UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
GRADUAÇÃO EM ENGENHARIA DA COMPUTAÇÃO

Redes Abstratas Profundas para Dados Tabulares: Técnicas e seus Desempenhos.

MARIA EDUARDA BARROS MOTA

RECIFE
2023

MARIA EDUARDA BARROS MOTA

REDES ABSTRATAS PROFUNDAS PARA DADOS TABULARES: TÉCNICAS E SEUS DESEMPENHOS.

Monografia apresentada ao curso de Engenharia da
Computação do Centro de Informática da Universidade Federal de Pernambuco, como requisito parcial para obtenção do Título de Bacharel em Engenharia da Computação.

Orientador: Prof. Germano Crispim Vasconcelos

RECIFE
2023

Ficha de identificação da obra elaborada pelo autor,
através do programa de geração automática do SIB/UFPE

Mota, Maria Eduarda Barros.
Redes abstratas profundas para dados tabulares: técnicas e seus desempenhos / Maria Eduarda Barros Mota. - Recife, 2023.
41 p. : il., tab.

Orientador(a): Germano Crispim Vasconcelos
Trabalho de Conclusão de Curso (Graduação) - Universidade Federal de Pernambuco, Centro de Informática, Engenharia da Computação - Bacharelado, 2023.

1. Aprendizado profundo. 2. Dados tabulares. 3. Aprendizado de máquina. I. Vasconcelos, Germano Crispim. (Orientação). II. Título.

000 CDD (22.ed.)

Trabalho de conclusão de curso apresentado **Maria Eduarda Barros Mota** ao programa de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **Redes Abstratas Profundas para Dados Tabulares: Técnicas e seus Desempenhos**, orientado pelo Prof. Germano Crispim Vasconcelos e aprovada pela banca examinadora formada pelos professores:

_____

Prof. Alex Sandro Gomes
Centro de Informática/UFPE

_____

Prof. Germano Crispim Vasconcelos
Centro de Informática/UFPE

RECIFE
2023

AGRADECIMENTOS

Em primeiro lugar, gostaria de expressar minha gratidão a Deus e à Virgem Maria, que me iluminaram e protegeram durante toda a minha caminhada, além de trazerem paz e serenidade para o meu coração durante os obstáculos enfrentados.

Meus sinceros agradecimentos, também, à minha amada família, que sempre me apoiou incondicionalmente, confiando sempre na minha capacidade. Agradeço aos meus pais, Margarida e Luis Gustavo, que são exemplos para mim, e por me proporcionarem todo o suporte e apoio para que eu me desenvolvesse como pessoa e como profissional, aos meus irmãos Bruno e Maria Amélia, por serem meus melhores amigos e me acompanharem em todas as etapas da minha vida. Também agradeço à minha avó e madrinha, Maria José, que sempre foi presente e cuidou de mim, e à minha tia-avó Maria do Carmo, que me acompanhou desde pequena em etapas para o meu desenvolvimento.

Aos meus queridos amigos, das diversas áreas da minha vida, expresso minha gratidão por todas as orientações, apoio e partilha de experiências. Vocês foram fundamentais para que essa trajetória fosse mais leve, divertida e de muito aprendizado. Às minhas amigas Luisa Cavalcante, Taíse Araújo e Maria Luisa, meu profundo agradecimento pela ajuda e motivação, vocês são inspirações para mim.

Agradeço ao professor Germano Vasconcelos, que admiro e é uma referência para mim, com uma trajetória incrível na área de atuação que desenvolvi grande interesse ao passar do tempo. Suas orientações foram fundamentais para a execução desse trabalho, além de me permitir expandir meu olhar sobre o tema. Também agradeço aos membros da banca, pela atenção e por fazerem parte desta fase tão importante da minha vida.

Agradeço a todos os professores que tive a honra de aprender e trabalhar ao longo desta jornada acadêmica, vocês foram muito importantes para o meu crescimento, trazendo diferentes perspectivas e novos desafios.

Ao Centro de Informática, expresso meu imenso orgulho e gratidão por proporcionarem uma estrutura e oportunidades ímpares durante minha formação, além da excelente formação acadêmica durante todos esses anos.

A todos que tive e tenho a felicidade de compartilhar a vida, meu profundo agradecimento.

RESUMO

Dados de tipo tabular para mapeamento e documentação de informações sempre possuíram uma forte presença em diversas áreas da sociedade, e estão ainda mais numerosos dado o avanço da tecnologia. Como consequência, tem-se um aumento no desenvolvimento de pesquisas e modelos voltados para a utilização e maior domínio dessas informações.

Por possuir uma menor complexidade computacional e bons resultados, a predominância da atuação de modelos mais tradicionais de *Machine Learning* para a resolução de problemas com dados tabulares ficou cada vez mais expressiva, gerando, assim, uma certa disparidade na pesquisa e desenvolvimento de outros possíveis métodos que poderiam performar em equidade ou até de maneira mais efetiva em contextos de Classificação e Regressão. Assim, a partir desse cenário, torna-se necessário uma maior visibilidade e entendimento acerca de diferentes modelos, com o objetivo de buscar alternativas mais assertivas na resolução de determinadas tarefas.

Desse modo, esse Trabalho de Graduação aborda a aplicação de Redes Abstratas Profundas (DANets) em Dados Tabulares, trazendo uma explicação detalhada deste modelo de *Deep Learning* para Dados Tabulares, juntamente com uma análise crítica acerca das premissas tomadas. Além disso, é desenvolvido um estudo comparativo com outros principais modelos usados para problemas de Classificação, como *MLP, Random Forest e Gradient Boosting* (além de ensembles mistos das estruturas).

Assim, é evidenciado neste trabalho que o modelo DANets apresenta uma boa performance no contexto testado, superando até alguns modelos mais tradicionais em uma tarefa complexa de classificação de dados, medida pela métrica estatística KS (*Kolmogorov-Smirnov Test*).

ABSTRACT

Tabular data for mapping and documenting information have always had a strong presence in various society domains, and they have become more abundant with the advance of technology. As a result, there has been an increase in the research and development of better models for dealing with this type of data.

Due to lower computational complexity and good results, the predominance of traditional Machine Learning models in solving tabular data problems has become more expressive, resulting in a certain disparity in the research and development of other potential methods that could perform equally or even more effectively in classification and regression contexts. In this scenario, there is a need for creating and evaluating more sophisticated models when seeking for more accurate solution in some specific tasks.

In particular, this undergraduate project addresses the application of Deep Abstract Networks (DANets) to Tabular Data (Chen et al. 2022), providing a detailed explanation of this Deep Learning model for tabular data, along with a critical analysis of the assumptions made. Additionally, a comparative study is conducted with other typical models used for classification problems, such as MLP, Random Forest, and Gradient Boosting (as well as mixed ensembles of these structures).

As a result, this work evidences that DANets performs well in the tested context, even outperforming some more traditional models in a complex data classification task, as measured by the Kolmogorov-Smirnov Statistical Test (KS).

SUMÁRIO

1. Introdução 7
   1.1. Motivação 9
   1.2.Objetivos
   1.3.Estrutura do Trabalho 10

2. Fundamentação Teórica 11
   2.1. Revisão da Literatura 11
   2.2. Dados Tabulares 11
   2.3. Aprendizagem de Máquina e Aprendizado Profundo 12
      2.3.1 *Machine Learning* 13
      2.3.2 Redes Neurais 14
      2.3.3 *Deep Learning* 15
   2.4. Problemas de Classificação 16
   2.5. Funções de Transformação 18

3. Redes Profundas DANets com Dados Tabulares 20
   3.1. Objetivos do Modelo 20
   3.2. Metodologia 22
      3.2.1 Estruturação do Modelo 22
   3.3. Implementação 25
      3.3.1. DANets 25
      3.3.2. Modelos Convencionais de *Machine Learning* 27
   3.4. Estruturação dos Testes 28

4. Comparação do Modelo DANets e Métodos Convencionais 32
   4.1.1. Comparação Teórica entre os Modelos 32
   4.1.2. Análise Comparativa dos resultados 33

5. Conclusão 38
6. Referências Bibliográficas 39

1 - INTRODUÇÃO

1.1 - Motivação

Desde os primórdios da sociedade, a utilização de tabelas com a finalidade de agrupar e organizar informações sempre foi uma alternativa muito empregada, sendo as tábuas um dos primeiros documentos escritos da história, que possuíam a finalidade de registrar e contar peças como jarras e grãos.[37] No entanto, é fácil notar que, com o passar do tempo, a quantidade de informações armazenadas aumentou de maneira exponencial, sendo ainda mais importante uma ênfase na visão analítica, melhores mapeamentos e documentações de fluxos, comportamento e atividades, além de uma maior atenção em relação a análises e tratamento desses dados. [5] [13]

Essas informações organizadas em colunas e linhas, comumente chamadas de 'Dados Tabulares', estão presentes em diversas áreas de atuação. Setores da Ciência, Negócios, Finanças, Educação, e tantos outros, utilizam esses tipos de dados com o objetivo de se obter um maior controle das informações, e, a partir disso, desenvolver análises mais profundas. Com isso, pesquisas e desenvolvimentos de modelos que utilizavam dados tabulares como matéria prima se iniciaram de maneira mais intensa, a fim de trazer soluções, classificações e novos meios de análises com esses dados. [14]

Como Dados Tabulares normalmente possuem informações heterogêneas e não contínuas, soluções utilizando *Machine Learning* começaram a ganhar destaque como abordagens mais eficazes para lidar com os problemas envolvendo informações mais irregulares. Por possuírem propriedades que atenuam o impacto de dados ruidosos e, também, por se ter a possibilidade de um aprendizado conjunto e gradual, alguns modelos se tornaram mais utilizados nesse contexto. [9] [4] [5]

Em paralelo a isso, pesquisas e desenvolvimentos de modelos de *Deep Learning* começaram a ter um grande avanço em áreas como Visão Computacional, Processamento de Linguagem Natural (NLP), Reconhecimento de Fala, Análise de Sentimento, entre outras. A partir dos resultados desses estudos, as aplicações de Aprendizagem Profunda se mostraram extremamente eficientes e capazes de possuir uma maior flexibilidade e capacidade de formação de estruturas mais complexas para a solução de problemas. [1] [4] [21] [12]

Desse modo, partindo desse fato, é gerado um questionamento: por que, no contexto de problemas de classificação e regressão de informações dispostas em dados tabulares, são poucas as abordagens que utilizam *Deep Learning* como principal método?

Sumarizando algumas das principais características de se utilizar *Deep Learning* para a resolução de determinadas tarefas, existe uma relação direta entre volume de dados e performance. Assim, indo em oposição ao comportamento de muitos outros modelos, quanto

mais dados são dispostos para modelos de *Deep Learning*, melhor tende a ser o desempenho alcançado. Assim, pensando-se no contexto de dados tabulares com um volume expressivo de entradas, essa particularidade pode ser benéfica. Outra característica marcante da performance de arquiteturas de Aprendizado Profundo é a capacidade de adaptação dos modelos a depender do contexto que é dado. Com estruturas interligadas das mais diversas formas, torna-se possível uma maior adequação ao desafio proposto. Além disso, durante o processo de treinamento são realizados diversos ajustes dos parâmetros mediante as marcações de erros em cada iteração, fazendo com que o modelo consiga se adaptar ao longo do tempo, trazendo um resultado cada vez melhor. [16] [6] [4]

No entanto, apesar dos diversos pontos positivos para o desenvolvimento de modelos de *Deep Learning*, com um processo de entendimento e avaliação de pesquisas e testes prévios, fica claro que, atualmente, poucos modelos regulares de Aprendizado Profundo possuem uma boa eficiência quando se trata de dados tabulares como entrada. Uma das características principais que evidenciam um maior distanciamento dessa ótima performance é a dificuldade desses de lidar com a forte característica heterogênea e, na maioria das vezes, esparsa em informações organizadas em tabelas. [21] [9] [1] [4]

Desse modo, levando em conta as características positivas e outras não desejáveis para a aplicação desse tipo de algoritmo em dados em tabela, esta pesquisa visa a explorar os pontos fortes que podem ser aproveitados para esse tipo de problema, além de identificar algumas ações que podem atenuar possíveis limitações na performance geral de modelos de *Deep Learning*. Um dos principais pontos de expectativa foi a possibilidade de que, com a otimização adequada de hiperparâmetros, juntamente com uma modelagem apropriada, a performance de um modelo de *Deep Learning* para dados tabulares possa superar métodos tradicionais de *Machine Learning*.

Somada à visão analítica, a pesquisa dá continuidade com um maior entendimento e teste de um modelo de Aprendizado Profundo chamado DANets (*Deep Abstract Networks*), que foi desenvolvido para solucionar problemas de Classificação e Regressão de Dados Tabulares. O entendimento detalhado dessa aplicação é importante para se fazer uma análise comparativa com os pontos positivos e negativos elencados em performances desses tipos de algoritmo. [1]

A aplicação do DANets é uma proposta de incorporar o uso de Redes Neurais e Aprendizado Profundo em problemas de Regressão e Classificação. Em uma visão geral, o DANets é uma Rede Abstrata Profunda, composta por Camadas Abstratas (chamadas de AbstLay). Uma das características distintivas da AbsLay reside na sua capacidade de executar algumas etapas de processamento mais complexas nos dados de entrada. Inicialmente, é realizado o cálculo de uma Máscara Aprendível com base nos dados de entrada, sendo uma

etapa fundamental para assegurar que o modelo possa atribuir diferentes graus de importância, ou seja, pesos variados, para os diferentes atributos presentes nos dados iniciais.

Com o intuito de aprofundar o entendimento e fortalecer a validação do modelo DANets, esse projeto traz uma série de testes com uma base de dados extensa e diversificada de Análise de Crédito de Clientes. A escolha dessa base se deu pela presença de informações reais (ainda que devidamente anonimizadas), e por possuir características bastante variadas, contendo dados categóricos e numéricos.

Inicialmente, tem-se a necessidade de realizar uma série de tratamento e avaliação dos dados, começando com uma limpeza de valores inconsistentes, seguida pela normalização dos dados e balanceamento das classes. Assim, após todo esse processo, é realizada uma divisão entre conjuntos de Treinamento, Validação e Teste. A partir dessa etapa, torna-se possível executar o modelo e analisar o comportamento de maneira mais detalhada.

A execução do modelo DANets foi conduzida utilizando a ferramenta *Optuna* [22], a fim de detectar a melhor combinação de hiperparâmetros para ser utilizada nesse contexto de Análise de Crédito. Durante esse processo, foram avaliados alguns parâmetros como o erro quadrático médio (MSE) ao longo dos conjuntos de teste e validação, além da análise da acurácia e do valor de KS. [21] [22] [8]

Também, com os mesmos conjuntos de Treinamento, Validação e Teste da base de Análise de Crédito, foram realizados testes com modelos mais tradicionais para problemas envolvendo classificação com Dados Tabulares como informação de entrada. Assim, a busca de melhores performances de modelos como *Random Forest* [35], *MLP* [36] e *Gradient Boosting* [35] foi igualmente realizada, a fim de, ao final, ter-se uma análise comparativa em relação ao desempenho de cada um comparado com o modelo DANets estudado. [9] [16] [5]

Desse modo, é possível obter uma compreensão mais abrangente acerca da capacidade dos modelos de Deep Learning em produzir resultados superiores ou equiparáveis aos modelos mais tradicionais.

1.2 - Objetivos

De modo geral, este trabalho visa a detalhar e desenvolver uma maior explicação acerca das vantagens e desvantagens nas abordagens para a resolução de problemas de Classificação e Regressão com Dados Tabulares, utilizando tanto com modelos de *Machine Learning* (que são mais tradicionais em abordagens com esse tipo de entrada) quanto com modelos de *Deep Learning* mais avançados.

Assim, após uma estruturação desses fatores, o entendimento mais aprofundado juntamente com o teste do modelo de Aprendizado Profundo chamado DANets (*Deep Abstract Networks*) traz uma avaliação mais prática de como é o real comportamento de um modelo de

Deep Learning em um contexto diferente do usual (como em Visão Computacional e Reconhecimento de Fala) para um contexto de Dados Tabulares. [1] [16]

Além disso, a comparação prática com o resultado de modelos de *Machine Learning* mais usuais para esse contexto, contribuindo, assim, para uma compreensão mais completa acerca das possíveis nuances que podem ser detectadas em diferentes abordagens para a resolução de um mesmo problema.

1.3 – Estrutura do Trabalho

Inicialmente, o projeto tem como principal intuito trazer uma base teórica acerca dos principais tópicos que serão abordados, além de ser escrito em uma ordem cronológica condizente com o que foi realmente realizado durante toda a pesquisa. Ao final da leitura, será possível entender a distinção de modelos de Machine Learning e Deep Learning e suas principais vantagens e desvantagens diante do cenário de resolução de problemas de Classificação com Dados Tabulares. Juntamente a isso, um maior entendimento do modelo de Deep Learning DANets será contemplado através da descrição do modelo e uma análise dos resultados dos experimentos realizados.

No Capítulo 2, será realizada uma **Revisão da Literatura**, a fim de entregar uma definição clara e concisa de termos importantes para a sequência do estudo, como a definição de Dados Tabulares, Aprendizagem de Máquina e Aprendizado Profundo. Além disso, um maior detalhamento de problemas de Classificação e Funções de Transformação torna-se necessário para facilitar o processo de estruturação e detalhamento do modelo DANets.

Dando seguimento, no Capítulo 3 (**Redes Profundas DANets com Dados Tabulares)**, vamos abordar com mais detalhes os objetivos e funcionamento do modelo DANets, além de explicar o processo de estruturação do mesmo. Por fim, tem-se uma maior elaboração acerca do processo de implementação, tanto do modelo estudado quanto dos modelos convencionais de Machine Learning, que vão ser utilizados para uma comparação, além de um maior detalhamento do processo dos Testes dos modelos escolhidos para avaliação.

No Capítulo 4, será realizado uma **Comparação do Modelo DANets e Métodos Convencionais** de Machine Learning para Dados Tabulares. A análise comparativa terá um viés tanto teórico quanto prático em relação aos modelos trabalhados.

Ao final, o Capítulo 5, **Conclusão**, apresenta uma sumarização das ideias dissertadas durante todo o projeto, além de sintetizar a análise dos resultados apresentados. Com isso, teremos uma visão mais clara de tudo o que foi abordado.

Também, é interessante pontuar que, durante toda a pesquisa, foram visitados diversos artigos, pesquisas e conteúdos informativos, a fim de se garantir um forte embasamento teórico nas abordagens do projeto.

2 – FUNDAMENTAÇÃO TEÓRICA

2.1– Revisão da Literatura

É de conhecimento geral que áreas como *Machine Learning* e *Deep Learning*, além de possuírem cada vez mais uma posição consolidada na resolução de problemas, estão em um progresso contínuo, com desenvolvimento de pesquisas e aplicações em muitas áreas de atuação. Com isso, foram revisitadas diversas pesquisas e artigos a fim de se obter um maior entendimento da presença e atuação de cada uma das áreas.

Estudos comparativos acerca de aplicações com Dados Tabulares também foram analisados com um maior nível de detalhe, a fim de se desenvolver um maior embasamento teórico, com visões e abordagens diversas, além de observações práticas acerca do tema discutido. Sem dúvidas, é um processo essencial para todo tipo de argumentação e tomada de decisões acerca do caminho para o desenvolvimento da pesquisa. [4] [5] [9] [21] [14] [12]

Este projeto tem como principal foco a área de *Deep Learning* e sua possível aplicabilidade em problemas envolvendo Dados Tabulares. Nessa etapa, os tópicos serão desenvolvidos para uma maior elucidação do contexto. Iniciando com a definição de Dados Tabulares, juntamente com Aprendizado de Máquina e Aprendizado Profundo, tem-se um maior cuidado para o entendimento e construção de uma análise crítica acerca da proposta da pesquisa.

2.2– Dados Tabulares

Dados tabulares nada mais são do que informações que estão organizadas em forma de tabelas. Assim, essa tabela possui diversas colunas, que são chamadas de atributos ou variáveis, e cada informação contida na tabela é um elemento. Desse modo, tem-se a organização dos dados a partir de linhas e colunas. Esse tipo de informação está cada vez mais presente em diversos setores e contextos no mundo. [5] [13] [14] [15]

O tipo de estrutura tabular é útil por organizar dados e informações de uma maneira intuitiva, ocupando o mínimo de espaço possível, e é aplicável para diversas ferramentas. Um dos principais objetivos de se ter a estruturação de dados em tabela é de minimizar a complexidade das informações, gerando, assim, uma maior praticidade ao comparar dados. Além disso, como o uso de dados tabulares diversas ações são possíveis: [15] [4]

- **Geração de Gráficos e Representações Visuais para os dados**: A partir dos dados em tabelas, é possível utilizar ferramentas para facilitar a visualização das informações, tornando-as mais interpretáveis e possibilitando um viés mais analítico.

- **Análise Exploratória dos Dados**: Gerar questionamentos, análise de tendências e agrupamento dos dados para buscar padrões, detecção de outliers, formulação de hipóteses, entre outras abordagens que podem ser realizadas para se extrair informações dos dados. [13]

- **Gestão de Informações e tomada de decisões**: É possível mapear fluxos, responsabilidades, tarefas, e, consequentemente, gerar decisões baseada em dados.

- **Modelagem Estatística**: Entendimento de tendências futuras, correlações entre variáveis, entre outros.

- **Criação de uma base histórica**: Armazenamento e organização dos dados passados.

- **Desenvolvimento de Modelos**: A partir das informações das tabelas, é possível se desenvolver modelos de predição, análise de comportamento, detecção de anomalias, entre outros.

Com isso, é notório que esse meio de organização e armazenamento de dados é muito utilizado, por se mostrar bastante útil e prático, e, por isso, é importante se ter cada vez mais avanços que permitam extrair mais valor dessa estrutura.

Por fim, é importante pontuar que existem diversos tipos de dados em tabelas. Dados numéricos (conhecidos também como dados qualitativos), dados categóricos, dados temporais, geoespaciais, entre outros. A partir dessas informações, é possível, depois de um tratamento adequado, estruturar e realizar aplicações mais condizentes para cada contexto que é dado.

Abaixo, temos um exemplo do que é uma estrutura de Dados Tabulares.

Cliente	Idade	Renda Mensal	Conta
Ana	30	8k	Corrente
João	67	12k	Corrente
Felipe	18	3k	Poupança
Amélia	35	34k	Premium
Pedro	40	18k	Corrente

Figura 1. *Exemplo de Dados Tabulares para um mapeamento clientes de um Banco, com dados numéricos (ex: Idade) e categóricos (ex: Conta).*

2.3– Aprendizagem de Máquina e Aprendizado Profundo

Inicialmente, é importante pontuar que há uma forte correlação entre Machine Learning, Redes Neurais e *Deep Learning*. Inicialmente, a área de *Machine Learning* engloba diversas técnicas e modelos que têm como principal objetivo o aprendizado e evolução em execução de tarefas ao longo do tempo. Existem modelos clássicos de *Machine Learning* que são

vastamente utilizados para resolução de problemas. Como o escopo da área de Aprendizagem de Máquina é grande, tem-se algumas subáreas com características mais idiossincráticas. [16]

Um modelo que permeia *Machine Learning* e é base para a estrutura das aplicações de *Deep Learning* é o modelo de Redes Neurais. Em uma visão geral, é um modelo computacional que é inspirado no cérebro humano, com o principal objetivo de realizar tarefas através de aprendizado de padrões (a partir dos dados pertencentes ao conjunto de treinamento). [6] [11]

Por fim, temos a área de *Deep Learning*, que é uma subárea de *Machine Learning*, voltada no desenvolvimento de Redes Neurais Profundas para a resolução de problemas. Modelos de Aprendizado Profundo possuem várias camadas de neurônios (que são uma unidade básica de uma Rede). [16]

Abaixo, é possível observar um diagrama da representação de como esses três pontos se conectam dentro do âmbito da Inteligência Artificial.

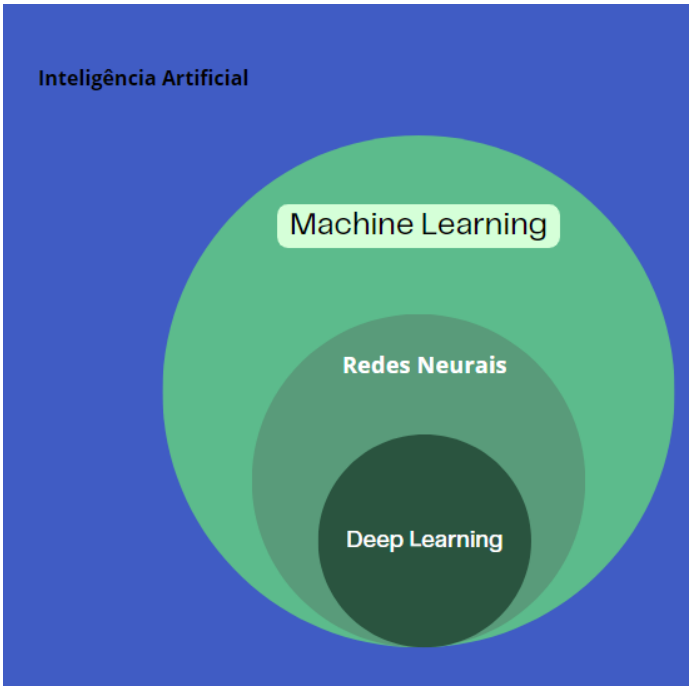


Figura 2. *Diagrama de Relação das Áreas, inseridas no contexto de Inteligência Artificial*

2.3.1– Machine Learning

Como disposto na Figura 2, temos *Machine Learning* como uma subárea de Inteligência Artificial. O principal objetivo dessa área é transformar a capacidade de uma máquina em inteligência para resolver problemas em diversas aplicações. Com isso, modelos de *Machine Learning* são capazes de, a partir das entradas recebidas, aprender com as informações passadas, identificando padrões e níveis de importância de cada dado, e, com o passar do tempo, melhorar sua performance na execução de tarefas determinadas. [6] [11] [17]

Diversos métodos de Aprendizado de Máquina são utilizados atualmente, sendo uma área que está em constante crescimento em diversos setores por possuir uma característica de adaptabilidade muito grande. A partir disso, tem-se modelos como Árvores de Decisão, *Gradient Boosting*, *Support Vector Machines* (SVM) que são muito empregados para a resolução de problemas atuais.

A resolução de problemas nesse contexto pode ser realizada de maneira Supervisionada ou Não Supervisionada. Com a Aprendizagem Supervisionada, tem-se o treinamento dos dados já previamente categorizados para o aprendizado das relações e, por fim, resolução de problemas como classificação e regressão. Já no contexto de Aprendizagem Não Supervisionada, o dado de entrada possui uma estrutura não ordenada, fazendo com que o Modelo de Aprendizagem tenha que detectar padrões, estabelecer métricas e identificar características mais importantes para a resolução de tarefas sem a interferência externa. [17]

Por fim, é importante pontuar que a presença de *Machine Learning* é essencial para a evolução e presença de Inteligência Artificial no desenvolvimento de soluções.

2.3.2– Redes Neurais

Uma Rede Neural, como o nome já expressa, é um conjunto de neurônios, agrupados de maneira que exista uma conexão entre eles. Em suma, é um modelo computacional que é inspirado no cérebro humano, com o principal objetivo de realizar tarefas através do aprendizado de padrões (a partir dos dados presentes no conjunto de treinamento) e até processar informações de uma maneira mais otimizada. Fazendo uma analogia completa, os nós fariam o papel de neurônios, e as conexões entre eles seriam as sinapses, como acontece no corpo humano. [6] [11]

A sua estrutura é composta por uma camada de entrada, uma ou mais camada intermediárias e uma camada de saída. Em cada camada, pode-se ter um ou mais nós como composição. Ou seja, com a informação de entrada, tem-se um conjunto de conexões, ajustes de peso e propagação das informações, a fim de que a informação processada seja compreendida corretamente pelo modelo, que vai realizar uma determinada atividade, como classificação, regressão, recomendação, entre outras. Na Figura 3 pode-se perceber que, em cada conexão entre nós, tem-se um peso específico associado, e limites impostos via função de ativação (que vão permitir que o nó seja ativado ou não).

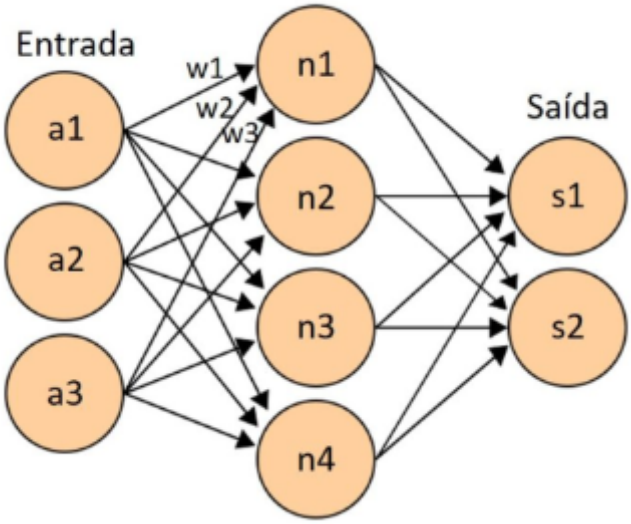


Figura 3. *Exemplo de uma Estrutura de uma Rede Neural Simples* **[6]**

Na imagem acima, o número de nós e camadas foi escolhido arbitrariamente, mas, essa quantidade é muito importante para formar um modelo capaz de contemplar o problema, dado um determinado contexto. Sendo assim, é necessário se fazer testes e validações dessas quantidades, a fim de buscar uma melhor performance sempre. Ou seja, não existe um número fixo ideal de camadas e nós, isso é uma variável que muda de acordo com o contexto do problema e a partir da estrutura da base de dados de entrada que é fornecida.

Com isso, existem dois tipos principais desse modelo: Redes Neurais Rasas e Profundas. Para o caso em que a Rede Neural possui poucas camadas intermediárias, por possuir menos complexidade, essa arquitetura demanda menos capacidade computacional para resolver problemas, mas possui uma certa limitação quando se trata de problemas mais complexos e robustos. No entanto, é uma ferramenta bastante útil e simples para situações adequadas para o contexto.

2.3.3– Deep Learning

A área de Aprendizado Profundo, que é também uma subárea de *Machine Learning*, possui uma grande variedade de arquiteturas de Aprendizado Profundo e Redes Neurais Profundas, sendo capaz de solucionar problemas complexos devido a sua robustez. Como o nome já diz, Redes Neurais Profundas são estruturas complexas e interligadas, com o objetivo final de aprender e melhorar o desempenho de tarefas a partir das entradas que são dadas. [17]

Uma característica de arquiteturas de *Deep Learning* é que, quanto maior o tamanho dos dados de entrada, o modelo terá uma capacidade melhor e maior para aprender com as informações, identificando padrões e construindo abstrações para a execução das tarefas. No

âmbito de Redes Neurais Profundas, a presença de diversas camadas ocultas permite uma maior generalização de modelos. [16]

Para aplicações de Deep Learning, diferentemente de modelos tradicionais de Aprendizagem de Máquina, não é necessário se fazer um pré-processamento ou engenharia dos dados de entrada. Com isso, as informações geralmente são não estruturadas ou semi-estruturadas, como imagens, sons, textos, etc. A partir dessas estruturas, atividades como detecção de objetos em imagens, reconhecimento de voz, análise de texto, entre outras, são contempladas devido a estruturação e funcionamento do modelo.

De uma maneira geral, o funcionamento de modelos de Deep Learning funciona da maneira análoga à explicada na sessão de Redes Neurais, no entanto, possui a diferença de que se tem mais camadas e uma maior complexidade durante sua execução. Como ilustrado na Figura 4**,** temos o fluxo de funcionamento da resolução do problema.

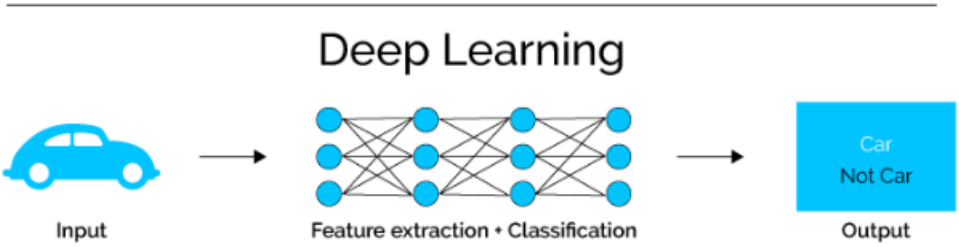


Figura 4. *Ilustração de uma aplicação de Deep Learning para resolução de um problema de Classificação* **[38]**

Desse modo, em muitas aplicações, modelos de Deep Learning são utilizados para realização de demandas e atividades. Por possuir uma complexidade maior, essa estrutura consegue resolver problemas que demandam uma maior riqueza de detalhes e análise abstrata das informações. No entanto, tem-se uma demanda computacional e de processamento maior, podendo não ser a melhor escolha em algumas situações.

2.4– Problemas de Classificação

Quando se trabalha com dados, uma grande variedade de tarefas pode ser demandada, a depender do contexto. Uma das atividades mais requeridas nesse cenário é a resolução de Problemas de Classificação. [7]

Formalmente, um modelo de classificação tem como principal objetivo definir uma função, representada por f, que, quando aplicada a um conjunto de atributos X, produza resultados precisos e significativos. Esse comportamento é dado pela equação:

$$Y = f(X)$$

De maneira sucinta, problemas de Classificação atribuem categorias aos elementos de um certo conjunto de dados, levando em consideração suas características e atributos. Dentro desse problema, existem diversos tipos de classificação. Alguns dos exemplos são:

- Classificação Binária: Saída é resumida em "positivo" ou "negativo".

Exemplo: Avaliação Física (Doente/Saudável)

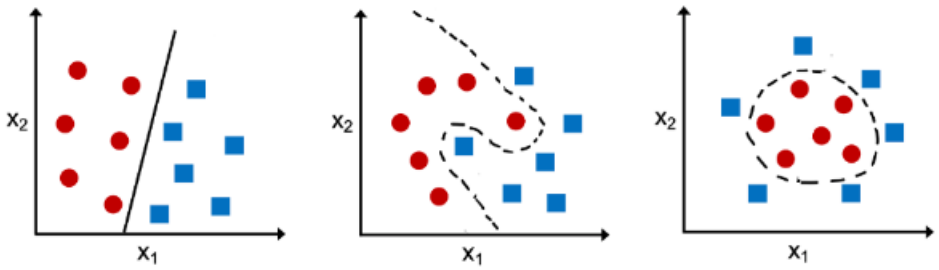


Figura 5. *Três diferentes exemplos de Classificação Binária* **[7]**

- Classificação Multiclasse: Dados são categorizados em três ou mais classes diferentes.

Exemplo: A partir de uma base de imagens com animais, classificar entre cachorro, gato e canário.

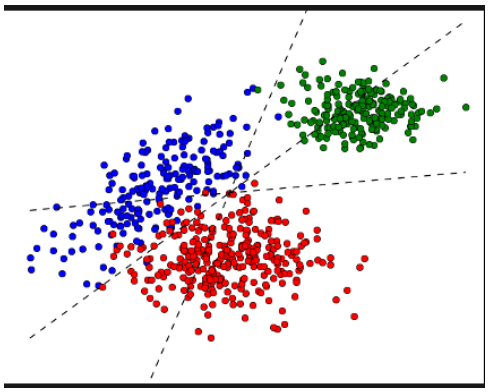


Figura 6. *Exemplo de um problema de Classificação Multiclasse, no qual tem-se 3 categorias diferentes* **[8]**

Em problemas de Machine Learning, diversos algoritmos são utilizados para a execução de tarefas de classificação ou até regressão, sendo os principais modelos Árvores de Decisão, Random Forest, Gradient Boosting, Redes Neurais, entre outros. A partir disso, o processo de escolha do melhor modelo a ser adotado a partir de um contexto deve ser analisado de uma maneira individual, considerando o objetivo final, estrutura das informações e até capacidade computacional disposta. [22] [9]

2.5– Funções de Transformação

Em geral, Funções de Transformação auxiliam o processo de adaptação das informações a uma determinada tarefa ou contexto. Funções como argmax e softmax são amplamente utilizadas em aplicações estatísticas, modelagens matemáticas e, também, em Aprendizado de Máquina. No entanto, nesse projeto iremos focar, inicialmente, na estrutura da função sofmax, a fim de se desdobrar para as suas variações.

A fórmula da função Softmax para um vetor de entrada z é dada por:

$$Softmax\ (z)_i = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \quad \textbf{(Eq 1.)}$$

Onde N é o número de classes, Zi é o valor do elemento i do vetor de entrada z, e $Softmax\ (z)_i$ é a probabilidade de pertencer à classe i.

A função Softmax tem o papel de transformar um conjunto de valores em um vetor de probabilidade que indica a propensão de cada elemento de pertencer a uma classe específica. Entretanto, pela natureza da fórmula, não existirá nenhuma classe que pode possuir o valor de probabilidade exatamente zero, dado que o denominador é uma função exponencial. Desse modo, alguns empecilhos em relação a problemas com uma grande quantidade de outputs começaram a surgir devido a não possibilidade de se zerar completamente a probabilidade de classes. [24]

A partir desse contexto, foi desenvolvida a Função de Transformação sparsemax [1] [3], com o principal objetivo de trazer uma alternativa de se reduzir a complexidade de modelos, descartando possibilidades remotas. A função tem a principal característica de retornar o mesmo vetor de probabilidades que é gerado pela função softmax, mas consegue atribuir o valor zero para classes que, antes, possuíam um valor muito próximo a zero. Esta função é dada pela fórmula:

$$sparsemax(z) := argmin_{p \in \Delta^{K-1}} \|p - z\|.^2 \quad \textbf{(Eq 2.)}$$

Como pode-se observar na Equação 2, a função sparsemax opera no vetor z, calculando a Projeção Euclidiana no conjunto de probabilidades. A partir desse resultado, tem-se que boa parte das projeções irão ao encontro de um limite do espaço, o que resulta em valores iguais a zero, nessa distribuição. Assim, tem-se uma quantidade expressiva de valores nulos (que podem ser descartados), juntamente com uma menor parcela de valores a serem considerados.

Desse modo, a função sparsemax trabalha como um filtro para situações em que o output possui muitas classes, ou até um componente para identificar qual grupo de variáveis são mais relevantes, potencialmente ajudando o modelo a interpretar de uma maneira mais efetiva o problema disposto.

Ao longo do tempo, uma terceira função, chamada Entmax, foi desenvolvida com o objetivo de interligar ainda mais os benefícios das aplicações anteriores. Essa função de

transformação combina características tanto do Softmax quanto Sparsemax, a fim de se ter um grau de adaptabilidade ainda maior. A função em si permite um maior controle em relação à taxa de esparsidade aplicada, a partir de um valor de α atribuído. [2] [1]

3 – REDES PROFUNDAS DANets COM DADOS TABULARES

Nesta etapa, teremos um maior detalhamento acerca da importância de alternativas para a solução de problemas que envolvem Dados Tabulares, além de mostrar o potencial do uso de *Deep Learning* nesse contexto. A partir disso, o modelo DANets será explicado de maneira mais aprofundada, a fim de se exemplificar, de maneira mais prática, quais principais ajustes podem ser realizados para que estruturas de Aprendizagem Profunda sejam capazes de atuar nessa área, que, atualmente, é maior dominada por modelos mais tradicionais de *Machine Learning*.

Dessa forma, tem-se a motivação de instigar a curiosidade do leitor acerca dos maiores detalhes das implementações, e sanar possíveis dúvidas de aplicações.

3.1– Objetivos do Modelo

O DANets (Chen et al. 2022) [1] é um modelo que visa resolver problemas de Classificação e Regressão com uma base de Dados Tabulares como entrada. Assim, tem-se o aproveitamento da forte natureza flexível que uma estrutura de *Deep Learning* possui, com o objetivo de se gerar um bom resultado e uma boa performance no processo de execução de tarefas.

Devida à natureza heterogênea dos Dados Tabulares, o modelo DANets, através de uma Rede Neural Profunda composta por Redes Abstratas, realiza uma abstração dos atributos, juntamente com um agrupamento dessas informações, por um método de análise de correlação das informações usando uma Máscara Aprendível. Além disso, são incorporados atalhos na estrutura, que fazem uma combinação entre alguns dados já abstraídos e dados em estrutura original. Desse modo, é proporcionada uma maior variedade entre as características dos grupos, e, consequentemente, tem-se uma busca para manter apenas grupos com correlações consistentes e de maior relevância. [1]

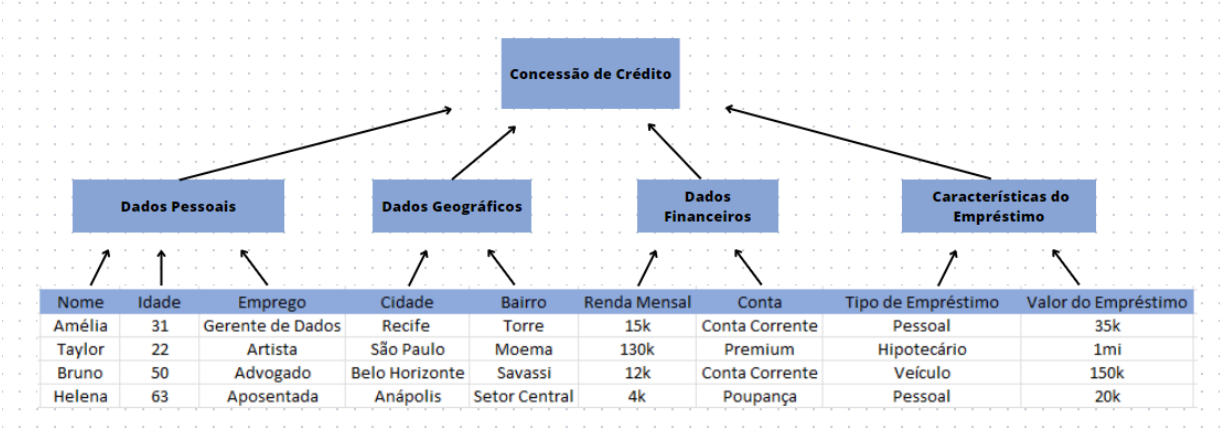
Para uma melhor ilustração da proposta do modelo, vamos considerar um problema no contexto de Análise de Concessão Crédito.

Nome	Idade	Emprego	Cidade	Bairro	Renda Mensal	Conta	Tipo de Empréstimo	Valor do Empréstimo
Amélia	31	Gerente de Dados	Recife	Torre	15k	Conta Corrente	Pessoal	35k
Taylor	22	Artista	São Paulo	Moema	130k	Premium	Hipotecário	1mi
Bruno	50	Advogado	Belo Horizonte	Savassi	12k	Conta Corrente	Veículo	150k
Helena	63	Aposentada	Anápolis	Setor Central	4k	Poupança	Pessoal	20k

Figura 7. *Exemplo de base com informações de clientes para Análise de Concessão de Crédito.*

Antes de qualquer tarefa de Classificação, é interessante realizar um processo de maior entendimento e estruturação das informações dispostas. Assim, um caminho possível para análise dos dados é de a realização de agrupamentos de características que possuem

informações semelhantes. Na tarefa de Análise de Concessão de Crédito, por exemplo, é possível realizar essa divisão em 4 grupos: Dados Pessoais, Dados Geográficos, Dados Financeiros e Características do Empréstimo, como indicado na Figura 8.

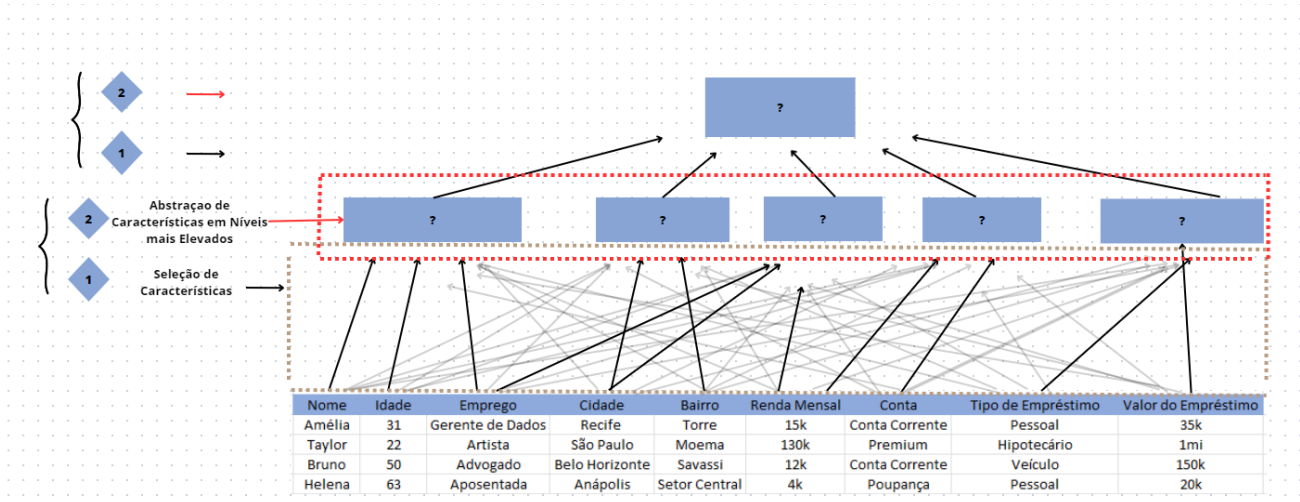


Nome	Idade	Emprego	Cidade	Bairro	Renda Mensal	Conta	Tipo de Empréstimo	Valor do Empréstimo
Amélia	31	Gerente de Dados	Recife	Torre	15k	Conta Corrente	Pessoal	35k
Taylor	22	Artista	São Paulo	Moema	130k	Premium	Hipotecário	1mi
Bruno	50	Advogado	Belo Horizonte	Savassi	12k	Conta Corrente	Veículo	150k
Helena	63	Aposentada	Anápolis	Setor Central	4k	Poupança	Pessoal	20k

Figura 8. *Estruturação de Grupos de Correlação a partir das características dispostas.*

Com essa estrutura, é possível atribuir diferentes pesos e níveis de relevância de cada informação com maior facilidade, a fim de que se obtenha um bom resultado na resolução da tarefa demandada. No entanto, é comum nos depararmos com bases de dados com uma grande variedade de informações, o que tornaria esta tarefa manual ainda mais difícil.

Assim, o modelo DANets tem como objetivo realizar todo esse processo de análise de correlações e atribuição de relevância dos dados, como pode ser observado na Figura 9.



Nome	Idade	Emprego	Cidade	Bairro	Renda Mensal	Conta	Tipo de Empréstimo	Valor do Empréstimo
Amélia	31	Gerente de Dados	Recife	Torre	15k	Conta Corrente	Pessoal	35k
Taylor	22	Artista	São Paulo	Moema	130k	Premium	Hipotecário	1mi
Bruno	50	Advogado	Belo Horizonte	Savassi	12k	Conta Corrente	Veículo	150k
Helena	63	Aposentada	Anápolis	Setor Central	4k	Poupança	Pessoal	20k

Figura 9. *Lógica de funcionamento do modelo DANets, com as etapas de Seleção de Características e Abstração das Características em Níveis mais elevados.*

Dando seguimento na análise da imagem acima, é possível perceber que é realizado, de maneira sequencial, um processo de seleção de Características para formar grupos de maiores

correlação, seguido por uma abstração dos atributos. Um ponto interessante do modelo DANets é que ele permite que um atributo faça parte de um ou mais grupos de correlação, possibilitando uma maior diversidade das informações. Essas etapas ocorrem até chegar a uma estrutura final para a tarefa de Concessão de Crédito.

Com isso, o objetivo principal dessa modelagem é trazer um bom caminho alternativo em relação ao contexto de Dados Tabulares, podendo ser utilizado para diversas aplicações com esse tipo de informações.

3.2– Metodologia

Nesse projeto, foi desenvolvido um estudo mais aprofundado para o entendimento e validação do funcionamento do modelo DANets para problemas de Classificação. Ao longo dos testes com uma base de dados extensa, variada e devidamente balanceada, tornou-se necessário realizar alguns ajustes adicionais em relação à estrutura original, a fim de se obter um maior acompanhamento da performance dos resultados nos conjuntos de Treinamento e Validação. O acompanhamento do MSE (Erro Quadrático Médio) no Treino e Validação, juntamente com o teste estatístico Kolmogorov-Smirnov ao final do processo tornaram a análise mais assertiva e com uma visão mais crítica acerca de falsos resultados positivos consequentes de um possível overfitting, além de uma avaliação da consistência das distribuições. [20] [25]

Em adição, ajustes estruturais de passagens de parâmetros em certas funções foram executados, apenas para facilitar o fluxo das informações e se ter um mapeamento do que estava acontecendo mais detalhadamente. Esse processo teve como principal objetivo analisar possíveis melhorias para o modelo.

3.2.1– Estruturação do Modelo

Como mencionado na Sessão 3.1, a aplicação do DANets [1] é uma proposta de incorporar o uso de Redes Neurais e Aprendizado Profundo em problemas de Regressão e Classificação com Dados Tabulares. Em uma visão geral, o DANets é uma Rede Abstrata Profunda, composta por diversas Camadas Abstratas (chamadas de AbstLay).

Desse modo, temos dois principais termos que serão utilizados para o detalhamento do modelo:

- Abstract Layer (AbstLay): Componente Neural do modelo de Aprendizado Profundo, que opera em um nível superior de abstração.

- Deep Abstract Networks (DANets): Na tradução, Redes Abstratas Profundas, são estruturadas a partir de conjuntos de Camadas de Abstração (AbstLays). Assim, tem-se a construção de uma Rede Neural Profunda, capaz de solucionar problemas.

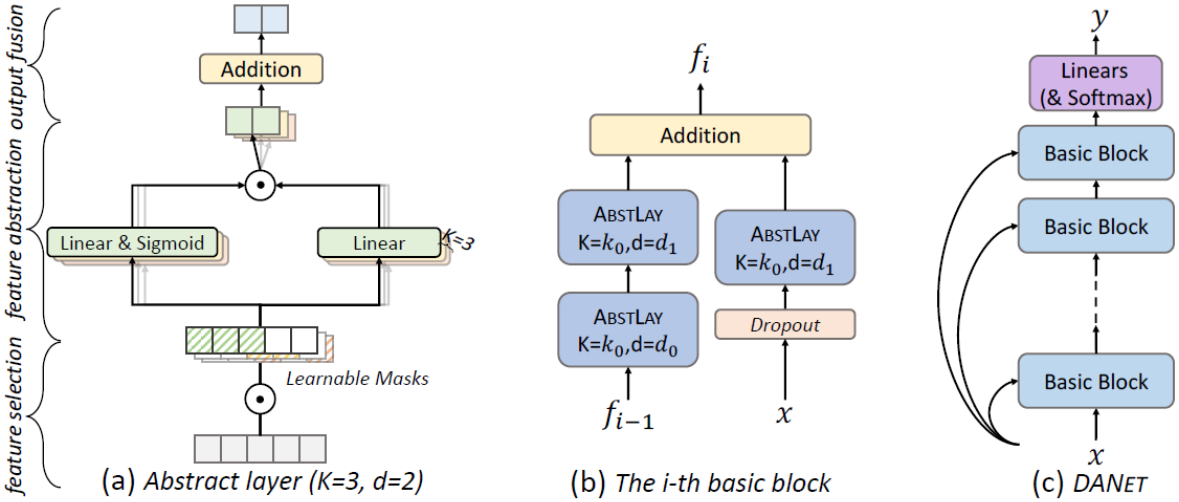


Figura 10. *Fluxo de Estruturação do modelo DANets. (a) Composição da Camada Abstrata (AbstLay), com a presença da Máscara Aprendíveis e Funções de Transformação, sendo K o número de Máscaras e d é a dimensão de saída do Output. Além disso, fica claro o fluxo das etapas de Seleção e Abstração de Características, seguido da Fusão de Saída. (b) Composição do Bloco Básico, que possui uma combinação de Camadas Abstratas (AbstLays), juntamente com um atalho de entrada que possui uma taxa de Dropout atribuída.*

(c) Estrutura geral do DANets, composta por diversos Blocos Básicos em sequência, cada um com atalhos e, ao final, uma Função de Transformação **[1]**

Com o objetivo de facilitar a compreensão do modelo, vamos dividir a explicação a partir das estruturas da Camada Abstrata e a composição de um Bloco Básico do modelo do DANEts. Depois, um maior detalhamento acerca do comportamento de cada processo será realizado para que, ao final, tenhamos um mapeamento completo de todos os principais pontos.

Como podemos ver na Figura 10 (a), a Camada Abstrata (AbstLay) é formada por Máscaras Aprendidas, funções de Transformação e uma junção das abstrações e seleções realizadas. É nessa estrutura que acontecem algumas etapas de processamento mais complexas, a partir dos dados de entrada, que serão mais detalhadas em breve.

Na Figura 10 (b), tem-se a estruturação do Bloco Básico, que é uma unidade básica de construção em uma arquitetura de Rede Neural. Assim, esta unidade básica é composta por duas Camadas de Abstração (AbstLay). A primeira camada é a receptora dos dados de entrada originais, e, também, responsável pela abstração das informações, sendo seguida pela segunda AbstLay, que tem como entrada a saída intermediária da primeira camada. Assim, é gerada uma combinação com um nível maior de abstração para a saída.

Por fim, a Figura 10 (c) ilustra a composição final do modelo DANets, que possui uma grande sequência de Blocos Básicos, todos eles possuindo atalhos para que se tenha uma maior variedade e nível de aleatoriedade dos dados para processamento e treinamento.

Agora, visualizando com um nível maior de detalhes, na primeira etapa para a criação da estrutura do DANets, é realizada a inicialização da Camada de Rede, somada a uma lista que contém as camadas adicionais com os parâmetros já pré-definidos (número de camadas, dimensão da saída e tamanho virtual do lote). Cada camada é composta por um Bloco Básico, que já foi detalhado previamente.

A partir disso, é interessante pontuar que, à medida que os dados são processados no modelo, a camada AbstLay é responsável pela realização de 3 principais processamentos, sendo essas etapas:

Seleção de Características: Essa fase consiste na construção de grupos de características, sendo elas as mais relevantes, ou até que possuam uma maior relação entre si. O processo é iniciado com o cálculo de Máscaras Aprendíveis com base nos dados de entrada. A Máscara Esparsa Aprendida é representada pela letra M, W é o Vetor de Parâmetros Aprendíveis e f um vetor de características. Com isso, temos:

$$M = entmax_{\alpha^*}(W_{mask}) \quad \textbf{(Eq 3.)}$$

$$f' = M \odot f \quad \textbf{(Eq 4.)}$$

Normalmente, para problemas de Classificação Multiclasse em modelos de Redes Neurais é utilizada a função softmax na camada de saída, que é uma função que realiza a transformação de um vetor de valores em um vetor de probabilidades. No modelo DANets, como pode ser observada na Eq 3, é usada a função Entmax [2], que combina características tanto do Softmax quanto Sparsemax [3] (esse tópico foi mais detalhado no Capítulo 2.5). O valor default para essa função é de α=1.5, assim, tem-se a possibilidade de que alguns outputs tenham valor nulo. Dessa maneira, os atributos de f′ equivalentes a zero já são descartados, resultando, assim, em uma formação inicial de subgrupos. [24]

Abstração de Características: Na fase de Abstração, como o nome já demonstra, é realizado o processo de extrair as representações mais importantes e aplicáveis para uma determinada tarefa. Com isso, tem-se um vetor de atenção q, que serve para ponderar a importância das características de f ′ (Eq 4.).

$$q = \text{sigmoid}(BN(W_1 * f')) \quad \textbf{(Eq 5.)}$$

Em seguida, usando o vetor de atenção q, tem-se o resultado da abstração, que é dada pela função:

$$f^* = \text{ReLU}(q \odot BN(W_2\, f')), \quad \textbf{(Eq 6)}$$

Assim, com a realização do cálculo com diferentes Vetores de Parâmetros Aprendíveis (W1 e W2), é possível se ter uma maior variabilidade de contexto para se encontrar os parâmetros, de fato, mais relevantes [1]. É importante pontuar também que, no modelo, é usado o 'Ghost Batch Normalization' [26] para a Normalização em Lote (BN), dado que essa técnica performa melhor para problemas em que se possui lotes com tamanhos expressivos.

Com isso, temos o output f*, que é o resultado da abstração realizada.

Operação de Fusão de Saída e Processamento em Paralelo: Depois da Seleção e Abstração de Características, tem-se o processo de fusão de características que foram abstraídas de diferentes grupos. Como as AbstLays estão em sequência, é possível utilizar outputs de diferentes níveis de abstração para serem processadas e mapear os grupos e pesos dos atributos de uma maneira mais efetiva (possibilitando, também, o descarte de características pouco relevantes).

Desse modo, o AbstLay garante que o modelo possa atribuir diferentes graus de importância (pesos) para diferentes atributos dos dados de entrada. Além disso, tem-se uma redução da complexidade da Camada de Abstração, a fim de que tenha uma menor demanda computacional durante o processo.

Voltando à estrutura geral do modelo, a sequência de vários AbstLays tem como a principal função de, ao longo das camadas, fazer uma abstração dos atributos e ir desenvolvendo uma lógica de correlação entre os atributos mais importantes e uma semântica global. Além disso, atalhos de entrada são introduzidos no Bloco Básico para aumentar o nível de aleatoriedade e diversidade nos dados, fazendo uma busca ainda mais acurada para encontrar grupos com correlações coerentes.

3.3– Implementação

3.3.1– DANets

A partir de uma base de dados tabular como entrada, inicialmente deve ser feito um tratamento dos dados. Análises de possíveis dados duplicados, ausentes, ou até mesmo inconsistentes devem ser realizadas para garantir uma melhor performance e aprendizado do modelo. A partir disso, é realizada a divisão dos dados em 3 partes: dataset de Treinamento, Validação e Teste.

Para os primeiros passos com o modelo do DANets, é necessário fazer algumas atribuições iniciais de parâmetros para a sequência de todo o processo. Informações como classificação do problema (sinalização se o problema é de Regressão ou Classificação), número

de épocas, patience, taxa de aprendizado, número de camadas, drop_rate são pedidas como entrada. [1]

Para trazer uma maior facilidade na execução e performance do modelo durante a pesquisa, foi incorporado o uso da biblioteca Optuna [22], que tem como a principal funcionalidade a otimização de hiperparâmetros e tuning de modelos. Com o uso dessa ferramenta, tem-se a busca de quais serão os melhores conjuntos hiperparâmetros, a partir de um contexto dado, realizando-se testes com diversas combinações, gerando, assim, um conjunto ótimo.

Após a estruturação dos parâmetros do modelo, temos a passagem das informações e ajustes de modos de execução. Caso a atividade seja de Classificação, terá a criação do modelo DANetClassifier, e caso a atividade seja de Regressão, o modelo DANetRegressor é utilizado. Ambos os modelos possuem características semelhantes, mudando apenas alguns ajustes no fluxo dos dados e métodos de avaliação de performance.

Assim, antes do início do processo de treinamento propriamente dito, existe uma importante etapa que consiste no processo de atualizar os parâmetros de ajuste do modelo, que é dada pela função *update_fit_params*. Na etapa de treinamento, tem-se vários ajustes nas informações, a fim de que o modelo seja treinado de uma maneira não enviesada. Elencando de maneira ordenada, temos as seguintes tarefas:

- Verificação da dimensão de saída
- Verificação dos rótulos do conjunto de treinamento e validação
- Atualização dos parâmetros e mapeamento dos rótulos de classe para índices numéricos.

A criação e construção de DataLoaders também é realizada. Os DataLoaders são objetos do Pytorch, e, com eles, é possível realizar o carregamento dos dados em lotes durante o processo de treinamento e validação. [27] Depois dessa etapa, tem-se a estruturação do modelo DANets, que foi descrita na Sessão 3.2.1 do relatório. Com o modelo já estruturado e com os parâmetros já incrementados, inicializa-se o processo de Treinamento do modelo.

Em relação ao processo de Treinamento, é de conhecimento geral que em muitas situações, ao se criar uma rede neural, dependendo da base de dados que será trabalhada e dependendo da estrutura do modelo, tem-se uma tendência ao aprendizado da máquina ser realizado de uma maneira enviesada. Com isso, o modelo tende a ‘decorar’ a base de Treinamento, performando muito bem nessa etapa, e, ao receber novas informações durante as etapas de Validação e de Teste, não manter a taxa de acertos elevada. Esse comportamento pode se dar quando o modelo se prende em informações muito detalhadas que não possuem tanta importância, ou até, também, atribui uma taxa de importância muito alta a certos ruídos que

estão na base de dados. Esse comportamento é comumente chamado de ‘Overfitting’, e pode ser perigoso, pois cria a falsa impressão de que o modelo está, de fato, aprendendo corretamente, e com uma boa taxa de acurácia, quando, na verdade, isso não é o que acontece. [25]

Desse modo, para evitar o overfitting no modelo DANets tem-se a definição de uma taxa de Dropout. A técnica de Dropout é o processo de desativar, de maneira aleatória, uma determinada porcentagem dos neurônios durante a etapa de Treinamento. Com isso o modelo é forçado a diversificar seu modo de aprendizado, e, possivelmente, desprende-se de alguns neurônios que possam estar enviesados. [10]

Também, durante o processo de Treinamento e Validação do modelo, a função de Callback é constantemente chamada, com o principal objetivo de se ter um maior monitoramento dos parâmetros e desempenho do modelo. Informações como acurácia, MSE (Erro Quadrático Médio) e parâmetro de perda (calculado pela função de perda *crossentropy*) são mensurados e expostos em cada época. Com isso, analisando o comportamento e evolução dessas métricas ao longo do tempo, é possível, também, ter um maior controle em relação a situações de Overfitting, dado a alguns comportamentos típicos diante desse contexto. Um exemplo claro de possível Overfitting é quando a métrica MSE do conjunto de Treinamento está decaindo ao passar do tempo, enquanto o MSE do conjunto de Validação está estável ou com valores crescentes. [25]

Por fim, a etapa de Teste é implementada a fim de se avaliar a performance do modelo em um conjunto de dados diferente. Nesse processo, métricas como KS, acurácia e AUROC são avaliadas a fim de se observar a performance do modelo. A partir disso, tem-se a geração de gráficos e matrizes de confusões para um maior entendimento e visualização das informações. [20] [8]

3.3.2– Modelos Convencionais de Machine Learning

Para a estruturação dos Modelos Convencionais de Machine Learning para problemas com Dados Tabulares, foi necessário um entendimento das principais estruturas e aplicações a partir do contexto fornecido. [23] [5] Durante esse processo, diversos modelos foram selecionados para realizar a atividade. A etapa seguinte envolveu um conjunto de testes de combinações de hiperparâmetros para cada um desses modelos, utilizando a biblioteca Optuna [22]. Com essa abordagem, tem-se a busca de melhores arquiteturas dos modelos para o contexto trabalhado, almejando uma performance próxima ao Estado da Arte.

Em resumo, os modelos de MLP, Random Forest e Gradient Boosting foram selecionados para o desenvolvimento do projeto, sendo feita uma análise inicial (com uma estrutura simplificada de cada modelo) para validação de um mínimo funcionamento de cada

um, a partir da base de dados trabalhada. Em sequência, com o funcionamento do modelo já garantido, a complexidade das estruturas foi aumentada para a obtenção dos melhores resultados.

3.4– Estruturação dos Testes

A fim de se ter uma análise crítica e comparativa entre o modelo estudado (DANets), foi necessário realizar uma pesquisa de modelos convencionais que são utilizados para problemas de Classificação com Dados Tabulares. Assim, a partir dessa busca em pesquisas e aplicações reais, alguns modelos tradicionais de *Machine Learning* e de MLPs foram encontrados com frequência. Desse modo, foram escolhidos os seguintes modelos a serem comparados:

- MLP
- *Random Forest*
- *Gradient Boosting*
- Ensemble Misto

A escolha dos modelos foi fundamentada não apenas devido a verificação de bons desempenhos e resultados em diversas aplicações, mas também levando em consideração características teóricas positivas que fazem parte de cada estrutura. [23] [5]

Além disso, para se realizar testes funcionais é imprescindível fazer a seleção de uma base completa, extensa e bem diversificada (com valores numéricos e categóricos). Essa etapa é de grande importância, dado que, caso se tenha uma escolha de uma base pouco variada, ou até enviesada, todo o processo de treinamento, teste e validação dos modelos podem ser pouco confiáveis. Com isso, por possuir dados reais (ainda que anonimizados), uma excelente variação de atributos e um tamanho expressivo (390 mil registros para treinamento e 190 mil registros para teste, além de 137 variáveis), uma base de Análise de Crédito foi selecionada para ser testada nos diversos modelos (incluindo o DANets).

Após a escolha da base de dados, tem-se a etapa de análise e tratamento das informações. Para evitar interpretações erradas, faz-se a verificação e remoção de duplicatas dos dados, além de uma normalização dos dados e balanceamento das classes. A etapa do balanceamento das classes é importante, visto que alguns casos podem ser menos frequentes, podendo dificultar o aprendizado das características de uma classe minoritária. Em seguida, foi realizada a separação dos conjuntos de Treino, Validação e Teste. [13]

Com essas informações bem estruturadas, inicia-se, de fato, a realização dos testes dos modelos a partir desse contexto escolhido. Em suma, o problema se resume em, com base no perfil do cliente (a partir dos atributos disponíveis na base de dados), decidir a quem conceder crédito (risco de inadimplência).

Inicialmente, foram realizados diversos testes com o modelo DANets [1], a fim de se observar o desempenho do modelo por si só. Ou seja, a partir de métricas já conhecidas, ter uma visão se o modelo estaria performando de uma maneira satisfatória ou não. Métricas como KS (Kolmogorov-Smirnov), AUROC e acurácia foram levadas em conta para essa primeira avaliação. [20] [8]

Como já abordado no Capítulo ***3.3.1***, devido ao contexto e à base de dados utilizada, foi necessário realizar ajustes na implementação do modelo para a obtenção de uma maior usabilidade, acompanhamento de resultados e melhor performance. Além disso, foram implementados cálculos de Erro Quadrático Médio (MSE) durante o treinamento e validação. [25]

Como o problema, no contexto da base de Análise de Crédito, é de Classificação, o modelo *DANetClassifier* [1] foi selecionado para a realização da atividade. Além disso, com o objetivo de se encontrar os melhores parâmetros para a execução do modelo, a ferramenta Optuna [22] foi adotada. Em uma visão geral, a utilização do Optuna simplifica e facilita o processo de busca do conjunto de melhores hiperparâmetros de um modelo dado um determinado problema, fazendo a variação dessas combinações juntamente com a análise dos resultados gerados a partir da execução do mesmo. Desse modo, o mapeamento de melhores estruturas e um potencial de melhor comportamento dado algum determinado parâmetro fica mais fácil de ser analisado.

Dessa forma, os seguintes parâmetros foram configurados para a variação no Optuna:

- Learning Rate = [0.0005, 0.00006, 0.0008, 0.008, 0.0009, 0.0012]
- Drop Rate = 0.0 até 0.5, utilizando um step de 0.1
- Tamanho do Lote (Batch Size) = [64, 512, 2048, 8192]

Sendo *Learning Rate* a Taxa de Aprendizado do modelo, que tem como principal função definir o ritmo de ajuste dos pesos em relação à taxa de erro [28][29], *Drop Rate* a taxa de abandono de unidades da Rede [29] e o Tamanho do Lote o número de exemplos usados em uma iteração. [30]

Além disso, foi inserida uma taxa de Patience igual a 23, com o objetivo de reduzir o tempo de execução dos testes, caso se tenha uma percepção de que não está havendo uma melhoria na performance do modelo ao longo das épocas. Desse modo, após a obtenção de conjuntos de hiperparâmetros que possuíram uma melhor performance em relação ao resultado, foi iniciado o processo de buscar melhorias e ajustes na estrutura do modelo, a fim de superar o desempenho inicial.

Depois de uma análise mais aprofundada na estrutura do modelo DANets [1], uma das etapas importantes no processo é a abstração das informações, por meio da camada abstrata AbstLay. Durante essa parte, que é dada pela presença da Função de Transformação Entmax

[2], explicada no *Capítulo 2.5,* o valor default do nível de esparsidade é de $\alpha = 1.5$. A fim de analisar o comportamento em diferentes situações, além do modelo ser testado com o valor default, posteriormente, foram aplicados valores abaixo e acima de 1.5. Desse modo, pode-se ter uma avaliação se há uma melhora ou não na performance do modelo diante da situação dada.

Após um trabalho de testes variados com o modelo DANets, tem-se a etapa de comparação com Métodos Convencionais. Como citado anteriormente, modelos de MLP, Random Forest e Gradient Boosting foram executados para se ter uma análise comparativa mais detalhada e completa. É importante ressaltar que foram utilizados os mesmos conjuntos de Treino, Validação e Teste, a fim de se ter as mesmas premissas utilizadas no modelo DANets.

Todos os modelos foram testados com parâmetros simplificados inicialmente. Em seguida, o uso do Optuna foi necessário para encontrar os melhores conjuntos de hiperparâmetros, a fim de se buscar um valor muito próximo à melhor performance de cada modelo. A seguir, vamos descrever rapidamente quais variações foram realizadas em cada modelo:

- MLP (Formada por 3 camadas densas)
    - Quantidades de neurônios nas Camadas Ocultas: [128, 256, 512,1024]
    - Taxa de Dropout: 0.0 até 0.5, com o step de 0.1
    - Otimizador: [‘Adam’, ‘sgd’, ‘rmsprop’]
    - Tamanho do Lote: [32, 64, 128]
    - Função de Ativação: [‘relu’, ‘sigmoid’, ‘tanh’]

O hiperparâmetro da Taxa de Dropout, como mencionado anteriormente, indica a taxa de abandono de unidades da Rede [29]. Em seguida, tem-se o otimizador, que são algoritmos que, como o nome já expressa, focam em otimizar a performance do modelo, minimizando a função de perda, ajustando os pesos da Rede e, consequentemente, aumentando a acurácia do modelo.[31] O Tamanho do Lote [30] e a função de ativação também foram testados com diferentes valores, sendo a função de ativação uma transformação não linear, que é realizada a partir do sinal de entrada.[32]

- Random Forest
    - Número de Árvores: 100 a 500, com step de 100
    - Criterion (função para mensurar a qualidade de uma divisão): [‘gini’, ‘entropy’]
    - Profundidade máxima das Árvores: 2 a 10, com step de 2

- min_samples_split (número mínimo de amostras necessárias para dividir um nó interno): 4 a 10, com step de 2
- min_samples_leaf (Número mínimo de amostras necessárias para ser uma folha): 4 a 10, com step de 2
- max_features (número máximo de recursos a serem considerados para dividir um nó): [‘sqrt’, ‘log2’]

Em adição às informações, é interessante pontuar que, na aplicação do Random Forest, tem-se o teste do Número de Árvores do modelo, que define a quantidade de árvores de decisão a serem construídas no conjunto e a sua profundidade máxima (a depender do valor atribuído, isso pode afetar a complexidade e capacidade de generalização do modelo). [33]

- Gradient Boosting
  - Função de Perda: [Logarítmica, exponencial]
  - Taxa de Aprendizado: [0.001, 0.1]
  - Número de estimadores: 100 a 500, com step de 100

Em relação aos hiperparâmetros acima, a Função de Perda faz a quantificação do erro entre as previsões do modelo e valores reais, atribuindo diferentes penalizações para essas discrepâncias. Com isso, a depender de qual função de perda é utilizada, tem-se uma mudança no comportamento de avaliação e correção de erros no modelo.[34] Além disso, a Taxa de Aprendizado define o ritmo de ajuste dos pesos em relação à taxa de erro, e o Número de Estimadores indica o volume de combinações sequenciais durante o Treinamento do modelo. [28][29]

- Ensemble Misto
  - MLP (melhor combinação dos hiperparâmetros)
  - Random Forest (melhor combinação dos hiperparâmetros)
  - Gradient Boosting (melhor combinação dos hiperparâmetros)

Assim, a partir desses parâmetros, os modelos foram executados, juntamente com uma sequência de métricas dispostas ao longo do treinamento, a fim de possibilitar uma avaliação de desempenho e entendimento caso aconteçam comportamentos que tendam para um Overfitting.

4 – COMPARAÇÃO DO MODELO DANets E MÉTODOS CONVENCIONAIS

Para uma Análise Comparativa mais assertiva, após a estruturação dos Testes com o modelo DANets, além dos modelos de MLP, e de Modelos Tradicionais de Machine Learning como Random Forest, Gradient Boosting, Ensemble Misto, é necessário se ter um maior entendimento do contexto prático e teórico.

4.1.1- Comparação Teórica entre os Modelos

Antes de partirmos para uma análise dos resultados práticos, é interessante se ter uma análise mais detalhada em características dos modelos utilizados durante os testes, trazendo uma visão de possíveis pontos positivos e negativos que podem ser fatores determinantes para o desempenho final da execução de uma tarefa. Como pontuado anteriormente, por possuir características intrínsecas favoráveis, certos Modelos Tradicionais de *Machine Learning* possuem um bom desempenho quando se é falado em atividades envolvendo Dados Tabulares. [3] [4] [5] [23]

Em uma visão inicial, por se tratar de informações com um viés heterogêneo, tem-se uma demanda clara de um modelo que consiga se ajustar de maneira mais robusta. Em suma, devido a um grande conjunto de informações diferentes e de importância variada para uma determinada atividade, modelos que possuem uma melhor resposta a dados ruidosos podem ser bons candidatos para uma boa performance. [1] [4] [5]

Métodos para otimização de performance são amplamente utilizados em modelos de Machine Learning, o que pode levantar alguns pontos de reflexão acerca das características positivas em modelos mais tradicionais, quando se trata de Dados Tabulares. Inicialmente, tem-se dois principais métodos que, quando utilizados de maneira correta, podem ser decisivos para bons resultados: Bagging e Boosting.

De maneira mais direta, técnicas de Bagging envolvem um processo de treinamento de diferentes amostras de um mesmo modelo, de maneira independente, com o objetivo final de se alcançar um comportamento médio e mais robusto em termos de desempenho. [18] Já abordando a técnica de Boosting, a principal ideia é a combinação sequencial dos modelos fracos a serem treinados, a fim de ter um maior fator de correção em relação a falhas, sendo capaz de desenvolver uma heurística mais bem estruturada para executar a atividade de uma maneira mais efetiva. [19]

Assim, esses dois métodos visam uma maior evolução na performance do modelo, sendo o processo de treinamento o maior ponto de diferença entre os dois, sendo o Boosting com um processo de treinamento sequencial e o Bagging com um processo de treinamento dos modelos de maneira paralelizada (simultaneamente). [18] [19]

Além das técnicas descritas acima, é relevante destacar que, em estruturas como Árvores de Decisão e *Random Forest*, ruídos são suavizados devido a maior distribuição das informações em árvores, juntamente com uma maior agregação de previsões. Assim, tem-se mais um indicativo de que, utilizando Dados Tabulares como entrada, essa propriedade pode ser benéfica para que se tenha uma maior segurança acerca do resultado. [23] [5] [4]

Em uma visão geral e comparativa, modelos de *Deep Learning*, por possuírem características muito centradas em aprendizado por localidade, ou até por padrões, muitas vezes não são bem-vistos para problemas com Dados Tabulares, já que a estrutura geralmente é heterogênea e sem correlações bem estruturadas entre os atributos.

No entanto, ao se observar as principais atuações e comportamentos desses modelos, torna-se evidente uma série de qualidades que essas estruturas mais robustas proporcionam. A característica de maior valor é, sem dúvidas, a alta capacidade de adaptação e flexibilidade dos modelos a depender de cada contexto que é dado, sendo possível, assim, realizar tarefas mais complexas de uma maneira efetiva. Desse modo, ainda que modelos tradicionais de *Machine Learning* são soluções mais frequentes, é extremamente válido desenvolver modelos e realizar testes e para avaliar a o potencial de performance de estruturas de *Deep Learning* nesta área de atuação. [1] [4]

4.1.2- Análise Comparativa dos Resultados

Com os testes para diversas combinações estruturais do modelo DANets e os Modelos Tradicionais de Machine Learning, é possível realizar uma análise mais detalhada acerca dos comportamentos e desempenhos de cada modelo. Com isso, é importante se ter métricas e informações estruturadas em todos os modelos a fim de se ter uma comparação justa e igualitária diante das diversas performances.

É importante reforçar que, no modelo DANets, para se ter um acompanhamento mais detalhado do treinamento do modelo, o cálculo do MSE, acurácia e parâmetro de perda (pela função de perda crossentropy) foram calculados durante as épocas. Ao fim do processo de treinamento e validação, as métricas da etapa do Teste também são geradas para incorporar na análise do desempenho. As principais métricas observadas são o KS e acurácia.

Durante os testes, valores de α nas camadas abstratas (AbstLay) foram variados a fim de se analisar o desempenho do modelo com diversos níveis de esparsidade no processo de abstração dos atributos. No entanto, os valores α testados (entre 1.2 e 1.8) resultaram em desempenhos semelhantes, sendo apenas a combinação dos hiperparâmetros de melhor performance a principal diferença. Com isso, manteremos o valor default de $\alpha = 1.5$ ao mostrar os melhores resultados aferidos.

Para avaliação do desempenho do modelo dado o problema de Análise de Crédito, explicado no *Capítulo 3.4*, tem-se 3 melhores resultados, avaliando o KS como parâmetro prioritário de desempenho, sendo seguido pela Acurácia:

Modelo DANets:

Modelo	Taxa de Aprendizado	Taxa de Queda (drop rate)	Tamanho do Batch
A	0.008	0.3	8192
B	0.0005	0.2	64
C	0.0008	0.0	64

Tabela 1. *Hiperparâmetros das 3 melhores estruturas do modelo DANets para a aplicação de Análise de Crédito.*

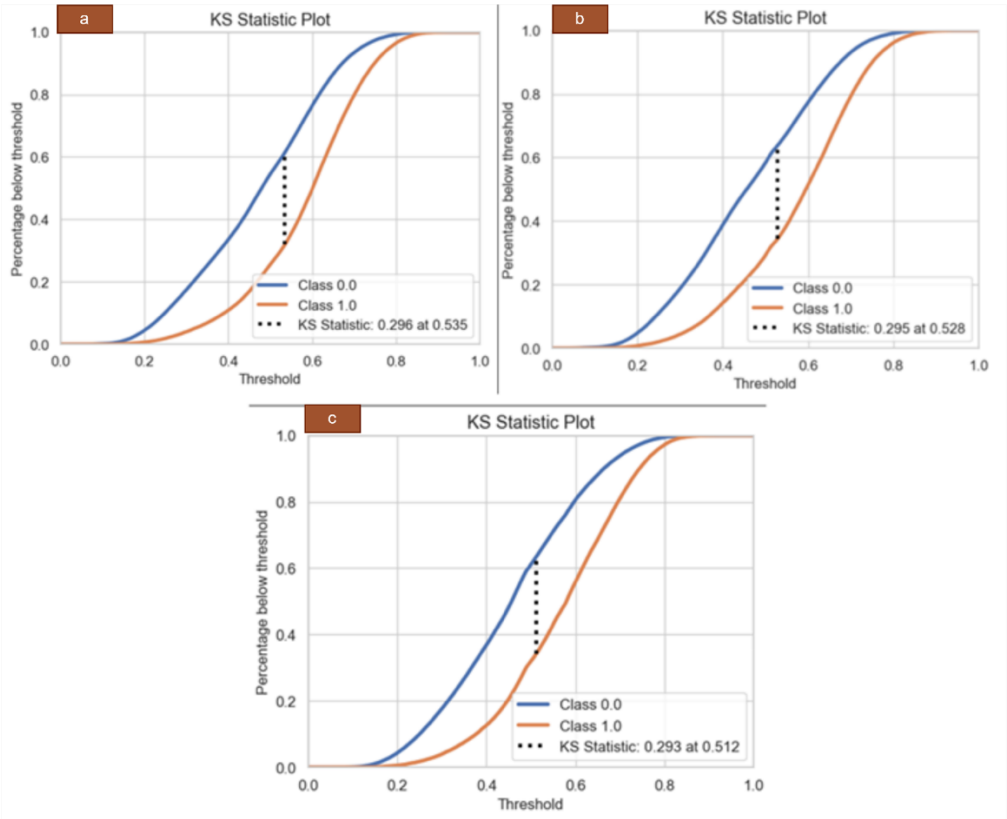


Figura 11. *Curva do KS dos 3 modelos resultantes, sendo cada modelo indicado por A, B e C.*

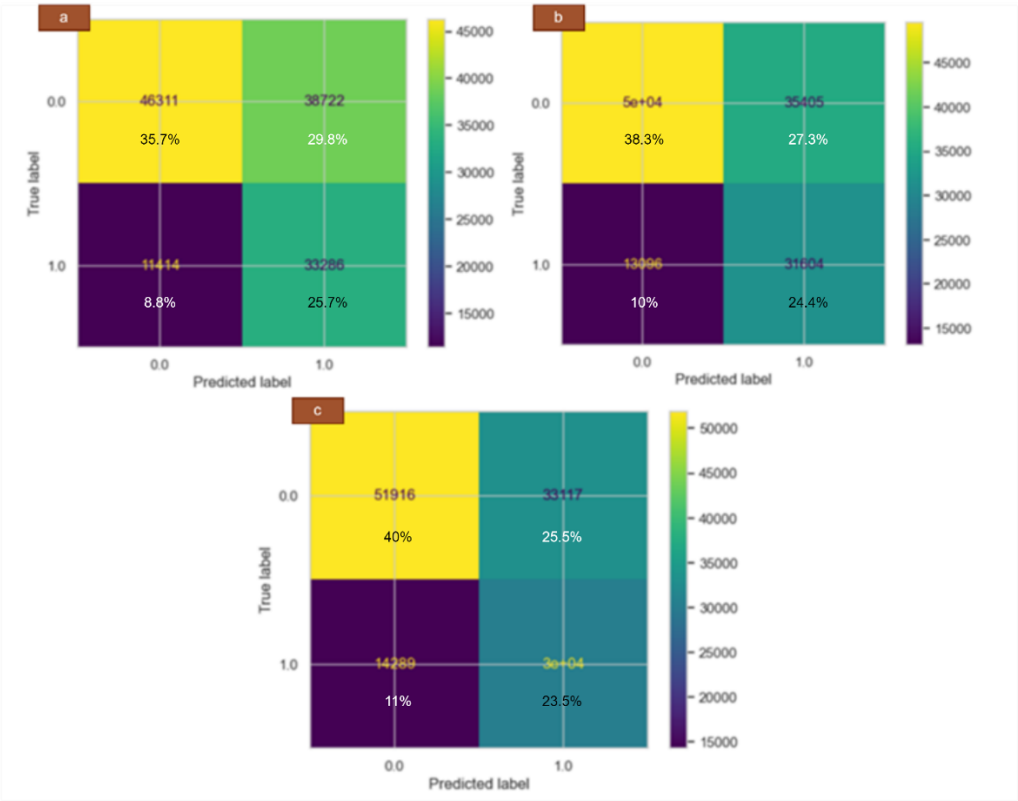


Figura 12. *Matriz de Confusão dos 3 modelos resultantes, sendo cada modelo indicado por A, B e C.*

Modelo	KS	Acurácia	AUROC	Recall
A	0.2956	0.6135	0.7016	0.7447
B	0.2947	0.6261	0.7002	0.7070
C	0.2928	0.6346	0.7013	0.6803

Tabela 2. *Conjunto de Métricas para avaliação dos resultados das 3 melhores estruturas do modelo DANets para a aplicação de Análise de Crédito.*

Observando a variação entre os resultados das três estruturas, é possível notar um desempenho condizente, quando compara-se com o Estado da Arte para essa aplicação (KS próximo a 0.32). Além disso, as métricas de acurácia e AUROC se mostram bastante sólidas, juntamente com a métrica de Recall, que também valida o resultado positivo.

Em seguida, temos os resultados dos Modelos Tradicionais de *Machine Learning*. É importante ressaltar que os resultados que serão expostos correspondem aos modelos já com o conjunto de hiperparâmetros de melhor desempenho, sendo o processo realizado pelo Optuna.

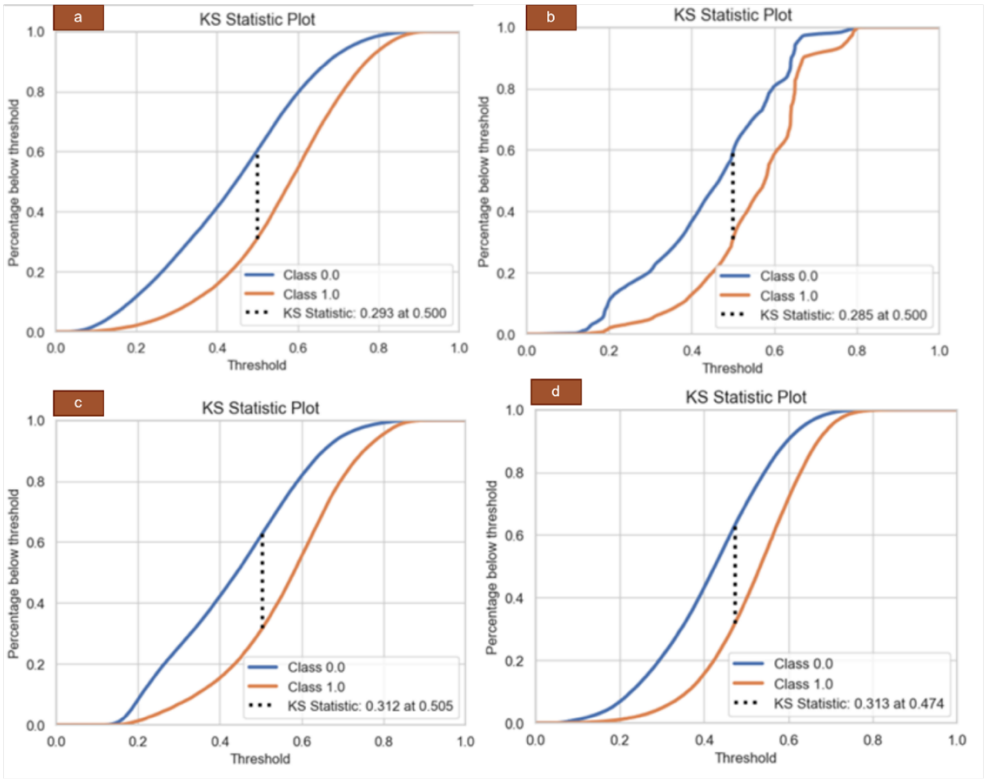


Figura 13. *Curva do KS dos 4 modelos tradicionais de Machine Learning. (a) Modelo de MLP, (b)Random Forest,(c)Gradient Boosting e (d)Ensemble Misto.*

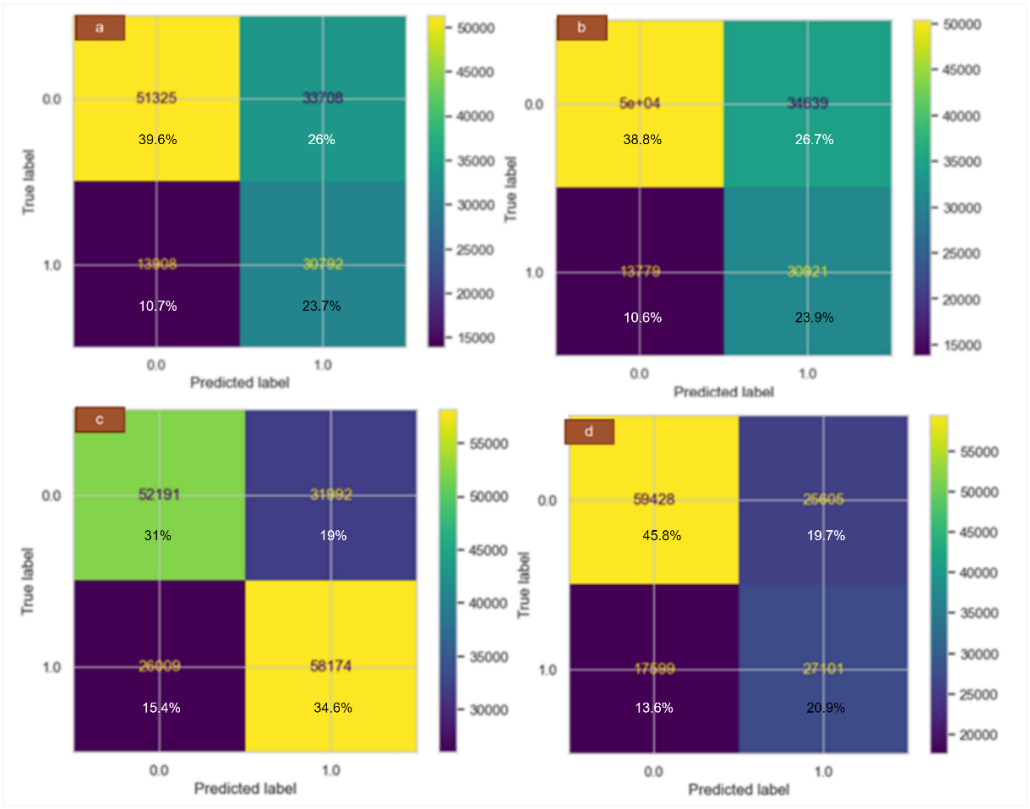


Figura 14. *Matriz de Confusão dos 4 modelos tradicionais de Machine Learning. (a) Modelo de MLP, (b)Random Forest, (c) Gradient Boosting e (d) Ensemble Misto.*

Modelo	KS	Acurácia	AUROC	Recall
MLP	0.2924	0.6330	-	0.6889
Random Forest	0.2846	0.6268	0.6939	0.6917
Gradient Boosting	0.3123	0.6555	0.7129	0.6910
Ensemble Misto	0.3134	0.6670	0.7126	0.6063
Modelo DANets A	0.2956	0.6135	0.7016	0.7447
Modelo DANets B	0.2947	0.6261	0.7002	0.7070
Modelo DANets C	0.2928	0.6346	0.7013	0.6803

Tabela 3. *Conjunto de Métricas para avaliação dos resultados dos 4 modelos tradicionais de Machine Learning, juntamente com os 3 melhores resultados do modelo DANets, para a aplicação de Análise de Crédito.*

Por fim, com todos os resultados dispostos, temos uma visão mais consistente em relação aos principais desempenhos dado o contexto do problema de Análise de Crédito (que, pelas características diversas e completas da base de dados, possui um viés de confirmação de desempenho ainda maior). Como esperado, modelos mais utilizados para o contexto de Dados Tabulares, como o Gradient Boosting e até o Ensemble Misto dos modelos mais comuns de Machine Learning obtiveram bons resultados, sendo o KS muito próximo ao Estado da Arte.

No entanto, fazendo um comparativo com a performance do DANets, a aplicação supera tanto o modelo ideal de Random Forest quanto o de MLP, mostrando-se, assim, com um bom resultado, em um contexto global.

5 – CONCLUSÃO

O trabalho realizado contribui para um entendimento e uma validação mais consolidada acerca do modelo DANets no contexto de Dados Tabulares, trazendo, também um embasamento teórico e comparativo em relação a outras aplicações. Assim, torna-se possível uma abertura na visão de aplicabilidade de modelos de Deep Learning no contexto de Dados Tabulares, mas ainda mantendo a consideração de performance de modelos mais tradicionais.

Diante de toda a pesquisa, percebe-se que a aplicação de *Deep Learning* para problemas com Dados Tabulares pode, e deve, ser considerada, a depender do contexto. Foi realizada uma explicação detalhada acerca das principais aplicações para a resolução de problemas, como o escopo de Machine Learning, Redes Neurais e Deep Learning, e, além disso, foi necessário um maior esclarecimento e explicação de tópicos pertinentes ao processo de estruturação do modelo estudado.

Em relação ao modelo DANets, foi possível inferir que a arquitetura, de certa forma, busca atenuar o gap das características dos Dados Tabulares, que possuem o agrupamento das informações de maneira mais heterogênea. Assim, por modelos de Deep Learning possuírem um viés mais centrado em Aprendizado por Localidade, busca por padrões e correlações entre as informações, o DANets desenvolve um meio para aproveitar essas características para esse contexto em que, não se tinha a possibilidade de se trabalhar anteriormente. Assim, por meio da estruturação da Máscara de Aprendível, tem-se uma abordagem de avaliação de características de correlação entre atributos, sendo um diferencial em relação a Modelos Tradicionais de *Machine Learning*. [1]

Com os resultados dispostos no Capítulo 4.1.2, percebe-se um desempenho competitivo do modelo DANets ao se comparar com os demais. Assim, mesmo com o fato de o modelo não ter a melhor performance de todas nessa aplicação, é importante pontuar que, em contextos mais complexos ou diferentes do que foi aplicado durante a fase de testes, o DANets pode se manter robusto e assertivo quando se fala de problemas com Dados Tabulares.

Por fim, é imprescindível que a pesquisa e o processo analítico sempre sejam mantidos, principalmente no âmbito da tecnologia, que sempre demanda renovação e busca por mais e melhores resultados ótimos para a resolução de problemas.

6 – REFERÊNCIAS BIBLIOGRÁFICAS

[1] (Chen et al.2022). *DANets: Deep Abstract Networks for Tabular Data Classification and Regression.*

[2] Peters, B. et al(2019). *Sparse Sequence-to-Sequence Models.*

[3] (Martins et al.2016). *From Softmax to Sparsemax: A Sparse Model of Attention and Multi-Label Classification.*

[4] (Borisov et al.2022). *Deep Neural Networks and Tabular Data: A Survey*

[5] (Shwartz-Ziv; Armon. 2021). *Tabular Data: Deep Learning is Not All You Need*

[6] O que são Redes Neurais e Deep Learning. Disponível em: **https://didatica.tech/introducao-a-redes-neurais-e-deep-learning/**

[7] Classification Problems Real-life Examples. Disponível em: **https://vitalflux.com/classification-problems-real-world-examples/**

[8] Metric for Classifiers. Medium. Disponível em: **https://medium.com/@bharti.kindra04/metric-for-classifiers-c4ad32e3553b**

[9] Léo Grinsztajn, Edouard Oyallon, Gaël Varoquaux. Why do tree-based models still outperform deep learning on typical tabular data?. NeurIPS 2022 Datasets and Benchmarks Track, Nov 2022, New Orleans, United States. ffhal-03723551v2

[10] Combatting Overfitting with Dropout Regularization. Medium. Disponível em: **https://towardsdatascience.com/combating-overfitting-with-dropout-regularization-f721e8712fbe#:~:text=Let's%20recap%20%E2%80%94%20dropout%20is%20a,the%20input%20and%20hidden%20layers**

[11] O que são Redes Neurais. IBM. Disponível em: **https://www.ibm.com/br-pt/topics/neural-networks**

[12] Revisiting Deep Learning Models for Tabular Data. Disponível em: **https://wandb.ai/sauravm/RTDL/reports/Revisiting-Deep-Learning-Models-for-Tabular-Data--VmlldzoxNDE1Njk0**

[13] Exploratory Data Analysis for Tabular Data. Disponível em: **https://neptune.ai/blog/exploratory-data-analysis-for-tabular-data**

[14] Tabular data as a challenge. Disponível em: **https://datuum.ai/media/tabular-data-as-a-challenge/**

[15] Tabular Presentation of Data: Meaning, Objectives, Features and Merits. Disponível em: **https://www.geeksforgeeks.org/tabular-presentation-of-data-meaning-objectives-features-and-merits/**

[16] Deep Learning: do Conceito às Aplicações. Medium. Disponível em: **https://medium.com/data-hackers/deep-learning-do-conceito-%C3%A0s-aplica%C3%A7%C3%B5es-e8e91a7c7eaf**

[17] Machine learning, explained. MIT Management Sloan School. Disponível em: **https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained**

[18] O que é bagging?. IBM. Disponível em: **https://www.ibm.com/br-pt/topics/bagging**

[19] O que é boosting?. AWS. Disponível em: **https://aws.amazon.com/pt/what-is/boosting/**

[20] Kolmogorov Smirnov Test: When and Where To Use It. Disponível em: **https://arize.com/blog-course/kolmogorov-smirnov-test/**

[21] A Short Chronology Of Deep Learning For Tabular Data. Disponível em: **https://sebastianraschka.com/blog/2022/deep-learning-for-tabular-data.html**

[22] Optuna. Disponível em: **https://optuna.org/**

[23] Top 6 Machine Learning Algorithms for Classification. Medium. Disponível em: **https://towardsdatascience.com/top-machine-learning-algorithms-for-classification-2197870ff501**

[24] (Nwankpa et al.2018). *Activation Functions: Comparison of trends in Practice and Research for Deep Learning.*

[25] Underfitting vs. Overfitting. Scikit-learn. Disponível em: **https://scikit-learn.org/stable/auto_examples/model_selection/plot_underfitting_overfitting.html**

[26] (Dimitriou; Arandjelovic 2020). *A New Look at Ghost Normalization*

[27] Datasets & Dataloaders. Pytorch. Disponível em: **https://pytorch.org/tutorials/beginner/basics/data_tutorial.html**

[28] Understanding Learning Rate. Medium. Disponível em: **https://towardsdatascience.com/https-medium-com-dashingaditya-rakhecha-understanding-learning-rate-dd5da26bb6de**

[29] Dicas de Aprendizado Profundo. Stanford.edu. Disponível em: https://stanford.edu/~shervine/l/pt/teaching/cs-229/dicas-aprendizado-profundo

[30] Capítulo 37 – O Efeito do Batch Size no Treinamento de Redes Neurais Artificiais. Deep Learning Book. Disponível em: **https://www.deeplearningbook.com.br/o-efeito-do-batch-size-no-treinamento-de-redes-neurais-artificiais/#:~:text=Tamanho%20do%20lote%20(Batch%20Size)%20%C3%A9%20um%20termo%20usado%20em,de%20itera%C3%A7%C3%A3o%20e%20%C3%A9pocas%20equivalentes.**

[31] A Comprehensive Guide on Optimizers in Deep Learning. Disponível em: **https://www.analyticsvidhya.com/blog/2021/10/a-comprehensive-guide-on-deep-learning-optimizers/**

[32] Capítulo 8 – Função de Ativação. Deep Learning Book. Disponível em: **https://www.deeplearningbook.com.br/funcao-de-ativacao/**

[33] Hyperparameter Tuning the Random Forest in Python. Disponível em: **https://towardsdatascience.com/hyperparameter-tuning-the-random-forest-in-python-using-scikit-learn-28d2aa77dd74**

[34] Understanding Loss Function in Deep Learning. Disponível em: **https://www.analyticsvidhya.com/blog/2022/06/understanding-loss-function-in-deep-learning/**

[35] 1.11 Ensemble methods. Scikit-learn. Disponível em: https://scikit-learn.org/stable/modules/ensemble.html#random-forests

[36] 1.17 Neural Network Models (supervised). Scikit Learn. Disponível em: https://scikit-learn.org/stable/modules/neural_networks_supervised.html

[37] O que diz o primeiro documento escrito da história. BBC News. Disponível em: https://www.bbc.com/portuguese/geral-39842626

[38] Introduction to Machine Learning vs Deep Leaning. Disponível em: https://hpc.mediawiki.hull.ac.uk/Programming/Deep_Learning