UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
PRÓ-REITORIA DE GRADUAÇÃO
CENTRO DE CIÊNCIAS EXATAS E NATURAIS
DEPARTAMENTO DE COMPUTAÇÃO
CURSO DE BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO
TRABALHO DE CONCLUSÃO DE CURSO (2024)

AVALIAÇÃO DO USO DE *FEW-SHOT LEARNING* EM MODELOS PARA A PREDIÇÃO DA EVASÃO ESCOLAR:
UMA ANÁLISE COMPARATIVA

Rian Carlos Silva Lima[1],
Angélica Félix de Castro[2],
Paulo Gabriel Gadelha Queiroz[3]

RESUMO
Este estudo analisou a viabilidade do uso de *few-shot learning* e *prototypical networks* para predição da evasão escolar de alunos do ensino superior. Um estudo anterior desenvolveu e comparou diversos modelos de aprendizado de máquina com o objetivo de predizer alunos evadidos, obtendo bons índices de acerto em algumas métricas, mas baixos resultados em outras. Com o objetivo de melhorar a predição da evasão, este trabalho implementou um modelo baseado em *prototypical networks* e comparou as métricas de desempenho com os modelos anteriores. Os resultados mostraram uma melhora significativa nas métricas de *recall* e *F1-score*, com o *recall* subindo de 60,00% para 91,18% (um aumento de 51,97%) e o *F1-score* passando de 66,28% para 88,27% (um aumento de 33,18%).

Palavras-chave: Evasão Escolar; Few-Shot Learning; Aprendizado de Máquina; Prototypical Networks

1 INTRODUÇÃO

A evasão de alunos do curso superior é um problema grave que afeta instituições de ensino brasileiras (Nascimento, 2022; Nascimento *et al*., 2024)e estrangeiras. Esse problema já foi pesquisado nos trabalhos de Marques *et al.*(2019), Marques *et al.* (2020) e Agrusti *et al.*(2019), dentre outros, a fim de encontrar possíveis soluções para mitigar os transtornos econômicos e sociais causados pela evasão. Em relação à Universidade Federal Rural do Semi-Árido (UFERSA), Nascimento (2022) por meio de análise dos dados fornecidos pela Divisão de Registros Acadêmico (DRA) em seu site[4], os cursos de Biotecnologia, Engenharia Agrícola e Ambiental e Ciência e Tecnologia (CeT), respectivamente, tiveram taxas de evasão de 17,05%, 18,46% e 17,70%, no período de 2009 a 2019. Este estudo focou no curso de CeT pois também de acordo com o DRA é o curso com maior número de alunos matriculados com 828 no diurno e 383 no noturno.

---

[1] Estudante do curso de Ciência da Computação no Centro de Ciências Exatas e Naturais da Universidade Federal Rural do Semiárido; e-mail: riancarlos.sl@gmail.com
[2] Professora do Centro de Ciências Exatas e Naturais da Universidade Federal Rural do Semiárido; e-mail: angelica@ufersa.edu.br
[3] Professor do Centro de Ciências Exatas e Naturais da Universidade Federal Rural do Semiárido; e-mail: pgabriel@ufersa.edu.br
[4] https://dra.ufersa.edu.br/estatisticas/

Ao considerar o curso de Ciência e Tecnologia diurno e noturno, de acordo com dados fornecidos pelo DRA, 297 alunos ingressaram no semestre de 2023.1. Para o mesmo período, apenas 123 alunos se tornaram egressos, dos quais 47 cancelaram espontaneamente o vínculo com a universidade (evadiram), 61 concluíram o curso e os demais (15) prestaram um novo vestibular ou foram transferidos. Esses dados mostram que as altas taxas de desistência persistem na universidade. Estes cancelamentos de vínculo e desistência, sabidamente geram danos socioeconômico, levando em consideração que o investimento anual por aluno na educação superior brasileira no ano de 2020 foi de US$ 14.735 (OECD, 2023) considerando PPP (Paridade de Poder de Compra), que ajusta os valores conforme o custo de vida nos diferentes países, permitindo uma comparação mais justa do poder de compra.

Esses impactos socioeconômicos levantam um alerta sobre a necessidade de mitigar a evasão de alunos nas universidades. Entretanto, como os recursos são escassos, os esforços para mitigar a evasão devem ser direcionados aos alunos que se encontram em maior risco. Portanto, o primeiro desafio se concentra em identificar esses alunos com antecedência. Uma solução promissora oferecida pela Ciência da Computação para enfrentar esse desafio é o *Machine Learning* (ML), ou Aprendizado de Máquina, que configura sistemas para identificar padrões em dados e fazer previsões (Janiesch *et al.*, 2021). Uma subárea do *machine learning* é o *deep learning* (DL), conceito de *machine learning* baseado em redes neurais profundas (Neural Networks, NN).

As redes neurais são inspiradas na estrutura do cérebro humano, apesar de se comportarem e aprender de forma diferente, elas são compostas por uma camada de neurônios artificiais que processam informações de forma hierárquica. Elas têm a capacidade de analisar grandes volumes de dados, detectar correlações sutis entre eles e prever ou classificar algo com precisão (Janiesch *et al.*, 2021). Modelos de ML não apenas aprendem com os dados, mas também se ajustam conforme mais informações são inseridas, o que pode torná-los mais eficientes com o tempo. Ao serem treinadas com dados de estudantes, como desempenho e condições socioeconômicas, essas redes podem prever quais alunos têm mais chances de evadir. Dessa forma, torna-se possível realizar intervenções preventivas, ajudando a reduzir a evasão com base em análises precisas e contínuas.

O estudo de Nascimento (2022) coletou dados socioeconômicos de 229 alunos egressos do curso de Ciência e Tecnologia da UFERSA e usou os dados para treinar e avaliar diversos modelos de ML com o objetivo de identificar os melhores resultados a partir de métricas de avaliação tais como *recall, accuracy, F1-Score* e *precision*. Estas métricas são usadas para avaliar a performance de modelos de classificação. *Recall* indica a capacidade do modelo de identificar corretamente as instâncias positivas (taxa de verdadeiros positivos) (Grandini *et al.*, 2020). *Precision* mede a proporção de verdadeiros positivos entre as previsões positivas. *Accuracy* reflete a proporção de previsões corretas sobre o total de previsões (Grandini *et al.*, 2020). Já o *F1-Score* é a média harmônica entre *Precision* e *Recall*, oferecendo um equilíbrio útil quando há uma distribuição desbalanceada de classes (Grandini *et al*., 2020). O estudo obteve bons resultados ao prever quando os alunos iriam se formar, com 92% de precisão e 90% de acurácia. No entanto, teve dificuldades em prever a evasão, alcançando apenas 60% de recall e um *F1-score* de 68%, o que indica que, embora a precisão fosse alta, o modelo não conseguiu identificar bem todos os casos de evasão. O modelo que teve o melhor desempenho foi baseado no *Support Vector Machine* (SVM) com um *recall* de 60%, *precision* de 93%, *F1-Score* de 68% e *Accuracy* de 90%. O baixo *recall* pode ter ocorrido devido à pequena quantidade de dados para treino que correspondeu a 80% dos dados em um dataset com 229 entradas e o desbalanceamento do dataset (45 evadidos e 184 concluintes). Por esse motivo, este trabalho tem como objetivo melhorar os resultados obtidos no trabalho de Nascimento (2022) por meio do uso da técnica *few-shot learning* (FSL).

A técnica de FSL tem se mostrado promissora (Chaves *et al*., 2021; Chaves *et al*.,

2022; Sun *et al*., 2019). Ela consiste em uma estrutura de ML pré-treinada capaz de generalizar novas categorias de dados, precisando apenas de alguns poucos dados para ser treinada e classificar os dados. Diversos modelos podem ser usados para aprendizado com poucos exemplos. Redes Neurais Siamesas, por exemplo, são capazes de identificar similaridades entre pares (Koch *et al.*, 2015), mas podem ser adaptadas para realizar classificação. *Matching Networks* utilizam exemplos salvos na memória para comparar com novas entradas (MN) (Vinyals *et al*., 2016). Modelos de *Meta-Learning*, como o *Model-Agnostic Meta-Learning* (MAML) (Finn *et al*., 2017), podem ser adaptados para aprender rapidamente novos conceitos com poucos exemplos. Por fim, as *Prototypical Networks* (PN) (Snell *et al.*, 2017) são projetadas para classificar com poucas tentativas, aprendendo a representação média de cada classe (protótipo) e calculando a distância entre novas entradas e esses protótipos.

O modelo de *prototypical networks* se mostrou como um forte candidato para tal tarefa e foi escolhido para o estudo, pois é simples e se mostrou eficiente em estudos anteriores (Snell *et al.*, 2017) para lidar com este tipo de classificação. Ao implementar essa técnica de ML é possível, a partir de métricas escolhidas para este estudo, como *recall, accuracy, F1-Score* e *precision*, verificar se ela é mais eficaz para prever quando alunos irão evadir, apenas utilizando um conjunto de dados pequeno.

Diante do exposto, a questão de pesquisa que norteia este trabalho é: como os modelos baseados em FSL se comparam com modelos comuns de ML para identificar alunos com risco de evasão a partir de um pequeno conjunto de dados de treino? Para responder a esta questão de pesquisa, nesta pesquisa foi definido um modelo baseado em FSL a partir do conjunto de dados obtido no trabalho de Nascimento (2022) e realizada uma comparação entre os resultados obtidos por Nascimento (2022) e os resultados obtidos no presente trabalho.

O restante deste trabalho está organizado da seguinte forma. No capítulo 2, apresenta-se a fundamentação teórica que norteou este trabalho. No capítulo 3, detalha-se a metodologia utilizada durante a pesquisa. No capítulo 4, é apresentado os resultados e discussão sobre as métricas do novo modelo e análise estatística. No capítulo 5 é apresentada a conclusão.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Few-Shot learning

Foi realizada uma revisão da literatura para identificar técnicas de ML promissoras para apoiar o desenvolvimento de modelos com desempenho superior aos modelos abordados no trabalho de Nascimento (2022). Uma técnica que se mostrou promissora para esse contexto foi o *few-shot learning* (FSL). O FSL tem diversas abordagens com um objetivo: Ensinar modelos com uma quantidade muito limitada de dados e permitir que eles sejam capazes de realizar tarefas como classificação e regressão. Em alguns casos apenas um pouco de exemplos de treino já é o suficiente para treinar uma rede com bastante eficácia (Parnami, 2022). Isto é diferente de modelos tradicionais de *Machine Learning* (ML) e *Deep Learning* (DL), que geralmente precisam de uma base de dados grande e variada para contemplar os diferentes casos desejados e ajustar adequadamente para os dados de entrada. Este conceito é importante para o desenvolvimento de modelos em cenários com dados limitados. As várias abordagens podem ser aplicadas para diversos objetivos, conforme discutido na sessão 1, os principais modelos nos quais FSL podem ser aplicados são: Modelos de *Meta-Learning*, Modelos baseados em Memória (*Memory-Augmented Networks*) como *Memory-Augmented Neural Networks* (MANNs) e os modelos baseados em métricas como as redes siamesas que

conseguem aprender a similaridade entre pares e as *Prototypical Networks*.

Cada uma das abordagens de FSL funciona de uma forma diferente, para casos diferentes e tem suas implementações próprias, com algumas mais próximas de modelos comuns de ML como a *prototypical network* que foi a escolhida para a realização deste trabalho e é apresentada em mais detalhes a seguir.

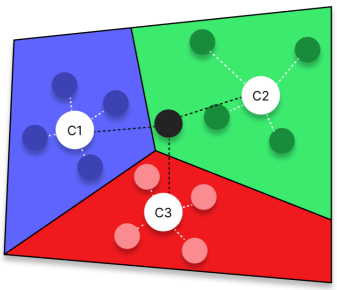
2.2 Prototypical network

Prototypical Networks (PN) é um tipo de rede projetada para comparar a distância entre a representação média de uma classe e os novos dados de entrada (Snell *et al.*, 2017). Ao medir essa distância o algoritmo classifica a entrada baseada no protótipo mais próximo. Essa técnica é particularmente útil para tarefas de classificação com poucos dados por ter uma implementação mais simples e próxima a modelos de RN padrões, podendo aprender com apenas *one-shot* ou *zero-shot*, que não é o caso desta pesquisa.

Para treinar uma PN, é necessário definir o número de classes (n-way) e o número de exemplos por classe (n-shot) (Snell *et al.*, 2017) . Além disso, é preciso separar um *support set*, que será usado para calcular os protótipos, e um *query set*, composto pelas amostras que queremos classificar ou prever durante o treinamento (Snell *et al.*, 2017) . Com essas definições, o processo de aprendizado da rede pode começar. No início de cada ciclo de treino, a rede seleciona *n* exemplos de cada classe para o aprendizado. Cada exemplo passa por um *encoder*, que transforma os dados em uma *embedding* que é uma representação vetorial em um espaço de menor dimensão (Snell *et al.*, 2017) . A escolha do *encoder* (como *Multilayer Perceptron* também conhecido como MLP, redes neurais convolucionais, entre outros) depende do tipo de problema, pois diferentes modelos podem ser mais adequados para gerar essas representações.

Depois que todos os exemplos de cada classe são convertidos em *embeddings*, o modelo calcula os protótipos das classes (Snell *et al.*, 2017) . Isso é feito pela média aritmética das *embeddings* de cada classe, resultando nos protótipos que o modelo usará para a classificação. Na Figura 1, um espaço de embeddings é representado, o *query* input de tamanho 1 (n-query = 1) é representado em preto, e cada classe c1, c2 e c3 (3-way) é definida pela média de quatro exemplos(nshot = 4).

Figura 1 - Representação de um Embedding Space com few-shots.



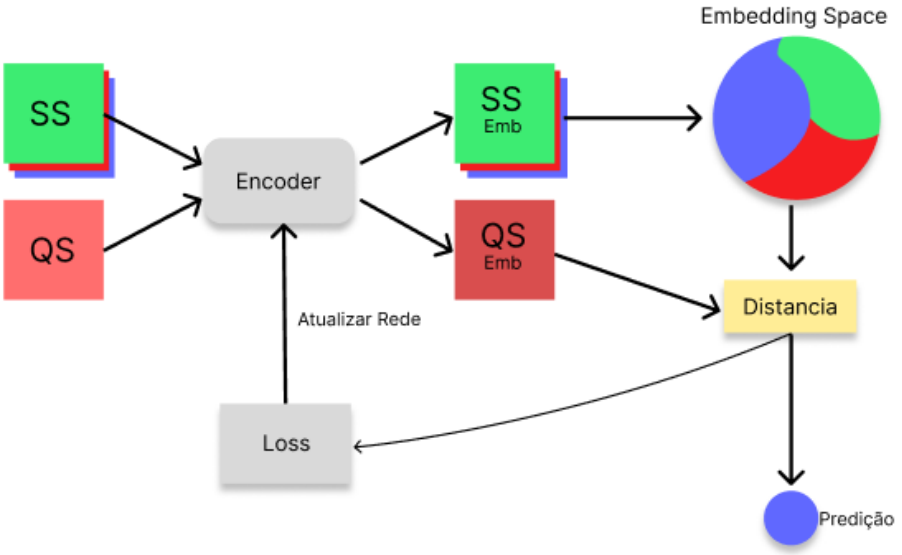
Fonte: Autoria própria (2024)

A classificação no modelo é feita após calcular os *embeddings* para cada entrada do *query set*, usando a mesma rede utilizada para gerar os *embeddings* do *Support set*. Com os protótipos já calculados a partir do *support set* e os *embeddings* do *query set* prontos, a classificação é realizada medindo a distância entre cada *embedding* do *query set* e os protótipos (Snell *et al.*, 2017). Normalmente, essa distância é calculada utilizando a distância euclidiana (Snell *et al.*, 2017), o que permite identificar a classe mais próxima de cada

exemplo no *query set*.

A perda (loss) é calculada com base na diferença entre as previsões do modelo e os rótulos corretos do *query set*. Essa perda é utilizada para ajustar os pesos e vieses do encoder por meio do processo de *backpropagation*, permitindo que o modelo aprenda a gerar *embeddings* mais precisas a cada ciclo de treinamento. Ao final de todos os ciclos, os parâmetros do modelo podem ser salvos para uso futuro. Na Figura 2 ilustra-se o modelo de uma *Prototypical Network*, destacando o *Support Set* (SS) e o *Query Set* (QS), cada conjunto pode ter uma ou mais amostras . Ambos os conjuntos passam pelo encoder, que transforma seus exemplos em embeddings. As embeddings geradas a partir do *Support Set* (SS emb) são utilizadas para calcular os protótipos no espaço de embeddings. Já as *embeddings* do *Query Set* (QS emb) são usadas para calcular as distâncias em relação aos protótipos, permitindo a classificação das amostras do QS. Este ciclo de aprendizado consegue com poucas amostras e em poucas épocas aprender a classificar ou prever classes com alta acurácia.

Figura 2 - Representação de uma Prototypical Network, cada cor representa uma classe.



Fonte: Autoria Própria (2024).

3 METODOLOGIA

Foi realizada uma pesquisa sobre a técnica de *few shot learning* e suas abordagens, na qual a *prototypical networks* foi estudada e escolhida como a abordagem para os problemas identificados. Revisão e reprodução de resultados analisados por Nascimento (2022). Neste passo foi identificado que o *recall* dos modelos foi baixo, que os dados estavam desbalanceados e em baixa quantidade e isto poderia ser um problema a ser estudado, e alguns modelos foram reproduzidos para confirmar os resultados. Depois com a abordagem selecionada o próximo passo foi a implementação de uma PN e o cálculo de suas métricas. E por fim foi realizado uma comparação estatística para aferir se o modelo de PN é mais eficiente e é diferente dos demais.

3.1 Estudo bibliográfico e reprodução de resultados anteriores de Nascimento (2022)

Este estudo dá continuidade à dissertação de Nascimento (2022), que realizou uma pesquisa sobre o tema, utilizando um formulário anônimo para coletar dados de alunos da UFERSA e, posteriormente, analisou esses dados para avaliar a evasão estudantil. Este

formulário contém dados sociais e econômicos de alunos. Os dados passaram por um tratamento e os dados da tabela foram transformados em valores numéricos. A primeira coluna identifica os *labels,* 0 significa concluinte e 1 evadido. Nascimento (2022) criou e avaliou diversos modelos de *Machine Learning* (ML), comparando métricas como acurácia, precisão, *recall* e *F1-score*. Os modelos implementados por Nascimento (2022) foram *Perceptron*, *Multilayer Perceptron* (MLP), KNN, *Native Bayes* (NB), *Decision Tree, Random Forest* e *Support Vector Machine* (SVM), com *4k-folds* de *cross validation* seguindo as configurações do apêndice (O conceito de *folds* será apresentado na análise estatística). As questões mais problemáticas em relação aos modelos de Nascimento (2022) são os baixos *recall* e *F1-score,* provavelmente devido ao uso de um conjunto de dados desbalanceado e um pequeno número de amostras de treino (229) no referido estudo.

Após executar cada modelo e sua variações por 50 vezes, e aplicar o método estatístico de Friedman que é utilizado para comparar modelos de ML, foi identificado que os modelos não apresentam uma diferença significativa de eficácia apesar de terem uma pequena variação em suas métricas de *Accuracy*, *Recall*, *Precision* e *F1-Score*, com exceção dos modelos de NB que apresentaram uma eficácia abaixo dos demais, para o caso de dados de ex-alunos do curso de CeT. Todos os modelos apresentaram acurácia, acima dos 81% com destaque para SVM que foi o único que atingiu os 90%. Entretanto, quando observadas a métrica de *recall* de 60%, o modelo SVM mostrou que conseguiu identificar corretamente 60% dos alunos que realmente evadiram. Isso significa que, entre todos os casos de evasão (representados pelo valor 1 no dataset), o modelo acertou a previsão de evasão para 40% deles (Verdadeiro Negativo). No entanto, 60% dos alunos que evadiram foram classificados incorretamente como concluintes (Falso Negativo). Em outras palavras, o modelo previu que esses alunos concluíram, mas, na realidade, eles acabaram evadindo.

O primeiro passo consistiu em um estudo bibliográfico das tecnologias utilizadas por Nascimento (2022), com o objetivo de compreender as razões para os valores baixos observados em algumas das métricas obtidas. Além disso, foi realizada uma análise detalhada sobre aprendizado de máquina (ML) e redes neurais (RN) para entender seu funcionamento. O estudo também abrangeu o conceito de *few-shot learning* e suas principais abordagens.

O segundo passo do projeto consistiu em reproduzir alguns dos modelos avaliados por Nascimento (2022). Foram escolhidos para serem reproduzidos os 3 modelos com as melhores métricas: SVM, MLP e Perceptron. Na Tabela 1, são apresentados os resultados das métricas *Accuracy*, *Precision*, *Recall* e *F1-Score* a partir da reprodução do código apresentado no apêndice do trabalho de Nascimento (2022)..

Tabela 1 - Modelos reproduzidos e resultado de suas métricas.

Modelo	Accuracy	Precision	Recall	F1-Score
Perceptron	81,73%	84,09%	52,27%	53,33%
MLP	87,34%	83,33%	35,79%	48,64%
SVM	90,41%	92,70%	60,60%	68,28%

Fonte: Autoria Própria (2024)

Os modelos foram reproduzidos utilizando a mesma base de dados e os códigos de Nascimento et al. (2022) sem nenhuma alteração, na ferramenta Google Colaboratory[5] utilizando Python 3. Após reproduzir os modelos e obter métricas semelhantes as do estudo anterior, ficou claro que os modelos, apesar de eficazes para identificar quando um aluno irá se formar, não se mostram eficazes para mostrar quando um aluno irá evadir como pode ser visto na Tabela 1 (*Recall* e *F1-Score*). Isto pode ocorrer principalmente por dois motivos, o

[5] https://colab.google

fato de a quantidade de dados ser limita com apenas 229 amostras, e os dados estarem desbalanceados, com 184 entradas para estudantes que concluíram e 45 para estudantes evadidos. Diante da limitação dos modelos em relação ao *recall* considerou-se necessário estudar outras abordagens e realizar testes para aferir se é possível melhorar essas métricas.

3.2 Modelo de *prototypical network*

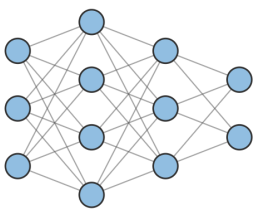
O modelo também foi desenvolvido na plataforma Google Colaboratory, utilizando a linguagem de programação Python 3, com a biblioteca PyTorch[6] como base. PyTorch foi escolhida por sua flexibilidade, permitindo a adaptação e desenvolvimento do modelo de *Prototypical Networks*, uma vez que ele não é implementado de forma nativa nas principais bibliotecas de *Machine Learning*. Além disso, algumas ferramentas e métodos do TensorFlow e Keras foram usados em paralelo para complementar o processo.

Para otimizar a definição dos parâmetros de treinamento, evitando abordagens por tentativa e erro, foi implementado um algoritmo de *grid search*. Esse algoritmo busca identificar os melhores hiperparâmetros, garantindo resultados mais eficientes e precisos no treinamento. Alguns *encoders* também foram testados para descobrir o quão relevante para o problema seria o modelo da rede utilizado. O *encoder* é uma rede neural responsável por transformar os dados de entrada (como imagens, textos ou outras informações) em uma representação vetorial ou *embedding* de dimensão mais baixa. Por se tratar de um modelo de classificação, de dados tabulares existem alguns possíveis candidatos a *encoder*, mas para o caso do problema apresentado, um modelo do tipo MLP foi escolhidio, pois é um modelo facil de ser implementado, além de ter sido um dos modelos analisados sem a utilização de PN por Nascimento (2022).

O *encoder* escolhido é composto por três camadas principais. A primeira camada é totalmente conectada (Linear) com n neurônios, seguida por uma normalização em lote (*BatchNorm*) para estabilizar o treinamento e uma ativação ReLU para introduzir não-linearidade. A segunda camada reduz o número de neurônios pela metade ($n/2$), novamente usando *BatchNorm*, e adiciona uma camada de *Dropout* com uma taxa de $0{,}3$ para evitar *overfitting*. Finalmente, a terceira camada reduz a dimensionalidade para m neurônios, preparando a saída final, com uma última normalização em lote para melhorar a estabilidade. Em todas estas redes $n,m,$ foram definidos pelo algoritmo de *grid search.* Na figura 3, apresenta-se um exemplo de MLP simplificada, que foi o *encoder* escolhido.

Para lidar com o problema do balanceamento de dados foi identificada a possibilidade de balancear os dados para este estudo, selecionando subconjuntos minoritários do *dataset* original e aplicando um *oversampling*, ou *undersampling* que iguala a quantidade de dados de evadidos com concluintes e assim verificar se estes melhoram os resultados.

Figura 3. Exemplo de rede MLP com duas camadas, utilizado que pode ser utilizado como encoder, o encoder testado segue esta arquitetura porém o número de neurônios de entrada é 15 , a quantidade de camadas foi maior e quantidades de neurônios por camada varia de acordo com os parâmetros do *grid search.*



Fonte: Autoria Própria (2024)

[6] https://pytorch.org/docs/stable/index.html

Após a definição do encoder, a *Prototypical Network* (PN) foi baseada na implementação de *apoorvnandan* (GitHub)[7] e no trabalho de Snell *et al.* (2017). Foram necessárias algumas alterações, entre as quais destaca-se a alteração na forma que os *batches* são criados, pois o código original foi feito para funcionar apenas com imagens do dataset Omniglot. As alterações consistiram na forma com que os *batches* usam os dados, deixando de percorrer a matriz tridimensional da imagem (largura, altura e RGB) e percorrendo apenas o *dataset* que consiste em linhas e colunas. A estrutura do modelo se aproxima de muitos outros modelos tradicionais de *Machine Learning*, com a diferença crucial de que, em cada ciclo de treinamento, as *embeddings* geradas pelo *encoder* são utilizadas para construir os protótipos. Esses protótipos representam as médias dos *embeddings* de cada classe no *support set*, servindo como referência para a classificação dos exemplos no *query set*.

Durante o desenvolvimento, foram testadas algumas técnicas de balanceamento do *dataset* e analisado o comportamento do modelo. As técnicas de *undersampling* e *oversampling* foram consideradas. O *undersampling* consiste em reduzir a quantidade de amostras da classe majoritária, podendo ser realizado por meio da seleção aleatória de quais amostras serão removidas (Shelke *et al.*, 2017). Por outro lado, o *oversampling* aumenta a quantidade de amostras da classe minoritária, replicando as amostras até que o *dataset* fique balanceado (Shelke *et al.*, 2017).

Para definir a melhor abordagem de utilização do dataset, foi realizado um cálculo do desvio padrão, comparando as três técnicas: *undersampling*, *oversampling* e o dataset original. O modelo foi executado 100 vezes com os mesmos parâmetros: encoder = MLP, 10 épocas, 7 nshot, 6 nquery, 256 hidden_dim (Quantidade de neurônios na camada escondida), 32 output_dim (Quantidade de neurônios na camada de saída) e 0,01 de learning rate (Taxa de aprendizagem ou lr). O valor de 6 nquery foi escolhido pois se mostrou o ideal depois de alguns testes. Com poucos nquery a rede não aprendia adequadamente e muitos nquery ultrapassaram a quantidade de dados disponíveis. Não foi executado o *grid search*, pois o objetivo deste teste foi analisar o comportamento do modelo com configurações constantes. Portanto, a acurácia obtida não representa o valor máximo possível, mas sim uma avaliação consistente do desempenho em diferentes cenários de balanceamento de dados. Na Tabela 2 são apresentados os resultados de cada configuração do *dataset* e nela vemos que que apesar do *oversampling* ter a melhor acurácia média, o desvio padrão mostrou uma instabilidade maior que utilizando *undersampling* ou sem nenhuma modificação. Por apresentar o menor desvio padrão, a maior acurácia comparado ao uso de *undersampling*, e permitir uma comparação mais justa com o trabalho de Nascimento (2022), o *dataset* sem alteração foi o escolhido para ser utilizado durante os testes da pesquisa.

Tabela 2 - Acurácia e desvio padrão de cada configuração do *dataset.*

DataSet Sample	Acurácia (%)	Desvio Padrão
Nenhum	82,03%	0.0115
Oversampling	88,20%	0.0303
Undersampling	81,80%	0.0156

Fonte: Autoria Própria (2024)

O motivo desta variação não foi estudado pois fugia do escopo deste trabalho de conclusão de curso. Entretanto é perceptível que está ligado a quantidade de dados utilizado para o treino e muito provavelmente está relacionado ao momento em que os *batches* são criados, pois eles selecionam os dados randomicamente para garantir que a rede generalize e

[7] https://apoorvnandan.github.io/2020/12/14/prototypical-networks/

não aprenda apenas com exemplos específicos. Dessa forma, é possível que ao criar um *oversampling* durante a criação dos *batches* gere a instabilidade dos modelos. É importante destacar que o desempenho das *Prototypical Networks* (PN) não depende apenas dos parâmetros do modelo, mas também da qualidade da média das *embeddings* geradas. Em muitos casos, um bom resultado pode ser alcançado com poucas épocas de treinamento, sendo que até mesmo uma única época já pode produzir resultados satisfatórios.

Ao executar o *grid search* para otimizar o modelo, testando diferentes parâmetros (conforme apresentado no Quadro 1), foi identificado que os *encoders* não exerceram grande influência nos resultados, basta ter um bom número de camadas e neurônios para obter boas *embeddings*. O modelo também foi executado utilizando validação cruzada *5k-folds* que será apresentada na análise estatística. No entanto, dentro da variação esperada, observou-se que parâmetros como o número de camadas ocultas e de camadas de saída apresentaram, de maneira geral, mudanças significativas. Apesar disso, os melhores parâmetros identificados no *grid search* podem variar ligeiramente entre as execuções. Essa variação ocorre devido à flutuação nos resultados de acurácia, mesmo quando os mesmos parâmetros são utilizados em execuções repetidas.

Quadro 1 - Parâmetros do Grid Search.

Parâmetro	Array de valores
Learning rate	[0,005; 0,01]
Hidden Dim	[64; 128; 256]
Output Dim	[16; 32; 64]
N shot	[2; 5; 7]

Fonte: Autoria Própria (2024)

No Quadro 2 são exibidos a acurácia e os parâmetros utilizados na pior e melhor execução do modelo no *grid search*. O encoder alcançou uma boa acurácia nas três execuções registradas. No entanto, ainda é perceptível uma certa instabilidade nos resultados. Embora as acurácias tenham sido semelhantes, os melhores parâmetros variaram em algumas execuções, assim como os piores. Isso é um indicativo de que o modelo, em algumas situações, apresenta métricas melhores ou piores para os mesmos parâmetros, o que corrobora com os dados observados no Tabela 2. O *grid search* foi executado três vezes para entender como a instabilidade modifica os resultados. A base não passou por *undersampling* ou *oversampling* se mantendo original, o melhor resultado foi utilizado no resto da pesquisa.

Quadro 2 - Acurácia e o melhor parâmetro para encoder.

Execução	Melhor Acurácia (%)	Pior Acurácia (%)	Melhor Parâmetro	Pior Parâmetro
1°	88,36%	71,21%	{'lr': 0.01, 'nshot': 7, 'hidden_dim': 128, 'output_dim': 64, 'dropout_rate': 0.3}	{'lr': 0.005, 'nshot': 2, 'hidden_dim': 64, 'output_dim': 64, 'dropout_rate': 0.3}
2°	87,48%	72,78%	{'lr': 0.01, 'nshot': 7, 'hidden_dim': 128, 'output_dim': 16, 'dropout_rate': 0.3}	{'lr': 0.005, 'nshot': 2, 'hidden_dim': 128, 'output_dim': 16, 'dropout_rate': 0.3}
3°	87,91%	72,83%	{'lr': 0.01, 'nshot': 7, 'hidden_dim': 256, 'output_dim': 16, 'dropout_rate': 0.3}	{'lr': 0.005, 'nshot': 2, 'hidden_dim': 128, 'output_dim': 32, 'dropout_rate': 0.3}

Fonte: Autoria Própria (2024).

3.3 Análise estatística entre os modelos de Nascimento e a prototypical network.

O teste de Friedman é um teste estatístico não paramétrico utilizado para detectar diferenças significativas entre múltiplos grupos em um experimento com medições repetidas (DEMŠAR, 2006). Ele é especialmente útil quando os dados não seguem uma distribuição normal, o que é comum em comparação a modelos de aprendizado de máquina. A hipótese nula (H0) do teste de Friedman afirma que não há diferença significativa entre os grupos, ou seja, todos os modelos apresentam desempenho semelhante. A hipótese alternativa (H1) indica que pelo menos um modelo apresenta um desempenho diferente dos demais.

Após a aplicação do teste de Friedman, se a hipótese nula for rejeitada, o teste de Nemenyi (DEMŠAR, 2006) pode ser realizado como um pós teste para identificar quais grupos (modelos) diferem entre si. O pós-teste de Nemenyi compara todas as combinações possíveis de pares de modelos, permitindo identificar em quais modelos ocorrem as diferenças significativas.

Neste estudo, utilizaremos métricas como acurácia, *precision* , *recall* e *F1-score* para avaliar o desempenho dos modelos, proporcionando uma análise detalhada das suas capacidades em classificar corretamente os dados.

Os modelos utilizaram o *dataset* original com em 5-*fold* cross-validation, o que significa que 80% dos dados foram utilizados para treino e 20% para validação. Validação cruzada, usada para avaliar o desempenho de modelos de *machine learning* (Ramezan *et al.*, 2019).

Após a execução do algoritmo em *5K-folds* a média das métricas resultantes foram utilizadas nos testes de Friedman e Nemenyi. Os testes foram feitos utilizando python na ferramenta Google Colaboratory e com a biblioteca *scikit-posthocs*[8].

4 RESULTADOS E DISCUSSÕES

Após o desenvolvimento do modelo, foram calculadas as diferentes métricas obtidas em cada execução, foram utilizados os modelos com a mesma configuração fornecida por Nascimento (2022). Devido a instabilidade ao utilizar as técnicas de *undersampling* e *oversampling,* foi mantido o dataset original, permitindo uma melhor comparação com os modelos reproduzidos, garantindo que ambas as classes não tivessem repetição exagerada de

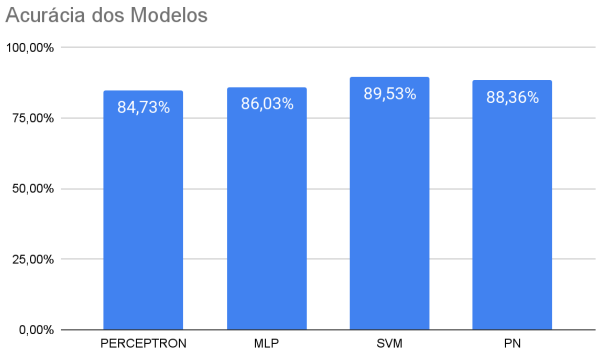
[8] https://scikit-posthocs.readthedocs.io/en/latest/

exemplos. Essa abordagem contribuiu para uma avaliação mais precisa do desempenho do modelo, refletindo sua eficácia em diferentes cenários. Os resultados analisados foram obtidos utilizando o *grid search* e *5k-folds* para todos os modelos, foi realizada a média das métricas obtidas pelos *5k-folds*. As métricas diferem da Tabela 1 pois foi necessário modificar a quantidade de *folds* dos modelos reproduzidos para se ter uma comparação justa.

4.1 Acurácia dos modelos

A primeira métrica a ser avaliada é a acurácia. Os modelos utilizam *prototypical networks,* atingiram valores próximos ou melhores que o SVM, modelo que apresentou o melhor resultado no estudo de Nascimento (2022) que pode ser visto na figura 4.

Figura 4 - Gráfico de barras com a acurácia dos modelos reproduzidos e a PN.



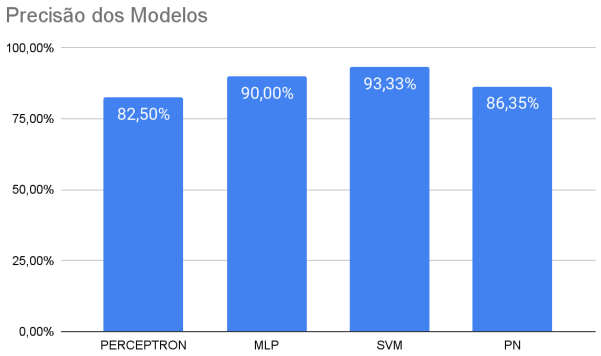
Fonte: Autoria Própria (2024)

A acurácia alta do modelo PN era prevista, tendo em consideração que as técnicas de *few-shot leaning* tem como objetivo atingir boas métricas com poucos exemplos.

4.2 Precisão dos modelos

A segunda métrica avaliada é a Precisão que também teve resultados muito próximos aos obtidos por Nascimento (2022). O resultado dessa métrica pode ser observado na figura 5.O modelo PN teve uma precisão acima do *Perceptron* e do MLP e se aproximou do modelo que usa SVM.

Figura 5 - Gráfico de barras com a precisão dos modelos reproduzidos e a PN.

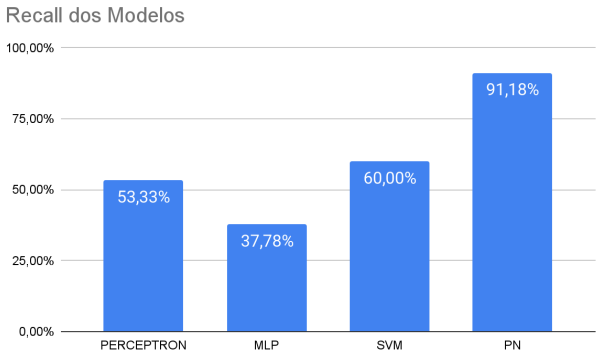


Fonte: Autoria Própria (2024)

4.3 Recall dos modelos

A terceira métrica analisada foi o *recall*, que mede a capacidade do modelo de identificar corretamente as instâncias positivas. O principal desafio enfrentado pelos modelos de Nascimento (2022) foi o baixo desempenho nessa métrica, o que pode ser visto na figura 6, refletindo a dificuldade em detectar todos os casos positivos, o que pode comprometer sua eficácia em aplicações nas quais a identificação dos positivos é fundamental.

Figura 6 - Gráfico de Barras com o recall dos modelos reproduzidos e a PN.



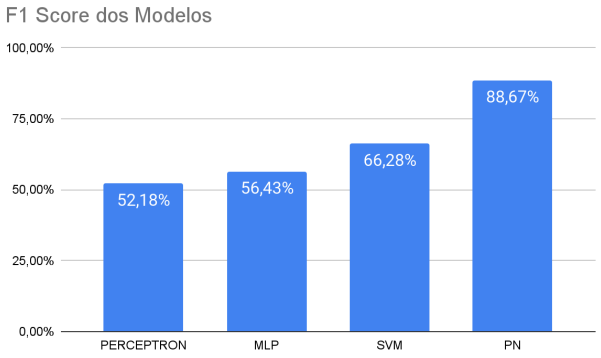
Fonte: Autoria Própria (2024)

Como mostrado na Figura 6, os modelos que utilizam *few-shot learning* com a abordagem de *Prototypical Networks* apresentaram um *recall* substancialmente superior aos modelos de Nascimento (2022). Enquanto o maior *recall* obtido por Nascimento foi de 60%, o modelo baseado em *Prototypical Networks* PN alcançou mais de 90% em seus melhores cenários. Isso significa que o novo modelo foi mais eficaz na identificação correta das instâncias positivas, tornando-os uma escolha promissora para cenários nas quais a detecção de todos os casos positivos é fundamental.

4.4 F1 Score dos modelos

A quarta métrica analisada é o *F1-score*, que combina *precision* e recall em um único valor para avaliar o equilíbrio entre a identificação correta de casos positivos e a minimização de falsos positivos. Nos modelos tradicionais de *Machine Learning*, o *F1-score* apresentou resultados relativamente baixos como mostrado na Figura 7, isso indica que, embora o modelo tenha conseguido um bom equilíbrio, ainda há margem para melhorar a capacidade de identificar corretamente os casos positivos sem aumentar os erros.

Figura 7 - Gráfico de barras com o F1-Score dos modelos reproduzidos e a PN.



Fonte: Autoria Própria (2024)

Ao analisar o modelo que utiliza *Prototypical Networks* (PN), é possível observar que foi obtida uma boa taxa de *F1-score*, em torno de 88%, conforme mostrado na Figura 7. Esse resultado está diretamente ligado às altas taxas de recall alcançadas pelos modelos.

4.5 Resultado estatístico

O teste de Friedman foi realizado para comparar o desempenho de quatro modelos (Perceptron, MLP, SVM e PN-MLP) com base nas médias de diferentes métricas obtidas em 5 folds. A estatística do teste resultou em um valor de 8,09, indicando a magnitude da variação entre os desempenhos dos modelos. O p-valor obtido foi de 0,0439, o que é menor que o nível de significância tradicional de 0,05. Como o p-valor é menor que 0,05, podemos rejeitar a hipótese nula (H0), que afirma que não há diferenças significativas entre o desempenho médio dos modelos. Isso significa que, com os dados fornecidos, há evidências suficientes para concluir que existe uma diferença significativa entre algum dos modelos nas métricas avaliadas. O teste post-hoc de Nemenyi, que analisa comparações par a par, identificou diferenças significativas entre pares específicos de modelos, indicando que ao menos um modelo se destaca em relação aos outros.

Na Tabela 3, resultante do teste post-hoc de Nemenyi, são apresentadas as comparações de desempenho entre os quatro modelos avaliados, mostrando a similaridade entre eles em termos de suas métricas. Os valores variam de 0 a 1. 1 indica uma correspondência perfeita e 0 representa nenhuma similaridade.

Tabela 3 - Resultado do pós teste de Nemenyi

	Perceptron	MLP	SVM	PN
Perceptron	1,000000	0,844289	0,065541	0,125707
MLP	0,844289	1,000000	0,354318	0,518694
SVM	0,065541	0,354318	1,000000	0,992827
PN	0,125707	0,518694	0,992827	1,000000

Fonte: Autoria Própria. (2024)

No geral, a tabela reforça a conclusão do teste de Friedman, que indica que há diferenças significativas entre os modelos, uma vez que muitos deles apresentam alta correlação em seus desempenhos. Os valores de p acima de 0,9 entre os modelos SVM e PN (0,99), sugerem que esses pares de modelos têm desempenhos muito semelhantes. Em contraste, os pares Perceptron e SVM ou Perceptron e PN indicam uma menor similaridade, com valores de p de 0,065 e 0,126, respectivamente. Portanto, o teste post-hoc de Nemenyi

identificou diferenças significativas entre alguns pares. Os modelos SVM e PN podem ser considerados ligeiramente superiores em termos de desempenho em comparação aos outros.

5 CONCLUSÕES

Foi desenvolvido um modelo de FSL com *prototypical network* que foi capaz de aprender com a pouca quantidade de dados disponíveis para estudo com este modelo pronto, foi feita a comparação entre ele e os demais modelos previamente testados. A questão principal desta pesquisa é: como os modelos baseados em FSL se comparam com modelos comuns de ML para identificar alunos com risco de evasão a partir de um pequeno conjunto de dados de treino? Com o resultado das métricas obtidas com o novo modelo e a comparação estatística com os outros modelos previamente estudados foi possível responder esta questão.

Após desenvolvimento e implementação da *Prototypical Network*, foi possível detectar uma melhora nos resultados das métricas de *recall* e *f1-score* até mesmo nos piores resultados obtidos da rede. A acurácia e *precision* não tiveram grandes saltos em consideração aos modelos comuns de ML, ambos tiveram bons resultados e mantiveram bons valores.

Alguns problemas e comportamentos foram detectados durante o desenvolvimento, como a instabilidade nos resultados do modelo. Isso resultou em acurácias diferentes em execuções seguidas com os mesmos parâmetros. Essas variações podem ter várias causas, mas estão principalmente relacionadas ao *dataset*, já que a falta de balanceamento ainda pode ser um problema. Mesmo com a aplicação de técnicas de *oversampling* e *downsampling*, houve maior inconsistência ou piora nas métricas em algumas execuções.

Com isto em mente, a técnica de *few-shot learning* com *prototypical network* se mostrou bastante promissora para resolver o problema de evasão de alunos do curso de Ciência e Tecnologia da UFERSA e pode ser uma aliada para prever quando alunos do curso irão evadir, mas a pequena inconsistência no resultado dos modelos levanta pontos que podem ser melhorados e analisados a fundo.

5.1 Trabalhos futuros

Como trabalhos futuros, o estudo de diferentes abordagens de *few-shot learning* podem ser analisadas, como Modelos de *Meta-Learning*, modelos baseados em Memória *(Memory-Augmented Networks)* dentre outros. Adicionalmente, é possível estudar outras configurações do dataset afim e qual seus impactos em cada modelo e outros cálculos de distância utilizados no modelo.

REFERÊNCIAS

AGRUSTI, F.; BONAVOLONTÀ, G.; MEZZINI, M. University Dropout Prediction through Educational Data Mining Techniques: A Systematic Review. **Journal of e-Learning and Knowledge Society**, v. 15, n. 3, p. 161-182, 12 Oct. 2019. Disponível em: https://je-lks.org/ojs/index.php/Je-LKS_EN/article/view/1135017. Acesso em: 5 out. 2024

CHAVES, Aleson G. S. *et al*. Extração de entidades de produtos utilizando técnicas de few-shot learning. In: **Simpósio Brasileiro de Automação Inteligente-SBAI, XV**, 2021. Disponível em: https://sba.org.br/open_journal_systems/index.php/sbai/article/view/2771/2314. Acesso em: 20 out. 2024.

CHAVES, Aleson G. S. *et al*. Classificação de Produtos Utilizando Técnicas de Few-Shot Learning. In: **Congresso Brasileiro de Automática-CBA, XXIV**, 2022. Disponível em: https://www.sba.org.br/open_journal_systems/index.php/cba/article/view/3229/2757. Acesso em: 20 out. 2024.

DEMŠAR, Janez. Statistical comparisons of classifiers over multiple data sets. **The Journal of Machine learning research**, v. 7, p. 1-30, 2006. Disponível em: https://www.jmlr.org/papers/volume7/demsar06a/demsar06a.pdf. Acesso em: 29 out. 2024.

FINN, Chelsea; ABBEEL, Pieter; LEVINE, Sergey. Model-agnostic meta-learning for fast adaptation of deep networks. In: **International conference on machine learning. PMLR, 2017**. p. 1126-1135. Disponível em: https://proceedings.mlr.press/v70/finn17a/finn17a.pdf. Acesso em: 29 out. 2024.

GRANDINI, Margherita; BAGLI, Enrico; VISANI, Giorgio. Metrics for multi-class classification: an overview. **arXiv preprint arXiv:2008.05756**, 2020. Disponível em: https://arxiv.org/pdf/2008.05756. Acesso em: Acesso em: 24 out. 2024.

JANIESCH, Christian; ZSCHECH, Patrick; HEINRICH, Kai. Machine learning and deep learning. **Electronic Markets**, v. 31, n. 3, p. 685-695, 2021. Disponível em: https://link.springer.com/article/10.1007/s12525-021-00475-2. Acesso em: 24 out. 2024.

KOCH, Gregory et al. Siamese neural networks for one-shot image recognition. In: **ICML deep learning workshop**. 2015. p. 1-30. Disponível em: https://www.cs.utoronto.ca/~rsalakhu/papers/oneshot1.pdf. Acesso em: 22 out. 2024.

MARQUES, L. T., De Castro, A. F., Marques, B. T., Silva, J. C. P., & Queiroz, P. G. G. (2019). Mineração de dados auxiliando na descoberta das causas da evasão escolar: Um mapeamento sistemático da literatura. **Revista Novas Tecnologias na Educação**, 17(3), 194-203. Disponível em: https://doi.org/10.22456/1679-1916.99470. Acesso em: 10 out. 2024.

MARQUES, L. T., Marques, B. T., Rocha, R. S., Chaves, L., de Castro, A. F., Queiroz, P. G. G. Evasão acadêmica e suas causas em cursos de bacharelado em ciência da computação: um estudo de caso na UFERSA. *In* CONGRESSO BRASILEIRO DE INFORMÁTICA NA EDUCAÇÃO, IX, 2020, Natal RN. **Anais [...]**, Natal RN: SBC, 2020. Disponível em: https://www.researchgate.net/profile/Angelica-Castro-4/publication/347148752_Evasao_Academica_e_suas_Causas_em_Cursos_de_Bacharelado_em_Ciencia_da_Computacao_Um_Estudo_de_Caso_na_UFERSA/links/63f9883f0cf1030a564bc310/Evasao-Academica-e-suas-Causas-em-Cursos-de-Bacharelado-em-Ciencia-da-Computacao-Um-Estudo-de-Caso-na-UFERSA.pdf. Acesso em: 20 out. 2024.

NASCIMENTO, F. F. do. **Implementação e avaliação de diferentes modelos de machine learning aplicados à predição de estudantes em risco de evasão estudantil em diferentes cursos do ensino superior.** 2022. 145 f. Dissertação (Mestrado em Ciência da Computação) – Universidade Federal Rural do Semi-Árido; Universidade do Estado do Rio Grande do Norte, Mossoró, 2022. Disponível em: https://ppgcc.ufersa.edu.br/wp-content/uploads/sites/42/2023/04/FernandaFerreiraDoNascimento-dissertacao.pdf . Acesso em: 15 de julho de 2024.

NASCIMENTO, F. F. do; DANTAS, L. C. de O.; CASTRO, A. F. de; QUEIROZ, P. G. G. Técnicas de Mineração de Dados e Aprendizado de Máquina aplicados à Evasão Estudantil: um mapeamento sistemático da literatura. **Revista Brasileira de Informática na Educação**, [S. l.], v. 32, p. 270–294, 2024. DOI: 10.5753/rbie.2024.3296. Disponível em: https://journals-sol.sbc.org.br/index.php/rbie/article/view/3296. Acesso em: 20 set. 2024.

OECD. **Education at a Glance 2023:** OECD Indicators. Paris: OECD Publishing, 2023. Disponível em: https://doi.org/10.1787/e13bef63-en. Acesso em: 16 out. 2024.

PARNAMI, Archit; LEE, Minwoo. Learning from few examples: A summary of approaches to few-shot learning. **arXiv 2022. arXiv preprint arXiv:2203.04291**. Disponível em: https://arxiv.org/abs/2203.04291. Acesso em: 29 out. 2024

RAMEZAN, A.; WARNER, Timothy A.; MAXWELL, Aaron E. Evaluation of sampling and cross-validation tuning strategies for regional-scale machine learning classification. **Remote Sensing**, v. 11, n. 2, p. 185, 2019. Disponível em: https://www.mdpi.com/2072-4292/11/2/185. Acesso em: 29 out. 2024.

SHELKE, Mayuri S.; DESHMUKH, Prashant R.; SHANDILYA, Vijaya K. A review on imbalanced data handling using undersampling and oversampling technique. Int. **J. Recent Trends Eng**. Res, v. 3, n. 4, p. 444-449, 2017. Disponivel em: https://d1wqtxts1xzle7.cloudfront.net/80622764/a-review-on-imbalanced-data-handling-using-undersampling-and-oversampling-technique-libre.pdf?1644642246=&response-content-disposition=inline%3B+filename%3DA_Review_on_Imbalanced_Data_Handling_Usi.pdf&Expires=1730474478&Signature=fx1cYlSscmBhPp2WJVt1EUTSpVjofwX5F-GdS1f5NQvvdhpVcGpweMwOY9f8uWxy1u2ULxHl13WIQC7-eGf7I~GV9VNf0LyaMCzRn6rxNXq5c8bfLnkI5ajEAebwNnHjRLyE6EmqW3B8J5PTPTVxNLe1SAIlEgSjvZ0FEzDgwfcurzbGz4rxF5RaOSC48OryIiJs2HYTGAL1nvaqkmgMuvGR~UjDSaBMGTUu7fbEdj~XfmJwnEOZsWgyFxLFRX4WH6-NP1ScVXbQXE1copegi8vDD4woqXU1izn5p33TJo0MH~oZd5HnaLgZeJ4kIfRwcJzDvDlSg2L6VwiVrv9dew__&Key-Pair-Id=APKAJLOHF5GGSLRBV4ZA. Acesso em: 29 out. 2024.

SNELL, Jake; SWERSKY, Kevin; ZEMEL, Richard. Prototypical networks for few-shot learning. **Advances in neural information processing systems**, v. 30, 2017.Disponivel em:https://proceedings.neurips.cc/paper_files/paper/2017/file/cb8da6767461f2812ae4290eac7cbc42-Paper.pdf. Acesso em: 5 out. 2024.

SUN, Qianru et al. Meta-transfer learning for few-shot learning. In: **Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.** 2019. p. 403-412. Disponível em: https://openaccess.thecvf.com/content_CVPR_2019/papers/Sun_Meta-Transfer_Learning_for_Few-Shot_Learning_CVPR_2019_paper.pdf. Acesso em: 28 out. 2024.

VINYALS, Oriol et al. Matching networks for one shot learning. **Advances in neural information processing systems,** v. 29, 2016. Disponível em: https://proceedings.neurips.cc/paper_files/paper/2016/file/90e1357833654983612fb05e3ec9148c-Paper.pdf. Acesso em: 29 out. 2024.