Aplicação de Técnicas de Otimização de Hiperparâmetros em Modelos de *Machine Learning* na Tarefa de Classificar Bons e Maus Clientes.

Armando Pereira Pontes Júnior
Programa de Pós-graduação em Engenharia da Computação
Universidade de Pernambuco - UPE
Recife-PE, Brasil
appj@ecomp.poli.br

Roberta Andrade de A. Fagundes
Departamento de Engenharia da Computação
Universidade de Pernambuco - UPE
Recife-PE, Brasil
roberta.fagundes@upe.br

***Resumo*— A indústria financeira necessita de soluções rápidas, de menor custo e mais assertivas na classificação de risco de crédito. Métodos de *machine learning* estão cada vez mais sendo incorporados para executar essa tarefa. Além disso, técnicas de otimização de hiperparâmetros podem melhorar o desempenho de classificadores. Neste trabalho foram aplicadas duas importantes técnicas de otimização com cinco classificadores distintos, são eles: *Decision Tree* (DT), *Random Forest* (RF), *MultiLayer Perceptron* (MLP), *eXtreme Gradient Boost* (XGBoost) e *Light Gradient Boost Machine* (LGBM). Para medir o desempenho dos classificadores foram utilizadas as seguintes métricas: *Accuracy*, *Precison*, *Recall* e *F1-Score*. Os modelos foram treinados e testados com informações de duas bases de dados. Inicialmente foram utilizadas as configurações padrões dos hiperparâmetros, e, posteriormente, foram aplicadas as otimizações *bayesiana* e por *Particle Swarm Optimization* (PSO) para alcançar melhores resultados. A otimização *bayesiana* apresentou melhorias nas métricas de todos os modelos, com destaque para o *Recall*,chegando a uma assertividade de 91,4% no classificador MLP. Porém, a otimização por PSO não apresentou melhoria no desempenho.**

Keywords—credit scoring, machine learning, Bayesian optimization, PSO.

I. INTRODUÇÃO

O nível de atividade econômica de um país é essencial para o crescimento, para geração de emprego e renda, como também, estabelece melhorias nas condições socioeconômicas da população. A atividade produtiva é altamente dependente, por um lado, dos investimentos realizados pelas empresas na produção de bens e serviços, e por outro lado, pelo consumo, que tem sua maior fatia realizada pelas famílias [1].

Boa parte deste consumo só consegue ser realizado com auxílios de empréstimos financeiros/crédito. As instituições financeiras atuam como agentes de intermediação no mercado captando recursos juntos alguns investidores pessoas físicas, empresas e entidades públicas que possuem reservas excedentes e as canalizam àqueles que necessitam de recursos extras para financiarem seus déficits orçamentários, ou para efetuarem aquisições novos bens/serviços.

Assim, o aumento da quantidade de empréstimos traduz também num incremento da economia, dinamizando antigos e novos negócios [2]. Entretanto, a cada nova proposta de concessão de empréstimo, demandam-se novas avaliações de risco de crédito por parte das instituições credoras. Essas avaliações levam em consideração à análise sobre as mais diversas características do propenso mutuário, necessita de tempo e envolvem custos elevados para sua conclusão. Uma classificação errada suscita em perdas financeiras para as instituições envolvidas [3].

É neste contexto de agilizar e automatizar as avaliações de risco de crédito que se desenvolveu uma importante ferramenta computacional: o *credit scoring*. No início, as avaliações de risco de crédito estavam muito voltadas aos conhecimentos tácitos dos analistas de crédito, suas percepções e crenças sobre os demandantes por crédito [4]. Posteriormente, foram sendo utilizadas técnicas probabilísticas paramétricas, como o uso, principalmente, de análise de discriminante (AD) e regressão logística (RL). Porém, essas duas técnicas partem do pressuposto que os dados se comportam de forma regular, seguindo uma distribuição conhecida e bem comportada [5].

No início da década de 90, com o crescimento e o desenvolvimento da inteligência artificial (IA), especificamente métodos de *machine learning*, e com o aumento do poder de processamento dos computadores, que passavam a ser massivamente incorporados nas instituições financeiras, as avaliações de risco de crédito começaram a ser automatizadas. O conhecimento advindo da IA passou também a ser estudado e aplicados nas pesquisas da indústria financeira [6]. Algumas das vantagens de aplicar métodos de *machine learning* é que eles trazem mais rapidez nas análises de crédito, como também, maior assertividade [7]. Ao contrário dos métodos estatísticos, os principais métodos de *machine learning* não exigem que haja um relacionamento conhecido entre as variáveis, visto que estes métodos tentam extrair conhecimentos e padrões a partir dos dados.

Vários estudos na literatura demonstram o benefício na aplicação de métodos de *machine learning* para construção de modelos de *credit scoring* [8]. Estas pesquisas incluem deste o uso de Rede Neurais (NN) [9][10], *Random Forest* (RF) [11], *Support Vector Machine* (SVM) [12] e métodos de *ensemble learning (boosting e bagging)* [13] [14].

Especificamente em se tratando de *ensemble learning*, estudos demonstram que estas técnicas respondem de forma mais robustas e precisas, com alto poder de generalização e sendo mais assertivos que os modelos simples com um único classificador [15]. Inclusive, a aplicação de técnicas de *ensemble learning* visam minimizar a variância e o viés que os classificadores individuais (também referenciados na língua inglesa como *weak leaning*) demonstram ter nas suas classificações [16].

Assim, este trabalho contribui com esta área de pesquisa avaliando duas técnicas de otimização de hiperparâmetros,

com intuito de incrementar as performances dos classificadores. Os resultados obtidos auxiliam os analistas de crédito na gestão e mensuração de risco de suas carteiras de clientes.

Este artigo está organizado e dividido em cinco partes. Na seção II descrevem-se alguns trabalhos recentes relacionados com à área de *credit scoring*.Na seção III está descrita a metodologia aplicada nessa pesquisa. A seção IV encontram-se os resultados obtidos, e ficou reservado para seção V a conclusão e a indicações de possíveis trabalhos futuros.

II. TRABALHOS RELACIONADOS

Uma grande quantidade de pesquisas vêm sendo desenvolvidas ao longo dos anos na área de risco de crédito. Estes estudos se mostraram bem diversificados, pesquisando e aplicando vários métodos de *machine learning*, uso de diversas bases de dados, e, ainda, avaliando diversos problemas reais (fraudes em operações de cartão de crédito, risco de crédito em empréstimos para pessoas físicas e jurídicas, avaliação de risco soberano, entre outros). Alguns trabalhos importantes guardam similaridade com essa pesquisa, a saber:

A pesquisa desenvolvida em [17] propôs o uso de um SVM e uma otimização por PSO, criando um modelo que foi chamado de PSO-SVM. A acurácia deste modelo combinado foi comparada com o próprio classificador SVM para demonstrar um ganho de performance com a otimização.

Já no estudo realizado por [18] foram aplicados cinco conhecidos métodos de *machine learning*, que utilizaram os algoritmos *Logistic Regression* (RL), *Decision Tree* (DT), *AdaBoost*, *K-Nearest Neighbor* (KNN) e *Support Vector Machine* (SVM) para avaliar a tarefa de detectar operações fraudulentas com cartão de crédito. Além disso, foram utilizadas técnicas de seleção de atributos (*features selection*) e otimização de hiperparâmetros na tentativa de melhorar as performances das classificações. O resultado obtido foi que o modelo SVM com auxílio de análise de variância (ANOVA), na função de eliminar atributos irrelevantes, obteve os melhores resultados nas métricas de *F1-Score*, *Precision e Recall*.

No trabalho desenvolvido em [19] propôs o uso de PSO para otimização de três classificadores individuais (árvore de decisão, rede neural artificial e KNN). Foram utilizadas diferentes populações de partículas (20, 50, 100 e 150). O resultado mostrou que o modelo com 100 partículas foi o que obteve a melhor performance. Seu resultado ainda foi comparado com seis outros modelos de *ensemble* com performances superiores na maioria dos casos.

Por fim, no estudo conduzido em [20] comparou algoritmos de *ensemble* (*Random Forest, AdaBoost, XGBoost, LGBM e Stacking*) com as performances de cinco classificadores individuais (*Neural Network, Decision Tree, Logistic Regression, Naïves Bayes e Suport Vector Machine*), usando as métricas *accuracy* (ACC), *area under the curve* (AUC), Kolmogorov-Smirnov (KS), Brier Score (BS) e tempo de execução. O resultado mostrou que os modelos de *ensembles* apresentaram performances melhores que os classificadores individuais, com exceção para o *Adaboost*. Além disso, a *Random Forest* foi o modelo que apresentou os melhores resultados nas cinco métricas, seguido de *XGBoost* e *LGBM*.

III. METODOLOGIA DE PESQUISA

A. *Base de dados*

Neste trabalho serão utilizadas duas bases de dados para aplicação das técnicas de otimização dos hiperparâmetros dos métodos de *machine learning* utilizados, são elas:

1) A *German Credit Data* é uma base que se encontra disponível no site da UCI *machine learning repository*. A base de dados contém 1.000 registros, sendo que para cada registro estão vinculadas 21 características, que incluem informações pessoais, patrimoniais, emprego e renda e informações sobre a própria operação. A Tabela 1 descreve os atributos encontrados na base de dados da *German Credit*:

Tabela 1. Dicionário dos Atributos – *German Credit Data*

No	Atributo	Tipo	Descrição
1	default	categórica	Código 0 = adimplente; código 1 = inadimplente
2	conta_corrente	categórica	Saldo na conta corrente (faixas de valores)
3	prazo_emprestimo_meses	numérica	Prazo em meses
4	historico_credito	categórica	Histórico de atraso de outros empréstimo
5	proposito_emprestimo	categórica	Finalidade da operação de crédito
6	valor_emprestimo	numérica	Valor da concessão do empréstimo
7	reserva_cc	categórica	Se há saldo na conta corrente
8	tempo_emprego_atual	categórica	Tempo que se encontra empregado atualmente
9	taxa_comp_salario	categórica	Comprometimento (em %) da prestação no salário
10	sexo_est_civil	categórica	Combinação entre sexo (masc/fem) e estado civil (solteiro/casado/...)
11	outros_fiadores	categórica	Se há fiadores na operação de crédito
12	anos_residencia_atual	categórica	Tempo que se encontra na atual residência
13	propriedade	categórica	Existência de outros bens
14	idade	numérica	Idade do cliente
15	outros_planos_financiamento	categórica	Existência de outros financiamentos (bancário ou não)
16	tipo_residencia	categórica	Própria, alugada, sem custo
17	n_creditos_banco	categórica	Quantidade de outros créditos no banco
18	status_emprego	categórica	Tipo de qualificação de emprego
19	n_dependentes	categórica	Se há dependentes
20	telefone	categórica	Se possui telefone
21	trabalhador_estrangeiro	categórica	Se é um trabalhador de outro país

Fonte: próprio autor

A variável '*default*' é o atributo que possui os rótulos da classificação binária: adimplente versus inadimplente. Essa variável é a saída (*output*) que cada algoritmo de classificação utilizará para fazer suas predições.

2) A segunda base de dados é de propriedade de uma cooperativa de crédito com forte atuação na região sul do Brasil. A base de dados contém 40.320 registros de operações de crédito que ocorreram entre o período de 2015 a 2017. Para cada registro estão vinculadas 21 características que incluem informações pessoais, patrimoniais, emprego e renda e informações sobre a própria operação. As operações dizem respeito a comercialização de 3 produtos de crédito pessoal: cheque especial (CP), crédito direto ao consumidor (CDC) e cartão de crédito (Cartão). A Tabela 2 descreve o dicionário dos atributos encontrados na base de dados da Cooperativa:

Tabela 2. Dicionário dos Atributos – Cooperativa

No	Atributo	Tipo	Descrição
1	ID	numérica	Identificação da operação
2	Setor_Atividade	categórica	Setor de atividade que cliente trabalha (diversos valores)
3	DT_NASCTO	data	Data nascimento no formato ano-mês-dia
4	DATA_PROPOSTA	data	Data da proposta no formato ano-mês-dia
5	ESTADO_CIVIL	categórica	Casado, solteiro, viúvo, divorciado, União estável, Separado Judicialmente
6	SEXO	categórica	Masculino, Feminino
7	GRAUINSTRUCAO	categórica	Ensino Fundamental, Ensino Médio, Ensino Superior, superior Incompleto, Não Afalbetizado
8	CEP	numérica	Código do endereço postal do cliente
9	UF_ENDERECO	categórica	Estado de residência do cliente - 2 letras
10	RESDDD	numérica	Código do DDD do telefone
11	RESTIPOFONE1	categórica	Se o número de telefone é residencial ou celular
12	RESTIPORESIDENCIAL	categórica	Alugada sozinho, Própria quitada, Própria Financiada, Funcional, Parentes, Alugada Parcial, Outros
13	CLASSEPROFISSIONAL	categórica	Assalariado, Aposentado, Autônomo, Empresário, Profissional Liberal
14	RENDA	numérica	Valor da renda principal do cliente - em reais
15	OUTRAS_RENDAS	numérica	Valor de outras rendas do cliente - em reais
16	DEPENDENTES	numérica	Quantidade de dependentes
17	RENDA_CONJUGE	numérica	Valor da renda do(a) cônjuge
18	PRODUTO_UNICO	categórica	Tipo de produto contratado: Cheque Especial, CDC ou Cartão de Crédito
19	IDADE	numérica	Idade do cliente - em ano(s)
20	TEMPO_EMPREGO	numérica	Tempo, em ano(s), no atual emprego
21	ALVO	categórica	código 0 = adimplente; código 1 = inadimplente

Fonte: próprio autor

Originalmente as bases possuíam atributos categóricos, numéricos discretos e numéricos contínuos. Os valores numéricos foram sumarizados em gráficos de *boxplot* onde é possível visualizar suas distribuições, identificando valores máximos e mínimos, mediana e distribuição por quartis. As Figuras 1 e 2 descrevem os gráficos de *boxplot* nas duas bases de dados.

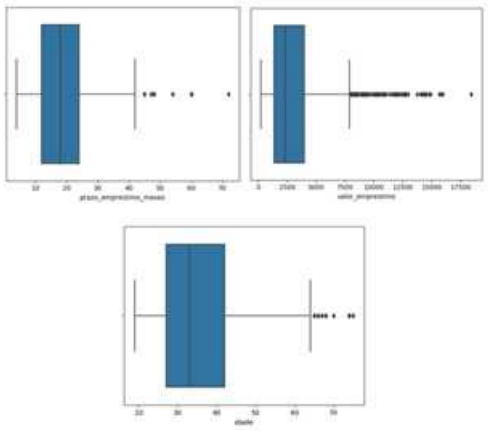

Figura 1. *Boxplot* para as variáveis numéricas na *German Credit Data*

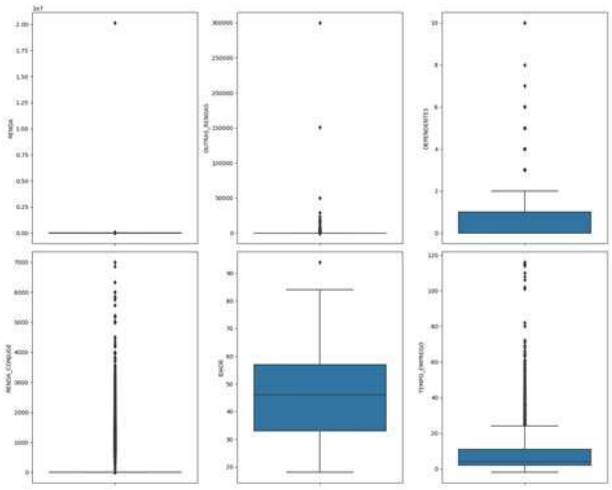

Figura 2. *Boxplot* para as variáveis numéricas na base da Cooperativa

Esta descrição gráfica traz muitas informações interessantes, das quais, uma é a sinalização da existência de valores discrepantes que possam prejudicar o aprendizado dos classificadores. Estes valores serão tratados no pré-processamento como *outliers.*

B. FLUXO

O fluxo das etapas executadas para alcançar os resultados propostos nesta trabalho foi apresentada na Figura 3. O primeiro momento foi realizada o pré-processamento dos dados das bases, posteriormente, foram feitas as análises exploratórias dos dados para entender suas distribuições, a existência de covariância entre os atributos, a identificação de dados ausentes e discrepantes. Logo em seguida, foi executado o balanceamento das classes nas basesde dados. A *German Credit* é uma base com 70% dos dados pertencentes ao rótulo adimplente. A base da Cooperativa, 64% dos dados formam a classe majoritária.

O terceiro passo foi efetuar a classificação com o auxílio dos 5 modelos de *machine learning* . Esta classificação inicial foi feita com uso de 4 métricas: *Accuracy*, *Precision, Recalle F1-Score* (explicadas na seção IV). A última tarefa a ser executada foi a aplicação de duas conhecidas técnicas otimização de hiperparâmetros em cada um dos métodos de *machine learning*, onde foi possível verificar o quanto a otimização contribuiu na performance dos modelos escolhidos.

Figura 3. Fluxo das etapas da metodologia

- Pré-processamento: é um dos passos mais importante na atividade de criação de uma máquina preditiva. Neste contexto, explorar e entender cada variável, verificar a existência de dados ausentes (*missing values*) ou de dados discrepantes (*outliers*). Identificar se há correlações entre as variáveis fará com que o os modelos preditivos fiquem mais rápidos, com menor custo de processamento e com melhor acurácia. Ao identificar informações que atrapalham o aprendizado do modelo proposto (*outliers*, dados redundantes, informação ausente), evita-se também que o modelo sofra sobreajustes (*overfiting*) aos dados de treino e não consigam, posteriormente, umaboa generalização sobre os novos dados. A base de crédito alemã é uma

base já consolidada e muito utilizada por diversos pesquisadores, ela não apresenta valores ausentes ou *outliers*. O único ajuste feito no pré-processamento foi o balanceamento da base para que as duas categorias da variável *default* ficassem na mesma proporção. Já a base de dados da Cooperativa precisou passar por ajustes nos seguintes pontos: foram retirados os atributos ID, Setor Atividade (havia muitos dados ausentes, além das categorias estarem com nomenclaturas diferentes para a mesma profissão) e CEP. Além disso, foi preciso ajustar as variáveis GRAUINSTRUCAO, excluindo 5 dados ausentes, e TIPORESFONE1 que no lugar dos valores ausentes foram colocados o valor da moda.

- Balanceamento: Há desbalanceamento entre as duas classes nas duas bases trabalhadas, o que pode levar os modelos a ficarem viesados na classe majoritária ao longo da fase de treinamento. Para ajustar esse desequilíbrio optou-se por utilizar a técnica de *oversampling*, que é a criação sintética de amostras da classe minoritária até que ela fique na mesma proporção da classe majoritária. O método de *oversampling* escolhido foi o *SMOTE (Synthetic Minority Oversampling Tecnique*). Este método cria novas instâncias da classe minoritária através do uso de características dos k-vizinhos mais próximos [21][22].

- Normalização: Muito classificadores são influenciados pela ordem de grandeza dos dados. A importância da informação não está diretamente ligada a escala do número. Colocar todos os atributos na mesma escala [0, 1] é uma tarefa que deve ser feita para evitar que algum classificador trabalhe de forma enviesada pelos atributos de maiores grandezas. A Equação 1 mostra como foi procedido a normalização dos dados nas duas bases:

$$Xnorm = \frac{X - X_{min}}{X_{max} - X_{min}} \quad (1)$$

Onde:

$Xnorm$ é o valor resultante após a normalização;

X é o valor original do atributo;

X_{min} é o menor valor assumido por aquele atributo;

X_{max} é o maior valor assumido por aquele atributo.

- Otimização de hiperparâmetros: A otimização dos hiperparâmetros é a etapa da pesquisa que visa melhorar o desempenho nas predições dos modelos através de ajustes nas configurações padrões dos hiperparâmetros de cada modelo. A maioria dos modelos de *machine learning* possuem uma quantidade grande de hiperparâmetros que podem ser ajustados. A escolha de quais hiperparâmetros trabalhar envolve muito do conhecimento do pesquisador no problema alvo de sua pesquisa, aliado ao que já foi realizado e publicado em trabalhos similares. Assim, escolhidos os hiperparâmetros e os espaços de busca que cada um pode assumir num conjunto de valores (que podem ser valores contínuos, discretos ou nominais, a depender do hiperparâmetro que está sendo trabalhado). Há diversas técnicas que executam esta tarefa de encontrar a melhor combinação. A primeira técnica utilizada é suportada em probabilidade para inferir quais são os melhores valores. A segunda técnica usou inteligência de enxames. Abaixo, descrevem-se de forma resumida as principais características das técnicas escolhidas:

Otimização bayesiana - é uma abordagem que usa o teorema de Bayes para direcionar a otimização de uma função de difícil modelagem matemática e que possui alto custo computacional para ser encontrada [25]. Na otimização bayesiana é indicado para o algoritmo as faixas de valores de cada hiperparâmetro a ser testado e a métrica escolhida a ser maximizada e/ou minimizada. Num primeiro momento, valores aleatórios são escolhidos e formam uma curva *surrogate* que estima a função objetivo da métrica. Cada ponto desta função representará a probabilidade de se obter um valor ótimo na métrica, dado um conjunto de hiperparâmetros. Logo após, uma função de melhoria esperada indica a probabilidade de um valor de hiperparâmetro obter um resultado melhor que o atual. A cada iteração a função *surrogate* será aperfeiçoada, e o algoritmo encontrará os melhores valores para o conjunto dos hiperparâmetros [26]. Ao contrário de técnicas de otimização como *Grid Search* ou *Random Search* que buscam novos pontos de teste de forma gulosa (*Grid Search*) ou aleatória (*Random Search*), a otimização bayesiana aprende a cada iteração quando calcula a probabilidade do próximo conjunto de hiperparâmetro ser melhor que o anterior.

Particle Swarm Optimization – PSO: é um algoritmo de otimização que utiliza meta-heurística e é classificado na categoria dos algoritmos bioinspirados [23]. Mais especificamente, o PSO é inspirado na inteligência dos enxames que se baseia no comportamento coletivo de animais ou insetos. Nestes grupos os indivíduos ao interagirem entre si, formam um sistema inteligente e auto organizável. É desta forma que os enxames se comportam ao se deslocarem (por exemplo, voo dos pássaros), ao procurarem comida (por exemplo, colônia de formigas) ou para fugirem de predadores. No PSO cada indivíduo (também chamado de partícula) é uma possível solução para o problema pesquisado. As partículas possuem apenas dois atributos: sua posição e sua velocidade. A posição representa o local no espaço de busca que cada partícula se encontra no instante *i*, enquanto que a velocidade representa a direção e a velocidade que a partícula irá se deslocar, originando uma nova posição (no instante *i*+1). Matematicamente estes dois atributos podem ser descritos conforme as Equações 2 e 3 abaixo:

$$x_{i+1} = x_i + v_i \quad (2)$$

$$v_{i+1} = \omega \,.\, v_i + c_1 . rand(\,) . (pbest_i - x_i) + c_2 . rand(\,) . (gbest_i - x_i) \quad (3)$$

Onde:

x_i é a posição da *i-ésima* partícula;

v_i é a velocidade da *i-ésima* partícula;

ω é um fator de inércia;

$rand(\,)$ é um número randômico entre (0, 1);

$pbest_i$ é a melhor posição da *i-ésima* partícula;

$gbest_i$ é a melhor posição registrada por todas as partículas;

c_1 *e* c_2 são constantes.

Apenas com estes dois parâmetros de controle do algoritmo, as partículas conseguem trocar informações e convergirem para a solução ótima de forma rápida e com baixo custo computacional. Logo, a grande força do PSO está baseada na interação social e na troca do conhecimento sobre o espaço de busca [24].

IV. RESULTADO DOS EXPERIMENTOS

A. *Software e Hardware*

Os modelos de classificação e as otimizações dos hiperparâmetros foram processados através da ferramenta Google Colaboratory, na linguagem Python, na versão 3.10.11. As otimizações foram desenvolvidas com auxílio das bibliotecas Sckit-Optimize, versão 0.9.0. e Optunity, versão 1.1.1. Esses softwares rodaram num computador de 8 GB de memória RAM, com processador Intel Core i5 e sistema operacional Windows 10.

B. *Modelos de Machine Learning (ML)*

Para executar a classificação entre bons e maus clientes foram escolhidos cinco modelos de ML: *Decision Tree* (DT), *Random Forest* (RF), *Multilayer Perceptron* (MLP), *Light Gradient Boost Machine* (LGBM) e *eXtreme Gradient Boost* (XGBoost). Estes modelos estão entre os mais utilizados, de acordo com pesquisa bibliográfica conduzida. Especificamente, o LGBM e o XGBoost são modelos que vêm sendo muito utilizados em competições de programação, por apresentarem robustez, velocidade na execução e bons resultados.

C. *Métricas*

As métricas utilizada na avaliação dos modelos de classificação foram: *Accuracy, Precision, Recall e F1 – Score*. Essas métricas são comumente utilizadas na tarefa de classificação, elas refletem a performance de cada classificador, sendo que cada uma tem um foco diferente de avaliação. Estas métricas são facilmente calculadas através de uma matriz de confusão, como mostra a Figura 4 e as Equações 4 a 7.

		Valor Predito	
		Sim	**Não**
Real	**Sim**	Verdadeiro Positivo (TP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (TN)

Figura 4. Matriz de Confusão

- *Accuracy*: mede o acerto global do classificador (verdadeiros positivos + verdadeiros negativos). Pode falsear a performance de um classificador quando a base é desbalanceada. A Equação 4 mostra como calcular a *Accuracy*.

$$Accuracy = \frac{TP+TN}{TP+FN+FP+TN} \quad (4)$$

- *Precision*: mede o acerto do classificador nas predições positivas. Ou seja, de todos os valores preditos como positivos quais realmente eram positivos. A Equação 5 traz a forma de calcular a métrica.

$$Precision = \frac{TP}{TP+FP} \quad (5)$$

- *Recall*: é a métrica que calcula de todos os valores positivos reais, quantos o modelo acertou na predição. É calculada conforme a Equação 6.

$$Recall = \frac{TP}{TP+FN} \quad (6)$$

- *F1 – Score*: é uma média harmônica entre *Precision* e *Recall*. A Equação 7 mostra como calcular essa métrica.

$$F1 - Score = \frac{(2*Precision*Recall)}{(Precision+Recall)} \quad (7)$$

D. *Resultados*

As Tabelas 3 e 4 trazem os resultados obtidos em cada uma das bases trabalhadas. As tabelas demostram, por classificador proposto, os valores das quatro métricas escolhidas para avaliar as performances. Para essa primeira rodada de classificação foram mantidas as configurações padrões disponibilizadas pela biblioteca S*ckit-Learn* em cada algoritmo. Os valores obtidos servirão de valor base para comparar o quanto as técnicas de otimização de hiperparâmetros melhoraram a tarefa de classificação. Cabe ressaltar que o treinamento de cada algoritmo foi feito através de validação cruzada *k-fold*, com k=10, e o procedimento foi repetido 30 vezes para evitar que os resultados figurassem como aleatórios. Assim, os resultados que constam nas tabelas foram registrados como a média das 30 execuções.

Tabela 3. Resultados dos modelos sem otimização - CV 10 Folds para *German Credit Data*

Algoritmo	**Accuracy (DP)**	**Precision (DP)**	**Recall (DP)**	**F1- Score (DP)**
DT	0,7079 (0,0099)	0,7102 (0,0217)	0,7241 (0,0267)	0,7175 (0,0060)
RF	**0,8132 (0,0394)**	**0,7999 (0,0426)**	**0,8473 (0,0423)**	**0,8281 (0,0380)**
MLP	0,7640 (0,0308)	0,7582 (0,0331)	0,7881 (0,0348)	0,7704 (0,0300)
LGBM	0,7954 (0,0347)	0,7849 (0,0376)	0,8260 (0,0352)	0,8045 (0,0334)
XGBoost	0,7906 (0,0349)	0,7825 (0,0390)	0,8250 (0,0338)	0,8001 (0,0332)

Fonte: próprio autor

Tabela 4. Resultados dos modelos sem otimização - CV 10 Fold para Cooperativa				
Algoritmo	**Accuracy (DP)**	**Precision (DP)**	**Recall (DP)**	**F1- Score (DP)**
DT	0,6544 (0,0029)	0,6509 (0,0044)	0,6577 (0,0243)	0,6541 (0,0094)
RF	0,7200 (0,0312)	0,7137 (0,0431)	0,7294 (0,0201)	0,7218 (0,0241)
MLP	**0,7667 (0,0343)**	**0,7553 (0,0369)**	**0,8087 (0,0382)**	**0,7804 (0,0333)**
LGBM	0,7149 (0,0050)	0,7126 (0,0012)	0,7156 (0,0083)	0,7140 (0,0109)
XGBoost	0,7178 (0,0006)	0,7163 (0,0035)	0,7157 (0,0042)	0,7160 (0,0104)

Fonte: próprio autor

Na base da *German Credit,* a *Random Forest* obteve as melhores performances nas quatro métricas, seguida da *LGBM* e do *XGBoost*. Já na base da Cooperativa, coube ao *MLP* as melhores performances, também nas quatro métricas. Em segundo ficou o *XGBoost* e em terceiro a *LGBM*.

Num segundo momento foram feitas as otimizações bayesiana e com uso do PSO para verificar se estas técnicas melhorariam as performances dos modelos, visto que nessa nova rodada de classificação os modelos estariam trabalhando com as melhores combinações de hiperparâmetros. As Tabelas 5 e 6 resumem os valores das métricas para as bases de *German Credit* e da Cooperativa, respectivamente, após a otimização bayesiana.

Cabe destacar que os hiperparâmetros escolhidos para otimização foram aqueles que poderiam evitar o *overfiting*, e foram escolhidos seguindo o que se recomenda na literatura. Os hiperparâmetros selecionados e os espaços de busca que foram passados para os otimizadores estão descritos na Tabela 9, no final deste artigo.

Tabela 5. Resultados dos modelos - otimização Bayesian para *German Credit Data*				
Algoritmo	**Accuracy (DP)**	**Precision (DP)**	**Recall (DP)**	**F1- Score (DP)**
DT	0,7292 (0,0076)	0,7356 (0,0197)	0,8444 (0,0190)	0,7409 (0,0018)
RF	**0,8006 (0,0378)**	0,7765 (0,0356)	0,8636 (0,0239)	**0,8130 (0,0387)**
MLP	0,7681 (0,0023)	0,7665 (0,0021)	**0,9104 (0,0026)**	0,7855 (0,0039)
LGBM	0,7898 (0,0274)	**0,7898 (0,0255)**	0,8595 (0,0175)	0,8006 (0,0229)
XGBoost	0,7854 (0,0106)	0,7726 (0,0201)	0,8721 (0,0198)	0,7971 (0,0117)

Fonte: próprio autor

Tabela 6. Resultados dos modelos - otimização bayesiana para base da Cooperativa				
Algoritmo	**Accuracy (DP)**	**Precision (DP)**	**Recall (DP)**	**F1- Score (DP)**
DT	0,6817 (0,0027)	0,6743 (0,0015)	0,7968 (0,0023)	0,6990 (0,0011)
RF	0,6977 (0,0237)	0,6722 (0,0198)	0,7975 (0,0202)	0,7152 (0,0227)
MLP	**0,7692 (0,0112)**	**0,8346 (0,0231)**	**0,858 (0,0277)**	**0,7832 (0,0198)**
LGBM	0,719 (0,0092)	0,7146 (0,0099)	0,7646 (0,0113)	0,7156 (0,0148)
XGBoost	0,7175 (0,0288)	0,7157 (0,0197)	0,7608 (0,0199)	0,7164 (0,0201)

Fonte: próprio autor

Verifica-se que a otimização bayesiana melhorou de forma majoritária as métricas de todos os algoritmos. Com destaque para a melhoria substancial na métrica *Recall*. Está métrica, por exemplo, chegou a uma assertividade de 91,04% na *German Credit* para o MLP (sem a otimização o valor era de 78,81%). Porém, houve um decréscimo nas métricas de *Accuracy, Precision, F1-Score* na *Random Forest* (RF) e XGBoost e piora, também, nas métricas *Accuracy* e *F1-Score* na da LGBM.

De maneira semelhante, também foram feitas tentativa de melhorias com a otimização via PSO. Como na otimização anterior, os mesmos hiperparâmetros e espaços de buscas foram passados para PSO, para que ao final houvesse a possibilidade de comparar as performances dos otimizadores. As Tabelas 7 e 8 trazem os resultados da *German Credit* e da Cooperativa após otimização via PSO.

Tabela 7. Resultados dos modelos - otimização com PSO para a *German Credit Data*				
Algoritmo	**Accuracy (DP)**	**Precision (DP)**	**Recall (DP)**	**F1- Score (DP)**
DT	**0,7498 (0,0320)**	0,7332 (0,0276)	0,8023 (0,0199)	0,7600 (0,0233)
RF	0,7366 (0,0112)	0,7384 (0,0204)	0,8113 (0,0111)	0,7631 (0,0217)
MLP	0,7075 (0,0103)	0,7262 (0,0127)	**0,9168 (0,0093)**	0,7243 (0,0108)
LGBM	0,7194 (0,0129)	**0,7401 (0,0201)**	0,8695 (0,0077)	**0,7752 (0,0233)**
XGBoost	0,7291 (0,0102)	0,7303 (0,0188)	0,7484 (0,0196)	0,7465 (0,0113)

Fonte: próprio autor

Tabela 8. Resultados dos modelos - otimização com PSO para base da Cooperativa				
Algoritmo	**Accuracy (DP)**	**Precision (DP)**	**Recall (DP)**	**F1- Score (DP)**
DT	0,6797 (0,0337)	0,6601 (0,0273)	0,7686 (0,0147)	0,6979 (0,0200)
RF	0,6755 (0,0311)	0,6577 (0,0266)	0,7753 (0,0239)	0,6979 (0,0186)
MLP	**0,7035 (0,0297)**	**0,7025 (0,0199)**	**0,8007 (0,0204)**	**0,7119 (0,2330)**
LGBM	0,6964 (0,0288)	0,6891 (0,0202)	0,7653 (0,0107)	0,7079 (0,0205
XGBoost	0,6983 (0,0301)	0,6898 (0,0188)	0,7390 (0,0113)	0,7050 (0,0257)

Fonte: próprio autor

Como pode ser verificado a otimização por PSO não foi uma boa estratégia para melhorar as performances de classificação nas duas bases trabalhadas. Todas as métricas em todos os modelos pioraram. Inclusive quando comparados com os modelos sem otimização (configuração padrão da biblioteca *Scikt-Learn*). Apesar de ser um algoritmo muito robusto e que demonstra ter bons resultados em diversos problemas de otimização, o PSO ficou muito aquém em relação aos resultados obtidos com a otimização bayesiana.

Uma explicação para o desempenho abaixo do esperado do PSO é que as partículas convergiram para um máximo local no conjunto de soluções e não foram capazes de continuar explorando outras áreas com soluções melhores, com intuito de encontrar o máximo global. A aplicação do PSO neste trabalho foi do algoritmo padrão. Porém, existem modificações no algoritmo do PSO que melhoram suas possibilidades de fugir de mínimos/máximos locais.

V. CONCLUSÕES

Modelos de *credit scoring* são muito utilizados na indústria financeira com objetivo de automatizar, diminuir os custos de análise e incrementar as performances das classificações. Erros nas classificações representariam desperdícios de recursos financeiros. Este é um setor que movimenta bilhões de reais por ano, e cada pequena melhoria nos indicadores de performances representaria ganhos extraordinários (maiores lucros).

Desta forma, baseado nos resultados encontrados, chega-se a algumas conclusões: (a) a otimização pelo algoritmo bayesiano se mostrou bastante eficaz no papel de incrementar

as predições dos modelos sem otimização (configuração padrão); (b) após a otimização bayesiana, a maior parte das métricas demonstraram melhorias, com destaque para o *Recall* nas duas bases trabalhadas; e (c) cabe destacar que a otimização via PSO não foi tão eficiente nestas duas bases, visto que, a priori, o algoritmo ficou preso em máximos locais e não encontrou o máximo global.

Para trabalhos futuros, sugere-se à incorporação no fluxo de tarefas o uso de técnicas de seleção de características *(features selection)* para selecionar os atributos mais relevantes e eliminar informação redundante. Também será proposta a aplicação de outras técnicas de otimização de hiperparâmetros, como por exemplo: *i*) uso de modelos modificados e aprimorados do PSO, *ii)* outras heurísticas de inteligência de enxames (ACO e FSS), *iii*) uso de modelos evolutivos, a exemplo do CMAES (*Covariance Matrix Adaptation Evolution Strategy*) ou Algoritmos Genéticos (GA). Além disso, é possível refinar e aumentar o conjunto dos hiperparâmetros a serem otimizados. Caberá, por fim, à aplicação de testes estatísticos para uma melhor comparação dos resultados obtidos.

REFERÊNCIAS

[1] Estatística Monetária e de Crédito. Banco Central do Brasil, 2023. Disponível em: <https://www.bcb.gov.br/estatisticas/estatisticasmonetariascredito> Acessado em: 17/06/2023.

[2] Grmanová, E., & Ivanová, E.. "Efficiency of banks in Slovakia: Measuring by DEA models." Journal of International Studies Vol, 11(1), 2018.

[3] N. C. Armando, "Relações entre crédito e crescimento econômico no Brasil, 2000 a 2006", Tese apresentada à Universidade Federal de Viçosa – MG, 2007.

[4] E. I. Altman, A. Saunders, "Credit risk measurement: Developments over the last 20 years." Journal of Banking & Finance, v. 21, p. 1721–1742, 1998.

[5] Z. Huang et al. Credit rating analysis with support vector machines and neural networks: a market comparative study. Decision Support Systems, Elsevier BV, v. 37, n. 4, p. 543–558, Sep 2004. ISSN 0167-923.

[6] Altman, E. I., & Saunders, A. "Credit risk measurement: Developments over the last 20 years. Journal of Banking & Finance," 21(11-12), 1721-1742, 1997.

[7] S. Lessmann et al. "Benchmarking state-of-art classification algorithms for credit scoring: an update of research." European Journal of Operational Research, 247, 124-136, 2015.

[8] 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, 21(2), 117-134, 2016.

[9] C. F. Tsai and J.-W. Wu, "Using neural network ensembles for bankruptcy prediction and credit scoring," Expert systems with applications, vol. 34, no. 4, pp. 2639–2649, 2008.

[10] D. West, Neural network credit scoring models, Computers and Operations Research 27 (2000) 1131–1152.

[11] Malekipirbazari M, Aksakalli V. Risk assessment in social lending via random forests. Expert Systems with Applications, 2015, 42(10): 4621-4631.

[12] C.L. Huang, H. Chen, C.-J. Hsu, W.-H. Chen, S. Wu, Credit rating analysis with support vector machines and neural networks: a market comparative study, Decision Support Systems 37 (2004) 543–558.

[13] Xia, Y., Liu, C., Li, Y., & Liu, N. "A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring." Expert Systems with Applications, 78, 225-241, 2017.

[14] Chen, T., & Guestrin, C. "Xgboost: A scalable tree boosting system." In Proceedings of the 22nd acm sigkdd international on Knowlegde Discovery and Data Mining, 2016.

[15] F. Shen, X. Zhao, Z. Li, K. Li, and Z. Meng, "A novel ensemble classification model based on neural networks and a classifier optimisation technique for imbalanced credit risk evaluation," Physica A: Statistical Mechanics and its Applications, vol. 526, p. 121073, 2019.

[16] Ahn, K.-J. K. H. "Corporate Credit Rating using Multiclass Classification Models with order Information". World Academy of Science, Engineering and Technology, 2011.

[17] T. Haoxin et al. "Risk assessment of credit field based on PSO – SVM." 2nd International Conference on Economic Management na Model Engineering (ICEMME), 2020.

[18] Yunus, K. Musta, Ç. "Comparative evaluation of machine learning algorithms with parameter optimization and features elimination for fraud detection.". International Conference on Electrical, Computer and Energy Technologies (ICECET), 2021.

[19] W. Chaoqum, H. Zhongyi, C. Raymond, D. Sandeep, C. Yuan, B. Yukun, "A PSO-Based ensemble model Peer-to-Peer credit scoring", 14th International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery (ICNC-FSKD), 2018.

[20] Y. Li, W. Chen, "A comparative performance assessment of ensemble learning for credit scoring", Mathematics: vol. 8, p. 1756, 2020.

[21] X. Xu, W. Chen, andY. Sun, "Over-sampling algorithm for imbalanced data classificat ion," J. Syst . Eng. Elect ron., vol. 30, no. 6, pp. 1182–1191, 2019.

[22] N. V Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: synthetic minority over-sampling technique," J. Art if. Intell. Res., vol. 16, pp. 321–357, 2002.

[23] J. Kennedy, R. Eberhart. "Particle swarm optimization," in: Neural Networks, 1995. Proceedings., IEEE International Conference on, vol. 4, 1995, pp. 1942–1948.

[24] P. Gustavo; OSÓRIO, Fernando. "Otimização por Enxame de Partículas aplicado à formação e atuação de grupos robóticos." Scientia, [S. l.], v. 20, n. 2, p. 94–106, 2009.

[25] M. Pelikan et al. "BOA: The Bayesian Optimization Algorithm."

[26] Xia, Yufei, et al. "A boosted decision tree approach using Bayesian hyper-parameter optimization for credit scoring." Expert systems with applications 78 (2017): 225-241.

Tabela 9. Hiperparâmetros selecionados para otimização

Modelo	**Hiperparâmetros (espaço de busca)**
DT	"max_depth" : Integer (2, 10), "min_samples_split" : Integer (2, 64), "min_samples_leaf" : Integer (2, 64), "criterion" : Categorical(["gini", "entropy"])
RF	"max_depth": Integer (2, 10), "max_features": Categorical (['auto', 'sqrt', 'log2']), "min_samples_leaf": Integer(2,64), "min_samples_split": Integer (2, 64), "n_estimators": Integer (100, 400), "criterion" : Categorical (['gini', 'entropy','log_loss'])
MLP	"activation": Categorical (['identity', 'logistic', 'tanh', 'relu']), "batch_size": Integer(32,64), "max_iter": Integer (100, 500), "solver" : Categorical (['lbfgs', 'sgd','adam'])
LGBM	num_leaves': Integer (2, 64), 'max_depth': Integer (2,10), 'min_gain_to_split': Integer (0,15), 'bagging_fraction': Real(0.5, 1)
XGBoost	"learning_rate": Real (1e-3, 1.0), "algorithm": Categorical (['SAMME', 'SAMME.R']), "n_estimators": Integer (100, 400)

Fonte: próprio autor