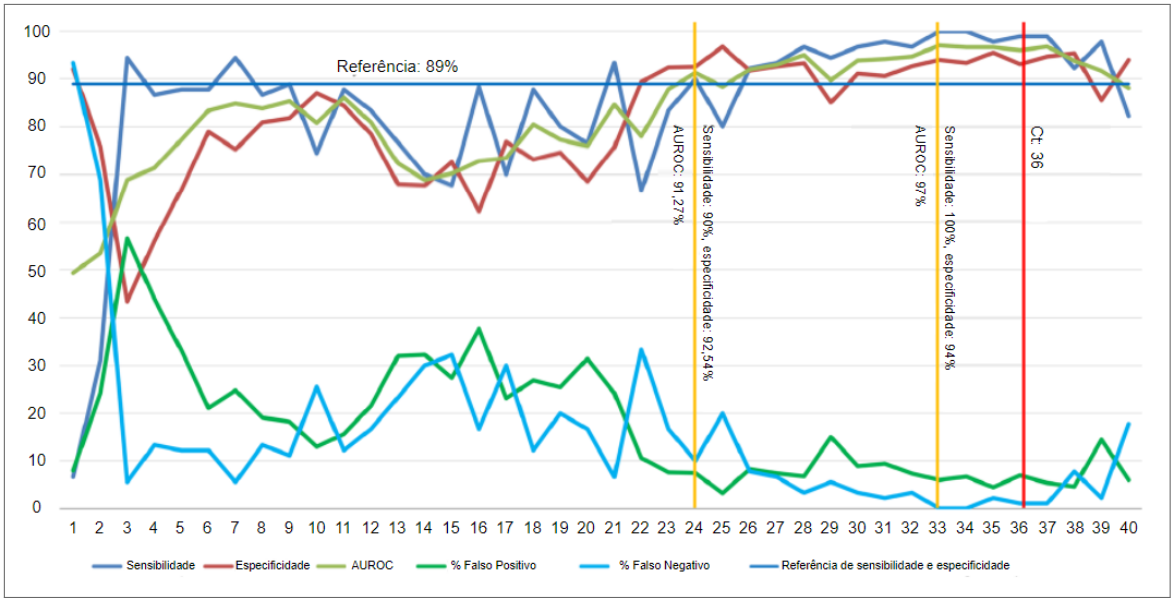
a divisão feita em 50% para treino e teste, os resultados foram apresentados considerando somente 90 casos positivos. Ainda, não foi especificado sobre quais genes estão sendo avaliadas as curvas de fluorescência, e se estão sendo incluídas curvas de controle. Essa distinção se faz importante pois as curvas de amostras de controle têm comportamento mais estável frente aos dados de amostras coletadas por humanos. Dessa forma, o desempenho obtido com a inclusão dessas curvas mais estáveis pode ocultar o que seria o desempenho nas amostras humanas. Outro ponto importante, é que não foi especificado como foi feita a marcação do alvo utilizado como treinamento. Diferentes protocolos podem ser adotados por diferentes laboratórios, na intenção de se descartar falsos positivos, e também no intuito de refazer exames considerados inconclusivos.

São utilizadas métricas como sensibilidade, especificidade e AUROC para comparar o desempenho dos modelos, e o Valor Preditivo Positivo (VPP), Valor Preditivo Negativo (VPN) e acurácia para apresentar os diferentes resultados em três países. No trabalho não é mencionada a forma de decisão do ponto de corte, para então serem calculadas as métricas sensibilidade e especificidade, por exemplo. Como o LSTM é um algoritmo que tem como saída uma probabilidade, a decisão sobre o diagnóstico sobre se é positivo ou negativo, depende desse fator importante. Para comparativo dos desempenhos obtidos, é considerado um valor de referência para a sensibilidade sendo 89%, de acordo com a meta-análise feita em (KIM; HONG; YOON, 2020). A Figura 16 apresenta o resumo do comparativo do desempenho do treinamento para o crescente número de ciclos, incluindo o valor de referência.

O número de ciclos necessário para exceder o valor de referência de 89%, considerando tanto a especificidade e sensibilidade, foi 24, com AUROC, sensibilidade e especificidade de 91,27%, 90% e 92,54%. A partir disso, o autor indica a possibilidade de redução do tempo de diagnóstico pelo exame por quase a metade. O maior valor de AUROC foi atingido para o modelo com 33 ciclos, com valor de 97%, e sensibilidade e especificidade de 100% e 94%, respectivamente. A partir dos resultados gerais obtidos, o autor sugere a utilização do valor de C_t de 36 em vez de 40, para uma redução do tempo por meio do desenvolvimento do modelo.

Os resultados foram gerados para uma base fixa de teste, sem a realização de reamostragem e troca dos dados de treino e teste para avaliar a capacidade de generalização. Essa análise limitada do desempenho do algoritmo pode incorrer em erros de sobreajuste durante o treinamento, escolhendo hiperparâmetros que maximizem o resultado em um conjunto específico de teste.

Figura 16 – Comparação de métricas de desempenho entre os modelos para diferentes ciclos.



Fonte: Adaptado de LEE et al. (2022)

3.3 CONSIDERAÇÕES

Os trabalhos relacionados abordados neste capítulo foram primeiramente sobre as pesquisas e análises sobre o método de PCR, utilizado décadas antes do surgimento da COVID-19. Posteriormente foram expostos diferentes exames de diagnóstico utilizados para detecção de COVID-19 e os trabalhos recentes que utilizam dos dados de fluorescência do RT-qPCR para suporte ao diagnóstico.

Foi apresentado que para detecção da COVID-19 são utilizadas, além do PCR, outras formas de diagnóstico que são aplicadas a depender do contexto. No caso de uma simples triagem, usualmente são feitas perguntas sobre a presença de sintomas, como tosse, febre e dor de cabeça e características fisiológicas, como gênero e idade. Aumentando a disponibilidade de recursos disponíveis, podem ser observadas a utilização de exames de sangue de rotina para essa tarefa de detecção, e exames de imagens como raio-X. Vale destacar que as pesquisas realizadas para observar o diagnóstico com esses métodos alternativos, utilizam o PCR como referência de desempenho.

Tratando dos trabalhos que utilizam os dados do RT-qPCR na Seção 3.2, vale o destaque para (LEE et al., 2022), que utiliza de abordagem similar na antecipação dos ciclos do exame. No entanto, ao apresentar os desempenhos com a variação de ciclos, se observa variações abruptas, e até mesmo perda de desempenho com adição de ciclos em determinada faixa. Dada

a metodologia do trabalho, esse comportamento provavelmente se deve à baixa volumetria dos dados de treino e teste, falta de especificação do tipo de amostra (se foram incluídas amostras de controle, por exemplo), e de técnicas de reamostragem, para fins de cálculo de desvio padrão e testes estatísticos.

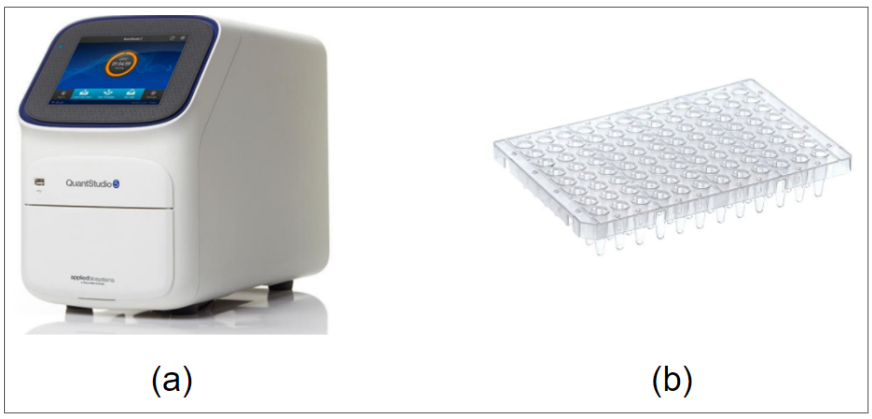
4 DESENVOLVIMENTO DA BASE DE DADOS

Este capítulo apresenta o desenvolvimento da base de dados utilizada para o treinamento dos algoritmos preditivos de AM. Nesse processo de construção da base, é descrita inicialmente a obtenção dos dados do exame na Seção 4.2, e na Seção 4.3 a estruturação das características em uma visão de modelagem, de modo que sejam utilizados para treinamento dos algoritmos. Posteriormente, em 4.4 são detalhados as filtragens necessárias, seguindo os protocolos estabelecidos pelo laboratório do NUPIT, para um aprendizagem dos algoritmos sem viés de erro.

4.1 CONCEITOS PRELIMINARES

Antes de descrever o processo de criação da base de dados, faz-se necessário definir alguns conceitos do contexto do exame de RT-qPCR. A começar pelos equipamentos, ilustrados na Figura 17, em que se vê uma placa de poços que contém diversas amostras, tanto coletadas de humanos, quanto amostras de controle interno. É importante destacar que esse controle auxilia na identificação de amostras contendo substâncias que possam interferir no resultado final. A placa é inserida na máquina para ser realizado um processo de reação química das amostras, chamado de amplificação.

Figura 17 – Equipamentos utilizados para o exame qPCR (a) QuantStudio™ 5 System (b) Placa de 96 poços (0,2 mL).



Fonte: Thermo Fisher

Na configuração de experimento de RT-qPCR realizados, são usados dois alvos para detecção do SARS-CoV-2: N1 e N2, regiões presentes no gene N do vírus. Além da avaliação de amplificação desses genes do vírus, também é utilizado como alvo o RNaseP (RP), para

controle de qualidade de extração do RNA, sendo um controle endógeno marcador de metabolismo das células humanas (FREIRE-PASPUEL; GARCIA-BEREGUIAIN, 2021). A partir dessa configuração de alvos, são organizados tipos de amostras na placa a ser inserida no exame, para que se tenha um controle de comportamentos esperados para cada caso. Para isso, há uma identificação para cada poço da placa do tipo de amostra a qual ela se refere. Uma posição da placa pode ser identificada com as seguintes classificações:

- CONT HUMAN: Amostras utilizadas para controle, em que deve ser observada a amplificação do alvo RP;
- *Negative Template Control* (NTC) ou CONT EXTR: Amostras de controle cuja amplificação tanto dos alvos do vírus, quanto do controle endógeno devem ser negativas;
- CONT SARS2: Amostras contendo parte do vírus, utilizado para impor a amplificação dos genes N.

A técnica de PCR é caracterizada por uma reação química de amplificação de alvo. Ou seja, na amostra inicial se tem potencialmente o alvo (N1, N2 ou RP), os reagentes, e a reação química ocorre na tentativa de replicar a quantidade desse alvo. A amplificação ocorre em divisões de tempo regulares denominados ciclos de máquina, que para esse caso, totalizam 45 ciclos. Em (BUSTIN; NOLAN, 2020) é dito que o tempo médio do exame para o SARS-CoV-2 é de 1 hora e 27 minutos, para realização de 40 ciclos. Nesse cenário, considerando uma linearidade do tempo, se teria 1 hora e 38 minutos para a totalidade de 45 ciclos.

Além das amostras coletadas e reagentes que compõem a reação química, na técnica de qPCR (PCR quantitativo) é adicionado um agente fluorescente. Esse elemento adicional é inserido com o objetivo de se medir a quantidade de alvo no decorrer da reação, dado que se tem uma maior fluorescência para uma maior quantidade do alvo.

4.2 OBTENÇÃO DOS DADOS

A base de dados foi construída a partir de exames de RT-qPCR, realizados pelo NUPIT-UFPE. Esses exames obtidos estão compreendidos no período de junho a dezembro de 2020, com amostras provenientes de diferentes regiões, tanto da capital Recife como atendendo a demandas de municípios do interior. Foi disponibilizado pelo NUPIT uma totalidade de 1.392 arquivos de planilhas eletrônicas de um mesmo equipamento utilizando as mesmas

configurações. Aind foi necessário realizar uma triagem desses arquivos, para filtrar casos a serem repetidos segundo protocolos seguidos pelo laboratório, e casos em que ocorreu algum tipo de contaminação, de acordo com o processo descrito na Seção 4.4.

As informações geradas ao final de uma rodada de exame nos equipamentos ilustrados na Figura 17 podem ser exportados por meio de um arquivo em formato de planilha eletrônica, contendo cinco abas. Em cada uma delas, há um cabeçalho contendo as informações gerais de configuração, além de conferir se foram ou não realizadas as diferentes calibrações necessárias.

As principais configurações utilizadas que podem ser citadas para a realização dos exames, estão detalhados na Tabela 1. Os dados que cada aba contém, além do cabeçalho, estão descritos a seguir:

Tabela 1 – Configurações do *Sample Setup* dos experimentos de RT-qPCR.

Configuração	**Valor**
Block Type	96-Well 0.2-mL Block
Chemistry	TAQMAN
Experiment Type	*Standard Curve*
Instrument Type	QuantStudio™ 5 System
Passive Reference	ROX
Quantification Cycle Method	C_t
Signal Smoothing On	True
Stage/Cycle where Ct Analysis is performed	Stage2, Step2

Fonte: Elaborada pelo autor (2022)

- *Sample Setup*: Nesta aba, consta as características para os 96 poços da placa inserida na máquina, como localização, nome da amostra, nome do alvo, e qual o *Reporter* e o *Quencher* utilizado.

- *Raw Data*: Para cada posição do poço e para cada ciclo de máquina, são informados os valores das colunas $x1 - m1$, $x1 - m2$, $x2 - m2$, $x2 - m3$, $x3 - m3$, $x4 - m4$, $x4 - m5$, $x5 - m5$ e $x5 - m6$. Eles representam os valores brutos quantitativos de fluorescência.

- *Amplification Data*: Tabela contendo também para cada posição de poço e para cada ciclo, o nome do alvo (N1, N2 ou RP), e os valores de Rn e ΔRn. Aqui estão concentrados os dados com relação temporal.

- *Multicomponent Data*: Valores de ROX e FAM para cada ciclo e posição de poço.

- *Results*: Tabela contendo as 96 linhas correspondente aos poços. Valor de C_t, C_t *threshold*, C_q *Conf*, *Baseline Start*, *Baseline End*, e o alvo de interesse, *Amp Status*.

4.3 MONTAGEM DA VISÃO DE MODELAGEM

A partir dos milhares de arquivos obtidos de exames de diagnóstico contendo as informações descritas na Seção 4.2, fez-se necessária a criação de uma visão de modelagem. A partir dessa nova estruturação dos dados, tornou-se possível treinar algoritmos de classificação a partir das características de interesse, que tenham uma relação temporal com os ciclos da máquina, associadas à resposta de amplificação do gene. Dado o grande número de arquivos, foi escrita uma rotina na linguagem de programação Python para automatizar a leitura das planilhas, extraindo as informações necessárias, contidas nas abas *Amplification Data* e *Results*.

Dos arquivos em que foi possível realizar a leitura dos dados, foi realizado um cruzamento das informações das duas abas, por meio da chave "$well\_position$", que identifica a amostra. Assim, ao final tem-se uma tabela com, no máximo, 96 linhas (uma para cada posição utilizada na placa), com suas respectivas características. Dentre essas características, pode-se destacar as relativas à serie temporal dos ciclos executados, sendo valores absolutos, Rn e relativos, ΔRn, totalizando 90 colunas com relação temporal. Além dessas, tem-se também outras colunas cuja descrição encontra-se sumarizada na Tabela 2.

Tabela 2 – Descrição das características da base de dados.

Nome da coluna	Descrição
sample_name	Chave de identificação – Nome da amostra
well_position	Chave de identificação – Posição do poço na placa
ct_mean	Valor calculado pela máquina ao fim dos 45 ciclos
ct_threshold	Valor calculado pela máquina ao fim dos 45 ciclos
cqconf	Valor numérico entre 0 e 1
target_name	Identificação do tipo de alvo (N1, N2 ou RP)
baseline_start	Valor inteiro identificando início da região de *baseline*
baseline_end	Valor inteiro identificando fim da região de *baseline*
rn_01 a *rn_45*	Valores absolutos de fluorescência de cada ciclo
delta_rn_01 a *delta_rn_45*	Valores relativos de fluorescência de cada ciclo
amp_status	Resposta utilizada, que identifica a amplificação

Fonte: Elaborada pelo autor (2022)

Como apresentado na Tabela 2, os dados do RT-qPCR são calculados de forma absoluta

e relativa. Por um lado, a quantificação absoluta mostra-se importante em casos específicos em que o número exato da cópia da transcrição precisa ser determinado. No entanto, em casos de estudos de enfermidades, a análise relativa se sobressai (YUAN et al., 2006). Esse uso é importante principalmente para a comparação de diferentes amostras de teste, como nos casos de exame para diagnóstico, em que é realizada a normalização para compensação de diferentes quantidades de material biológico nas amostras de teste (KUBISTA et al., 2006).

4.4 FILTRAGENS DA BASE E DIVISÃO DE AMOSTRAS

Na composição da placa de poços, são inseridas amostras coletadas de humanos por meio de *swab* nasofaríngeo, e também elementos para o controle de qualidade. São eles o NTC (do inglês, *Negative Template Control*), controle negativo que não se espera amplificação devido ao fato de ser preparado somente com reagentes, sem a adição de amostra. Nesse caso, quando há algum sinal de amplificação para o NTC, considerando o sistema TaqMan utilizado, trata-se de indicativo que houve uma contaminação, e o experimento deve ser refeito. O outro controle presente é o endógeno, em que ao contrário do anterior, se espera sempre uma amplificação.

Na primeira etapa de filtragem, são removidos alguns poucos casos com a resposta de amplificação como inconclusiva, ou valor nulo. Como está sendo considerado uma aprendizagem supervisionada, é necessária uma resposta para cada caso.

A segunda etapa refere-se ao procedimento de controle do exame, em que são verificadas as amostras de controle negativo e endógeno. Assim, em uma rodada de exame de PCR, caso seja identificada uma contaminação, será necessária a repetição do procedimento para todas as amostras consideradas. As condições para essa repetição desse caso são apresentadas na Tabela 3 ocorre quando se tem uma amplificação da amostra NTC, ou para a não amplificação de uma amostra do endógeno (vírus SARS-CoV-2). Com esse procedimento, foram detectados 305 arquivos em que houve contaminação (21,9%), dos 1.392.

Tabela 3 – Cenários de casos em que é necessária a repetição do exame para toda a placa de poços.

Tipo de amostra	**N1**	**N2**	**RP**
SARS-Cov-2	Amplifica	Amplifica	Não amplifica
CONT HUMAN	Não amplifica	Não amplifica	Amplifica
CONT EXTR e NTC	Não amplifica	Não amplifica	Não amplifica

Fonte: Elaborada pelo autor (2022)

Para a última etapa de filtragem, considerou-se o protocolo utilizado pelo NUPIT descrito na Tabela 4, que indica alguns cenários em que se deve realizar novamente o exame para uma amostra humana específica, e não mais para toda a placa do exame. As amostras que atendem essas condições são removidas, pois, de acordo com o protocolo adotado, seu resultado não é de confiabilidade.

Tabela 4 – Cenários de casos em que é necessária a repetição do exame para a amostra.

	N1	**N2**	**RP**
Caso 1	$C_t < 40$	$C_t > 40$	-
Caso 2	$C_t > 40$	$C_t < 38$	-
Caso 3	Não amplifica	Não amplifica	$C_t > 38$

Fonte: Elaborada pelo autor (2022)

Servindo de apoio ao protocolo de filtragem das amostras, foi criada para cada instância uma coluna denominada 'REPETICAO', em que é atribuída esse valor caso tenha sido necessária a repetição para aquela amostra. Em caso afirmativo, classifica entre um dos casos descritos na Tabela 4. Isso foi feito para fins de controle, e os números podem ser vistos na Tabela 5. Dessa forma, somente as amostras contendo os valores "NAO" e "Amostras de Controle" são elegíveis, as demais são eliminadas.

Tabela 5 – Quantidades e tipos de repetição para a amostra de exames analisados.

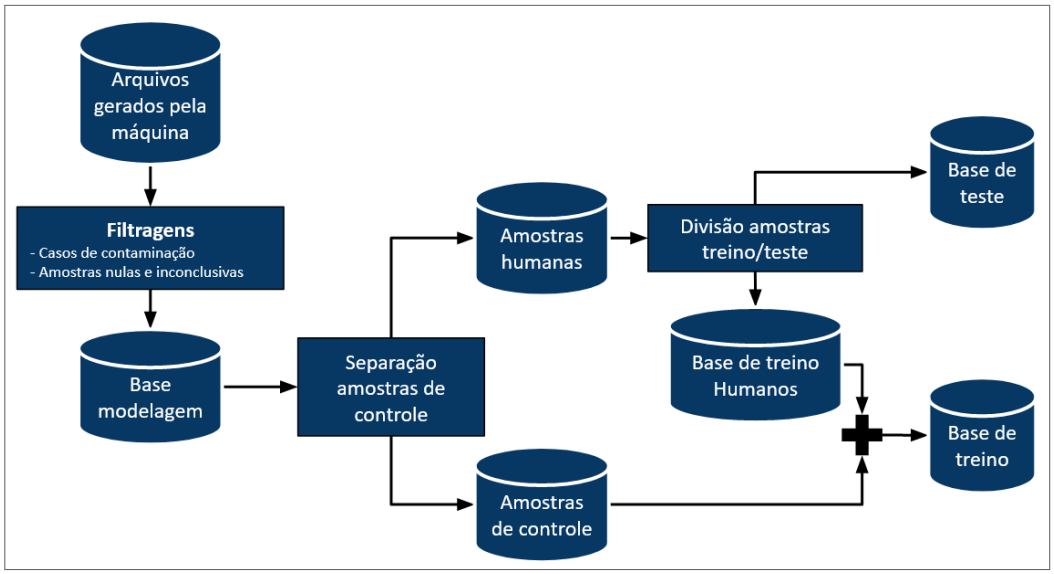
Repetição	**Quantidade**	**Percentual (%)**
NAO	62.676	81,81%
Amostra de Controle	13.041	17,02%
Caso 1	761	0,99%
Caso 2	90	0,12%
Caso 3	39	0,05%
Total	76.607	100%

Fonte: Elaborada pelo autor (2022)

Por fim, com a devida filtragem dos casos, é necessária também a divisão das amostras entre humanas e de controle. Essa divisão foi realizada pensando-se na aprendizagem e na avaliação dos algoritmos preditivos, pois pretende-se estimar o desempenho desconsiderando-se as amostras de controle, simulando um desempenho real. Assim, a base de treino é constituída de ambas as amostras, humana e de controle, mas a de teste apresenta somente amostras humanas.

O processo geral discutido nesta seção pode ser resumido no diagrama da Figura 18. Dispondo das 75.716 amostras elegíveis, ao final tem-se na base de teste 15.646 e 60.070 na base de treino. Esta, ainda pode ser dividida entre treino e validação, indo para as quantidades de 45.052 e 15.018, respectivamente. Sobre o balanceamento das classes, a base de amostras humanas possui 52% de casos positivos, enquanto as amostras de controle, 25%. Devido à menor volumetria das amostras de controle, sua taxa de positivos não impacta tanto na distribuição de classe das bases de treino e teste, que por sua vez possuem taxa de positivos por volta de 46% e 53%, respectivamente.

Figura 18 – Processo de filtragem e divisão das amostras para definição das bases de treino e teste.



Fonte: Elaborada pelo autor (2022)

5 CLASSIFICAÇÃO ANTECIPADA PARA EXAMES RT-QPCR

Neste capítulo são apresentados os principais resultados deste trabalho, que tratam da aprendizagem dos algoritmos de classificação do exame de RT-qPCR, em diferentes cenários de número de ciclos. Nesse âmbito, inicialmente é apresentado em 5.1 o método proposto para a redução do número de ciclos do exame, que pode ser entendido como uma série temporal. Em seguida, em 5.2 são apresentados os algoritmos elencados para o estudo, e a metodologia utilizada para treinamento, de forma que se analise se houve ou não uma generalização do aprendizado, em predição de casos não conhecidos anteriormente. Para efeito de comparação de desempenho, na Seção 5.3 discute-se as métricas tipicamente utilizadas na área de AM, em especial no contexto de diagnóstico, como sensibilidade e especificidade, que diferentemente da AUROC, faz-se necessário estabelecer um ponto de corte, tópico discutido na Seção 5.4.

Depois de contextualizado e discutido as questões relativas aos algoritmos, métodos de treinamento e medição de desempenho, são apresentados os resultados iniciais para o caso geral de treinamento com a totalidade dos ciclos. Nessa etapa, mesmo considerando os valores padrões de hiperparâmetros, todos os algoritmos, com exceção do NB obtiveram AUROC maior que 99%, chegando a 99,94% com XGBoost, 99,93% com o TSF, e 99,89% com o MLP. Com esses resultados iniciais, o estudo é continuado com a antecipação da série temporal, e consequente observação do impacto no desempenho, tendo ao final uma visão da variação do número de ciclos, indo dos 45 até somente 10 ciclos, para o treinamento dos algoritmos. Dado que a tarefa de classificação fica mais complexa, é realizado na Seção 5.7 uma otimização de hiperparâmetro dos algoritmos que se destacaram nos experimentos precedentes, citados anteriormente. Os valores médios de AUROC para o MLP e o TSF ficam próximos de 99%, tendo o MLP apresentado maior benefício em face da otimização. Por fim, em 5.9 serão analisadas as respostas da predição dos algoritmos, por meio de testes de hipótese, em que é comparado se há ou não uma melhoria estatisticamente significativa, para alguns cenários elencados. Com esse respaldo estatístico, vê-se que de fato MLP e TSF obtiveram maior desempenho que o XGBoost, mas não foi possível rejeitar a hipótese nula de igualdade de distribuição de AUROC entre eles.

5.1 MÉTODO PROPOSTO DE REDUÇÃO DE CICLOS

A partir da base de dados com a devida separação entre as amostras humanas e de controle, pretende-se utilizar o conceito de classificação antecipada de séries temporais (*Early Classification of Time Series*). Enquanto o objetivo principal de uma tarefa de classificação de série temporal está na maximização de alguma métrica de desempenho, a finalidade desse conceito é classificar uma série temporal incompleta assim que seja possível, de forma a atingir um nível de desempenho satisfatório desejado. Assim, o desafio se torna a otimização de dois objetivos que são conflitantes entre si, aumento de desempenho e diminuição do tempo (ou quantidade de dados da série temporal) (GUPTA et al., 2020).

O método adotado para abordar a classificação antecipada consiste no treinamento de diferentes classificadores com uma variação no número de ciclos utilizados. Tem-se inicialmente um desempenho considerado ótimo a partir da totalidade de ciclos, ou seja, da série temporal completa, que nesse caso são 45 ciclos. Posteriormente, serão treinados algoritmos preditivos com janelas mais curtas de tempo, e avaliado o decréscimo no desempenho, até atingir um nível que se considere aceitável.

5.2 SELEÇÃO DE ALGORITMOS E TREINAMENTO

No capítulo 2 já foi apresentada de forma geral alguns conceitos básicos que fundamentam a área de Inteligência Artificial, e mais especificamente a de Aprendizagem de Máquina. Além disso, foram apresentados diversos algoritmos, que utilizam diferentes abordagens matemáticas e estatísticas para se realizar a classificação. Esses algoritmos elencados para o estudo são os seguintes:

- *K-Nearest Neighbors*;
- Bayes Ingênuo (NB);
- SVM;
- Redes Neurais Artificiais na arquitetura de MLP (*Multilayer Perceptron*);
- Árvore de decisão;
- *Gradient Boosting*;

- XGBoost;
- *Time Series Forest*.

Foram escolhidos para este trabalho não só algoritmos de uso geral, usualmente utilizados para classificação supervisionada, para se ter um *benchmark*, como também algoritmos específicos que levam em consideração o contexto e os dados. É importante destacar a classe de algoritmos que leva em consideração a relação de ordem, nesse caso, temporal, entre as características. O algoritmo denominado *Time Series Forest* (TSF) (DENG et al., 2013), une aspectos das árvores de decisão, com os próprios de séries temporais, extraindo características como média, desvio padrão e inclinação de uma determinada janela temporal, especificada por um hiperparâmetro. Aqui neste trabalho foi utilizada uma implementação em Python (FAOUZI; JANATI, 2020). Um algoritmo clássico para a área de classificação de séries temporais é o 1-NN DTW (BAGNALL et al., 2016), adaptação do K-NN, utilizando o número de vizinhos $K = 1$, e a distância DTW (do inglês, *Dynamic Time Warping*) (ERUHIMOV; MARTYANOV; TUV, 2007). Apesar da grande utilização na área, possui alto custo computacional como desvantagem. Isso pode ser ilustrado por experimentos preliminares feitos com a predição de 1.365 instâncias, que teve uma média de 64 segundos para que o algoritmo classificasse cada caso. Extrapolando a predição para todos os 15.694 registros de teste, isso resultaria em mais de 11 dias de processamento, tornando ainda mais demorada a análise com reamostragens. Além disso, pensando no cenário em produção, seria necessário executar o algoritmo para 96 amostras, impactando no tempo de exame, indo de encontro ao objetivo inicial de redução de tempo. Outro ponto em desfavor do 1-NN DTW, é a comparação com o TSF feita em (DENG et al., 2013), em que este o supera em desempenho nos experimentos realizados.

Sobre a metodologia utilizada para treinamento dos algoritmos, foi feita a separação da base entre treino e teste (com volumetria de 70% e 30%, respectivamente) segundo a formação da Figura 18, em que na base de treino constam tanto amostras humanas como de controle, enquanto que na base de teste há apenas amostras humanas. Dessa forma, o aprendizado se dá no conjunto de treino, e as predições do classificador são feitas na base de teste, para serem coletadas as métricas de desempenho. Para se ter uma maior confiabilidade nos resultados, utiliza-se de reamostragens da base de dados disponível, variando os conjuntos de treino e teste, de forma aleatória, repetindo-se 100 vezes o procedimento. Assim, o processo de aprendizagem do algoritmo será realizado em diferentes conjuntos de amostras, retirado-se um possível viés de desempenho em um conjunto de amostras mais favorável, e considera-se

ao final um desempenho médio obtido, sumarizado pela média e desvio padrão das métricas consideradas.

5.3 MÉTRICAS DE DESEMPENHO

Os algoritmos selecionados pertencem à uma classe em que se realiza uma probabilidade de classificação. Nesses casos, para cada instância o algoritmo prevê uma probabilidade dele pertencer à uma classe. Para o exemplo binário de probabilidade de classificar como positivo, pode-se então definir um limiar, também denominado ponto de corte, e caso a probabilidade de predição seja maior que esse valor, assume-se que foi classificado como positivo. A definição desse limiar possibilita o controle de métricas de classificação, priorizando alguns casos a depender do contexto aplicado. Para o cenário de exame de uma doença contagiosa, é preferível priorizar a sensibilidade, ou seja, identificar os casos verdadeiramente positivos, para que medidas de isolamento sejam mais efetivas. Vale ressaltar que a variação do ponto de corte implica que métricas como sensibilidade e especificidade irão em direções opostas, enquanto uma melhora, a outra tende a piorar.

Definindo-se um ponto de corte para a probabilidade de classificação do algoritmo, pode-se então ter a previsão das classes, positivo ou negativo, e comparando-se com o rótulo real da amostra, são estabelecidos os seguintes conceitos:

- **Verdadeiro Positivo** (VP): Quantidade de casos positivos que foram corretamente identificados como positivo;
- **Verdadeiro Negativo** (VN): Quantidade de casos negativos que foram corretamente identificados como negativo;
- **Falso Positivo** (FP): Quantidade de casos que foram identificados como positivos, mas eram negativos;
- **Falso Negativo** (FN): Quantidade de casos que foram identificados como negativos, mas eram positivos.

Com esses conceitos, pode ser gerada uma matriz de confusão, representada na Tabela 6, que dá origem à diversas métricas de classificação, mas aqui são destacadas as seguintes:

- Acurácia: Percentual considerando todos os acertos sobre o total de casos considerados,

$$acc = \frac{(VP + VN)}{(VP + VN + FP + FN)}. \tag{5.1}$$

- Sensibilidade: A origem vem justamente do contexto médico, já que em geral pretende-se medir o quão sensível um teste de detecção é para a presença da doença. Percentual de verdadeiro positivo, calculado pela quantidade de acertos de casos positivos, sobre o total de positivos preditos,

$$sensib = \frac{VP}{(VP + FN)}. \tag{5.2}$$

- Especificidade: Aparece em conjunto com a sensibilidade, e trata-se da taxa de verdadeiro negativo,

$$espec = \frac{VN}{(VN + FP)}. \tag{5.3}$$

Tabela 6 – Representação do conceito de matriz de confusão.

	Resultado da predição	
Real	**Amplifica**	**Não amplifica**
Amplifica	Verdadeiro Positivo (VP)	Falso Negativo (FN)
Não Amplifica	Falso Positivo (FP)	Verdadeiro Negativo (VN)

Fonte: Elaborada pelo autor (2022)

A expansão da análise além da matriz de confusão, que volta para a pontuação de probabilidade, pode ser realizada pela análise ROC (PROVOST; FAWCETT; KOHAVI, 1998). Essa análise pode ser resumida de forma quantitativa pela métrica associada AUROC. Essa métrica é também usualmente utilizada para otimização de parâmetros durante o treinamento de um modelo preditivo. Seu valor varia de 0,5, representando uma decisão aleatória, a 1, modelo que prediz corretamente todos os casos.

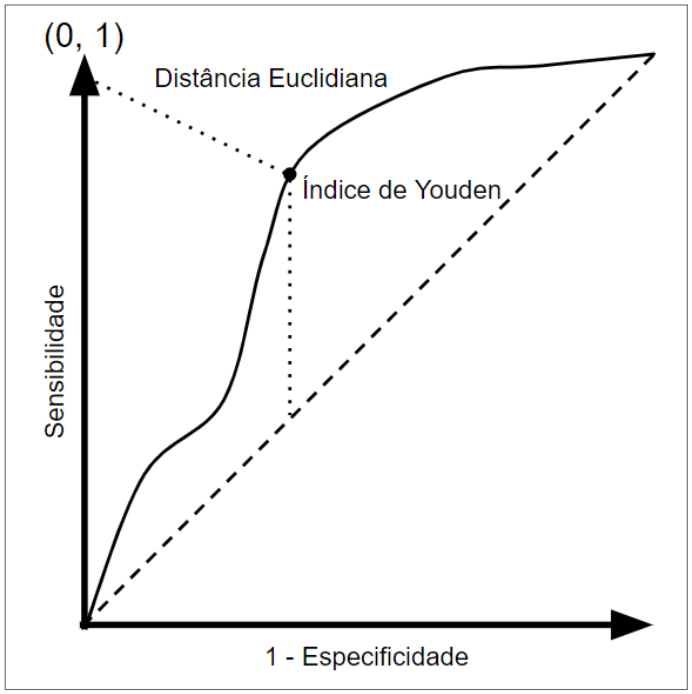
5.4 DECISÃO PELO PONTO DE CORTE ÓTIMO

As métricas de classificação calculadas por meio da matriz de confusão necessitam de predição binária, decidida por meio de ponto de corte. Diferentes métodos foram desenvolvidos para decidir o melhor valor de ponto de corte para as predições contínuas dos algoritmos (HAJIAN-TILAKI, 2013). A análise da curva ROC, utilizada para basear decisões de ponto de corte (HABIBZADEH; HABIBZADEH; YADOLLAHIE, 2016), possibilita trabalhar com diferentes

pontos do gráfico, sendo que cada ponto está associado um ponto de corte, que representa um compromisso entre métricas conflitantes, como a sensibilidade e especificidade. A partir dessa análise da curva ROC, três principais métodos podem ser utilizados para a determinação do ponto de corte, são eles:

- Distância Euclidiana: Nesse método se minimiza a distância geométrica da curva com o lado esquerdo do gráfico, ponto (0, 1), como mostrado na Figura 19. Essa distância é interpretada como $d^2 = (1 - sensib)^2 + (1 - espec)^2$.
- Índice de Youden (YOUDEN, 1950): Maximiza a linha vertical entre a curva ROC e a linha diagonal (Figura 19), resultando em: $Y = sensib + espec - 1$. Também pode ser interpretado como o valor máximo de soma entre sensibilidade e especificidade;
- Método de Liu (ou método do produto): Maximiza o produto entre sensibilidade e especificidade.

Figura 19 – Demonstração gráfica dos pontos ótimos, Índice de Youden e Distância Euclidiana.



Fonte: Elaborada pelo autor (2022)

Os métodos anteriores não definem uma importância maior entre uma métrica ou outra (sensibilidade ou especificidade). Caso seja de interesse priorizar uma das duas, pode-se utilizar uma abordagem Bayesiana, em que é necessário saber a probabilidade pré-teste, que determina a chance de um paciente ter uma doença, estimada antes da realização do teste, e também

o custo de incorrer em uma classificação errada (HABIBZADEH; HABIBZADEH; YADOLLAHIE, 2016). Apesar desse método se mostrar mais interessante e preciso, as premissas para o cálculo desses fatores não são universais, dependendo tanto da região, quanto do período e da doença considerada, por isso, são raramente utilizadas na literatura de diagnósticos (INDRAYAN, 2017). Dessa forma, pretendendo-se generalizar os resultados a serem obtidos, pela facilidade de cálculo e grande uso na literatura, optou-se pela maximização do índice de Youden.

5.5 EXPERIMENTOS INICIAIS

Para iniciar os experimentos com a base de dados desenvolvida, o primeiro cenário é a utilização da totalidade de ciclos, ou seja, 45. Os resultados são obtidos executando cada algoritmo em uma iteração de 100 amostragens diferentes de treino e teste. Para esse cenário base, com as configurações padrões dos algoritmos, chega-se à Tabela 7, sumarizando as métricas de desempenho.

Tabela 7 – Desempenho dos algoritmos com hiperparâmetros padrões para 45 ciclos.

Modelo	**AUROC**	**Acurácia**	**Especificidade**	**Sensibilidade**
K-NN	99,78 ± 0,04	99,45 ± 0,05	99,48 ± 0,12	99,42 ± 0,13
NB	59,28 ± 16,22	61,40 ± 15,27	19,40 ± 32,73	99,14 ± 0,50
SVM	99,83 ± 0,04	99,36 ± 0,06	99,41 ± 0,14	99,32 ± 0,12
MLP	99,89 ± 0,04	99,45 ± 0,05	99,53 ± 0,14	99,38 ± 0,14
Árvore de Decisão	99,23 ± 0,07	99,23 ± 0,07	99,24 ± 0,10	99,21 ± 0,10
Gradient Boosting	99,86 ± 0,04	99,50 ± 0,05	99,53 ± 0,11	99,46 ± 0,10
XGBoost	99,94 ± 0,02	99,57 ± 0,05	99,57 ± 0,09	99,57 ± 0,08
TSF	99,93 ± 0,02	99,60 ± 0,04	99,62 ± 0,08	99,57 ± 0,08

Fonte: Elaborada pelo autor (2022)

5.6 ANTECIPAÇÃO DA SÉRIE TEMPORAL

Com o intuito de realizar o procedimento de classificação de forma mais rápida, serão apresentados os cenários de antecipação da série temporal, contendo os dados do exame RT-qPCR. Com a apresentação de resultados análoga à da Seção 5.5, a seguir tem-se as tabelas dos desempenhos com o treinamento e predição considerando a antecipação dos ciclos.

Como primeiro resultado, apresentado na Tabela 8 com 40 ciclos, não são observadas

quedas significativas nas métricas consideradas, tanto de AUROC, quanto as outras que levam em consideração o ponto de corte ótimo adotado. Isso indica que os últimos 5 ciclos contêm pouca informação decisiva para os classificadores na maioria esmagadora dos casos, dado que para alguns algoritmos, a diferença de AUROC se dá somente na segunda casa decimal.

Tabela 8 – Desempenho dos algoritmos com hiperparâmetros padrões para 40 ciclos.

Modelo	AUROC	Acurácia	Especificidade	Sensibilidade
K-NN	99,74 ± 0,04	99,42 ± 0,05	99,36 ± 0,13	99,48 ± 0,13
NB	54,88 ± 11,32	57,21 ± 10,54	10,76 ± 23,15	98,94 ± 0,87
SVM	99,79 ± 0,05	98,82 ± 0,09	98,57 ± 0,19	99,05 ± 0,21
MLP	99,86 ± 0,03	99,41 ± 0,06	99,49 ± 0,12	99,35 ± 0,13
Árvore de Decisão	99,08 ± 0,07	99,08 ± 0,07	99,09 ± 0,11	99,07 ± 0,11
Gradient Boosting	99,85 ± 0,04	99,44 ± 0,06	99,45 ± 0,12	99,42 ± 0,12
XGBoost	99,93 ± 0,02	99,52 ± 0,05	99,49 ± 0,10	99,54 ± 0,09
TSF	99,93 ± 0,02	99,56 ± 0,04	99,56 ± 0,08	99,56 ± 0,09

Fonte: Elaborada pelo autor (2022)

Seguindo a redução temporal, agora no cenário de 35 ciclos, tem-se a Tabela 9. Apesar dos valores de AUROC da maioria dos modelos estarem acima de 99, é observada uma queda maior na sensibilidade, assim como na acurácia.

Tabela 9 – Desempenho dos algoritmos com hiperparâmetros padrões para 35 ciclos.

Modelo	AUROC	Acurácia	Especificidade	Sensibilidade
K-NN	98,97 ± 0,09	97,84 ± 0,11	98,47 ± 0,35	97,27 ± 0,37
NB	53,11 ± 8,76	55,52 ± 8,03	7,39 ± 19,02	98,76 ± 1,85
SVM	99,11 ± 0,07	96,63 ± 0,15	97,31 ± 0,39	96,03 ± 0,38
MLP	99,47 ± 0,05	97,87 ± 0,11	98,47 ± 0,27	97,33 ± 0,28
Árvore de Decisão	96,66 ± 0,14	96,66 ± 0,14	96,55 ± 0,25	96,77 ± 0,19
Gradient Boosting	99,47 ± 0,06	97,93 ± 0,11	98,35 ± 0,20	97,56 ± 0,21
XGBoost	99,59 ± 0,04	98,11 ± 0,10	98,53 ± 0,21	97,74 ± 0,22
TSF	99,69 ± 0,04	98,32 ± 0,10	98,72 ± 0,18	97,96 ± 0,20

Fonte: Elaborada pelo autor (2022)

Nos resultados de treinamentos com trinta ciclos, apresentados na Tabela 11, é observada uma diminuição maior da AUROC em todos os modelos de algoritmos, indo todos para valores abaixo de 99. O algoritmo com melhor desempenho nesse caso, considerando os hiperparâmetros padrões, é o *Time Series Forest*. Nessa janela de ciclos, vê-se uma mudança mais drástica,

sendo portanto os ciclos da série temporal maiores que 30 importantes na decisão se há ou não amplificação.

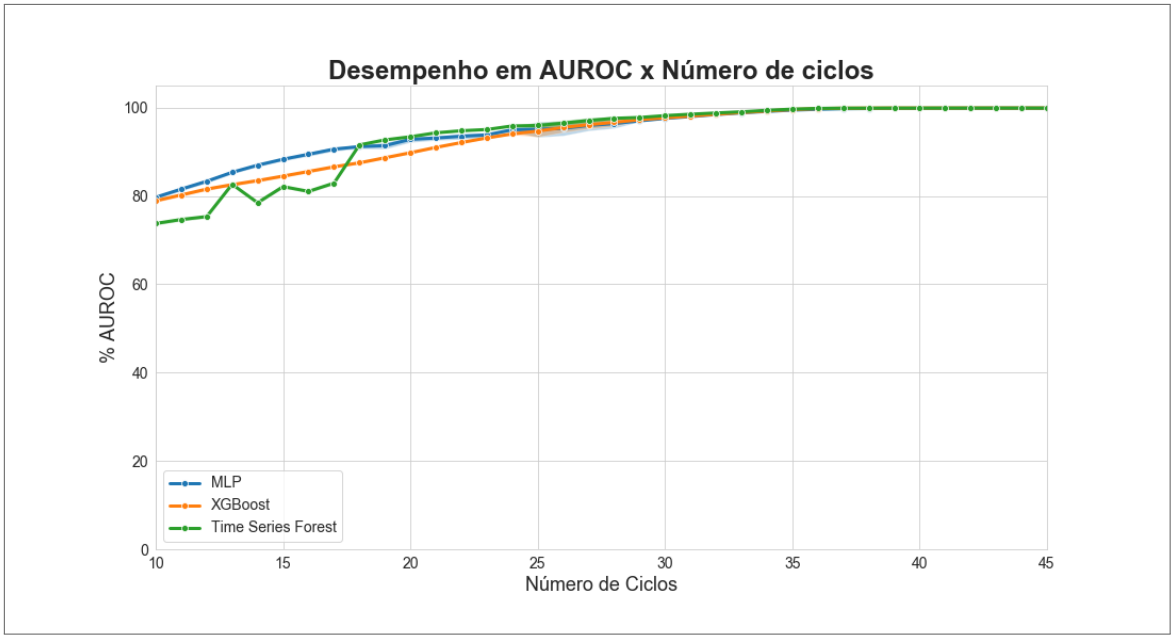
Tabela 10 – Desempenho dos algoritmos com hiperparâmetros padrões para 30 ciclos.

Modelo	AUROC	Acurácia	Especificidade	Sensibilidade
K-NN	96,63 ± 0,16	94,24 ± 0,19	99,00 ± 0,12	89,97 ± 0,34
NB	52,44 ± 7,97	54,84 ± 7,09	6,41 ± 18,99	98,37 ± 3,63
SVM	95,62 ± 0,19	93,29 ± 0,23	97,65 ± 0,34	89,38 ± 0,36
MLP	97,59 ± 0,13	94,33 ± 0,18	98,61 ± 0,30	90,48 ± 0,44
Árvore de Decisão	91,07 ± 0,23	91,09 ± 0,23	90,72 ± 0,36	91,43 ± 0,34
Gradient Boosting	97,56 ± 0,11	94,37 ± 0,18	98,67 ± 0,25	90,51 ± 0,38
XGBoost	97,72 ± 0,11	94,48 ± 0,18	98,85 ± 0,28	90,55 ± 0,42
TSF	98,22 ± 0,09	94,86 ± 0,17	98,80 ± 0,29	91,32 ± 0,39

Fonte: Elaborada pelo autor (2022)

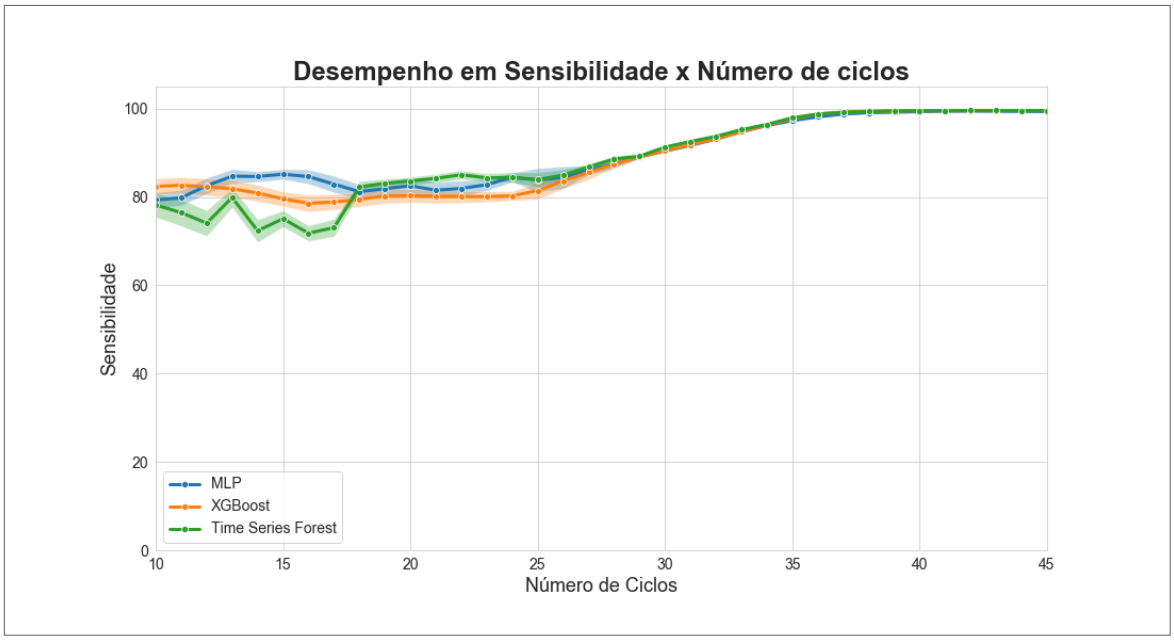
Por fim, sumarizando os resultados de variação de número de ciclos desde uma janela de 10 a 45, para a métrica AUROC, temos a Figura 20. Para essa análise mais completa, foram escolhidos os algoritmos MLP, XGBoost e *Time Series Forest*, que obtiveram desempenhos que se destacaram nas últimas etapas. Além da AUROC, na Figura 21 tem-se a mesma variação do número de ciclos, mas observando a sensibilidade, uma métrica que depende do ponto de corte. Nesse último caso, vemos uma maior variabilidade do desempenho, dado pelo desvio padrão, representado pelas margens de erro nos gráficos.

Figura 20 – Desempenho AUROC x Número de ciclos - MLP, XGBoost e TSF.



Fonte: Elaborada pelo autor (2022)

Figura 21 – Desempenho Sensibilidade x Número de ciclos - MLP, XGBoost e TSF.



Fonte: Elaborada pelo autor (2022)

5.7 OTIMIZAÇÃO DE HIPERPARÂMETROS

Diante dos cenários apresentados nas seções 5.5 e 5.6, em que se tem um panorama geral dos algoritmos elencados, parte-se agora para a otimização dos hiperparâmetros dos algoritmos que demonstraram maior desempenho nos experimentos iniciais, que são o *Time*

Series Forest, o MLP, e o XGBoost. Estes algoritmos caracterizam e sumarizam as diferentes características que apresentam potencial de desempenho nesse contexto. O primeiro, TSF, leva em consideração a ordem das características, exemplificando bem um algoritmo adaptado para as séries temporais. Em segundo, o MLP representa a classe de algoritmos de redes neurais que desempenha papel muito importante na área de AM, dando origem inclusive às redes neurais profundas. Finalmente, o XGBoost, extrai, junto com o TSF, as características do seu algoritmo base, árvore de decisão, e também o método de *boosting*, sendo um algoritmo de destaque quando se trata de dados tabulares.

Para a otimização dos hiperparâmetros, foi utilizado o *framework* Optuna. Diante da diversidade de características próprias de cada algoritmo, serão detalhadas a seguir, para cada caso, as características escolhidas, e o espaço de busca definido para que o Optuna encontre a melhor configuração.

5.7.1 Otimização MLP

A arquitetura é uma das questões mais discutidas quanto se trata de redes neurais artificiais. Não somente o número de camadas e de perceptrons em cada uma, mas a forma de ligação entre eles é importante. Por isso, esse foi o principal ponto a ser otimizado para o MLP, cujo espaço de busca definido está a seguir:

- **Número de camadas**: trial.suggest_int('n_layers', 1, 10);
- **Número de neurônios por camada**: for i in range(n_layers): layers.append(trial.suggest_int(f'n_units_i', 1, 100));
- **Otimizador**: trial.suggest_categorical('optimizer', ['adam', 'lbfgs', 'sgd']);
- **Função de ativação**: trial.suggest_categorical('activation', ['identity', 'logistic', 'tanh', 'relu'])

Para essa configuração de espaço de busca, obteve-se os seguintes valores: Otimizador sendo o Adam, função de ativação ReLu, quatro camadas, sendo os números de neurônios por camadas: 91, 38, 43 e 30, respectivamente. Em termos de configuração em código, $optimizer =' adam'$, $activation =' relu'$ e $hidden\_layer\_sizes = (91, 38, 43, 30)$.

5.7.2 Otimização XGBoost

No caso do algoritmo baseado em árvore de decisão, com a técnica de *boosting*, os hiperparâmetros considerados englobam tanto os mais básicos relacionados a um algoritmo de árvore de decisão, como outros relativos ao *booster* e de tarefa de aprendizagem. Dessa forma, foi definido o seguinte espaço de busca:

- **Quantidade de estimadores** : trial.suggest_int('n_estimators', 50, 2000);
- **Máxima profundidade**: trial.suggest_int('max_depth', 5, 50);
- **Regularização Alpha**: trial.suggest_int('reg_alpha', 0, 5);
- **Regularização Lambda**: trial.suggest_int('reg_lambda', 0, 5);
- **Peso mínimo das folhas**: trial.suggest_int('min_child_weight', 0, 5);
- **Mínimo ganho permitido para separação de nó**: trial.suggest_int('gamma', 0, 5);
- **Taxa de aprendizagem**: trial.suggest_loguniform('learning_rate', 0.005, 0.5);
- **Subamostragem de colunas**: trial.suggest_discrete_uniform('colsample_bytree', 0.1, 1, 0.01).

Ao final da busca, a configuração encontrada pelo Optuna foi: $n\_estimators = 972$, $max\_depth = 23$, $reg\_alpha = 5$, $reg\_lambda = 1$, $min\_child\_weight = 2$, $gamma = 0$, $learning\_rate = 0,0388586$, $colsample\_bytree = 0,72$.

5.7.3 Otimização *Time Series Forest*

Em geral os algoritmos baseados em árvore compartilham alguns hiperparâmetros, como número de estimadores, profundidade, número máximo de folhas, entre outros. Dessa forma, os hiperparâmetros e o respectivo espaço foram os seguintes:

- **Quantidade de estimadores**: $trial.suggest\_int('n\_estimators', 50, 2000)$;
- **Quantidade de janelas**: $trial.suggest\_uniform('n\_windows', 0.1, 1.0)$;
- **Tamanho mínimo da janela**: trial.suggest_uniform('min_window_size', 0.1, 1.0);

- **Máxima profundidade**: trial.suggest_int('max_depth', 1, 45);
- **Quantidade máxima de colunas**: trial.suggest_categorical('max_features', ['auto', 'sqrt', 'log2']);
- **Quantidade máxima de nós**: trial.suggest_int('max_leaf_nodes', 1, 10000).

Com a convergência da métrica de AUROC durante o processo de busca, chegou-se ao final com os hiperparâmetros: $n\_estimators = 343$, $n\_windows = 0,963452$, $max\_depth = 19$, $max\_features =' sqrt'$, $min\_windows\_size = 0,88153$, e $max\_leaf\_nodes = 5782$.

5.8 APRENDIZAGEM E DESEMPENHO DOS ALGORITMOS OTIMIZADOS

A curva de aprendizagem representa uma ótima ferramenta para monitorar o desempenho de algoritmos de aprendizagem supervisionada, assim como diagnosticar possíveis problemas. Esse monitoramento se dá pela construção de um gráfico, em que são coletados pontos de alguma métrica de desempenho, como a AUROC, para diferentes números de instâncias de treinamento. Dessa forma, se mede o impacto no aprendizado que se tem com o aumento da quantidade de instâncias.

A principal análise de diagnóstico com essa ferramenta, é a capacidade de aprendizagem do modelo, dinâmica que se caracteriza pelos extremos:

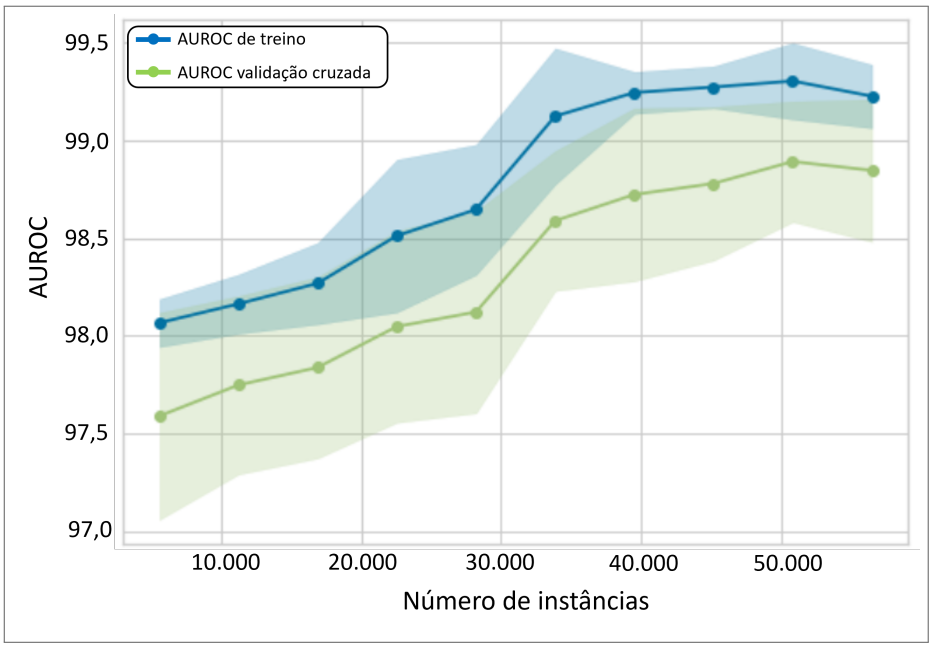
- Sobreajuste: É quando um modelo e seus hiperparâmetros estão muito ajustados para os dados específicos do conjunto de treinamento, de tal forma que o aprendizado não se generaliza para casos ainda não vistos. Um indicativo que esse problema possa estar ocorrendo, é ter um alto desempenho no conjunto de treino, mas apresentar uma queda drástica no conjunto de validação.
- Sub-ajuste: Situação contrária, em que o modelo não é complexo o suficiente para acompanhar as características necessárias para predizer um novo caso. Nesse caso, tanto o desempenho de treino como de validação são baixos.

Nesse contexto, nas figuras 22, 23 e 24 são apresentados os gráficos do processo de aprendizagem, com a dinâmica da AUROC para diferentes tamanhos de conjunto de treino, para os três algoritmos elencados para essa etapa. Vale ressaltar que aqui foram utilizadas somente as amostras humanas tanto para treinamento como para validação. Para se obter um

resultado mais consistente para cada número de instâncias de treinamento, foi considerada uma estratégia de validação cruzada denominada k-*folds*, com dez partes. Com isso, realiza-se dez vezes a etapa de treinamento e validação em um conjunto separado, em que se pode projetar margens de erro e valores médios, como são apresentados nos gráficos.

No primeiro caso, da Figura 22, tem-se o gráfico para o algoritmo MLP. Aqui se acompanha uma contínua variação do desempenho AUROC tanto nos conjuntos de treino, como de validação. É interessante notar que os desempenhos dos conjuntos são bem próximos, se sobrepondo nas margens de erro.

Figura 22 – Curva de Aprendizagem por meio da AUROC x Número de instâncias - Algoritmo MLP Otimizado com Optuna.



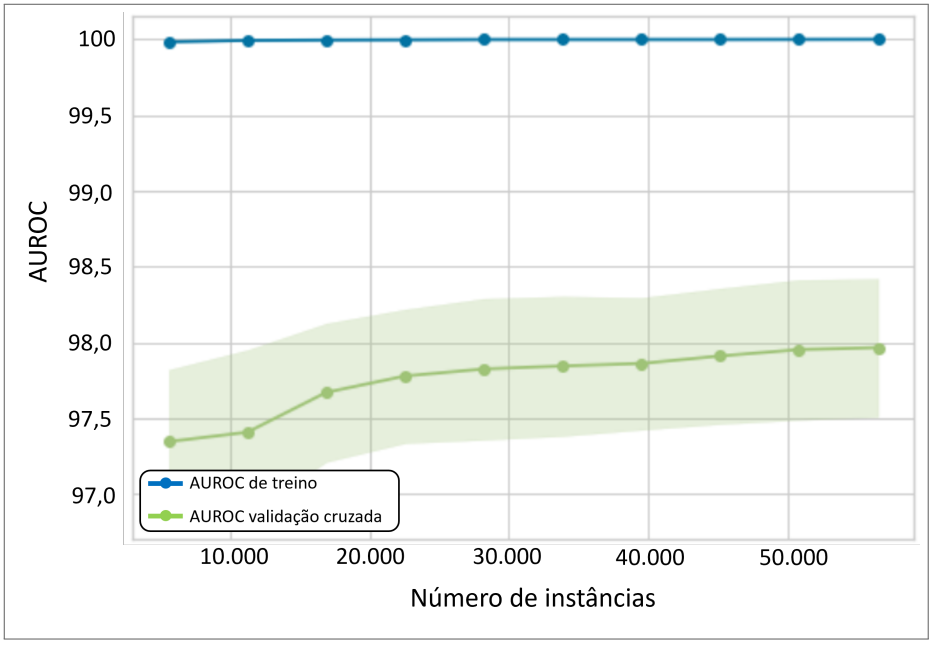
Fonte: Elaborada pelo autor (2022)

Na Figura 23, apresentando a aprendizagem do XGBoost otimizado com Optuna, observa-se uma maior diferença entre as métricas de treinamento e validação. Apesar de aparentemente alta, tem-se ao final uma diferença de 0,02 de AUROC, representando aproximadamente 2%.

Por fim, na Figura 24, observa-se uma melhor aproximação do ponto de estabilidade da AUROC de validação, chegando aos 99 de AUROC, considerando a margem de erro.

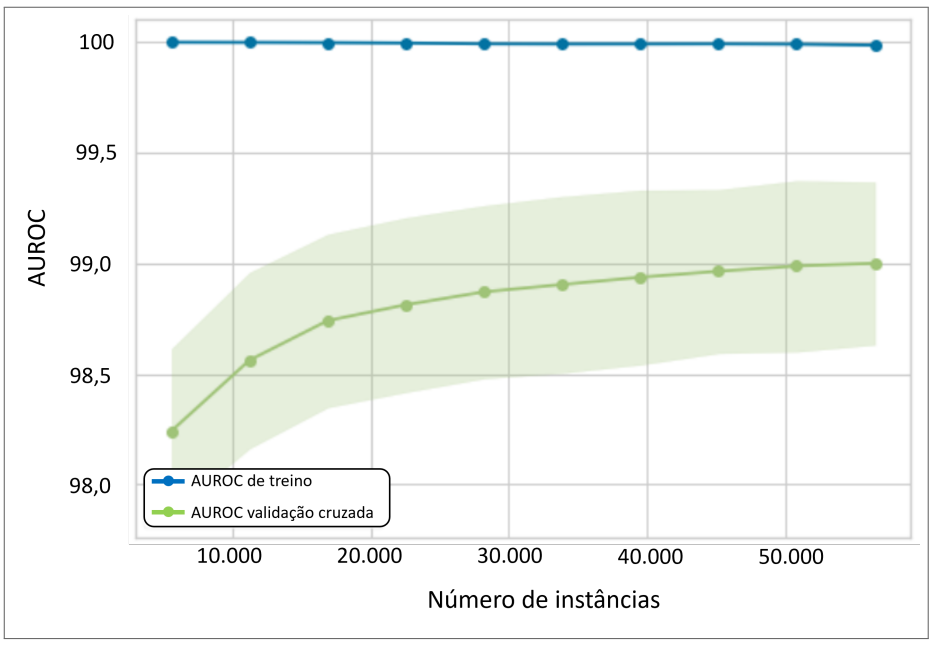
Além de acompanhar o processo de aprendizagem, replicou-se o estudo de desempenho feito anteriormente nas seções 5.5 e 5.6. Ou seja, agora os algoritmos com hiperparâmetros otimizados foram treinados com reamostragens de cem vezes. Os resultados finais, de média e desvio padrão, sumarizando essas rodadas de treinamento, encontram-se na Tabela 12. Retomando os resultados apresentados anteriormente anteriormente para 30 ciclos e hiperpa-

Figura 23 – Curva de Aprendizagem por meio da AUROC x Número de instâncias - Algoritmo XGBoost Otimizado com Optuna.



Fonte: Elaborada pelo autor (2022)

Figura 24 – Curva de Aprendizagem por meio da AUROC x Número de instâncias - Algoritmo TSF Otimizado com Optuna.



Fonte: Elaborada pelo autor (2022)

râmetros padrões, Tabela 11.

Tabela 11 – Desempenho dos algoritmos com hiperparâmetros padrões para 30 ciclos.

Modelo	AUROC	Acurácia	Especificidade	Sensibilidade
K-NN	96,63 ± 0,16	94,24 ± 0,19	99,00 ± 0,12	89,97 ± 0,34
NB	52,44 ± 7,97	54,84 ± 7,09	6,41 ± 18,99	98,37 ± 3,63
SVM	95,62 ± 0,19	93,29 ± 0,23	97,65 ± 0,34	89,38 ± 0,36
MLP	97,59 ± 0,13	94,33 ± 0,18	98,61 ± 0,30	90,48 ± 0,44
Árvore de Decisão	91,07 ± 0,23	91,09 ± 0,23	90,72 ± 0,36	91,43 ± 0,34
Gradient Boosting	97,56 ± 0,11	94,37 ± 0,18	98,67 ± 0,25	90,51 ± 0,38
XGBoost	97,72 ± 0,11	94,48 ± 0,18	98,85 ± 0,28	90,55 ± 0,42
TSF	98,22 ± 0,09	94,86 ± 0,17	98,80 ± 0,29	91,32 ± 0,39

Fonte: Elaborada pelo autor (2022)

Tabela 12 – Desempenho dos algoritmos com otimização do Optuna para 30 ciclos.

Modelo	AUROC	Acurácia	Especificidade	Sensibilidade
MLP	98,94 ± 0,19	95,14 ± 0,34	97,57 ± 0,76	92,97 ± 1,16
XGBoost	97,99 ± 0,10	94,66 ± 0,17	98,78 ± 0,29	90,97 ± 0,42
TSF	98,98 ± 0,07	95,46 ± 0,16	98,20 ± 0,42	93,00 ± 0,52

Fonte: Elaborada pelo autor (2022)

5.9 COMPARAÇÃO DE DESEMPENHO POR TESTES DE HIPÓTESE

O conceito de aprendizagem em IA envolve extrapolar o desempenho de um algoritmo para dados ainda não vistos. Com o treinamento de diversos classificadores em repetidas amostragens, pode-se chegar a casos espúrios de desempenho, que não tem rigor estatístico. Uma boa comparação de desempenho de classificadores, requer um teste estatístico adequado (BOUCKAERT; FRANK, 2004), que elimina esses casos espúrios, a partir de suposições sobre as amostras de métricas, e estabelecido um intervalo de confiança.

Uma prática comum de comparação de desempenho de classificadores em uma base de dados, é por meio da validação cruzada k-*fold*, juntamente com a utilização do teste t de Student pareado. No entanto, para esse teste estatístico é necessário assumir que a distribuição das amostras segue uma normal, e além disso, que elas sejam independentes. No método k-*fold*, é feito uma divisão das amostras em k partes iguais mutuamente exclusivas, e consequentemente, o treinamento realizado em um caso terá uma dependência dos outros (caso $k > 2$). Dessa forma, será utilizada aqui uma divisão aleatória da base em treino e teste (70% e 30%, respectivamente) com geração de 100 reamostragens, para se ter uma boa quantidade

de amostras para o teste estatístico. Outra questão considerada foi dispensar a necessidade de distribuição normal, e utilizar então um teste não-paramétrico, o teste de sinais com postos de Wilcoxon (DEMSAR, 2006). Definido o teste estatístico, as hipóteses consideradas estão no seguinte formato:

$$\begin{cases} H_0 : \text{O desempenho dos classificadores é equivalente.} \\ H_1 : \text{O desempenho do classificador 1 é maior do que o do classificador 2.} \end{cases} \tag{5.4}$$

Para avaliar as hipóteses, em cada treinamento realizado, é obtido o desempenho nas métricas consideradas, para os 100 casos de repetição. Logo, cada valor de métrica obtida representa uma amostra que será utilizada para o teste estatístico. O teste retorna o valor-p, que será comparado com o nível de significância $\alpha = 0,95$. Caso $p \leq 1 - \alpha$, é rejeitada a hipótese nula, e para $p > 1 - \alpha$, falha-se em rejeitá-la, e aceita-se a hipótese alternativa H_1.

A partir dos diversos cenário apresentados anteriormente, nas tabelas de resultados, foram elaborados os seguintes questionamentos para serem avaliados pelos testes de hipótese:

- **Algoritmos otimizados com Optuna para 30 ciclos:** O TSF teve desempenho em AUROC melhor do que o MLP?
- **Algoritmos otimizados com Optuna para 30 ciclos:** O TSF teve desempenho em AUROC melhor do que o XGBoost?
- **Utilização do Optuna para 30 ciclos:** O TSF otimizado com Optuna tem desempenho em AUROC melhor do que sua versão com valores padrões de hiperparâmetros?
- **Utilização do Optuna, comparando-se 30 com 35 ciclos:** O TSF com valores padrões de hiperparâmetro com 35 ciclos, tem desempenho em AUROC melhor do que o TSF otimizado com 30 ciclos?

Os resultados dos testes, representados pelos valores-p, são apresentados na Tabela 13. Para o nível de significância considerado, $\alpha = 0,95$, é possível rejeitar somente a primeira hipótese, em que o TSF é superior em AUROC ao MLP, no cenário de 30 ciclos com Optuna. Ou seja, não se foi possível identificar diferença estatística entre seus desempenhos. Para a segunda hipótese, conclui-se que o TSF possui desempenho em AUROC maior que o XGBoost, para as otimizações realizadas. No cenário de comparação entre o algoritmo TSF otimizado com Optuna e sua versão com valores de hiper-parâmetro padrão, se vê uma melhoria no

desempenho, como já se era esperado devido ao aumento de AUROC. No entanto, no comparativo de 35 contra 30 ciclos, superou o algoritmo treinado com mais informações temporais, mesmo que utilizando de valores de hiper-parâmetro padrões.

Tabela 13 – P-valor dos testes estatísticos propostos.

Classificador 1	**Classificador 2**	**p-valor**
TSF 30 ciclos com Optuna	MLP 30 ciclos com Optuna	0,143239
TSF 30 ciclos com Optuna	XGBoost 30 ciclos com Optuna	1,95E-18
TSF 30 ciclos com Optuna	TSF 30 ciclos padrão	1,95E-18
TSF 35 ciclos padrão	TSF 30 ciclos com Optuna	1,95E-18

Fonte: Elaborada pelo autor (2022)

6 CONSIDERAÇÕES FINAIS

Em meio a necessidade de diferentes formas de combate à pandemia do COVID-19, o diagnóstico se mostra como um importante pilar. A partir dos arquivos em formato .eds disponibilizados diretamente pela máquina que realiza o exame RT-qPCR, foi construída a visão de modelagem própria para ser utilizada para os algoritmos de AM. Nisso, tem-se uma tabela em que cada linha possui uma amostra com seus respectivos valores de fluorescência para cada ciclo de máquina, a ser classificada como positiva ou negativa.

O laboratório NUPIT da UFPE, parceiro responsável pela disponibilização dos arquivos do exame, preconiza determinados protocolos do RT-qPCR para COVID-19, que filtram casos de contaminação e outros que necessitam repetição. Após aplicados esses filtros, chegou-se a uma base de dados de 62.676 amostras humanas, e 13.040 de controle negativo. A taxa de casos positivos é de 52% e 25%, respectivamente, sendo portanto boa representatividade para balanceamento das classes. Para efeito de medição do desempenho, foram analisados somente as amostras ditas como humanas, que tem como alvo os genes $N1$ e $N2$, retirando-se as de controle, RP.

O problema inicial de classificação a partir da totalidade dos ciclos disponíveis no exame de RT-qPCR se mostrou relativamente simples. Isso porque obteve-se um desempenho bastante similar dentre os algoritmos elencados e com exceção do NB, todos acima de 99% de AUROC. Foram elencados tanto algoritmos tidos como genéricos para problemas de classificação, como outros que consideram a ordenação das características. Comparando-se algoritmos com diversidade de características, pôde-se ter uma ideia melhor acerca da complexidade do problema.

No intuito de antecipar a predição do exame, tornando-o mais rápido para disponibilização do diagnóstico, foi feito o treinamento dos algoritmos com um número de ciclos menor que 45. Com o uso de cada vez menos características nos modelos preditivos, o desempenho observado foi diminuindo, sendo necessário realizar uma otimização dos hiperparâmetros para que o algoritmo se adaptasse melhor ao contexto dos dados disponíveis. Essa última etapa de otimização foi realizada somente para os três melhores algoritmos que se destacaram na predição em seus valores de hiperparâmetro padrões, nos casos analisados de 30, 35, 40 e 45 ciclos: o MLP, o XGBoost e o *Time Series Forest*.

Segundo (BUSTIN; NOLAN, 2020), o tempo médio de realização de um exame de RT-qPCR

para o SARS-CoV-2 gira em torno de 1 hora e 38 minutos, para completar 45 ciclos. Tendo esse valor como referência, a redução de 45 para 30 ciclos, com o suporte do algoritmo, resultaria em 1 hora e 5 minutos de processamento do exame, reduzindo em 33 minutos. Para o cenário de redução para 18 ciclos, em que o TSF treinado com valores padrões de hiper-parâmetros ainda possui AUROC maior que 90, o tempo total iria para 39 minutos, reduzindo para 40% o tempo total, sendo quase 1 hora de ganho no experimento.

Para a comparação dos resultados dos algoritmos, é de grande importância na literatura médica a utilização de métricas em conjunto que são conflitantes entre si, como sensibilidade e especificidade. Estas são calculadas a partir da atribuição do algoritmo como sendo positivo ou negativo. Como os algoritmos utilizados tem como saída uma probabilidade, é necessário especificar um ponto de corte para a binarização do resultado. Foram apresentados algumas formas diferentes de se decidir o valor ótimo de ponto de corte, e escolhido aquele que maximiza o valor da soma entre sensibilidade e especificidade, denominado índice de Youden. Além dessas métricas, é importante apresentar um desempenho que independe do ponto de corte, como é o caso da AUROC, para se ter também um resultado sem esse viés, e comparação mais direta do desempenho dos algoritmos.

Por meio dos testes estatísticos elaborados, foi possível realizar as seguintes conclusões a respeito da otimização dos algoritmos, assim como da redução do número de ciclos:

- Com a otimização realizada pelo Optuna, foi observado que tanto o TSF como o MLP tiveram desempenhos considerados semelhantes para trinta ciclos. Pôde ser visto também que o XGBoost teve desempenho abaixo desses dois, apesar de apresentar um valor de AUROC próximo;

- Observando o caso de 30 ciclos para o TSF, foi vista uma melhoria de desempenho utilizando-se a otimização dos hiperparâmetros, em comparação aos valores padrões.

- A otimização do TSF mostrou-se efetiva comparando os casos quando se usa a mesma quantidade de ciclos. No entanto, a versão otimizada com 30 ciclos não superou o desempenho da versão com 35 ciclos que utiliza valores padrões de hiperparâmetros. Isso demonstra a relevância dessa faixa de ciclos para o treinamento do algoritmo, especificamente para esse contexto de base de dados de RT-qPCR para COVID-19.

Esses resultados estatísticos e de desempenho são respaldados primeiramente pela grande quantidade de amostras, que totalizam 75.716, sendo 15.646 separadas para teste, e também

pelas 100 reamostragens da base de dados realizadas, permutando as amostras de treino e teste. Essa abordagem mais robusta difere-se do apresentado em (LEE et al., 2022), que apesar de realizar uma abordagem de antecipação da classificação do RT-qPCR pelos dados de fluorescência, não dispunha de boa representatividade de amostras, principalmente de casos positivos. Outra desvantagem é a não realização de testes estatísticos, e a utilização de uma base de teste fixa, podendo implicar em problemas de superajuste.

Devido as bases de dados serem diferentes, com provável diferença também em protocolos adotados, a comparação direta de desempenho dos trabalhos não é a ideal. Não foi especificado em (LEE et al., 2022), por exemplo, quais genes são utilizados como referência para detecção no exame, e se são incluídas as amostras de controle nos resultados apresentados. Apesar dessas considerações, os algoritmos TSF e MLP otimizados, que apresentaram melhor desempenho nos 30 ciclos, tiveram uma AUROC maior do que todos os modelos considerados no trabalho similar encontrado, mesmo com a versão treinada com 36 ciclos. Caso seja adotado o valor de referência de 89% como mínimo para sensibilidade e especificidade informado pela meta-análise de (KIM; HONG; YOON, 2020), então a antecipação para 30 ciclos poderia se tornar viável sob esses critérios, podendo então serem realizadas novas pesquisas para avanço nessa frente.

É importante destacar as limitações inerentes aos algoritmos de aprendizagem de máquina. Apesar de ser observado um decréscimo de apenas 1 ponto percentual na AUROC para uma antecipação de 33% no tempo de máquina do RT-qPCR, não se tem garantia sobre a continuidade desse desempenho em outros contextos, como diferença nos protocolos do exame, diferente taxa de positividade, mutação do vírus, entre outros cenários. Identificando a possível diferença de contexto, é necessário a criação de uma nova base de teste, com a marcação do alvo de amplificação, e o teste do algoritmo nesse novo cenário. A depender dos resultados observados, pode ser necessário um retreinamento do algoritmo, incluindo esse novo conjunto de dados.

Além dos resultados apresentados particularmente para o caso de detecção do SARS-CoV-2, com esse trabalho deixa-se a abertura e o incentivo para que seja aplicado em outras frentes que utilizem do mesmo método de detecção de gene. Para isso, é necessária que outras pesquisas desenvolvam bases de dados dos genes específicos a serem detectados. Cabe investigar, para cada caso, se a antecipação dos ciclos se faz de uma maneira mais fácil ou não, dado que a configuração experimental pode impactar bastante no resultado da amplificação da fluorescência. Com essa visão, vê-se a importância dessa pesquisa pela possibilidade de replicação

e escalabilidade da metodologia apresentada aqui, de aplicar algoritmos de aprendizagem de máquina para antecipação de exames RT-qPCR.

O desenvolvimento e disponibilização de base de dados no contexto de aprendizagem de máquina, é uma etapa crucial para que a comunidade científica possa avançar no tema. Graças à contribuição deste trabalho, será possível ter referência de desempenho quanto à antecipação do número de ciclos no RT-qPCR para SARS-CoV-2, e ter como parâmetro possíveis melhorias de otimização, utilização de novos algoritmos, entre outras estratégias.

6.1 TRABALHOS FUTUROS

Com a experiência obtida no decorrer do desenvolvimento desse trabalho, foram consideradas diferentes possibilidades de continuação da pesquisa, listadas a seguir:

- Devido a dinamicidade da área de de aprendizagem de máquina, em que constantemente são propostos novos algoritmos, os resultados apresentados podem ser constantemente desafiados por novos algoritmos implementados;
- Com os resultados favoráveis dessa pesquisa, que indica o benefício de redução de tempo sem muita diminuição do desempenho, sugere-se avançar nas pesquisas para adoção do modelo desenvolvido e aplicação da redução do tempo de processamento do exame, avaliando os protocolos já existentes. Concomitantemente, deve-se ser feita uma análise mais minuciosa a respeito dos casos de falsos positivos, e dos que foram considerados inconclusivos, de forma a não se realizar novamente o exame, o que dobraria o tempo de realização;
- Da mesma forma que realizado em (CORDARO et al., 2021), é interessante propor uma estratégia de otimização de parâmetros da configuração do experimento, de forma a aumentar a taxa de sucesso do exame de qPCR, diferentemente da proposta do artigo, em que se restringe ao método convencional. Esse é um trabalho que necessita ser realizado com muita interação com pesquisadores de um laboratório onde é realizado o exame. Com o desenvolvimento desse novo estudo, pode-se beneficiar tanto desse aumento na taxa de sucesso do exame, como também do proposto neste trabalho, de antecipar a classificação do exame;

- Uma alternativa que pode melhorar o desempenho do modelo, seria a combinação dos dados de fluorescência com dados clínicos, que são de fácil obtenção. Dessa forma, primeiramente se faz necessária a construção de uma base de dados que agregue esses dados, para que se possa analisar a melhoria de desempenho com a adição dessas novas variáveis no modelo preditivo.

- Considerando que o exame de RT-PCR também é utilizado para detecção de outros genes e doenças, fica o indício da utilização da mesma abordagem adotada aqui, para antecipação de resultados. Para isso, no entanto, é necessário o desenvolvimento de novas bases de dados para os contextos específicos. Com isso, se terá mais informação sobre quais as condições mais propícias que indicariam uma possibilidade maior de redução do número de ciclos.

REFERÊNCIAS

AGRAWAL, T. *Hyperparameter optimization in machine learning : make your machine learning and deep learning models more efficient*. Berkeley: Apress, 2021. ISBN 1484265793.

AKIBA, T.; SANO, S.; YANASE, T.; OHTA, T.; KOYAMA, M. Optuna: A next-generation hyperparameter optimization framework. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. [S.l.]: ACM, 2019.

ALFEILAT, H. A. A.; HASSANAT, A. B.; LASASSMEH, O.; TARAWNEH, A. S.; ALHASANAT, M. B.; SALMAN, H. S. E.; PRASATH, V. S. Effects of distance measure choice on k-nearest neighbor classifier performance: A review. *Big Data*, Mary Ann Liebert Inc, v. 7, n. 4, p. 221–248, dec 2019.

ALIMADADI, A.; ARYAL, S.; MANANDHAR, I.; MUNROE, P. B.; JOE, B.; CHENG, X. Artificial intelligence and machine learning to fight COVID-19. *Physiological Genomics*, American Physiological Society, v. 52, n. 4, p. 200–202, apr 2020.

ALJAME, M.; AHMAD, I.; IMTIAZ, A.; MOHAMMED, A. Ensemble learning model for diagnosing COVID-19 from routine blood tests. *Informatics in Medicine Unlocked*, Elsevier BV, v. 21, p. 100449, 2020.

ALJAME, M.; IMTIAZ, A.; AHMAD, I.; MOHAMMED, A. Deep forest model for diagnosing COVID-19 from routine blood tests. *Scientific Reports*, Springer Science and Business Media LLC, v. 11, n. 1, aug 2021.

ALOUANI, D. J.; RAJAPAKSHA, R. R. P.; JANI, M.; RHOADS, D. D.; SADRI, N. Specificity of sars-cov-2 real-time pcr improved by deep learning analysis. *Journal of clinical microbiology*, v. 59, maio 2021. ISSN 1098-660X.

BAGNALL, A.; LINES, J.; BOSTROM, A.; LARGE, J.; KEOGH, E. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, Springer Science and Business Media LLC, v. 31, n. 3, p. 606–660, nov 2016.

BERGSTRA, J.; BARDENET, R.; BENGIO, Y.; KéGL, B. Algorithms for hyper-parameter optimization. In: SHAWE-TAYLOR, J.; ZEMEL, R.; BARTLETT, P.; PEREIRA, F.; WEINBERGER, K. Q. (Ed.). *Advances in Neural Information Processing Systems*. [S.l.]: Curran Associates, Inc., 2011. v. 24.

BERGSTRA, J.; BENGIO, Y. Random search for hyper-parameter optimization. *Journal of Machine Learning Research*, v. 13, n. 10, p. 281–305, 2012.

BERGSTRA, J.; KOMER, B.; ELIASMITH, C.; YAMINS, D.; COX, D. D. Hyperopt: a python library for model selection and hyperparameter optimization. *Computational Science & Discovery*, IOP Publishing, v. 8, n. 1, p. 014008, jul 2015.

BERNDT, D. J.; CLIFFORD, J. Using dynamic time warping to find patterns in time series. In: *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*. [S.l.]: AAAI Press, 1994. (AAAIWS'94), p. 359–370.

BISHOP, C. M. *Pattern Recognition and Machine Learning*. [S.l.]: Springer-Verlag New York Inc., 2011. ISBN 0387310738.

BOSER, B. E.; GUYON, I. M.; VAPNIK, V. N. A training algorithm for optimal margin classifiers. In: *Proceedings of the fifth annual workshop on Computational learning theory - COLT '92*. [S.l.]: ACM Press, 1992.

BOUCKAERT, R. R.; FRANK, E. Evaluating the replicability of significance tests for comparing learning algorithms. In: *Advances in Knowledge Discovery and Data Mining*. [S.l.]: Springer Berlin Heidelberg, 2004. p. 3–12.

BRADLEY, A. P. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, Elsevier BV, v. 30, n. 7, p. 1145–1159, jul 1997.

BRAGA. *Redes neurais artificiais : teoria e aplicações*. Rio de Janeiro: LTC Editora, 2007. ISBN 9788521615644.

BREIMAN, L. Bagging predictors. *Machine Learning*, Springer Science and Business Media LLC, v. 24, n. 2, p. 123–140, aug 1996.

BREIMAN, L. Prediction games and arcing algorithms. *Neural Computation*, MIT Press - Journals, v. 11, n. 7, p. 1493–1517, oct 1999.

BREIMAN, L. Random forests. *Machine Learning*, Springer Science and Business Media LLC, v. 45, n. 1, p. 5–32, 2001.

BREIMAN, L.; FRIEDMAN, J.; STONE, C. J.; OLSHEN, R. *Classification and regression trees*. New York: Chapman & Hall, 1984. ISBN 9780412048418.

BRINATI, D.; CAMPAGNER, A.; FERRARI, D.; LOCATELLI, M.; BANFI, G.; CABITZA, F. Detection of COVID-19 infection from routine blood exams with machine learning: A feasibility study. *Journal of Medical Systems*, Springer Science and Business Media LLC, v. 44, n. 8, jul 2020.

BUSTIN, S. A.; NOLAN, T. RT-qPCR testing of SARS-CoV-2: A primer. *International Journal of Molecular Sciences*, MDPI AG, v. 21, n. 8, p. 3004, apr 2020.

BUTANTAN, I. *Entenda o que é uma pandemia e as diferenças entre surto, epidemia e endemia*. 2021. Disponível em: <https://butantan.gov.br/covid/butantan-tira-duvida/tira-duvida-noticias/entenda-o-que-e-uma-pandemia-e-as-diferencas-entre-surto-epidemia-e-endemia>.

CHEN, T.; GUESTRIN, C. Xgboost: A scalable tree boosting system. arXiv, 2016.

CHENG, M. P.; PAPENBURG, J.; DESJARDINS, M.; KANJILAL, S.; QUACH, C.; LIBMAN, M.; DITTRICH, S.; YANSOUNI, C. P. Diagnostic testing for severe acute respiratory syndrome–related coronavirus 2. *Annals of Internal Medicine*, American College of Physicians, v. 172, n. 11, p. 726–734, jun 2020.

CORDARO, N. J.; KAVRAN, A. J.; SMALLEGAN, M.; PALACIO, M.; LAMMER, N.; BRANT, T. S.; DUMONT, V.; GARCIA, N. D.; MILLER, S.; JOURABCHI, T.; SAWYER, S. L.; CLAUSET, A. Optimizing polymerase chain reaction (PCR) using machine learning. Cold Spring Harbor Laboratory, aug 2021.

COVER, T.; HART, P. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, Institute of Electrical and Electronics Engineers (IEEE), v. 13, n. 1, p. 21–27, jan 1967.

CYBENKO, G. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems*, Springer Science and Business Media LLC, v. 2, n. 4, p. 303–314, dec 1989.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, JMLR.org, v. 7, p. 1–30, dec 2006. ISSN 1532-4435.

DENG, H.; RUNGER, G.; TUV, E.; VLADIMIR, M. A time series forest for classification and feature extraction. *Information Sciences*, Elsevier BV, v. 239, p. 142–153, aug 2013.

DIETTERICH, T. G. Ensemble methods in machine learning. In: *Multiple Classifier Systems*. [S.l.]: Springer Berlin Heidelberg, 2000. p. 1–15.

DUDA, R. O.; HART, P. E.; STORK, D. G. *Pattern Classification*. [S.l.]: Wiley John + Sons, 2000. ISBN 0471056693.

EGGENSPERGER, K. Towards an empirical foundation for assessing bayesian optimization of hyperparameters. In: *NIPS workshop on Bayesian Optimization in Theory and Practice*. [S.l.: s.n.], 2013. v. 10, n. 3.

EGGENSPERGER, K.; HUTTER, F.; HOOS, H. H.; LEYTON-BROWN, K. Efficient benchmarking of hyperparameter optimizers via surrogates. In: *AAAI*. [S.l.: s.n.], 2015.

ELSHAWI, R.; MAHER, M.; SAKR, S. Automated machine learning: State-of-the-art and open challenges. jun. 2019.

ERUHIMOV, V.; MARTYANOV, V.; TUV, E. Constructing high dimensional feature space for time series classification. In: *Knowledge Discovery in Databases: PKDD 2007*. [S.l.]: Springer Berlin Heidelberg, 2007. p. 414–421.

FAOUZI, J.; JANATI, H. pyts: A python package for time series classification. *J. Mach. Learn. Res.*, v. 21, p. 46–1, 2020.

FIX, E.; HODGES, J. *Discriminatory Analysis: Nonparametric Discrimination: Consistency Properties*. [S.l.]: USAF School of Aviation Medicine, 1951.

FREIRE-PASPUEL, B.; GARCIA-BEREGUIAIN, M. A. Analytical sensitivity and clinical performance of a triplex RT-qPCR assay using CDC n1, n2, and RP targets for SARS-CoV-2 diagnosis. *International Journal of Infectious Diseases*, Elsevier BV, v. 102, p. 14–16, jan 2021.

FREUND, Y.; SCHAPIRE, R. E. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, Elsevier BV, v. 55, n. 1, p. 119–139, aug 1997.

FREUND, Y.; SCHAPIRE, R. E. et al. Experiments with a new boosting algorithm. In: CITESEER. *icml*. [S.l.], 1996. v. 96, p. 148–156.

FRIEDMAN, J.; HASTIE, T.; TIBSHIRANI, R. Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors). *The Annals of Statistics*, Institute of Mathematical Statistics, v. 28, n. 2, apr 2000.

FRIEDMAN, J. H. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, Institute of Mathematical Statistics, v. 29, n. 5, p. 1189–1232, 2001.

FRIEDMAN, J. H. Stochastic gradient boosting. *Computational Statistics & Data Analysis*, Elsevier BV, v. 38, n. 4, p. 367–378, feb 2002.

GHEBREYESUS, T. A. Who director-general's opening remarks at the media briefing on covid-19. 2020. *Geneva: World Health Organization*, 2020.

GOLOVIN, D.; SOLNIK, B.; MOITRA, S.; KOCHANSKI, G.; KARRO, J.; SCULLEY, D. Google vizier. In: *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. [S.l.]: ACM, 2017.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016.

GOODMAN-MEZA, D.; RUDAS, A.; CHIANG, J. N.; ADAMSON, P. C.; EBINGER, J.; SUN, N.; BOTTING, P.; FULCHER, J. A.; SAAB, F. G.; BROOK, R.; ESKIN, E.; AN, U.; KORDI, M.; JEW, B.; BALLIU, B.; CHEN, Z.; HILL, B. L.; RAHMANI, E.; HALPERIN, E.; MANUEL, V. A machine learning algorithm to increase COVID-19 inpatient diagnostic capacity. *PLOS ONE*, Public Library of Science (PLoS), v. 15, n. 9, p. e0239474, sep 2020.

GOZES, O.; FRID-ADAR, M.; GREENSPAN, H.; BROWNING, P. D.; ZHANG, H.; JI, W.; BERNHEIM, A.; SIEGEL, E. Rapid ai development cycle for the coronavirus (covid-19) pandemic: Initial results for automated detection & patient monitoring using deep learning ct image analysis. mar. 2020.

GUNAY, M.; GOCERI, E.; BALASUBRAMANIYAN, R. Machine learning for optimum CT-prediction for qPCR. In: *2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA)*. [S.l.]: IEEE, 2016.

GUPTA, A.; GUPTA, H. P.; BISWAS, B.; DUTTA, T. Approaches and applications of early classification of time series: A review. *IEEE Transactions on Artificial Intelligence*, Institute of Electrical and Electronics Engineers (IEEE), v. 1, n. 1, p. 47–61, aug 2020.

HABIBZADEH, F.; HABIBZADEH, P.; YADOLLAHIE, M. On determining the most appropriate test cut-off value: the case of tests with continuous results. *Biochemia Medica*, Croatian Society for Medical Biochemistry and Laboratory Medicine, p. 297–307, 2016.

HAJIAN-TILAKI, K. Receiver operating characteristic (roc) curve analysis for medical diagnostic test evaluation. *Caspian journal of internal medicine*, v. 4, p. 627–635, 2013. ISSN 2008-6164.

HANSEN, L.; SALAMON, P. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Institute of Electrical and Electronics Engineers (IEEE), v. 12, n. 10, p. 993–1001, 1990.

HARRISON, A. G.; LIN, T.; WANG, P. Mechanisms of SARS-CoV-2 transmission and pathogenesis. *Trends in Immunology*, Elsevier BV, v. 41, n. 12, p. 1100–1115, dec 2020.

HASTIE, T. *The elements of statistical learning : data mining, inference, and prediction*. New York: Springer, 2009. ISBN 9780387848846.

HAYKIN, S. *Redes neurais : Princípios e prática*. Porto Alegre: Bookman, 2001. ISBN 8573077182.

HO, T. K. Random decision forests. In: *Proceedings of 3rd International Conference on Document Analysis and Recognition*. [S.l.]: IEEE Comput. Soc. Press, 1995. v. 1, p. 278–282 vol.1.

HOLLAND, J. *Adaptation in natural and artificial systems : an introductory analysis with applications to biology, control, and artificial intelligence*. Cambridge, Mass: MIT Press, 1992. ISBN 9780262275552.

HUTTER, F. *Automated machine learning : methods, systems, challenges*. Cham, Switzerland: Springer, 2019. ISBN 9783030053185.

HUTTER, F.; HOOS, H. H.; LEYTON-BROWN, K. Sequential model-based optimization for general algorithm configuration. In: *Lecture Notes in Computer Science*. [S.l.]: Springer Berlin Heidelberg, 2011. p. 507–523.

INDRAYAN, A. *Medical biostatistics*. Boca Raton: Chapman & Hall/CRC, 2017. ISBN 9781498799539.

KIM, H.; HONG, H.; YOON, S. H. Diagnostic performance of CT and reverse transcriptase polymerase chain reaction for coronavirus disease 2019: A meta-analysis. *Radiology*, Radiological Society of North America (RSNA), v. 296, n. 3, p. E145–E155, sep 2020.

KOCH, P.; GOLOVIDOV, O.; GARDNER, S.; WUJEK, B.; GRIFFIN, J.; XU, Y. Autotune: A derivative-free optimization framework for hyperparameter tuning. abr. 2018.

KRIZHEVSKY, A.; SUTSKEVER, I.; HINTON, G. E. ImageNet classification with deep convolutional neural networks. *Communications of the ACM*, Association for Computing Machinery (ACM), v. 60, n. 6, p. 84–90, may 2017.

KUBISTA, M.; ANDRADE, J. M.; BENGTSSON, M.; FOROOTAN, A.; JONÁK, J.; LIND, K.; SINDELKA, R.; SJöBACK, R.; SJöGREEN, B.; STRöMBOM, L.; STÅHLBERG, A.; ZORIC, N. The real-time polymerase chain reaction. *Molecular Aspects of Medicine*, Elsevier BV, v. 27, n. 2-3, p. 95–125, apr 2006.

KUPERVASSER, O. The mysterious optimality of naive bayes: Estimation of the probability in the system of "classifiers". *Pattern Recognition and Image Analysis*, Pleiades Publishing Ltd, v. 24, n. 1, p. 1–10, mar 2014.

LECUN, Y.; BENGIO, Y.; HINTON, G. Deep learning. *Nature*, Springer Science and Business Media LLC, v. 521, n. 7553, p. 436–444, may 2015.

LEE, Y.; KIM, Y.-S.; LEE, D. in; JEONG, S.; KANG, G.-H.; JANG, Y. S.; KIM, W.; CHOI, H. Y.; KIM, J. G.; CHOI, S. hoon. The application of a deep learning system developed to reduce the time for RT-PCR in COVID-19 detection. *Scientific Reports*, Springer Science and Business Media LLC, v. 12, n. 1, jan 2022.

LEWIS, D. D. Naive (bayes) at forty: The independence assumption in information retrieval. In: *Machine Learning: ECML-98*. [S.l.]: Springer Berlin Heidelberg, 1998. p. 4–15.

MANABE, Y. C.; SHARFSTEIN, J. S.; ARMSTRONG, K. The need for more and better testing for COVID-19. *JAMA*, American Medical Association (AMA), v. 324, n. 21, p. 2153, dec 2020.

METSKY, H. C.; FREIJE, C. A.; KOSOKO-THORODDSEN, T.-S. F.; SABETI, P. C.; MYHRVOLD, C. CRISPR-based surveillance for COVID-19 using genomically-comprehensive machine learning design. Cold Spring Harbor Laboratory, mar 2020.

MITSA, T. *Temporal Data Mining*. Hoboken: CRC Press, 2010. ISBN 9781420089769.

OLSON, R. S.; CAVA, W. L.; MUSTAHSAN, Z.; VARIK, A.; MOORE, J. H. Data-driven advice for applying machine learning to bioinformatics problems. *Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing*, v. 23, p. 192–203, 2018. ISSN 2335-6936.

PROVOST, F.; FAWCETT, T.; KOHAVI, R. The case against accuracy estimation for comparing induction algorithms. In: *In Proceedings of the Fifteenth International Conference on Machine Learning*. [S.l.]: Morgan Kaufmann, 1998. p. 445–453.

QUINLAN, J. R. Induction of decision trees. *Machine Learning*, Springer Science and Business Media LLC, v. 1, n. 1, p. 81–106, mar 1986.

QUINLAN, J. R. *C4.5*. [S.l.]: Elsevier Science & Techn., 2014. ISBN 9780080500584.

RAJKOMAR, A.; DEAN, J.; KOHANE, I. Machine learning in medicine. *New England Journal of Medicine*, Massachusetts Medical Society, v. 380, n. 14, p. 1347–1358, apr 2019.

RANGANATHAN, S.; NAKAI, K.; SCHONBACH, C. *Encyclopedia of Bioinformatics and Computational Biology*. [S.l.]: Elsevier Science & Techn., 2018. ISBN 9780128114322.

RODRÍGUEZ, J. J.; ALONSO, C. J.; BOSTROM, H. Boosting interval based literals. *Intelligent Data Analysis*, IOS Press, v. 5, n. 3, p. 245–262, may 2001.

ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958.

ROSSI, R. J. *Mathematical Statistics: An Introduction to Likelihood Based Inference*. [S.l.]: WILEY, 2018. ISBN 1118771044.

RUMELHART, D. E.; HINTON, G. E.; WILLIAMS, R. J. Learning representations by back-propagating errors. *Nature*, Springer Science and Business Media LLC, v. 323, n. 6088, p. 533–536, oct 1986.

RUSSELL, S. *Artificial intelligence a modern approach*. BostonColumbusIndianapolisNew YorkSan FranciscoBostonColumbusIndianapolisNew YorkSan Francisco: PearsonPearson, 2016. ISBN 9781292153964.

RUTKOWSKI, L.; JAWORSKI, M.; PIETRUCZUK, L.; DUDA, P. The CART decision tree for mining data streams. *Information Sciences*, Elsevier BV, v. 266, p. 1–15, may 2014.

SCHAPIRE, R. E. The strength of weak learnability. *Machine Learning*, Springer Science and Business Media LLC, v. 5, n. 2, p. 197–227, jun 1990.

SUTTON, R. S.; BARTO, A. G. *Reinforcement learning: An introduction*. [S.l.]: MIT Press Ltd, 2018. ISBN 0262039249.

TAHAMTAN, A.; ARDEBILI, A. Real-time RT-PCR in COVID-19 detection: issues affecting the results. *Expert Review of Molecular Diagnostics*, Informa UK Limited, v. 20, n. 5, p. 453–454, apr 2020.

TURING, A. M. I.—COMPUTING MACHINERY AND INTELLIGENCE. *Mind*, Oxford University Press (OUP), LIX, n. 236, p. 433–460, oct 1950.

VAPNIK, V. *The Nature of Statistical Learning Theory*. New York: Springer, 2010. ISBN 1441931600.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, L.; POLOSUKHIN, I. *Attention Is All You Need*. [S.l.]: arXiv, 2017.

WAN, S.; YANG, H. Comparison among methods of ensemble learning. In: *2013 International Symposium on Biometrics and Security Technologies*. [S.l.]: IEEE, 2013.

WANG, L. L.; LO, K.; CHANDRASEKHAR, Y.; REAS, R.; YANG, J.; EIDE, D.; FUNK, K.; KINNEY, R.; LIU, Z.; MERRILL, W.; MOONEY, P.; MURDICK, D.; RISHI, D.; SHEEHAN, J.; SHEN, Z.; STILSON, B.; WADE, A. D.; WANG, K.; WILHELM, C.; XIE, B.; RAYMOND, D.; WELD, D. S.; ETZIONI, O.; KOHLMEIER, S. Cord-19: The covid-19 open research dataset. abr. 2020.

WANG, Y.; HU, M.; LI, Q.; ZHANG, X.-P.; ZHAI, G.; YAO, N. Abnormal respiratory patterns classifier may contribute to large-scale screening of people infected with covid-19 in an accurate and unobtrusive manner. fev. 2020.

WILIMITIS, D. *The Kernel Trick in Support Vector Classification*. 2018. Https://towardsdatascience.com/the-kernel-trick-c98cdbcaeb3f.

XING, Z.; PEI, J.; YU, P. S. Early classification on time series. *Knowledge and Information Systems*, Springer Science and Business Media LLC, v. 31, n. 1, p. 105–127, apr 2011.

YANG, L.; SHAMI, A. On hyperparameter optimization of machine learning algorithms: Theory and practice. *Neurocomputing*, Elsevier BV, v. 415, p. 295–316, nov 2020.

YOUDEN, W. J. Index for rating diagnostic tests. *Cancer*, v. 3, p. 32–35, jan. 1950. ISSN 0008-543X.

YUAN, J. S.; REED, A.; CHEN, F.; STEWART, C. N. Statistical analysis of real-time PCR data. *BMC Bioinformatics*, Springer Science and Business Media LLC, v. 7, n. 1, feb 2006.

ZHOU, Z.-H.; FENG, J. Deep forest. *National Science Review*, Oxford University Press (OUP), v. 6, n. 1, p. 74–86, oct 2018.

ZIENELDIEN, T.; KIM, J.; CAO, J.; CAO, C. COVID-19 vaccines: Current conditions and future prospects. *Biology*, MDPI AG, v. 10, n. 10, p. 960, sep 2021.

ZOABI, Y.; DERI-ROZOV, S.; SHOMRON, N. Machine learning-based prediction of COVID-19 diagnosis based on symptoms. *npj Digital Medicine*, Springer Science and Business Media LLC, v. 4, n. 1, jan 2021.

APÊNDICE A – BASE DE DADOS

A base de dados desenvolvida e utilizada neste trabalho encontra-se disponível em: `https://github.com/karlvandesman/rtqpcr_covid/`