Revista Brasileira de Computação Aplicada, Abril, 2024

DOI: 10.5335/rbca.v16i1.14456
Vol. 16, Nº 1, pp. 26–37
Homepage: seer.upf.br/index.php/rbca/index

ARTIGO ORIGINAL

Uso de técnicas de aprendizado de máquina para predição do tempo de graduação dos discentes de Engenharia da Computação na região Sudeste do Brasil

Use of machine learning techniques to predict the graduation time of Computer Engineering students in the Southeast region of Brazil

Bruno da Silva Macedo [1], and Camila Martins Saporetti [2]

[1]Universidade Federal de Lavras, [2]Instituto Politécnico/Universidade do Estado do Rio de Janeiro

*bruno.macedo2@estudante.ufla.br; camila.saporetti@iprj.uerj.br

Recebido: 20/03/2023. **Revisado:** 25/10/2023. **Aceito:** 11/03/2024.

Resumo

O Exame Nacional de Desempenho de Estudantes (ENADE) foi criado para avaliar o rendimento dos estudantes nos cursos das instituições superiores. Através do desempenho dos estudantes estima a qualidade dos cursos. O abandono ou atraso do curso acarreta uma ruim gestão universitária, já que o orçamento que as graduações recebem tem como fator o número de alunos formandos. Analisar dados do ENADE pode gerar *insights* sobre o tempo que os discentes permanecem na graduação. Como os dados do ENADE contém um número elevado de informações, realizar análises visualmente é algo inviável. Para contornar essa situação, técnicas de aprendizado de máquina podem ser utilizadas com intuito de automatizar essa tarefa e apresentar os resultados. Nesse contexto, o objetivo deste trabalho é determinar, através da base do ENADE 2019, o tempo de permanência dos estudantes na graduação, tendo em vista os cursos de Engenharia da Computação na região Sudeste do Brasil. A metodologia envole o pré-processamento, a seleção de características, balanceamento dos dados, abordagem de seleção de parâmetros *Grid-Search*, validação cruzada e classificação. Os resultados mostram que o *Random Forest* obteve acurária de 83.3% nos experimentos realizados que a aplicação do SMOTE para balanceamento dos dados se faz necessária.

Palavras-Chave: Engenharia da Computação; ENADE; *Grid-Search*; Aprendizagem de Máquina

Abstract

The National Student Performance Examination (ENADE) was created to assess student performance in courses at higher institutions. Through the performance of the students, it estimates the quality of the courses. Leaving or delaying the course leads to poor university management, since the budget that graduations receive is based on the number of graduating students. Analyzing ENADE data can generate insights into how long students remain in graduation. As the ENADE data contains a large amount of information, performing analysis visually is impracticable. To work around this situation, machine learning techniques can be used in order to automate this task and present the results. In this context, the objective of this work is to determine, through the ENADE 2019 database, the length of stay of students in graduation, considering Computer Engineering courses in the Southeast region of Brazil. The methodology involves pre-processing, feature selection, data balancing, Grid-Search parameter selection approach, cross-validation and classification. The results show that Random Forest obtained an accuracy of 83.3% in the experiments carried out and that the application of SMOTE for data balancing is necessary.

Keywords: Computer Engineering; ENADE; Grid-Search; Machine Learning

1 Introdução

O Exame Nacional de Desempenho de Estudantes (ENADE) foi criado pela Lei 10.861/2004 com o intuito de avaliar o rendimento dos estudantes nos cursos das instituições superiores. Através do desempenho dos estudantes ele estima a qualidade dos cursos (Feldmann e Souza, 2016).

O ENADE é aplicado pelo Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira (INEP), onde todos os estudantes selecionados devem prestar o exame. Todavia, senão puderem realizar o exame, um documento solicitado através do Ministério da Educação (MEC) deverá ser apresentado. Ademais, o ENADE é uma ferramenta importante para avaliar a qualidade do ensino superior no Brasil, como vem sendo conduzido o processo de aprendizagem dos discentes, suas habilidades e competências. Dessa forma, o Estado tem uma visão do que vem sendo feito e as diretrizes traçadas para que os resultados sejam melhorados (Feldmann e Souza, 2016).

Um fator que acarreta uma ruim gestão universitária é o abandono dos estudantes ou atraso na graduação, uma vez que o orçamento que os cursos de graduação das universidades recebem tem como fator o número de alunos que estão se graduando. O abandono dos estudantes onera o cofre público, pois todo dinheiro investido na faculdade é com base na quantidade de estudantes que os cursos comportam. Já o atraso na graduação exige mais investimentos porque toda estrutura, tecnologia e recursos devem ser mantidos para atender os estudantes que permanecem além do tempo (Oliveira e Barbosa, 2016).

Diversos fatores influenciam no abandono e no atraso na graduação, alguns desses fatores são a condição financeira, estrutura familiar, idade, personalidade, motivação pessoal, habilidades como estudante, evolução escolar, planejamento pessoal e profissional, entre outros fatores (Oliveira e Barbosa, 2016).

Percebe-se que analisar dados do ENADE pode ser uma forma de ter *insights* a respeito do tempo que os discentes permanecem na graduação. Dessa forma, as instituições de ensino superior, juntamente com o MEC, poderão avaliar e tomar medidas para que o rendimento dos discentes nas instituições sejam melhorados.

Como a base de dados do ENADE contém um número elevado de informações, como também é dividida em vários arquivos, realizar análises de forma visual é algo que se torna inviável. Para contornar essa situação, técnicas de aprendizado de máquina podem ser utilizadas com intuito de automatizar essa tarefa e apresentar os resultados.

Nesse contexto, o objetivo deste trabalho é determinar, através da base do ENADE 2019, o tempo de permanência dos estudantes na graduação, tendo em vista os cursos de Engenharia da Computação na região Sudeste do Brasil. Tais análises serão realizadas por meio de métodos de aprendizado de máquina.

O restante do artigo está organizado da seguinte maneira: na Seção 2 são discutidos os trabalhos relacionados. A Seção 3 apresenta a base de dados utilizada e a metodologia empregada, descrevendo pré-processamento, seleção de características, balanceamento, métodos de classificação, validação cruzada, *Grid-Search* e métricas para desempenho de classificação. Os resultados são abordados na Seção 4, e por fim, na Seção 5 a conclusão.

2 Trabalhos Relacionados

A identificação do tempo de permanência no ensino superior é um tema que vem sendo estudado por diversos pesquisadores. Nesses trabalhos, análises são realizadas e ferramentas propostas no contexto de aprendizado de máquina para compreender o problema e buscar soluções para predizer os resultados.

Peixoto et al. (2012) fez um estudo sobre aspectos relacionados a permanência de graduandos e pós-graduandos em disciplinas semipresenciais, na Faculdade de Ciências da Saúde da Universidade de Brasília (UnB). Neste estudo somente 67.3% participaram do estudo. Os fatores que facilitaram a permanência nas disciplinas foram o ambiente na qual o estudante estuda e a interação com os aspectos das disciplinas.

No trabalho de Manhães et al. (2012) foi utilizada técnicas de Mineração de Dados para identificar os estudantes que terminaram o curso, alunos que não terminaram o curso e os que permaneceram além do prazo médio para terminar a graduação. O algoritmo utilizado foi o *Naive Bayes* (NB), no qual obteve uma precisão de 80%.

Rolim e Silva (2021) realizou uma análise sobre o perfil de retenção no curso de graduação em Ciência da Computação da Unioeste, campus de Foz do Iguaçu. Foram utilizados os algoritmos *K-Nearest Neighbors* (KNN) e *Support Vector Machine* (SVM). Com os parâmetros e métricas iniciais, os algoritmos apresentaram as acurácias acima de 92%. Já com os parâmetros e métricas melhoradas as acurácias foram acima de 94%.

Góes (2022) apresentou uma análise sobre o tempo de permanência dos discentes nos cursos da Universidade Federal da Bahia (UFBA). Foram utilizadas os dados do Censo do Ensino Superior (CENSUP) de 2015 a 2019. A metodologia utilizada foi a de eventos competitivos, onde percebeu que estudantes do sexo feminino possuem uma chance 2.22 vezes maior de formar em um tempo menor. Outra percepção foi de que se o estudante realizar alguma atividade extracurricular, o risco de formar em menor tempo aumenta em 10.30 vezes.

No estudo de Silva et al. (2022), foi realizado uma análise de retenção dos estudantes de graduação na Escola Politécnica de Pernambuco (POLI)/ Universidade de Pernambuco (UPE), utilizando os dados do ENADE de 2010 a 2012. Os algoritmos de Aprendizagem de Máquina utilizados foram Regressão Logística (RL), *Random Forest* (RF) e *MultiLayer Perceptron* (MLP). Os algoritmos mostraram acima de 80% de acurácia em pelo menos uma configuração de base.

Os artigos encontrados não utilizam uma abordagem que trate o desbalanceamento dos dados, em muitos casos os parâmetros dos métodos utilizados não são apresentados o que dificulta a avaliação da metodologia proposta e a veracidade dos resultados apresentados. Ademais, o curso de Engenharia da Computação não foi tratado de forma particular. Dessa forma, nota-se que há uma lacuna para a criação de uma metodologia que identifique o tempo de formação desses alunos de forma automática em que é realizado uma busca pelo melhor modelo.

3 Metodologia

Nesta seção estão descritas as etapas realizadas para alcançar o resultado da pesquisa, como a base de dados utilizada, o pré-processamento, a seleção de características e o balanceamento realizado. Além disso, os algoritmos de classificação empregados, a abordagem de seleção de parâmetros *Grid-Search*, a técnica de validação cruzada e as métricas de avaliação de desempenho.

Na metodologia proposta optou-se por utilizar dois métodos de balanceamento SMOTE e Near Miss. A escolha destes métodos se deu pelo fato de possuírem duas abordagens diferentes *oversampling* e *undersampling*, além de serem utilizados em diversos trabalhos(Alghamdi et al., 2017; Sarkar et al., 2019; Wang et al., 2019; Casuat et al., 2020; Gunturi e Sarkar, 2021; Mqadi et al., 2021; Oladunni et al., 2021; Rattan et al., 2021).

Os classificadores utilizados são baseados em árvore de decisão. Escolheu-se além do método clássico, alguns *ensembles* que possuem comportamentos diferentes. São métodos indicados em vários estudos por obterem uma acurácia maior dentre os comparados: AdaBoost (Chengsheng et al., 2017; Hatwell et al., 2020; Wu et al., 2020), Decision Tree (Ekle et al., 2023; Korchi et al., 2023; Wandani et al., 2023), Extreme Gradient Boosting (Fatty et al., 2023; Matsumoto et al., 2023; Shi et al., 2023) e Random Forest (Lai et al., 2023; Song et al., 2023; Vidal Bezerra et al., 2023).

3.1 Base de Dados

Para realizar as análises foi utilizado a base de dados do ENADE 2019. A base possui 433.930 amostras e 133 atributos, como código do curso, sexo, estado civil, cor/raça, nacionalidade, renda total da família, tipo de escola no ensino médio, ano de ingresso do estudante na graduação, ano que realizou a prova do ENADE, código do município, entre outros.

Inicialmente utilizou-se os dados do CENSUP para encontrar os códigos dos cursos de Engenharia da Computação na região Sudeste. A partir desses códigos pode-se fazer a seleção dos discentes que fizeram o ENADE.

3.2 Pré-Processamento

No pré-processamento foram removidos todos os dados de discentes que não pertencem aos cursos de Engenharia da Computação da região Sudeste do Brasil. Dessa forma, a base de dados do ENADE 2019 ficou com 1.928 amostras. Além disso, as variáveis que foram julgadas que não possuem influência nas análises foram excluídas. Outro tratamento realizado foi a transformação de algumas variáveis, utilizando o método de *Label Encoder* para transformação de categórica para numérica. Por exemplo, o atributo sexo que tem como possibilidades M (Masculino) e F (Feminino) foi transformado para 1 e 0, respectivamente. Outra abordagem utilizada foi o *Ordinal Encoder* para transformação numérica, mas preservando a ordem dos dados categóricos. Como exemplo, o atributo QE_I107 que é a respeito da renda familiar, na qual tem as categorias A a E, conforme o total o valor recebido pelas pessoas que compõe a família. Essas categorias foram modificadas para 1, 2, 3, 4 e 5, respectivamente.

Foi adicionado duas colunas: uma de tempo na base, em que esse atributo contém a diferença do ano do ENADE pelo ano de ingresso do estudante na graduação e uma com as classes criadas conforme o tempo de formação. A definição das classes em relação ao tempo pode ser observada na Tabela 1.

Tabela 1: Classes em relação ao tempo.

Classe	Tempo
1	< 5
2	≥ 5 e ≤ 7
3	> 7

3.3 Seleção de Características

Reduzir a dimensão dos atributos é uma das maneiras de remover informações que não são fundamentais, de forma que aumente a precisão da aprendizagem e se tenha um melhor entendimento do resultado. Selecionar os atributos é uma das formas mais simples de reduzir o tamanho de um conjunto de dados, pelo fato de somente um número pequeno, comparado ao original, de características são selecionadas no conjunto de dados (Almeida et al., 2021).

O método utilizado para selecionar as variáveis é o *Select K-Best* que é um método que emprega a estratégia de seleção univariada, que por meio de testes estatísticos faz a verificação do relacionamento de uma variável alvo com cada variável. O *Chi-square* (chi^2) foi o teste estatístico univariado empregado. Outro atributo passado como parâmetro é um valor K que é o número de características a serem selecionadas. A técnica realiza uma pontuação no cálculo de cada atributo e na remoção de cada atributo também, exceto os com maior pontuação até o limite de K (Almeida et al., 2021). O (chi^2) é calculado pela Eq. (1):

$$x^2 = \frac{(O - E)^2}{E} \tag{1}$$

na qual O representa a frequência notada e E representa a frequência de classe esperada, se não houver relação entre o atributo e o atributo de destino. Os atributos são independentes se a frequência analisada for próxima da frequência notada. As variáveis são selecionadas para o treinamento do algoritmo com base na maior pontuação da variável com a variável alvo, indicando forte relação (Almeida et al., 2021).

3.4 Balanceamento

Um dos problemas mais comuns em bases de dados é o desbalanceamento, que se dá quando uma classe contém um número maior de amostras quando comparado com outras classes. A Tabela 2 mostra o número de amostras para cada classe, onde é possível observar que a classe 1 possui um número bem maior de amostras se comparado as outras duas classes. O desbalanceamento afeta o treinamento dos métodos, uma vez que vão tender a aprender mais com a classe que possui um número maior de amostras.

Tabela 2: Nº de amostras em cada classe.

Classes	Nº de amostras
1	1102
2	708
3	118

Entretanto, há técnicas para contornar tal situação, as duas técnicas mais utilizadas para tratar o desbalanceamento é o *Undersampling* e *Oversampling*. O *Undersampling* remove os dados da classe majoritária até que ela tenha o mesmo número de amostras da classe minoritária. Já o *Oversampling* faz com que a classe minoritária se iguale, em número, às amostras da classe majoritária (Mani e Zhang, 2003).

Para tratar o desbalanceamento da base de dados foi utilizado os algoritmos *Near Miss* e SMOTE que utilizam das técnicas *Undersampling* e *Oversampling*, respectivamente. O intuito de aplicar as duas técnicas é comparar os resultados para verificar qual abordagem é a mais adequada para o problema em questão.

Near Miss é baseado na distância dos registros da classe majoritária aos registros da classe minoritária. É uma abordagem de k vizinho mais próximo, onde a distância euclidiana pode ser usada. A menor distância média entre a classe majoritária e os três registros mais próximos da classe minoritária é utilizada para identificar as amostras da classe majoritária que serão excluídas (Brownlee, 2020).

O *Synthetic Minority Oversampling Technique* (SMOTE) é uma abordagem de sobreamostragem na qual a classe minoritária é sobreamostrada criando amostras sintéticas. O processo de criação das amostras ocorre da seguinte forma: seleciona cada amostra de classe minoritária e cria amostras sintéticas ao longo das linhas que unem os k vizinhos mais próximos da classe minoritária (Chawla et al., 2002).

3.5 Métodos de Classificação

Na etapa de classificação quatro métodos foram aplicados e comparados: *AdaBoost*, *Decision Tree*, *Random Forest* e *Extreme Gradient Boosting*.

AdaBoost é um procedimento iterativo que tenta aproximar um classificador $C*(x)$ combinando vários classificadores fracos. Começando com a amostra de treinamento não ponderada, o *AdaBoost* constrói um classificador, por exemplo, uma árvore de classificação, que produz rótulos de classe. Se um ponto de dados de treinamento for classificado incorretamente, o peso desse ponto de dados de treinamento será aumentado (aumentado). Um segundo classificador é construído com os novos pesos, que não são mais iguais. Novamente, dados de treinamento mal classificados têm seus pesos aumentados e o procedimento é repetido (Hastie, Rosset, Zhu e Zou, 2009). Uma pontuação é atribuída a cada classificador, e o classificador final é definido como a combinação linear dos classificadores de cada estágio.

Considerando $T(x)$ um classificador multiclasse fraco que atribui um rótulo de classe a x, então o algoritmo *AdaBoost* procede da seguinte forma:

i. Inicialize os pesos de $w_i = 1/n, i = 1, 2, ..., n$.

ii. Para m = 1to M:
(a) Ajuste um classificador $T^{(m)}(\mathbf{x})$ aos dados de treinamento usando pesos w_i.
(b) Calcular

$$err^{(m)} = \sum_{i=1}^{n} w_i I(c_i \neq T^{(m)}(\mathbf{x}_i)) / \sum_{i=1}^{n} w_i$$

(c) Calcular

$$\alpha^{(m)} = log\frac{1 - err^{(m)}}{err^{(m)}}$$

(d) Defina

$$w_i = w_i \cdot exp(\alpha^{(m)} \cdot I(c_i \neq T^{(m)}(\mathbf{x}_i)))$$

, para i = 1, ..., n
(e) Renormalize w_i

iii. Saída

$$C(x) = argmax_k \sum_{m=1}^{M} \alpha^{(m)} \cdot I(T^{(m)}(\mathbf{x}) = k)$$

Decision Tree (DT) é um método supervisionado que possui estrutura de árvore, onde o conjunto de dados é dividido em subconjuntos cada vez menores enquanto uma árvore de decisão associada é desenvolvida de forma incremental. Ao final tem-se uma árvore com nós de decisão e nós de folha. DT recebe como entrada um conjunto de atributos e retorna uma decisão, que é a classe predita do valor de entrada. A decisão é tomada executando uma sequência de testes: cada nó interno da árvore corresponde a um teste do valor das características, e os ramos desse nó identificam possíveis valores de teste. Cada nó folha específica o retorno de seu valor se a folha for alcançada (Hastie, Tibshirani e Friedman, 2009).

Random Forest (RF) é um método de aprendizagem por *ensemble* que opera construindo k árvores de decisão a partir do conjunto de treinamento em k iterações. Em cada iteração, primeiro é selecionado aleatoriamente um conjunto de amostras do conjunto de treinamento. Para reproduzir uma árvore de decisão desse subconjunto, o RF escolhe aleatoriamente um subconjunto de atributos candidatos para cada nó. Desta forma, cada árvore de decisão é construída através do *ensemble*, empregando subconjuntos aleatórios independentes de características e amostras. A predição de uma nova classe de amostra é realizada da seguinte forma, cada classificador individual vota e a classe mais votada é eleita (Breiman, 2001).

O *Extreme Gradient Boosting* (XGB) é uma adaptação do *Gradiente Boosting* que demonstrou ser mais eficiente na resolução de problemas de aprendizagem supervisionada (Ibrahem Ahmed Osman et al., 2021). As seguintes etapas são executadas: Adotando um conjunto de dados com recursos m e um número n de amostras $(x_1, y_1), ..., (x_n, y_n)$ onde $x_i \in \mathbb{R}^n$ e $y_i \in \mathbb{R}$, $i = 0, ...n$. Seja $\hat{y}_i$ a saída prevista de

XGB dada por:

$$\phi(x_i) = \sum_{k=1}^{K} h_k(x_i), \qquad h_k \in \mathbb{H} \tag{2}$$

onde K indica o número de árvores de decisão, h_k é o modelo da $k$ésima árvore de decisão e $m_{profundidade}$ é a profundidade máxima do modelo h_k. Para gerar o modelo é necessário minimizar a função de perda de regularização

$$L(\phi) = \sum_j l(y_j, \phi(x_j)) + \frac{1}{2}\left(\sum_j w_j^2\right)^{\frac{1}{2}} \tag{3}$$

onde a função de perda é $l = \|\hat{y}_i - y_i\|$, $\hat{y}_i$ e y_i são a saída estimada e verdadeira, respectivamente, e $w = (w_1, w_2, \dots)$ é o vetor peso da folha.

3.6 Validação Cruzada e Grid-Search

A Validação Cruzada (VC) é uma técnica para avaliar a capacidade de generalização de um modelo a partir do conjunto de dados. Essa abordagem é muito aplicada em problemas onde o objetivo é realizar a predição. VC divide o banco de dados em dois conjuntos distintos: treinamento e teste. O conjunto de treinamento é usado para estimar os parâmetros do modelo e o conjunto de teste é usado para validar o modelo.

Para avaliar os classificadores foi empregada a estratégia *K-Fold* (KF) (Kohavi, 1995). KF no banco de dados disponível contém N amostras e é dividido em K subconjuntos, onde $K > 1$. Após particionar o banco de dados, os subconjuntos gerados por K-1 são usados para treinamento e os conjuntos restantes são usados para teste. Desta forma, ao final do procedimento, mede-se o erro de validação. Este processo é repetido K vezes, cada vez usando um conjunto de teste diferente em cada iteração.

O desempenho do classificador é calculado em K testes. O objetivo de repetir os testes várias vezes é treinar da melhor forma possível a máquina para que ela possa generalizar entradas futuras.

O procedimento *Grid-Search* é uma busca exaustiva pelos melhores parâmetros obtidos a partir da implementação do algoritmo para um intervalo de parâmetros (Bergstra e Bengio, 2012). Ao usar o *Grid-Search* com a validação cruzada, o processo é repetido para cada combinação de parâmetros ajustados para um classificador para maximizar a acurácia. O intervalo de parâmetros para cada modelo no procedimento *Grid-Search* com validação cruzada pode ser vista na Tabela 3.

3.7 Métricas para Desempenho de Classificação

Para avaliar o desempenho dos métodos foram utilizadas as seguintes métricas: Acurácia (AC), Recall (Re), F1, Kappa e Matriz de Confusão. A Acurácia (AC), definida na Eq. (4), mede o percentual de acerto do método comparando as classes preditas com as classificadas pelo método manual, por contagem direta.

$$\text{AC} = \frac{1}{N}\sum_{i=1}^{N} I(f(x_i) = y_i) \tag{4}$$

onde, $f(x_i)$ é a classe predita as amostras de teste e y_i é a classe verdadeira dessas amostras. Considerando que $I(true) = 1$ e $I(false) = 0$.

Recall (Re) é dado por

$$\text{Re}(c_k) = \frac{TP_k}{TP_k + FN_k} \tag{5}$$

onde mede a porcentagem de amostras positivas reais classificadas como positivas. TP_k e FN_k são o número de verdadeiros positivos e falsos positivos para a classe c_k, respectivamente.

F1 score é escrito como

$$\text{F1}(c_k) = \frac{2TP_k}{2TP_k + FP_k + FN_k} \tag{6}$$

onde TP_k é o número de amostras positivas classificadas corretamente, FP_k é o número de amostras negativas classificadas como positivas e FN_k é o número de amostras positivas classificadas como negativas. F1 atinge seu melhor valor em 1 e a pior pontuação em 0.

O Teste Kappa é um critério de concordância entre observadores e calcula o nível de concordância além do que seria previsto somente pelo acaso. Esta medida de concordância assume um valor máximo de 1; valores próximos e até abaixo de 0 indicam ausência de concordância. O coeficiente Kappa (KAPPA) é calculado consoante a Eq. (7).

$$\text{KAPPA} = \frac{P_o - P_E}{1 - P_E} \tag{7}$$

onde

$$P_o = \frac{\text{no. agreement}}{\text{no. agreement + no. disagreement}} \tag{8}$$

e

$$P_E = \sum_{i=1}^{N}(p_{i1} \times p_{i2}) \tag{9}$$

onde N é o número de categorias, i é o índice de categorias, p_{i1} é a ocorrência da categoria de proporção i para o avaliador 1, p_{i2} é a ocorrência da categoria de proporção i para o avaliador 2.

Tabela 4 mostra a interpretação da Estatística Kappa, utilizada para avaliar o nível de concordância entre os avaliadores (Landis e Koch, 1977).

A matriz de confusão é uma tabela avaliativa de comparar as classes que um método apontou em relação às classes reais. Após o treinamento de um modelo e classificação sobre o conjunto de teste, o resultado gerado será indicado em uma coluna com as classes preditas.

A Tabela 5 exemplifica uma matriz de confusão para

Tabela 3: Parâmetros usados em *Grid-Search* com validação cruzada. Mais informações sobre os parâmetros podem ser encontradas em Pedregosa et al. (2011).

Classificadores	Parâmetros	Variação
AB	Taxa de Aprendizado	[0.1, 0.01, 0.001,0.0001]
	Nº de estimadores	[50, 100, 200, 300, 400]
DT	Pronfundidade Máxima da Árvore	[10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
	Nº Minimo de nós folhas	[5, 10, 20, 30, 40, 50]
RF	Nº de estimadores	[50, 100, 200, 300]
	Pronfundidade Máxima das Árvores	[5, 10, 20, 30]
XGB	Taxa de Aprendizado	[0.1, 0.01, 0.001,0.0001]
	Nº de estimadores	[50, 100, 200, 300]
	Pronfundidade Máxima das Árvores	[2,5,10,20]

Tabela 4: Nível de Concordância Estatística Kappa.

Estatística Kappa	Nível de Concordância
< 0.0	Pobre
0.00 – 0.20	Pouca
0.21 – 0.40	Razoável
0.41 – 0.60	Moderado
0.61 – 0.80	Substancial
0.81 – 1.00	Perfeita

um caso em que há duas classes Positivo (1) e Negativo (0). Na diagonal principal da matriz, VP (Verdadeiro Positivo) e VN (Verdadeiro Negativo) são as quantidades de amostras preditas corretamente pelo método para cada classe. Fora da diagonal principal estão a quantidade de amostras preditas erroneamente pelo método, representadas por FP (Falso Positivo) e FN (Falso Negativo).

Tabela 5: Matriz de Confusão para Classes 0 e 1 .

	Real Positivo = 1	Real Negativo = 0
Predito Positivo = 1	VP	FP
Predito Negativo = 0	FN	VN

4 Resultados e Discussões

Foram realizadas 30 iterações independentes para avaliar a metodologia. O valor do *K* utilizado no *K-Fold* foi igual a 5. A primeira análise realizada foi sem o balanceamento das amostras. Na Tabela 6 pode-se visualizar os resultados obtidos, com a média e o desvio padrão das iterações para cada métrica. O melhor resultado obtido foi com o algoritmo XGB, com uma acurácia e recall de 0.6759, F1 de 0.6655 e um Kappa de 0.3668, mostrando um nível de concordância razoável do resultado.

A Fig. 1 apresenta a matriz de confusão para os métodos empregados. Pode-se observar que o maior percentual de erros na classificação foi referente às classes 2 e 3 e a classe 1 apresentou o maior percentual de acertos. Isso mostra o efeito do desbalanceamento no processo de classificação, onde os métodos tiveram maior dificuldade de aprender nas classes com o menor número de amostras.

O melhor modelo que o XGB apresentou pode ser visualizado na Tabela 7, onde obteve uma taxa de acerto de 73%.

Na segunda análise foi utilizado o método de balanceamento SMOTE, cujo objetivo é aumentar o número de amostras das menores classes, igualando ao número de amostras da maior classe. A Tabela 8 apresenta os resultados obtidos, com a média e o desvio padrão das 30 iterações para cada métrica. O algoritmo que obteve o melhor resultado foi o RF com uma acurácia e recall de 0.8381, F1 de 0.8374, Kappa de 0.7570 mostrando um nível de concordância substancial dos resultados. Todavia, o XGB também apresentou um bom resultado, com uma acurácia de 83.01%.

A Fig. 2 apresenta a matriz de confusão para os métodos empregados após o balanceamento com o SMOTE. Nota-se a proporção de acertos em relação às classes aumentou se comparado a base de dados desbalanceada. A maioria dos métodos teve maior porcentagem de erro na classe 2 e de acertos na classe 3.

O RF com a base de dados balanceada pelo SMOTE obteve uma acurácia de 86% no melhor modelo, conforme pode ser visto na Tabela 9.

Na última análise foi utilizado o método de balanceamento *Near Miss*, que faz que as classes fiquem com o mesmo número de amostras da classe com o tamanho de amostra menor. A média e o desvio padrão das 30 iterações para cada métrica é apresentado na Tabela 10. O algoritmo que apresentou o melhor resultado foi RF, com uma acurácia de 58.32%, recall de 58.32%, F1 de 58.54% e um kappa de 0.3733, mostrando que o nível de concordância dos resultados é razoável. Além disso, o algoritmo XGB apresentou uma acurácia de 56.85%, sendo muito próxima do RF.

A Fig. 3 apresenta a matriz de confusão para os métodos empregados após o balanceamento com o *Near Miss*. Ao observar os resultados, pode-se dizer que a abordagem de *undersampling* por meio do *Near Miss* não trouxe uma melhora de desempenho, pois os métodos estão classificando amostras de uma determinada classe em outras classes, como, por exemplo, para o RF que acertou 65% das classificações na classe 0, classificou 32% como pertencentes a classe 1 e 10% como classe 2. Pelo fato de ter poucas amostras para treinar os métodos não conseguiram aprender o suficiente, apresentando dificuldade nas três classes.

A Tabela 11 exibe o melhor modelo do RF utilizando o *Near Miss*. O método, nessa configuração, atinge uma taxa de acertos de 67%.

Os melhores desempenhos dos algoritmos foram obti-

Tabela 6: Média e desvio padrão da Acurácia, Recall, F1 e Kappa para os dados desbalanceados. Os melhores resultados aparecem em negrito.

Classificador	AC	Re	F1	kappa
AB	0.5886 ± 0.0265	0.5886 ±0.0265	0.5457 ±0.0428	0.1398 ± 0.0557
DT	0.6201 ± 0.0273	0.6201 ± 0.0273	0.6085± 0.0276	0.2550 ± 0.0480
RF	0.6739 ± 0.0231	0.6739± 0.0231	0.6532 ± 0.0268	0.3419 ± 0.0418
XGB	**0.6759 ± 0.0196**	**0.6759 ± 0.0196**	**0.6655 ± 0.0224**	**0.3668 ± 0.0357**

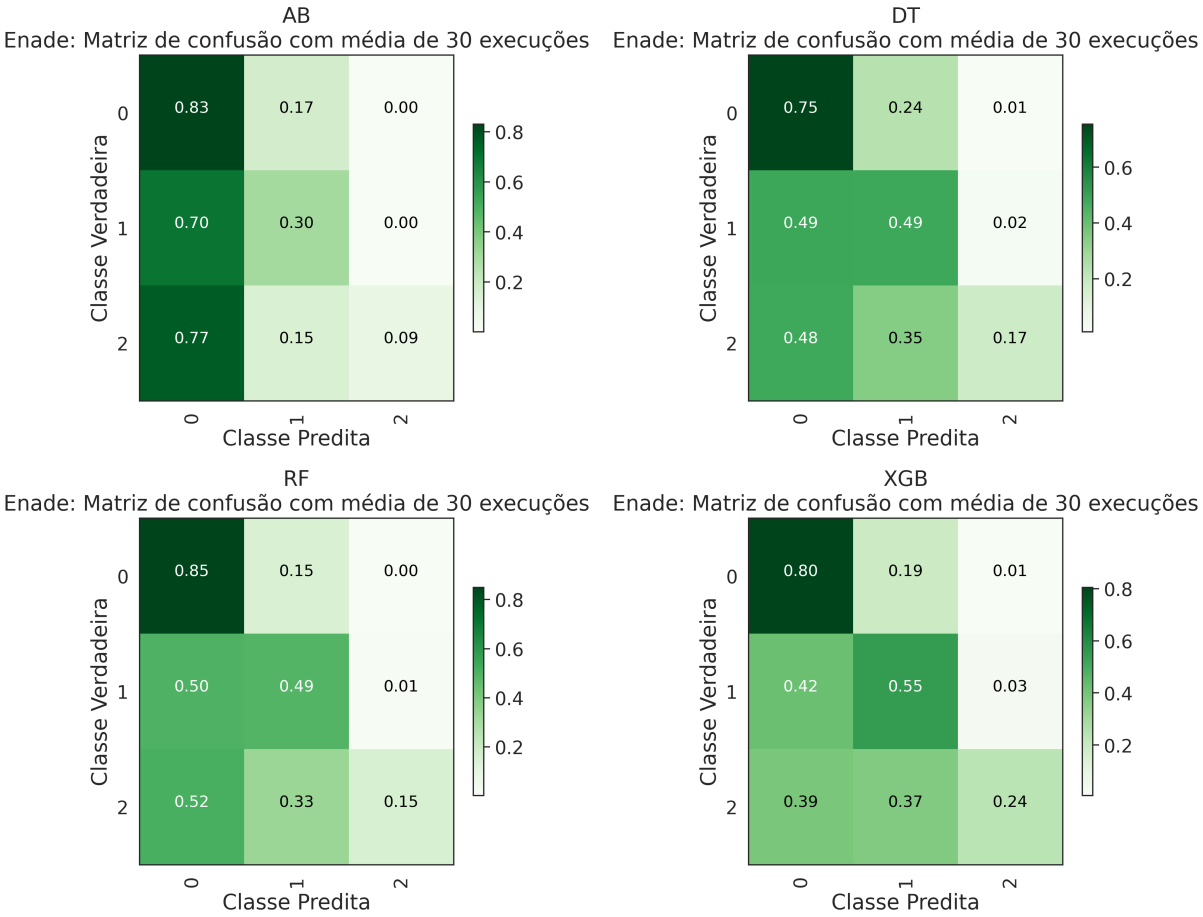

Figura 1: Matriz de confusão no conjunto de dados de teste para AB, DT, RF e XGB sem balanceamento, respectivamente. Os resultados foram obtidos usando validação cruzada (5-folds). As entradas normalizadas foram calculadas em média em 30 execuções independentes.

Tabela 7: Melhor Modelo XGB - sem o balanceamento das amostras.

Parâmetros	AC	Re	F1	kappa
Taxa de Aprendizado = 0.1, Profundidade Máxima das Árvores = 20,Nº de Estimadores = 50	0.7351	0.7351	0.7324	0.4748

dos com o método de balanceamento SMOTE, que aumenta o número de amostras das menores classes, igualando ao número de amostras da maior classe, com acurácias médias de aproximadamente 50.6%, 49.1%, 83.8% e 83.0% para AB, DT, RF e XGB, respectivamente. Já com o método de balanceamento *Near Miss*, que diminui o número de amostras das classes maiores igualando a classe de menor amostra, os resultados não foram bons com acurácias médias de aproximadamente 50.6%, 49.1%, 58.3% e 56.8% para AB, DT, RF e XGB, respectivamente, assim, mostrando que para esse tipo de problema o método *Near Miss* não é adequado para a tarefa em questão. Os algoritmos XGB e RF foram os que apresentaram as maiores acurácias em todas as análises, mostrando que usar métodos *ensemble* na qual fazem a combinação de múltiplos modelos se mostraram eficientes. Ademais, percebe-se

Tabela 8: Média e desvio padrão da Acurácia, Recall, F1 e Kappa para os dados balanceados com SMOTE. Os melhores resultados aparecem em negrito.

Classificador	AC	Re	F1	kappa
AB	0.5060 ± 0.0539	0.5060 ± 0.0538	0.5083 ± 0.0528	0.2602 ± 0.0776
DT	0.4914 ± 0.0649	0.4914 ± 0.0649	0.4855 ± 0.0688	0.2355 ± 0.0909
RF	**0.8381 ± 0.0125**	**0.8381 ± 0.0125**	**0.8374 ± 0.0125**	**0.7570 ± 0.0188**
XGB	0.8301 ± 0.0121	0.8301 ± 0.0121	0.8297 ± 0.0123	0.7449± 0.0183

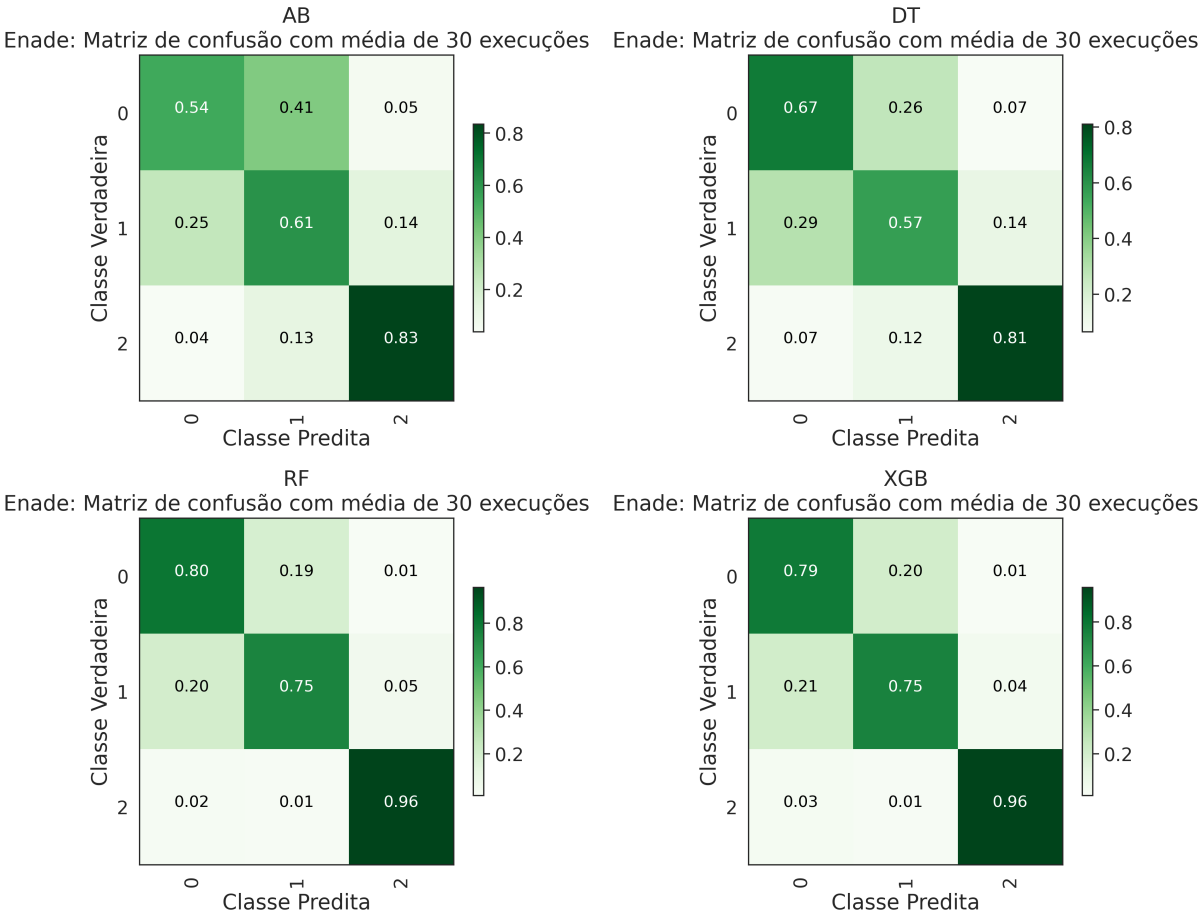

Figura 2: Matriz de confusão no conjunto de dados de teste para AB, DT, RF e XGB aplicando o SMOTE, respectivamente. Os resultados foram obtidos usando validação cruzada (5-folds). As entradas normalizadas foram calculadas em média em 30 execuções independentes.

Tabela 9: Melhor Modelo RF - SMOTE.

Parâmetros	AC	Re	F1	kappa
Profundidade Máxima das Árvores = 20, Nº de Estimadores = 300	0.8669	0.8669	0.8671	0.8003

que o desbalanceamento dos dados interfere no processo de aprendizado dos métodos e aplicar técnicas de balanceamento das amostras é importante para aumentar a acurácia dos classificadores.

As vantagens da metodologia proposta são conseguir realizar a classificação de uma forma automática, onde se encontrou o melhor modelo por meio de uma busca exaustiva e aplicaram-se técnicas de seleção de características e de balanceamento de dados para utilizar informações relevantes para a tarefa em questão além de tratar o desbalanceamento permitindo que os métodos fossem treinados de uma forma igualitária entre as classes. Pode-se citar como desvantagens o fato da busca dos melhores parâmetros ser feita de forma discreta entre os valores colocados na lista e a restrição do domínio da aplicação, em termos do curso e da região. Como trabalhos futuros pode-se aplicar

Tabela 10: Média e desvio padrão da Acurácia, Recall, F1 e Kappa para os dados balanceados com *Near Miss*. Os melhores resultados aparecem em negrito.

Classificador	AC	Re	F1	kappa
AB	0.5060 ± 0.0539	0.5060 ± 0.0539	0.5083 ± 0.0528	0.2602 ± 0.0776
DT	0.4914 ± 0.0649	0.4914 ± 0.0649	0.4855 ± 0.0688	0.2355 ± 0.0909
RF	**0.5832 ± 0.0442**	**0.5832 ± 0.0442**	**0.5854 ± 0.0422**	**0.3733 ± 0.0655**
XGB	0.5685 ± 0.0587	0.5685 ± 0.0587	0.5736 ± 0.0557	0.3518 ± 0.0865

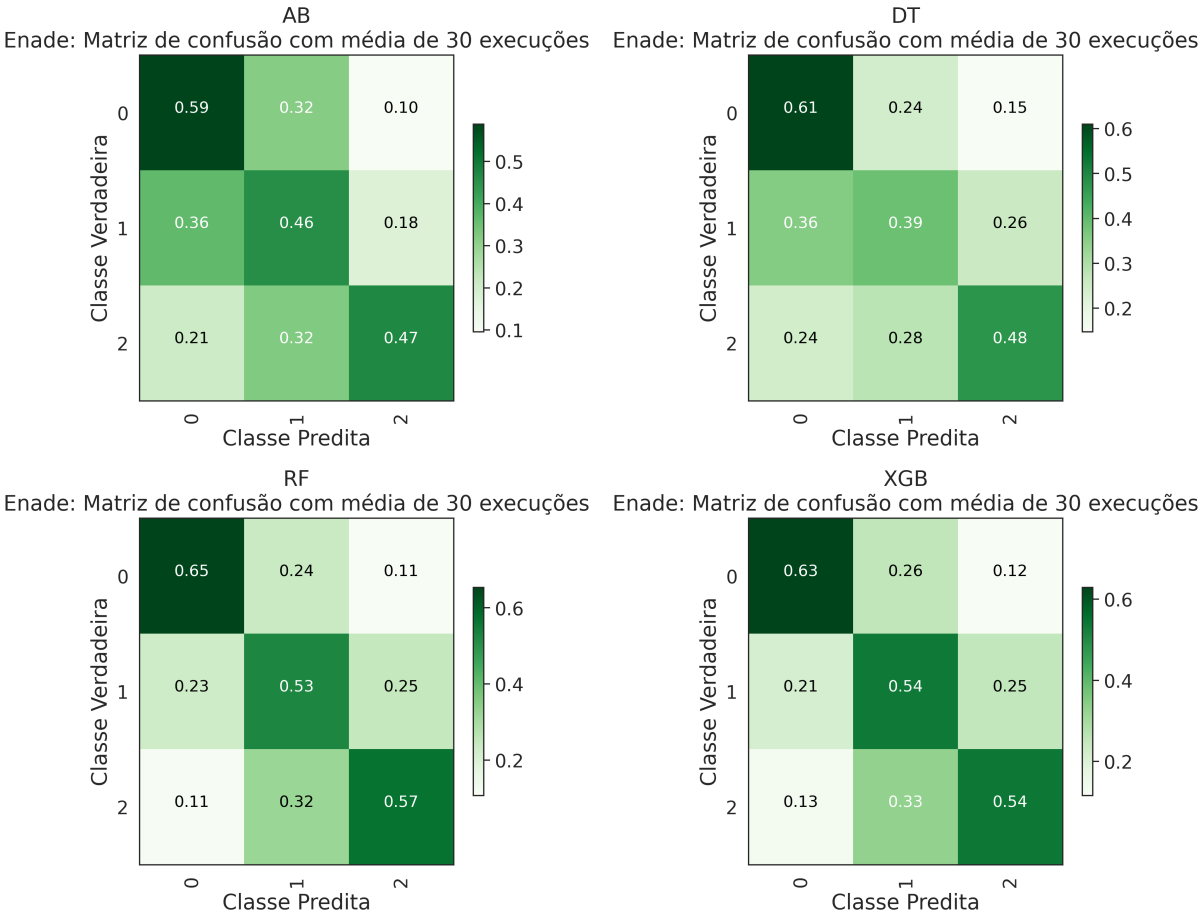

Figura 3: Matriz de confusão no conjunto de dados de teste para AB, DT, RF e XGB aplicando o *Near Miss*, respectivamente. Os resultados foram obtidos usando validação cruzada (5-folds). As entradas normalizadas foram calculadas em média em 30 execuções independentes.

Tabela 11: Melhor Modelo RF - Near Miss.

Parâmetros	AC	Re	F1	kappa
Profundidade Máxima das Árvores = 20, Nº de Estimadores = 200	0.6714	0.6714	0.6705	0.4924

meta-heurísticas para buscar os parâmetros ótimos dos classificadores e ampliar a aplicação da metodologia para outros cursos e regiões do país.

5 Conclusão

Neste artigo foi analisado o desempenho de quatro classificadores (AB, DT, RF e XGB) para classificação de tempo de formação dos estudantes de Engenharia da Computação da região Sudeste. Com intuito de contornar as desvantagens impostas pelo desbalanceamento da base de dados, foi empregado um método *oversampling*, o SMOTE, e um

undersampling, *Near Miss*.

Para os dados desbalanceados as acurácias não foram ótimas, para classe 1 o percentual de acerto foi bom, pelo fato de ter mais amostras, mas para as classes 2 e 3 por terem poucas amostras teve um maior percentual de erro, assim, dificultando o treinamento dos algoritmos. O algoritmo que obteve o melhor resultado foi XGB, com uma acurácia de 67.59% seguido do RF com 67.39% . Percebeu-se através da matriz de confusão que os métodos classificaram mais amostras erroneamente nas classes minoritárias (classes 1 e 2), o que influenciou no desempenho dos algoritmos.

O SMOTE foi a técnica de balanceamento que obteve os melhores resultados, muito devido à maior quantidade de amostras das classes, o que possibilitou ter um melhor treinamento dos algoritmos. O algoritmo que obteve a melhor acurácia foi o RF com 83.81%.

O *Near Miss* não apresentou bons resultados, muito devido ao número de amostras para o treinamento dos algoritmos. O algoritmo que obteve o melhor resultado foi o RF com uma acurácia de 58.32%. Portanto, para esse tipo de problema abordado do trabalho, este método de balanceamento não é adequado pois diminui a taxa de acerto dos classificadores comparado aos dados desbalanceados.

A abordagem proposta mostrou-se eficiente, pois utilizou técnicas de balanceamento, seleção de características para selecionar as informações que seriam fundamentais para aprendizagem dos algoritmos. Além disso, com o uso *Grid-Search* possibilitou buscar os melhores parâmetros para maximizar a acurácia dos classificadores, como também, fez se o uso de algoritmos *ensembles* que fazem a combinação de múltiplos modelos.

Referências

Alghamdi, M., Al-Mallah, M., Keteyian, S., Brawner, C., Ehrman, J. e Sakr, S. (2017). Predicting diabetes mellitus using smote and ensemble machine learning approach: The henry ford exercise testing (fit) project, *PloS one* **12**(7): e0179805. Disponível em https://doi.org/10.1371/journal.pone.0179805.

Almeida, D. S. S. et al. (2021). Saúde digital: predição do risco de reinternação em hospitais universitários federais. Disponível em https://bdtd.ucb.br:8443/jspui/handle/tede/2838.

Bergstra, J. e Bengio, Y. (2012). Random search for hyper-parameter optimization, *Journal of Machine Learning Research* **13**(Feb): 281–305. Disponível em https://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf.

Breiman, L. (2001). Random forests, *Machine learning* pp. 5–32. Disponível em https://doi.org/10.1023/A:1010933404324.

Brownlee, J. (2020). Undersampling algorithms for imbalanced classification, *Machine Learning Mastrey* **27**.

Casuat, C. D., Festijo, E. D. e Alon, A. S. (2020). Predicting students' employability using support vector machine: a smote-optimized machine learning system, *International Journal* **8**(5). Disponível em https://doi.org/10.30534/ijeter/2020/102852020.

Chawla, N. V., Bowyer, K. W., Hall, L. O. e Kegelmeyer, W. P. (2002). Smote: synthetic minority over-sampling technique, *Journal of artificial intelligence research* **16**: 321–357. Disponível em https://doi.org/10.1613/jair.953.

Chengsheng, T., Huacheng, L. e Bing, X. (2017). Adaboost typical algorithm and its application research, *MATEC Web of Conferences*, Vol. 139, EDP Sciences, p. 00222. Disponível em https://doi.org/10.1051/matecconf/201713900222.

Ekle, F., Bakpo, F., Udanor, C. e Eneh, A. (2023). A machine learning model and application for heart disease prediction using prevalent risk factors in nigeria, *International Journal of Mathematical Analysis and Modelling* **6**(2). Disponível em https://tnsmb.org/journal/index.php/ijmam/article/view/104.

Fatty, A., Li, A.-J. e Qian, Z.-G. (2023). An interpretable evolutionary extreme gradient boosting algorithm for rock slope stability assessment, *Multimedia Tools and Applications* pp. 1–24. Disponível em https://doi.org/10.1007/s11042-023-17445-9.

Feldmann, T. e Souza, O. d. (2016). A governamentalidade e o exame nacional de desempenho de estudantes-enade, *Avaliação: Revista da Avaliação da Educação Superior (Campinas)* **21**: 1017–1032. Disponível em https://doi.org/10.1590/S1414-40772016000300017.

Góes, J. F. (2022). Análise do tempo de permanência em cursos da ufba: uma aplicação de modelagem com tempos discretos. Disponível em https://repositorio.ufba.br/handle/ri/36110.

Gunturi, S. K. e Sarkar, D. (2021). Ensemble machine learning models for the detection of energy theft, *Electric Power Systems Research* **192**: 106904. Disponível em https://doi.org/10.1016/j.epsr.2020.106904.

Hastie, T., Rosset, S., Zhu, J. e Zou, H. (2009). Multi-class adaboost, *Statistics and its Interface* **2**(3): 349–360. Disponível em https://dx.doi.org/10.4310/SII.2009.v2.n3.a8.

Hastie, T., Tibshirani, R. e Friedman, J. (2009). *The Elements of Statistical Learning*, Springer, New York. Disponível em https://doi.org/10.1007/978-0-387-21606-5.

Hatwell, J., Gaber, M. M. e Atif Azad, R. M. (2020). Adawhips: explaining adaboost classification with applications in the health sciences, *BMC Medical Informatics and Decision Making* **20**(1): 1–25. Disponível em https://doi.org/10.1186/s12911-020-01201-2.

Ibrahem Ahmed Osman, A., Najah Ahmed, A., Chow, M. F., Feng Huang, Y. e El-Shafie, A. (2021). Extreme gradient boosting (xgboost) model to predict the groundwater levels in selangor malaysia, *Ain Shams Engineering Journal* **12**(2): 1545–1556. Disponível em https://doi.org/10.1016/j.asej.2020.11.011.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection, **2**: 1137–1143. Disponível em https://www.ijcai.org/Proceedings/95-2/Papers/016.pdf.

Korchi, A., Messaoudi, F., Abatal, A. e Manzali, Y. (2023). Machine learning and deep learning-based students' grade prediction, *Operations Research Forum*, Vol. 4, Springer, p. 87. Disponível em https://doi.org/10.1007/s43069-023-00267-8.

Lai, H., Li, X.-Y., Xu, F., Zhu, J., Li, X., Song, Y., Wang, X., Wang, Z. e Wang, C. (2023). Applications of machine learning to diagnosis of parkinson's disease, *Brain Sciences* **13**(11): 1546. Disponível em https://doi.org/10.3390/brainsci13111546.

Landis, J. R. e Koch, G. G. (1977). The measurement of observer agreement for categorical data, **33**: 159–174. Disponível em https://doi.org/10.2307/2529310.

Manhães, L. M. B., da Cruz, S. M. S., Costa, R. J. M., Zavaleta, J. e Zimbrão, G. (2012). Identificação dos fatores que influenciam a evasão em cursos de graduação através de sistemas baseados em mineração de dados: Uma abordagem quantitativa, *Anais do VIII Simpósio Brasileiro de Sistemas de Informação*, SBC, pp. 284–295. Disponível em https://doi.org/10.5753/sbsi.2012.14413.

Mani, I. e Zhang, I. (2003). knn approach to unbalanced data distributions: a case study involving information extraction, *Proceedings of workshop on learning from imbalanced datasets*, Vol. 126, ICML, pp. 1–7. Disponível em https://www.site.uottawa.ca/~nat/Workshop2003/jzhang.pdf.

Matsumoto, K., Nohara, Y., Sakaguchi, M., Takayama, Y., Fukushige, S., Soejima, H., Nakashima, N., Kamouchi, M. et al. (2023). Temporal generalizability of machine learning models for predicting postoperative delirium using electronic health record data: Model development and validation study, *JMIR Perioperative Medicine* **6**(1): e50895. Disponível em https://doi.org/10.2196/50895.

Mqadi, N. M., Naicker, N. e Adeliyi, T. (2021). Solving misclassification of the credit card imbalance problem using near miss, *Mathematical Problems in Engineering* **2021**: 1–16. Disponível em https://doi.org/10.1155/2021/7194728.

Oladunni, T., Tossou, S., Haile, Y. e Kidane, A. (2021). Covid-19 county level severity classification with imbalanced class: A nearmiss under-sampling approach, *medRxiv* pp. 2021–05. Disponível em https://doi.org/10.1101/2021.05.21.21257603.

Oliveira, R. T. D. d. e Barbosa, J. D. (2016). Retenção universitária: Fatores condicionantes e ações da gestão acadêmica no curso de administração da ufs. Disponível em https://repositorio.ufsc.br/bitstream/handle/123456789/172049/OK%20-%20101_00350%20-%20OK%20-%20R.pdf.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. e Duchesnay, E. (2011). Scikit-learn: Machine learning in Python, *Journal of Machine Learning Research* **12**: 2825–2830. Disponível em https://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf.

Peixoto, H. M., Peixoto, M. M. e Alves, E. D. (2012). Aspectos relacionados à permanência de graduandos e pós-graduandos em disciplinas semipresenciais, *Acta Paulista de Enfermagem* **25**: 48–53. Disponível em https://doi.org/10.1590/S0103-21002012000900008.

Rattan, V., Mittal, R., Singh, J. e Malik, V. (2021). Analyzing the application of smote on machine learning classifiers, *2021 International Conference on Emerging Smart Computing and Informatics (ESCI)*, IEEE, pp. 692–695. Disponível em https://doi.org/10.1109/ESCI50559.2021.9396962.

Rolim, J. P. e Silva, R. C. (2021). Mineração de dados educacionais para identificação de perfil de retenção em um curso de ciência da computação, *Anais da IX Escola Regional de Informática de Goiás*, SBC, pp. 195–204. Disponível em https://doi.org/10.5753/erigo.2021.18444.

Sarkar, S., Vinay, S., Raj, R., Maiti, J. e Mitra, P. (2019). Application of optimized machine learning techniques for prediction of occupational accidents, *Computers & Operations Research* **106**: 210–224. Disponível em https://doi.org/10.1016/j.cor.2018.02.021.

Shi, Y., Zhang, Y., Cao, Z., Ma, L., Yuan, Y., Niu, X., Su, Y., Xie, Y., Chen, X., Xing, L. et al. (2023). Application and interpretation of machine learning models in predicting the risk of severe obstructive sleep apnea in adults, *BMC Medical Informatics and Decision Making* **23**(1): 230. Disponível em https://doi.org/10.1186/s12911-023-02331-z.

Silva, K. O. A., Silva, P. J. R. e Maciel, A. M. (2022). Análise de retenção escolar em cursos de graduação na poli/upe usando mineração de dados, *Revista de Engenharia e Pesquisa Aplicada* **7**(2): 108–117. Disponível em https://doi.org/10.25286/repa.v7i2.2223.

Song, L., Zhang, C., Hua, J., Li, K., Xu, W., Zhang, X. e Duan, C. (2023). A data-driven model to determine the infiltration characteristics of air curtains at building entrances, *Physics of Fluids* **35**(11). Disponível em https://doi.org/10.1063/5.0173678.

Vidal Bezerra, F. D., Pinto Marinho, F., Costa Rocha, P. A., Oliveira Santos, V., Van Griensven Thé, J. e Gharabaghi, B. (2023). Machine learning dynamic ensemble methods for solar irradiance and wind speed predictions, *Atmosphere* **14**(11): 1635. Disponível em https://doi.org/10.3390/atmos14111635.

Wandani, S., Suherman, A., Sultoni, K., Ruhayati, Y., Damayanti, I., Rahayu, N. I. et al. (2023). Classifying physical activity levels in early childhood using actigraph and machine learning method, *Indonesian Journal of Sport Management* **3**(2). Disponível em https://doi.org/10.31949/ijsm.v3i2.7173.

Wang, Y., Wu, X., Chen, Z., Ren, F., Feng, L. e Du, Q. (2019). Optimizing the predictive ability of machine learning methods for landslide susceptibility mapping using smote for lishui city in zhejiang province, china, *International journal of environmental research and public health* **16**(3): 368. Disponível em https://doi.org/10.3390/ijerph16030368.

Wu, Y., Ke, Y., Chen, Z., Liang, S., Zhao, H. e Hong, H. (2020). Application of alternating decision tree with adaboost and bagging ensembles for landslide susceptibility mapping, *Catena* **187**: 104396. Disponível em https://doi.org/10.1016/j.catena.2019.104396.