João Gabriel de Araújo Vasconcelos

TRANSFORMADORES EM DADOS TABULARES: UMA ANÁLISE COMPARATIVA

Trabalho de Graduação

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE
2023

Universidade Federal de Pernambuco
Centro de Informática
Graduação em Engenharia da Computação

João Gabriel de Araújo Vasconcelos

TRANSFORMADORES EM DADOS TABULARES: UMA ANÁLISE COMPARATIVA

Trabalho apresentado ao Programa de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Bacharel em Engenharia da Computação.

Orientador: *Germano Crispim Vasconcelos*

RECIFE
2023

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

Vasconcelos, João Gabriel de Araújo.
Transformadores em dados tabulares: uma análise comparativa. / João Gabriel de Araújo Vasconcelos. - Recife, 2023.
75 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.
Inclui referências, apêndices.

1. Inteligência Artificial. 2. Transformer. 3. Dados Tabulares. 4. Estatística. 5. Aprendizagem de Máquina. I. Vasconcelos, Germano Crispim. (Orientação). II. Título.

000 CDD (22.ed.)

João Gabriel de Araújo Vasconcelos
Transformadores em Dados Tabulares: Uma Análise Comparativa / João Gabriel de Araújo Vasconcelos. – RECIFE, 2023-
75 p. : il. (algumas color.) ; 30 cm.

Orientador Germano Crispim Vasconcelos

Trabalho de Graduação – Universidade Federal de Pernambuco, 2023.

1. Inteligência Artificial. 2. Transformer. I. Dados Tabulares. II. Universidade UFPE. III. Faculdade de Engenharia. IV. TRANSFORMADORES EM DADOS TABULARES: UMA ANÁLISE COMPARATIVA

CDU 02:141:005.7

Trabalho de conclusão de curso apresentado por **João Gabriel de Araújo Vasconcelos** ao programa de Graduação em Engenharia da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título **Transformadores em Dados Tabulares: Uma Análise Comparativa** , orientada pelo **Prof. Germano Crispim Vasconcelos** e aprovada pela banca examinadora formada pelos professores:

Documento assinado digitalmente
gov.br
ALEX SANDRO GOMES
Data: 26/09/2023 16:28:23-0300
Verifique em https://validar.iti.gov.br

Prof. Alex Sandro Gomes
Centro de Informática/UFPE

Documento assinado digitalmente
gov.br
GERMANO CRISPIM VASCONCELOS
Data: 26/09/2023 14:48:53-0300
Verifique em https://validar.iti.gov.br

Prof. Germano Crispim Vasconcelos
Centro de Informática/UFPE

RECIFE
2023

I dedicate this thesis to all my family, friends and professors
who gave me the necessary support to get here.

Agradecimentos

Primeiramente, quero expressar minha mais sincera gratidão à minha família, que sempre me deu apoio incondicional e teve confiança nas minhas capacidades e objetivos. Tanto meu pai, Wilmar, quanto minha mãe, Maria do Socorro, moldaram a pessoa que sou hoje e, por isso, eu sou imensamente grato. Agradeço também aos meus irmãos, Rafael e Luana, que não apenas foram referências absolutas, mas também serviram de porto seguro durante toda a minha caminhada até aqui. Suas palavras de encorajamento nos momentos difíceis e a alegria compartilhada nos momentos de celebração tornaram essa jornada muito mais significativa e memorável.

Aos meus amigos de Teresina-PI, meu agradecimento também é profundo. Pessoas com as quais cresci, evolui e compartilhei as mais diversas experiências até ingressar na Universidade. São amigos com os quais, independentemente da distância, sei que sempre poderei confiar. Aos amigos que fiz em Recife-PE, obrigado por terem me acolhido de forma tão generosa. Vocês não apenas compartilharam essa jornada acadêmica comigo, mas também foram um pilar de suporte na minha vida pessoal. A todos os meus amigos, à medida que avançamos em nossas respectivas trajetórias, espero que continuemos a nos apoiar e a compartilhar momentos de crescimento e alegria.

Gostaria de agradecer também ao meu orientador, Germano Crispim Vasconcelos, por ser uma referência na área que escolhi para atuar e por me guiar nessa fase acadêmica tão importante. Suas orientações e sugestões foram vitais para a realização desse projeto. Também nesse contexto, sou grato aos membros da banca por me acompanharem nessa etapa acadêmica, bem como a todo o corpo docente universitário, que me permitiu chegar a esse ponto, sempre me guiando de forma contundente nas mais diversas áreas.

A todos que tiveram um papel nessa jornada, direta ou indiretamente, meu mais sincero agradecimento.

João Gabriel de Araújo Vasconcelos

Nobody phrases it this way, but I think that artificial intelligence is almost a humanities discipline. It's really an attempt to understand human intelligence and human cognition.

—SEBASTIAN THRUN

Glossary

ARPANet Precursora da Internet. Primeira rede de computadores, construída em 1969 e desenvolvida pela ARPA (Advanced Research Projects Agency).. 31

Attention Mechanisms Componentes utilizados usualmente em modelos da arquitetura de Transformers que possibilitam a priorização de partes específicas dos dados de entrada.. 27–29

Auto-Regressivo Aspecto de um modelo em gerar uma sequência de Tokens utilizando previsões anteriores do mesmo modelo como entrada auxiliar.. 36

Decoder Porção de um modelo responsável pela transformação de um dado previamente transformado pelo Decoder para uma saída adequada.. 36

Dropout Técnica de Aprendizagem Profunda referente à desativação randômica de neurônios durante o treinamento para reduzir Overfitting e provêr menor interdependência entre as células de uma Rede Neural.. 36

Embedding Técnica para representar dados, geralmente palavras ou itens, em um espaço vetorial contínuo. É frequentemente utilizada no Processamento de Linguagem Natural para capturar relações semânticas entre palavras.. 28, 37, 39–42, 50

Encoder Porção de um modelo responsável pela transformação de uma entrada em um formato adequado para o processamento subsequente.. 36

Ensemble Técnica de combinação de modelos de aprendizado a fim de melhorar a performance preditiva de determinado sistema.. 13, 38, 56

Exploding Gradient Fenômeno em que os gradientes de um modelo de Redes Neurais Artificiais se tornam muito grandes durante o treinamento, impedindo o algoritmo de alcançar um "ponto mínimo" para a função de perda a partir do processo iterativo de otimização dos parâmetros.. 45

Feature Engineering Processo de criação, seleção e transformação de características para uma nova forma, a fim de melhor o desempenho de modelos de Aprendizado de Máquina.. 39

Feature Selection Evento de escolha de características mais relevantes em determinado modelo.. 38, 49

Gradiente Descendente Algoritmo de otimização utilizada na minimização de funções e posterior ajuste de parâmetros.. 35

Insights Informações significativas abstraídas da análise de dados.. 27

Interpretabilidade Capacidade de entender e explicar o funcionamento e as decisões tomadas por determinado modelo.. 27, 28, 33, 38, 65, 66

LabelEncoding Técnica de transfomração de variáveis categóricas em valores inteiros únicos.. 46

MinMaxScaling Transformação de dados de forma que os mesmos sejam escalados para o intervalo [0,1].. 45

Normalization Processo de escalar dados para que obedeçam regras de escala específicas.. 36

OneHotEncoding Técnica de transfomração de variáveis categóricas em vetores binários.. 46

Outliers Dados pontuais significativamente diferentes dos demais dados de uma amostra.. 26–28

Overfitting Evento de ocorrência relativo à capacidade de determinado modelo em aprender os dados de treinamento, mas performar de forma falha quando apresentado a dados novos, indicando uma incapacidade de generalização.. 34

Pipeline Sequência de processos executados em uma ordem específica a fim de ser chegar a um resultado.. 32, 41, 66

Regularization Técnica relativa ao acrésimo de "penalidades" aos coeficientes de determinado modelo, reduzindo a sua complexidade. É bastante utilizada como ferramente de mitigação ao Overfitting.. 36

Shrinkage Método estatístico de redução de magnitude de coeficientes de variáveis consideradas menos relevantes.. 34

Sigmoide Função de Ativação utilizada em Redes Neurais que transforma saídas de um Neurônio em valores no intervalo [0,1].. 47

Standardization Transformação de dados de forma que os mesmos passem a possuir uma média de 0 e um desvio padrão de 1.. 45

Trade-Off Melhoria de uma aspecto específico de um sistema em detrimento de outro.. 43

Vanishing Gradient Fenômeno em que os gradientes de um modelo de Redes Neurais Artificiais se tornam muito pequenos durante o treinamento, impedindo o algoritmo de alcançar um "ponto mínimo" para a função de perda a partir do processo iterativo de otimização dos parâmetros.. 45

Resumo

No contexto contemporâneo relativo a análise de dados, em que a obtenção e o processamento de informações tem se tornado cada vez mais presentes, surge a necessidade de métodos sofisticados que lidem com dados tabulares complexos. Tradicionalmente, os algoritmos com maior sucesso em preencher essa lacuna são algoritmos de *Ensemble*, como Árvores de Decisão Impulsionadas Por Gradiente (GBDT). Contudo, de modo a acompanhar a evolução exponencial que o campo da Inteligência Artificial tem sofrido nas últimas décadas, é conveniente analisar o potencial de modelos baseados em *"Transformers"* como uma alternativa promissora.

Esse trabalho de graduação aborda tal paradigma através de uma análise comparativa no quesito de dados tabulares, averiguando a perfomance obtida através de manipulações inspiradas nos modelos no estado da arte utilizados em Processamento De Linguagem Natural (PLN) em relação aos modelos tradicionais. O foco, por sua vez, recai especificamente sobre tarefas de classificação binária, com ênfase na capacidade de a nova abordagem capturar relações entre as características presentes em dados tabulares.

Através dessa análise, a pesquisa procura oferecer um panorama da viabilidade dessa nova abordagem, especificamente dos modelos TabNet, TabTransformer e FT-Transformer, no referente à performance, em comparação a uma gama de diferentes modelos.

Visto que o objetivo principal do trabalho é estudar o potencial dos modelos comentados anteriormente, a implementação do código foi realizada de modo a destacar as contribuições únicas e as limitações de cada abordagem, sem obscurecer as características intrínsecas dos dados, isto é, o foco principal foi na fase de treinamento e inferência e não na análise exploratória dos dados. Nesse contexto, selecionamos a estatística de teste Kolmogorov-Smirnov como nossa métrica principal de análise entre os modelos, além de procurarmos as melhores configurações hiper-paramétricas para cada abordagem.

Dado o aspecto contextual da solução obtida, o estudo tem sucesso em propor, implementar e validar os modelos levantados, de modo que os mesmos obtém uma performance competitiva em relação às abordagens mais tradicionais. Contudo, também é levantada/questionada uma perspectiva quanto à viabilidade dos novos modelos, tendo em visto uma limitação computacional e temporal para a obtenção dos mesmos.

Palavras-chave: Inteligência Artificial, Transformers, Dados Tabulares, Aprendizagem Profunda, Ensemble

Abstract

In the current context of data analysis, where the acquisition and processing of information have become increasingly prevalent, the need for sophisticated methods to handle complex tabular data arises. Traditionally, the algorithms most successful in bridging this gap are ensemble algorithms, such as Gradient Boosted Decision Trees (GBDT). However, in order to keep up with the exponential evolution that the field of artificial intelligence has undergone in recent decades, it is advisable to examine the potential of transformer-based models as a promising alternative.

This undergraduate thesis addresses this paradigm through a comparative analysis in terms of tabular data, investigating the performance achieved through manipulations inspired by state-of-the-art transformer models in relation to traditional models. The focus, in turn, specifically lies on classification tasks, with an emphasis on the new approach's ability to capture relationships among the features present in tabular data.

Through this analysis, the research aims to provide an overview of the feasibility of this new approach, specifically the TabNet and TabTransformer models, in terms of performance compared to a range of different models.

Since the main objective of the work is to study the potential of the aforementioned models, the code implementation was carried out to highlight the unique contributions and limitations of each approach, without obscuring the intrinsic characteristics of the data. In this context, we selected the Kolmogorov-Smirnov test statistic as our primary analysis metric between the models, and we sought the best hyperparameter configurations for each approach.

Given the contextual aspect of the obtained solution, the study successfully proposes, implements, and validates the raised models, demonstrating a competitive performance regarding more traditional approaches. However, a perspective on the viability of the new models is also raised, considering computational and time limitations for obtaining them.

Keywords: Artificial Intelligence, Transformers, Tabular Data, Deep Learning, Ensemble

Lista de Figuras

1.1 Esquema representativo da estrutura de uma Rede Neural arbitrária. 28

2.1 Exemplo de *Rede Neural Artificial*. O elemento "*a*" representa funções de ativação arbitrárias que agragam complexidade ao modelo. 36
2.2 Arquitetura *Transformer*. 37
2.3 Arquitetura *TabTransformer*. 39
2.4 Arquitetura *FT-Transformer*. 40
2.5 Partição do *Dataframe* de Treino. 41
2.6 Exemplo do *Pipeline* de alimentação dos Dados na arquitetura *Transformer* para o modelo FT-Transformer. 42
2.7 Exemplo representativo da etapa final de inferência no modelo FT-Transformer. 42
2.8 Representação arbitrária de uma Estatística-KS. A linha Azul representa a CDF emírica, enquanto a linha Vermelha representa a CDF de referência. A linha preta representa o Valor KS. 44

4.1 Gráfico KS do modelo Random Forest relativo ao Conjunto de Teste. 52
4.2 Matriz de Confusão do modelo Random Forest relativo ao Conjunto de Teste. . 52
4.3 Gráfico KS do modelo Gradient Boosting relativo ao Conjunto de Teste. 53
4.4 Matriz de Confusão do modelo Gradient Boosting relativo ao Conjunto de Teste. 53
4.5 Gráfico KS do modelo XGBoost relativo ao Conjunto de Teste. 54
4.6 Matriz de Confusão do modelo XGBoost relativo ao Conjunto de Teste. 54
4.7 Gráfico KS do modelo LGBM relativo ao Conjunto de Teste. 55
4.8 Matriz de Confusão do modelo LGBM relativo ao Conjunto de Teste. 55
4.9 Gráfico KS do modelo Ensemble relativo ao Conjunto de Teste. 56
4.10 Matriz de Confusão do modelo Ensemble relativo ao Conjunto de Teste. 56
4.11 Gráfico KS do modelo Perceptron Multilayer (MLP) relativo ao Conjunto de Teste. 57
4.12 Matriz de Confusão do modelo MLP relativo ao Conjunto de Teste. 57
4.13 Gráfico KS do modelo *TabNet* relativo ao Conjunto de Teste. 58
4.14 Matriz de Confusão do modelo *TabNet* relativo ao Conjunto de Teste. 58
4.15 Gráfico KS do modelo *TabTransformer* relativo ao Conjunto de Teste. 59
4.16 Matriz de Confusão do modelo *TabTransformer* relativo ao Conjunto de Teste. . 59
4.17 Gráfico KS do modelo *FT-Transformer* relativo ao Conjunto de Teste. 60
4.18 Matriz de Confusão do modelo *FT-Transformer* relativo ao Conjunto de Teste. . 60

Lista de Tabelas

4.1 Valores da métricas obtidas por cada modelo considerado no estudo. 61
4.2 Tempo que cada modelo levou para ser obtido (hh:mm:ss). 61

A.1 List of conferences on which the searches were performed. 73
A.2 List of journals in which the searches were performed. 74
A.3 Search string per Search Engine. 75

Lista de Acrônimos

NLP	Natural Language Processing	26
GBDT	Gradient Boosted Decision Trees	26
AM	Aprendizagem de Máquina	25
PLN	Processamento de Linguagem Natural	27
RNA	Redes Neurais Artificiais	26
MA	Mecanismos de Atenção	29
IA	Inteligência Artificial	31
RN	Redes Neurais	27
RF	Random Forest	27
GB	Gradient Boosting	27
XGB	Extreme Gradient Boosting	27
LGBM	Light Gradient Boosting Machine	27
MLP	Perceptron Multilayer	35
RNNs	Redes Neurais Recorrentes	33
RNCs	Redes Neurais Convolucionais	33
LLMs	Large Language Models	37

Sumário

1 Introdução **25**

1.1 Motivação . . . 25
1.2 Desafios e Complexidades . . . 26
1.3 Modelos Tradicionais e Transformers . . . 27
1.4 Objetivos da Pesquisa . . . 29

2 Fundamentação Teórica **31**

2.1 Introdução a Inteligência Artificial . . . 31
2.1.1 Aprendizagem de Máquina . . . 31
2.1.2 Aprendizagem Profunda . . . 32
2.2 Modelos . . . 33
2.2.1 Árvores de Decisão Impulsionadas por Gradiente . . . 34
2.2.1.1 Random Forest . . . 34
2.2.1.2 Gradient Boosting . . . 35
2.2.1.3 Extreme Gradient Boosting e Light Gradient Boosting Machine 35
2.2.2 Modelos baseados em Redes Neurais Artificiais . . . 35
2.2.3 Modelos Baseados em Transformers e Mecanismos de Atenção . . . 36
2.2.3.1 TabNet . . . 38
2.2.3.2 TabTransformer . . . 39
2.2.3.3 FTTransformer . . . 40
2.2.3.4 Exemplificação com Transformers . . . 40
2.3 Métricas de Avaliação . . . 43

3 Metodologia **45**

3.1 Introdução . . . 45
3.2 Objetivo e Dados Analisados . . . 45
3.2.1 Tratamento dos Dados . . . 45
3.3 Otimização e Métricas de Avaliação . . . 46
3.3.1 Especificações Técnicas . . . 47
3.4 Variações dos Algoritmos Estudados . . . 47
3.4.1 Gradient Boosted Decision Trees (GBDT) e seus Hiperparâmetros . . . 48
3.4.1.1 Random Forest . . . 48
3.4.1.2 Gradient Boosting . . . 48
3.4.1.3 XGBoost . . . 48
3.4.1.4 Light Gradient Boosting Machine . . . 48

3.4.1.5 Ensemble de Gradient Boosting Decision Trees 48
3.4.2 Implementação em Redes Neurais Tradicionais 49
3.4.2.1 Perceptron Multicamadas 49
3.4.3 Implementação de Modelos Baseados em Transformers 49
3.4.3.1 TabNet 49
3.4.3.2 TabTransformer 50
3.4.3.3 FT-Transformer 50
3.5 Avaliação dos Modelos 50

4 Experimentos e Resultados **51**
4.1 Introdução 51
4.2 Descrição dos Dados 51
4.3 Resultados e Desempenho 52

5 Discussão dos Resultados **63**
5.1 Introdução 63
5.2 Comparativo de Performance 63
5.3 Viabilidade e Limitações 63

6 Conclusão **65**
6.1 Recapitulação 65
6.2 Pesquisas Futuras 65

Referências **69**

Apêndice **71**

A Mapping Study's Instruments **73**

1 Introdução

Desde sua origem, compreendendo métodos rudimentares relacionados unicamente ao catálogo manual em planilhas e métodos similares, a análise de dados sempre foi uma área extremamente importante. Tal aspecto, embora tenha perdurado por um longo período, tem se intensificado cada vez mais nas últimas décadas, principalmente em virtude da evolução tecnológica emergente e frenética que vivenciamos.

O aspecto histórico dos diversos métodos utilizados na análise de dados contempla uma busca constante por eficiência e precisão. À medida que essa busca evoluiu, várias áreas foram sendo incorporadas de forma orgânica para otimizar os resultados almejados para os problemas apresentados, e o campo da Aprendizagem de Máquina (AM) é um exemplo notável dessa evolução. Tarefas que anteriormente exigiam apenas comparações simples entre grupos, formulações de hipóteses rudimentares sobre uma população e análises estatísticas básicas de um banco de dados passaram a incorporar uma gama de paradigmas mais complexos, incluindo algoritmos sofisticados de previsão, classificação, análises temporais, entre outros.

Em meio a essa revolução tecnológica sem precedentes, os dados tabulares emergem como uma forma única e essencial de representação de informações, encontrando aplicação em diversos campos, como finanças, varejo e medicina. Os dados tabulares são informações organizadas em forma de tabela, de forma que podem ser apresentados em linhas e colunas, onde cada linha representa uma entrada distinta, e cada coluna representa um atributo específico associado a essa entrada. No entanto, à medida que a área da ciência de dados expande para incorporar cenários mais complexos, surge também a demanda por métodos analíticos igualmente sofisticados, capazes de lidar com a crescente complexidade desses dados e de extrair informações valiosas.

1.1 Motivação

Apesar de os métodos atuais relativos à análise de dados se mostrarem eficazes nos mais variados cenários e compreenderem a maior porção de uso em cenários de classificação e regressão, eles ainda são pressionados pela complexidade crescente acoplada aos dados tabulares.

Nesse contexto, áreas como AM e mineração de dados tendem a fornecer alternativas como *Gradient Boosted Decision Trees (GBDT)*, demonstrando sucesso nos mais variados cenários relativos a dados de representação tabular. Essas técnicas tradicionais (baseadas na combinação de modelos mais fracos - árvores de decisão individuais) possuem a capacidade de capturar relações complexas entre os atributos de uma instância, indentificando padrões e, posteriormente, inferindo previsões precisas.

Por outro lado, a ascensão de abordagens baseadas na arquitetura de *"Transformers"* no campo de *Natural Language Processing (NLP)* potencializou uma nova dimensão no ecossistema de Redes Neurais Artificiais (RNA). Tradicionalmente, tal paradigma foi desenvolvido para tratar de problemas relacionados a cenários de nicho mais sofisticados e específicos, como abstração de textos em contexto vetorial para alimentação em modelos de AM, tratamento de dados temporais, cenários de inferência de imagem, entre outros. Contudo, a complexidade dessa arquitetura também estimulou pesquisas sobre o seu potencial na análise de dados tabulares.

Diante do interesse em capturar relações mais complexas no contexto de dados em tabela (provendo previsões mais acertivas), esses modelos sofisticados, que foram desenvolvidos inicialmente para compreender estruturas de dados sequenciais em forma de texto, foram adaptados para lidar com a complexidade inerente aos dados tabulares.

A motivação do estudo compreende a procura por uma compreensão mais acertiva das perspectivas dessas abordagens emergentes para lidar com a complexidade de dados estruturados em tabela.

1.2 Desafios e Complexidades

A complexidade intrínseca da análise de dados tabulares traz desafios novos. Muitas vezes, os dados estruturados apresentam aspectos não esperados, como relações não lineares complexas, distribuições variáveis, interações fortes entre os atributos, etc. Em meio a esse aspecto, a identificação de padrões signicativos requer abordagens analíticas que possam capturar nuances sutis, o que deve garantir conclusões mais "acertivas".

Além disso, na própria etapa de análise e pré-processamento dos dados, que pode ser demasiadamente exaustiva, outra complicação surge com a incorporação de dados ausentes ou inconsistentes (errôneos). Em um cenário real, como na extração de informações financeiras de uma companhia, é comum que erros ocorram e células que não deveriam ser capturadas sejam incluídas na análise. Por isso, compreender as limitações de cada abordagem tomada para inspeção das informações, bem como seus respectivos funcionamentos e tomadas de decisão é vital para fornecer justificativas claras e resultados congruentes.

Um exemplo tangente do que foi discutido anteriormente são modelos de *K-Nearest Neighbors (KNN)*, por exemplo, que podem ser sensíveis a *Outliers*, requerindo um maior tempo reservado ao pré-processamento de dados, enquanto modelos baseados em "Ensemble", como *GBDT*, possuem uma capacidade razoável em lidar com *Outliers* e distribuições diferentes.

1.3 Modelos Tradicionais e Transformers

Antes de adentrar nas perspectivas de análise comparativa entre os modelos tradicionalmente utilizados e *Transformers*, é instrutivo atentar quanto às nuâncias de cada abordagem. Historicamente, à medida que a quantidade e complexidade dos dados aumentavam, surgia também a necessidade de abordagens mais eficazes e automatizadas. Em virtude disso, a análise de dados, que antes era realizada manualmente e feita por meio de planilhas e visualizações gráficas simples, passou a expandir seu potencial, empregando modelos automatizados e complexos e não dependendo unicamente da intuição e conhecimento do analista/cientista sobre *Insights* qualitativos.

Em congruência com o que foi discutido, métodos variados, como Algoritmos de Regressão Linear, Regressão Logística, Máquinas de Vetores de Suporte (SVM) e RNA emergiram no contexto da automatização de tarefas de análise, cada uma com suas nuâncias. No contexto da arquitetura de *Transformers* em Processamento de Linguagem Natural (PLN), técnicas derivadas de *Attention Mechanisms*, como *TabNet* e *TabTransformer*, trouxeram uma nova perspectiva, possibilitando a captura de relações entre variáveis e uma Interpretabilidade aprimorada.

Em conformidade à averiguação realizada nesse estudo, modelos tradicionais são representados por algoritmos de *GBDT*, especificamente: *Random Forest (RF)*, *Gradient Boosting (GB)*, *Extreme Gradient Boosting (XGB)*, *Light Gradient Boosting Machine (LGBM)* e, por conveniência e quesito de comparação, também foram incluídas Redes Neurais (RN) Profundas convencionais. Por outro lado, a abordagem baseada em *Transformers* inclui modelos que utilizam técnicas estudadas após a publicação do artigo *Attention is All You Need* (VASWANI et al. (2017)), especificamente: *TabNet*(ARIK; PFISTER (2021)), *TabTransformer*(HUANG et al. (2020) e *FT-Transformer*(GORISHNIY et al. (2023))).

1. **Modelos Tradicionais de GBDT.** Os modelos tradicionais, como as árvores de decisão impulsionadas por gradiente (*GBDT*), baseiam-se em uma combinação ponderada de várias árvores de decisão individuais. Esses modelos têm se mostrado altamente eficazes em lidar com a complexidade dos dados tabulares, capturando interações entre variáveis e padrões não-lineares. As *GBDTs* podem lidar com valores ausentes e são relativamente resistentes a *Outliers*, especialmente quando utilizam um número suficiente de árvores para suavizar o impacto de valores extremos. Elas têm a capacidade de modelar relações complexas entre variáveis e realizar tarefas de previsão e classificação com precisão.

   No entanto, as *GBDT*s podem ter desafios ao lidar com dados de alta dimensionalidade e conjuntos de dados extremamente grandes, devido à necessidade de considerar todas as combinações possíveis de variáveis. Além disso, apesar de sua *Interpretabilidade* ser visualmente facilitada, ainda por ser limitada, uma vez que é difícil discernir os motivos pelos quais uma decisão específica foi tomada.

2. **Modelos Tradicionais de RNA.**As Redes Neurais Artificiais têm demonstrado sucesso em diversos domínios de AM e são conhecidas por sua capacidade de aprender representações complexas dos dados. Essa abordagem pode lidar com uma variedade de desafios, incluindo valores ausentes e diferentes distribuições. Redes Profundas podem capturar relações não-lineares e interações complexas entre variáveis. No entanto, a sensibilidade a *Outliers* pode variar dependendo da arquitetura da rede e do tratamento de dados realizado.

   As RNs também têm a capacidade de realizar tarefas de Interpretabilidade, com abordagens como *"Feature Importance"* e *"Layer-wise Relevance Propagation"* ajudando a entender as contribuições de cada variável para as previsões realizadas. No entanto, Redes Profundas podem ser mais difíceis de interpretar em comparação com modelos mais simples.

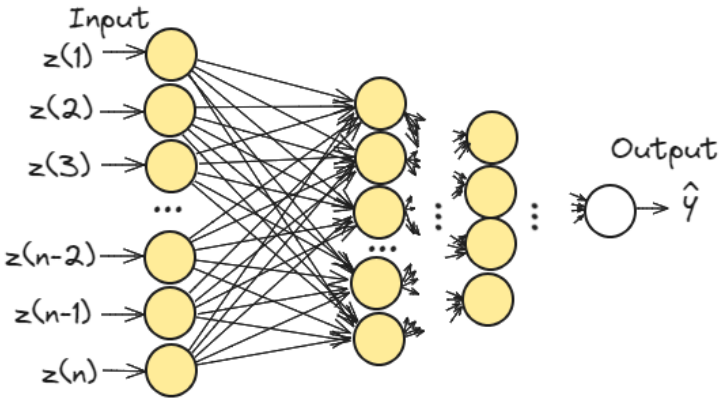

Figure 1.1: Esquema representativo da estrutura de uma Rede Neural arbitrária.

3. **Abordagem Baseada em Transformers.** A abordagem baseada em *Transformers*, originalmente desenvolvida para *PLN*, demonstrou sua capacidade de extrapolar além do processamento de sequências de palavras e abordar a complexidade dos dados tabulares. Modelos como *TabTransformer* e *FT-Transformer* empregam a arquitetura *Transformer* para capturar relações entre variáveis em dados tabulares. Eles usam mecanismos de atenção para focar nas interações mais relevantes entre as colunas e aprender representações ricas dos dados.

   Uma das vantagens desses modelos é a *Interpretabilidade* aprimorada fornecida pela etapa de *Embedding*. Os *Attention Mechanisms* permitem que os usuários entendam como as decisões são tomadas, identificando quais características tiveram maior influência nas previsões ou classificações. Essa capacidade de explicar as decisões é crucial em cenários onde a transparência é essencial. Todavia, vale ressaltar que, para esse trabalho, a busca de *Performance* foi sobreposta ao interesse de *Interpretabilidade*.

Outro ponto importante é que modelos baseados na arquitetura *Trasformers* podem exigir conjuntos de dados de treinamento maiores para atingir um desempenho comparável às *GBDTs*, especialmente em tarefas com menos exemplos. Além disso, o treinamento de modelos que usam *Transformers* pode ser computacionalmente mais intensivo.

1.4 Objetivos da Pesquisa

De modo a compreender a eficácia de abordagens distintas - Modelos Tradicionais de *GBDT*, Modelos Tradicionais de *RNA* e Modelos baseados em *Transformers* - na análise de dados tabulares, essa pesquisa tende a apresentar uma análise comparativa quanto à *Performance* de cada modelo sobre um conjunto de dados real. Através dessa análise, pode-se responder a seguinte pergunta:

Questionamento: *Como se compara o desempenho de Modelos Tradicionais (GBDT), Redes Neurais Artificiais (RNA) e Transformers e Mecanismos de Atenção (TabNet, TabTransformer e FT-Transformer) na análise de Dados Tabulares Complexos?*

Para responder esse questionamento, foi-se necessário realizar uma revisão abrangente e extensiva da literatura relacionada a todos os métodos comentados anteriormente. Posteriormente, realizou-se uma coleta e pré-processamento de dados adequados (representativos de cenários reais de dados tabulares) e uma implementação congruente e "justa" dos modelos selecionados. Com os dados em mão e os modelos preparados, foram feitas uma série de experimentos a fim de avaliar o desempenho preditivo de cada abordagem, avaliando as nuâncias dos resultados individuais obtidos. Por fim, a comparação dos resultados obtidos constituiu um indício de adequação de *Performance* para alguns modelos quando comparados a outros. Para essa pesquisa, especificamente, busca-se que as abordagens baseadas em *Transformers* e *Attention Mechanisms* performem tão bem quanto ou melhor que as alternativas relativas ao estado-da-arte para dados tabulares (*GBDT*).

Objetivo da Pesquisa *Esse trabalho propõe a realização de uma comparação prática de desempenho entre os modelos baseados em Transformers e Mecanismos de Atenção (MA), modelos baseados em RNA e modelos baseados em AM Tradicional para dados tabulares. É esperado que os resultados das abordagens baseadas em Transformers e MA minimamente atinjam os resultados obtidos pelas alternativas utilizadas no estado-da-arte.*

2

Fundamentação Teórica

2.1 Introdução a Inteligência Artificial

A *Inteligência Artificial (IA)* se destaca como uma área da Ciência da Computação cujo foco abrange a realização de tarefas que normalmente requerem a inteligência humana. De modo a replicar o pensamento e a tomada de decisão humanos, esse conceito passou a englobar um conjunto de técnicas e metodologias variadas, cada uma com suas nuâncias de funcionamento e aplicação.

A história da *IA* e sua relação com a estatística remetem a seu nascimento, em meados nos anos 1950, quando pesquisadores começaram a explorar como as máquinas poderiam imitar a inteligência humana.

É discutido que após a revolução provocada com a instituição da *Internet*, cuja tecnologia se iniciou em um contexto muito mais reservado do que é visto hoje em dia (com a ARPANet, em 1969), a *IA* se destaca como um novo tipo de revolução tecnológica que deve interferir de forma drástica no funcionamento econômico mundial. Na última década, a área transformou diversas indústrias e campos, incluindo finanças, saúde, varejo, entretenimento, entre outras. Seu uso, assim como a *Internet*, passou a integrar o cotidiano popular de forma orgânica e muitas vezes imperceptível, como acontece em Sistemas de Recomendação Personaliza, Automação de Processos, Diagnósticos Médicos , etc.

Ademais, relativo ao objetivo dessa pesquisa, a *IA* proporciona um contexto sólido para entender as diferentes abordagens utilizadas para lidar com a complexidade dos dados tabulares. Os progressos na área permitiram o desenvolvimento de técnicas que exploram a capacidade de computadores em estabelecer relações e tomar decisões, sendo essencial na análise moderna de dados.

2.1.1 Aprendizagem de Máquina

Destacando-se como uma vertente fundamental da *IA*, o *AM* se baseia em princípios estatísticos provendo a capacitação de sistemas cuja aprendizagem é realizada a partir da alimen-

tação de dados. Como comentado anteriormente, a *IA* engloba um acervo amplo de técnicas e sistemas que tem como objetivo simular a inteligência humana, baseando-se em lógica, regras, algoritmos e modelos estatísticos. Contudo, o *AM* contempla uma abordagem específica dentro da anterior, utilizando técnicas para identificar e abstrair padrões a partir de dados e, só então, tomar decisões lógicas e contundentes.

De forma sucinta, o *AM* permite que os sistemas aprendam a partir de dados em vez de seguir cegamente regras pré-estabelecidas. Essa área apresenta um processo de funcionamento bem estruturado, mas que pode variar levemente de acordo com o *Pipeline* requerido pelo usuário, sem sacrificar seu aspecto investigativo e preditivo. Esse processo, apesar de potencialmente apresentar leves variações, é usualmente estruturado da seguinte forma:

1. **Captura do Conjunto de Dados:** *O usuário faz a captação dos dados sobre os quais pretende fazer uma análise e uma potencial previsão quando apresentado a dados inéditos. Deve-se atentar quanto à coleta dos dados, visto que diferentes modelos reagem de formas diferentes de acordo com os atributos dos dados coletados no referente à distribuição, presença de dados errôneos ou nulos, presença de dados extrapolados, etc.*

2. **Análise:** *As características intrínsecas aos dados captados são analisadas de forma abrangente e intensiva, levantando atributos, como formatação, bem como destacando o objetivo da análise. Caso necessário, transformações são feitas nos dados de modo que a análise não seja comprometida e que, posteriormente, os memos possam ser "alimentados" para os modelos estatísticos.*

3. **Seleção de Modelo:** *Com base na análise realizada e no objetivo declarado, como classificação, regressão e agrupamento (Clustering), é selecionado e construído um modelo estatístico capaz de capturar os padrões dos dados.*

4. **Aprendizagem:** *O modelo selecionado na fase anterior é alimentado com os dados trabalhados, de modo que o mesmo se torne capaz de abstrair/inferir novas informações com base no que "aprendeu" estatisticamente.*

5. **Teste:** *Finalmente, o modelo é testado com novos dados que seguem a mesma distribuição dos dados originais (a partir de novas instâncias), inferindo na Performance do modelo.*

2.1.2 Aprendizagem Profunda

Remotando às decadas de 1950 e 1960, a *Aprendizagem Profunda*, também conhecida como *Deep Learning*, destaca-se como uma abordagem do *AM* que possui inspiração no funcionamento do cérebro humano, capacitando a criação de modelos baseados na abordagem de *RNs* de caráter complexo. Um de seus principais aspectos, que a difere de abordagens em *AM* usual, é

que ela busca isentar (ou diminuir) o papel exaustivo da extração manual de caracterírsticas (o que geralmente é responsabilidade do programador), sendo capaz de *"aprender"* características relevantes dos dados de forma automática. Esse aspecto torna essa abordagem ideal para a realização de tarefas específicas, como *Reconhecimento de Imagens*, *Reconhecimento de Áudio*, *PLN*, etc.

Um ponto que vale salientar é que, a fim de capacitar sua performance, o modelo baseado em *Aprendizagem Profunda* necessita de uma grande quantidade de dados para que possa *"competir"* com modelos mais tradicionais. Nas últimas décadas, contudo, o advento da *Internet* possibilitou uma disponibilidade massificada de dados, permitindo treinar esses modelos com maior precisão.

Seu destaque também possui alta influência em virtude dos avanços em *Hardware* sofridos nas últimas décadas, com *GPUs - Graphics Processing Units* (Unidades de Processamento Gráfico) - e *TPUs - Tensor Processing Unit* (Unidades de Processamento em Tensor) - que aumentam exponencialmente o treinamento e inferência de *RNs Profundas*. Além disso, novos algoritmos e técnicas, como *Redes Neurais Convolucionais (RNCs)* e *Redes Neurais Recorrentes (RNNs)*, foram desenvolvidos para melhorar seu desempenho e casos de uso.

Atualmente, a abordagem de *IA* em *RNAs* possui casos de uso variados, estando presente em projetos que englobam *Visão Computacional* (como *Sistemas de Reconhecimento Facial* e *Análise de Imagens*), *PLN* (como *Chatbots*, *Tradução de Textos*, *Análise de Sentimento*), Saúde (como no auxílio de *Diagnósticos Médicos*), entre outros casos de uso variados.

Todavia, sua alta eficácia em tarefas complexas não infere uma performance otimizada no tocante a dados tabulares. A simplicidade e *Interpretabilidade* de métodos mais tradicionais muitas vezes superam os benefícios oferecidos pela abordagens de *RNs*. Dados tabulares, usualmente, não possuem uma estrutura sequencial que possa ser explorada e aproveitada por *RNs*. Além disso, a dificuldade de *Interpretabilidade*, a necessidade de um *Hardware* sofisticado, a alta dimensionalidade das tabelas e o requerimento de um conjunto de dados massivo são fatores que limitam sua aplicação. Nesse quesito, técnicas que utilizam conceitos recentes de *Transformers* e *MA* surgem como alternativas em *RNs* para competir com os modelos de *AM*.

2.2 Modelos

Dentre os vários algoritmos, representações matemáticas e técnicas empregadas no contexto de classificação e tratamento em dados tabulares, considerando um aspecto histórico, os modelos que melhor performam são os baseados em *GBDT*. Apesar de existir um potencial de casos de uso para modelos de classificação mais primitivos em bases de dados simples, como *Modelos de Regressão Logística* e *"Support Vector Machines"*, esse trabalho, por apresentar um estudo relativo à inserção de novas abordagens em um contexto competitivo ao estado-da-arte, dirige suas considerações a 3 abordagens específicas: Modelos baseados em *GBDT*, Modelos Baseados em *RNs Usuais* e Modelos Baseados em *Transformers* e *MA*.

2.2.1 Árvores de Decisão Impulsionadas por Gradiente

Os modelos de *Árvores de Decisão Impulsionadas por Gradiente (GBDT)* são uma gama de algoritmos de *AM* utilizados em tarefas de *Regressão* e *Classificação*. O funcionamento dos modelos dessa abordagem se baseia em técnicas de *Ensemble Learning* (Aprendizagem em Conjunto), o que se refere à combinação de múltiplos modelos a fim de obter um modelo com performance otimizada. Alguns aspectos importantes dessa metodologia são o aspecto de Modelo Aditivo, em que as árvores são adicionadas sequencialmente e cada árvore foca nas amostras classificadas erroneamente pelas árvores anteriores. Além disso, como o nome sugere, o próprio processo de treinamento é orientado por um algoritmo de *Aprendizagem por Gradiente* em busca da minimização da função de perda. Por fim, as técnicas baseadas nessa abordagem usualmente incluem técnicas de *regularização*, como *Shrinkage* (Encolhimento) e Limitação da Profundidade das Árvores para evitar *Overfitting*.

Em meio à gama de modelos baseados na abordagem de *GBDT*, as que são estudadas nesse trabalho são, especifiamente: *RF, GB, XGB e LGBM*.

2.2.1.1 Random Forest

Random Forest (RF) é um algoritmo baseado em conjunto que constrói múltiplas Árvores de Decisão durante o treinamento e combina suas previsões para obter resultados precisos. Cada Árvore é treinada em uma amostra aleatória do conjunto de dados e produz uma previsão. As previsões são, por fim, feitas por voto majoritário (no contexto de *Classificação*). Seu funcionamento, usualmente, possui a seguinte configuração:

1. **Amostragem:** *Antes mesmo de construir qualquer árvore, é realizado uma amostragem aleatória do conjunto de dados de treinamento "com reposição" (cada elemento é devolvido à população antes da próxima seleção). Tal aspecto permite que algumas instâncias apareçam repetidamente na amostra.*

2. **Treinamento:** *Cada árvore é treinada utilizando um subconjunto de dados obtidos na etapa anterior. As árvores seguem um processo de divisão recursiva, procurando dividir os dados em grupos de acordo com suas características e com um critério de divisão previamente estabelecido. Tal critério, como Entropia ou Índice Gini, é utilizado exclusivamente para determinar a melhor forma de divisão dos dados em cada "nó" da árvore.*

3. **Votação:** *Após o treinamento de todas as árvores, é realizada uma combinação de cada árvore para fazer uma previsão final. Especificamente para esse projeto, em que é analisado um problema de Classificação, a combinação é feita pelo critério de votação majoritária, em que a classe mais comum prevista por todas as árvores é escolhida como previsão final.*

2.2.1.2 Gradient Boosting

Gradient Boosting define uma abordagem generalista que "impulsiona" *Árvores de Decisão*. Ele se baseia na construção sequencial de árvores de decisão (ou outros modelos fracos) a fim de criar um modelo robusto. Essa técnica constitui um processo iterativo que busca minimizar a função de perda, utilizando algoritmos de otimização sofisticados, como *Gradient Descent* - Gradiente Descendente. De forma geral, esse modelo se diferencia dos modelos de *RF* no quesito de abordagem de aprendizagem, visto que em *GB* as árvores de decisão são treinadas em sequência com cada árvore se ajustando aos erros residuais das árvores anteriores, e não de forma independente.

2.2.1.3 Extreme Gradient Boosting e Light Gradient Boosting Machine

Tanto *XGB* quanto *LGBM* são definidos como implementações eficientes de *GB*, incluindo uma gama de recursos como regularização, tratamento de valores ausentes, gerenciamento de características, possibilidade de paralelismo no treino e inferência. Esses modelos performam de modo bastante similar, apresentando pequenas nuâncias de aprendizagem.

2.2.2 Modelos baseados em Redes Neurais Artificiais

Os modelos baseados em *RNA*, como *Perceptron Multilayer (MLP) - Multilayer Perceptron* - são definidos como uma classe de algoritmos de *IA* inspirada no funcionamento do cérebro humano. Para esse trabalho, a contexto da visibilidade de potencial do método, um dos modelos estudados é o *Perceptron Multicamadas*, o qual é composto por várias camadas de unidades de processamento denominadas *Neurônios Artificiais*. Esse modelo consiste basicamente de uma camada de entrada, uma ou mais camadas ocultas e, por fim, uma camada de saída, em que cada neurônio contido em uma camada está conectado a todos os neurônios da camada seguinte.

Através dessa rede, o método de *Forward Propagation* define processo em que a entrada é passada através das camadas ocultas até a camada de saída, gerando uma previsão. Cada neurônio nas camadas ocultas aplica uma transformação linear seguida de uma função de ativação. Posteriormente, é realizado o *Backpropagation*, que é o processo de atualização dos parâmetros da *RNA* com base no que foi "aprendido" no treino. Ele calcula o Gradiente da função de perda em relação aos pesos, começando pela camada de saída e retrocedendo até a camada de entrada. Esses Gradientes são usados para ajustar os pesos de forma a minimizar a perda durante o treinamento.

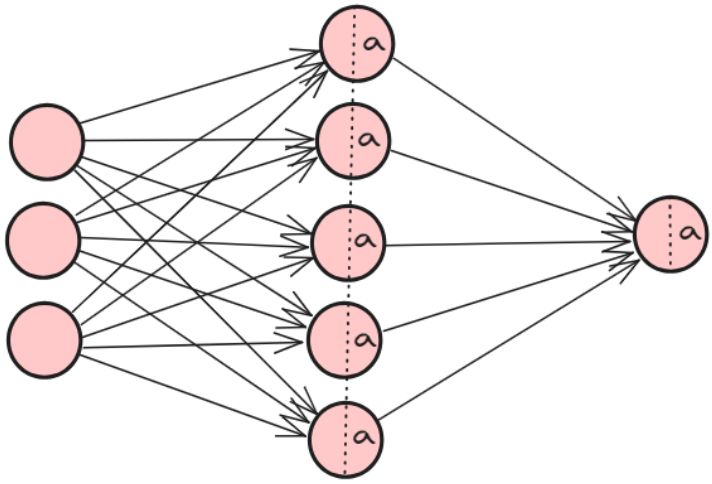

Figure 2.1: Exemplo de *Rede Neural Artificial*. O elemento "*a*" representa funções de ativação arbitrárias que agragam complexidade ao modelo.

Esses modelos possuem inúmeras nuâncias e melhorias que podem ser manipuladas de acordo com o problema apresentado e os dados trabalhados, como implementação de *Regularization, Normalization, Dropout, Variação de Camadas de Ativação*, etc.

2.2.3 Modelos Baseados em Transformers e Mecanismos de Atenção

Os modelos baseados nos conceitos de *MA* e na arquitetura moderna de *Transformers* são uma partição de modelos de *RNA* que se destacam em tarefas de *PLN* e sequências em geral, como tarefas de tradução e geração de texto.

Os *MA* são componentes importantes centrais dos modelos baseados em *Transformers*. Esses mecanismos possibilitam que o modelo estudado atribua pesos diferentes a cada segmento da entrada em questão, priorizando os elementos relevantes para a tarefa. O funcionamento do mesmo é descrito através do cálculo de *"Pesos de Atençao"* baseados na semelhança entre os elementos atribuídos. Nesse contexto, os valores dos pesos são encontrados através de produtos escalares entre *"Vetores de Consulta"*, *"Vetores de Chave"* e *"Vetores de Valor"*, o que resulta em uma representação contextualizada da entrada.

A arquitetura de *Transformers*, por sua vez, foi introduzida pela primeira vez em 2017 no artigo *Attention is All You Need* (VASWANI et al. (2017)) e desde então tem revolucionado a forma como *PLN* é realizado. A arquitetura é composta por camadas de *"Auto-Atenção"*, permitindo que o modelo capture as relações de longa distância que possam existir nos dados.

Nessa arquitetura, existe um *Encoder* responsável por mapear uma sequência de símbolos de entrada para uma sequência de representações contínuas. A partir dessa última sequência, o *Decoder* gera uma sequência de saída dos símbolos em cada passo de tempo. Cada passo, por sua vez, é *"Auto-Regressivo"*, tomando os símbolos gerados no passo anterior como entrada adicional na geração de texto. A Figura 2.2 ilustra essa arquitetura.

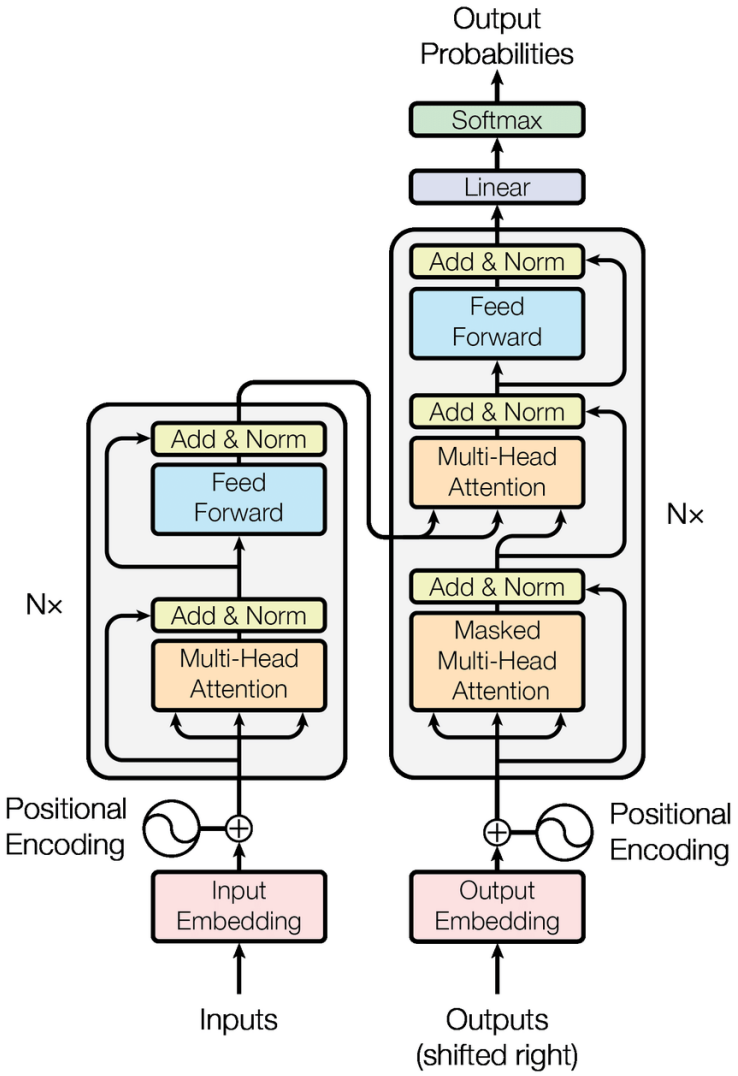

Figure 2.2: Arquitetura *Transformer*.

Resumidamente, o *Transformer* consiste em uma camada *Multi-Head* de *Auto-Atenção* seguida por uma camada *Feed-Forward*, juntamente à adição elemento a elemento e normalização da camada sendo feitas após cada camada.

A camada de *Auto-Atenção*, por sua vez, consiste em 3 matrizes: Chave (*Key*), Consulta (*Query*) e Valor (*Value*). Cada coluna categórica de entrada, ou seja, o *Embedding*, é projetada nessas matrizes para gerar seus vetores correspondentes de chave, consulta e valor. A partir desse cálculo, é ponderada a importância das partes de uma sequência em relação a outras partes, permitindo a captura de relações complexas pelo modelo.

No meio de *PLN*, a arquitetura discutida é bastante utilizada nos chamados *Large Language Models (LLMs)*, que são modelos de linguagem treinados em grandes quantidades de dados textuais, chegando a possuir bilhões de parâmetros. Nos últimos anos (2022-2023), vários modelos de geração de texto, como Chat GPT (*Generative Pre-trained Transformer*), emergiram na indústria como ferramentas disruptivas não utilizadas apenas pelos *"experts"* em *AM*, mas por

usuários das mais diversas áreas.

2.2.3.1 TabNet

A *TabNet* (ARIK; PFISTER (2021)) é uma arquitetura de *RN* feita especificamente para lidar com dados tabulares, combinando conceitos de *Feature Selection* e *MA*. Ela difere de abordagens usuais de Aprendizagem Profunda pelos seguintes aspectos:

1. **Atenção Sequencial:** *Essa abordagem usa atenção sequencial a fim de escolher quais features considerar em cada passo de decisão. Isso permite que o modelo saliente em features mais "importantes" no passo em questão, provendo uma aprendizagem mais eficiente, além de maior Interpretabilidade.*

2. **Feature Selection por Instância:** *As features são selecionadas individualmente para cada entrada, permitindo que o modelo seja adaptativo na escolha das features mais relevantes para cada instância, aumentando a Performance.*

3. **Processamento Não Linear:** *O modelo é capaz de capturar relações complexas e padrões nos dados através da incorporação de não-linearidade no processamento das features selecionadas.*

4. **Simulação de Ensembling:** *O modelo é capaz de simular a etapa de Ensemble através do uso de mais dimensões e passos na sua arquitetura.*

5. **Tomada de Decisões Interpretável:** *A abordagem provê uma tomada de decisão interpretável, através de uma visualização de máscaras de seleção. Essas máscaras mostram quais features foram selecionadas em cada passo de decisão, perfmitindo um melhor entendimento do comportamento do modelo.*

Essa técnica utiliza um *Mecanismo de Atenção Sequencial* a fim de selecionar um *"subset"* de *features* a cada passo de decisão. São empregados múltiplos blocos de decisão que focam nesse processamento do subgrupo de *"features"* de entrada.

A etapa de *Feature Selection*, por sua vez, é realizada por meio de *Máscaras* Aprendíveis, obtidas usando através do *Attention Transformer*. Isso permite à *TabNet* escolher em quais características focar em cada etapa de decisão. As *features* selecionadas são processadas pelos transformadores de características, que consistem em camadas compartilhadas e camadas dependentes da etapa de decisão. Essas camadas realizam transformações não-lineares nas características para melhorar a capacidade de aprendizagem. O processo de seleção de características e processamento de características é repetido por várias *Etapas de Decisão*, e as saídas de cada etapa são agregadas para tomar a decisão final.

Para o problema relacionado a esse trabalho, a existência de uma quantidade considerável de *features* no *Dataframe* em questão permite que a etapa comentada anteriormente constitua um aproveitamento extenso para melhorar a performance do modelo final.

Além disso, na arquitetura *TabNet*, diferentemente da maioria dos modelos tradicionais de *AM*, não há necessidade para *Feature Engineering*, permitindo a alimentação das "colunas" diretamente no modelo, que deve selecionar os melhores atributos automaticamente.

2.2.3.2 TabTransformer

O *TabTransformer* (HUANG et al. (2020)) é apresentado como uma arquitetura profunda de modelagem de dados capaz de transformar conjuntos de features categóricas em *Embeddings* contextuais robustos. Essa abordagem consiste em uma camada de *Embedding*, uma sequência de camadas de *Transformers* e um *MLP*, onde cada camada de *Transformer* inclui uma camada *Head* de *Auto-Atenção* seguida por uma camada *Position-Wise Feed-Forward*. Além disso, para aumentar a performance, a técnica incorpora um pré-treinamento de duas fases, além de um processo de refinamento para dados tabulares.

Sua principal inovação está na implementação de *Embedding* em uma camada à parte, transformando features categóricas em representações vetoriais complexas. Essas representações capturam informação contextual das features categóricas, o que é o ponto crucial de melhoria que o método propõe para garantir uma aperfeiçoamento em relação a modelos de *RNs* tradicionais. Por não produzir um *Embedding* em variáveis contínuas (não-categóricas), o modelo só constituí um método de melhoria quando apresentado a dados que possuam *features* categóricas.

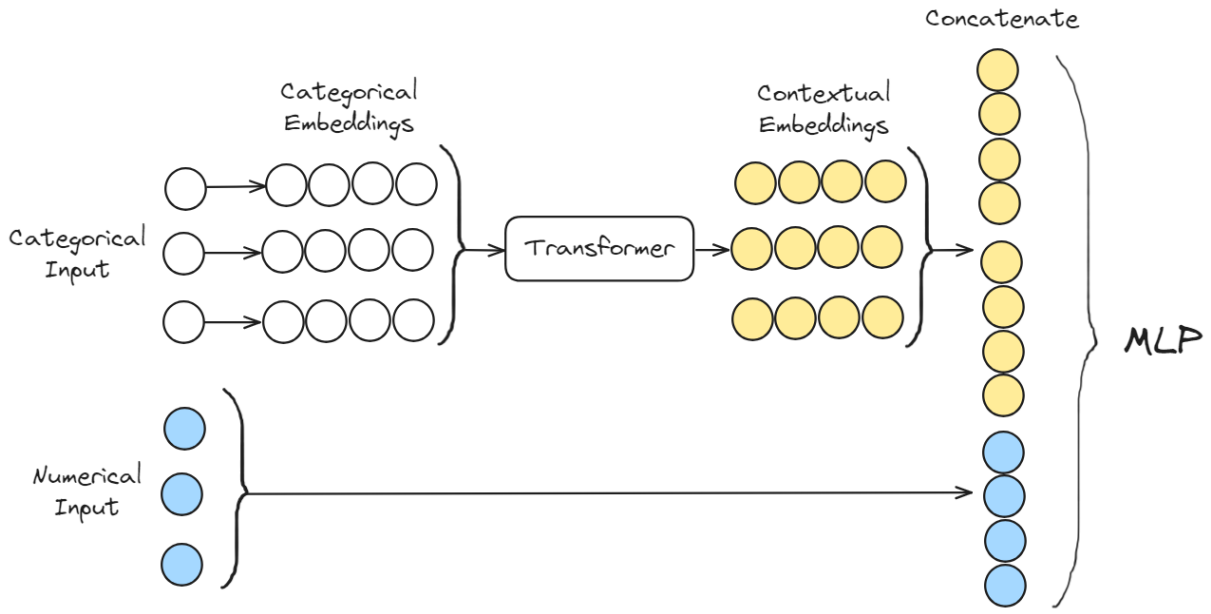

Figure 2.3: Arquitetura *TabTransformer*.

Como pode ser visto na imagem 2.3, a camada de *Embedding* de colunas atribui codificações paramétricas a cada característica categórica, enquanto as camadas *Transformer* processam esses *Embeddings* para gerar valores contextuais. O MLP combina esses valores contextuais com as características contínuas para fazer previsões.

2.2.3.3 FTTransformer

O *FT-Transformer* (GORISHNIY et al. (2023)) destaca sua sofisticação com uma simples adaptação da arquitetura de *Transformers* para o tratamento de dados tabulares. Essa abordagem utiliza uma esquematização mais simples que, diferente do modelo de *TabTransformer* comentado anteriormente, oferece também uma etapa de *Embedding* para os valores contínuos (não categóricos) da base de dados.

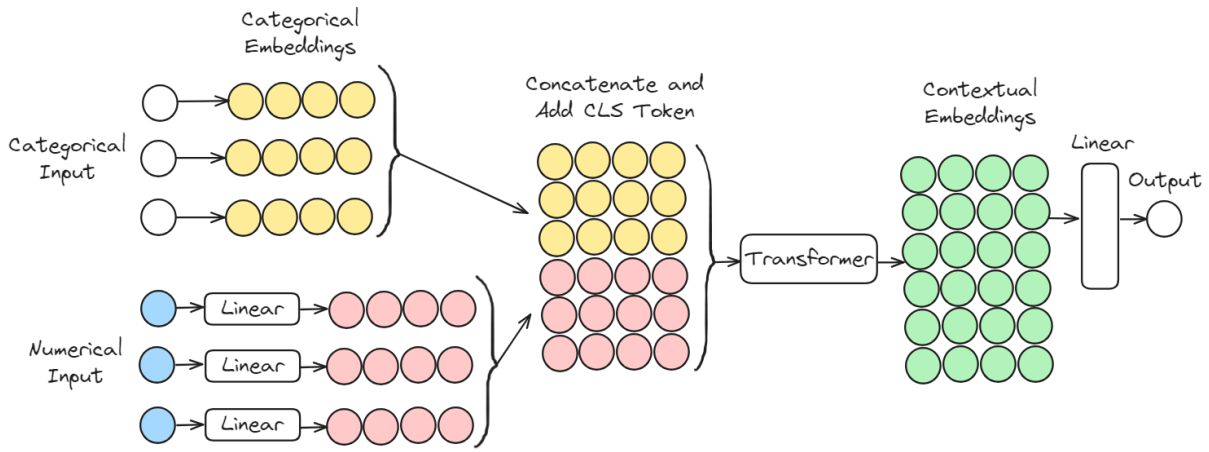

Figure 2.4: Arquitetura *FT-Transformer*.

Os *FT-Transformers* transformam *features* em *Embeddings* através de um *Feature Tokenizer*. Esses *Embeddings*, por sua vez, são processados pelo módulo *Transformer*, e a representação final do *Token [CLS]* é usada para fazer a predição. Esse *Token* é utilizado como representante da sequência inteira a fim de fazer a predição da classe de modo que, após o treinamento, a saída correspondente ao *Token [CLS]* é utilizada para determinar a classe da sequência de dados relativos.

O módulo *Feature Tokenizer* comentado é responsável unicamente por converter as características de entrada em *Embeddings*. Esse módulo utiliza viés de características e multiplicações elemento a elemento para calcular os *Embeddings* para cada característica. O módulo *Transformer* aplica múltiplas camadas de *Auto-Atenção* e Redes *Feed-Forward* aos *Embeddings*.

2.2.3.4 Exemplificação com Transformers

De modo a exemplificar o funcionamento das abordagens que utilizam a arquitetura *Transformers* a fim de classificar instâncias tabulares, considera-se o banco de dados utilizado nesse trabalho (4.2). Esse banco de dados simula uma gama de clientes de um banco financeiro. O *dataset* de Treino, especificamente, possui 389196 instâncias. Em relação a esse estudo, o *dataset* em questão possuí um único atributo binário (*IND_BOM_1_2*), o qual é interpretado como classe a ser prevista. A figura a seguir demonstra um segmento desses dados:

	UF_1	UF_2	UF_3	UF_4	UF_5	UF_6	UF_7	IDADE	SEXO_1	NIVEL_RELACIONAMENTO_CREDITO01	...	CEP4_7	CEP4_8	CEP4_9
0	1	1	1	0	0	0	0	0.135098	1	0.222222	...	0	0	1
1	1	0	1	0	0	1	0	0.273504	1	0.111111	...	0	1	0
2	1	0	1	0	0	1	0	0.281910	0	1.000000	...	1	1	0
3	1	1	1	0	0	0	0	0.225741	0	0.111111	...	1	1	0
4	1	1	0	0	0	1	0	0.480403	0	0.111111	...	1	1	1
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
389191	1	1	0	0	0	0	1	0.787827	1	0.111111	...	0	1	1
389192	1	0	1	0	0	0	1	0.470010	1	0.111111	...	1	0	1
389193	0	1	0	1	0	0	1	0.436048	0	0.000000	...	1	0	0
389194	1	0	1	0	1	0	0	0.677875	0	0.111111	...	0	1	0
389195	1	1	0	0	1	0	0	0.690540	0	0.111111	...	1	1	0

389196 rows × 245 columns

Figure 2.5: Partição do *Dataframe* de Treino.

Empiricamente, constatou-se que algumas colunas possuiam dados categóricos, enquanto outras colunas apresentavam dados numéricos contínuos. Como cada instância catalogada continha tais atributos, os modelos baseados em *Transformers* poderiam se beneficiar de tal circunstância através da implementação de uma camada de *Embedding*. Em conformidade, os dados foram segmentados de acordo com sua natureza (Categórica ou Numérica) e processados pela camada de *Embedding*. Vale ressaltar que, como comentado nas Seções *TabTransformer* (2.2.3.2) e *FT-Transformer* (2.2.3.3), a etapa em questão pode variar de acordo com o modelo selecionado para análise. Contudo, com a descrição do *Pipeline* de apenas um dos modelos, pode-se abstrair o funcionamento dos demais. Posteriormente, houve uma concatenação dos dados já codificados (considerando manipulações auxiliares necessárias para a alimentação em um modelo de *Transformers*, como o acréscimo do *Token [CLS]*). À exemplo do modelo *FT-Transformer*, tem-se:

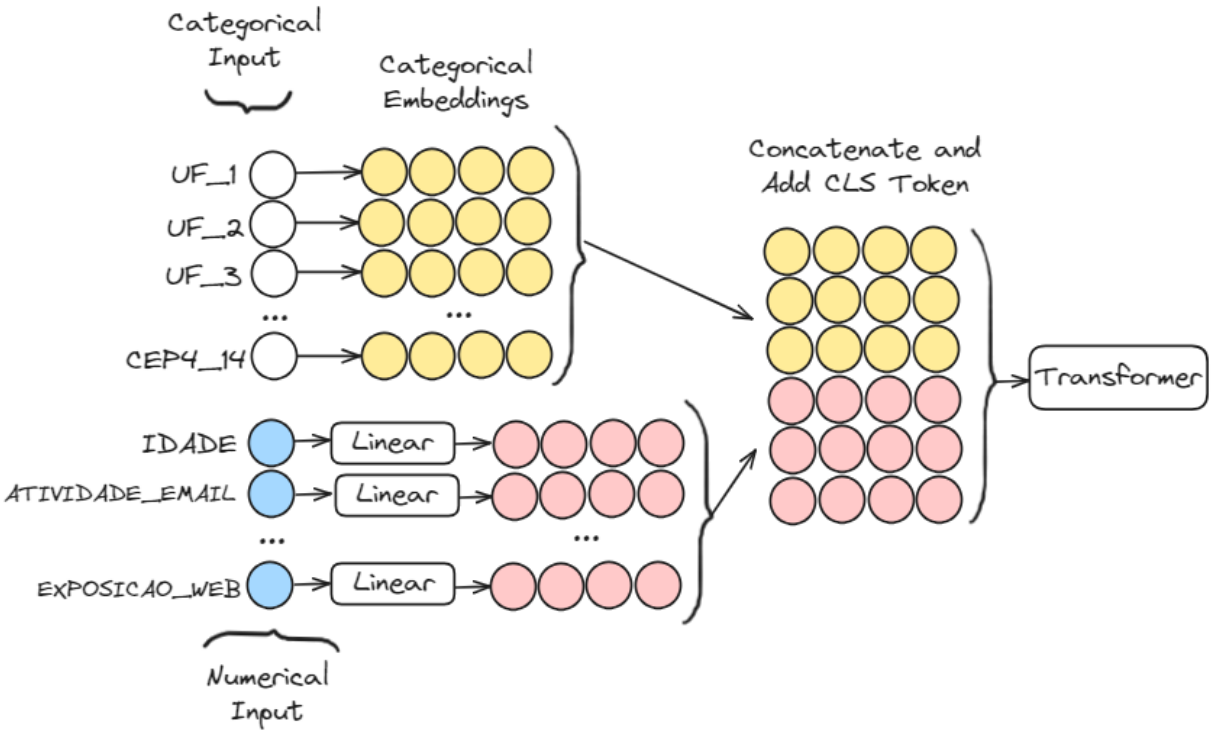

Figure 2.6: Exemplo do *Pipeline* de alimentação dos Dados na arquitetura *Transformer* para o modelo FT-Transformer.

Com a transformação de todas as *features* em *Embeddings* e a aplicação de um conjunto de camadas de *Transformers*, a representação final dos dados, juntamente ao *Token [CLS]*, são utilizados para realizar a previsão da classe considerada.

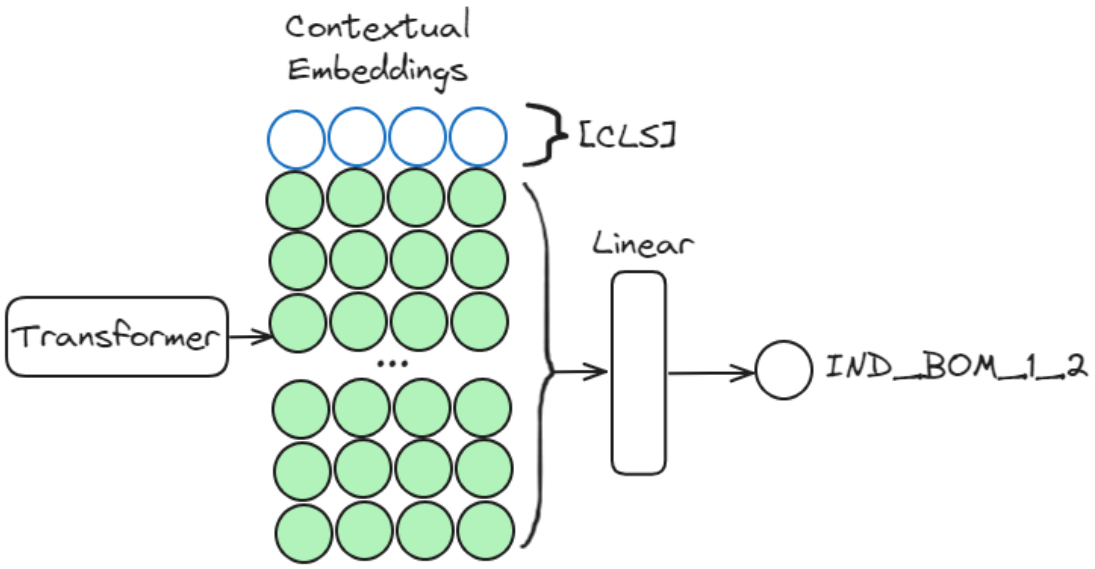

Figure 2.7: Exemplo representativo da etapa final de inferência no modelo FT-Transformer.

2.3 Métricas de Avaliação

As *Métricas de Avaliação* desempenham o papel de medidores de desempenho do modelo avaliado sobre uma tarefa específica. Tais métricas permitem aos desenvolvedores compreenderem se determinado modelo atende aos requisitos de qualidade ou não. Além disso, elas não se restringem apenas à avaliação de qualidade dos modelos, mas, como é feito nesse trabalho, também são utilizadas para o ajuste de hiperparâmetros dos modelos, servindo como critérios de otimização.

No quadro de tomadas de decisão, as escolhas feitas a partir dos resultados obtidos por um modelo de IA dependem da métricas obtidas. Suponha um modelo de diagnóstico médico para casos de câncer, por exemplo: Em virtude dos riscos, é muito mais imprescindível o modelo não apresentar *Falsos Negativos* (pessoas com câncer erroneamente diagnosticadas) que qualquer outra avaliação, visto que esse erro constituiria na liberação do paciente, levando ao quadro de agravamento do câncer.

Em virtude dos pontos levantados, é perceptível a importância da captação da maior quantidade possível de métricas relativas à performance dos modelos analisados. As medidas utilizadas nesse trabalho foram:

1. **Acurácia:** *Métrica baseada na proporção de previsões corretas em relação ao número de total de previsões realizadas. Essa métrica, apesar de ser bastance simples (conceitualmente), apresenta um elevado potencial enganoso, visto que é sensível a dados desbalanceados.*

2. **Precisão:** *Mede a proporção de Verdadeiros Positivos (TP) em relação a todos os Positivos previstos pelo modelo (TP + FP). Possui maior utilidade quando o interesse majoritário está na minimização Falsos Positivos.*

3. ***Recall*/Sensibilidade:** *Baseada na proporção de Verdadeiros Positivos em relação ao número total de Verdadeiros Positivos e Falsos Negativos (TP + FN), sendo mais útil quando se quer identificar os casos positivos, mesmo que resulte em falsos positivos.*

4. **F1-*Score*:** *Métrica que combina Recall/Sensibilidade e Precisão em uma única abordagem, calculando a Média Harmônica de ambas (um Trade-Off das duas métricas).*

5. **Área sob a Curva ROC - ROC AUC:** *Métrica extensivamente utilizada em casos de classificação binária, avaliando a capacidade do modelo de distinguir entre Classes Positivas e Negativas. É um gráfico representativo da taxa de Verdadeiros Positivos (TPR) em função de Falsos Positivos (FPR).*

6. **KS - Kolmogorov-Smirnov:** *Utilizada principalmente para classificação binária, bastante utilizada no contexto de econometria. Essa métrica mede a distância máxima entre as curvas de distribuição cumulativa (CDF) das classes positivas e negativas.*

A *Métrica KS*, priorizada nesse trabalho, avalia a similaridade entre duas distribuições de probabilidade e é bastante utilizada para o estudo de econometria, como é o exemplo do banco de dados estudado nesse trabalho. A abordagem de cálculo utilizada nessa pesquisa, denominada *"KS Two Sample Test"*, é mais comumente utilizada para averiguar se duas amostras advêm da mesma distribuição.

$$KS = max|F(x) - G(x)|$$

F(x)	Função de Distribuição Cumulativa Empírica (ECDF)
G(x)	Função de Distribuição Cumulativa de Referência (RCDF)
max	Distância Máxima

O cálculo dessa métrica é descrito pela *Máxima Distância* entre duas *funções de distribuição cumulativa* (CDF), quando comparadas. No caso do banco de dados analisado, as distribuições comparadas são referentes à previsão binária probabilística fornecida por cada modelo em relação à classe real de cada instância. Uma maior valoração da métrica estudada infere uma maior capacidade de o modelo conseguir separar/dissimilar as distribuições, e é o que pretende ser maximizado em cada uma das abordagens. Vale constar que o estado-da-arte para a classificação binária de instâncias no banco de dados trabalhado (*gbdt*) apresenta uma valoração para a estatística KS de aproximadamente "0.32", sendo esse o valor que o estudo pretende superar (ou se aproximar) com a implementação de novas abordagens.

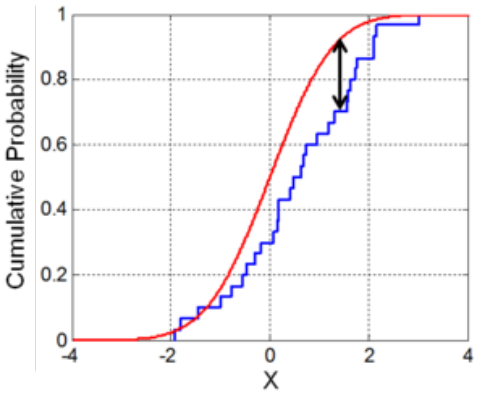

Figure 2.8: Representação arbitrária de uma Estatística-KS. A linha Azul representa a CDF emírica, enquanto a linha Vermelha representa a CDF de referência. A linha preta representa o Valor KS.

3

Metodologia

3.1 Introdução

Para esse trabalho, o objetivo central de comparação de performance para cada um dos modelos estudados apenas pode ser constatado mediante uma metodologia "justa" e uniforme. Nesse quesito, a fim de avaliar o comportamento dos modelos, foram programadas implementações individuais para cada um deles, de modo que, com uma análise posterior, pudessem ser feitas inferências quanto ao potencial das novas abordagens de *Transformers* e *MA*. Tal plano de desenvolvimento é bastante comum na área de Ciência de Dados, em que a escolha de métodos estatísticos e matemáticos, juntamente a algoritmos de AM, podem inferir a capacidade de cada abordagem em resolver um problema real, como é frequentemente visto em problemas de *Classificação* e *Regressão*. Os resultados obtidos, por sua vez, tendem a abstrair uma avaliação de desempenho sobre a análise realizada.

3.2 Objetivo e Dados Analisados

3.2.1 Tratamento dos Dados

Em virtude do objetivo da pesquisa ser direcionado à análise comparativa de modelos, buscando não beneficiar nenhuma das abordagens quanto ao *pré-processamento* dos dados, o tratamento realizado antes da implementação dos modelos foi realizado de forma a instituir um exame justo para cada método. Os dados de treinamento, portanto, foram balanceados, visto que abordagens como RNA poderiam apresentar um viés à classe analisada mais frequente, enquanto outros modelos, como GBDT, são mais tolerantes a tal desbalanceamento.

Além disso, os dados em questão também foram *Normalizados* através do uso de técnicas como *Standardization* e *MinMaxScaling*, que garantem uma melhoria na eficiência do treinamento e, para alguns casos, chega a possibilitar o treinamento, como na abordagem em RNA, que pode ser afetada por dados *não normalizados* apresentando problemas de "Exploding Gradient" ou "Vanishing Gradient".

Tem-se em vista que, para esse trabalho, um único banco de dados previamente segmentado em *Teste* e *Treino* foi considerado. Em virtude do objetivo de extrair o máximo potencial de cada um dos modelos e, posteriormente, comparar os resultados obtidos, os dados de Treinamento foram segmentados em *Conjunto de Treino* e *Validação*, de modo que a proporção final de instâncias para Treino, Teste e Validação, constituísse uma proporção de aproximadamente 50%, 25% e 25%, respectivamente. A inserção de um *Conjunto de Validação* na análise permite que, antes de os resultados finais de cada modelo possam ser obtidos e comparados entre si, cada um dos modelos possa ter seus parâmetros otimizados da melhor forma possível a fim de apresentarem uma métrica final adequada. Além disso, a inferência de cada modelo sobre o conjunto de validação provê uma estimativa intermediária do desempenho de cada abordagem.

Por fim, como comentado na Seção *TabTransformer*(2.2.3.2), alguns dos modelos da abordagem em *RNs* apenas se beneficiam de suas arquiteturas na análise e treinamento de dados quando apresentados a dados categóricos. Esse quesito, para a abordagem em *GBDT*, foi analisado tanto com a conversão de atributos numéricos para *features* categóricas quanto sem tal transformação, de forma que não houveram alterações significativas nos resultados obtidos ao final do estudo. Dessarte, foi preferido transformar todos os atributos contendo menos de *101* valores únicos para atributos categóricos, através de técnicas de *OneHotEncoding* e *LabelEncoding*. O valor do *Threshold* que define quantos valores únicos são necessários para que uma *feature* seja considerada categórica ou não, por sua vez, foi selecionado empiricamente.

3.3 Otimização e Métricas de Avaliação

Várias são as formas de refinar os hiper-parâmetros de cada modelo analisado a fim de se extrair o máximo potencial quanto a previsões realizadas sobre um banco de dados arbitrário. Esse refinamento nada mais é que a variação de hiper-parâmetros de um modelo de forma exaustiva com o intuito de otimizar uma métrica previamente selecionada para estudo. Como forma de acelerar o processo de busca pelos melhores hiper-parâmetros, várias são as bibliotecas que possibilitam uma otimização automatizada, como *GridSearch* e *Optuna*.

Para esse trabalho, especificamente, a biblioteca de otimização de parâmetros escolhida foi o *Optuna*, que frequentemente apresenta resultados melhores que as demais ferramentas utilizadas no mesmo contexto, como *Grid-Search*. Isso acontece porque o Optuna utiliza uma estratégia de otimização inteligente baseada em *Estatística Bayseiana* e em um método denominado *Tree-Structured Parzen Estimator (TPE)*, o que permite ao estudo priorizar suas variações de hiper-parâmetros em área "promissoras" do espaço.

A métrica selecionada para ser otimizada perante a variação dos hiper-parâmetros em cada modelo, por sua vez, foi a estatística *KS (Kolmogorov-Smirnov)*.

Através do uso da biblioteca *Scipy* da linguagem de programação *Python*, calculou-se a *Estatística KS* para as duas amostras consideradas. Os parâmetros desse método foram 2:

1. Probabilidades correspondentes aos *"não-eventos"*.

2. Probabilidades correspondentes aos *"eventos"*.

Além disso, apesar de o KS ser a métrica principal de avaliação/otimização, outras métricas também são analisadas nesse trabalho, sendo esssas: *Área Sob a Curva ROC (ROC AUC), Acurácia, Precisão, Recall/Sensibilidade e F1-Score*. Além disso, o tempo destinado às otimizações de cada modelo também foi catalogado.

Adendo: Vale ressaltar que, para o cálculo da *Estatística KS*, o valor fornecido deve ser uma entrada probabilística referente à classe prevista pelo modelo. Para isso, o método *"predict proba"* incluso nos modelos contidos na biblioteca *sklearn* (que foi utilizada na implementação do código) é bastante relevante, visto que fornece a confiança do modelo em suas previsões binárias (geralmente através do uso de uma função logística).

Para a abordagem em RNA, todavia, essa função não possui implementação nativa. Felizmente, com a inserção de uma função *Sigmoide* na saída da rede, consegue-se obter uma abordagem "probabilística" capaz de inferir uma *Estatística KS* contundente/adequada.

3.3.1 Especificações Técnicas

Especificamente para esse trabalho, foi utilizada uma máquina com as seguintes especificações básicas: *NVIDIA GeForce RTX 2060 Mobile, 32.768 MB RAM*. Essas configurações inferem uma capacidade relativamente competitiva no quesito de tratamento de dados e realização de operações matemáticas de forma paralela. A execução de código, por sua vez, foi integralmente realizada na ferramenta JupyterLab, que apresenta ambiente interativo de código aberto que facilita a criação e compartilhamento de documentos interativos contendo código, visualizações e texto explicativo. Dessarte, todas as operações foram devidamente explicadas ao longo do corpo do documento gerado por essa ferramenta.

3.4 Variações dos Algoritmos Estudados

Nessa seção, é apresentada uma análise superficial da variação de hiper-parâmetros de cada um dos modelos considerados, de modo a contextualizar as decisões tomadas.

Para cada cada um dos modelos, a biblioteca *Optuna* instancia um objeto chamado *"Study"*, responsável pelo gerenciamento do processo de otimização, registro das tentativas e coordenação da busca pelos melhores *hiper-parâmetros*. Nesse estudo, as variáveis "estudadas" foram definidas de forma explícita a fim de guiar o processo de otimização. Além disso, também foi criada uma função *"Objective"* responsável por avaliar o desempenho do modelo estudado com base na métrica previamente selecionada para embasamento.

Relativo à quantidade de variações que foram experimentadas em cada estudo de modelo, a biblioteca *Optuna* permitiu a inferência de uma quantidade arbitrária de *"trials"*. A variável

"trials" é referente ao conjunto de diferentes hiper-parâmetros que foram testados em cada treinamento a fim de se obter uma perfomance otimizada, de modo que uma valoração maior para essa variável constitui em uma quantidade maior de chances de serem encontrados os melhores hiper-parâmetros para os modelos e, consequentemente, uma melhor performance. Vale salientar que um número maior de *"trials"* também requisita um maior intervalo de tempo, visto que mais casos de estudo são considerados, o que deve ser levado em conta quando se quer abstrair a melhor configuração do modelo.

3.4.1 Gradient Boosted Decision Trees (GBDT) e seus Hiperparâmetros

3.4.1.1 Random Forest

Para a abordagem de *RF*, cada Árvore é treinada em uma amostra aleatória do conjunto de dados e produz uma previsão. Nesse modelo os parâmetros variados foram: *n_estimators, criterion, max_depth,min_samples_split, min_samples_leaf, max_features* e *class_weight*.

3.4.1.2 Gradient Boosting

Para a abordagem de *GB*, o modelo funciona em etapas sequenciais, corrigindo os erros dos modelos anterirores. Nesse modelo, os parâmetros variados foram: *loss, learning_rate, n_estimators, max_depth, min_samples_split, min_samples_leaf, max_features* e *subsample*.

3.4.1.3 XGBoost

Para a abordagem de *XGB*, o modelo possui uma implementação otimizada de *GB* que acelera o treinamento e possui nuâncias de decisão no algoritmo. Nesse modelo, os parâmetros variados foram: *objective, eval_metric, booster, eta, max_depth, subsample, colsample_bytree, min_child_weight, gamma* e *n_estimators*.

3.4.1.4 Light Gradient Boosting Machine

Para a abordagem *LGBM*, uma estratégia *"Leaf-Wise"* é utilizada de modo a reduzir a perda e garantir maior eficiência durante o trainemnto, além de agrupar valores e recursos em histogramas para reduzir a memória durante o treinamento com o uso de uma técnica de nome "Histogram-Based Learning". Nesse modelo, os parâmetros variados foram: *objective, metric, boosting_type, learning_rate, max_depth, subsample, colsample_bytree, min_child_samples, num_leaves, reg_alpha* e *reg_lambda*.

3.4.1.5 Ensemble de Gradient Boosting Decision Trees

Para finalizar a análise a partir da abordagem de modelos tradicionais de *GBDT*, foi criado um *Ensemble* utilizando os modelos de *RF, GB, XGB e LGBM* através de um *Voting*

Classifier, que combina a previsões dos modelos de forma individual a fim de tomar uma decisão final baseada nos pontos fortes dos modelos considerados.

A técnica de *Voting Classifier*, especificamente, foi implementada com uma abordagem de *"Soft Voting Classifier"*, de modo que as previsões dos modelos são ponderadas de acordo com a confiança de cada modelo em sua previsão. Desse modo, os modelos que são mais confiantes em suas previsões possuem um maior peso, beneficiando o resultado do modelo final gerado.

3.4.2 Implementação em Redes Neurais Tradicionais

Antes da exploração de eficácia provida por modelos baseados em *Transformers* e *Mecanismos de Atenção*, também foi considerado válido constatar como modelos de *Redes Neurais Tradicionais* usualmente utilizadas em tarefas simples de Classificação e Regressão performam em comparação com as abordagens já apresentadas.

3.4.2.1 Perceptron Multicamadas

Para abordagem em MLP, o modelo constitui uma RNA composta por múltiplas camadas de *Neurônios*, cada uma conectada com a camada seguinte. De forma direta, essa técnica pode ser considerada como um aproximador de função universal, o que pode ser aproveitado pela complexidade do banco de dados. De modo a sofisticar o funcionamento e complexidade providos por essa abordagem, a RN implementada foi personalizada empiricamente de modo que, após uma série consecutiva de análises, foi obtido um resultado adequado para a análise feita. Para isso, foram inseridas instâncias de *Batch Normalization*, *DropOut* e de diferentes *Funções de Ativação*. Além disso, com o uso do *Optuna*, os parâmetros variados foram: *hidden_size, learning_rate* e *dropout_rate*.

3.4.3 Implementação de Modelos Baseados em Transformers

A partir dos modelos considerados a seguir, já é realizada uma segmentação prévia entre as *"features"* categórica e numérias, de forma que as abordagens podem abstrair novas relações de cada tipo de dados fornecido. Todas as implementações utilizadas na abordagem da arquitetura *Transformers*, foram realizadas com base na biblioteca de Aprendizagem Profunda *"PyTorch"*.

3.4.3.1 TabNet

Baseando-se na implementação fornecida em AI (2019 https://github.com/dreamquark-ai/tabnet) *TabNet* para Pytorch, o TabNet é essencialmente um algoritmo de Aprendizagem de Máquina que combina RNA com MA e técnicas utilizadas em Feature Selection. Vale ressaltar que, nesse modelo, existem dois conjuntos de hiperparâmetros utilizados na configuração: um conjunto relacionado ao número de características dos dados e suas dimensões, e outro conjunto

relacionado à arquitetura do próprio modelo e treinamento. Além disso, para a implementação nessa abordagem, os dados foram alimentados no modelo de forma integral, destacando explicitamente apenas os *"índices"* dos dados de natureza categórica e numérica/contínua, de modo que o modelo pudesse aproveitar tais atributos. Com o uso do *Optuna*, os parâmetros variados foram: *n_d, n_a, gamma, lambda_sparse, n_independent, n_shared* e *cat_emb_dim*.

3.4.3.2 TabTransformer

Baseando-se na implementação em LUCIDRAINS (2019 https://github.com/lucidrains/tab-transformer-pytorch) *TabTransformer*, as vantagens providas pela arquitetura de transformers são aproveitadas de forma a permitir que o modelo aprenda padrões complexos e dependências entre as colunas categóricas. Diferentemente do modelo *TabNet*, os dados foram explicitamente segmentados em dois *datasets*: Categórico e Numérico. Só então, os dados foram "alimentados" no modelo, para que o mesmo pudesse se beneficiar em sua arquitetura. Com o uso do *Optuna*, os parâmetros variados foram: *dim, depth, heads, attn_dropout, ff_dropout, mlp_hidden_mults_0, mlp_hidden_mults_1* e *learning_rate*.

3.4.3.3 FT-Transformer

Para a abordagem LUCIDRAINS (2019 https://github.com/lucidrains/tab-transformer-pytorch) *FT-Transformer* implementada em *PyTorch*, além de ser aproveitada a arquitetura transformers de forma a permitir a aprendizagem de relaçoes entre as *"features"* categóricas, as colunas com valores numéricos contínuos também sofrem *"Embedding"* e são aproveitadas. Favoravelmente, como os dados já haviam sido segmentados entre categóricos e numéricos em uma etapa anterior, essa implementação não apresentou um desafio. Com o uso do *Optuna*, os parâmetros variados foram: *dim, depth, heads, attn_dropout, ff_dropout* e *learning_rate*.

3.5 Avaliação dos Modelos

Para cada modelo levantado nesse trabalho, foi implementada uma visualização personalizada de performance sobre os dados de teste contendo o gráfico da *Estatística KS*, bem como uma matriz de confusão da qual se é possível abstrair as métricas de *Acurácia, Recall(Sensibilidade), Precisão*, e *F1-Score*. Foi implementada também a abstração da métrica de Área Sob a Curva (AUC-ROC).

4

Experimentos e Resultados

4.1 Introdução

Essa *Seção* contextualiza os experimentos mencionados na metodolgogia, de forma a apresentar uma análise concreta das performances de cada um dos modelos estudados e, posteriormente, apresentar uma prerrogativa de uso para cada abordagem. Vale ressaltar que os experimentos foram realizados para um número de 7 repetições *(trials)*, de acordo com a biblioteca *Optuna*. Isso significa que cada abordagem possuiu 7 tentativas para encontrar os *hiper-parâmetros* que otimizassem sua performance quanto às previsões feitas sobre os dados de validação. Os dados de teste, por sua vez, não apresentaram influência alguma sobre o treinamento, sendo utilizados unicamente a fim de mostrar um resultado/desempenho não enviesado dos modelos trabalhados. O código pode ser encontrado em Baseando-se na implementação em GVASCONS (2023 https://github.com/Gvascons/Tabular-Transformer-Thesis) *Thesis Code*.

4.2 Descrição dos Dados

De modo a apresentar um caso de estudo que possua complexidade de um problema real, o dataset de treino utilizado para análise apresenta um teor robusto de 389.196 linhas (instâncias) e 245 colunas (atributos). Esse dataset, especificamente, representa um conjunto de clientes de um banco financeiro, onde cada cliente é uma instância e possui 245 *features* catalogadas. Uma das *features* em questão é denominada *'IND_BOM_1_2'*, e é um atributo binário que representa o caso de um cliente receber ou não uma concessão de crédito. A partir disso, pode-se abstrair que o problema apresentado é um problema de *Classificação Binária*, o qual é bastante utilizado em cenários de "Lending"(Empréstimo) em Ciência de Dados.

Além de possuir o *dataset* de Treino mencionado, também foi fornecido um *dataset* de Teste contendo 129.733 clientes e 245 atributos para que, ao final na análise, implementação e treinamento dos modelos, pudesse ser inferida a performance de cada uma das abordagens sobre o banco de dados em questão.

4.3 Resultados e Desempenho

Quanto aos resultados individuais obtidos por cada modelo, vale ressaltar que algumas abordagens, como *XGB* e *TabTransformer*, permitem uma paralelização de cálculos que terminam por acelerar o processo de treinamento e inferência. Dessarte, é natural que o tempo necessário para o obtenção dos melhores *hiper-parâmetros* para cada abordagem esteja fortemente relacionado com o dispositivo em que tal manipulação paramétrica seja realizada. Para cada uma das abordagens comentadas no trabalho, os seguintes resultados individuais relativos ao *dataset* de Teste foram obtidos:

1. **Resultados para Random Forest**

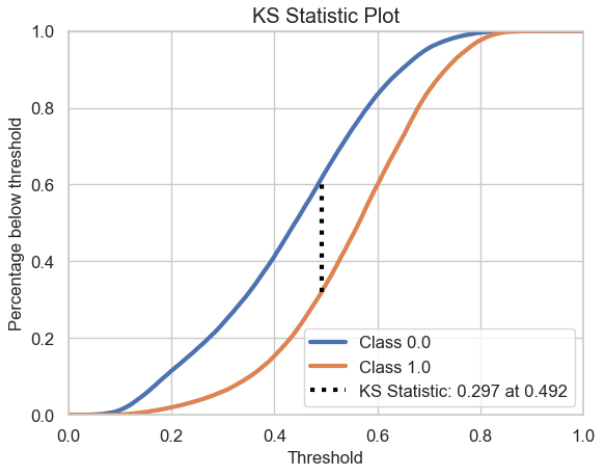

Figure 4.1: Gráfico KS do modelo Random Forest relativo ao Conjunto de Teste.

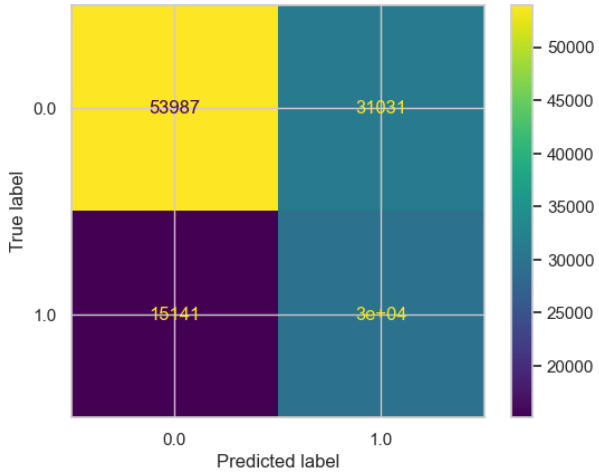

Figure 4.2: Matriz de Confusão do modelo Random Forest relativo ao Conjunto de Teste.

Para a abordagem em *RF*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6441; ***Recall***: 0.6613; ***Precisão***: 0.4878; ***F1***: 0.5615; ***AUROC***: 0.7043; ***KS***: 0.2966. O tempo

total destinado à obtenção dos melhores hiper-parâmetros foi de: *16h12min17seg*.

2. **Resultados para Gradient Boosting**

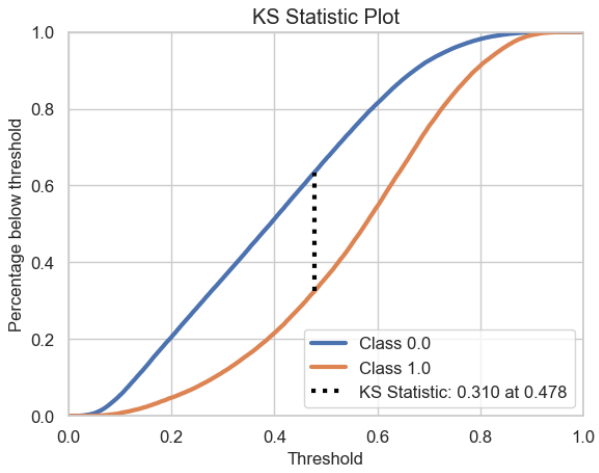

Figure 4.3: Gráfico KS do modelo Gradient Boosting relativo ao Conjunto de Teste.

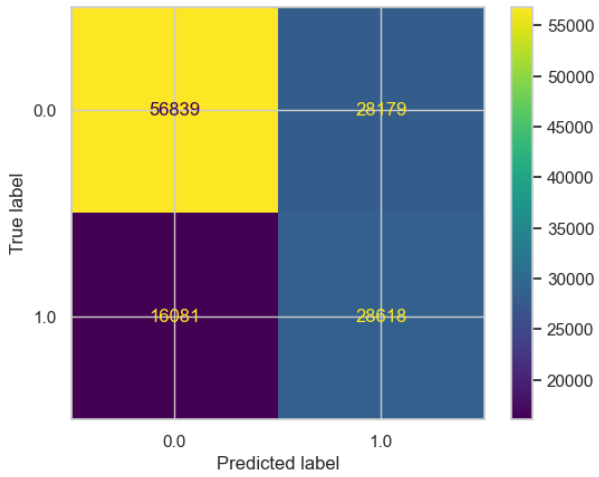

Figure 4.4: Matriz de Confusão do modelo Gradient Boosting relativo ao Conjunto de Teste.

Para a abordagem em *GB*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6588; ***Recall***: 0.6402; ***Precisão***: 0.5039; ***F1***: 0.5639; ***AUROC***: 0.7144; ***KS***: 0.3100. O tempo total destinado à obtenção dos melhores *hiper-parâmetros* foi de: *14h10min52seg*.

3. **Resultados para XGBoost**

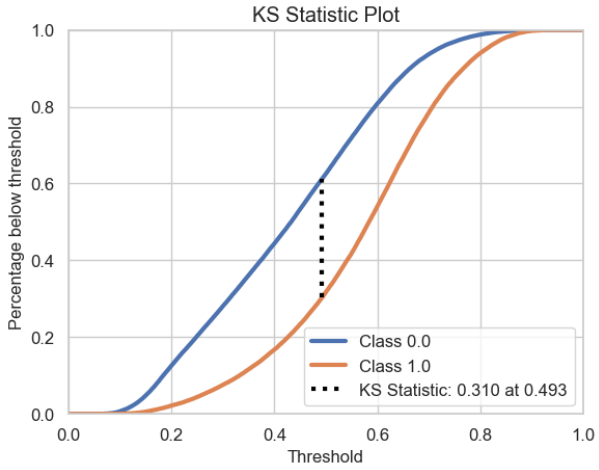

Figure 4.5: Gráfico KS do modelo XGBoost relativo ao Conjunto de Teste.

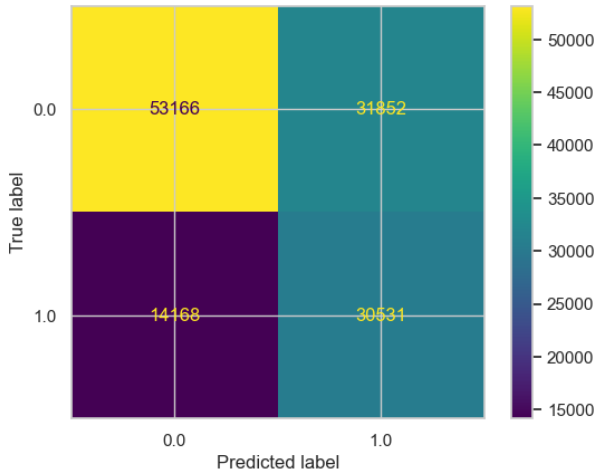

Figure 4.6: Matriz de Confusão do modelo XGBoost relativo ao Conjunto de Teste.

Para a abordagem em *XGB*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6452; ***Recall***: 0.6830; ***Precisão***: 0.4894; ***F1***: 0.5702; ***AUROC***: 0.7131; ***KS***: 0.3098. O tempo total destinado à obtenção dos melhores hiper-parâmetros foi de: *8min42seg*.

4. **Resultados para LGBM**

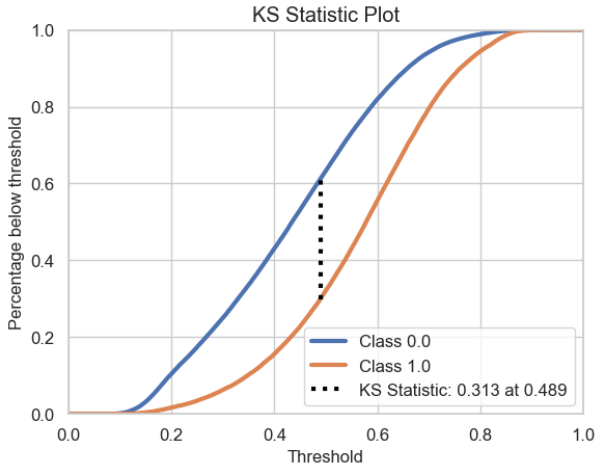

Figure 4.7: Gráfico KS do modelo LGBM relativo ao Conjunto de Teste.

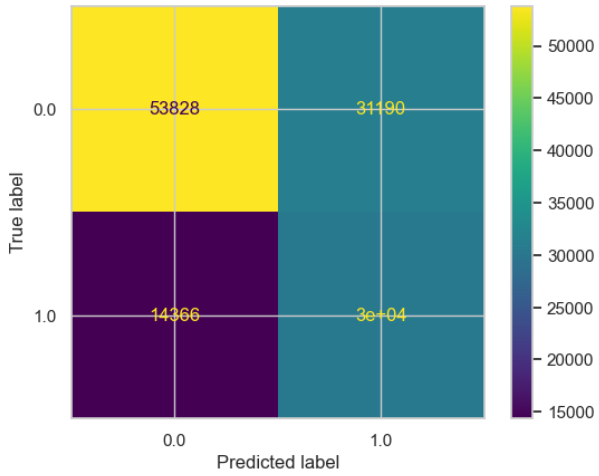

Figure 4.8: Matriz de Confusão do modelo LGBM relativo ao Conjunto de Teste.

Para a abordagem em *LGBM*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6488; ***Recall***: 0.6786; ***Precisão***: 0.4930; ***F1***: 0.5711; ***AUROC***: 0.7151; ***KS***: 0.3132. O tempo total destinado à obtenção dos melhores hiper-parâmetros foi de: *9min2seg*.

5. **Ensemble**

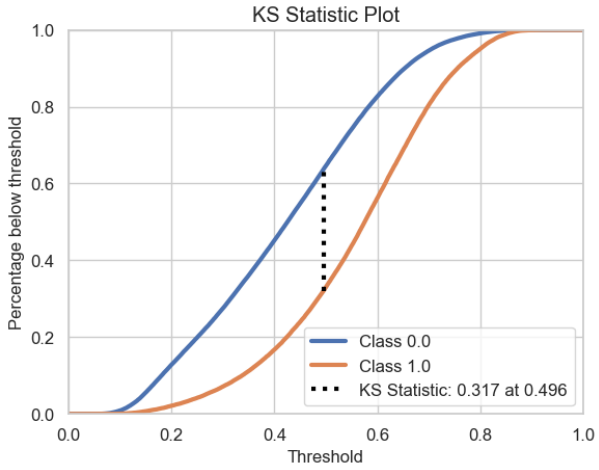

Figure 4.9: Gráfico KS do modelo Ensemble relativo ao Conjunto de Teste.

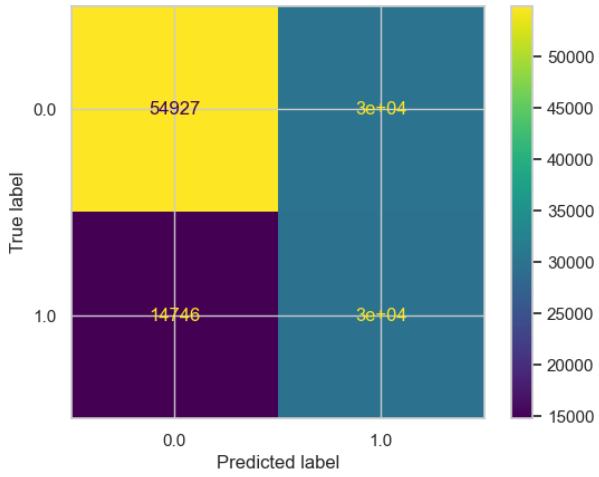

Figure 4.10: Matriz de Confusão do modelo Ensemble relativo ao Conjunto de Teste.

Para a abordagem em *Ensemble*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6543; ***Recall***: 0.6701; ***Precisão***: 0.4989; ***F1***: 0.5719; ***AUROC***: 0.7178; ***KS***: 0.3169. O tempo total destinado à obtenção dos melhores hiper-parâmetros foi de: *35h14min49seg*.

Vale ressaltar que, para a abordagem em *Ensemble*, o tempo total obtido contemplou 2 fatores: O tempo destinado para a etapa de *Voting Classifier*, em que houve a combinação dos modelos de AM considerados no trabalho através da comparação de média das probabilidades previstas, bem como o tempo de obtenção individual dos melhores modelos para cada uma das abordagens estudadas.

6. **Resultados para MLP**

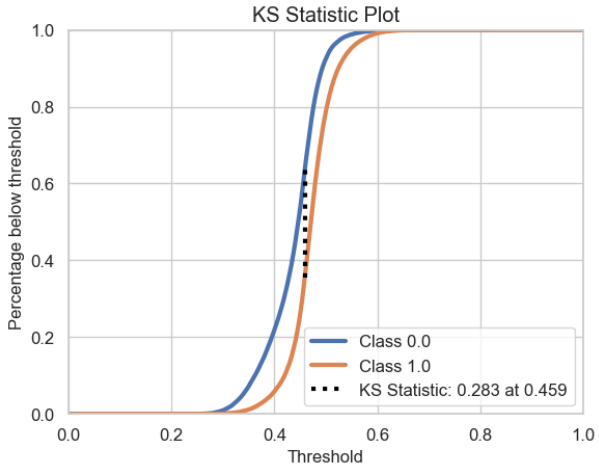

Figure 4.11: Gráfico KS do modelo MLP relativo ao Conjunto de Teste.

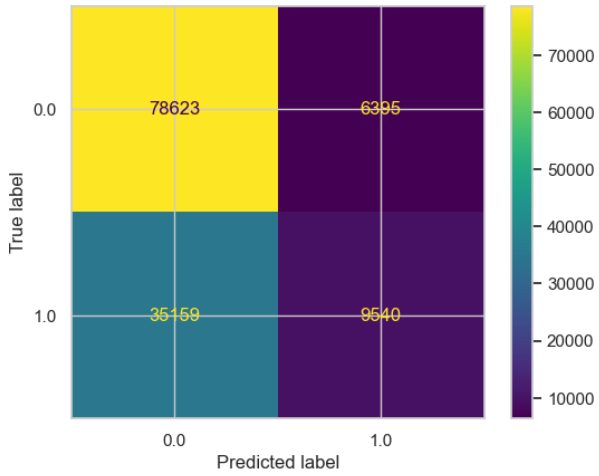

Figure 4.12: Matriz de Confusão do modelo MLP relativo ao Conjunto de Teste.

Para a abordagem em *MLP*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6797; ***Recall***: 0.2134; ***Precisão***: 0.5987; ***F1***: 0.3147; ***AUROC***: 0.6940; ***KS***: 0.2830. O tempo total destinado à obtenção dos melhores hiper-parâmetros foi de: *6h26min13seg*.

7. **Resultados para TabNet**

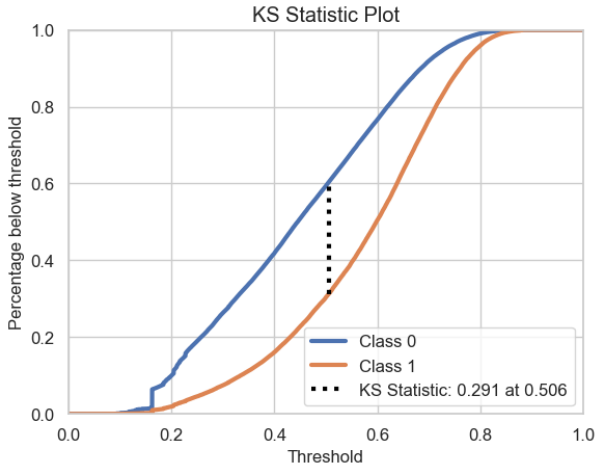

Figure 4.13: Gráfico KS do modelo *TabNet* relativo ao Conjunto de Teste.

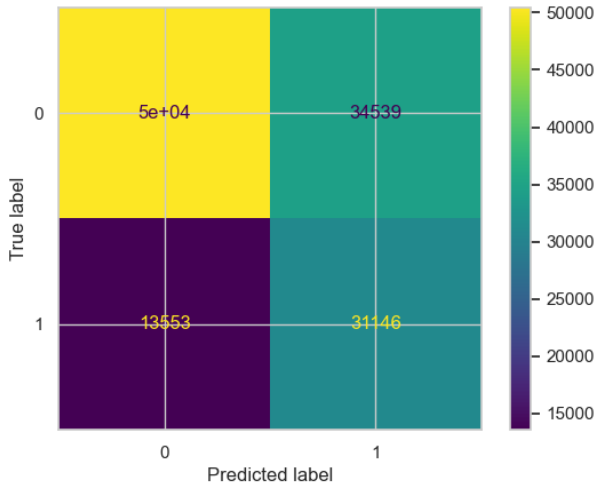

Figure 4.14: Matriz de Confusão do modelo *TabNet* relativo ao Conjunto de Teste.

Para a abordagem em *TabNet*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6293; ***Recall***: 0.6968; ***Precisão***: 0.4742; ***F1***: 0.5643; ***AUROC***: 0.6998; ***KS***: 0.2915. O tempo total destinado à obtenção dos melhores hiper-parâmetros foi de: *20h31min15seg*.

8. **Resultados para TabTransformer**

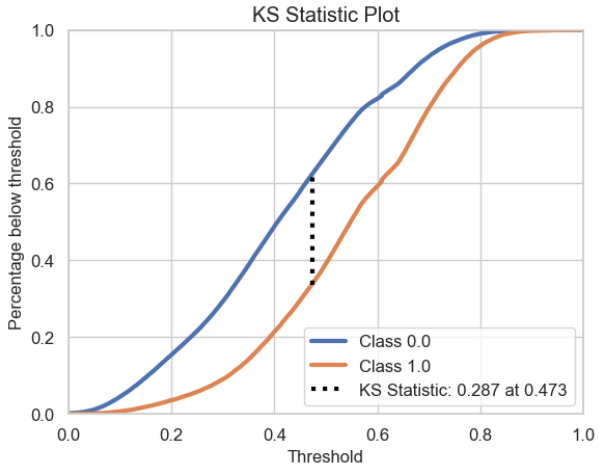

Figure 4.15: Gráfico KS do modelo *TabTransformer* relativo ao Conjunto de Teste.

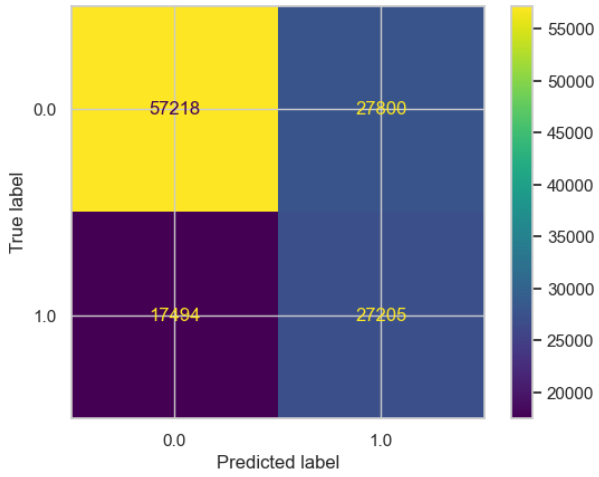

Figure 4.16: Matriz de Confusão do modelo *TabTransformer* relativo ao Conjunto de Teste.

Para a abordagem em *TabTransformer*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6508; ***Recall***: 0.6086; ***Precisão***: 0.4946; ***F1***: 0.5457; ***AUROC***: 0.6967; ***KS***: 0.2866. O tempo total destinado à obtenção dos melhores hiper-parâmetros foi de: *18h29min5seg*.

9. **Resultados para FT-Transformer**

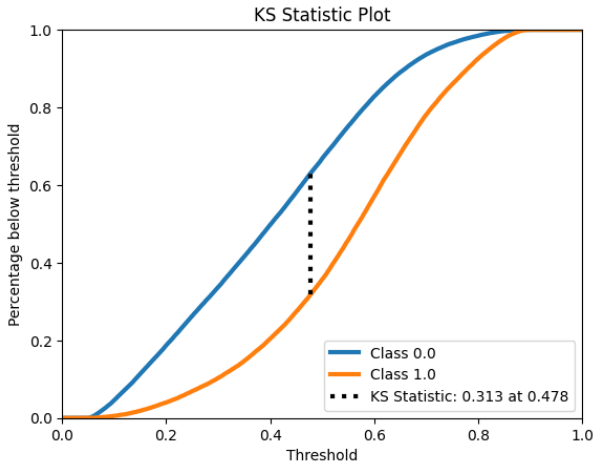

Figure 4.17: Gráfico KS do modelo *FT-Transformer* relativo ao Conjunto de Teste.

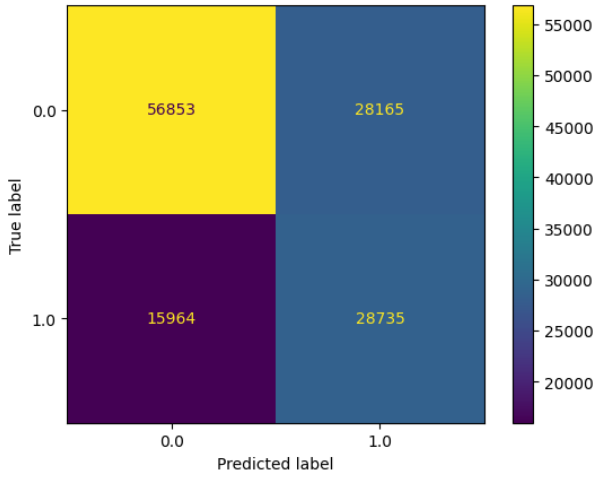

Figure 4.18: Matriz de Confusão do modelo *FT-Transformer* relativo ao Conjunto de Teste.

Para a abordagem em *FT-Transformer*, foram obtidas as seguintes métricas: ***Acurácia***: 0.6598; ***Recall***: 0.6429; ***Precisão***: 0.5050; ***F1***: 0.5657; ***AUROC***: 0.7140; ***KS***: 0.3126. O tempo total destinado à obtenção dos melhores hiper-parâmetros foi de: *31h5min7seg*.

De modo a contemplar uma visualização comparativa entre os resultados apresentados por cada uma das abordagens nesse trabalho, as tabelas 4.1 e 4.2 evidenciam todas a métricas obtidas, em ordem de análise.

Modelo	KS	ROC_AUC	Acurácia	Precisão	Recall	F1
MLP	0.2830	0.6940	0.6797	0.5987	0.2134	0.6940
TabTransformer	0.2866	0.6967	0.6508	0.4946	0.6086	0.5457
TabNet	0.2915	0.6998	0.6293	0.4742	0.6968	0.5643
Random Forest	0.2966	0.7043	0.6441	0.4878	0.6613	0.6508
GXB	0.3098	0.7131	0.6452	0.4894	0.6830	0.5702
Gradient Boosting	0.3100	0.7144	0.6588	0.5039	0.6402	0.5639
FT-Transformer	0.3126	0.7140	0.6598	0.5050	0.6429	0.5657
LightGBM	0.3132	0.7151	0.6488	0.4930	0.6786	0.5711
Ensemble	0.3169	0.7178	0.6543	0.4989	0.6701	0.5719

Table 4.1: Valores da métricas obtidas por cada modelo considerado no estudo.

Apesar de os resultados obtidos na tabela 4.1 constatarem uma similaridade no tocante à métrica escolhida para análise (KS), é importante analisar o tempo destinado a obtenção dos melhor hiper-parâmetros para cada abordagem.

Modelo	Tempo
Ensemble	35:14:49
FT-Transformer	31:05:07
TabNet	20:31:15
TabTransformer	18:29:05
RF	16:12:17
GB	14:10:52
MLP	06:26:13
LGBM	00:09:02
XGB	00:08:42

Table 4.2: Tempo que cada modelo levou para ser obtido (hh:mm:ss).

5

Discussão dos Resultados

5.1 Introdução

Em virtude da métrica escolhida para ser otimizada ser única (*Estatística KS*), todos as abordagens estudadas apresentaram uma performance relativamente similar entre os modelos. Nesse quesito, as valorações tenderam a variar de aproximadamente 0.28 a 0.32.

5.2 Comparativo de Performance

Dentre os modelos de *GBDT*, a melhor *Métrica KS* encontrada foi de 0.3149 (*Ensemble*). Para as abordagens de *Transformers* e *Mecanismos de Atenção*, todavia, a melhor métrica encontrada foi de 0.3026 (FT-Transformer), o que, apesar de não superar o melhor KS obtido, apresenta o potencial da abordagem como uma forte competidora no quesito de métricas obtidas para o inferência sobre dados tabulares.

5.3 Viabilidade e Limitações

Antes de discutir quanto à *Viabilidade* de uso dos modelos obtidos, vale ressaltar que, nesse trabalho, as LLMs foram introduzidas como modelos complexos que utilizam a arquitetura de *Transformers* e possuem bilhões de parâmetros. Esses atributos conferem ao uso dessa abordagem a necessidade de uma quantidade de memória considerável, além de um poder computacional elevado, o que muitas vezes não é acessível ao *"usuário comum"*. O treino desses modelos, além de requerer um enorme conjunto de dados, também exige um tempo considerável, aumentando os custos de obtenção.

A partir dessa problemática, as empresas já inseridas no mercado como referência no meio de *LLMs* terminam por disponibilizar seus modelos em *Nuvem*, consolidando os custos e oferecendo acesso a modelos de alta qualidade a uma gama de usuários que não precisam realizar o treinamento em "dispositivos individuais", além de provêr escalabilidade, manutenção, segurança, entre outros aspectos importantes para o mantimento de uma ferramenta de *IA*.

Quanto ao quesito de *Viabilidade*, apesar de a eficácia dos modelos variarem de acordo com a natureza/característica dos dados tabulares apresentados, foi constatado que, por apresentarem uma complexidade maior de implementação, o custo computacional provido pelo uso de *Transformers* resultou em um maior tempo destinado à obtenção dos melhores hiperparâmetros quando comparado às abordagens de GBDT. Mesmo apresentando uma performance similar, o treinamento e inferência realizados pelos modelos baseados em *Transformers* - como é evidenciado na *Tabela Comparativa de Métricas 4.2* - levaram aproximadamente 60 vezes mais tempo para serem concluídos, quando comparados aos modelos de *XGB* e *LGBM*. Os modelos considerados, em geral, apresentaram uma performance similar no quesito da *Métrica KS* selecionada para estudo, contudo o *Trade-Off* temporal não justificou a utilização de modelos de *Transformers* em prol dos modelos de *gbdt*, que permaneceram sendo, além de eficientes, interpretáveis.

Para cenários em que o tempo é um elemento crucial, a escolha da técnica para treinamento e inferência sobre um banco de dados tabular arbitrário se torna clara para os modelos tradicionais.

6

Conclusão

6.1 Recapitulação

Com a consolidação dos resultados e descobertas obtidas no contexto desse estudo, a aplicação de modelos baseados em *Transformers* se mostrou competitiva no quesito de *Performance*, mas temporalmente ineficiente em problemas de Classificação para Dados Tabulares. Foi possível, ao longo do trabalho, investigar o desempenho dos novos modelos *FT-Transformer, TabTransformer* e *TabNet* em comparação aos modelos tradicionais *RF, GB, XGB* e *LGBM*, além de também considerar modelos de abordagem básica em *RNA*, como *MLP*. Os resultados revelaram o potencial dessa nova abordagem em capturar relações complexas e dependências entre os atributos de dados tabulares, o que sugere uma possibilidade de aplicabilidade para cenários reais. Contudo, o tempo de obtenção dos melhores *hiper-parâmetros* de tais modelos, bem como os tempos de Treino e Inferência se mostraram um problema quando comparados aos intervalos de tempo relativos aos modelos de GBDT. Especificamente, foi fornecida uma avaliação comparativa abrangente dos modelos de GBDT e *Transformers* em dados tabulares de *Classificação Binária*, cuja métrica de avaliação priorizada foi a *Estatística KS (Kolmogorov-Smirnov)*, apresentando uma valoração de desempenho similar para as duas abordagens (0.30 - 0.32).

6.2 Pesquisas Futuras

Visto que esse trabalho apresentou um viés exlusivamente relacioando à obtenção de *Performance* de cada um dos modelos levantados, o quesito de precisão foi sobreposto ao aspecto de *Interpretabilidade*. Historicamente, modelos de *RNA* não apresentam uma *Interpretabilidade* tão didática/visual, como pode ser visto em modelos baseados em *Árvores de Decisão*. Isso acontece visto que as decisões tomadas em modelos de árvore são feitas de forma hierárquica, refletindo em uma maior explicabilidade de acordo com os caminhos tomados na árvore (que podem ser visualizados e compreendidos). Em contraposição a isso, modelos baseados em *RNAs*, como *MLP*, aprendem representações complexas e abstratas dos dados em aboragem de aproxi-

mação de funções (incluindo interações não-lineares, camadas ocultas, regularização etc), o que significa que as características aprendidas pelo modelo tendem a não ser visualmente explicáveis de forma trivial. Todavia, por apresentarem uma etapa de *Embedding* e Mecanismos de *Máscara* e *Atenção*, os modelos baseados em *Transformers* apresentam um nível de *Interpretabilidade* Vetorial, isto é, o mapeamento de atributos para um espaço vetorial contínuo, onde cada atributo categórico, por exemplo, é representado por um vetor numérico capaz de lhe prover uma relação semântica com outras categorias consideradas "similares". Dessarte, para análises em que a interpretabilidade é um ponto importante, cabe um estudo que abranja esse fator como uma métrica de avaliação para a escolha de modelos.

Em resumo, reconhecer uma mera classificação e otimização de métricas provenientes de diferentes modelos não é suficiente. Um outro aspecto imprescindível está na interpretabilidade dos resultados obtidos. Portanto, também é proposto a exploração da capacidade dos modelos de fornecer uma interpretação significativamente compreensível dos dados.

Quanto ao quesito temporal de treinamento e inferência discutidos na abordagem de Modelos baseados em *Transformers*, também é proposta uma análise de novos modelos que busquem soluções nessa arquitetura moderna, mas que procurem também otimizar sua eficiência temporal. Modelos como *SAINT: Improved Neural Networks for Tabular Data via Row Attention and Contrastive Pre-Training* (SOMEPALLI et al. (2021)), apresentado pela primeira vez em 2021, podem apresentar uma nova perspectiva que supere as abordagens analisadas nesse trabalho, tanto em quesito de acertabilidad, quanto temporal.

Por fim, considera-se também o aspecto randômico atribuído pelas bibliotecas utilizadas na implementação de cada um dos modelos estudados nesse trabalho. Apesar de haver uma forma de padronização de declaração das variáveis utilizadas no código - através da fixação de *"seeds"* - a randomicidade atribuída na análise não foi integralmente realizada. Nesse aspecto, apesar de repetições consecutivas do código apresentarem o mesmo *Pipeline* de análise, também permitem a ocorrência de leves variações quanto à obtenção das métricas, o que pode ser analisado mais profundamente em pesquisas futuras.

Algumas referências adicionais auxiliares utilizadas para embasamento e apoio no contexto dessa pesquisa - que não foram mencionadas ao longo do trabalho - podem ser encontradas a seguir:

DATA (2021 https://towardsdatascience.com/pytorch-widedeep-deep-learning-for-tabular-data-9cd1c48eb40d),

TABULAR DATA: TABTRANSFORMER DEEP DIVE (2022 https://towardsdatascience.com/transformers-for-tabular-data-tabtransformer-deep-dive-5fb2438da820),

TABULAR DATA (2021 https://www.kaggle.com/code/enigmak/tabnet-deep-neural-network-for-tabular-data/notebook),

STRUCTURED (2021 https://towardsdatascience.com/tabnet-deep-neural-network-for-structured-tabular-data-39eb4b27a9e4),

TEST (2017 https://towardsdatascience.com/kolmogorov-smirnov-test-84c92fb4158d),

KOLMOGOROV-SMIRNOV (KS),
PYTHON (2020 https://www.listendata.com/2019/07/KS-Statistics-Python.html)

Referências

AI dreamquark. **TabNet** : attentive interpretable tabular learning. 2019 https://github.com/dreamquark-ai/tabnet.

ARIK, S. ; PFISTER, T. TabNet: attentive interpretable tabular learning. **Proceedings of the AAAI Conference on Artificial Intelligence**, [S.l.], v.35, n.8, p.6679–6687, May 2021.

DATA pytorch-widedeep: deep learning for tabular. **reference1**. 2021 https://towardsdatascience.com/pytorch-widedeep-deep-learning-for-tabular-data-9cd1c48eb40d.

GORISHNIY, Y. et al. **Revisiting Deep Learning Models for Tabular Data**. 2023.

GVASCONS. **Tabular-Transformer-Thesis**. 2023 https://github.com/Gvascons/Tabular-Transformer-Thesis.

HUANG, X. et al. **TabTransformer**: tabular data modeling using contextual embeddings. 2020.

KOLMOGOROV-SMIRNOV (KS) TEST, C. sample distributions with the. **reference7**. 2022 https://towardsdatascience.com/comparing-sample-distributions-with-the-kolmogorov-smirnov-ks-test-a2292ad6fee5.

LUCIDRAINS. **TabTransformer**. 2019 https://github.com/lucidrains/tab-transformer-pytorch.

PYTHON, C. K. S. W. **reference8**. 2020 https://www.listendata.com/2019/07/KS-Statistics-Python.html.

SOMEPALLI, G. et al. SAINT: improved neural networks for tabular data via row attention and contrastive pre-training. **CoRR**, [S.l.], v.abs/2106.01342, 2021.

STRUCTURED, T. D. TabNet — Deep Neural Network for. **reference4**. 2021 https://towardsdatascience.com/tabnet-deep-neural-network-for-structured-tabular-data-39eb4b27a9e4.

TABULAR DATA hTabNet — Deep Neural Network for. **reference3**. 2021 https://www.kaggle.com/code/enigmak/tabnet-deep-neural-network-for-tabular-data/notebook.

TABULAR DATA: TABTRANSFORMER DEEP DIVE, T. for. **reference2**. 2022 https://towardsdatascience.com/transformers-for-tabular-data-tabtransformer-deep-dive-5fb2438da820.

TEST, K. **reference6**. 2017 https://towardsdatascience.com/kolmogorov-smirnov-test-84c92fb4158d.

VASWANI, A. et al. Attention Is All You Need. In: ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS. **Anais...** [S.l.: s.n.], 2017. p.6000–6010.

Apêndice

A

Mapping Study's Instruments

Table A.1: List of conferences on which the searches were performed.

Acronym	Conference
APSEC	Asia Pacific Software Engineering Conference
ASE	IEEE/ACM International Conference on Automated Software Engineering
CSMR	European Conference on Software Maintenance and Reengineering
ESEC	European Software Engineering Conference
ESEM	International Symposium on Empirical Software Management and Measurement
ICSE	International Conference on Software Engineering
ICSM	International Conference on Software Maintenance
ICST	International Conference on Software Testing
InfoVis	IEEE Information Visualization Conference
KDD	ACM SIGKDD International Conference on Knowledge Discovery and Data Mining
MSR	Working Conference on Mining Software Repositories
OOPSLA	Object-Oriented Programming, Systems, Languages and Applications
QSIC	International Conference On Quality Software
SAC	ACM Symposium on Applied Computing
SEAA	EUROMICRO Conference on Software Engineering and Advanced Applications
SEDE	19th International Conference on Software Engineering and Data Engineering
SEKE	International Conference on Software Engineering and Knowledge Engineering

Table A.2: List of journals in which the searches were performed.

Journal title
ACM Transactions on Software Engineering and Methodology
Automated Software Engineering
Elsevier Information and Software Technology
Elsevier Journal of Systems and Software
Empirical Software Engineering
IEEE Software
IEEE Computer
IEEE Transactions on Software Engineering
International Journal of Software Engineering and Knowledge Engineering
Journal of Software: Evolution and Process
Software Quality Journal
Journal of Software
Software Practice and Experience Journal

Table A.3: Search string per Search Engine.

Search Engine	Search String
Google Scholar	bug report OR track OR triage “change request” issue track OR request OR software OR “modification request” OR “defect track” OR “software issue” repositories maintenance evolution
ACM Portal	Abstract: "bug report" or Abstract:"change request" or Abstract:"bug track" or Abstract:"issue track" or Abstract:"defect track" or Abstract:"bug triage" or Abstract: "software issue" or Abstract: "issue request" or Abstract: "modification request") and (Abstract:software or Abstract:maintenance or Abstract:repositories or Abstract:repository
IEEExplorer (1)	((((((((((“Abstract": "bug report") OR "Abstract":"change request") OR "Abstract":"bug track") OR "Abstract":"software issue") OR "Abstract":"issue request") OR "Abstract":"modification request") OR "Abstract":"issue track") OR "Abstract":"defect track") OR "Abstract":"bug triage") AND "Abstract":software)
IEEExplorer (2)	((((((((((“Abstract": "bug report") OR "Abstract":"change request") OR "Abstract":"bug track") OR "Abstract":"software issue") OR "Abstract":"issue request") OR "Abstract":"modification request") OR "Abstract":"issue track") OR "Abstract":"defect track") OR "Abstract":"bug triage") AND "Abstract":maintenance)
IEEExplorer (3)	((((((((((“Abstract": "bug report") OR "Abstract":"change request") OR "Abstract":"bug track") OR "Abstract":"software issue") OR "Abstract":"issue request") OR "Abstract":"modification request") OR "Abstract":"issue track") OR "Abstract":"defect track") OR "Abstract":"bug triage") AND "Abstract":repositories)
IEEExplorer	((((((((((“Abstract": "bug report") OR "Abstract":"change request") OR "Abstract":"bug track") OR "Abstract":"software issue") OR "Abstract":"issue request") OR "Abstract":"modification request") OR "Abstract":"issue track") OR "Abstract":"defect track") OR "Abstract":"bug triage") AND "Abstract": repository)
Citeseer Library	(abstract: "bug report" OR abstract:"change request" OR abstract:"bug track" OR abstract:"issue track" OR abstract:"defect track" OR abstract:"bug triage" OR abstract: "software issue" OR abstract: "issue request" OR abstract: "modification request") AND (abstract:software OR abstract:maintenance OR abstract:repositories OR abstract:repository)
Elsevier	("bug report" OR "change request" OR "bug track" OR "issue track" OR "defect track" OR "bug triage" OR "software issue" OR "issue request" OR "modification request") AND (software OR maintenance OR repositories OR repository)
Scirus	("bug report" OR "change request" OR "bug track" OR "issue track" OR "defect track" OR "bug triage" OR "software issue" OR "issue request" OR "modification request") AND (software maintenance OR repositories OR repository) ANDNOT (medical OR aerospace)
ScienceDirect	("bug report" OR "change request" OR "bug track" OR "issue track" OR "defect track" OR "bug triage" OR "issue request" OR "modification request") AND LIMIT-TO(topics, "soft ware")
Scopus	("bug report" OR "change request" OR "bug track" OR "issue track" OR "defect track" OR "bug triage" OR "software issue" OR "issue request" OR "modification request") AND (software maintenance OR repositories OR repository)
Wiley	("bug report" OR "change request" OR "bug track" OR "issue track" OR "defect track" OR "bug triage" OR "software issue" OR "issue request" OR "modification request") AND (software maintenance OR repositories OR repository)
ISI Web of Knowledge	("bug report" OR "change request" OR "bug track" OR "issue track" OR "defect track" OR "bug triage" OR "software issue" OR "issue request" OR "modification request") AND (software maintenance OR repositories OR repository) ANDNOT (medical OR aerospace)
SpringerLink	("bug report" OR "change request" OR "bug track" OR "issue track" OR "defect track" OR "bug triage" OR "software issue" OR "issue request" OR "modification request") AND (software maintenance OR repositories OR repository) ANDNOT (medical OR aerospace)