Revista Brasileira de Informática na Educação – RBIE
Brazilian Journal of Computers in Education
(ISSN online: 2317-6121; print: 1414-5685)
http://br-ie.org/pub/index.php/rbie

Submission: 01/03/2021; 1st round notif.: 27/04/2021; New version: 03/05/2021; 2nd round notif.: 31/05/2021;
Camera ready: 01/06/2021; Edition review: 02/06/2021; Available online: 13/06/2021; Published: 13/06/2021;

Processo de Mineração de Dados Educacionais aplicado na Previsão do Desempenho de Alunos: Uma comparação entre as Técnicas de Aprendizagem de Máquina e Aprendizagem Profunda

Title: Educational Data Mining Process applied to Student Performance Prediction: A comparison between Machine Learning and Deep Learning Techniques

Vanessa Faria de Souza
Programa de Pós-Graduação em Informática na Educação – UFRGS
vanessa.souza@ibiruba.ifrs.edu.br

Tony Carlos Bignardi dos Santos
Programa de Pós-Graduação em Informática na Educação – UFRGS
tonybignardi@gmail.com

Resumo

Com o aumento da disponibilidade de dados, sobretudo no contexto educacional, surgiram áreas específicas para extração de informações relevantes, como a Mineração de Dados Educacionais (MDE), que integra inúmeras técnicas que dão suporte à captação, processamento e análises desses conjuntos de registros. A principal técnica associada a MDE é a Aprendizagem de Máquina (AM), que vem sendo empregada a décadas no processamento de dados em diversos contextos, mas com a evolução tecnológica outras técnicas têm se sobressaído como a Aprendizagem Profunda (AP), baseada na aplicação de Redes Neurais Artificiais Multicamadas. Com foco neste contexto, esse estudo tem como objetivo realizar a previsão do desempenho de alunos, em um conjunto de dados públicos, e comparar as técnicas de AM e AP, ademais indicar quais os principais atributos preditores para o desempenho dos alunos. Para isso foi implementado um processo de MDE baseado em 4 etapas: 1) Coleta de dados; 2) Extração de recursos e limpeza de dados (pré-processamento e transformação); 3) Processamento analítico e algoritmos; e 4) Análise e interpretação dos resultados. Como resultado foi identificado que os modelos gerados a partir dos algoritmos tradicionais de AM têm um bom desempenho, mas inferior ao modelo AP que teve uma acurácia de 94%, bem como foi constatado que atributos relacionados às atividades escolares são mais preditores para o desempenho dos alunos do que os dados de características demográficas e socioeconômicas.
Palavras-Chave: *Mineração de Dados Educacionais, Aprendizagem Profunda, Aprendizagem de Máquina, Previsão de Desempenho.*

Abstract

With the increase in the availability of data, especially in the educational context, specific areas have emerged for the extraction of relevant information, such as Educational Data Mining (EDM), which integrates numerous techniques that support the capture, processing and analysis of these sets of records. The main technique associated with MDE is Machine Learning (ML), which has been used for decades in data processing in different contexts, but with the technological evolution other techniques have stood out such as Deep Learning (DL), based on the application of Multilayer Artificial Neural Networks. With a focus on this context, this study aims to predict the performance of students, using a set of public data, and to compare ML and DL techniques, in addition to indicating which are the main predictive attributes for student performance. For this, an EDM process based on 4 steps was implemented: 1) Data collection; 2) Resource extraction and data cleaning (pre-processing and transformation); 3) Analytical processing and algorithms; and 4) Analysis and interpretation of results. As a result, it was identified that the models generated from the traditional ML algorithms have a good performance, but inferior to the DL model, which had an accuracy of 94%, and it was found that attributes related to school activities are more predictive for the performance of students. students than data on demographic and socioeconomic characteristics.
Keywords: *Educational Data Mining, Deep Learning, Machine Learning, Performance Prediction.*

Cite as: Souza, V. F., & Santos, T. C. B. (2021). Educational Data Mining Process applied to Student Performance Prediction: A comparison between Machine Learning and Deep Learning Techniques (Processo de Mineração de Dados Educacionais aplicado na Previsão do Desempenho de Alunos: Uma comparação entre as Técnicas de Aprendizagem de Máquina e Aprendizagem Profunda). Brazilian Journal of Computers in Education (Revista Brasileira de Informática na Educação - RBIE), 29, 519-546. DOI: 10.5753/RBIE.2021.29.0.519

1 Introdução

Nos últimos anos a educação tem se modificado, em decorrência do avanço tecnológico disponível que direcionou a uma instrumentação do setor educacional, tanto em softwares voltados para o ensino, como na administração digital dos registros acadêmicos pelos gestores das instituições, bem como no uso da internet para a aprendizagem, em especial pela popularização do *e-learning*. Todos esses fatores impulsionaram um crescimento exponencial no volume de dados educacionais, e para se analisar uma grande quantidade de dados, é imprescindível contar com recursos computacionais, caso contrário a tarefa torna-se impraticável (Baker, 2015).

Dessa forma, as técnicas de mineração de dados estão ganhando cada vez mais importância no setor educacional, pois são uma forma de acompanhar, analisar e avaliar o processo de aprendizagem (Romero & Ventura, 2020). Provavelmente, as técnicas de mineração de dados podem fornecer aos formuladores de políticas educacionais modelos para apoiar seus objetivos de aprimorar a eficiência e a qualidade do ensino e da aprendizagem (Romero & Ventura, 2020). Além disso, o uso de diferentes técnicas de mineração de dados pode ser visto como base para uma mudança sistêmica, capaz de impactar de maneira positiva nas soluções de problemas específicos das Instituições de Ensino, por exemplo, viabilizando soluções que envolvam a personalização dos ambientes educacionais ou fornecendo suporte para o processo de tomada de decisão no ambiente educacional (Baker & Inventado, 2014; Baker, 2015; Romero & Ventura, 2013; e Romero & Ventura, 2020).

Nesse cenário, destaca-se a Mineração de Dados Educacionais (MDE) que utiliza as técnicas da Mineração de Dados (MD) para extrair informações relevantes de conjuntos diversificados de dados educacionais. Segundo a Sociedade Internacional de Mineração de Dados Educacionais, esta área pode ser definida da seguinte forma:

> É uma disciplina emergente, preocupada com o desenvolvimento de métodos para explorar dados únicos e cada vez mais em larga escala, provenientes de contextos educacionais e usa esses métodos para entender melhor os alunos e as configurações em que aprendem (MDE, 2020).

Em outras palavras, a MD refere-se a um conjunto de técnicas computacionais para extrair informações de grandes massas de dados, e quando os dados analisados são provenientes de contextos educacionais, chama-se MDE (Romero & Ventura, 2013). Igualmente, De Los Reyes *et al.* (2019) definem MDE como uma área voltada ao desenvolvimento de métodos para explorar dados oriundos de ambientes educacionais e utilizá-los para compreender melhor os processos de ensino e aprendizagem. Nessa acepção, Baker, Isotani & Carvalho (2011) alegam que a MDE é definida como a área de pesquisa que tem como finalidade o aperfeiçoamento e amadurecimento de técnicas para investigar conjuntos de dados obtidos em cenários educacionais. Conforme os autores, a natureza destes dados é mais diversa do que a observada nos dados tradicionalmente utilizados em tarefas de mineração, demandando adaptações e novas abordagens. Ao mesmo tempo, essa diversidade nos dados representa um potencial de implementação de recursos fundamentais para auxílio na melhoria da educação (Baker, Isotani & Carvalho, 2011; De Los Reyes *et al.,* 2019; Rigo *et al.,* 2014).

Sendo assim, necessita-se de técnicas e ferramentas que auxiliem na tarefa de verificar, interpretar e relacionar esses dados, com o intuito de gerar conhecimento útil e relevante, o que, segundo De Los Reyes *et al.* (2019) já era um objetivo das técnicas de MD, empregadas para identificar padrões de comportamento e encontrar insights que provoquem melhorias em produtos e serviços.

No que se refere ao processo de aplicação da MDE, este é similar ao da MD. Para Aggarwal (2015) o fluxo de trabalho de um processo típico de *Data Mining* contém as seguintes fases: 1)

Coleta de dados; 2) Extração de recursos e limpeza de dados (Pré-Processamento e Transformação) – para tornar os dados adequados para processamento; 3) Processamento analítico e algoritmos – projetar métodos analíticos eficazes para extrair informações e conhecimentos relevantes a partir dos dados processados; e o autor ainda sugere que os resultados precisam ser analisados e/ou interpretados, por isso cabe ao pesquisador verificar a melhor forma de realizar essa análise. A sequência das etapas do processo proposto por Aggarwal (2015) é apresentada na Figura 1, na qual pode-se observar que o processo de MD pode ser iterativo.

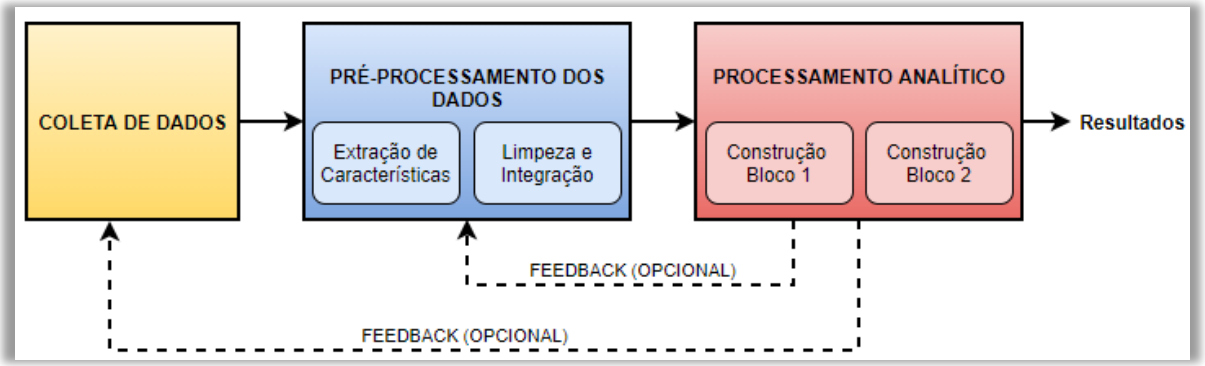


Figura 1: Processo de Data Mining proposto por Aggarwal (2015).
Fonte: Adaptado Aggarwal (2015).

Com relação à etapa do processamento analítico, há algumas técnicas mais proeminentes que podem ser utilizadas para este fim, como a estatística descritiva e inferencial, Aprendizagem de Máquina (AM), ou Aprendizagem Profunda (AP). Com relação a AP, desde 2006 essa técnica tem atraído muita atenção e tem sido aplicada com sucesso em muitas áreas, como reconhecimento de padrões, de fala e imagem (Schmidhuber, 2015), monitoramento da integridade da máquina (Zhao *et al.,* 2019), visão computacional, processamento de linguagem natural, detecção de intrusões e previsões médicas.

No entanto, de acordo com Yang, Zhang & Su (2019) as aplicações da AP no contexto educacional são relativamente escassas, pelo menos até o momento, em comparação à AM, mais estabelecida como técnica de Mineração de Dados. A AP pode ser empregada na extração de recursos, reconhecimento e classificação de padrões, portanto é uma abordagem capaz de solucionar problemas no âmbito educacional (Yang, Zhang & Su, 2019). Em recente mapeamento sistemático de literatura, Souza & Perry (2020) identificaram que dos 158 artigos mapeados, apenas 9 utilizaram como técnica de MDE a AP. Sendo muito extenso o corpo de estudos sobre AM na MDE, em detrimento a AP que é crescente em diversas outras áreas, todavia ainda não está consolidada no contexto da pesquisa educacional. Dessa forma, é importante a realização de estudos como este que denotem a eficácia da Aprendizagem Profunda e como ela pode ser útil em pesquisas nesse âmbito, em que a sua utilização pode levar a um crescente corpo de pesquisa com foco na melhoria da modelagem do comportamento e desempenho dos alunos, ampliando os horizontes de estudos na Mineração de Dados Educacionais.

Um dos principais objetivos da MDE é a previsão do desempenho de alunos, que procura identificar com antecedência como será a performance do aluno no decorrer do curso, para poder intervir caso necessário e assim melhorar seu processo de aprendizagem (Souza & Perry, 2020). No mapeamento citado, Souza & Perry (2020) apontaram 18 estudos que abordam esta temática, sendo a segunda mais investigada entre os pesquisadores (perdendo apenas para Análise de Comportamento). Isso se deve, ao desempenho dos alunos ser uma parte essencial nas instituições de ensino, visto que um dos critérios para que escolas e universidades sejam consideradas de alta qualidade é baseado em seu excelente histórico de realizações acadêmicas (Shahiri, Husain & Rashid, 2015). Dessa forma, Shahiri, Husain & Rashid (2015) afirmam que

prever o desempenho dos alunos é muito útil para ajudar educadores e alunos a melhorar o processo de ensino e aprendizagem.

Diante desse contexto, esse estudo tem como objetivo realizar a previsão do desempenho de alunos; para esse fim foi utilizado um conjunto de dados público do repositório *UCI Machine Learning*[1], com isso foi possível comparar técnicas já consolidadas no âmbito da MDE, com a técnica de AP. A abordagem utilizada foi a aprendizagem supervisionada para classificação, em que são previstas as notas dos alunos, mas estas foram divididas em quatro categorias, não sendo utilizados seus valores numéricos. Com a realização destas previsões, foi possível ainda, verificar se os atributos que compõem a base de dados são suficientes para realizar a geração de modelos eficazes na previsão do desempenho dos alunos, ademais avaliar se a Aprendizagem Profunda é uma boa alternativa às técnicas mais tradicionais empregadas no âmbito da MDE. Por fim, pretende-se com esse estudo disponibilizar para os interessados na área, um documento que apresenta de maneira detalhada como realizar o processo de Mineração de Dados Educacionais.

Para esse fim, este documento está organizado da seguinte forma: na seção 2 são tratados os principais aspectos sobre a técnica de Aprendizagem Profunda, bem como a arquitetura de Redes Neurais Artificiais Multicamadas utilizada nesse estudo; a seção 3 aborda trabalhos relacionados ao estudo aqui desenvolvido, trazendo quatro autores que utilizaram a técnica de Aprendizagem Profunda na MDE; a seção 4 trata dos procedimentos metodológicos deste estudo, em que é desenvolvido o processo de Mineração de Dados proposto por Aggarwal (2015); na seção 5 são expostos os resultados alcançados com esta investigação; por fim, na seção 6 são descritas as conclusões dos autores com o desenvolvimento deste estudo.

2 Aprendizagem Profunda

A AP estende as técnicas de AM, para resolver problemas que geralmente requerem maior capacidade de processamento. A AP é muito empregada para solução de problemas relacionados à visão computacional, reconhecimento de fala, processamento de linguagem natural e reconhecimento de áudio, e se baseia na implementação de Redes Neurais Artificiais. Destaca-se que na AP não há uma variedade de algoritmos como na AM, nesse âmbito são implementadas as Redes Neurais Artificiais Multicamadas (RNAM), conhecidas também como Redes Neurais Artificiais Profundas.

Para Lecun, Bengio & Hinton (2015, p. 436) AP é definida da seguinte forma:

> A AP permite que modelos computacionais compostos por várias camadas de processamento aprendam representações de dados com vários níveis de abstração. Esses métodos melhoraram drasticamente o estado da arte em reconhecimento de fala, reconhecimento visual de objetos, detecção de objetos e muitos outros domínios, como descoberta de medicamentos e estudos sobre o genoma. Com a otimização do algoritmo de retropropagação para indicar como a máquina deve alterar seus parâmetros internos as Redes Neurais Profundas são capazes de descobrir estruturas complexas de informações a partir de grandes conjuntos de dados. As Redes Convolucionais trouxeram avanços no processamento de imagens, vídeo, fala e áudio, enquanto as Redes Recorrentes se sobressaíram em dados sequenciais, como texto e fala.

Muito se tem pesquisado sobre Aprendizagem Profunda, sobretudo na área médica e biológica, em que esta técnica é proeminente em diferentes domínios, em especial para explorar o *Big Data*, para análises de diferentes aplicações como: reconhecimento de padrões, reconhecimento de fala, visão computacional, processamento de linguagem natural, detecção de intrusões e previsões médicas (Boulemtafes, Derhab & Challal, 2020). Nesse sentido, muitos autores definem esta técnica em seus estudos (Goodfellow, Bengio & Courville, 2016; Xin *et*

[1] https://archive.ics.uci.edu/ml/datasets/Student+Performance

al., 2018; Yang, Zhang & Su, 2018; Soffer *et al*., 2019; Le, Torrisi & Pollastri, 2020; Sengupta *et al.*, 2020; Murat *et al.*, 2020; Badar, Haris & Fatima, 2020; Boulemtafes, Derhab & Challal, 2020; Sezer, Gudelek & Ozbayoglu; 2020).

Destas definições, vale apena evidenciar a visão de Le, Torrisi & Pollastri (2020) sobre a AP, para eles esta técnica é um subcampo de AM baseado em Redes Neurais Artificiais Multicamadas, que enfatizam o uso de múltiplas camadas conectadas para transformar entradas em recursos passíveis de prever saídas correspondentes; os autores completam dizendo que diante de um conjunto de dados suficientemente grande de pares entrada-saída, um algoritmo de AP pode ser usado para aprender automaticamente o mapeamento de entradas com relação as saídas, ajustando um conjunto de parâmetros em cada camada da rede. Ainda destaca-se a conceituação realizada por Boulemtafes, Derhab & Challal (2020), em que os autores afirmaram que a AP é uma das abordagens mais avançadas da AM e atraiu muita atenção na pesquisa, em especial por prover a capacidade de superar a dependência de recursos mapeados à mão, que são enfrentados pelos algoritmos de AM tradicionais.

A datar de 2006, com a publicação de Hinton, Osindero & Teh (2006), na qual os autores apresentam um algoritmo (retropropagação) para o treinamento de RNAM que as tornaram mais rápidas e eficientes, a AP despontou como um campo capaz de modelar generalizações sobre dados utilizando três tipos de camadas de processamento: 1) Camada de entrada: que recebe os atributos da base de dados; 2) Camada oculta: que encapsulam as operações matemáticas intermediárias que dão suporte para a rede definir os resultados finais (na AP geralmente são empregadas várias camadas ocultas); e 3) Camada de saída: responsável por indicar a classificação da instância analisada. Nesse sentido, o termo "*deep* – profundo" refere-se a essa multiplicidade de camadas e a uma grande quantidade de camadas ocultas, por meio das quais os atributos são transformados.

Além da otimização do algoritmo para o treinamento, a consolidação da AP só foi possível por causa do aumento do volume de dados disponíveis e do avanço dos recursos computacionais (Lecun, Bengio & Hinton, 2015). Nessa perspectiva, um dos desafios da aplicação da AP era a dependência de grandes volumes de dados para expandir seu potencial em cada contexto de aplicação, pois como as redes neurais são treinadas por meio de exemplos, sua eficiência melhora à medida que a quantidade de dados processadas aumenta; contudo deve-se evidenciar que a quantidade de dados, apesar de apoiar a evolução da AP, não é o único fator, pois a qualidade destes dados também impacta, somente grandes dimensões de dados podem gerar *underfitting* ou *overfitting*[2], pois as instâncias podem estar desbalanceadas. À medida que a AP evolui, ela tem potencial de identificar novas informações sobre conjunto de dados, que as tecnologias até então existentes não conseguiram (Hinton, Osindero & Teh, 2006; Lecun, Bengio & Hinton, 2015); nesse sentido, pode-se dizer que a era do *Big Data* possibilitou um amplo potencial de inovação.

Em relação aos avanços dos recursos computacionais, destaca-se seu aspecto decisivo para a difusão da AP, sobretudo porque as Redes Neurais Profundas possuem várias camadas de processamento, estruturas que dependem de muitos recursos computacionais, por vezes não disponíveis em um computador comum. Em função da necessidade de otimização de processamento, atualmente pode-se usar "placas de vídeo" - GPU (Graphics Processing Unit) – pois ainda que não sejam tão rápidas quanto uma CPU, podem realizar operações com pontos flutuantes mais rapidamente, o que as tornam interessantes para renderizar gráficos. Com a popularização da internet com banda larga, é possível submeter os modelos a servidores que

[2] O o*verfitting* ocorre quando o modelo se adaptou muito bem aos dados com os quais está sendo treinado; porém, não generaliza bem para novos dados. Ou seja, o modelo "decorou" o conjunto de dados de treino, mas não aprendeu de fato o que diferencia aqueles dados para quando precisar enfrentar novos testes. O *underfitting* ocorre quando o modelo não se adapta bem sequer aos dados com os quais foi treinado.

contém "fazendas" de GPUs, que podem executá-los remotamente. Exemplos desse tipo de serviço são: Pytorch (Facebook), TensorFlow (Google), MXnet (Apache) e H2O (RStudio). Tudo isso permitiu a adoção e pesquisa acentuada utilizando essa técnica, que está sendo cada vez mais aplicada em processos de tomada de decisão e se configura um campo de pesquisa ativa (Lecun, Bengio & Hinton, 2015).

Outra particularidade relevante na AP é que, embora não possua uma grande variedade de algoritmos, as arquiteturas são diferenciadas, e nesse sentido, Aggarwal (2018) considera como as principais arquiteturas: Redes *Multiplayer Perceptrons* (MLP), Redes Neurais Convolucionais, Redes Neurais Recorrentes e Redes pré-treinadas – não supervisionadas. *Redes Multiplayer Perceptrons (MLP):* são utilizadas principalmente em problemas de classificação com o intuito de identificar padrões em conjuntos dados. As Redes Neurais do tipo MLP são formadas por uma camada de entrada, uma camada de saída e um número variável de camadas ocultas (desde que maior que dois). *Redes Neurais Convolucionais:* são empregadas no reconhecimento de imagens e identificação de objetos, e têm este nome pois usam a convolução em pelo menos uma de suas camadas. *Redes Neurais Recorrentes:* são projetadas para tratamento de problemas que envolvam dados sequenciais, como frases de texto, séries temporais, e outras tipos de sequências, sendo muito aplicadas em tarefas de mineração de texto. As Redes Neurais Recorrentes são redes com loops, permitindo que as informações persistam, assim elas podem se conectar com informações anteriores para uma tarefa atual. *Redes pré-treinadas – não supervisionadas:* cujo principal objetivo é a redução da dimensionalidade, compactação e fusão (possui os mesmos objetivos da Análise dos Componentes Principais).

Por fim, cabe destacar que no contexto da Aprendizagem Profunda e Aprendizagem de Máquina há uma extensa terminologia, e para um melhor entendimento desse estudo é preciso conhecer alguns desses termos. No caso desta pesquisa, é importante compreender os conceitos: Treinamento; Teste e Modelo – Treinamento é a fase em que o algoritmo de AM ou AP é aplicado a uma base de dados e este deve buscar padrões sobre os dados; Teste é a fase em que o modelo é avaliado; e Modelo é o produto da submissão dos dados a um algoritmo, muitas vezes o modelo é confundido com o próprio algoritmo, os dois são formados por códigos, mas não correspondem a mesma estrutura. Depois da fase de treinamento de um algoritmo, ele gera um modelo que pode ser salvo para ser posteriormente aplicado a uma nova base de dados, desde que essa possua os mesmos atributos.

Como visto há alguns tipos de arquiteturas de Aprendizagem Profunda mais proeminentes, neste estudo foi empregada a arquitetura *Multiplayer Percetron,* que embora seja uma das mais simples, devido também a simplicidade da base de dados se ajusta bem aos objetivos. Dessa forma esta arquitetura é descrita na sequência levando em consideração autores como Kubat (2017), Igual & Seguí (2017), Aggarwal (2015) e Aggarwal (2018).

2.1 Arquitetura *Multiplayer Perceptron*

A arquitetura da rede neural do tipo Multiplayer Perceptron (MLP) é a mais simples, porém uma das mais adequadas para resolução de problemas de classificação em geral, essa tarefa está estreitamente relacionada com à identificação de padrões em dados. As redes do tipo MLP têm funcionalidade parecida a algoritmos de classificação de AM, porque pertencem à classe de RNAM de Aprendizagem Supervisionada, entretanto em muitos contextos têm apresentado capacidade de processamento de dados superior aos algoritmos tradicionais (Aggarwal, 2018; Bishop, 1995; Hand, 1997; Igual & Seguí, 2017; Kubat, 2017; Ripley, 1996).

A arquitetura MLP consiste em uma rede de nós que são estruturas de códigos dispostos em camadas, muitos autores também denominam essa estrutura como perceptron ou neurônio artificial (Aggarwal, 2018; Bishop, 1995; Hand, 1997; Igual & Seguí, 2017; Kubat, 2017; Ripley, 1996). Uma rede típica de MLP consiste em três tipos de camadas de processamento: uma

camada de entrada que recebe dados externos; um número arbitrário de camadas ocultas que fazem os cálculos intermediários e auxiliam a rede a encontrar os valores finais; e uma camada de saída que constrói os resultados da classificação. Destaca-se, que a diferença entre uma Rede Neural Artificial simples e uma RNAM é justamente a quantidade de camadas ocultas, na RNAM o número de camadas ocultas na maioria dos casos é superior a dois. Uma ilustração de como as camadas estão organizadas em uma rede MLP é apresentada na Figura 2.

O número de nós da camada de entrada é determinado pela quantidade de atributos que há na base de dados fornecida. O número de nós na camada de saída é definido pela quantidade de classes do problema, um problema de classificação binária, por exemplo tem 2 nós na camada de saída. Para a determinação do número de camadas ocultas e a quantidade de nós em cada uma dessas camadas, existem uma variedade de metodologias para a escolha de tais aspectos, mas usualmente essa definição é realizada de forma empírica, com base em testes para cada contexto (Aggarwal, 2018). Ao contrário das outras camadas, nenhum cálculo está envolvido na camada de entrada. O princípio da rede é que quando os dados são apresentados na camada de entrada, os nós da rede realizam cálculos nas camadas sucessivas até que um valor de saída seja obtido para cada um dos nós, esse sinal de saída deve poder indicar a classe apropriada para as instâncias da base de dados. Um nó em uma Rede Neural Artificial pode ser representado como na Figura 3 (Aggarwal, 2018).

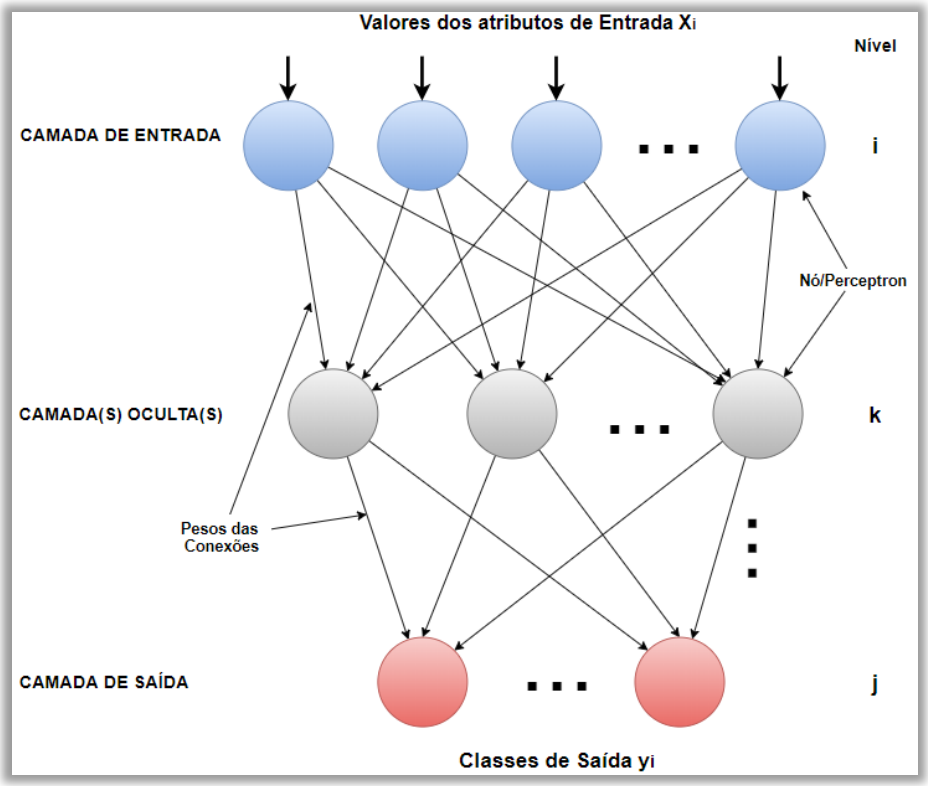


Figura 2: Arquitetura de Rede Multiplayer Perceptron.
Fonte: Adaptado Aggarwal (2018).

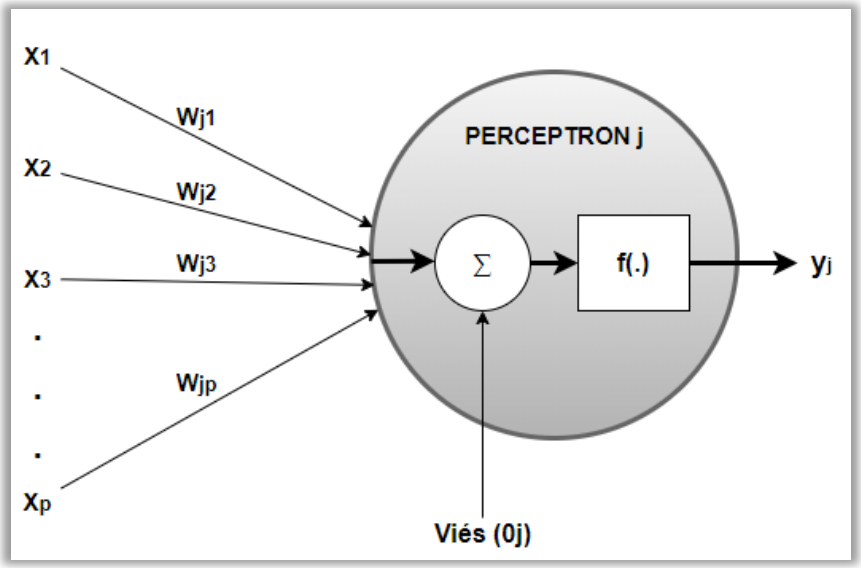


Figura 3: Um nó da Rede MLP – Perceptron.
Fonte: Adaptado Aggarwal (2018).

As Redes Neurais Artificiais são algoritmos que efetuam transformações matemáticas sobre as bases de dados que recebem para processar. Em cada nó de cada camada, é realizada uma multiplicação entre o valor de entrada pelo peso do nó correlato, soma-se com o viés vinculado a esse nó e encaminha-se esse resultado adiante, aplicando a concepção de *feed forward* (avançar). O viés é similar ao intercepto acrescido em uma equação linear, ele funciona como um parâmetro adicional que é aplicado para ajustar a saída junto da soma ponderada das entradas para o perceptron, é uma constante que auxilia o modelo a se adaptar melhor aos dados fornecidos (AggarwaL, 2018).

De forma geral, sem expor o aspecto matemático do processamento, o funcionamento de uma RNAM pode ser visualizado na Figura 4 que representa o fluxograma dos procedimentos que compõem o funcionamento do algoritmo de retropropagação, que serve para treinar uma rede do tipo MLP. O algoritmo básico da retropropagação presente nas pesquisas de Bishop (1995) e Duda, Hart & Strok (2001) e otimizado por Hinton, Osindero & Teh (2006) funciona de forma simplificada, executando as seguintes etapas: 1) Inicializam-se todos os pesos de conexão *w* com pequenos valores aleatórios de um gerador de sequência; 2) Baseado nos atributos de saída (aprendizagem supervisionada) realizam-se os cálculos com os pesos e se calcula o erro; 3) Calcula as mudanças nos pesos e os atualiza, repete-se até a convergência (que ocorre quando o erro estiver abaixo de um valor predefinido).

Logo na primeira execução esse processo vai encontrar valores para as saídas, embora dificilmente sejam valores precisos, principalmente porque os pesos são atribuídos de forma aleatória, ao passo que esses devem condizer com os dados fornecidos na base, por isso devem ser atualizados até que se encontre o melhor conjunto de pesos para os dados fornecidos. Ressalta-se que depois de aplicado o algoritmo de retropropagação, correspondente ao treinamento da Rede Neural Profunda, gera-se um modelo de classificação, formado pelos pesos assimilados pela rede em acordo com a base de dados fornecida, esse modelo pode então ser salvo e aplicado a dados novos no contexto desejado.

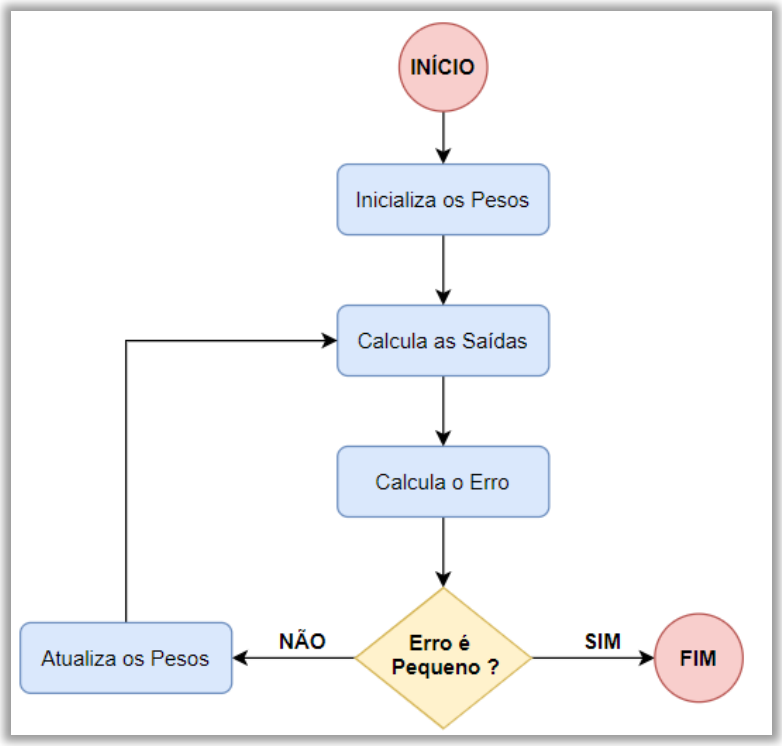


Figura 4: Treinamento da uma RNAM.
Fonte: Autores.

Com o exposto foi possível entender, de forma geral, e como é a arquitetura de uma RNAM utilizada no contexto de problemas de classificação, salienta-se que todo o processo matemático foi omitido desse documento e é encapsulado pelas ferramentas utilizadas para sua implementação. Entretanto, sem o conhecimento dos elementos que o compõem seria difícil testar e alterar, quando necessário, os parâmetros fundamentais para processamento de dados por modelos de AP.

Com relação às ferramentas para a aplicação de AP, é importante destacar que a escolha é um pouco mais complexa do que apenas a utilização das bibliotecas disponíveis na linguagem de ciência de dados a ser utilizada. A partir da decisão de se utilizar a AP como técnica de MDE, é necessária a elaboração de uma pesquisa sobre as principais e melhores ferramentas disponíveis, de preferência sob a licença de código aberto para sua aplicação. A escolha pelo código aberto, diz respeito sobretudo pela flexibilidade para configuração das ferramentas, não disponível nas proprietárias. Para a aplicação de AP é necessário um *framework* especializado neste tipo de algoritmo, até mesmo por questões físicas do computador – quanto ao hardware – que devem ser levadas em consideração neste cenário, pois o processamento requerido pelas RNAM é mais complexo e com bases de dados de grandes dimensões, muito recurso computacional é exigido. De acordo com Alvim (2010) framework é uma coleção de classes que contribuem entre si propiciando melhores práticas de desenvolvimento e diminuição da repetição de tarefas. Além disso, evita variações de soluções diferentes para um mesmo tipo de problema, o que facilita a reutilização e customização dos códigos.

Neste sentido, analisou-se 3 pesquisas que realizaram comparações entre *frameworks* para *Deep Leaning*: Bahrampour *et al.* (2015) que verificaram os desempenhos do *Caffe, do Neon, do TensorFlow, do Theano e do Torch*; Kovalev, Kalinovsky & Kovalev (2016) que realizaram uma avaliação do funcionamento e eficácia do *Theano, do Torch, do Caffe, do TensorFlow e do DeepLearning4J;* e por fim, NG *et al.* (2016) que implementaram um estudo comparativo do desempenho do *Singa* e do *H2O.* Das conclusões dos autores destacam-se: O TensorFlow é suficientemente flexível, todavia seu desempenho em tempo de treinamento com uma única GPU

não é satisfatória em comparação aos outros; o framework Deeplearning4J tem um desempenho inferior para o treinamento, mesmo modificando a quantidade de camadas da Rede Neural; quanto à complexidade de programação o mais complicado de se utilizar é o Torch; e O H2O obteve um desempenho estável e precisa para as bases de dados testadas, possui baixa complexidade de implementação, podendo ser utilizado com as linguagens R ou Python e proporciona acesso a várias GPUs integradas em servidores on-line.

A partir dos itens destacados, foi possível escolher uma ferramenta flexível, de baixa complexidade e eficaz, o H2O, como ele está disponível para várias linguagens de programação optou-se por usar o R, pois essa linguagem já vem sendo utilizada pelos pesquisadores em diversos estudos. Além disso, o *Framework* para linguagem R possibilita a utilização de atributos categóricos – sem precisar convertê-los para numéricos, como no Python – um diferencial decisivo para sua escolha, esse fator diminui muito a complexidade do pré-processamento e transformação a serem empregados nas bases de dados em linhas de código. Outra questão muito importante é o treinamento do modelo que é realizado em GPU, devido ao aumento da dimensionalidade da base, assim como da complexidade de processamento realizado pela RNAM, sem essa possibilidade seria inviável a execução em um computador comum.

3 Trabalhos Relacionados*:* Aplicação de Aprendizagem Profunda como técnica de MDE

Dentro do domínio de aplicação da AP no âmbito educacional, alguns estudos começaram a despontar como por exemplo Lin *et al.,* (2019), que propuseram uma Rede Neural Convolucional para analisar vídeo aulas de cursos on-line, com o propósito de detectar e classificar de forma automática os conteúdos exibidos nesses vídeos a fim de melhorar o desempenho da plataforma de aprendizado on-line. No que diz respeito à Rede Neural Convolucional, sua implementação foi executada com o Framework Tensorflow e foi treinada e testada em um conjunto de dados de 16 mil imagens e 72 horas de áudio, pertencentes a dois cursos diferentes.

Ademais, destaca-se a investigação realizada por Guo *et al.* (2019), na qual os autores implementaram uma Rede Neural Profunda Híbrida para a identificação de postagens "urgentes" que requerem atenção imediata de instrutores em fóruns de discussão em cursos on-line. Quanto à estrutura, é uma Rede Neural Convolucional combinada com uma Rede Neural Recorrente, implementada para mineração de texto e funciona em 3 etapas: 1) assimilar simultaneamente as informações semânticas e estruturais das sentenças de texto das postagens; 2) utilizar as Redes Convolucionais em nível de caractere para capturar informações – isso foi necessário devido à muito ruído, como erros de ortografia e emoticons no texto das postagens; e 3) associar as informações semânticas e estruturais com as informações de caracteres e assim chegar a representação final da frase. Guo *et al.* (2019) chegaram a resultados que superam a precisão de soluções presentes no estado da arte em até 2,4%, e concluíram que sua pesquisa pode auxiliar professores e tutores a priorizar suas respostas e gerenciar melhor várias postagens, de modo que esses profissionais da educação possam responder às perguntas dos alunos em tempo hábil e ajudar a reduzir as taxas de evasão nesses cursos.

Wen *et al.* (2020) também utilizaram um modelo de AP para pesquisa no âmbito educacional, com o objetivo de identificar antecipadamente a desistência em Massive Open Online Courses (MOOCs). Nesse sentido, depois que Wen *et al.* (2020) realizaram uma análise dos padrões de comportamento de aprendizagem dos alunos de um MOOC, relataram que esses estudantes geralmente exibem comportamentos de aprendizagem semelhantes em vários dias consecutivos (o status de aprendizagem de um aluno para o dia subsequente, provavelmente será semelhante ao do dia anterior). Embasados nessa premissa Wen *et al.* (2020) propuseram uma base de dados formada por atributos relacionados à correlação local de comportamentos de

aprendizagem, sobre a qual aplicaram uma Rede Neural Convolucional, gerando um modelo para prever o abandono de alunos em MOOCs. Por fim, o modelo proposto obteve uma precisão que variou de 86% a 89%, e os autores destacaram que as principais contribuições da pesquisa foram: 1) definição do conceito de status de aprendizagem para encontrar a correlação local de comportamentos de aprendizagem; 2) construção de uma base de dados formada a partir dos atributos da correlação local de comportamentos de aprendizagem; e 3) implementação de um modelo construído a partir de uma Rede Neural Convolucional para previsão do abandono de alunos em MOOCs.

Por fim, cita-se o estudo de Waheed *et al.* (2020) que desenvolveram um modelo de Rede Neural Artificial Multicamadas com arquitetura MLP, com o objetivo de prever o desempenho de alunos em MOOCs. Para isso os autores utilizaram relatórios de 7 MOOCs, com um total 32 mil alunos, e os atributos utilizados foram: perfil demográfico, fluxo de cliques e desempenho nas avaliações. O estudo foi conduzido com base na mineração de dois conjuntos de dados: 1) notas das atividades avaliativas na plataforma e perfil demográfico; e 2) atributos trimestrais do fluxo de cliques de cada aluno, o resultado foi um modelo para prever o risco de reprovação. Os autores relataram que em contraste com os métodos estatísticos, Redes Neurais Artificias Multicamadas facilitam a generalização, o que possibilita inferir padrões escondidos nos dados, dando suporte a fazer suposições sobre eles. A precisão alcançada pelo modelo ficou entre 84% e 93% nos experimentos realizados, e Waheed *et al.,* (2020) concluíram que esses resultados demonstram a efetividade do modelo implementado para a previsão precoce do desempenho de alunos em MOOCs. Os autores ainda ressaltaram que estudos como esse, orientados a dados, são necessários para auxiliar instituições de ensino na formulação de uma estrutura de análise de aprendizagem, contribuindo para o processo de tomada de decisão.

Tais investigações denotam o potencial da Aprendizagem Profunda no contexto educacional, contudo, apesar do bom desempenho frequentemente relatado desses modelos aplicados a MDE, o grande número de parâmetros que devem ser configurados, alguns deles expostos na seção 2.1 (para mais detalhes sobres esses parâmetros consulte Apêndice A), e o entendimento necessário sobre eles, os tornam difíceis de interpretar e implementar, supõe-se ser este um dos motivos dessa técnica não ser ainda tão explorada neste contexto (Yang, Zhang & Su, 2019).

4 Processo de MDE para Previsão do Desempenho de Alunos

Este estudo tem como principal objetivo realizar a previsão do desempenho de alunos, em duas disciplinas no ensino tradicional, utilizando técnicas de MDE. Com a realização desse processo é também possível identificar qual das técnicas aplicadas foi a mais eficaz na predição do desempenho desses estudantes, considerando os atributos que compõem a base de dados. À vista disso, formulou-se as questões de pesquisa que nortearam este estudo:

- ✓ *Questão de pesquisa 1 (QP1)* – Qual a eficácia de modelos gerados a partir de algoritmos baseados em *Aprendizagem de Máquina* e *Aprendizagem Profunda* na previsão do desempenho de alunos no ensino tradicional?
- ✓ *Questão de pesquisa 2 (QP2)* – Modelos baseados em algoritmos de *Aprendizagem Profunda* têm uma eficácia superior a modelos baseados em algoritmos tradicionais utilizados na *Mineração de Dados Educacionais*?
- ✓ *Questão de pesquisa 3 (QP3)* – Qual conjunto de atributos tem mais influência na previsão do desempenho de alunos?

Para responder essas questões foram empregados alguns procedimentos que configuram a metodologia adotada nesta pesquisa, que em termos gerais resume-se na realização do processo de *Data Mining* proposto Aggarwal (2015). Como descrito na introdução esse processo é

composto basicamente por 3 fases: 1) Coleta de dados; 2) Extração de recursos e limpeza de dados (Pré-Processamento e Transformação); e 3) Processamento analítico e algoritmos; por fim, de acordo com Aggarwal (2015) uma quarta fase de análise e/ou interpretação dos resultados deve ser acrescentada, conforme os objetivos do processo de mineração de dados realizado. Essas etapas compõem basicamente os procedimentos metodológicos adotados nessa pesquisa e são detalhados na sequência.

4.1 Primeira Fase: Coleta de Dados

Na primeira fase os dados foram coletados do repositório de dados público o *UCI Machine Learning.* Estes dados abordam o desempenho dos alunos no ensino secundário de duas escolas portuguesas. Os atributos dos alunos incluem notas e características demográficas (sociais e escolares). Tais informações foram reunidas por meio de relatórios escolares e questionários. Foram fornecidos dois conjuntos de dados de 1044 alunos relativos ao desempenho em duas disciplinas distintas: Matemática e Língua Portuguesa. Os atributos constantes na base de dados extraída estão descritos na Tabela 1. Os dados sistematizados formaram um *Data Frame* com 1044 linhas e 33 colunas.

Tabela 1: Atributos da Base de Dados.

ID	ATRIBUTOS	DESCRIÇÃO
1	Escola	Escola do aluno (binário: 'GP' - Gabriel Pereira ou 'MS' - Mousinho da Silveira)
2	Gênero	Gênero do aluno (binário: 'F' - feminino ou 'M' - masculino)
3	Idade	Idade do aluno (numérico: de 15 a 22)
4	Endereço	Tipo de endereço residencial do aluno (binário: 'U' - urbano ou 'R '- rural)
5	Famsize	Tamanho da família (binário:' LE3 '- menor ou igual a 3 ou' GT3 '- maior que 3)
6	Pstatus	Status de coabitação dos pais (binário:' T '- morando junto ou' A '- à parte)
7	Medu	Escolaridade da mãe (numérico: 0 - nenhum, 1 - ensino fundamental (4ª série), 2 - 5ª a 9ª série, 3 - ensino médio ou 4 - ensino superior)
8	Fedu	Escolaridade do pai (numérico: 0 - nenhuma, 1 - ensino primário (4º ano), 2 - 5º ao 9º ano, 3 - ensino secundário ou 4 - ensino superior)
9	Mjob	Trabalho da mãe (nominal: 'professor', 'saúde' relacionado, 'serviços' civis (por exemplo, administrativo ou policial), 'em_casa' ou 'outro')
10	Fjob	Trabalho do pai (nominal: 'professor', 'saúde' relacionado, civil 'serviços' (por exemplo, administrativo ou policial), 'em_ casa' ou 'outro')
11	Razão	Razão para escolher esta escola (nominal: perto de 'casa', escola 'reputação', 'curso' preferência ou 'outro')
12	Tutor	Tutor do aluno (nominal: 'mãe', 'pai' ou 'outro')
13	Tempo de Viagem	Tempo de viagem de casa para a escola (numérico: 1 - <15 min., 2 - 15 a 30 min., 3 - 30 min. A 1 hora, ou 4 -> 1 hora)
14	Horas de Estudo	Tempo de estudo semanal (numérico: 1 - <2 horas, 2 - 2 a 5 horas, 3 - 5 a 10 horas ou 4 -> 10 horas)
15	Reprovações	Número de reprovações anteriores nas aulas (numérico: n se 1 <= n <3, senão 4)
16	Schoolup	Suporte educacional extra (binário: sim ou não)
17	Famsup	Suporte educacional familiar (binário: sim ou não)
18	Pago	Aulas extras pagas dentro da disciplina (matemática ou português) (binário: sim ou não)
19	Atividades	Atividades extracurriculares (binário: sim ou não)
20	Creche	Cursou creche (binário: sim ou não)
21	Superio	Deseja cursar o ensino superior (binário: sim ou não)
22	Internet	Acesso à internet em casa (binário: sim ou não)
23	Romântico	Com um relacionamento romântico (binário: sim ou não)
24	Famrel	Qualidade das relações familiares (numérico: de 1 - muito ruim a 5 - excelente)
25	Tempo Livre	Tempo livre depois da escola (numérico: de 1 - muito baixo a 5 - muito alto)
26	Gooout	Saindo com os amigos (numérico: de 1 - muito baixo a 5 - muito alto)
224	Dalc	Consumo de álcool durante o trabalho (numérico: de 1 - muito baixo a 5 - muito alto)

28	Walc	Consumo de álcool no fim de semana (numérico: de 1 - muito baixo a 5 - muito alto)
29	Saúde	Estado de saúde atual (numérico: de 1 - muito ruim a 5 - muito bom)
30	Faltas	Número de faltas na escola (numérico: de 0 a 93)
32	G1	Nota do primeiro período (numérico: de 0 a 20)
32	G2	Nota do segundo período (numérico: de 0 a 20)
33	G3	Nota final (numérico: de 0 a 20, meta de saída)

Fonte: *UCI – Machine Learning:* https://archive.ics.uci.edu/ml/datasets/Student+Performance

4.2 Segunda Fase: Extração de recursos e limpeza de dados (Pré-Processamento e Transformação)

Com relação à segunda fase os dados foram pré-processados para se adequarem à aplicação das técnicas de MDE – algoritmos de AM e AP – utilizadas na previsão do desempenho dos alunos, para isso foi utilizada a linguagem de programação e ciência de dados R, e várias tarefas foram realizadas:

1. Junção dos dados dos alunos das duas disciplinas em uma única base, para todos os alunos – como os dados disponíveis no *UCI Machine Learning* estavam divididos em duas bases de dados (uma para os dados da disciplina de matemática e outra para os dados de português), foi elaborado um script para unir os dados em uma única base.
2. Transformação do atributo G3 de numérico para níveis de classificação – nesse procedimento cada faixa de notas recebeu um valor no formato de caractere, atribuindo uma categoria/classe para os registros: notas entre 20 e 16 = "A", notas entre 15 e 11 = "B", notas entre 10 e 4 = "C", notas entre 4 e 0 = "D". Essa abordagem foi inspirada no estudo desenvolvido por Zhang & Wu (2019), os autores tinham como objetivo prever o desempenho de alunos, mais especificamente a previsão de notas dos alunos. Para isso, os autores utilizaram dados de estudantes que cursaram um curso MOOC sobre Programação em Linguagem C, os atributos eram basicamente sobre: informações básicas do perfil dos alunos, pontuação nas atividades avaliativas, número de questões solucionadas, pontuação final, postagens nos fóruns de discussão. Quanto a abordagem para a solução do problema, embora fosse esperado que os autores utilizassem Regressão, pois as notas são valores contínuos, os autores dividiram os resultados dos alunos em classes de 85 a 100 – A, de 70 a 85 – B, de 60 a 70 – C e >60 – D, portanto tornando-se um problema de classificação. Após a formatação e classificação da base de dados os autores realizaram a aplicação de 3 algoritmos de AM para geração de modelos de previsão de notas: ID3, C4.5 e CART – todos baseados em Árvore de Decisão. As precisões alcançadas pelos modelos sobre a base de dados de teste foram: IDE3 – 81%, C4.4 – 75%, CART – 76%. Nesse sentido, considerou-se essa abordagem bastante eficaz e aderente ao estudo aqui desenvolvido, por isso optou-se por fazer de forma semelhante, realizando a categorização das notas, em quatro classes A, B, C e D.
3. Formatação dos dados como *Data Frame*[3]: como a base de dados possui atributos numéricos e categóricos a única opção de formatação na linguagem R é o *Data Frame*.
4. Divisão da base de dados em treinamento e teste, em que 85% dos dados da base foram definidos para treino (888 registros para treino e 156 para teste): como a base de dados utilizada neste estudo é relativamente pequena (apenas 1044 linhas de informação), optou-se por usar mais dados para o treinamento dos algoritmos, e menos dados para sua avaliação. Existem autores na literatura que corroboram este tipo de adaptação quando se possui base de dados pequenas (Japkowicz & Shah, 2014).

[3] Diferentemente de vetores, que não podem agregar elementos de classes diferentes, um *Data Frame* pode conter colunas com vetores numéricos e variáveis categóricas. É por esse motivo que a maior parte dos dados que são utilizados em mineração de dados e análises estatísticas são formatados como *Data Frames*.

5. Transformação do atributo a ser previsto (atributo meta – o desempenho final do aluno) para *Factor*[4] (G3): para a aplicação de algoritmos de AM e AP com a linguagem R, o atributo meta deve estar configurado com o formato *Factor*, isso é um pré-requisito das bibliotecas do R.

4.3 Terceira Fase: Processamento analítico e algoritmos

No que tange a Processamento analítico e algoritmos, é nesta fase que são escolhidos os algoritmos para geração dos modelos, bem como das previsões projetadas. Nesse sentido, é importante destacar um estudo que utilizou a mesma base de dados desta pesquisa: Cortez & Silva (2008). Neste estudo os dois conjuntos de dados, foram modelados com uma classificação de cinco níveis e foi utilizada a regressão, pois o atributo a ser previsto era a nota (numérico). O objetivo de Cortez & Silva (2008) era analisar o desempenho dos alunos, sob uma perspectiva de quais desses mais influenciam na previsão do desempenho. Para isso, os autores utilizaram quatro algoritmos: Árvores de decisão, *Random Forest*, Redes Neurais Simples e *Suporte Vector Machines*. Os resultados dos autores mostraram que uma boa precisão preditiva pode ser alcançada, desde que estejam disponíveis as primeiras e/ou segundas séries do período escolar. Cortez & Silva (2008) ressaltam ainda que o desempenho do aluno é altamente influenciado por avaliações anteriores e pelo número de faltas. Como resultado direto desta pesquisa os autores relatam que ferramentas mais eficientes de previsão do aluno podem ser desenvolvidas, melhorando a qualidade da educação e aprimorando a gestão dos recursos escolares.

Em contraste, como já destacado, neste estudo as notas de desempenho dos alunos foram classificadas como A, B, C ou D, e, portanto, foram empregados algoritmos de classificação para realização das previsões, diferentemente de Cortez & Silva (2008) que utilizaram os algoritmos para regressão. Tanto a classificação como a regressão são tarefas de Aprendizagem Supervisionada, nesse tipo de técnica a base de dados possui colunas com categorias que servem para treinar o modelo, que deve, na próxima etapa, identificar as categorias de cada linha. A Aprendizagem Supervisionada é utilizada para resolver dois tipos diferentes de problemas: classificação e regressão. A classificação refere-se ao processo de previsão de valores de categorias, como no caso deste estudo uma faixa de notas que foi classificada em uma categoria. Problemas de regressão buscam prever um valor numérico, por exemplo prever o valor da nota dos alunos, como fizeram Cortez & Silva (2008). Alguns dos algoritmos de AM para classificação mais conhecidos são: Naïve Bayes; Árvore de Decisão, *Random Forest* (RF) e *Suport Vector Machines* (SVM), estes foram selecionados para serem utilizados nesta pesquisa. Quanto à AP, a arquitetura *Multiplayer Perceptron* (MLP) (descrita na seção 2.1), é uma boa opção para classificação em bases formadas por atributos simples (que não incorporam a mineração em texto, em imagem ou em vídeos).

Para realizar a escolha desses algoritmos, para este estudo, foram feitas várias pesquisas na literatura sobre o tema, todavia um estudo em particular foi muito útil para tomar a decisão em escolher esses algoritmos, a revisão sistemática de literatura realizada por Shahiri, Husain & Rashid (2015), essa forneceu uma visão geral das técnicas de mineração de dados que eram usadas para prever o desempenho dos alunos, em publicações datadas entre 2002 e 2015, sistematizando as melhores pesquisas nesta área. O estudo também se concentrou em como os algoritmos de previsão poderiam ser usados para identificar os atributos mais importantes dentre a diversidade de dados dos alunos, como também se realiza neste estudo. Nessa revisão, os autores seguiram duas questões de pesquisa para estruturar os resultados: 1) Quais são os

[4] classe de objetos para trabalhar com atributos categóricos na linguagem R. A linguagem R adapta automaticamente a resposta dos comandos quando o objeto é um *Factor*.

atributos mais importantes empregados na previsão do desempenho dos alunos; e 2) Quais as técnicas/algoritmos de previsão mais eficientes.

Quanto aos principais atributos, Shahiri, Husain & Rashid (2015) apontaram que foram usados com frequência a média cumulativa de notas e a avaliação interna, empregados por 10 dos 30 artigos selecionados para a revisão. Os autores também chegaram à conclusão, que a Aprendizagem de Máquina era a técnica mais utilizada e quanto à eficácia dos algoritmos as Redes Neurais tiveram a maior precisão (98%) para previsão do desempenho dos alunos, seguida das Árvores de Decisão (91%), depois as Máquinas de Vetores de Suporte e K-Nearest Neighbors (KNN – K-ésimo Vizinho mais Próximo) com a mesma eficácia (83%), por fim, o método menos preciso foi o Naive Bayes (76%). No presente estudo, apenas não foi utilizado o KNN, devido ao fato que esse algoritmo utiliza do cálculo da distância entre dois pontos, visto que a base de dados não é apenas numérica, considerou-se melhor utilizar apenas os algoritmos que processam atributos categóricos de forma direta, sem a necessidade de substituição; desta forma, no lugar do KNN foi utilizado o *Randon Forest*. De acordo com Kubat (2017); Igual & Seguí (2017); e Aggarwal (2015), esses algoritmos podem ser descritos da seguinte forma:

1. Naïve Bayes: é um algoritmo supervisionado de AM baseado no teorema de Bayes e fundamentado no princípio de independência de recurso, que afirma que os recursos de um conjunto de dados não têm relação entre si. Devido a essa suposição de independência, o algoritmo tem essa denominação de ingênuo e é o mais simples de todos os algoritmos de aprendizado de máquina e, no entanto, é muito aplicado por ser eficaz.
2. A Árvore de Decisão é um algoritmo de AM baseado em entropia, o princípio por trás de seu trabalho é que cada atributo no conjunto de dados é tratado como um nó na árvore de decisão. Em cada nó é tomada uma decisão sobre qual caminho escolher na árvore, dependendo do valor do atributo nesse nó específico, o processo continua até que o nó da folha seja alcançado, porque esse contém a decisão final sobre a classificação da instância.
3. *Random Forest*: uma única Árvore de Decisão pode ser enviesada, dependendo dos dados, uma abordagem que pode melhorar essa falha é utilizar várias Árvores de Decisão que fazem sua própria previsão e a previsão final é encontrada calculando a média de todas as previsões feitas por todas as árvores. Essa abordagem é conhecida como *ensemble learning* (aprendizado em conjunto). No aprendizado em conjunto, vários algoritmos de tipos iguais ou diferentes são unidos para criar uma maior capacidade para o modelo de AM. O *Random Forest* é um tipo de modelo de aprendizado em conjunto, esse algoritmo une vários algoritmos de Árvore de Decisão, criando uma floresta.
4. *Suport Vector Machine* (SVM): O algoritmo SVM se originou nos anos 60 e é um dos mais famosos algoritmos de AM, e tem sido muito utilizado desde então, antes das Redes Neurais Artificiais se popularizarem ele era considerado o algoritmo de AM mais preciso. Para classificar uma nova instância, limites de decisão diferentes podem ser utilizados, dessa forma, objetivo do algoritmo SVM é encontrar o limite de decisão que classifica os registros de tal maneira que as chances de a classificação ser incorreta seja minimizada. O algoritmo faz isso maximizando a distância entre os atributos de instâncias mais próximos de todas as classes na base dados, e ele consegue encontrar esse limite com a ajuda de vetores de suporte, por isso o seu nome. Os vetores de suporte passam pelos atributos de dados mais próximos das classes para classificação, o trabalho do algoritmo é maximizar a distância entre esses vetores, traçando uma linha paralela no meio deles, esse limite de decisão é considerado o limite de decisão ideal. Uma das razões, pela qual o SVM é tão amplamente aplicável é que ele pode ser facilmente estendido para bases de dados complexas que não são linearmente separáveis. Isso é feito mapeando os registros de treinamento para um espaço de maior dimensão, onde eles se tornam um conjunto linearmente separável, essa técnica é denominada truque do *kernel* (*kernel trick*).

Dessa forma, ressalta-se que há apoio na literatura da área, visto que a técnica de AM é a mais utilizada (Shahiri, Husain & Rashid, 2015; Souza & Perry, 2020), para classificar esses algoritmos como mais tradicionais, pois são frequentemente empregados na MDE; por isso eles são utilizados neste estudo em comparação as RNAM, para previsão do desempenho dos alunos, oportunizando comparar à técnica de Aprendizagem de Máquina com a técnica de Aprendizagem Profunda. Para realização da aplicação desses algoritmos foram utilizadas bibliotecas específicas do R: Naïve Bayes – bliblioteca "e1071"; Árvores de Decisão – biblioteca "rpart"; *Random Forest* – biblioteca "randomForest"; Suport Vector Machine – biblioteca "e1071"; e para AP foi utilizado o framework "H2O". Cabe destacar que o processo para escolha do framework H2O foi exposto na seção 2.1.

Além das bibliotecas, outra importante questão a ser considerada na aplicação de algoritmos de AM e AP é a configuração de seus parâmetros, elementos que influenciam diretamente na eficácia dos modelos gerados. As configurações feitas nos algoritmos utilizados neste estudo estão sistematizadas na Tabela 2 e todos os detalhes sobre as configurações padrão (default) podem ser analisadas no Apêndice A. Os valores dos parâmetros dos algoritmos *Randon Forest*, SVM e RNAM, diferentes do padrão, foram determinados, por meio de testes e consulta a documentação do R. Tais testes consistiram em definir os valores, rodar o algoritmo e verificar os resultados alcançados, estes foram repetidos até chegar a um conjunto de solução satisfatória, com relação às métricas de avaliação utilizadas; tais métricas são descritas na próxima seção.

Tabela 2: Configuração do Algoritmos.

ALGORITMO	PARÂMETROS CONFIGURADOS
Naïve Bayes	Configuração Default.
Árvore de Decisão	Configuração Default.
Random Forest	Configuração Default, e foi definido uma floresta com 30 árvores.
SVM	Foi definido o kernel "radial" e um valor de custo de 5.0. Os demais parâmetros seguem a configuração Default.
RNAM	Foram definidas 3 camadas ocultas, com 200 neurônios cada, a quantidade de épocas de ajuste foi de 800 e a função de ativação foi a "rectifier". Os demais parâmetros seguem a configuração Default.

Fonte: Autores

4.4 Quarta Fase: Análise dos Resultados

Por fim, no intuito de complementar o processo proposto por Aggarwal (2015), acrescenta-se mais uma fase no processo de mineração de dados que de acordo com o autor depende do propósito do estudo a ser desenvolvido, que se refere à interpretação dos resultados alcançados. Essa etapa nesse estudo pretende avaliar a eficácia na previsão do desempenho dos alunos em cada modelo gerado pelos algoritmos. Nesse sentido, para realizar a verificação dos resultados de um modelo de classificação são necessários dois itens: os métodos de avaliação e as métricas de interpretação. Os dois devem ser aplicados em conjunto para que seja possível observar se um modelo é eficaz ou não. Os métodos indicam como esse modelo será avaliado, e as métricas traduzem os resultados da aplicação desses métodos em números que possam ser interpretados.

Para este estudo o método de avaliação empregado foi o de Treinamento e Teste, em que a base de dados é dividida de forma aleatória em duas porções, uma para treinamento e outra para teste, de acordo com Japkowicz & Shah (2014) geralmente são empregados 85% das instâncias para treinamento e 15% para teste. O algoritmo ao ser aplicado sobre a base de treinamento recolhe informações sobre os atributos das instâncias e gera um modelo de classificação ou regressão com base nesses atributos e informações, após isso esse modelo é aplicado sobre a base de teste (que contém registros diferentes da base de treinamento) e então as métricas de avaliação são calculadas sobre essa aplicação. Apenas a aplicação do método de avaliação não indica se o modelo é eficaz ou não, para isso devem ser utilizadas métricas que possibilitem

interpretação do quanto o modelo foi preciso em suas classificações, em outras palavras quantificar o seu desempenho. Algumas das métricas mais utilizadas no contexto de avaliação de modelos de Aprendizagem Supervisionada estão resumidas na Tabela 3.

Tabela 1: Métricas de Avaliação de Algoritmos de AM e AP.

MÉTRICA	DEFINIÇÃO
Matriz de confusão	Uma matriz de confusão fornece uma análise mais detalhada das classificações corretas e incorretas para cada classe. Uma breve explicação de como interpretar uma matriz de confusão é a seguinte: os elementos da diagonal principal representam o número de pontos para os quais o rótulo previsto é igual ao rótulo verdadeiro, enquanto qualquer coisa fora da diagonal principal foi rotulada incorretamente pelo classificador. Portanto, quanto mais altos os valores presentes na diagonal principal da matriz de confusão, melhor, indicando muitas previsões corretas.
Precisão da classificação (Acurácia)	A precisão é uma métrica de avaliação comum para problemas de classificação. É o número de previsões corretas feitas como uma proporção de todas as previsões realizadas sobre a base de testes. Em outras palavras, é a porcentagem de instâncias classificadas corretamente de todas as instâncias, pode ser considerada mais útil em uma classificação binária do que em problemas de classificação de várias classes, porque pode ser menos claro exatamente como a precisão se divide nessas classes.
Intervalo de Confiança (IC)	Corresponde a uma métrica que indica que há uma probabilidade de 95% que a verdadeira precisão do modelo algorítmico testado esteja dentro desse intervalo.
Taxa de não informação	Essa é a precisão alcançável, sempre prevendo a categoria da classe majoritária. Portanto, corresponde a melhor escolha, sem outras informações.
Valor de P	Consiste em um teste unilateral para verificar se a *precisão* é melhor que a *taxa de não informação*, considerando a maior porcentagem da classe dos dados.
Kappa	Corresponde a uma medida de concordância usada em escalas nominais que fornece uma ideia do quanto as observações se afastam daquelas esperadas, fruto do acaso, indicando assim o quão legítimas as interpretações são. É parecida com a precisão, excetuando por ser normalizada na linha de base do acaso no conjunto de dados. É considerada uma medida mais utilizada para problemas com desequilíbrio nas classes.
Área sob curva (AUC) –Taxas: Sensibilidade e Especificidade	A área sob a curva é uma métrica de desempenho para medir a capacidade de um classificador binário de discriminar entre classes positivas e negativas. Exemplos: 1) Uma área de 1,0 representa um modelo que fez todas as previsões perfeitas; 2) Uma área de 0,5 representa um modelo tão bom quanto aleatório. A AUC pode ser dividida em *Sensibilidade* e *Especificidade*. *Sensibilidade* é a verdadeira taxa positiva, são as instâncias numéricas da classe positiva que realmente foram previstas como positivas. A *Especificidade* é a verdadeira taxa negativa, ou seja, é o número de instâncias da classe negativa que foram realmente previstas como negativa.
Valores Preditivos Positivo e Negativo	*Valor preditivo positivo* – mostra o número da classe positiva prevista corretamente como uma proporção do total de previsões da classe positiva realizadas. *Valor preditivo negativo* – mostra o número da classe negativa prevista corretamente como uma proporção do total de previsões da classe negativa realizadas. Esses parâmetros descrevem o desempenho de um teste de diagnóstico.
Prevalência	Mostra com que frequência a classe positiva realmente ocorre na amostra.
Taxa de detecção	Denota o número de previsões positivas corretas da classe feitas como uma proporção de todas as previsões realizadas.
Prevalência de detecção	Apresenta o número de previsões positivas de classe feitas como uma proporção de todas as previsões realizadas.
Precisão Balanceada	Atribui essencialmente a média das taxas reais positivas e negativas, isto é – (sensibilidade + especificidade)/2.

Fonte: Adaptado Japkowicz & Shah (2014) e Documentação do R.

5 Resultados

Os resultados desse estudo foram sistematizados de acordo com as três questões de pesquisa, e são apresentados na sequência.

5.1 QP1 – Qual a eficácia de modelos gerados a partir de algoritmos baseados em Aprendizagem de Máquina e Aprendizagem Profunda na previsão do desempenho de alunos no ensino tradicional?

Foi gerado um modelo para cada algoritmo utilizado nesse estudo, a partir de sua aplicação na base de dados, salienta-se que para a aplicação dos algoritmos foram utilizados todos os atributos da base de dados descritos na tabela 1, e o atributo meta utilizado foi a nota final classificada em 4 categorias: A, B, C e D. As métricas mais proeminentes para a análise da eficácia dos modelos de classificação gerados foram sistematizadas na Tabela 4. Com esses resultados percebe-se que excluindo o algoritmo Naïve Bayes, que realmente tem uma estrutura bastante simplificada, os demais algoritmos, tiveram bom desempenho com precisão de classificação acima de 80%, em todos os casos, sendo boas opções para a previsão do desempenho de alunos, em bases de dados compostas por registros demográficos, sociais e de rendimento escolar.

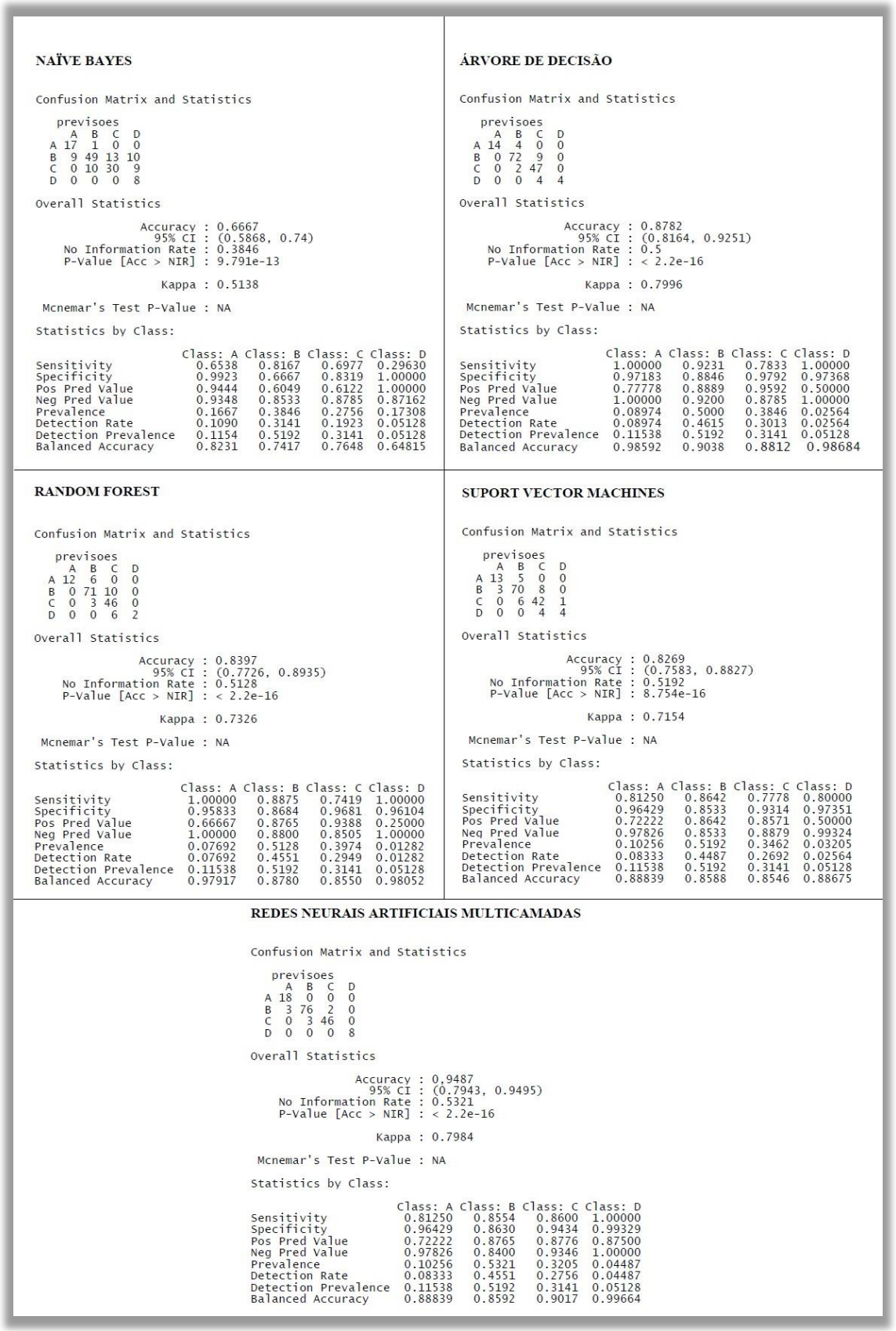
Tabela 4: Análise dos algoritmos.

MÉTRICA	Naïve Bayes	Árvore de Decisão	*Random Forest*	SVM	RNAM
Acurácia	0,66	0,87	0,83	0,82	0,94
Intervalo de confiança de 95%	58-74%	81-92%	77-89%	75-88%	79-94%
Taxa de não informação	0,38	0,5	0,51	0,51	0,53
Valor de p	9.791e-13	2.2e-16	2.2e-16	8.754e-16	2.2e-16
Kappa	0,51	0,79	0,73	0,71	0,79

Fonte: Autores.

No que diz respeito às métricas apresentadas na Tabela 5, a primeira é a acurácia – número de previsões corretas divididas pelo número total de previsões – correspondente a 94% nas RNAM e a 87% no algoritmo de Árvore de Decisão, com um intervalo de confiança de 79-94% e 81-92% respectivamente, o que significa que há uma probabilidade de 95% que a verdadeira precisão desses modelos esteja dentro desse intervalo. Logo após, encontra-se a taxa de não informação que corresponde a 50% para Árvore de Decisão e 53% para as RNAM, essa métrica indica a precisão alcançável sempre prevendo a categoria da classe majoritária. Conforme o valor de p – que é igual para os dois – pode-se afirmar, que os modelos gerados por esses algoritmos, oferecem um desempenho significativamente melhor sobre a taxa de não informação. Na sequência a estatística Kappa, que apresentou valor de 79%, para ambos, mostra quão bem as previsões dos modelos corresponderam às categorias reais da classe, de acordo com as diretrizes propostas por Landis & Koch, (1977) a Kappa nada mais é que uma concordância justa entre o modelo e as verdadeiras categorias de uma classe, uma vez que a precisão aleatória é controlada.

Os demais resultados de todas as métricas descritas na Tabela 3, podem ser visualizados na Figura 5, gerada por meio da aplicação das bibliotecas "*caret'* e "*e1071"* do R, com o método *"confusionMatrix"*. Nesta figura, primeiramente pode ser observada a matriz de confusão de cada algoritmo (previsões); na sequência estão os resultados indicados na Tabela 4; em seguida estão os valores de AUC: Taxas de Sensibilidade e Especificidade; Valores Preditivos Positivo e Negativo; Prevalência; Taxa de detecção; Prevalência de detecção; e Precisão Balanceada.



NAÏVE BAYES

```
Confusion Matrix and Statistics

          previsoes
            A  B  C  D
        A 17  1  0  0
        B  9 49 13 10
        C  0 10 30  9
        D  0  0  0  8

Overall Statistics

                      Accuracy : 0.6667
                        95% CI : (0.5868, 0.74)
           No Information Rate : 0.3846
           P-Value [Acc > NIR] : 9.791e-13

                         Kappa : 0.5138

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D
Sensitivity            0.6538   0.8167   0.6977  0.29630
Specificity            0.9923   0.6667   0.8319  1.00000
Pos Pred Value         0.9444   0.6049   0.6122  1.00000
Neg Pred Value         0.9348   0.8533   0.8785  0.87162
Prevalence             0.1667   0.3846   0.2756  0.17308
Detection Rate         0.1090   0.3141   0.1923  0.05128
Detection Prevalence   0.1154   0.5192   0.3141  0.05128
Balanced Accuracy      0.8231   0.7417   0.7648  0.64815
```

ÁRVORE DE DECISÃO

```
Confusion Matrix and Statistics

          previsoes
            A  B  C  D
        A 14  4  0  0
        B  0 72  9  0
        C  0  2 47  0
        D  0  0  4  4

Overall Statistics

                      Accuracy : 0.8782
                        95% CI : (0.8164, 0.9251)
           No Information Rate : 0.5
           P-Value [Acc > NIR] : < 2.2e-16

                         Kappa : 0.7996

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D
Sensitivity           1.00000   0.9231   0.7833  1.00000
Specificity           0.97183   0.8846   0.9792  0.97368
Pos Pred Value        0.77778   0.8889   0.9592  0.50000
Neg Pred Value        1.00000   0.9200   0.8785  1.00000
Prevalence            0.08974   0.5000   0.3846  0.02564
Detection Rate        0.08974   0.4615   0.3013  0.02564
Detection Prevalence  0.11538   0.5192   0.3141  0.05128
Balanced Accuracy     0.98592   0.9038   0.8812  0.98684
```

RANDOM FOREST

```
Confusion Matrix and Statistics

          previsoes
            A  B  C  D
        A 12  6  0  0
        B  0 71 10  0
        C  0  3 46  0
        D  0  0  6  2

Overall Statistics

                      Accuracy : 0.8397
                        95% CI : (0.7726, 0.8935)
           No Information Rate : 0.5128
           P-Value [Acc > NIR] : < 2.2e-16

                         Kappa : 0.7326

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D
Sensitivity           1.00000   0.8875   0.7419  1.00000
Specificity           0.95833   0.8684   0.9681  0.96104
Pos Pred Value        0.66667   0.8765   0.9388  0.25000
Neg Pred Value        1.00000   0.8800   0.8505  1.00000
Prevalence            0.07692   0.5128   0.3974  0.01282
Detection Rate        0.07692   0.4551   0.2949  0.01282
Detection Prevalence  0.11538   0.5192   0.3141  0.05128
Balanced Accuracy     0.97917   0.8780   0.8550  0.98052
```

SUPORT VECTOR MACHINES

```
Confusion Matrix and Statistics

          previsoes
            A  B  C  D
        A 13  5  0  0
        B  3 70  8  0
        C  0  6 42  1
        D  0  0  4  4

Overall Statistics

                      Accuracy : 0.8269
                        95% CI : (0.7583, 0.8827)
           No Information Rate : 0.5192
           P-Value [Acc > NIR] : 8.754e-16

                         Kappa : 0.7154

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D
Sensitivity           0.81250   0.8642   0.7778  0.80000
Specificity           0.96429   0.8533   0.9314  0.97351
Pos Pred Value        0.72222   0.8642   0.8571  0.50000
Neg Pred Value        0.97826   0.8533   0.8879  0.99324
Prevalence            0.10256   0.5192   0.3462  0.03205
Detection Rate        0.08333   0.4487   0.2692  0.02564
Detection Prevalence  0.11538   0.5192   0.3141  0.05128
Balanced Accuracy     0.88839   0.8588   0.8546  0.88675
```

REDES NEURAIS ARTIFICIAIS MULTICAMADAS

```
Confusion Matrix and Statistics

          previsoes
            A  B  C  D
        A 18  0  0  0
        B  3 76  2  0
        C  0  3 46  0
        D  0  0  0  8

Overall Statistics

                      Accuracy : 0,9487
                        95% CI : (0.7943, 0.9495)
           No Information Rate : 0.5321
           P-Value [Acc > NIR] : < 2.2e-16

                         Kappa : 0.7984

 Mcnemar's Test P-Value : NA

Statistics by Class:

                     Class: A Class: B Class: C Class: D
Sensitivity           0.81250   0.8554   0.8600  1.00000
Specificity           0.96429   0.8630   0.9434  0.99329
Pos Pred Value        0.72222   0.8765   0.8776  0.87500
Neg Pred Value        0.97826   0.8400   0.9346  1.00000
Prevalence            0.10256   0.5321   0.3205  0.04487
Detection Rate        0.08333   0.4551   0.2756  0.04487
Detection Prevalence  0.11538   0.5192   0.3141  0.05128
Balanced Accuracy     0.88839   0.8592   0.9017  0.99664
```

Figura 5: Métricas de Avaliação dos Algoritmos.
Fonte: Autores.

Dentre as métricas apresentadas na Figura 5, destacam-se os valores apresentados da Matriz de Confusão, que é uma das métricas mais utilizadas para analisar a eficácia de algoritmos de classificação, pois ela mostra exatamente o número de classificações corretas e incorretas. Neste

sentido, evidencia-se que o algoritmo Árvore de Decisão classificou corretamente: 14 instâncias da classe A; 72 da classe B; 47 da classe C; e 4 da classe D; errando a classificação de 19 instâncias ao total. Destaca-se também o Algoritmo RNAM que classificou corretamente: 18 instâncias da classe A; 76 da classe B; 46 da classe C; 8 da classe D; e errou a classificação de 8 instancias no total. Dessa forma, observa-se um melhor desempenho das RNAM.

5.2 QP2 – Modelos baseados em algoritmos de *Aprendizagem Profunda* tem uma eficácia superior a modelos baseados em algoritmos tradicionais utilizados na *Mineração de Dados Educacionais*?

De acordo com os resultados da avaliação o modelo de AP obteve uma acurácia superior a dos algoritmos mais tradicionais, em torno de 94%, confirmando estudos como o de Wen *et al.* (2020), Waheed *et al.* (2020) que encontraram valores de eficácia variando de 84% a 93%. Todavia, cabe salientar que esses valores dependem muito de como as bases de dados foram formatadas, assim como do framework e das configurações utilizadas para a aplicação das RNAM, e sendo assim esses resultados não podem ser generalizados de forma abrangente, para outras bases de dados. Ademais, as Árvore de Decisão tiveram uma precisão alta nesse conjunto de dados chegando a 87%, com um Intervalo de confiança de 81-92%, o que devido a sua simplicidade de aplicação e velocidade de processamento pode representar uma alternativa promissora em coleções de registros similares, desde que o estudo desenvolvido não necessite de acurácias muito altas.

Portanto, mesmo que o modelo baseado em AP tenha uma precisão maior, não é muito superior ao algoritmo Árvore de Decisão, no contexto desse estudo, cabe então ao pesquisador decidir. Para essa decisão 4 fatores são determinantes: precisão; simplicidade de configuração; recursos computacionais requeridos; e tempo de processamento. O modelo de AP é mais preciso; mas a tarefa de configuração é mais complexa, pois há mais parâmetros a considerar (consulte Apêndice A), sendo necessário ter conhecimento prévio sobre a teoria de RNAM para realizar essa tarefa de forma satisfatória. Ademais, processamento das RNAM necessita de mais recursos computacionais, como por exemplo memória interna com maior capacidade (RAM), mesmo com treinamento em servidor on-line há consumo de recursos locais, e caso a base de dados seja de grandes dimensões há até mesmo a possibilidade de não ocorrer o processamento.

Quanto ao tempo de processamento, as RNAM por serem um algoritmo mais complexo demoram mais para serem executadas; neste estudo por exemplo foram cronometrados os tempos de treinamento até a geração dos resultados, entre os algoritmos: Árvore de Decisão e RNAM. A Árvore de Decisão gasta poucos segundos pra rodar e gerar os resultados, em torno de 45 segundos. Enquanto a RNAM gasta em torno de 5 minutos, primeiramente o H2O deve solicitar a conexão ao servidor on-line (GPU), posteriormente o algoritmo é treinado até reduzir o erro ao valor definido, ou até totalizar a quantidade de épocas especificadas, e nessa etapa dependendo da quantidade de dados pode demorar vários minutos. Em contra partida, o algoritmo Árvore de Decisão tem uma acurácia menor, por isso cabe ao pesquisador identificar qual algoritmo pode gerar o modelo mais aderente a sua pesquisa.

5.3 QP3 – Qual conjunto de atributos tem mais influência na previsão do desempenho de alunos?

Para definir quais os elementos foram mais influentes na previsão do desempenho, foi gerado o gráfico – Figura 6 – de Árvore de Decisão (com a biblioteca "rpart.plot"), este propicia visualizar quais são os atributos que estão mais no topo da árvore. Tais atributos, devido aos cálculos de entropia realizados para geração da árvore de decisão, são os mais importantes para prever o atributo meta. Nesse sentido, não houve surpresas, ou descobertas, pois de acordo com a Figura 6 os principais atributos para prever o desempenho dos alunos são as notas (G2 na raiz da árvore e nos níveis 2 e 3; e G1 no nível 4), e a quantidade de faltas (*absences* no nível 2), o que é

confirmado pela pesquisa de Cortez & Silva (2008). Dessa forma, demonstrando que os atributos vinculados às atividades escolares são mais preditores para o desempenho que dados de características demográficas e socioeconômicas. Como o corte padrão da Árvore de Decisão considera apenas os atributos mais relevantes para a classificação das instâncias da base de dados, os atributos referentes às características demográficas e socioeconômicas, não aparecem no gráfico da Árvore de Decisão gerada.

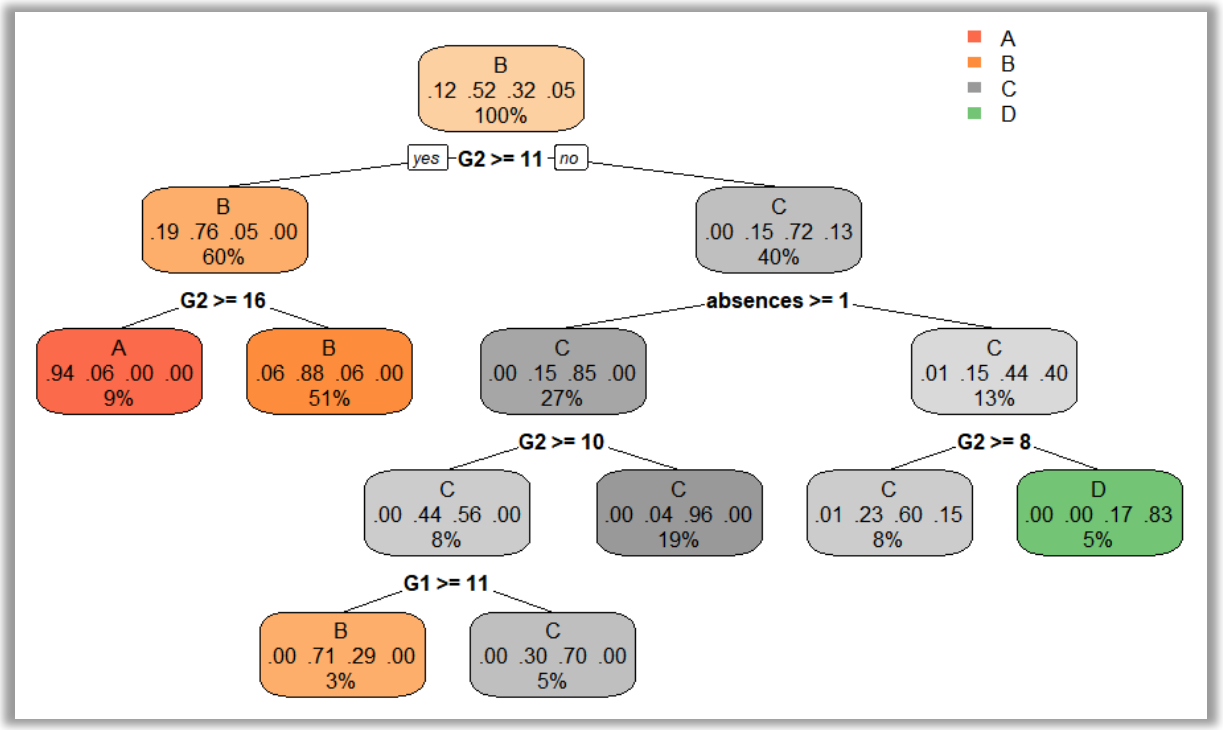


Figura 6: Gráfico de Árvore de Decisão.
Fonte: Autores.

Todavia, não se pode descartar a influência desses elementos no desempenho dos alunos, pois é de conhecimento que estudantes podem ter seu rendimento escolar prejudicado, ou abaixo do esperado, por estarem enfrentando alguma adversidade em casa, o que impacta em suas notas e pode ocasionar uma baixa frequência. Como não há registros de questionários aplicados para entender melhor esses elementos, não há uma confirmação dessas suposições, que são consideradas adequadas, mas não podem ser verificadas. Devido a essa falta de evidências, com base nos indicadores gerados pela aplicação das técnicas de MDE, os principais atributos que influenciam são os relacionados ao desempenho escolar.

6 Conclusão

Este estudo teve como principal objetivo realizar a previsão do desempenho de alunos, utilizando um conjunto de dados público e comparar técnicas já consolidadas no âmbito da MDE, com a técnica de AP. Ademais, foi possível identificar quais são os principais atributos que dão melhor suporte na previsão de desempenho de alunos.

Com relação à previsão de desempenho as técnicas de MDE aplicadas foram adequadas, em que os resultados alcançados são os seguintes: Naïve Bayes com uma acurácia de 66%; Árvore de Decisão com 87%; *Random Forest* com 83%; Suport Vector Machine com 82%; e RNAM com 94% de acurácia. Esses resultados confirmam que a AP aplicada no âmbito da MDE tem apresentado um bom desempenho, com uma eficácia promissora, o que confirma estudos mais

amplos como os desenvolvidos por Wen *et al.* (2020), Waheed *et al.* (2020) e Shahiri, Husain & Rashid (2015).

No estudo de Wen *et al.* (2020), os autores empregaram um modelo de Aprendizagem Profunda, uma Rede Neural Convolucional, para prever a desistência em cursos do tipo MOOC, como salientado na seção trabalhos relacionados. No intuito de verificar a eficácia do modelo desenvolvido os autores realizaram uma comparação com diversos algoritmos tradicionais de AM: Árvore de Decisão (tipo de implementação - CART), Naïve Bayes, *Linear Discriminant Analysis*, Regressão Logística, SVM, *Random Forest* e *Gradient Boosted Decision Tree*. Em todos os 4 experimentos realizados a acurácia da Rede Neural Convolucional foi superior aos algoritmos de AM, permanecendo entre 86% e 89%.

Na pesquisa desenvolvida por Waheed *et al.,* (2020), os autores tinham como objetivo desenvolver um modelo baseado em Aprendizagem Profunda para prever o desempenho acadêmico dos alunos. Para validar seu modelo os autores o compararam com dois algoritmos tradicionais de AM: a Regressão Logística e o SVM. Os resultados apontaram que o modelo de AP obteve uma acurácia entre 84% e 93%, enquanto a Regressão Logística atingiu acurácia entre 79% e 85% e o SVM alcançou acurácia entre 79% e 89%.

Neste sentido, é interessante também citar a revisão sistemática de literatura realizada por Shahiri, Husain & Rashid (2015), os autores forneceram um panorama das principais técnicas de mineração de dados que eram usadas para prever o desempenho dos alunos, em publicações datadas entre 2002 e 2015, os autores destacaram que as Redes Neurais correspondem ao algoritmo que gera os modelos mais eficazes, atingindo altos índices de precisão em torno de 98%.

Entretanto, no estudo de Wen *et al.*(2020), bem como neste estudo, outros algoritmos mais simples, produziram modelos que obtiveram também bons desempenhos, como exemplo cita-se as Árvore de Decisão, que na pesquisa aqui desenvolvida, obteve uma acuraria de 87% e em Wen *et al.*(2020) de 75%. Estes são resultados relevantes para modelos originados de um algoritmo tecnicamente simples e que necessita de pouco esforço para configuração, e sobretudo tem um tempo de processamento bem inferior ao das RNAM. Por isso, é importante destacar que para estudos que não requeiram altos valores de precisão, algoritmos mais simples podem ser boas opções. Nessa perspectiva, Shahiri, Husain & Rashid (2015) também identificaram que o algoritmo Árvore de Decisão é capaz de gerar modelos com alto valor de precisão, por volta de 91%, sendo o segundo algoritmo mais preciso no contexto educacional, logo após as Redes Neurais, mais um indício de que este estudo foi conduzido da forma coerente com a literatura da área.

Corroborando as afirmações quanto à relevância da precisão dos modelos baseados em Árvores de Decisão, Zhang & Wu (2019) afirmaram que tais modelos são consideravelmente simples de serem implementados, e têm precisão relativamente satisfatória, em seu estudo os autores encontraram valores de acurácia variando entre 75% a 81% em modelos baseados em Árvores de Decisão. Por isso, os autores salientaram que estes podem ser empregados para apoiar ações para análise do desempenho de alunos, em diversos cenários educacionais. Outro aspecto importante a ser destacado, é que além de ser mais simples e com desempenho rápido, o algoritmo de Árvore de Decisão permite a visualização gráfica; dessa forma, se o pesquisador possuir o interesse no entendimento dos atributos e suas relações, esse algoritmo é uma boa opção.

Em relação ao conjunto de atributos com mais influência na previsão do desempenho de alunos, não foi identificada uma descoberta relevante, pois de acordo com o gráfico de Árvore de Decisão gerado, os atributos referentes às notas e às faltas dos alunos são os mais preditivos para o desempenho, fato que não provocou surpresa; visto que, além de ser uma conclusão lógica,

no estudo conduzido por Cortez & Silva (2008), como a mesma base de dados, os autores já haviam destacado essa informação. Embora, possa levantar questionamentos sobre o que ocasionou um desempenho abaixo do esperado em alguns alunos. Essa constatação é confirmada também em outros estudos, como os revisados por Shahiri, Husain & Rashid (2015) que salientaram que os principais atributos utilizados para a previsão do desempenho de alunos foi a média cumulativa de notas e a avaliação interna, ou seja variáveis relacionados as notas dos alunos, usadas por 10 dos 30 artigos selecionados para a revisão desenvolvida pelos autores.

Por fim, esta pesquisa apresenta como principal contribuição demonstrar a aplicação do processo de MDE, em um conjunto de dados público, que pode ser replicado por outros pesquisadores, com um nível de detalhe pouco encontrado em textos dessa área. Além disso, os resultados das avaliações dos algoritmos exibidos, podem dar suporte na escolha de métodos mais eficazes para a aplicação em conjuntos de dados educacionais, levando em consideração a Aprendizagem Profunda, que sem dúvida é uma técnica bastante relevante no contexto da Ciência de Dados.

Referências

Aggarwal, C. C. (2015). Data Mining: The Textbook. 1. ed. New York, USA: Springer. E-book. doi: 10.1007/978-3-319-14142-8

Aggarwal, C. C. (2018). Neural Networks and *Aprendizagem Profunda*: A Textbook. 1. ed. New York, USA: Springer, 2018. E-book. doi: 10.1007/978-3-319-94463-0

Alvim, P. (2010). Open Source com jCompany© Developer Suite. 3a Ed. ed. Belo Horizonte: E-book. [GS Search]

Badar, M., Haris, M., & Fatima, A. (2020). Application of *Aprendizagem Profunda* for retinal image analysis: A review. *Computer Science Review*, 35, 1–18. doi: 10.1016/j.cosrev.2019.100203 [GS Search]

Bahrampour, S., *et al.* (2015). Comparative Study of *Aprendizagem Profunda* Software Frameworks. *Cornell Univeristy,* 3, 1–9, 2015. [GS Search]

Baker, R., Isotani, S., & Carvalho, A. (2011). Mineração de Dados Educacionais: Oportunidades para o Brasil. *Revista Brasileira de Informática na Educação*, 19, 02, 3–13. doi: 10.5753/rbie.2011.19.02.03 [GS Search]

Baker, R. S. J. D. (2015). Big data and education. 2. ed. New York, USA: A Massive Online Open Textbook (MOOT) - Teachers College, Columbia University. [GS Search]

Baker, R. S., & Inventado, P. S. (2014). Educational Data Mining and Learning Analytics. In: J.A. Larusson and B. White (EDS.) (org.). Learning Analytics: From Research to Practice. 1. ed. New York, USA: Springer, 1–195. E-book. doi: 10.1007/978-1-4614-3305-7 [GS Search]

Bishop, C. M. (1995). Neural networks for pattern recognition. 1. ed. EUA. E-book. [GS Search]

Boulemtafes, A., Derhab, A., & Challal, Y. (2020). A review of privacy-preserving techniques for *Aprendizagem Profunda*. *Neurocomputing*, 384, 21–45. doi: 10.1016/j.neucom.2019.11.041 [GS Search]

Cortez, P., & Silva, A. (2008). Using Data Mining to Predict Secondary School Student Performance. In *A. Brito and J. Teixeira Eds., Proceedings of 5th FUture BUsiness TEChnology Conference (FUBUTEC 2008)*. [GS Search]

De Los Reyes, D. A. G. *et al.* (2019). Predição de sucesso acadêmico de estudantes: uma análise sobre a demanda por uma abordagem baseada em transfer learning. *Revista Brasileira de Informática na Educação,* 27, 1, 1–25. doi: 10.5753/rbie.2019.27.01.01 [GS Search]

De Souza, V. F., & Perry, G. T. (2020). Tendências de Pesquisas em Mineração de Dados Educacionais em MOOCs: um Mapeamento Sistemático. Revista Brasileira de Informática na Educação, 28, 491-508. doi: 10.5753/rbie.2020.28.0.491 [GS Search]

MDE. Sociedade Internacional de *Mineração de Dados Educacionais*. (2020). Disponível em: http://educationaldatamining.org/. Acesso em: 31 jan. 2021.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Aprendizagem Profunda*. Cambridge, MA, USA, 2016. E-book. [GS Search]

Guo, S. X., *et al.* (2019). Attention-Based Character-Word Hybrid Neural Networks With Semantic and Structural Information for Identifying of Urgent Posts in MOOC Discussion Forums. *IEEE Access*, 7, 120522–120532. doi: 10.1109/ACCESS.2019.2929211 [GS Search]

Hand, D. J. (1997). Construction and Assessment of Classification Rules. 1. ed. New York. E-book. [GS Search]

Hinton, G. E., Osindero, S., & Teh, Y. W. (2006). A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18, 7, 1527–1554. doi: 10.1162/neco.2006.18.7.1527 [GS Search]

Igual, L., & Seguí, S. (2017). Introduction to Data Science: A Python Approach to Concepts, Techniques and Applications. 1. Ed. Springer. E-book. doi: 10.1007/978-3-319-50017-1

Japkowicz, N., & Shah, M. (2014). Evaluating Learning Algorithms: A Classification Perspective. 1a Ed. ed. Cambridge, E-book. [GS Search]

Kovalev, V., Kalinovsky, A., & Kovalev, S. (2016). *Aprendizagem Profunda* with Theano, Torch, Caffe, TensorFlow, and Deeplearning4J: Which One Is the Best in Speed and Accuracy? In: *13th International Conference on Pattern Recognition and Information Processing (PRIP 2016)*, 99–103. [GS Search]

Kubat, M. (2017). An Introduction to Aprendizagem de Máquina. 2. ed. *Coral Gables, FL, USA: Springer*. E-book. doi: 10.1007/978-3-319-63913-0

Landis, J. R., & Koch, G. G. (1977). An Application of Hierarchical Kappa-type Statistics in the Assessment of Majority Agreement among Multiple Observers. *Biometrics*, 33, 2, 363–374, 1977. doi: 10.2307/2529786 [GS Search]

Le, Q., Torrisi, M., & Pollastri, G. (2020). *Aprendizagem Profunda* methods in protein structure prediction. *Computational and Structural Biotechnology Journal*, 426, 1–10. doi: 10.1016/j.csbj.2019.12.011 [GS Search]

Lecun, Y., Bengio, Y., & Hinton, G. (2015). *Aprendizagem Profunda*. *Nature*, 521, 7553, 436–444. doi: 10.1038/nature14539 [GS Search]

Lin, J., *et al.* (2019). Automatic Knowledge Discovery in Lecturing Videos via Deep Representation. *IEEE Access*, 7, 33957–33963. doi: 10.1109/ACCESS.2019.2904046 [GS Search]

MURAT, F., *et al.* (2020). Application of *Aprendizagem Profunda* techniques for heartbeats detection using ECG signals-analysis and review. *Computers in Biology and Medicine*, 120, 1–14. doi: 10.1016/j.compbiomed.2020.103726 [GS Search]

NG, S. S. Y. *et al.* (2016). An independent study of two *Aprendizagem Profunda* platforms - H2O and SINGA. In: *2016, Bali, Indonesia. International Conference on Industrial Engineering and Engineering Management (IEEM 2016)*. Bali, Indonesia: 1279–1283. doi: 10.1109/IEEM.2016.7798084 [GS Search]

Rigo, S. J. *et al.* (2014). Aplicações de Mineração de Dados Educacionais e Learning Analytics com foco na evasão escolar: oportunidades e desafios. *Revista Brasileira de Informática na Educação*, 22, 01, 168–177. doi: 10.5753/RBIE.2014.22.01.132 [GS Search]

Ripley, B. D. (1996). Pattern recognition and neural networks. 1. ed. Cambridge, E-book. [GS Search]

Romero, C., & Ventura, S. (2013). Data mining in education. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 3, 1, 12–27. doi: 10.1002/widm.1075 [GS Search]

Romero, C., & Ventura, S. (2020). Educational data mining and learning analytics: An updated survey. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 10, 3, 1–21. doi: 10.1002/widm.1355 [GS Search]

Schmidhuber, J. (2015). *Aprendizagem Profunda* in neural networks: An overview. *Neural Networks,* 61, 85–117. doi: 10.1016/j.neunet.2014.09.003 [GS Search]

Sengupta, S. *et al.* (2020). Ophthalmic diagnosis using *Aprendizagem Profunda* with fundus images – A critical review. *Artificial Intelligence in Medicine*, 102, 1–36. doi: 10.1016/j.artmed.2019.101758 [GS Search]

Sezer, O. B., Gudelek, M. U., & Ozbayoglu, A. M. (2020). Financial time series forecasting with *Aprendizagem Profunda*: A systematic literature review: 2005–2019. *Applied Soft Computing Journal*, 90, 1–65, 2020. doi: 10.1016/j.asoc.2020.106181 [GS Search]

Shahiri, A. M., Husain, W., & Rashid, N. A. (2015). A Review on Predicting Student's Performance Using Data Mining Techniques. *Procedia Computer Science*, 72, 414–422. doi: 10.1016/j.procs.2015.12.157 [GS Search]

Soffer, S., *et al.* (2019). Convolutional Neural Networks for Radiologic Images: A Radiologist's Guide. *Radiology*, 290, 3, 590–606. doi: 10.1148/radiol.2018180547 [GS Search]

Waheed, H., *et al.* (2020). Predicting academic performance of students from VLE big data using *Aprendizagem Profunda* models. *Computers in Human Behavior*, 104, 1–13, 2020. doi: 10.1016/j.chb.2019.106189 [GS Search]

Wen, Y., *et al.* (2020). Consideration of the local correlation of learning behaviors to predict dropouts from MOOCs. *Tsinghua Science and Technology*, 25, 3, 336–347. doi: 10.26599/TST.2019.9010013 [GS Search]

Xin, Y., *et al.* (2018). Aprendizagem de Máquina and *Aprendizagem Profunda* Methods for Cybersecurity. *IEEE Access,* 20, 1–9. doi: 10.1109/ACCESS.2018.2836950 [GS Search]

Yang, J., Zhang, X. L., & Su, P. (2018). Deep-Learning-Based Agile Teaching Framework of Software Development Courses in Computer Science Education. *Procedia Computer Science*, 154, 137–145, 2018. doi: 10.1016/j.procs.2019.06.021 [GS Search]

Zhang, Y., & Wu, B. (2019). Research and application of grade prediction model based on decision tree algorithm. In: 2019, Chengdu, China. Turing Celebration Conference (ACM TURC 2019). Chengdu, China: ACM, 1–6. doi: 10.1145/3321408.3322857 [GS Search]

Zhao, Rui *et al.* (2019). *Aprendizagem Profunda* and its applications to machine health monitoring. *Mechanical Systems and Signal Processing*, 115, 213–237. doi: 10.1016/j.ymssp.2018.05.050 [GS Search]

Agradecimentos

O presente trabalho foi realizado com apoio do Instituto Federal de Educação, Ciência e Tecnologia do Rio Grande do Sul (IFRS) e do Instituto Federal de Educação, Ciência e Tecnologia de Mato Grosso do Sul (IFMS).

Apêndice A – Descrição das Configurações de Parâmetros Default dos Algoritmos Utilizados

---

CLASSIFICADOR NAÏVE BAYES

Biblioteca: e1071 – Método: naiveBayes
Descrição: Calcula as probabilidades a-posteriores condicionais de uma variável de classe categórica dadas variáveis preditoras independentes usando a regra de Bayes. **Detalhes**: O classificador Naïve Bayes padrão (pelo menos esta implementação) assume independência das variáveis preditoras e distribuição Gaussiana (dada a classe de destino) dos preditores métricos. Para atributos com valores ausentes, as entradas de tabela correspondentes são omitidas para previsão.

Aplicação

```
## S3 method for class 'formula'
naiveBayes(formula, data, laplace = 0, ..., subset, na.action = na.pass)
## Default S3 method:
naiveBayes(x, y, laplace = 0, ...)
## S3 method for class 'naiveBayes'
predict(object, newdata,type = c("class", "raw"), threshold = 0.001, eps = 0)
```

Para conhecer todas as descrições dos parâmetros default do classificador Naïve Bayes, acesse: https://cran.r-project.org/web/packages/e1071/e1071.pdf

---

ÁRVORE DE DECISÃO

Biblioteca: rpart– Método: rpart

Aplicação

```
rpart(formula, data, weights, subset, na.action = na.rpart, method, model =
       FALSE, x = FALSE, y = TRUE, parms, control, cost)
```

Para conhecer todas as descrições dos parâmetros default do algoritmo Árvore de Decisão, acesse: https://cran.r-project.org/web/packages/rpart/rpart.pdf

---

RANDON FOREST

Biblioteca: randomForest – Método: randomForest
Descrição: randomForest padrão do R implementa o algoritmo de floresta aleatória de Breiman (baseado no código Fortran original de Breiman e Cutler) para classificação e regressão. Também pode ser usado no modo não supervisionado para avaliar proximidades entre pontos de dados.

Aplicação

```
## S3 method for class 'formula'
randomForest(formula, data=NULL, ..., subset, na.action=na.fail)
## Default S3 method:
randomForest(x,   y=NULL,   xtest=NULL,   ytest=NULL,   ntree=500,   mtry=if
             (!is.null(y) && !is.factor(y)) max(floor(ncol(x)/3), 1) else
             floor(sqrt(ncol(x))),   replace=TRUE,   classwt=NULL,   cutoff,
             strata,    sampsize    =    if    (replace)    nrow(x)    else
             ceiling(.632*nrow(x)),   nodesize   =   if   (!is.null(y)   &&
             !is.factor(y)) 5 else 1, maxnodes = NULL, importance=FALSE,
             localImp=FALSE,   nPerm=1,   proximity,   oob.prox=proximity,
             norm.votes=TRUE,  do.trace=FALSE,  keep.forest=!is.null(y)  &&
             is.null(xtest), corr.bias=FALSE, keep.inbag=FALSE)
## S3 method for class 'randomForest'
print(x, ...)
```

Para conhecer todas as descrições dos parâmetros default do algoritmo Random Forest, acesse: https://cran.r-project.org/web/packages/randomForest/randomForest.pdf

---

SUPORT VECTOR MACHINES

Biblioteca: e1071 – Método: SVM
Descrição: O SVM é usado para treinar uma máquina de vetores de suporte. Ele pode ser usado para realizar regressões e classificações gerais (do tipo nu e épsilon), bem como estimativas de densidade.

Aplicação

```
## S3 method for class 'formula'
svm(formula, data = NULL, ..., subset, na.action = na.omit, scale = TRUE)
## Default S3 method:
svm(x, y = NULL, scale = TRUE, type = NULL, kernel = "radial", degree = 3,
    gamma = if (is.vector(x)) 1 else 1 / ncol(x), coef0 = 0, cost = 1, nu =
    0.5, class.weights = NULL, cachesize = 40, tolerance = 0.001, epsilon =
    0.1, shrinking = TRUE, cross = 0, probability = FALSE, fitted = TRUE,
    ..., subset, na.action = na.omit)
```

Para conhecer todas as descrições dos parâmetros default do algoritmo SVM, acesse: https://cran.r-project.org/web/packages/e1071/e1071.pdf

---

REDES NEURAIS ARTIFICIAIS MULTICAMADAS

Biblioteca: h2o – Método: h2o.deeplearning
Descrição: constrói uma rede neural artificial multicamadas feed-forward em um H2OFrame.

Aplicação

```
h2o.deeplearning(x, y, training_frame, model_id = NULL, validation_frame =
                 NULL, nfolds = 0, keep_cross_validation_models = TRUE,
                 keep_cross_validation_predictions = FALSE,
                 keep_cross_validation_fold_assignment = FALSE,
                 fold_assignment = c("AUTO", "Random", "Modulo",
                 "Stratified"), fold_column = NULL, ignore_const_cols = TRUE,
                 score_each_iteration = FALSE, weights_column = NULL,
                 offset_column = NULL, balance_classes = FALSE,
                 class_sampling_factors = NULL, max_after_balance_size = 5,
                 checkpoint = NULL, pretrained_autoencoder = NULL,
                 overwrite_with_best_model = TRUE, use_all_factor_levels =
                 TRUE, standardize = TRUE, activation = c("Tanh",
                 "TanhWithDropout", "Rectifier", "RectifierWithDropout",
                 "Maxout", "MaxoutWithDropout"), hidden = c(200, 200), epochs
                 = 10, train_samples_per_iteration = -2,
                 target_ratio_comm_to_comp = 0.05, seed = -1, adaptive_rate =
                 TRUE, rho = 0.99, epsilon = 1e-08, rate = 0.005,
                 rate_annealing = 1e-06, rate_decay = 1, momentum_start = 0,
                 momentum_ramp = 1e+06, momentum_stable = 0,
                 nesterov_accelerated_gradient = TRUE, input_dropout_ratio =
                 0, hidden_dropout_ratios = NULL, l1 = 0, l2 = 0, max_w2 =
                 3.4028235e+38, initial_weight_distribution =
                 c("UniformAdaptive", "Uniform", "Normal"),
                 initial_weight_scale = 1, initial_weights = NULL,
                 initial_biases = NULL, loss = c("Automatic", "CrossEntropy",
                 "Quadratic", "Huber", "Absolute", "Quantile"), distribution
                 = c("AUTO", "bernoulli", "multinomial", "gaussian",
                 "poisson", "gamma", "tweedie", "laplace", "quantile",
                 "huber"), quantile_alpha = 0.5, tweedie_power = 1.5,
                 huber_alpha = 0.9, score_interval = 5, score_training_samples
                 = 10000, score_validation_samples = 0, score_duty_cycle =
                 0.1, classification_stop = 0, regression_stop = 1e-06,
                 stopping_rounds = 5, stopping_metric = c("AUTO", "deviance",
                 "logloss", "MSE", "RMSE", "MAE", "RMSLE", "AUC", "AUCPR",
                 "lift_top_group", "misclassification",
                 "mean_per_class_error", "custom", "custom_increasing"),
                 stopping_tolerance = 0, max_runtime_secs = 0,
                 score_validation_sampling = c("Uniform", "Stratified"),
                 diagnostics = TRUE, fast_mode = TRUE, force_load_balance =
                 TRUE, variable_importances = TRUE, replicate_training_data =
                 TRUE, single_node_mode = FALSE, shuffle_training_data =
                 FALSE, missing_values_handling = c("MeanImputation",
                 "Skip"), quiet_mode = FALSE, autoencoder = FALSE, sparse =
                 FALSE, col_major = FALSE, average_activation = 0,
                 sparsity_beta = 0, max_categorical_features = 2147483647,
                 reproducible = FALSE, export_weights_and_biases = FALSE,
                 mini_batch_size = 1, categorical_encoding = c("AUTO", "Enum",
                 "OneHotInternal", "OneHotExplicit", "Binary", "Eigen",
                 "LabelEncoder", "SortByResponse", "EnumLimited"),
                 elastic_averaging = FALSE, elastic_averaging_moving_rate =
                 0.9, elastic_averaging_regularization = 0.001,
                 export_checkpoints_dir = NULL, verbose = FALSE)
```

Para conhecer todas as descrições dos parâmetros default do algoritmo RNAM (h2o. deeplearning), acesse: https://cran.r-project.org/web/packages/h2o/h2o.pdf