Gabriel dos Santos Scatena

Democratizando a concessão de crédito com dados do *Open Finance*

Monografia apresentada ao Centro de Ciências Matemáticas Aplicadas à Indústria do Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, como parte dos requisitos para conclusão do MBA em Ciências de Dados. *VERSÃO FINAL*

Área de Concentração: Ciências de Dados

Orientador: Prof. Dr. Júlio Cezar Estrella

USP – São Carlos
Julho de 2023

Ficha catalográfica elaborada pela Biblioteca Prof. Achille Bassi
e Seção Técnica de Informática, ICMC/USP,
com os dados inseridos pelo(a) autor(a)

S277d

Scatena, Gabriel dos Santos
Democratizando a concessão de crédito com dados do Open Finance / Gabriel dos Santos Scatena; orientador Júlio Cezar Estrella. -- São Carlos, 2024.
87 p.

Trabalho de conclusão de curso (MBA em Ciência de Dados) -- Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, 2024.

1. Aprendizado computacional. 2. Risco de Crédito. 3. Sistema de Finanças Abertas (Open Finance). 4. Modelagem de Dados. 5. CRISP. I. Estrella, Júlio Cezar, orient. II. Título.

Bibliotecários responsáveis pela estrutura de catalogação da publicação de acordo com a AACR2:
Gláucia Maria Saia Cristianini - CRB - 8/4938
Juliana de Souza Moraes - CRB - 8/6176

Gabriel dos Santos Scatena

Democratizing credit granting based on Open Finance data

Final Paper submitted to the Center for Mathematical Sciences Applied to Industry of the Institute of Mathematics and Computer Sciences – USP, in partial fulfillment of the requirements for the MBA in Data Science. *FINAL VERSION*

Concentration Area: Data Science

Advisor: Prof. Dr. Júlio Cezar Estrella

USP – São Carlos
July 2023

AGRADECIMENTOS

Agradeço ao professor Júlio Cezar Estrella pela orientação e aos colegas de turma por toda a assistência e suporte nesta etapa fundamental para o processo de aprendizagem e crescimento profissional.

"We cannot solve our problems with the same thinking we used when we created them."

Albert Einstein

RESUMO

SCATENA, GABRIEL DOS SANTOS. **Democratizando a concessão de crédito com dados do *Open Finance*** . 2023. 65 p. Monografia (MBA em Ciências de Dados) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

A justificativa e importância do projeto de é que a concessão de crédito no Brasil apresenta um desafio significativo, especialmente para trabalhadores autônomos e empresários com rendas informais e variáveis. Ao longo da história, o acesso ao crédito tem sido historicamente baixo para esse grupo de profissionais em comparação com outros países. Isso se deve, em grande parte, ao fato de que as instituições financeiras tradicionalmente utilizam critérios como histórico de renda e fluxo de caixa estáveis na análise de crédito. Esses critérios acabam excluindo aqueles que possuem rendas irregulares ou informais, dificultando sua obtenção de crédito para investimentos, expansão de negócios ou mesmo para lidar com desafios financeiros imprevistos. Diante dessa realidade, torna-se essencial encontrar maneiras de melhorar a análise e concessão de crédito para esses profissionais, a fim de ajudá-los a superar obstáculos financeiros e impulsionar seu desenvolvimento econômico. Este trabalho de conclusão de curso propõe o desenvolvimento de um modelo baseado em dados do *Open Finance*, ou Sistema de Finanças Abertas, que possa auxiliar as instituições financeiras a tomar decisões mais precisas e justas na concessão de crédito, visando melhorar o acesso ao crédito e a situação financeira dessas pessoas.

Palavras-chave: *Open Finance*, Sistema de Finanças Abertas, Crédito, Aprendizado de máquina.

ABSTRACT

SCATENA, GABRIEL DOS SANTOS. **Democratizing credit granting based on Open Finance data**. 2023. 65 p. Monografia (MBA em Ciências de Dados) – Instituto de Ciências Matemáticas e de Computação, Universidade de São Paulo, São Carlos – SP, 2023.

Credit granting in Brazil presents a significant challenge, especially for self-employed workers and entrepreneurs with informal and variable incomes. Throughout history, access to credit has been historically low for this group of professionals compared to other countries. This is largely due to the fact that financial institutions traditionally use criteria such as stable income history and cash flow in credit analysis. These criteria end up excluding those with irregular or informal incomes, making it difficult for them to obtain credit for investments, business expansion, or even to cope with unforeseen financial challenges. Given this reality, it is essential to find ways to improve credit analysis and granting for these professionals in order to help them overcome financial obstacles and drive their economic development. This thesis proposes the development of a model based on Open Finance data, or Open Finance System, which can assist financial institutions in making more accurate and fair credit decisions, aiming to improve credit access and the financial situation of these individuals.

Keywords: Open Finance, Credit, Machine learning.

LISTA DE ILUSTRAÇÕES

LISTA DE ABREVIATURAS E SIGLAS

ML	Machine Learning
DT	*Decision Tree*
LR	*Logistic Regression*
RF	*Random Forest*
SVM	*Supporting Vector Machine*

SUMÁRIO

CAPÍTULO

1

INTRODUÇÃO

Esta monografia visa o desenvolvimento e possível implementação de técnicas de aprendizado de máquina com o intuito da criação de um modelo que permita a democratização e concessão de crédito com base em dados do *Open Finance*.

O primeiro capítulo deste trabalho de conclusão de curso apresenta a motivação e o objetivo deste estudo, além de abordar a organização dos demais capítulos.

1.1 Objetivos

O presente projeto o desenvolvimento e possível implementação de técnicas de aprendizado de máquina com o intuito da criação de um modelo que permita a democratização e concessão de crédito com base em dados do *Open Finance*.

Objetivos específicos:

1. Avaliação de modelo preditivo de risco de crédito com base em dados do *Open Finance*;

2. Avaliação dos diversos algoritmos de aprendizado de máquina;

3. Comparação do das métricas dos modelos obtidos com o modelo utilizado pela empresa provedora dos dados.

O presente trabalho busca desenvolver um modelo preditivo, selecionado por avaliação de diversos algoritmos de Machine Learning (*ML*), como: *Logistic Regression* (LR), *Supporting Vector Machine* (SVM), *Decision Tree* (DT) e *Random Forest* (RF).

1.2 Organização

No Capítulo 2, realizamos uma revisão abrangente das técnicas e modelos relevantes para a seleção de métodos, fornecendo uma análise aprofundada de seus principais conceitos com referências à literatura citada. Além disso, apresentamos conceitos gerais relacionados à análise de risco de crédito no Brasil e exploramos o papel do *Open Finance* nesse contexto. Em seguida, no Capítulo 3, descrevemos o conjunto de dados, a metodologia adotada e apresentamos os resultados obtidos ao aplicar os métodos selecionados a esse conjunto de dados específico. Por fim, no Capítulo 4, concluímos a monografia discutindo os resultados encontrados, suas limitações e sugerindo possíveis direções para futuros desenvolvimentos nessa área.

CAPÍTULO

2

REVISÃO BIBLIOGRÁFICA

Este capítulo tem como objetivo apresentar os principais conceitos relacionados as analises de risco de crédito no Brasil.Bem como, explicar brevemente o papel do *Open Finance* no mercado financeiro.

2.1 Contextualização

2.1.1 O do problema da concessão de crédito no Brasil

A concessão de crédito no Brasil enfrenta desafios significativos, especialmente para trabalhadores autônomos, bolsistas e empresários com rendas informais e variáveis. Diferentemente de outros países, onde a concessão de crédito é mais acessível, às instituições financeiras no Brasil geralmente baseiam suas decisões em históricos de renda e fluxo de caixa estáveis. Isso cria obstáculos para profissionais que não se enquadram nesses critérios, dificultando o acesso ao crédito e afetando negativamente sua situação financeira. Diante dessa realidade, torna-se fundamental encontrar maneiras de aprimorar a análise e a concessão de crédito para esses grupos, a fim de promover a inclusão financeira. Este projeto de pesquisa propõe o desenvolvimento de um modelo baseado em dados do *Open Finance*, ou Sistema de Finanças Abertas, que possa auxiliar as instituições financeiras a tomar decisões mais precisas e justas na concessão de crédito, visando melhorar o acesso ao crédito e a situação financeira dessas pessoas.

2.1.2 Concessão de crédito e seus desafios

No contexto brasileiro, a concessão de crédito enfrenta diversos desafios, especialmente para trabalhadores autônomos e empresários com rendas informais e variáveis. Esses grupos enfrentam dificuldades em comprovar sua capacidade de pagamento devido à falta de históricos de renda estáveis e padronizados, o que resulta em baixa disponibilidade de crédito e limitações em suas atividades econômicas. Além disso, a falta de acesso ao crédito pode agravar a desigualdade

socioeconômica e limitar o desenvolvimento desses profissionais. Portanto, é essencial investigar e propor soluções para superar esses obstáculos e democratizar a concessão de crédito no país (BARBOSA, 2020).

2.1.3 Análise de crédito tradicional

A análise de crédito tradicional é baseada em modelos que utilizam critérios pré-determinados para avaliar a capacidade de pagamento do solicitante, bons pagadores são conhecidos como adimplentes e maus pagadores como inadimplentes. Esses modelos consideram principalmente informações financeiras e histórico de crédito, como renda declarada, histórico de pagamentos, dívidas em aberto, entre outros. Essas informações são utilizadas para calcular uma pontuação de crédito, score, que é então utilizada como base para a tomada de decisão na concessão de crédito. No entanto, esses modelos tradicionais apresentam limitações ao lidar com trabalhadores autônomos e empresários com rendas informais e variáveis, pois não conseguem capturar adequadamente sua capacidade de pagamento e risco de crédito. Essas limitações resultam em uma exclusão financeira significativa para esses grupos, impedindo seu acesso a oportunidades de investimento e crescimento econômico ou profissional. Sendo assim, novos modelos são necessários (HASAN; POPP; OLÁH, 2020) para a análise do risco de crédito (SHI *et al.*, 2022). Os métodos de aprendizado de máquina (BHATORE; MOHAN; REDDY, 2020) e a busca por melhoria contínua resultam em uma grande variedade e possibilidade de aplicações (LOUZADA; ARA; FERNANDES, 2016).

2.1.4 Os cinco Cs

Os "5 Cs"da análise de crédito são um conjunto de critérios que são tradicionalmente considerados pelos credores ao avaliar a concessão de crédito a um indivíduo ou empresa. Esses critérios fornecem uma estrutura para avaliar a capacidade de um tomador de empréstimo em honrar suas obrigações financeiras (BAZARBASH, 2019). A seguir, discutirei cada um dos "5 Cs"em detalhes:

Caráter *(Character)*

O caráter refere-se à reputação e histórico financeiro do solicitante de crédito. Os credores analisam a capacidade do indivíduo em cumprir seus compromissos financeiros anteriores, observando se há histórico de inadimplência, atrasos de pagamento ou registros negativos. Informações como histórico de crédito, histórico de pagamentos e referências comerciais podem ser consideradas para avaliar o caráter do tomador de empréstimo.

Capacidade *(Capacity)*

A capacidade refere-se à capacidade financeira do tomador de empréstimo para honrar suas obrigações de pagamento. Os credores avaliam a renda e a estabilidade financeira do indivíduo, comparando-a com suas despesas e dívidas existentes. É importante que o tomador

de empréstimo tenha uma fonte de renda estável e suficiente para cobrir os pagamentos do empréstimo.

Capital *(Capital)*

O capital refere-se aos ativos financeiros e patrimoniais do solicitante de crédito. Os credores analisam a posse de ativos, como propriedades, veículos ou investimentos, como forma de garantia ou como fonte potencial de pagamento em caso de inadimplência. O capital disponível pode influenciar a decisão de concessão de crédito, uma vez que oferece uma segurança adicional ao credor.

Colateral *(Collateral)*

O colateral refere-se aos bens ou garantias fornecidos pelo tomador de empréstimo para garantir o pagamento do crédito. Em alguns casos, um credor pode exigir que o solicitante de crédito forneça um bem, como imóvel, veículo ou outro ativo de valor, como garantia para o empréstimo. Isso oferece uma forma adicional de proteção ao credor caso o tomador de empréstimo não consiga cumprir suas obrigações.

Condições *(Conditions)*

As condições referem-se ao contexto econômico e às circunstâncias específicas em que o crédito está sendo solicitado. Os credores consideram fatores externos, como a estabilidade econômica, perspectivas do setor de negócios e regulamentações governamentais, para avaliar o risco associado ao empréstimo. Além disso, as condições também podem incluir a finalidade do crédito, o prazo do empréstimo, as taxas de juros e outros termos e condições relevantes.

É importante destacar que a importância de cada um dos "5 Cs"pode variar dependendo do contexto e das políticas internas de cada instituição financeira. A combinação desses critérios permite aos credores tomar decisões informadas e avaliar o risco associado à concessão de crédito a um tomador de empréstimo específico.

2.1.5 O papel dos dados do Open Finance

O *Open Finance*, ou Sistema de Finanças Abertas, refere-se ao movimento de disponibilização e acesso a dados financeiros por meio de APIs (Interfaces de Programação de Aplicativos) e padrões abertos (OMARINI, 2018). Esse conceito possibilita a transparência e a interoperabilidade entre diferentes instituições financeiras, permitindo o compartilhamento seguro de informações financeiras de clientes (HJELKREM; LANGE; NESSET, 2022). No contexto da concessão de crédito, o sistema de finanças abertas desempenha um papel crucial ao disponibilizar dados financeiros detalhados e atualizados, que podem ser utilizados para uma análise mais precisa da capacidade de pagamento dos solicitantes (LAPLANTE; KSHETRI, 2021). Ao utilizar os dados do *Open Finance*, as instituições financeiras podem obter uma visão mais completa e em tempo real da situação financeira dos clientes, reduzindo assim o risco de crédito

e ampliando as oportunidades de acesso ao crédito para trabalhadores autônomos, bolsistas e empresários com rendas informais e variáveis (ARAUJO, 2022).

A pesquisa bibliográfica revela uma ampla variedade de abordagens relacionadas à seleção de variáveis para a construção de modelos de análise de crédito. Embora a metodologia tradicional seja eficaz, existem várias oportunidades de melhoria. Uma solução definitiva para esses desafios permanece em aberto devido à complexidade do ecossistema. Nesse contexto, a utilização de técnicas de aprendizado de máquina, combinadas com o uso de dados provenientes do *Open Finance*, será avaliada. A metodologia adotada para essa análise será descrita detalhadamente no capítulo subsequente.

CAPÍTULO

3

METODOLOGIA

3.1 Metodologia

Nesse capítulo serão descritas as diversas técnicas e abordagens utilizadas em cada uma das etapas propostas. Além disso, será utilizado o modelo de CRISP, bem como o método científico como norteador do trabalho.

Etapas do CRISP:

1. Entendimento do Negócio
2. Entendimento dos Dados
3. Preparação dos Dados
4. Modelagem
5. Avaliação
6. Implantação
7. Monitoramento

Etapas do Método Científico:

1. Observação
2. Formulação da Pergunta de Pesquisa
3. Hipótese
4. Coleta de Dados

5. Análise de Dados
6. Interpretação dos Resultados
7. Repetição e Reprodução
8. Comunicação dos Resultados
9. Desenvolvimento de Teoria

Vale ressaltar que o objetivo é produzir um passo a passo e fornecer ferramentas básicas para a construção de um modelo de risco de crédito, do início ao fim. O foco está nos métodos utilizados, isto é, tratamentos, transformações e abordagens, e não no resultado final do modelo. Também é importante destacar que, apesar das etapas serem apresentadas de forma sequencial, na prática, existe a necessidade constante de voltar a etapas anteriores para ajustes e adequação dos dados para as etapas posteriores.

As etapas desse trabalho são:

1. Coleta de Dados
2. Limpeza de Dados (*Data Cleaning*)
3. Exploração e Análise de Dados
4. Pré-processamento de Dados
5. Depuração de Dados (*Data Cleansing*)
6. Lidando com Classes Desbalanceadas
7. Engenharia de Variáveis (ou Engenharia de Características)
8. Seleção de Variáveis
9. Seleção de Modelo
10. Treinamento de Modelo
11. Avaliação de Modelo
12. Ajuste de Hiperparâmetros
13. Interpretação de Modelo (Opcional)
14. Implantação de Modelo
15. Monitoramento e Manutenção de Modelo

3.1.1 Coleta de Dados

A primeira etapa do processo é a coleta de dados. Nesse estágio, os dados são geralmente coletados em ambientes de nuvem ou em bancos de dados relacionais, como SQL, ou em bancos de dados não relacionais, como o *MongoDB*.

Inicialmente, a *Klavi* forneceu uma base de dados com mais de seiscentos mil usuários registrados, em um ambiente de nuvem, utilizando o *SharePoint* como um arquivo CSV (separado por vírgulas). Este banco de dados foi gerado nos estágios iniciais da empresa e foi derivado de dados de *Open Finance*, que foram pré-processados em um sistema legado complexo que inclui tarefas de análise de texto e classificação. Os dados recebidos não vieram na forma bruta, mas na forma pré-processada, contendo as variáveis utilizadas pela empresa em análises posteriores. No total, há 540 variáveis rotuladas de K00001 a K00540 e outros campos como banco e *user_id*.

Na primeira etapa, que é a coleta de dados, é essencial exercer cuidado ao lidar com arquivos CSV, especialmente quando eles contêm texto. Algumas medidas de precaução incluem a especificação do separador utilizado, que normalmente é a vírgula, mas em alguns casos pode ser o ponto e vírgula. Além disso, para textos em português que contenham acentos e caracteres especiais, ou seja, além das 26 letras do alfabeto, é recomendável utilizar a opção de *encoding* como *latin-1*.

3.1.2 Limpeza de Dados

Na etapa de limpeza de dados, uma série de ações é executada para garantir a qualidade e a integridade dos dados. Isso envolve a remoção de colunas que são irrelevantes ou desnecessárias para o desempenho do modelo, pois manter apenas as informações essenciais é fundamental. Ainda, verifica-se a presença de inconsistências, valores atípicos e erros nos dados, com o intuito de corrigi-los ou tratá-los adequadamente. Outro aspecto crítico é lidar com dados ausentes ou incompletos, o que envolve estratégias como preenchimento dos valores em falta ou a exclusão de registros com informações ausentes. Essas ações visam preparar os dados para as etapas subsequentes do projeto, onde a análise e modelagem serão realizadas com base em informações confiáveis e consistentes.

A base de dados de seiscentos mil usuários, possui uma parte dos dados rotulados, isso é, são identificados bons e maus pagadores, de forma binária, 0 e 1, respectivamente. Porém, somente 27.188 dos usuários tem rótulos.

Outro fator importante é que muitos dos usuários possuem mais de uma conta cadastrada ou contas em múltiplos bancos. Uma vez que não há como especificar para qual das contas o rótulo é válido, somente usuários com rótulos e conta única serão mantidos.

As variáveis (colunas) que não serão utilizadas devem ser removidas, por exemplo: *'l1_score', 'l2_score', 'l3_score', 'trace_id', 'query_date', 'account', 'date', 'odds*', 'score'. Dessa forma, economiza-se memória, garantindo que no modelo seja utilizado apenas o que é

explicitamente necessário.

Grande parte dos valores faltantes, NaN, já estão identificados como -999999 (e suas variações: -999999.00 e -999999 .0). Esses valores foram substituídos por np.nan, com a finalidade de deixar todos esses valores em uma única forma.

Os dados da variável *user_id* foram "anonimizados", discretizados, transformando-os em números inteiros, a fim de economia de memória e evitar lidar com texto e caracteres especiais. Os dados de banco foram discretizados em valores numéricos inteiros, com a finalidade de tornar esta variável categórica. Um dicionário foi gerado e salvo para que seja possível obter os bancos originais quando necessário.

Desta forma todos os dados resultantes são do tipo numérico, especificamente do tipo *float64*. Para aumentar a economia de memória, muitas variáveis podem ser convertidas para tipo *int32*.

A detecção de *outliers* desempenha um papel crucial na tomada de decisões relacionadas a variáveis. Com o propósito de realizar essa tarefa, foram selecionadas 5 técnicas: *'RobustZScore', 'IsolationForest', 'OneClassSVM', 'IQR', 'ZScore'*. Para cada uma delas, foi atribuído um rótulo 1 para indicar um *outlier* e 0 para indicar que não é um *outlier*. Ao somar os valores obtidos de todas as técnicas, obtivemos uma pontuação de *outliers* para cada entrada na base de dados. Um valor de 5 indica que um registro é considerado valor atípico por todas as técnicas, enquanto um valor de 0 indica que não é considerado por nenhuma delas. Isso permite uma avaliação abrangente da presença de *outliers* em cada entrada de dados.

3.1.3 Exploração e Análise de Dados

A etapa de Exploração e Análise de Dados, muitas vezes referida como EDA (*Exploratory Data Analysis*), desempenha um papel fundamental no processo de ciência de dados. Nesta fase, mergulhamos profundamente nos dados para compreender a sua distribuição, identificar relações entre variáveis e descobrir padrões que podem revelar informações valiosas.

A análise exploratória fornece *insights* iniciais que orientam as decisões subsequentes ao longo do projeto, desde a seleção de recursos até a construção e ajuste de modelos. A visualização desempenha um papel crucial, permitindo-nos representar graficamente os dados de maneira clara e informativa, tornando possível a identificação de tendências e peculiaridades que podem não ser óbvias em uma análise puramente numérica.

A exploração profunda dos dados é a etapa que mais demanda tempo, todavia, possibilita o entendimento dos dados e extrair conhecimento valioso que impulsionará o projeto de ciência de dados.

Para uma análise descritiva preliminar a função *describe*, da biblioteca Pandas, fornece estatísticas como média, desvio padrão, máximo, mínimo e outras tendências centrais, além da

forma da distribuição. Os percentis podem ser escolhidos de acordo com a necessidade, por exemplo de 5 em 5 por cento. O resultado da contagem (*Count*) é a quantidade de valores não nulos. Para variáveis categóricas ou do tipo objeto, a função *unique* retorna os valores únicos, juntamente com o número de categorias. Adicionalmente, é possível obter a frequência (*freq*) do valor mais comum naquela coluna.

A contagem de valores faltantes e de valores zero por variável é relevante. Uma vez que uma variável possui apenas um desses valores, ela é considerada uma constante e não traz informação útil para o modelo, podendo ser eliminada.

Uma abordagem importante quando se têm dados rotulados é verificar a distribuição desses valores, especialmente os valores 0 e NaN, em cada uma das classes. A construção de *boxplots* para cada classe é uma maneira eficaz de realizar essa análise. Se os intervalos interquartis (caixas) de cada classe não se sobrepõem, isso sugere que esses valores não têm a mesma distribuição em ambas as classes. Isso pode indicar a possibilidade de criar uma variável com base nessa diferença de distribuição entre as classes. Analogamente, essa contagem pode ser realizada por linhas (*user_id*) ao invés de colunas (variáveis), e uma análise similar à anterior pode ser aplicada.

Essa metodologia pode ser valiosa na identificação de características altamente discriminantes entre as classes, permitindo a criação de novas variáveis que capturem informações importantes para o modelo de classificação. Isso, por sua vez, pode melhorar o desempenho do modelo ao fornecer informações adicionais e relevantes para a tarefa em questão. Portanto, a análise de distribuição de valores entre classes é um passo importante na preparação de dados para problemas de classificação em ciência de dados.

Vale ressaltar a importância de observar que a distribuição de valores em uma classe apresenta variações significativas em relação aos valores NaN (não disponíveis). Isso sugere um padrão de dados ausentes que não ocorre aleatoriamente, o que é conhecido como "*Missing Not At Random*"(MNAR). Em outras palavras, a ausência de dados está relacionada sistematicamente a informações não observadas, ou seja, a falta de dados está relacionada a eventos ou fatores que não foram medidos, como erros humanos. Essa observação é fundamental, uma vez que a natureza do padrão de dados ausentes pode afetar a análise e a modelagem de maneira significativa, requerendo estratégias específicas para lidar com esse tipo de ausência de dados. Portanto, é importante considerar o MNAR ao tratar de dados ausentes em uma análise de dados.

Há também o caso de variáveis da *klavi* que possuem valores diferentes de zero e nulo para apenas uma parcela pequena da população, como por exemplo, "renda proveniente de empresa própria"e "investimento em ações". Nesses casos, é possível que essas variáveis sejam altamente assimétricas, com a maioria dos valores concentrada em zero ou nulo, enquanto apenas uma pequena parte da população apresenta valores distintos.

Uma contagem das contas conectadas por banco é importante, uma vez que essa é uma

variável categórica e pode apresentar distribuição diferenciada para cada banco e para cada classe. Foram encontrados 24 bancos distintos, porém muitos deles apresentam poucos usuários na base de dados (menos de 150). Uma possibilidade é eliminar as entradas de bancos com poucos usuários. Fazendo isso, 95% das entradas são mantidas e restam 12 bancos distintos. A remoção dos dados associados a outros bancos não apenas contribui para a economia de memória e recursos computacionais, mas também aprimora a adequação do conjunto de dados para fins de aprendizado de máquina.

Na modelagem de risco de crédito e na seleção de características, a escolha do método de correlação depende da natureza dos dados e das relações entre as variáveis. Por exemplo:

- Se as relações forem lineares e normalmente distribuídas, a correlação de Pearson pode ser apropriada.
- Se as relações forem monótonas, mas não necessariamente lineares, as correlações de Spearman ou Kendall podem ser escolhas melhores.
- Quando se lida com tipos mistos de dados (categóricos, ordinais, contínuos), o coeficiente de correlação de Kendall Tau pode ser mais adequado devido à sua natureza não paramétrica.

Devido à presença de tipos mistos de dados, a natureza não linear das relações entre as características e o fato de que a distribuição dos dados não é homogênea, o coeficiente de correlação de Kendall é mais apropriado para lidar com o conjunto de dados da *klavi*. Essa escolha se alinha com a necessidade de adotar uma abordagem não paramétrica e considerar as características específicas do conjunto de dados para análises de risco de crédito.

A alta correlação entre as variáveis pode indicar redundância ou multicolinearidade. Isso sugere que algumas características podem ser supérfluas e potencialmente podem ser removidas para simplificar o modelo e reduzir o ruído.

3.1.4 Pré-processamento de Dados

Na etapa de pré-processamento de dados, várias ações são executadas para preparar os dados para a modelagem. Isso inclui:

- Lidar com características categóricas por meio de técnicas como codificação (por exemplo, *one-hot encoding, label encoding*) ou o uso de técnicas como *target encoding.*
- Dividir os dados em conjuntos de treinamento e teste/validação, permitindo avaliar o desempenho do modelo em dados não vistos.

- Dimensionar características numéricas, se necessário, utilizando técnicas como escalonamento Min-Max ou padronização, a fim de garantir que todas as características estejam na mesma escala e contribuam igualmente para o modelo.

Essas etapas são fundamentais para garantir que os dados estejam em um formato adequado para a modelagem e que o modelo seja capaz de aprender com eficiência a partir deles. Exemplo de variável categórica é banco, que foi previamente discretizada, restando apenas modificar seu tipo.

A divisão dos dados em conjuntos de treinamento e teste pode ser realizada utilizando a função *train_test_split* da biblioteca scikit-learn. Para esse trabalho utilizaremos uma divisão de 80% para treinamento e 20% para teste.

O escalonamento e a padronização devem ser realizados nos dados de treinamento e, em seguida, aplicados nos dados de teste, a fim de evitar vazamento de dados. Essa abordagem garante que o modelo seja treinado e avaliado em conjuntos de dados independentes.

A utilização da biblioteca do *scikit-learn*, `pipeline`, é útil para automatizar e simplificar o pré-processamento de dados, garantindo que as etapas sejam executadas de forma consistente em treinamento e teste. Com o `pipeline`, pode-se criar um fluxo de trabalho que inclui o escalonamento, a padronização e outras etapas de pré-processamento, facilitando a aplicação consistente dessas transformações em seus dados.

Essa abordagem não apenas evita erros, mas também torna o fluxo de trabalho de pré-processamento mais eficiente e menos propenso a problemas de vazamento de dados, garantindo que o modelo seja avaliado de maneira adequada.

3.1.4.1 A Abordagem GridSearch para Otimização da Preparação de Dados

Na busca por aprimorar o desempenho de modelos de risco de crédito, a seleção cuidadosa dos métodos ótimos de preparação de dados se torna de extrema importância. O objetivo primordial é alcançar resultados métricos superiores ao mesmo tempo em que se minimiza a variância.

A utilização da técnica semelhante ao *GridSearch* nesse contexto possibilita uma exploração sistemática de várias estratégias de imputação, procedimentos de normalização e configurações algorítmicas. Esse processo tem como alvo identificar a combinação mais adequada que maximiza o poder preditivo mantendo a estabilidade em relação a várias métricas. Estratégias como imputação com base na média, mediana, valor mais frequente e valor constante, combinadas com a normalização por meio do *MinMaxScaler*, facilitam uma avaliação abrangente do modelo.

O processo iterativo, abrangendo diversos solucionadores e estratégias de imputação, é executado de forma sistemática para garantir uma avaliação completa dos pipelines propostos.

A agregação de resultados para métricas como precisão, *recall*, pontuação F1, estatística KS, coeficiente de Gini e AUC (área sob a curva ROC) possibilita um processo de tomada de decisão informada na adaptação do pipeline de preparação de dados a tarefas específicas de modelagem de risco de crédito.

O *DataFrame* resultante engloba as métricas de desempenho resumidas, oferecendo i*nsights* sobre a interação entre técnicas de preparação e configurações algorítmicas. Esse processo visa a otimização do desempenho dos modelos de risco de crédito por meio da identificação das melhores práticas de preparação de dados e sua aplicação a tarefas de modelagem específicas.

No risco de crédito, métricas classificadoras podem determinar se um candidato pertence às categorias de inadimplência ou não inadimplência. O Gini é mais comumente utilizado em conjuntos de dados desequilibrados, nos quais a probabilidade por si só torna difícil prever um resultado. O coeficiente de Gini é uma métrica padrão na avaliação de risco porque a probabilidade de inadimplência é relativamente baixa.

3.1.5 Depuração de Dados (Data Cleansing)

Na etapa de limpeza de dados, as seguintes ações são realizadas:

- Identificar os tipos de dados de cada característica (por exemplo, numéricos, categóricos, texto).
- Verificar se há características com o tipo de dado incorreto e corrigi-las.

Essa etapa é fundamental para garantir que os dados estejam em um formato adequado para a análise e modelagem subsequentes. Identificar e corrigir problemas de tipos de dados incorretos é essencial para evitar erros e garantir a consistência dos dados ao longo do projeto de ciência de dados.

3.1.5.1 Tratamento de Duplicatas

Na etapa de tratamento de duplicatas, as seguintes ações são realizadas:

- Verificar a existência de duplicatas no conjunto de dados.
- Se duplicatas forem encontradas, removê-las para evitar que criem viés no modelo.

O tratamento de duplicatas é essencial para manter a integridade dos dados e evitar qualquer viés que possa surgir devido a registros duplicados. Isso garante que as observações no conjunto de dados sejam únicas e que não haja repetições que possam distorcer a análise e modelagem subsequentes.

3.1.5.2 Tratamento de Valores Nulos/NaN

Na etapa de tratamento de valores nulos/NaN, as seguintes ações são realizadas:

- Identificar as características que possuem valores ausentes (nulos ou NaN).
- Decidir sobre uma estratégia para lidar com os valores ausentes, como imputação (substituição de valores ausentes por um valor apropriado) ou exclusão (remoção de linhas ou colunas com valores ausentes). Existe também a possibilidade de "não fazer nada", uma vez que alguns algoritmos são capazes de lidar com valores faltantes.

O tratamento de valores nulos é fundamental para garantir a qualidade e a confiabilidade dos dados. Dependendo da natureza dos dados ausentes e dos objetivos da análise, é necessário determinar a melhor abordagem para lidar com esses valores ausentes, seja preenchendo-os com valores apropriados ou eliminando-os de forma adequada.

Algoritmos que geralmente requerem imputação:

1. Regressão Logística: A imputação é necessária, pois esse algoritmo assume características de entrada contínuas.
2. *k-Nearest Neighbors* (k-NN): A imputação é necessária porque o k-NN calcula distâncias entre pontos de dados, e valores ausentes podem distorcer os cálculos de distância.
3. *Support Vector Machines* (SVM): A imputação é necessária, pois o SVM requer características de entrada contínuas.
4. Redes Neurais: A imputação é necessária para redes neurais, pois elas requerem características de entrada contínuas, e valores ausentes podem interferir nos cálculos da rede.
5. *Naive Bayes*: A imputação é necessária, uma vez que o *Naive Bayes* assume características de entrada contínuas.

Algoritmos que podem lidar com valores ausentes:

1. *Decision Tree*: Árvores de decisão podem lidar diretamente com valores ausentes durante seu processo de criação de divisões.
2. *Random Forest*: pode lidar com valores ausentes ao fazer a média das previsões das árvores.
3. Máquinas de Gradiente *Boosting* (GBM): GBM pode lidar com valores ausentes de forma semelhante às árvores de decisão.
4. *XGBoost*: é capaz pode lidar com valores ausentes durante o processo de construção da árvore.

5. *LightGBM*: pode lidar com valores ausentes de forma semelhante ao XGBoost.

6. *HistGradientBoosting*: um algoritmo de *gradient boosting*, também é capaz de lidar com valores ausentes de maneira eficaz.

7. *CatBoost*: é projetado para lidar com variáveis categóricas e valores ausentes automaticamente durante o treinamento.

3.1.5.3 Validação Cruzada

Ao avaliar o desempenho de seus modelos, é recomendável usar validação cruzada em vez de uma única divisão de treinamento/teste. A validação cruzada fornece uma estimativa mais robusta do desempenho do modelo, ao calcular a média dos resultados em várias divisões de treinamento/teste. Isso ajuda a reduzir o viés associado a uma única divisão de dados e fornece uma avaliação mais confiável do desempenho do modelo.

3.1.6 Lidando com Classes Desbalanceadas

O desequilíbrio de classes em si não é necessariamente um problema, e em muitos casos, é uma característica natural dos dados que não requer intervenção. O desequilíbrio de classes torna-se um problema quando ele afeta negativamente o desempenho do modelo de aprendizado de máquina em relação a seus objetivos específicos.

No entanto, o desequilíbrio de classes pode se tornar uma complicação quando afeta a capacidade do modelo de aprender e generalizar adequadamente. Alguns dos problemas que podem surgir devido ao desequilíbrio de classes incluem:

- Viés do modelo: Modelos de aprendizado de máquina podem favorecer a classe majoritária e não aprender efetivamente as nuances da classe minoritária, levando a uma baixa capacidade de detecção dessa classe.

- Métricas enganosas: Métricas de avaliação de desempenho, como a acurácia, podem ser enganosas em cenários de classes desequilibradas, pois um modelo que prevê sempre a classe majoritária ainda pode ter uma acurácia alta. No entanto, esse modelo não será útil na prática.

- Perda de informações: O desequilíbrio pode levar à perda de informações valiosas sobre a classe minoritária, o que pode ser crítico em muitos contextos.

Portanto, embora o desequilíbrio de classes em si não seja um problema intrínseco, ele pode representar um desafio significativo em projetos de aprendizado de máquina e, nesses casos, é importante aplicar estratégias apropriadas para lidar com ele.

Etapas e técnicas a serem avaliadas:

3.1.6.1 Analisar a Distribuição das Classes

Verificar a distribuição das classes na variável alvo para identificar desequilíbrios significativos entre as classes.

3.1.6.2 Técnicas de Reamostragem de Dados

Se houver um desequilíbrio severo de classes, considerar o uso de técnicas de reamostragem para equilibrar as classes.

- ***Oversampling* (Superamostragem):** Aumentar o número de instâncias na classe minoritária duplicando ou gerando amostras sintéticas (por exemplo, usando *SMOTE - Synthetic Minority Over-sampling Technique*).
- ***Undersampling* (Subamostragem):** Reduzir o número de instâncias na classe majoritária removendo aleatoriamente amostras.
- **Reamostragem Híbrida:** Combinar as duas técnicas mencionadas acima, se apropriado.
- ***Class Weighting* (Ponderação de Classes):** Muitos algoritmos de classificação e bibliotecas permitem atribuir pesos mais altos às classes minoritárias durante o treinamento do modelo. Isso ajuda o modelo a dar mais atenção às classes sub representadas.

O tratamento de classes desbalanceadas pode ser importante para garantir que o modelo seja capaz de lidar adequadamente com todas as classes, especialmente as menos representadas. Porém, devemos avaliar se essas estratégias ajudam a melhorar o desempenho do modelo, estabilidade e robustez ao longo do tempo e não somente na fase de treino.

3.1.7 Engenharia de Variáveis (ou Engenharia de Características)

A Engenharia de Variáveis desempenha um papel fundamental no desenvolvimento de modelos. Essa etapa envolve a criação de novas características, quando necessário, com base em conhecimento de domínio ou *insights* obtidos durante a Análise Exploratória de Dados (EDA). Além disso, é crucial a transformação de características existentes para torná-las mais adequadas ao modelo. Isso pode incluir transformações como a aplicação de logaritmo ou a geração de características polinomiais. Essas ações visam aprimorar a representação dos dados, tornando-os mais informativos e relevantes para o modelo, contribuindo para a construção de soluções de ciência de dados mais eficazes e precisas.

Nesse contexto, o *Weight of Evidence (WOE)* é uma métrica amplamente usada em modelagem de risco de crédito e análise de crédito. Ele mede o poder informativo de uma variável em relação a uma variável de resposta binária, como a probabilidade de inadimplência em um

empréstimo. O WOE é calculado a partir da taxa de inadimplência em diferentes categorias da variável explicativa.

Para calcular o WOE variáveis explicativas que estão em valores contínuos são transformadas em categóricas. Optamos por criar 20 categorias, referente a percentis de 5% cada. Dessa forma, obtivemos uma maneira de representar o efeito das variáveis independentes na probabilidade de inadimplência. Um WOE maior indica uma maior chance de inadimplência, enquanto um WOE menor indica uma menor probabilidade de inadimplência.

3.1.8 Seleção de Variáveis

A seleção de variáveis desempenha um papel fundamental no desenvolvimento de modelos de aprendizado de máquina. Sua principal finalidade é escolher as variáveis mais relevantes que resultarão em métricas otimizadas para o modelo. Além disso, essa etapa visa a redução da dimensionalidade, o que não apenas economiza custo de processamento e memória, mas também ajuda a minimizar o ruído e a perda de desempenho causada por variáveis menos relevantes. Portanto, a seleção de variáveis é uma estratégia essencial para criar modelos mais eficientes, precisos e interpretáveis.

Nossa abordagem multifacetada incluiu três métodos principais: filtro, "*Embedded*"e "*Wrapper*". Um compilado das técnicas e nome tipo de método utilizado pode ser observado na tabela 1.

Filtro	**Wrapper**	**Embedded**
ANOVA, *chi-square*, IV	*Foward Elimination*, *Bi-directional elimination*	*Recursive Feature Elimination* e *Boruta*
Baixo risco de sobreajuste, rápido e direto	Deve ser avaliado para cada algoritmo separadamente	Intermediário entre filtro e *wrapper*. Reduz sobreajuste

Tabela 1 – Compilado das técnicas para seleção de variáveis.

Empregou-se o método *VarianceThreshold* para identificar e eliminar características constantes ou com variação muito baixa, escalonando os dados quando necessário. Realizamos análises univariadas para avaliar a relevância de cada característica em relação à variável alvo. Além disso, examinamos as correlações entre características, identificando relações entre pares de características que poderiam afetar o desempenho do modelo. O Valor de Informação (*IV – Information Value*) foi utilizado, uma vez que é uma métrica poderosa no contexto da seleção de características para modelos de classificação. Ele auxilia na avaliação do poder preditivo das características e pode orientar as decisões sobre manter ou descartar características específicas.

Para a seleção de variáveis contínuas, utilizamos o escore Z e a análise de variância (ANOVA). Já para as variáveis que foram transformadas em categóricas, empregamos o teste qui-quadrado (*chi-square*).

Foi utilizado a avaliação da importância das características pelo *Random Forest*. Também aplicamos métodos de embrulho (*Wrapper*), como a seleção avançada (*Foward Elimination*), e eliminação bidirecional (*Bi-directional elimination* ou *Stepwise Selection*). Além disso, empregamos técnicas iterativas de seleção de características, como *RFE (Recursive Feature Elimination)* e *Boruta*, que envolve a utilização de *Z-Score* e outros critérios de seleção de características.

O nome "Boruta"é uma referência ao deus eslavo da floresta, que ajuda a identificar o que é valioso. De maneira semelhante, o *Boruta* ajuda a identificar as características mais relevantes em um conjunto de dados. O funcionamento do *Boruta* é relativamente simples, mas eficaz. Ele cria cópias embaralhadas das características originais (conhecidas como "sombra"ou *"shadow"features*) e as mescla com o conjunto de dados real. Em seguida, aplica um algoritmo de aprendizado de máquina (geralmente um *Random Forest*) para classificar as características com base em sua importância. As características reais que têm uma importância estatisticamente significativa em relação às características sombra são mantidas, enquanto as características que não se destacam são descartadas.

Essa técnica é particularmente útil quando se lida com conjuntos de dados desequilibrados ou quando se deseja garantir a robustez da seleção de características. No passado, o *Boruta* provou ser uma ferramenta valiosa em nossa metodologia para garantir que apenas as características mais informativas fossem incluídas em nossos modelos de aprendizado de máquina.

3.1.9 Seleção de Modelo (algoritmo)

A escolha do algoritmo ideal para cada problema pode ser um desafio não trivial. Isso envolve a análise de diversos fatores, como a dimensionalidade dos dados, a presença de valores ausentes (NaN), a distribuição dos dados, o número de classes, o tipo das variáveis, entre outros. Com o objetivo de extrair o máximo de informações dos dados e aplicar todo o conhecimento adquirido durante o curso, optamos por avaliar vários algoritmos, a fim de selecionar o mais adequado para o problema em questão. Essa abordagem nos permite escolher o modelo que melhor se adapte às características específicas do nosso conjunto de dados, garantindo um melhor resultado das métricas obtidas.

Relação dos algoritmos avaliados:

1. *Logistic Regression*
2. *k-Nearest Neighbors* (k-NN)
3. *Support Vector Machines* (SVM)
4. *Neural Networks*
5. *Naive Bayes*
6. *Decision Trees*

7. *Random Forest*
8. *Gradient Boosting Machines* (GBM)
9. *XGBoost*
10. *LightGBM*
11. *CatBoost*
12. *HistGradientBoostingClassifier*

3.1.10 Treinamento de Modelo

No treinamento do modelo, adotamos uma abordagem estruturada e robusta. Inicialmente, para testes preliminares, utilizamos a divisão de dados por meio do método *"train test split,"* alocando 80% dos dados para treinamento, com uma semente aleatória (*random_seed*) de 42. Além disso, devido ao desbalanceamento dos dados, garantimos a estratificação (*stratify*) com base nos rótulos de classe, o que assegura que a distribuição das classes seja preservada tanto nos dados de treinamento quanto nos de teste.

Para análises mais aprofundadas e confiáveis, optamos pela validação cruzada (CV) com um valor de 10 *folds*. Sempre que possível, configuramos o parâmetro *"class_weight"* como *"balanced"*, o que ajuda a lidar com o desbalanceamento das classes, dando um peso maior às classes minoritárias.

É importante ressaltar que, uma vez que alguns algoritmos não aceitam valores ausentes (NaN), adotamos a estratégia de imputar esses valores por meio de valores numéricos, garantindo que os modelos possam ser treinados adequadamente e que nenhum dado seja perdido no processo. Essas práticas refletem nossa abordagem cuidadosa e abrangente no treinamento de modelos para obter resultados robustos e confiáveis.

3.1.11 Avaliação de Modelo

Na etapa de avaliação do modelo, adotamos uma abordagem abrangente para garantir uma análise completa de seu desempenho. Utilizamos uma ampla gama de métricas de avaliação, como acurácia, acurácia balanceada, F1-ponderado, precisão, *recall*, pontuação F1, coeficiente Kappa, coeficiente de correlação de Matthews (MCC), estatística KS, índice Gini e AUC (área sob a curva ROC).

Os resultados de cada modelagem são registrados e armazenados em um *dataframe* dedicado. Essa prática garante a rastreabilidade e reprodutibilidade dos experimentos, permitindo uma análise detalhada dos resultados ao longo do tempo. Além disso, os resultados da validação cruzada de cada algoritmo são representados em gráficos de *boxplot*, facilitando a visualização e análise das distribuições de desempenho.

Utilizamos o conjunto de teste/validação para avaliar o quão bem o modelo generaliza para novos dados, garantindo que ele não esteja superajustado aos dados de treinamento e seja capaz de realizar previsões precisas em situações do mundo real. Essa abordagem rigorosa de avaliação nos permite selecionar as métricas mais apropriadas para a tarefa em questão e garantir que o modelo atenda aos requisitos de desempenho desejados.

3.1.12 Ajuste de Hiperparâmetros

O ajuste dos hiperparâmetros se torna necessário uma vez que o melhor algoritmo foi selecionado. Esse processo é fundamental para otimizar ainda mais o desempenho do modelo. Embora o *grid search* seja uma opção comum, é importante destacar que ele não garante a descoberta do valor mínimo global, melhores valores das métricas do modelo, dos hiperparâmetros.

Diante disso, também exploramos outras abordagens, como a busca Bayesiana e a pesquisa aleatória (*Random Search*) de hiperparâmetros. Essas técnicas permitem uma exploração mais eficiente e abrangente do espaço de hiperparâmetros, auxiliando na identificação de configurações que levem a um desempenho superior do modelo. A combinação dessas estratégias nos permite afinar os hiperparâmetros de maneira mais eficaz e obter um modelo altamente otimizado.

3.1.13 Interpretação do Modelo (Opcional)

A interpretação do modelo é uma etapa valiosa em que exploramos ferramentas como a importância das características (*feature importance*), que nos fornece informações sobre quais características são mais relevantes para o modelo. No caso, *Random Forest Feature Importance*, avalia a diminuição na impureza dos nós da árvore de decisão, ajudando-nos a compreender como as decisões são tomadas.

Além disso, utilizamos técnicas como os valores SHAP (*SHapley Additive exPlanations*), que nos oferecem *insights* detalhados sobre como cada característica contribui individualmente para as previsões do modelo. Essas abordagens nos permitem interpretar o modelo de forma significativa e compreender o impacto de cada variável em suas decisões, agregando valor e *insights* importantes para o negócio.

3.1.14 Implantação do Modelo

A fase de implantação do modelo é uma etapa crucial, na qual buscamos levar o modelo da fase de desenvolvimento para um ambiente de produção, onde ele possa realizar previsões em novos dados não vistos anteriormente.

Uma opção é a implantação de modelos por meio de APIs (*Application Programming Interfaces*) que permitem a integração do modelo em sistemas de produção, como sites, aplicati-

vos ou sistemas empresariais. Com isso, podemos enviar requisições para a API do modelo, que retorna previsões em tempo real.

Para uma implantação eficaz e de fácil utilização, proponho uma abordagem simples e amigável, aproveitando uma ferramenta poderosa chamada *Streamlit*. A qual, oferece uma maneira intuitiva de criar aplicativos da web interativos com *Python*, através de uma interface amigável em que os usuários possam preencher um formulário com seus dados, como informações financeiras e pessoais.

Melhor ainda é poder fazer a concessão dos dados financeiros via *Open Finance*, sendo um processo mais rápido e seguro, além de reduzir a fricção com o usuário. Com base nesses dados, o modelo pode fazer previsões sobre as chances de um usuário ser aprovado para um empréstimo em diferentes bancos e qual o valor provável do empréstimo. Isso oferece aos usuários uma experiência direta e prática, permitindo que obtenham informações valiosas de forma rápida e fácil.

Além disso, podemos integrar o aplicativo *Streamlit* com o pipeline de produção, garantindo que os dados do usuário sejam processados e as previsões sejam geradas em tempo real. Com essa abordagem, estamos democratizando o acesso às análises de crédito e oferecendo aos usuários uma ferramenta poderosa para tomar decisões financeiras informadas. É uma maneira eficaz de aproveitar o modelo desenvolvido e fornecer benefícios tangíveis para o público-alvo.

3.1.15 Monitoramento e Manutenção do Modelo

Uma vez que o modelo tenha sido implantado em um ambiente de produção, o trabalho ainda não está concluído. É essencial estabelecer um sistema de monitoramento contínuo para avaliar o desempenho do modelo no mundo real. Isso implica acompanhar de perto as previsões que o modelo faz e verificar se elas permanecem precisas e relevantes ao longo do tempo.

Além disso, é fundamental estar preparado para realizar atualizações no modelo, caso seja necessário. À medida que os dados de entrada evoluem ou mudam, o modelo pode precisar de ajustes para manter sua eficácia. Esse processo de monitoramento e manutenção garante que o modelo continue sendo uma ferramenta valiosa e confiável para a organização, fornecendo informações precisas para apoiar a tomada de decisões.

No próximo capítulo os resultados e discussões sobre cada uma das etapas apresentadas na metodologia serão abordados.

CAPÍTULO

4

DESENVOLVIMENTO E RESULTADOS

4.1 Considerações Iniciais

Nesta seção, apresentaremos de maneira objetiva e direta os resultados obtidos, destacando suas repercussões. A ênfase será na concisão para facilitar a legibilidade. Os códigos utilizados durante a análise podem ser encontrados no repositório do Github, disponível no seguinte link: https://github.com/gabrielscatena/TCC_MBA_USP_2024.git

Vale lembrar que apesar das etapas serem apresentadas de forma sequencial, na prática, a criação de um modelo é um processo complexo que diversas vezes irá exigir que volte a etapas anteriores, desenvolva novas hipóteses, faça testes e para alcançar resultados melhores.

4.1.1 Coleta de dados

Os dados foram fornecidos pela klavi no formato CSV (valores separados por vírgula), compreendendo uma base de clientes de 604.581 entradas e 548 colunas, além de uma base contendo os rótulos de parte dos clientes, 27188 entradas. A empresa já havia realizado o pré processamento desses dados, abordando questões como duplicatas e valores ausentes. Para garantir uma leitura ausente de erros do arquivo por meio do Python, adotamos as seguintes medidas:

1. Especificação do *encoding* como *latin-1*;
2. Leitura em blocos (*chunks*) seguida pela concatenação desses blocos;
3. Junção do banco de dados de clientes com os dados que contêm as etiquetas (classes: bons pagadores e maus pagadores).

Após a junção das bases, verificou-se a existência de valores duplicados em relação ao *'user_id'*. Os valores duplicados a princípio não deveriam ser um problema, uma vez que cada usuário

pode ter mais de uma conta distinta conectada ao sistema de finanças abertas. Todavia os rótulos fornecidos eram os mesmo para todas conectadas de cada *'user_id'*, logo para evitar qualquer possibilidade de erro ou seleção da conta equivocada de cada usuário, foi necessária uma intervenção.

4.1.2 Limpeza de Dados (Data Cleaning)

Na base utilizada nesse trabalho foram mantidos apenas usuários com uma única conta conectado. Dessa forma, cada rótulo pertence inequivocamente àquela conta. Essa operação resultou em 18.983 entradas únicas, sendo classe 0 (bons pagadores) 17.983 usuários e classe 1 (maus pagadores) 1000 usuários.

Além disso, colunas que não seriam utilizadas no desenvolvimento do trabalho foram eliminadas, como por exemplo *'l1_score', 'l2_score', 'l3_score', 'trace_id', 'query_date', 'account'*. Esse procedimento resultou em uma redução do número de colunas de 552 para 543.

Os valores faltantes já haviam sido identificados, porém, estavam apresentados de maneira inconsistente, podendo assumir três variações: `-999999`, `-999999.0` e `-999999.00`. Portanto, foi necessária uma uniformização. Esses valores foram substituídos por `np.nan` (*Numpy Not a Number*).

Adicionalmente, notou-se que os dados em `'user_id'` e `'bank'` estavam no formato de *string* (texto). Com o intuito de otimizar o uso de memória e garantir a anonimização, esses dados foram discretizados, ou seja, convertidos em valores numéricos. Foram gerados dicionários para mapear os bancos, facilitando a identificação no final do processo de modelagem, por exemplo.

Com o objetivo de otimizar o uso de memória, foram identificados os valores máximo e mínimo no dataframe, que foram, respectivamente, 1.375.788,93 e -367.104,33. Essa análise permitiu a escolha adequada de tipagem para as variáveis, resultando na conversão de muitas delas de `float64` para `float32` ou `int32`. Essa abordagem levou a uma redução significativa na memória utilizada no dataframe, passando de 78,8 MB para 39,5 MB.

Após a conclusão dessas operações, a base de dados foi convertida para um arquivo Excel, visando assegurar rastreabilidade. Cada etapa do processo foi documentada, proporcionando uma trilha clara das transformações realizadas.

4.1.3 Exploração e Análise de Dados

A Exploração e Análise de Dados (EDA), parte essencial da ciência de dados, desempenha um papel crucial no entendimento da distribuição, relações entre variáveis e identificação de padrões nos dados. Essa fase fornece *insights* iniciais que orientam as decisões subsequentes no projeto, desde a seleção de recursos até a construção e ajuste de modelos.

Na análise descritiva preliminar, utilizamos a função `describe` da biblioteca Pandas, que

fornece estatísticas como média, desvio padrão, máximo, mínimo, e outras tendências centrais, além da forma da distribuição. A escolha dos percentis pode ser personalizada, por exemplo, a cada 5%. Além disso, foram incluídos valores importantes como moda e *information value* (IV), proporcionando uma visão abrangente dos dados.

Esses resultados foram salvos em formatos Excel, facilitando a verificação e acesso por parte de usuários menos familiarizados com Python, contribuindo para a acessibilidade e compreensão dos dados.

Inicialmente, foi realizada a contagem de valores `NaN` e valores iguais a zero. Essa contagem tem por finalidade verificar se existem colunas (*features*) que contêm unicamente um desses, em caso positivo, essa variável é considerada uma constante que não gera valor ao modelo e deve ser eliminada. Foram identificadas 18 *features* contendo somente zeros e 4 contendo somente `NaN`.

A contagem da presença de valores zeros e `NaN` conjuntamente foi realizada e constatou-se que 139 variáveis possuem mais de 99% desses valores. No entanto, nesse caso não podemos eliminar essas *features*, uma vez que zero e `NaN` são valores com significados distintos e podem trazer informação ao modelo. Uma lista dessas variáveis foi gerada, com a finalidade de investigar o significado de informação nas etapas posteriores.

Ainda nesse contexto, foi observado que a distribuição da contagem de valores `NaN` e valores zero para cada uma das classes tem perfil distinto. Desta forma, gera a possibilidade de criação de duas novas *features* '`Count of 0`' e '`Count of NaN`'. Esse tópico será discutido nas etapas posteriores.

Além disso, essas análises possibilitaram a identificação de algumas *features* com valores que seriam matematicamente impossíveis, pois eram geradas em dependência de outras duas variáveis. Entretanto, essas *features* apresentavam consideravelmente mais valores `NaN` do que as duas variáveis iniciais, indicando uma possível situação de MNAR (*Missing Not At Random*). No contexto de MNAR, a ausência de dados está relacionada de forma sistemática aos dados não observados, ou seja, a falta de observação está ligada a eventos ou fatores não medidos, como erros humanos. Dessa forma, essas variáveis foram removidas do conjunto de dados.

Por fim, constatou-se que ao restringir os dados apenas aos 12 bancos específicos (mais de 150 usuários), garantimos que mais de 95% dos IDs de usuário permaneçam no *dataframe*. A remoção dos dados associados a outros bancos não apenas contribui para economia de memória e recursos computacionais, mas também aprimora a adequação do conjunto de dados para fins de aprendizado de máquina.

Nesta fase, conduzimos uma análise de correlação entre as variáveis utilizando os métodos de `Pearson`, `Kendall` e `Spearman`, uma vez que cada um deles pode fornecer informações distintas. Estabelecemos um limite absoluto de correlação (*threshold*) de 0.9 para determinar quais *features* poderiam ser removidas de acordo com cada método. Avaliamos tanto a interseção

quanto a união das *features* a serem removidas nos três métodos. O mapa de calor com os coeficientes de correlação para cada método foi gerado, exemplo na figura 1, método de Kendall.

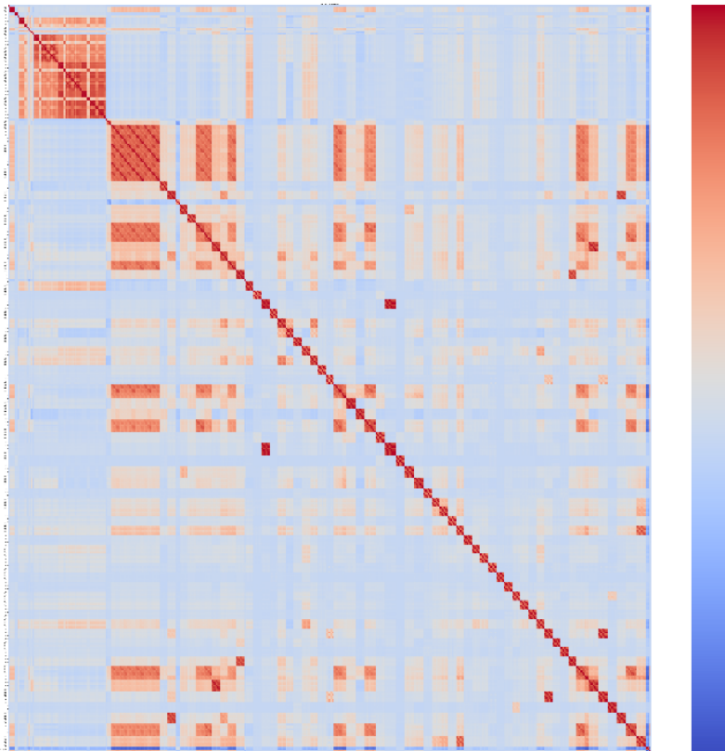

Figura 1 – Mapa de Calor para Coeficiente de Correlação de Kendall

A análise de correlação indica a possível eliminação de 188 variáveis pela interseção dos três métodos e 359 pela união. Para visualização desses resultados, foram gerados mapas de calor. Optamos por uma abordagem mais conservadora ao escolher a remoção de variáveis pela interseção dos três métodos como o método preferencial para dar continuidade aos estudos.

A correlação com a classe binária pode ser interpretada como uma análise de regressão logística, onde se busca entender a relação entre as variáveis independentes e a variável de resposta (classe binária). Neste contexto, uma correlação significativa pode indicar o impacto ou influência de determinadas variáveis na predição das classes.

Simultaneamente, conduzimos uma análise de correlação entre as variáveis e as classes (regressão logística). É notável observar que as variáveis `count_of_nan` e `count_of_zero` destacam-se como algumas das mais correlacionadas com cada uma das classes. Essa observação sugere que essas variáveis podem desempenhar um papel relevante na predição das classes no contexto da análise de regressão logística.

4.1.4 Pré-processamento de Dados

Durante a fase de pré-processamento, diversas ações são implementadas para preparar os dados para a modelagem. Por exemplo, a variável "banco,"previamente discretizada, foi modificada para o tipo categórica.

O escalonamento e a padronização são aplicados inicialmente nos dados de treinamento e, posteriormente, replicados nos dados de teste para evitar vazamento de informações. Essa abordagem assegura que o modelo seja treinado e avaliado em conjuntos de dados independentes.

Para auxiliar nessa tarefa e evitar vazamento de dados, a biblioteca `pipeline` do scikit-learn foi empregada para automatizar e simplificar o pré-processamento. Garantindo consistência nas etapas tanto no treinamento quanto no teste. Com o uso do `pipeline`, é possível criar um fluxo de trabalho que inclui escalonamento, padronização e outras transformações, facilitando a aplicação uniforme dessas etapas nos dados. Tornando o processo de pré-processamento mais eficiente e menos suscetível a problemas de vazamento de dados, garantindo uma avaliação adequada do modelo.

Na busca por aprimorar o desempenho dos modelos de risco de crédito, a seleção criteriosa dos métodos de preparação de dados é de extrema importância. O objetivo principal é alcançar métricas superiores ao mesmo tempo em que se minimiza a variância. A utilização de técnicas semelhantes ao `GridSearch` permite uma exploração sistemática de diversas estratégias de imputação, procedimentos de normalização e configurações algorítmicas.

Estratégias de imputação, como constante, média, mediana e mais frequente, foram aplicadas em combinação com diferentes técnicas de escalonamento e redução de dimensionalidade.

Estratégias avaliadas:

- Imputação: constante (zero), média, mediana, moda.
- Escalonamento Min-Max.
- Escalonamento padrão (*Standard*).
- Transformação Quantil.
- Discretização `KBins`.
- Análise de Componentes Principais (PCA).
- Decomposição de Valores Singulares Truncados (SVD).
- Power Transformation.
- Eliminação Recursiva de Características (RFE).*
- RFE com escalonamento padrão.*

*Devido ao alto custo computacional, RFE será utilizado na seleção de variáveis.

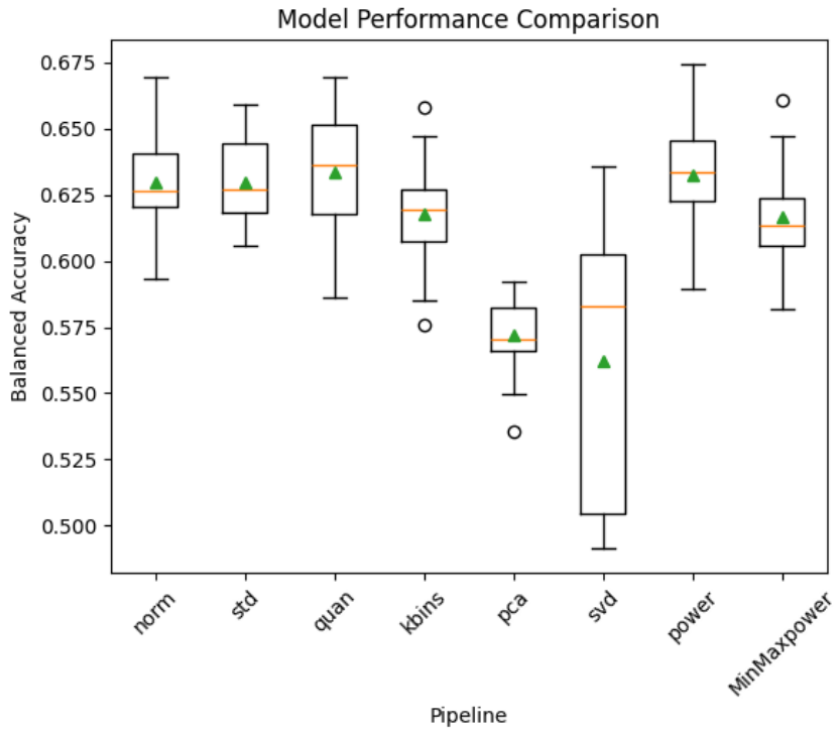

Figura 2 – Abordagem *GridSearch* para o pré-processamento de dados.

Vale ressaltar que nessa etapa as técnicas aplicadas possuem caráter exploratório e objetivo de prover indicativos do que é mais apropriado de ser utilizado no modelo final. Todavia, esse procedimento precisa ser realizado novamente após lidar com classes desbalanceadas, seleção de *features* e engenharia de *features*.

4.1.5 Depuração de Dados (Data Cleansing)

Boa parte da depuração dos dados foi realizada em etapas prévias, dentre elas:

- Eliminação de duplicatas.
- Abordagens para lidar com valores faltantes.
- Tipificação apropriada das variáveis.
- Eliminação de erros, por exemplo, o caso de eliminação de variáveis em possível situação de MNAR.

Para evitar redundância na discussão, os tópicos supracitados não serão abordados novamente.

4.1.6 Lidando com Classes Desbalanceadas

Embora o desequilíbrio de classes em si não seja um problema intrínseco, pode apresentar desafios significativos em projetos de aprendizado de máquina, tornando essencial a aplicação de estratégias apropriadas para enfrentá-lo. Nesse contexto, optamos por utilizar e avaliar duas técnicas específicas com esse propósito:

Técnicas de Reamostragem de Dados: Superamostragem, subamostragem e reamostragem híbrida são estratégias aplicadas para equilibrar a distribuição das classes no conjunto de dados.

Ponderação de Classes *(Class Weighting)*: Muitos algoritmos de classificação e bibliotecas permitem a atribuição de pesos mais elevados às classes minoritárias durante o treinamento do modelo, buscando mitigar o impacto do desequilíbrio. Os resultados detalhados dessas técnicas estão apresentados na tabela.

Técnicas Avaliadas

Superamostragem (Oversampling):

- ADASYN
- SMOTE
- SMOTETomek
- SMOTEENN

Subamostragem (Undersampling):

- EditedNearestNeighbours
- TomekLinks
- ClusterCentroids
- NearMiss
- RepeatedEditedNearestNeighbours
- RandomUnderSampler

Híbrida (Combined Sampling):

- SMOTETomek
- SMOTEENN

As técnicas foram aplicadas na base de dados em sua forma bruta e os parâmetros `Class Weight` com a opção balanceada ou `None` (nenhum). Idealmente, esse tipo de análise deve ser realizada na base com todos os tratamentos já realizados e somente com as variáveis relevantes.

Adicionalmente, podemos verificar que as técnicas de subamostragem `EditedNearestNeighbours`, `TomekLinks` e `RepeatedEditedNearestNeighbours` são muito sutis nos valores de reamostragem, mantendo um perfil de métricas muito similar ao original.

Para algumas dessas técnicas, é possível ajustar o parâmetro `sampling_strategy`, escolhendo quais classes serão reamostradas e até mesmo especificar a proporção entre as classes. No caso dessas 3 técnicas acima, não é possível especificar essa proporção.

Um aspecto crucial ao lidar com conjuntos de dados desbalanceados é evitar depender exclusivamente da acurácia como métrica, especialmente quando o parâmetro `class_weight` não é especificado (padrão = `None`). Essa abordagem pode levar o modelo a aprender a classificar todas as instâncias na classe majoritária. Na entrada 0 da tabela 3, verificamos uma acurácia de 95%; no entanto, observamos que o *F1 Score* é próximo de zero. Além disso, o valor de ROC AUC igual a 0.5 sugere que a classificação foi praticamente aleatória. Portanto, é essencial utilizar métricas como *F1 Score*, que consideram o desbalanceamento das classes, para uma avaliação mais precisa do desempenho do modelo. Os resultados detalhados dessas técnicas estão apresentados na tabela.

	Resampling Technique	Class Weight	Accuracy	F1 Score	Precision	ROC AUC	Balanced F1 Score	Balanced Accuracy	Kappa	Gini Coefficient	MCC	KS Statistic	AUC PRC	Sample shape X class0	Sample shape X class1
0	Original	None	0.95	0.01	0.20	0.50	0.92	0.50	0.01	0.00	0.02	0.19	0.08	14386	800
1	Original	balanced	0.58	0.13	0.07	0.59	0.70	0.59	0.04	0.19	0.08	0.22	0.08	14386	800
2	ADASYN	None	0.59	0.14	0.08	0.61	0.70	0.61	0.05	0.23	0.10	0.24	0.08	14386	14184
3	ADASYN	balanced	0.59	0.14	0.08	0.60	0.70	0.60	0.05	0.20	0.09	0.22	0.08	14386	14184
4	SMOTETomek	None	0.60	0.14	0.08	0.61	0.71	0.61	0.05	0.21	0.10	0.24	0.08	14129	14129
5	SMOTETomek	balanced	0.60	0.14	0.08	0.60	0.71	0.60	0.05	0.20	0.09	0.22	0.08	14160	14160
6	SMOTEENN	None	0.48	0.13	0.07	0.59	0.61	0.59	0.03	0.19	0.08	0.22	0.08	9201	12942
7	SMOTEENN	balanced	0.58	0.14	0.08	0.60	0.69	0.60	0.05	0.21	0.09	0.24	0.08	9196	12995
8	SMOTE	None	0.64	0.14	0.08	0.60	0.74	0.60	0.05	0.21	0.10	0.23	0.08	14386	12947
9	SMOTE	balanced	0.59	0.14	0.08	0.60	0.70	0.60	0.05	0.21	0.09	0.23	0.08	14386	12947
10	EditedNearestNeighbours	None	0.95	0.01	0.10	0.50	0.92	0.50	0.00	0.00	0.01	0.20	0.08	12328	800
11	EditedNearestNeighbours	balanced	0.58	0.14	0.08	0.60	0.69	0.60	0.05	0.21	0.09	0.21	0.08	12328	800
12	TomekLinks	None	0.95	0.01	0.50	0.50	0.92	0.50	0.01	0.00	0.05	0.18	0.08	14081	800
13	TomekLinks	balanced	0.58	0.13	0.07	0.59	0.70	0.59	0.04	0.18	0.08	0.22	0.08	14081	800
14	ClusterCentroids	None	0.27	0.11	0.06	0.53	0.36	0.53	0.01	0.06	0.03	0.19	0.07	800	800
15	ClusterCentroids	balanced	0.27	0.11	0.06	0.54	0.36	0.54	0.01	0.09	0.05	0.20	0.07	800	800
16	NearMiss	None	0.14	0.08	0.04	0.42	0.18	0.42	-0.02	-0.16	-0.11	0.22	0.04	800	800
17	NearMiss	balanced	0.14	0.08	0.04	0.42	0.18	0.42	-0.02	-0.16	-0.11	0.22	0.04	800	800
18	RepeatedEditedNearestNeighbours	None	0.94	0.02	0.10	0.50	0.92	0.50	0.01	0.00	0.01	0.22	0.08	11309	800
19	RepeatedEditedNearestNeighbours	balanced	0.57	0.14	0.08	0.61	0.69	0.61	0.05	0.21	0.09	0.22	0.08	11309	800
20	RandomUnderSampler	None	0.51	0.13	0.07	0.59	0.63	0.59	0.04	0.18	0.08	0.21	0.07	800	800
21	RandomUnderSampler	balanced	0.52	0.12	0.07	0.58	0.64	0.58	0.03	0.15	0.07	0.18	0.07	800	800

Figura 3 – Métricas das Técnicas de Reamostragem de Dados e Ponderação de Classes.

4.1.7 Engenharia de Variáveis (ou Engenharia de Características)

A Análise Exploratória de Dados inicial permitiu observar que, para a maioria das variáveis, apesar de serem contínuas, apresentavam alta assimetria, com muitos valores baixos e um rápido aumento nos percentis superiores (o contrário também ocorre para algumas variáveis).

Essa observação sugere a possibilidade de aplicar Engenharia de Variáveis para torná-las mais adequadas ao modelo. Dessa forma, a transformação logarítmica pode ser uma opção para construir o modelo.

Observou-se que o Weight of Evidence (WOE) havia sido previamente aplicado. Nesse método, valores contínuos são transformados em categóricos, criando 20 categorias representando percentis de 5% cada. Assim, um novo conjunto de dados foi gerado, tratando todas as variáveis como categóricas. Essa abordagem é relevante, pois NaN, *outliers* e outros valores que poderiam exigir tratamento prévio são agrupados em uma única categoria, eliminando a necessidade de pré-processamentos adicionais.

Essa ideia gerou resultados relevantes, sobretudo no que diz respeito à visualização dos dados. Em alguns casos, foi possível gerar gráficos nos quais a separação das classes era nítida e clara, sugerindo uma elevada capacidade discriminativa para determinada variável.

Foram geradas visualizações do tipo *boxplot* e *stripplot* para os dados contínuos e dados discretizados (agrupados) em 20 categorias. Para os valores NaN, foi criada uma categoria específica com rótulo '0'. Além disso, normalização e padronização foram aplicadas. Bem como, a utilização de reamostragem, no caso *undersampling* foi utilizado para as classes terem número de indivíduos comparáveis, como podemos ver nas figuras 4 e 5.

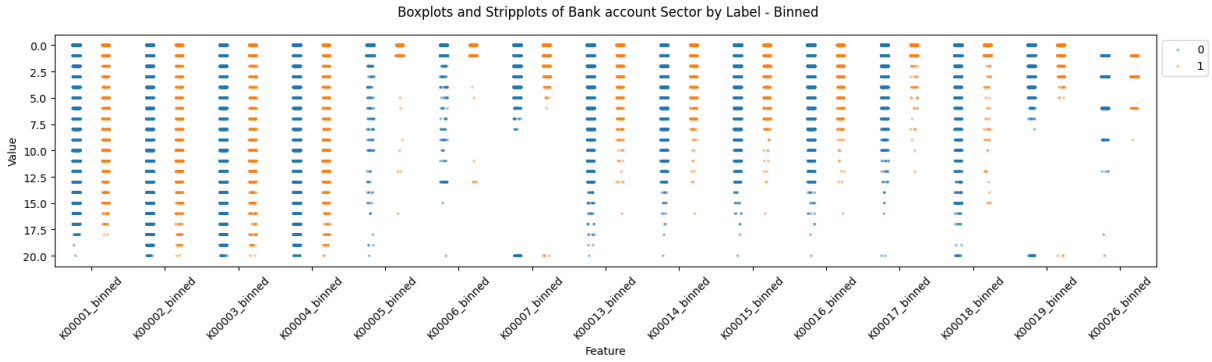

Figura 4 – Dados em *bins* sem reamostragem

Figura 5 – Dados em *bins* com reamostragem

Para algumas variáveis, a diferenciação de classes por meio dos dados agrupados se torna evidente. Esses resultados podem ser utilizados não só para a parte de visualização de dados,

mas também para seleção de variáveis. Posteriormente, o algoritmo *CatBoost*, que lida muito bem com dados categóricos, será avaliado utilizando esses dados. Em alguns casos, a variável pode ser visualmente distintas para cada cada classe, como vemos na figura 6 .

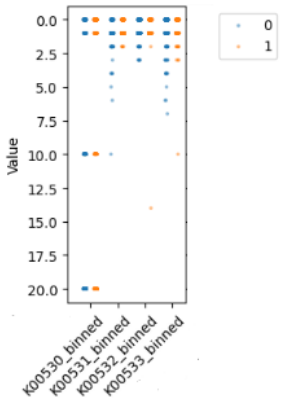

Figura 6 – Exemplo de distinção de classes utilizando dados agrupados.

4.1.8 Seleção de Variáveis

A seleção de variáveis é importante para o desenvolvimento de modelos de aprendizado de máquina, e várias abordagens podem ser adotadas para essa finalidade. Neste projeto, foi optado por realizar um compêndio dos métodos principais, a fim de avaliar sua influência nas métricas:

1. Filtro:
    - ANOVA (para variáveis contínuas)
    - *Chi-square* (para variáveis categóricas)
    - *Information Value* (IV)
    - *Mutual Information* (MI)
2. Métodos *"Embedded"*:
    - *Forward Elimination*
    - *Bi-directional Elimination*
3. Métodos *"Wrapper"*:
    - *Recursive Feature Elimination (RFE)*
    - *Boruta*

Figura 7 – ANOVA: Variáveis em vermelho são significativas com *p-value* < 0.05 e azul são as variáveis com maior *F-Score*. Variáveis com ambas as cores devem ser selecionadas.

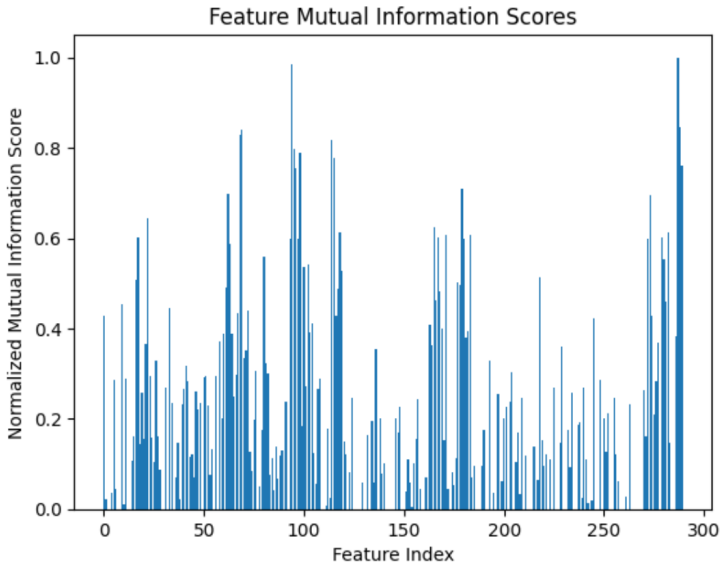

Figura 8 – *Mutual Information*: Para valores acima de 0.4 as variáveis são consideradas relevantes.

Por meio da utilização desses métodos, podemos selecionar variáveis, seja escolhendo um número máximo de variáveis que queremos manter ou escolhendo um valor limiar de cada método aplicado.

É importante ressaltar que o método *VarianceThreshold* foi empregado para identificar e eliminar características constantes ou com variação muito baixa. Além disso, variáveis com possíveis erros ou alta correlação foram previamente eliminadas.

Essa abordagem abrangente na seleção de variáveis permite explorar diferentes técnicas e considerar sua eficácia na otimização do modelo. Cada método tem suas vantagens e limitações, e a combinação de abordagens diversificadas pode resultar em uma seleção mais robusta e representativa das variáveis relevantes para o problema em questão, como visto na tabela 2.

Modelo	F1 Score	Precision	ROC AUC	Balanced		Tempo de Execução
				F1 Score	Accuracy	
Inicial	0.141	-	0.615	0.691	0.615	124 segundos
Selecionado	0.125	0.070	0.577	0.680	0.577	13 segundos

Tabela 2 – Comparação das métricas entre o modelo inicial e o modelo com variáveis selecionadas.

4.1.9 Seleção de Modelo

Com o propósito de aprender (na prática), extrair o máximo de informações dos dados e aplicar o conhecimento adquirido durante o curso, optamos por avaliar diversos algoritmos para selecionar o mais adequado ao problema em questão. Essa abordagem nos possibilita escolher o modelo que melhor se adapta às características específicas do nosso conjunto de dados, visando obter resultados mais otimizados nas métricas. Abaixo estão os algoritmos que foram avaliados:

- *Logistic Regression*
- *k-Nearest Neighbors (k-NN)*
- *Support Vector Machines (SVM)*
- *Neural Networks*
- *Naive Bayes*
- *Decision Trees*
- *Random Forest*
- *Gradient Boosting Machines (GBM)*
- *XGBoost*
- *LightGBM*
- *CatBoost*
- *HistGradientBoostingClassifier*

Cada algoritmo foi considerado com o intuito de identificar qual se destaca na resolução do problema, possibilitando a escolha do modelo mais eficaz para as características específicas do conjunto de dados em análise.

A escolha de incluir *CatBoost* e *HistGradientBoostingClassifier* no conjunto de algoritmos foi deliberada, visando uma análise comparativa das métricas nos *datasets* contínuos e discretizados por meio de *bins*. Essa abordagem permitirá avaliar como esses algoritmos lidam com diferentes representações dos dados e fornecer *insights* valiosos sobre seu desempenho em ambas as configurações.

Justificativa para a Seleção de Modelos

A escolha do algoritmo não é uma tarefa trivial, exigindo uma avaliação criteriosa para garantir uma seleção otimizada. Os principais pontos de atenção a serem considerados incluem: Complexidade, Tratamento de Dados Desbalanceados, Ajuste de Hiperparâmetros, Interpretabilidade, Escalabilidade, Tratamento de Dados Ausentes, Métodos de Ensemble, Sucesso Anterior

em Problemas Semelhantes, Limitações potenciais, Consideração de Overfitting e Métricas de Avaliação.

Adotando uma abordagem mais simplificada, especialmente diante de um conjunto de dados relativamente pequeno (com menos de 19 mil entradas), a avaliação de diversos algoritmos em sua forma bruta, sem ajuste de hiperparâmetros, e a análise das métricas resultantes, emerge como uma estratégia viável e plausível. Dessa forma, essa abordagem foi escolhida como o método condutor deste trabalho, visando uma compreensão inicial e abrangente do desempenho dos algoritmos diante do conjunto de dados disponível.

Para a seleção de um modelo considerando a escalabilidade, é fundamental levar em conta o tempo de execução médio. Especialmente quando se planeja colocar um modelo em produção, é vantajoso escolher um algoritmo com desempenho aceitável e menor tempo de execução. Os quatro melhores algoritmos foram: *HistGradientBoostingClassifier* (268 segundos), *Decision Trees* (269 segundos), *XGBoost* (272 segundos) e *LightGBM* (301 segundos). Por outro lado, os algoritmos que tiram maior tempo de execução médio foram: *Logistic Regression* (342 segundos), *k-Nearest Neighbors* (k-NN) (341 segundos), *Naive Bayes* (340 segundos) e *Neural Networks* (337 segundos).

A seleção de algoritmos iniciais, baseada em métricas preliminares sem ajuste de hiperparâmetros, fornece *insights* para direcionar a escolha do modelo. Na análise de risco de crédito, a métrica KS é frequentemente empregada devido à sua capacidade de avaliar a divergência entre as distribuições acumuladas das classes positiva e negativa. Um valor alto de KS indica uma maior capacidade de discriminação do modelo.

Além disso, a métrica AUC-ROC fornece uma medida da capacidade discriminativa global do modelo, sendo que um valor próximo a 0.5 sugere um comportamento quase aleatório, enquanto valores mais elevados indicam melhor desempenho na classificação.

A utilização da acurácia balanceada (*Balanced Accuracy*) como filtro adicional é particularmente relevante em cenários de desbalanceamento de classes. Esta métrica considera a distribuição das classes, sendo mais apropriada em casos nos quais não são aplicadas técnicas de reamostragem (*resampling*) ou ajustes nos pesos das classes (*class_weight*), proporcionando uma avaliação mais robusta do desempenho do modelo.

O resultado de cada um dos filtros e intersecção dos resultados combinados por ser observado na 3.

Tabela 3 – Algoritmos Selecionados com Base em Métricas Iniciais

Métrica AUC-ROC > 0.65	**Algoritmo**
AUC-ROC > 0.65	LightGBM
	XGBoost
	HistGradientBoostingClassifier
	Gradient Boosting Machines (GBM)
	Logistic Regression
	Random Forest
	Neural Networks
	k-Nearest Neighbors (k-NN)
Métrica KS > 0.3	**Algoritmo**
KS > 0.35	LightGBM
	Gradient Boosting Machines (GBM)
	HistGradientBoostingClassifier
	Random Forest
	XGBoost
Balanced Accuracy > 0.60	**Algoritmo**
Balanced Accuracy > 0.65	Gradient Boosting Machines (GBM)
	HistGradientBoostingClassifier
	XGBoost
	Random Forest
	LightGBM
Interseção das Listas	**Algoritmo**
Interseção	LightGBM
	XGBoost
	HistGradientBoostingClassifier
	Gradient Boosting Machines (GBM)
	Random Forest

Após uma análise das métricas iniciais, identificamos um conjunto de algoritmos que apresentaram desempenho robusto em diferentes aspectos, valores adequados e baixa variabilidade entre cada *fold*. Os algoritmos *LightGBM, XGBoost, HistGradientBoostingClassifier, Gradient Boosting Machines (GBM)*, e *Random Forest* demonstraram consistência, excedendo os limiares estabelecidos para métricas como AUC-ROC, KS e *Balanced Accuracy*. Este conjunto inicial de algoritmos será submetido a uma avaliação mais aprofundada, considerando o ajuste de hiperparâmetros e métricas resultantes após essa etapa. Restando apenas cinco algoritmos para a próxima fase de avaliação, esperamos refinar ainda mais a seleção para identificar o modelo mais adequado para o problema em questão.

A consideração de técnicas adicionais, como padronização, normalização e remoção de *outliers*, desempenha um papel crucial na otimização do desempenho dos algoritmos. Tais processos podem influenciar significativamente a convergência e estabilidade dos modelos, impactando diretamente nos resultados finais. No entanto, neste estágio, optamos por uma triagem inicial focada em métricas fundamentais, visando a seleção eficiente de um conjunto reduzido de algoritmos robustos. A próxima etapa envolverá uma análise mais detalhada, considerando esses

fatores adicionais, com o intuito de refinar ainda mais a escolha do modelo final.

4.1.10 Treinamento de Modelo

Para o treinamento do modelo, escolhemos utilizar a validação cruzada (CV) com 10 *folds*. Sempre que aplicável, configuramos o parâmetro *class_weight* como *'balanced'*, a fim de lidar de forma eficaz com o desbalanceamento das classes. É relevante destacar que, devido à necessidade de alguns algoritmos aceitarem apenas valores numéricos e não tolerarem valores ausentes (NaN), optamos por imputar esses valores por meio de estratégias numéricas. Essas decisões visam garantir uma avaliação robusta do desempenho do modelo, considerando a natureza específica do conjunto de dados e as características de cada algoritmo.

Na etapa anterior, constatamos que, para a maioria dos algoritmos, a imputação utilizando o valor zero proporcionou resultados superiores. Essa escolha é coerente com o contexto do nosso conjunto de dados, uma vez que o valor NaN indica a ausência de informação para a respectiva variável. Muitas variáveis apresentam valores NaN para a maioria da população, exemplificado pela variável "Renda proveniente de negócio próprio", onde a maioria dos indivíduos não possui sua própria empresa. Portanto, embora zero e NaN sejam valores distintos, neste caso específico, a imputação com zero se mostra mais apropriada em comparação com métodos como média ou mediana, que seriam arbitrários e inadequados para essa situação.

Sendo assim, para os melhores algoritmos selecionados previamente, optamos por avaliar se normalização e padronização terão efeitos relevantes na performance do modelo.

Para a normalização, utilizamos o *MinMaxScaler* com faixa entre [0,1], e [1,2]. Já para a padronização, empregamos o *StandardScaler*. Além disso, avaliou-se a opção de aplicar primeiramente o *StandardScaler*, seguido pelo *MinMaxScaler* ([0,1]). Essa escolha tem o objetivo de eliminar dados com valores negativos, uma vez que isso pode causar problemas para alguns algoritmos.

Os resultados obtidos foram significativos, pois tanto a normalização quanto a padronização geraram valores melhores das métricas, com menor variabilidade. De certa forma, isso foi inesperado, especialmente para os algoritmos de *gradient boosting*. Esperava-se que esses, até certo ponto, fossem capazes de lidar com *features*em diferentes escalas. Isso pode ser observado na figura 9. Nos blocos que parecem agrupados, foram aplicados os procedimentos descritos acima.

O valor da métrica de KS, nitidamente teve melhora quando comparado a não utilização de padronização ou normalização. Esse mesmo padrão pode ser observado para as métricas como AUC_ROC e *Balanced Accuracy*. Cada uma das caixas é um modelo distinto, os rótulos dos modelos foram removidos para não dificultar a legibilidade, um arquivo interativo, foi disponibilizado no repositório para melhor visualização e entendimento.

Vale citar que os cinco algoritmos alcançaram boas métricas com condições distintas.

Mas o GBM, apresentou tempo de execução, em média, 100 vezes maior para dados normalizados ou padronizados em relação aos demais. Sendo assim, ele pode ser

Figura 9 – Comparação entre modelos com e sem Padronização ou Normalização.

4.1.11 Avaliação de Modelo

Para avaliar o desempenho do modelo, empregamos uma ampla gama de métricas, incluindo acurácia, acurácia balanceada, F1 ponderado, precisão, recall, pontuação F1, coeficiente Kappa, coeficiente de correlação de Matthews (MCC), estatística KS, índice Gini e AUC (área sob a curva ROC), além do tempo de execução. É relevante destacar que, em análise de risco de crédito, as métricas mais comumente utilizadas são KS e Gini, proporcionando insights valiosos sobre a capacidade preditiva e a discriminação do modelo em relação às classes de interesse. A diversidade de métricas aplicadas visa fornecer uma avaliação abrangente do modelo em diferentes aspectos de seu desempenho.

Model Code	AR
LightGBM_None_None_MinMaxScaler [1, 2]	0.771
HistGradientBoostingClassifier_None_Zero Imputation_StandardScaler	0.767
HistGradientBoostingClassifier_None_Zero Imputation_MinMaxScaler[0,1]	0.759
HistGradientBoostingClassifier_Undersampling_None_StdScalerMinMaxS[0,1]	0.756
HistGradientBoostingClassifier_None_None_StandardScaler MinMaxScaler [0, 1]	0.755
HistGradientBoostingClassifier_None_None_StandardScaler MinMaxScaler [0, 1]	0.755
LightGBM_None_Zero Imputation_MinMaxScaler [1, 2]	0.754
XGBoost_Undersampling_None_StandardScaler MinMaxScaler [0, 1]	0.753
HistGradientBoostingClassifier_None_None_MinMaxScaler [1, 2]	0.753
XGBoost_None_Zero Imputation_MinMaxScaler [1, 2]	0.750

Tabela 4 – Top 10 Models and AUC_ROC (AR)

Model Code	KS
XGBoost_Undersampling_Zero Imputation_MinMaxScaler [1, 2]	0.441
LightGBM_None_None_MinMaxScaler [1, 2]	0.437
HistGradientBoostingClassifier_Undersampling_None_MinMaxScaler [0, 1]	0.436
HistGradientBoostingClassifier_None_Zero mputation_StdScalerMinMaxScaler[0,1]	0.430
LightGBM_None_Zero Imputation_MinMaxScaler [1, 2]	0.430
HistGradientBoostingClassifier_None_None_StandardScaler MinMaxScaler [0, 1]	0.427
XGBoost_Undersampling_None_StandardScaler MinMaxScaler [0, 1]	0.425
Random Forest_Undersampling_Zero Imputation_StandardScaler	0.421
HistGradientBoostingClassifier_None_Zero Imputation_StandardScaler	0.421
HistGradientBoostingClassifier_Undersampling_None_StdScalerMinMaxS[0,1]	0.419

Tabela 5 – Top 10 Models Ranked by KS

Model Code	BA
Random Forest_Undersampling_Zero Imputation_StandardScaler	0.704
Random Forest_Undersampling_Zero Imputation_StdScale MinMaxScaler[0,1]	0.703
HistGradientBoostingClassifier_Undersampling_None_MinMaxScaler [0, 1]	0.699
XGBoost_Undersampling_None_StandardScaler MinMaxScaler [0, 1]	0.697
XGBoost_Undersampling_None_StandardScaler MinMaxScaler [0, 1]	0.697
LightGBM_Undersampling_Zero Imputation_MinMaxScaler [1, 2]	0.696
XGBoost_Undersampling_Zero Imputation_MinMaxScaler [1, 2]	0.695
HistGradientBoostingClassifier_Undersampling_None_StdScalerMinMaxS[0,1]	0.694
HistGradientBoostingClassifier_Undersampling_None_MinMaxScaler [0, 1]	0.691
XGBoost_Undersampling_Zero Imputation_StandardScaler	0.691

Tabela 6 – Top 10 Models Ranked by Balanced Accuracy (BA)

Rank	Model Code
1	XGBoost_Undersampling_None_StandardScaler MinMaxScaler [0, 1]
2	Random Forest_Undersampling_Zero Imputation_StandardScaler
3	HistGradientBoostingClassifier_Undersampling_None_StdScalerMinMaxS[0,1]
4	Random Forest_Undersampling_Zero Imputation_StandardScaler
5	XGBoost_Undersampling_None_StandardScaler MinMaxScaler [0, 1]
6	Gradient Boosting Machines (GBM)_Oversampling_ZeroImputation_MinMaxS[1,2]
7	HistGradientBoostingClassifier_Undersampling_None_StdScalerMinMaxScaler[0,1]
8	Gradient Boosting Machines (GBM)_Oversampling_ZeroImputation_MinMaxS[1,2]
9	XGBoost_Undersampling_None_StandardScaler MinMaxScaler [0, 1]
10	Gradient Boosting Machines (GBM)_Oversampling_ZeroImputation_MinMaxS[0,1]

Tabela 7 – Top 10 Models

4.1.12 Ajuste de Hiperparâmetros

Com o objetivo de otimizar ainda mais o desempenho do modelo adotamos uma abordagem abrangente para otimização dos hiperparâmetros, utilizando três métodos distintos: *Grid Search*, *Random Search* e *Bayesian Search*. Essa estratégia visa explorar eficientemente o espaço de hiperparâmetros em busca das configurações ideais que maximizem o desempenho dos modelos.

O *Grid Search* realiza uma busca exaustiva em uma grade predefinida de valores, enquanto o *Random Search* explora aleatoriamente o espaço de busca. Já o *Bayesian Search* utiliza técnicas probabilísticas para direcionar a busca com base nas informações acumuladas durante as iterações.

Ao calcular e comparar métricas essenciais, como AUC_ROC, KS, *Accuracy, Balanced Accuracy*, F1 e F1 Ponderado, para cada abordagem de busca, buscamos identificar a configuração de hiperparâmetros que melhor se adequa ao nosso conjunto de dados, otimizando assim o desempenho preditivo dos modelos. Essa análise é crucial para garantir robustez e generalização dos modelos a serem empregados na etapa de avaliação e validação.

Aplicando as três técnicas no melhor modelo encontrado previamente, algoritmo XGBoost, podemos verificar primeiramente que o *Grid Search* apresenta pouca variabilidade para os parâmetros definidos, reduzindo a chance de encontrar um mínimo global, isso pode ser confirmado pela figura 10. Além disso, o *Grid Search* demanda um número maior de iterações para atingir métricas semelhantes, figura 11.

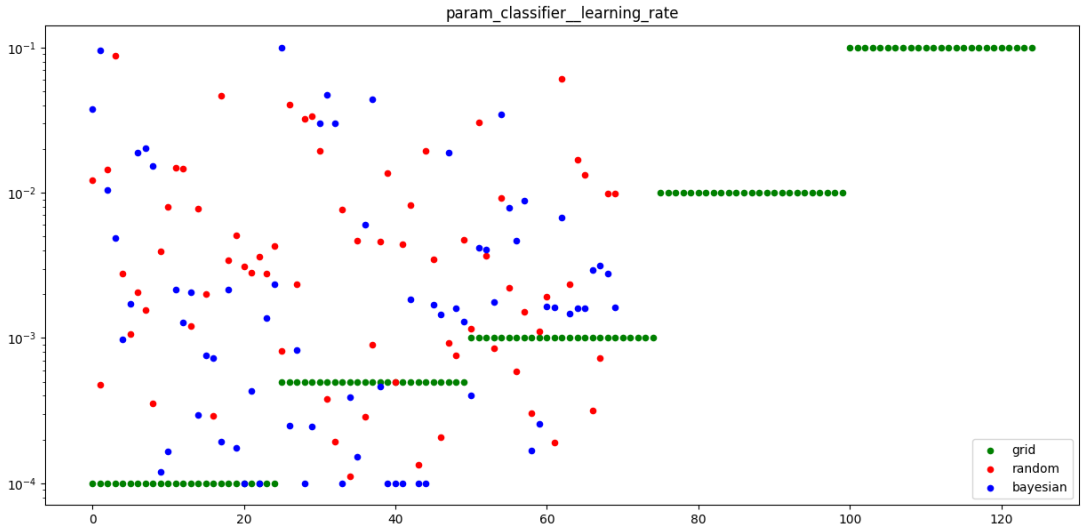

Figura 10 – Comparação entre *Grid, Random e Bayesian Search* pelo número de iterações e variação de *learning rate.*

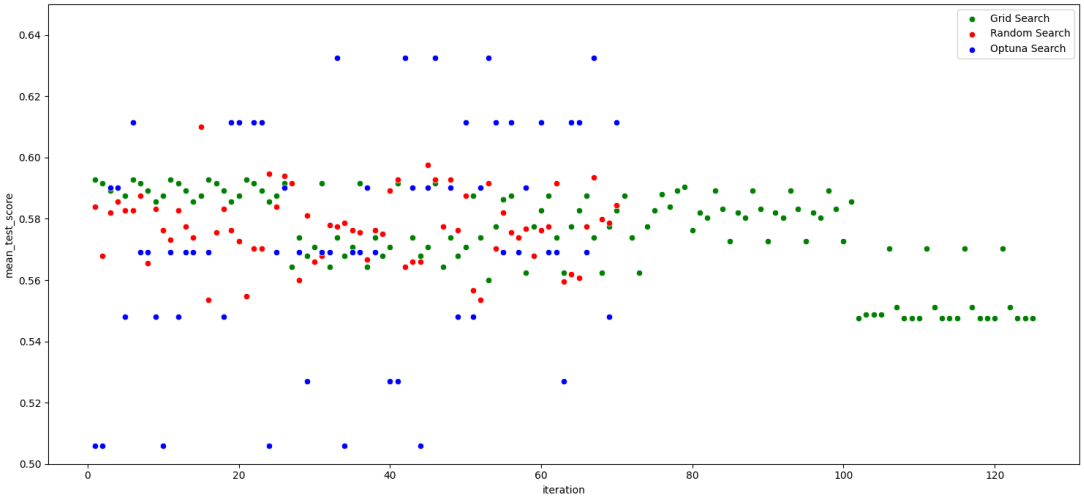

Figura 11 – Valor da métrica AUC_ROC para *Grid, Random e Bayesian Search* pelo número de iterações.

4.1.13 Interpretação de Modelo

Na relação dos algoritmos avaliados, é possível categorizá-los em termos de interpretabilidade, diferenciando entre modelos interpretais e modelos caixa-preta (*blackbox*).

Modelos Interpretáveis:

1. *Logistic Regression*
2. *k-Nearest Neighbors (k-NN)*
3. *Naive Bayes*
4. *Decision Trees*

Esses modelos são considerados interpretáveis, pois a relação entre as variáveis de entrada e a saída pode ser facilmente compreendida. *Logistic Regression*, *k-Nearest Neighbors*, *Naive Bayes* e *Decision Trees* geralmente oferecem uma visão mais clara das decisões tomadas pelo modelo.

Modelos Caixa-Preta (*Blackbox*):

1. *Support Vector Machines (SVM)*
2. *Neural Networks*
3. *Random Forest*
4. *Gradient Boosting Machines (GBM)*
5. *XGBoost*

6. *LightGBM*
7. *CatBoost*
8. *HistGradientBoostingClassifier*

Esses modelos são considerados caixa-preta, pois a relação entre as variáveis de entrada e a saída não é facilmente interpretável. Métodos como *Support Vector Machines*, Redes Neurais, *Random Forest* e *Gradient Boosting Machines* tendem a ser mais complexos e, portanto, menos transparentes em termos de interpretabilidade.

É importante considerar a interpretabilidade do modelo com base nos requisitos específicos do problema e nas necessidades práticas. Modelos interpretais são frequentemente preferíveis em situações onde a explicação das decisões do modelo é crítica, como em casos de análise de risco de crédito. Para alguns algoritmos temos a opção de verificar *feature importance*, que de modo geral, indica o quando cada variavél contribui para o modelo. Um exemplo pode ser visto na figura 12.

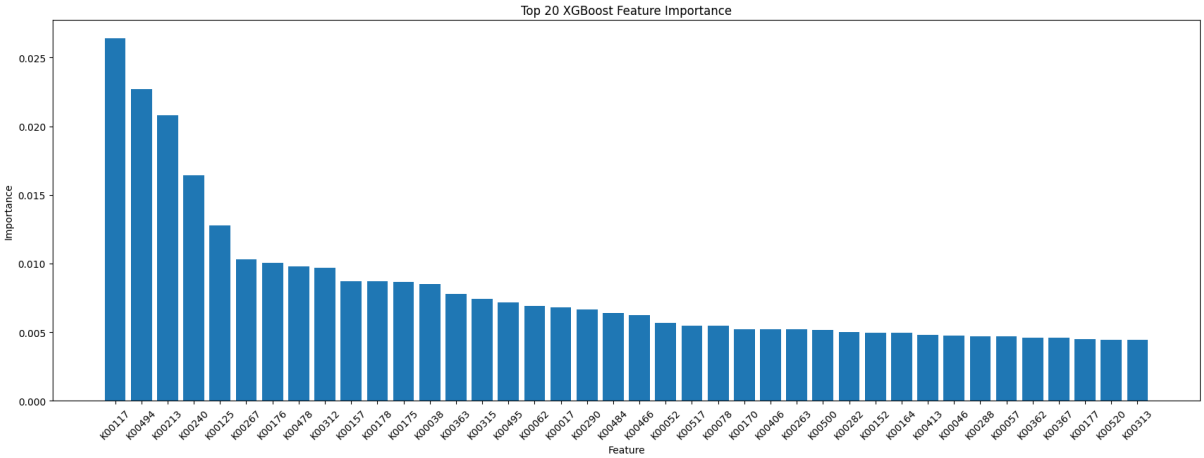

Figura 12 – *XGBoost Feature importance.*

Técnicas como os valores *SHAP* (*SHapley Additive exPlanations*) nos oferecem insights sobre como cada característica contribui individualmente para as previsões do modelo, tornando as *blackbox* um pouco "menos escuras" para modelos de *boosting*, por exemplo. Para modelo de *deep learning*, até menos esse tipo de interpretação costuma ser mais difícil.

4.1.14 Implantação de Modelo

A etapa de implementação do modelo em produção irá variar de acordo com cada ambiente. Algumas medidas e precauções podem ser adotadas de forma geral, sendo as mais relevantes:

- **Configuração do Ambiente de Produção:** Para configurar o ambiente de produção de forma eficiente, adotamos práticas específicas para garantir a estabilidade e o desempenho

do modelo. Utilizamos a ferramenta *pip freeze* para gerar um arquivo *requirements.txt*, contendo as versões exatas das bibliotecas utilizadas durante o desenvolvimento. Isso previne possíveis erros e conflitos de versões, assegurando a consistência do ambiente. Além disso, optamos por armazenar o modelo em um arquivo do tipo pickle para facilitar a carga e descarga, proporcionando maior agilidade na integração do modelo no ambiente de produção. A utilização de containers, como Docker, é recomendada para garantir a robustez e reprodutibilidade do ambiente em diferentes configurações, facilitando a escalabilidade e a manutenção.

- **Integração com Sistemas Existentes:** Na integração do modelo com sistemas existentes, adotamos uma abordagem prática para garantir uma implementação eficaz. Desenvolvemos interfaces customizadas para facilitar a comunicação entre o modelo e outros componentes do sistema, garantindo uma integração suave. Para facilitar a manutenção, utilizamos logs detalhados para monitorar a entrada e saída do modelo, permitindo uma rápida identificação de eventuais problemas. Essa abordagem prática visa minimizar os impactos operacionais e otimizar a contribuição do modelo no contexto mais amplo do ambiente de produção.

- **Na prática:** Optamos por integrar o aplicativo Streamlit com o pipeline de produção. Essa integração visa assegurar que os dados do usuário sejam processados de maneira eficiente e que as previsões sejam geradas em tempo real. O uso do Streamlit proporciona uma interface amigável e interativa para os usuários, tornando a experiência mais acessível e facilitando o acesso às funcionalidades do modelo.

Essa abordagem de integração permite uma transição suave do modelo do ambiente de desenvolvimento para um ambiente de produção, garantindo que o modelo treinado seja facilmente acessível e capaz de fornecer previsões em tempo real conforme necessário

4.1.15 Monitoramento e Manutenção de Modelo

A manutenção e monitoramento contínuo do modelo são aspectos críticos para garantir sua eficácia ao longo do tempo. Mesmo que o modelo ainda não tenha sido oficialmente implantado em produção, é fundamental antecipar práticas que garantam a robustez e relevância contínua. Aqui estão algumas sugestões teóricas para proceder:

Verificação de Data Drift:

- **Estabelecer uma Baseline:** Antes da implantação, registre uma "linha de base"para as características e distribuições dos dados de treinamento. Isso servirá como referência para futuras comparações.

- **Monitoramento Contínuo:** Mesmo sem a implementação oficial, inicie a prática de monitorar regularmente os dados de entrada para detectar qualquer desvio significativo em relação à linha de base. Isso pode ser feito por meio de visualizações gráficas ou métricas específicas.

Suposições e Mudanças de Comportamento:

- **Definir Suposições:** Documente claramente quais suposições estão embutidas no modelo durante o treinamento. Isso pode incluir expectativas sobre a distribuição dos dados, relacionamentos entre variáveis, etc.
- **Monitoramento de Suposições:** Regularmente verifique se as suposições feitas durante o treinamento do modelo ainda são válidas. Se ocorrerem mudanças significativas nos dados que violem essas suposições, isso pode impactar a confiabilidade do modelo.

Manutenção Proativa:

- **Atualização de Dados:** Mesmo que o modelo não esteja em produção, considere atualizar periodicamente os dados de treinamento. Isso pode ser especialmente útil se novos dados estiverem disponíveis e puderem melhorar a generalização do modelo.
- **Reavaliação de Variáveis:** Caso novas variáveis relevantes se tornem disponíveis ou se a importância das variáveis existentes mudar, reavalie a inclusão ou exclusão delas no modelo.

Documentação e Comunicação:

- **Manter Documentação:** Mantenha uma documentação clara sobre as mudanças realizadas, atualizações de dados e qualquer ajuste feito no modelo.
- **Comunicação Interdisciplinar:** Estabeleça uma comunicação eficaz entre as equipes técnica e de negócios. Isso garante que as alterações no modelo se alinhem com as necessidades e objetivos do negócio.

Ao seguir essas práticas, mesmo antes da implementação oficial, o cientista de dados estará preparado para enfrentar desafios futuros, garantindo a relevância e eficácia contínua do modelo de risco de crédito.

Estabilidade em função do tempo:

Na prática, a estabilidade temporal é crucial para assegurar que um modelo de risco de crédito mantenha sua eficácia ao longo do tempo. Para avaliar essa estabilidade, é fundamental

utilizar métricas específicas que possam detectar mudanças nas características dos dados ao longo de diferentes períodos. O acompanhamento do desempenho do modelo ao longo do tempo pode envolver métricas como a AUC (*área sob a curva ROC*), KS (*estatística de Kolmogorov-Smirnov*) e Gini, que são sensíveis a alterações na capacidade discriminativa do modelo.

Além disso, é recomendável utilizar visualizações gráficas, como gráficos de curva ROC ao longo do tempo, para uma compreensão mais holística da estabilidade. A identificação precoce de qualquer degradação no desempenho do modelo permite ajustes proativos, garantindo que o modelo continue sendo uma ferramenta confiável para a tomada de decisões em relação ao risco de crédito.

Finalmente, é relevante ressaltar que o processo de modelagem é intrinsecamente complexo e não segue uma sequência rígida de etapas. Em vez disso, envolve a formulação e teste contínuo de hipóteses, ajustes e refinamentos iterativos. Esse ciclo de melhoria contínua é uma parte essencial do acompanhamento do modelo ao longo de sua vida útil, visando garantir não apenas a confiabilidade, mas também a otimização contínua dos resultados enquanto o modelo está em produção. A flexibilidade para se adaptar a novos dados, condições de mercado e requisitos de negócios é fundamental para a eficácia a longo prazo do modelo de risco de crédito.

4.2 Conclusões

Este trabalho busca proporcionar uma introdução prática e básica ao desenvolvimento de modelos de concessão de crédito no contexto brasileiro, especialmente direcionado a trabalhadores autônomos e empresários com rendas informais e variáveis. Reconhecemos as limitações da análise de crédito tradicional nesse cenário e, por isso, exploramos alternativas utilizando dados do *Open Finance*.

A metodologia apresentada tem como objetivo desenvolver um modelo utilizando técnicas de aprendizado de máquina, visando ampliar a concessão de crédito e promover maior inclusão financeira para os profissionais mencionados. No entanto, é crucial destacar que este trabalho é um tutorial básico e prático. Não aprofundamos em técnicas específicas nem discutimos detalhadamente cada uma delas. A intenção foi fornecer um ferramental genérico para o desenvolvimento de um modelo ponta a ponta, incluindo técnicas valiosas nesse processo. Uma das etapas mais relevantes da metodologia CRISP, entendimento do negocio, não foi abordada.

É importante salientar que o trabalho apresenta limitações, como a dependência da disponibilidade e qualidade dos dados do *Open Finance*, além da necessidade de avaliação de aspectos éticos relacionados à privacidade dos dados dos clientes. Futuras pesquisas podem aprofundar essas questões e explorar outras abordagens para aprimorar a concessão de crédito a trabalhadores autônomos e empresários com rendas informais e variáveis.

Neste momento, os resultados são preliminares, e há ainda muito a ser feito no pré-

processamento dos dados e na modelagem. Este trabalho proporciona uma excelente oportunidade para aprendizado contínuo e aprimoramento das habilidades nesse campo em constante evolução.

Diante do que foi realizado deixo alguns questionamentos (ou tarefas, para aqueles que sempre buscam aprender mais):

- O *dataframe binned* foi gerado, quais seriam os resultados esperados se utilizarmos *XGBoost* diretamente nele? E caso utilizarmos *CatBoost*? Mais de um método de *feature selection* foi apresentado, porém
- O *dataframe* utilizado nas secções de resultados foi o bruto, só variáveis com erros graves foram eliminadas. Qual deve ser o melhor dos métodos e o que se espera do valor das métricas obtidas utilizando as variáveis selecionadas?
- O algoritmo NaiveBayes tem mais de um tipo. Por qual motivo foi utilizado *GaussianNB*? Outros tipos seriam mais apropriados?

Não esqueçam de ver o *notebook* e divirtam-se!

REFERÊNCIAS

ARAUJO, F. Initial steps towards a central bank digital currency by the Central Bank of Brazil. In: SETTLEMENTS, B. for I. (Ed.). **CBDCs in emerging market economies**. Bank for International Settlements, 2022, (BIS Papers chapters, v. 123). p. 31–37. Disponível em: <https://ideas.repec.org/h/bis/bisbpc/123-03.html>. Citado na página 24.

BARBOSA, K. Comment on: "why is bank credit in brazil the most expensive in the world?". **Brazilian Review of Finance**, Fundacao Getulio Vargas, v. 18, n. 4, p. 23–28, nov. 2020. Disponível em: <https://doi.org/10.12660/rbfin.v18n4.2020.82746>. Citado na página 22.

BAZARBASH, M. FinTech in financial inclusion: Machine learning applications in assessing credit risk. **IMF Working Papers**, International Monetary Fund (IMF), v. 2019, n. 109, p. 1, maio 2019. Disponível em: <https://doi.org/10.5089/9781498314428.001>. Citado na página 22.

BHATORE, S.; MOHAN, L.; REDDY, Y. R. Machine learning techniques for credit risk evaluation: a systematic literature review. **Journal of Banking and Financial Technology**, Springer Science and Business Media LLC, v. 4, n. 1, p. 111–138, abr. 2020. Disponível em: <https://doi.org/10.1007/s42786-020-00020-3>. Citado na página 22.

HASAN, M. M.; POPP, J.; OLÁH, J. Current landscape and influence of big data on finance. **Journal of Big Data**, Springer Science and Business Media LLC, v. 7, n. 1, mar. 2020. Disponível em: <https://doi.org/10.1186/s40537-020-00291-z>. Citado na página 22.

HJELKREM, L. O.; LANGE, P. E. de; NESSET, E. The value of open banking data for application credit scoring: Case study of a norwegian bank. **Journal of Risk and Financial Management**, MDPI AG, v. 15, n. 12, p. 597, dez. 2022. Disponível em: <https://doi.org/10.3390/jrfm15120597>. Citado na página 23.

LAPLANTE, P.; KSHETRI, N. Open banking: Definition and description. **Computer**, Institute of Electrical and Electronics Engineers (IEEE), v. 54, n. 10, p. 122–128, out. 2021. Disponível em: <https://doi.org/10.1109/mc.2021.3055909>. Citado na página 23.

LOUZADA, F.; ARA, A.; FERNANDES, G. B. Classification methods applied to credit scoring: systematic review and overall comparison. **Surveys in Operations Research and Management Science**, 2016. Citado na página 22.

OMARINI, A. Banks and fintechs: How to develop a digital open banking approach for the bank's future. **International Business Research**, v. 11, p. 23, 08 2018. Citado na página 23.

SHI, S.; TSE, R.; LUO, W.; D'ADDONA, S.; PAU, G. Machine learning-driven credit risk: a systemic review. **Neural Computing and Applications**, Springer Science and Business Media LLC, v. 34, n. 17, p. 14327–14339, jul. 2022. Disponível em: <https://doi.org/10.1007/s00521-022-07472-2>. Citado na página 22.