Classificação de Clientes Inadimplentes Utilizando Algoritmos de Aprendizagem de Máquina Supervisionados

(Trabalho final da Disciplina de Aprendizado de Máquina 2/2019)

Fernando Nakayama[1], **Agnaldo de Souza Batista**[1]

[1]Doutorando do Programa de Pós-Graduação em Informática da Universidade Federal do Paraná

fernandonakayama@ufpr.br, asbatista@ufpr.br

1. Introdução

O volume de dados gerados pelos diversos serviços e dispositivos atualmente é muito elevado, o que torna um desafio para o ser humano lidar com esses dados de maneira eficiente e eficaz. Uma solução empregada recentemente consiste em *ensinar* os computadores a aprenderem e, a partir desse aprendizado, tomar decisões. Essas decisões ocorrem em diversas atividades, desde a resolução de problemas matemáticos complexos, passando por aprender jogos e realizar predições em diversas áreas, tais como prospecção de petróleo, previsão do tempo, detecção de doenças como câncer, além de análises financeiras, dentre outras.

Nas atividades financeiras, o aprendizado de máquina pode ser empregado para auxiliar na identificação do perfil dos clientes, adimplentes ou inadimplentes, auxiliando instituições de crédito ou mesmo estabelecimentos comerciais a tomarem decisões acerca da concessão de empréstimos ou a venda de produtos e serviços, por exemplo. Clientes bons pagadores podem obter vantagens como juros menores e maiores prazos de financiamento, dentre outras facilidades. Porém, identificar o perfil desses clientes é uma tarefa complexa, que demanda lidar com grandes bases de dados. Realizar essa atividade manualmente é um processo lento e de grande dificuldade, mas que pode se beneficiar sobremaneira do uso de algoritmos de aprendizado de máquina.

Este trabalho apresenta um algoritmo de aprendizagem de máquina supervisionado para classificação de clientes inadimplentes. Os experimentos foram conduzidos usando os algoritmos implementados na biblioteca de aprendizado de máquina *scikit-learn* [scikit learn 2019], a fim de verificar a acurácia, *recall*, precisão e F1-*Score* dos modelos avaliados. Dentre esses, foi identificado aquele que melhor classifica ambas as classes de forma igualitária. O modelo empregou um *Voting Classifier* com *stacking* de classificadores - *Random Forest, Adaptive Boosting e Extreme Gradient Boosting*) - e usando a seleção de características RFECV+RF e F1-*Score* como parâmetro. Ele foi executado sobre uma base balanceada com a técnica *RandomUnderSample*. Os resultados indicam que esse modelo classificou a classe de bons pagadores com 60% de precisão e a classe de maus pagadores, 66%. Além disso, a sensibilidade para os bons pagadores atingiu 56%, enquanto para os maus pagadores ficou em 71%. A acurácia do modelo foi de 63% e é uma métrica adequada para esse caso, visto que a base foi balanceada.

Este artigo está organizado da seguinte forma: a Seção 2 descreve a metodologia empregada na classificação das classes de clientes; os experimentos realizados e seus

resultados encontram-se detalhados na Seção 3; os resultados são discutidos na Seção 4; e a conclusão é apresentada na Seção 5.

2. Metodologia empregada

O trabalho foi realizado seguindo-se o enunciado pré-estabelecido e consiste em classificar clientes inadimplentes utilizando-se algoritmos de aprendizagem de máquina. Foi disponibilizada uma base de dados de clientes que contraíram empréstimos junto a uma instituição financeira. Essa base possuía 672 atributos e um total de 219984 registros. Os clientes que realizaram o pagamento em até 30 dias possuíam valor 0 na variável Y. Caso contrário, Y recebeu o valor 1.

As bases para treinamento, validação e teste foram criadas a partir da divisão da base fornecida em três partes distintas, 50%, 20% e 30%, respectivamente. Considerando que se tratava de uma base desbalanceada, os resultados são reportados através de métricas como *Precision*, *Recall*, F_*score* (F1-*Score*) e curva ROC.

2.1. Formulação de um problema de classificação binário

Classificação binária se refere ao caso onde a variável alvo fornecida ao sistema pertence a uma de duas classes distintas. Neste trabalho, os clientes pertencem ou a categoria positiva ou a categoria negativa. As categorias podem ser modeladas como variáveis binárias randômicas, onde $Y \in \{0, 1\}$, 0 é definido como classe negativa (ou não-padrão) e 1 corresponde à classe positiva. Os valores 0 na base de dados fornecida representam os bons pagadores, enquanto os valores 1, os maus pagadores. A classe 1 é considerada positiva, pois através dela se busca detectar os clientes inadimplentes. Diante desse cenário, torna-se possível ajustar um modelo de aprendizagem de máquina supervisionada, uma vez que a base de dados apresenta rótulos para os bons e maus pagadores. A principal característica dessa aprendizagem é o conhecimento prévio da variável alvo para que sejam realizadas inferências entre ela e as predições.

Uma das principais características da base de dados disponibilizada é o desbalanceamento existente entre as classes, o que significa que existe uma elevada disparidade entre as quantidades de bons e maus pagadores. Nessa base, há uma relação de 0,26 bons pagadores para 1 mau pagador, ou seja, aproximadamente 4 maus pagadores para cada bom pagador (4:1). Desse forma, este trabalho consiste em classificar clientes inadimplentes utilizando algoritmos de aprendizagem de máquina. Dada a relação entre as classes na base de dados, essa tarefa não é extremamente complexa, uma vez que somente ignorando os bons pagadores já é possível conseguir uma acurácia de 74%. Porém, para um modelo balanceado e que generalize de forma eficiente bons e maus pagadores, será utilizado o princípio de que é necessário minimizar os prejuízos causados pelos maus pagadores, mas sem ignorar os bons, visto que a concessão de empréstimos com juros é tão importante quanto identificar e avaliar o risco de conceder empréstimo a maus pagadores.

2.2. Seleção de Ferramentas para Resolução do Problema

Os índices da base de dados fornecida para o desenvolvimento do trabalho eram genéricos, e.g., v1, v2, v3. Dessa forma, era impossível identificar especificamente uma característica. Sendo assim, recorreu-se a métodos matemáticos e estatísticos para a seleção das melhores características. Dentre os métodos disponíveis, foram empregados

um algoritmo genético (AG) e a eliminação recursiva de características com validação cruzada (do inglês, *Recursive feature elimination with cross-validation* - RFECV). Os algoritmos genéticos são amplamente utilizados em classificação binária [Desai et al. 1997] e também para avaliação de crédito [Zhang et al. 2019]. A RFECV é utilizada na seleção de características e, inclusive, na predição de maus pagadores [Ma et al. 2018].

A etapa seguinte consistiu na escolha dos algoritmos de classificação a serem avaliados na nova base com as melhores características. Inicialmente, utilizou-se os classificadores mais simples com o objetivo de selecionar os mais eficientes. Posteriormente, as técnicas de *bagging* e *boosting* foram avaliadas isoladamente. Por fim, avaliou-se um classificador baseado em votos com o empilhamento de classificadores. Dentre os diversos classificadores disponíveis, foram avaliados a árvore de decisão, KNN, *Naive Bayes*, SVM, *Random Forest*, *Adaptive Boosting*, *Gradient Boosting* e *Extreme Gradient Boosting*. Todos esses classificadores foram testados usando a biblioteca de aprendizado de máquina *scikit-learn* [Pedregosa et al. 2011].

Para uma melhor compreensão sobre os fatores que influenciam os resultados, os testes foram realizados com três formatos de base distintos, (i) base desbalanceada original, (ii) base balanceada descartando-se elementos da classe majoritária (*RandomUnderSampler*) e (iii) base balanceada agregando elementos sintéticos na base minoritária (*Synthetic Minority Over-sampling Technique* (SMOTE)). Os limites impostos no enunciado do trabalho, que previa a divisão das bases em 50% para treino, 20% para validação e 30% para teste, foram respeitados para cada uma das bases. Todos os resultados estão dispostos em formato de tabela e, para uma melhor organização desse trabalho, somente os melhores resultados serão discutidos.

3. Experimentos realizados

O processo de classificação de clientes inadimplentes envolveu a realização de diversos experimentos, a fim de se identificar o algoritmo de aprendizagem de máquina que oferecesse os melhores resultados. Esses experimentos envolveram a exploração dos dados existentes na base de dados fornecida e sua limpeza. Um processo de seleção de características foi conduzido, seguido de diversos testes. Foi realizado o balanceamento da base de dados aplicando-se algumas técnicas e realizados novos testes. Todos os experimentos são descritos a seguir. Os testes foram realizados com a biblioteca de aprendizagem de máquina *scikit-learn*, versão 0.21.3, rodando sobre *Python*, versão 3.7.5, no *Jupyter Notebook*, versão 6.0.1. Os notebooks desenvolvidos em Jupyter estão disponíveis em um repositório no *GitHub*.[1]

3.1. Exploração dos dados

A base de dados fornecida - credit.csv - possui 219984 exemplos com 672 características cada um, totalizando 436,4MB de dados. Dos exemplos existentes, 45063 correspondem a bons pagadores, que são representados pela classe 0, e 174921 maus pagadores, representados pela classe 1. Durante a exploração dos dados, observou-se que diversas características possuíam a informação NA, seja parcialmente ou em todos os exemplos. Adicionalmente, constatou-se o desbalanceamento entre as classes existentes, cuja distribuição encontra-se demonstrada na Figura 1.

[1] https://github.com/fernandonakayama/Disciplina_Machine_Learning

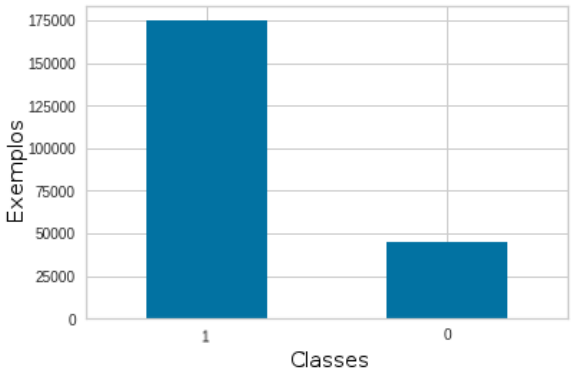


Figura 1. Distribuição das classes na base original

O processo de avaliação dos resultados obtidos é um dos principais problemas observados quando se lida com bases desbalanceadas. O uso de métricas mais simples como acurácia, por exemplo, pode levar a uma interpretação errônea dos resultados. Nesses casos, se o classificador sempre prediz a classe mais comum sem analisar as características, ainda assim ele obterá uma alta taxa de acurácia, que pode não corresponder à realidade. Logo, os resultados devem ser avaliados de outras formas. Dentre as diversas métricas existentes para essa avaliação, foram empregadas neste trabalho a curva ROC (do inglês, *Receiver Operator Characteristic Curve*) e a área sob a curva ROC (do inglês, *Area Under the Curve* - AUC). Adicionalmente, foram avaliadas a precisão, a revocação (*recall*) e o F1-*Score*.

3.2. Limpeza da base de dados

A primeira fase dos trabalhos envolveu a remoção da base de dados de todas as características que continham NA, seja parcialmente ou em todos os exemplos. Foram verificadas 477 características nessa situação. Após sua remoção, a dimensionalidade da base foi reduzida para 195 características e o seu tamanho diminuido para 165MB. Essa nova base de dados foi empregada nos experimentos. Entende-se que a remoção dos valores não disponíveis pode interferir nos resultados [Fernández et al. 2017], uma vez que a relação dos atributos é desconhecida. Dessa forma, a remoção dos NA se deu principalmente pelo desconhecimento do significado de cada variável e, por esse motivo, também não foram realizados experimentos envolvendo a junção de características.

Para que posteriormente houvesse algum parâmetro de comparação, realizou-se uma avaliação com um classificador de árvore de decisão (*DecisionTreeClassifier*) da biblioteca *scikit-learn*. Nesse teste preliminar, a base de dados foi dividida em 40% para treino e 60% para teste e, em seguida, o classificador foi aplicado. A acurácia obtida foi de 79%, o que é um bom valor se analisado de maneira isolada. Sua matriz de confusão é apresentada na Figura 2 e indica que esse classificador identifica corretamente a maioria dos maus pagadores. Porém, ele tem dificuldades em relação aos bons pagadores. Avaliou-se, ainda, um classificador baseado em *boosting* (*extreme gradient boosting classifier - XGBClassifier)* da biblioteca *scikit-learn*, que obteve uma acurácia de 80,21% para toda a base. Selecionando-se somente uma característica aleatória da base, e.g., 'v8', a acurácia caiu para 79,56%. Isso indica que o uso da base completa ou de apenas uma característica tem pouca relação com a acurácia e que ela não deve ser utilizada como parâmetro avaliativo em bases altamente desbalanceadas.

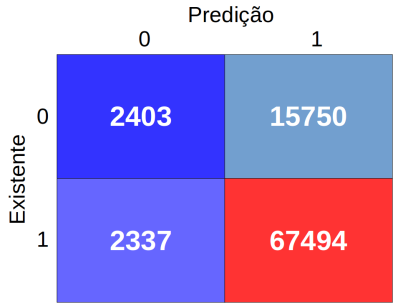


Figura 2. Matriz de confusão do classificador *DecisionTree*

3.3. Seleção de características

A seleção das características buscou verificar as características mais relevantes para avaliação de clientes inadimplentes dentre as 195 disponíveis. A redução da dimensionalidade permite determinar um subconjunto de características para construir um bom modelo para classificação ou predição [Maldonado et al. 2014]. Porém, esse processo é um desafio em bases de dados desbalanceadas. Neste trabalho foram empregadas as seguintes técnicas para selecionar as melhores características:

- RFECV com o classificador *Logistic Regression* (RFECV+LR);
- RFECV com o classificador *Random Forest* (RFECV+RF); e
- Algoritmo genético (AG).

Cada uma dessas técnicas apresentou um comportamento distinto sobre a base de dados, selecionando uma quantidade de características diferente. Os resultados obtidos encontram-se representados na Tabela 1, enquanto o comportamento de cada técnica para seleção de características está representada nas Figuras 3 e 4.

Após a seleção das características, a base de dados original, cujas caracteristicas contendo NA foram inicialmente removidas, foi dividida em 3 partes - treino, validação e testes - nas proporções de 50%, 20% e 30%, respectivamente. Visto que as quantidades de características selecionadas por cada algoritmo foram distintas, as novas bases ficaram com tamanhos diferentes, conforme observa-se na Tabela 1.

Tabela 1. Detalhes das novas bases de dados

Algoritmo	**Número de Características**	**Tamanho (MB)**			
		Treino	Validação	Teste	Total
AG	102	35	14,3	21,4	70,6
RFECV+LR	24	12,8	5,2	7,8	25,8
RFECV+RF	67	45	18,4	27,6	91,4

3.4. Testes aplicados na base desbalanceada

Os testes aplicados nas três bases desbalanceadas descritas na Subseção 3.3 seguiram a seguinte metodologia: teste com um classificador de árvore de decisão - *DecisionTreeClassifier*, teste com um classificador do tipo *bagging* - *RandomForestClassifier* - e

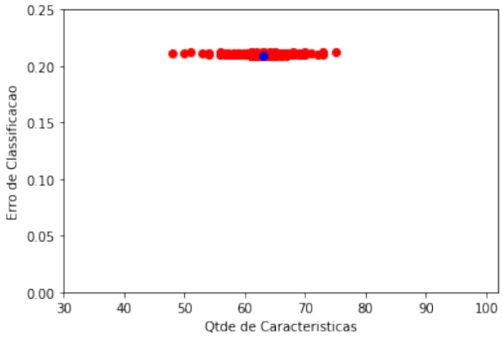
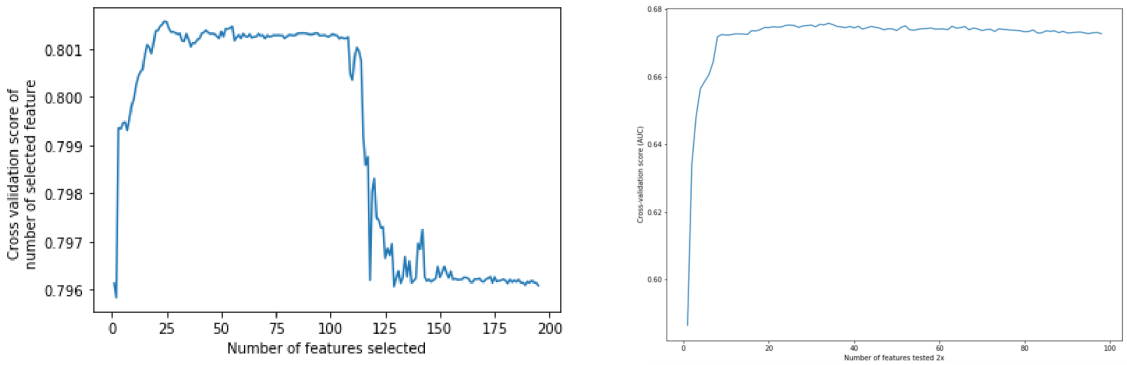


Figura 3. AG - Seleção de Características



(a) RFECV+LR (b) RFECV+RF

Figura 4. RFECV+LR e RFECV+RF: Seleção de características

testes com dois classificadores do tipo *boosting* - *AdaBoostClassifier* (*Adaptive Boosting*) e *XGBClassifier* (*Extreme Gradient Boosting Classifier*). Todos esses classificadores estão implementados na biblioteca *scikit-learn*. Nessa etapa, os parâmetros de cada classificador foram configurados manualmente nas bases de treino e validação, através de ajustes e tentativas. A base de testes foi apresentada aos classificadores após todas as avaliações. Considerando os melhores ajustes de cada classificador, implementou-se um modelo baseado em votos, privilegiando os classificadores que obtiveram melhor resultado nos testes individuais. Essa implementação foi feita utilizando o *VotingClassifier* da biblioteca *scikit-learn*.

Os resultados para os classificadores individuais e para o *Voting Classifier* encontram-se sumarizados na Tabela 2, estando em destaque os melhores modelos. O *Voting Classifier* utilizando a base com características extraídas através do RFECV+RF e o *XGBClassifier* utilizando o AG obtiveram resultados semelhantes. Em um primeiro momento, a acurácia de 0,80, *Recall* de 0,80 e F1-*Score* de 0,74 indicam um bom modelo. Entretanto, analisando-se minuciosamente, constata-se que alguns valores comprometem a eficiência geral do modelo. A primeira observação diz respeito ao uso da média ponderada na avaliação do modelo, visto que ela equilibra os valores de precisão, *Recall* e F1-*Score* para as duas classes. Em contrapartida, avaliando-se somente a média macro, os valores do modelo *Voting Classifier* usando as características extraídas com RFECV+RF caem para 0,71, 0,54 e 0,53, considerando-se a precisão, o *Recall* e o F1-*Score*, respectivamente.

Tabela 2. Resultados obtidos na base desbalanceada

Classificador	Base de dados	RFECV+RF					RFECV+LR					Algoritmo Genético				
		Acurácia	Precisão Macro	Precisão Média	*Recall* Médio	F1-*Score*	Acurácia	Precisão Macro	Precisão Média	*Recall* Médio	F1-*Score*	Acurácia	Precisão Macro	Precisão Média	*Recall* Médio	F1-*Score*
Decision Tree	Validação	0,7012	0,55	0,71	0,70	0,70	0,7396	0,56	0,71	0,74	0,72	0,7205	0,55	0,71	0,72	0,71
	Teste	0,6985	0,55	0,71	0,70	0,70	0,7394	0,57	0,71	0,74	0,72	0,7216	0,55	0,71	0,72	0,71
Random Forest	Validação	0,7884	0,63	0,74	0,79	0,74	0,7835	0,61	0,73	0,78	0,74	0,7934	0,65	0,74	0,79	0,74
	Teste	0,7871	0,63	0,74	0,79	0,74	0,7822	0,61	0,73	0,78	0,74	0,7905	0,65	0,75	0,79	0,74
AdaBoost	Validação	0,8004	0,70	0,76	0,80	0,74	0,7996	0,70	0,76	0,80	0,73	0,7998	0,69	0,76	0,80	0,74
	Teste	0,7996	0,70	0,77	0,80	0,74	0,7992	0,71	0,77	0,80	0,74	0,7994	0,70	0,76	0,80	0,74
XGB	Validação	0,8023	0,71	0,77	0,80	0,74	0,8012	0,71	0,77	0,80	0,74	0,8018	0,71	0,77	0,80	0,74
	Teste	0,8009	0,71	0,77	0,80	0,74	0,8003	0,72	0,77	0,80	0,74	**0,8006**	**0,71**	**0,77**	**0,80**	**0,74**
Voting Classifier	Validação	0,8017	0,71	0,77	0,80	0,74	0,8007	0,71	0,77	0,80	0,73	0,8016	0,71	0,77	0,80	0,74
(RF, DT, ADA, XGBC)	Teste	**0,8005**	**0,71**	**0,77**	**0,80**	**0,74**	0,80	0,71	0,77	0,80	0,74	0,7996	0,71	0,77	0,80	0,74

Legenda:

- **RFECV+RF**: Eliminação de características de forma recursiva com validação cruzada usando o classificador *Random Forest*
- **RFECV+RF+F**: Eliminação de características de forma recursiva com validação cruzada usando o classificador *Random Forest* e avaliando a métrica F1-*Score*
- **RFECV+LR**: Eliminação de características de forma recursiva com validação cruzada usando o classificador *Logistic Regression*
- **RF**: Classificador *Random Forest*
- **DT**: Classificador *Decision Tree*
- **ADA**: Classificador *AdaBoost*

Outro parâmetro utilizado para avaliação dos melhores modelos foi a curva ROC. Conforme se observa na Figura 5, a área sob a curva (AUC) foi de 0,54, um valor considerado baixo. Analisando-se todos o parâmetros avaliativos, conclui-se que os modelos obtidos a partir da base desbalanceada identificam a classe majoritária (Classe 1 - Maus pagadores) com eficiência, mas o mesmo não ocorre com a classe minoritária (Classe 0 - Bons pagadores). Visto que o objetivo estipulado é um modelo mais eficiente para classificar ambas as classes, evoluiu-se o trabalho no sentido de equalizar a base de dados através de técnicas disponíveis na literatura.

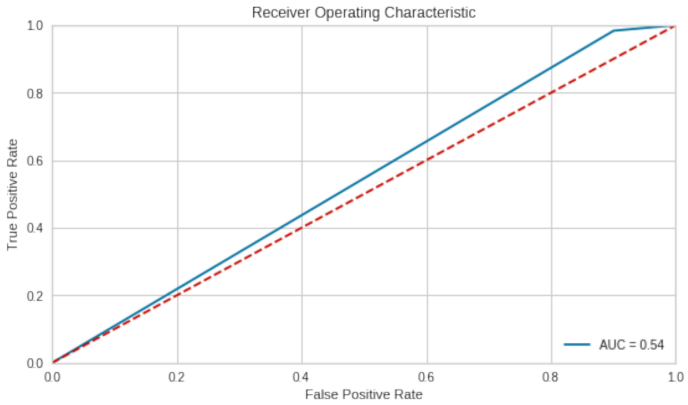


Figura 5. Curva ROC dos Melhores Modelos - Base desbalanceada

3.5. Balanceamento da base de dados

O desbalanceamento acentuado entre classes de uma base de dados impacta nas predições, especialmente neste caso em que há uma prevalência de maus pagadores. A relação entre maus e bons pagadores é de aproximadamente 4:1. A fim de se obter uma relação de 1:1 entre as duas classes, podem ser adotados dois procedimentos distintos. No primeiro caso, a base de dados majoritária é reduzida para igualar-se à base minoritária, técnica conhecida como *undersampling*. No segundo caso, a base minoritária é aumentada por meio da incorporação de dados sintéticos até igualar-se à outra classe, procedimento conhecido como *oversampling* [Sadatrasoul et al. 2015].

Nesse trabalho, o *oversampling* da classe minoritária foi feito por meio da técnica de geração de amostras sintéticas (do inglês, *Synthetic Minority Over-sampling Technique* - SMOTE). Essa técnica equaliza a classe minoritária à majoritária através da interpolação dos exemplos que estão próximos. Ela expande os clusters dos dados minoritários, fortalecendo as áreas limítrofes entre as classes. Entretanto, a SMOTE pode causar uma generalização excessiva na predição dos dados [Fernández et al. 2017]. A nova base de dados gerada com a técnica SMOTE encontra-se representada na Figura 6, onde se observa que ambas as classes possuem a mesma quantidade de exemplos, 87341. Constata-se que a SMOTE gerou 42278 novos exemplos para a classe de bons pagadores. O conjunto de resultados obtidos a partir da base ampliada com SMOTE é exibido na Tabela 3.

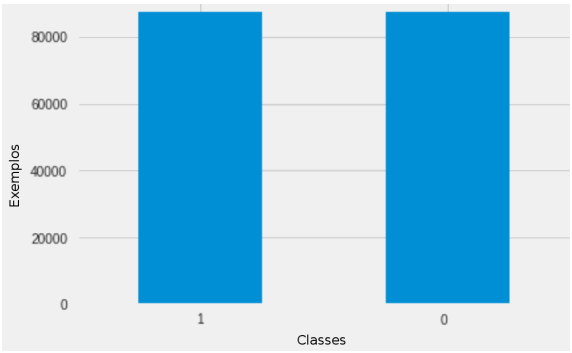


Figura 6. Balanceamento da base de dados com SMOTE

Tabela 3. *Voting Classifier* (RF, ADA, XGBC): Balanceamento com SMOTE

Classificador		**RFECV+RF**		**RFECV+LR**		**AG**		**RFECV+RF+F1**	
Base de Dados		Validação	Teste	Validação	Teste	Validação	Teste	Validação	Teste
Métrica	Acurácia	0,6269	0,6341	0,6278	0,6279	0,6291	0,6288	0,6274	0,6347
	Precisão Macro	0,63	0,64	0,63	0,63	0,63	0,63	0,63	0,64
	Precisão Média	0,63	0,64	0,63	0,63	0,63	0,63	0,63	0,64
	Recall Médio	0,63	0,63	0,63	0,63	0,63	0,63	0,63	0,63
	F1-*Score*	0,62	0,63	0,62	0,62	0,63	0,63	0,62	0,63

Adicionalmente, um outro processo de balanceamento da base de dados original foi realizado, porém empregando-se *undersampling*. Há diversas técnicas na literatura para redução de uma classe majoritária, tais como selecionar um subconjunto da classe majoritária randomicamente, selecionar exemplos da classe majoritária conforme sua distância em relação à outra classe, agrupar os exemplos da base de dados e selecionar os agrupamentos que possuem uma relação semelhante entre as classes existentes, entre outras [Yen and Lee 2009]. As técnicas mais comuns removem aleatoriamente exemplos da classe minoritária até atingir o limite desejado, o que pode causar perda de informação [Fernández et al. 2017]. Neste trabalho foi empregado o algoritmo *RandomUnderSampler* implementado na biblioteca *scikit-learn*. Ele executa uma redução da classe majoritária coletando exemplos da base de dados de forma aleatória, com ou seu reposição desses exemplos, até atingir a relação desejada dessa classe com a classe minoritária [Yen and Lee 2009]. Após esse procedimento, uma nova base dados foi criada, mantendo-se a mesma quantidade de características, 195. A classe de maus pagadores foi reduzida até ser igualada a de bons pagadores, também passando a contar com 45063 exemplos, o que é representado na Figura 7. Essa nova base foi dividida em 3 partes - treino, validação e testes - nas seguintes proporções 50%, 20% e 30%, respectivamente. A Tabela 4 apresenta os tamanhos dessas 3 novas bases criadas. O conjunto de resultados obtidos a partir da base reduzida com *RandomUnderSampler* é exibido na Tabela 5.

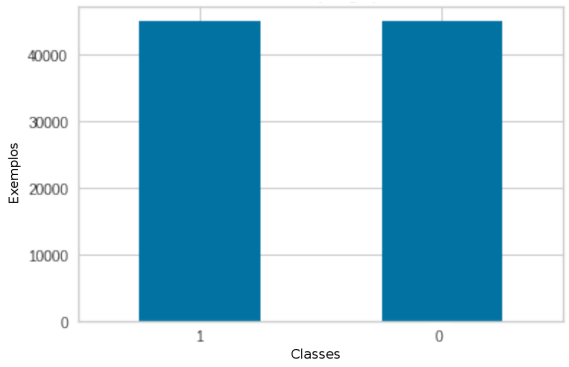


Figura 7. Distribuição das classes na base balanceada com *undersampling*

Tabela 4. Características das bases de dados balanceadas

Base de dados	**Tamanho (MB)**			
	Treino	Validação	Teste	Total
Balanceada	46,9	18,6	28	93,5

O processo de *undersampling* fez com que as novas bases construídas ficassem completamente distintas daquelas obtidas a partir da base desbalanceada e vistas na Tabela 1. Constata-se uma redução nos tamanhos das novas bases de treino, validação e teste. Enquanto o algoritmo genético reduziu o tamanho da base em 39%, o RFECV com o classificador *Random Forest* reduziu a base ainda mais, atingindo cerca de 80%. Em ambos os casos, a redução contribui para a melhoria no desempenho da predição dos bons pagadores. Em contrapartida, o RFECV com o classificador *Logistic Regression* apresentou um comportamento diametralmente oposto, o que é corroborado pela quantidade de características que ele selecionou. Enquanto na base desbalanceada ele selecionou 24 das 195 características disponíveis, conforme visto na Tabela 1, na base balanceada ele selecionou 86, como se observa na Tabela 6. Logo, ele fez com que a nova base gerada a partir da base balanceada fosse 35% maior que aquela obtida com a base desbalanceada.

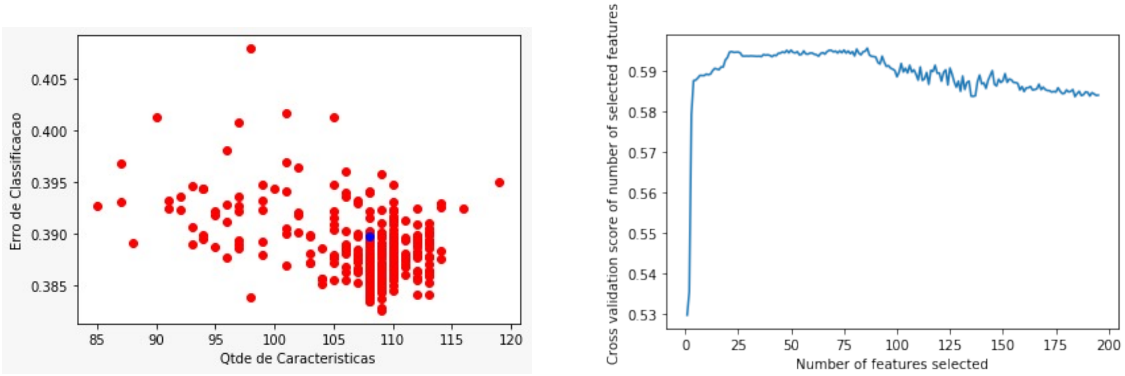
Tabela 5. *Voting Classifier* (RF, ADA, XGBC): Balanceamento *undersampling*

Classificador		**RFECV+RF**		**RFECV+LR**		**AG**		**RFECV+RF+F1**	
Base de Dados		Validação	Teste	Validação	Teste	Validação	Teste	Validação	Teste
Métrica	Acurácia	0,6270	0,6339	0,6285	0,6281	0,6293	0,6291	0,6269	0,6326
	Precisão Macro	0,63	0,64	0,63	0,63	0,63	0,63	0,63	0,64
	Precisão Média	0,63	0,64	0,63	0,63	0,63	0,63	0,63	0,64
	Recall Médio	0,63	0,63	0,63	0,63	0,63	0,63	0,63	0,63
	F1-*Score*	0,62	0,63	0,63	0,62	0,63	0,63	0,62	0,63

Tabela 6. Detalhes das novas bases de dados balanceadas

Algoritmo	Número de Características	Tamanho (MB)			
		Treino	Validação	Teste	Total
AG	91	21,5	8,6	12,8	42,9
RFECV+LR	86	17,5	7	10,5	35
RFECV+RF	31	9,1	3,5	5,3	17,9
RFECV+RF+F1-*Score*	37	11,1	6,5	4,3	21,9

Uma nova seleção de características foi realizada após o balanceamento das bases, empregando-se os mesmos algoritmos utilizados anteriormente e adicionou-se o RFECV com o classificador *Random Forest* dando prioridade ao F1-*Score* ao invés da acurácia. A Tabela 6 apresenta os resultados da seleção, destacando-se as o número de características mais importantes selecionadas por cada algoritmo. O comportamento desses algoritmos é demonstrado nas Figuras 8 e 9. Novas bases de dados - treino, validação e teste - foram criadas empregando-se as características selecionadas e nas mesmas proporções estabelecidas no enunciado do trabalho. A Tabela 6 sumariza os tamanhos dessas novas bases.

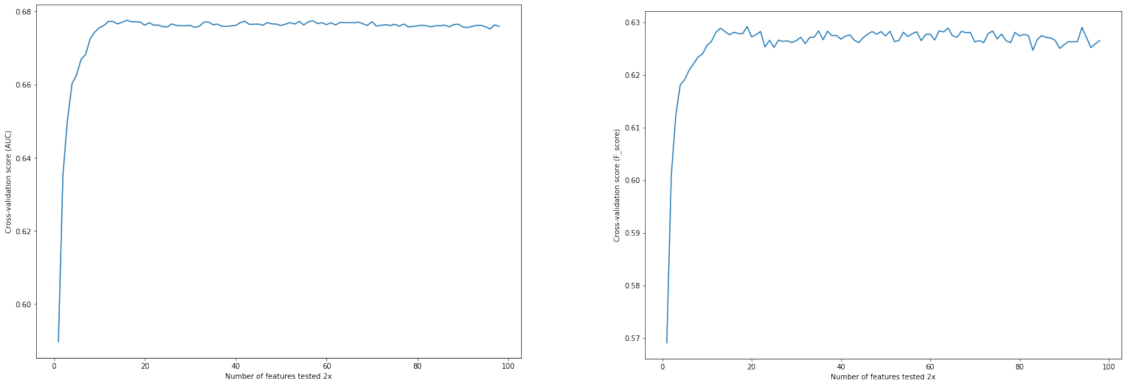


(a) AG (b) RFECV+LR

Figura 8. AG e RFECV+LR: Seleção de características

O *Voting Classifier* foi ajustado utilizando-se uma técnica distinta daquela aplicada na base desbalanceada. A mudança de estratégia é reflexo do aprimoramento dos conhecimentos adquiridos ao longo do desenvolvimento do trabalho, tanto em teoria quanto em prática. O emprego da técnica de *Grid Search* permitiu uma escolha otimizada dos parâmetros, porém elevou o tempo total de processamento, conforme ilustrado no trabalho [Feurer and Hutter 2019]. Essa técnica retorna o melhor conjunto de ajustes a partir de uma gama de parâmetros e seus respectivos valores oferecidos como entrada. Os melhores valores verificados pelo *Grid Search* para os parâmetros utilizados nos algoritmos selecionados estão relacionados na Tabela 7.

Os modelos otimizados foram inseridos no *Voting Classifier* configurados com os melhores parâmetros identificados pelo *Grid Search*. A seleção de votação também foi



(a) RFECV+RF (b) RFECV+RF+F1-*Score*

Figura 9. RFECV+RF: Seleção de características

Tabela 7. Melhores parâmetros obtidos com o *Grid Search*

Classificador	*Criterion*	*max_depth*	*n_estimators*	*learning_rate*	*objective*
RF	gini	8	300	-	-
ADA	-	-	600	0.4	-
XGBC	-	3	200	0.2	binary:logistic

alterada, visto que o comportamento entre os classificadores foi mais linear com as bases balanceadas. Dessa forma, foi feita a opção pelo *hard voting*, onde cada classificador vota e a maioria vence. Para um melhor funcionamento do modelo, o *Voting Classifier* foi composto com três classificadores: *Random Forest, ADABoost*, e *XGBC*. A Figura 10 exibe a proporção de votos por classificador.

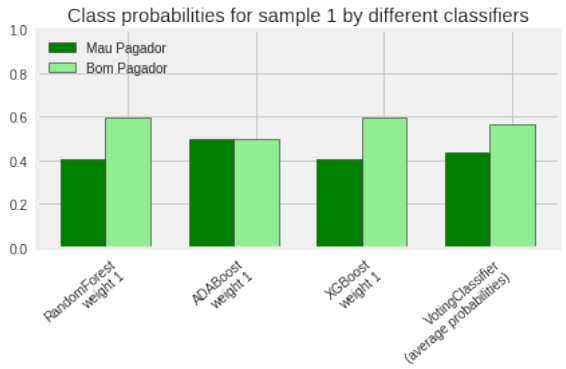


Figura 10. Votação dos Classificadores Após Ajuste

4. Discussão dos Resultados

A discussão de resultados será realizada a partir dos melhores modelos obtidos em cada uma das bases testadas - base desbalanceada, base balanceada com *oversampling* e base balanceada com *undersampling*. Resultados adicionais estão disponíveis no repositório do trabalho no *Github*, onde também se encontram todos os arquivos que permitem

a reprodução completa deste trabalho.

Conforme descrito na Seção 3, a análise foi iniciada pela base desbalanceada. O modelo que melhor representa essa base é o *Voting Classifier* com *stacking* de classificadores - *Random Forest, Decision Tree, AdaBoost e Extreme Gradient Boosting* - e utilizando a seleção de características do RFECV+RF. O Relatório 1 apresenta os resultados da classificação gerada pela *scikit-learn*.

Relatório 1. Resultados do *Voting Classifier* na base desbalanceada

```
Classification Report
              precision    recall  f1-score   support

           0       0.61      0.10      0.17     13442
           1       0.81      0.98      0.89     52553

    accuracy                           0.80     65995
   macro avg       0.71      0.54      0.53     65995
weighted avg       0.77      0.80      0.74     65995
```

Observando-se apenas a média ponderada, os critérios de avaliação são bons. Porém, os valores de *recall* e F1-*Score* para a classe negativa, representada pelos valores 0 - bons pagadores, são extremamente baixos. Tanto a precisão quanto o *recall* da classe positiva, representada por 1 - maus pagadores, são altos. Esse comportamento indica que o modelo é preciso ao classificar os mau pagadores (81%), além de errar muito pouco nos maus pagadores classificados, cerca de 2%. O valor de AUC obtido através da curva ROC também foi baixo, conforme ilustra a Figura 11. Define-se, então, que o problema apresentado no enunciado foi parcialmente resolvido. Entretanto, a classificação dos bons pagadores é de fundamental importância para qualquer instituição financeira, uma vez que os juros provenientes dos empréstimos pagos em dia são parte relevante da composição da renda total. Nesse sentido, constata-se que o sistema foi pouco eficiente na classificação dos bons pagadores. Logo, há necessidade de um modelo mais equilibrado, que classifique a classe positiva de maneira eficiente, mas que também tenha competência em identificar a classe negativa.

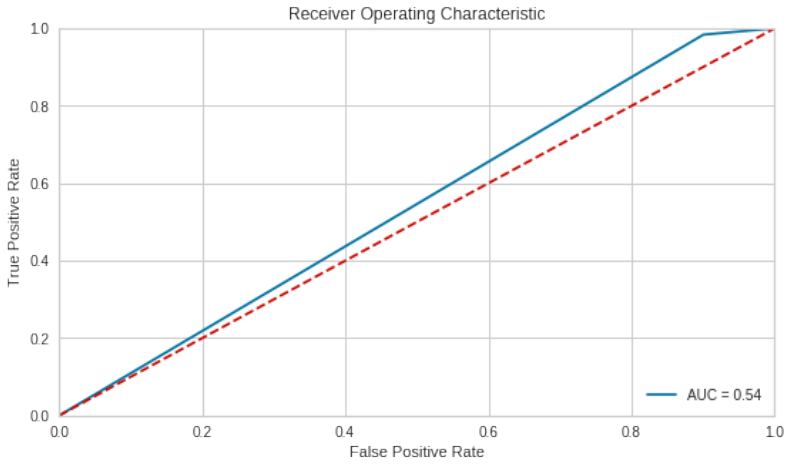


Figura 11. Curva ROC e valor AUC para base desbalanceada

Visando melhorar a classificação da classe com menos exemplos, foi realizada a equalização da base, permitindo aos classificadores distinguirem melhor as duas classes. Esse processo foi realizado através da aplicação da técnica de *undersampling* na base desbalanceada, que foi descrito na Seção 3. Com a base balanceada, o melhor resultado foi obtido com o *Voting Classifier* com *stacking* de classificadores - *Random Forest, Adaptive Boosting e Extreme Gradient Boosting*) - e usando a seleção de características RFECV+RF e F1-*Score* como parâmetro.

O Relatório 2 apresenta os resultados de classificação para a base balanceada. Nota-se que a precisão do modelo em reconhecer a classe positiva diminuiu, ao mesmo tempo que se manteve na classificação da classe negativa. Entretanto, observando-se o *recall* para ambas as classes, verifica-se que o sistema tem uma sensibilidade razoável para identificar a classe positiva e é ainda melhor na negativa, quando comparado ao modelo da classe desbalanceada. Enquanto sua sensibilidade era de 0,10 no modelo anterior, neste modelo atingiu o valor de 0,56. Importante ressaltar que as médias macro e ponderada exibem o mesmo valor no modelo atual, demonstrando seu equilíbrio.

Relatório 2. Resultados do *Voting Classifier* na base balanceada

```
Classification Report
              precision    recall  f1-score   support

           0       0.61      0.56      0.60     13530
           1       0.66      0.71      0.66     13508

    accuracy                           0.63     27038
   macro avg       0.64      0.63      0.63     27038
weighted avg       0.64      0.63      0.63     27038
```

A curva ROC e a área abaixo da curva (AUC) também foram empregadas para avaliar o modelo. Para a base balanceada com *RandomUnderSampler*, o valor da AUC foi superior, quando comparado ao melhor valor obtido com a base desbalanceada, conforme observado na Figura 12.

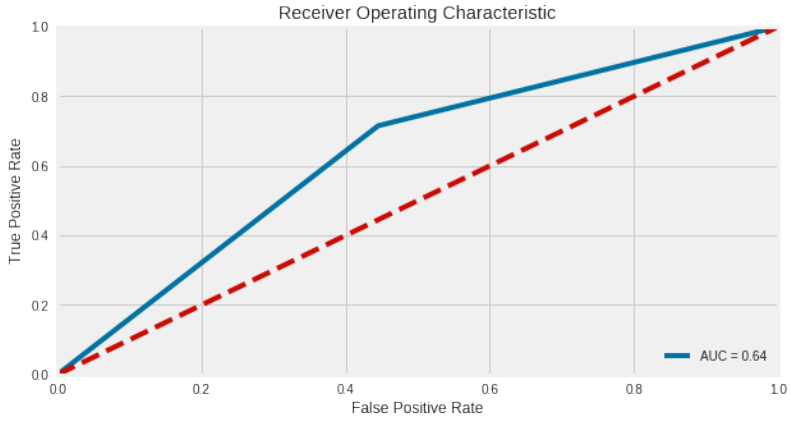


Figura 12. Curva ROC e valor AUC para base balanceada

Métricas como acurácia e curva ROC exigem cuidado na interpretação quando consideramos bases desbalanceadas [Ling et al. 2003]. Entretanto, ao avaliarmos o mo-

delo com a base balanceada com *undersampling*, notamos algumas características importantes. O modelo tem precisão razoável para ambas as classes, tem uma boa sensibilidade para identificar maus pagadores sem descartar os bons completamente, além de um bom valor de F1-*Score* médio e para cada classe individualmente. A Figura 13 ilustra a aprendizagem do classificador nas bases de treino e teste. Analisando-se as curvas de aprendizagem, nota-se que eventualmente o modelo poderia ser melhorado caso a base de treinamento fosse maior. Utilizando somente os dados da base original é impossível aumentar o número de exemplos da base de teste, uma vez que todos os exemplos da classe negativa foram utilizados e exemplos da classe positiva foram descartados.

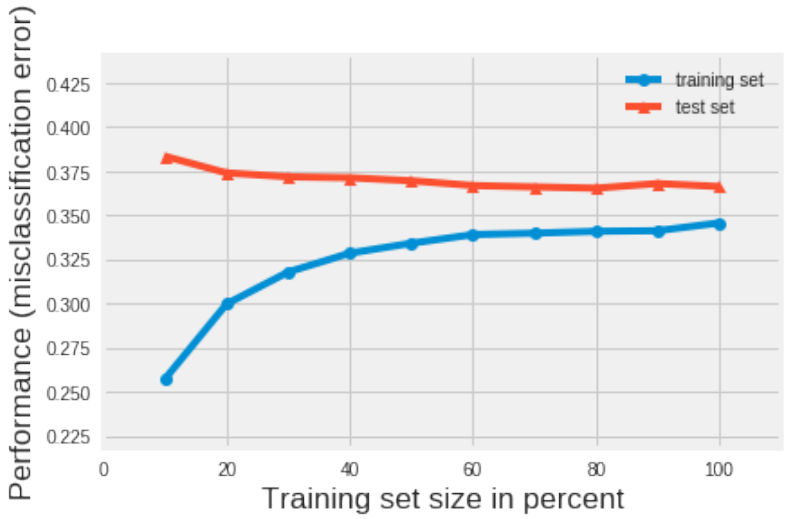


Figura 13. Base balanceada - Curvas de aprendizagem: Treino x Teste

Em nível experimental, foi empregada a técnica SMOTE para adicionar mais exemplos à base de treino, a fim de se avaliar o comportamento do modelo com uma base de treino maior. O uso dessa técnica não trouxe grandes benefícios ao modelo, como se observa no Relatório 3. Nota-se uma maior precisão na classe negativa e menor na positiva, significando que a adição de dados sintéticos privilegiou a classificação dos bons pagadores, mas prejudicou em alguma medida a classificação de maus pagadores. O F1-*Score* foi praticamente o mesmo, assim como a acurácia do sistema. A curva ROC apresentou um valor de AUC levemente inferior, conforme observado na Figura 14.

Relatório 3. Resultados do *Voting Classifier* na base com SMOTE

```
Classification Report
              precision    recall  f1-score   support

           0       0.66      0.53      0.59     13530
           1       0.61      0.72      0.66     13508

    accuracy                           0.63     27038
   macro avg       0.63      0.63      0.62     27038
weighted avg       0.63      0.63      0.62     27038
```

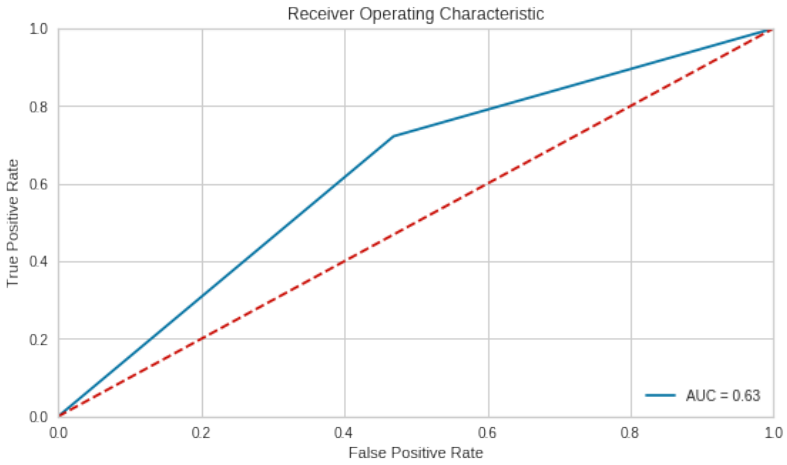


Figura 14. Curva ROC e valor AUC para base desbalanceada - SMOTE

5. Conclusão

A atividade de classificação de clientes inadimplentes exige, atualmente, um elevado esforço das instituições envolvidas, visto que o volume de dados disponível é muito grande. Isso impede sua execução exclusivamente por pessoas e exige a utilização de recursos computacionais para ter eficiência e atenda às necessidades das instituições financeiras com segurança e rapidez. Nesse contexto, o emprego de algoritmos de aprendizagem de máquina supervisionados podem ser muito úteis.

Neste trabalho foram empregados algoritmos de aprendizagem de máquina supervisionados para classificação de clientes inadimplentes, conforme estabelecido no seu enunciado. Nesse contexto, verificou-se que o emprego sobre a base desbalanceada do *Voting Classifier* com *stacking* de classificadores - *Random Forest, Decision Tree, AdaBoost e Extreme Gradient Boosting* - e utilizando a seleção de características RFECV+RF apresentou o melhor resultado. Ele consegue classificar os maus pagadores com 81% de precisão e com 2% de erro. Logo, ele atende ao solicitado no enunciado. Porém, a classificação dos bons pagadores é muito baixa, 10%. Portanto, os experimentos foram estendidos em busca de um modelo que atendesse de forma mais igualitária as classes de bons e maus pagadores.

O modelo que melhor atendeu a classificação de ambas as classes foi o *Voting Classifier* com *stacking* de classificadores - *Random Forest, Adaptive Boosting e Extreme Gradient Boosting*) - e usando a seleção de características RFECV+RF e F1-*Score* como parâmetro. A dimensionalidade da base de dados foi reduzida para 37 características. Ele foi empregado sobre uma base balanceada com a técnica de *undersampling*, o que reduziu seu tamanho e proporciou uma execução rápida e eficiente. Este modelo classificou a classe de bons pagadores com 60% de precisão e a classe de maus pagadores, 66%. Além disso, a sensibilidade para os bons pagadores atingiu 56%, enquanto para os maus pagadores ficou em 71%. A acurácia do modelo foi de 63% e é uma métrica adequada para esse caso, visto que a base foi balanceada. Dessa forma, neste trabalho esse modelo é considerado o mais adequado para atender à classificação de clientes de uma maneira mais ampla.

Como trabalhos futuros, identifica-se algumas possibilidades de melhorias nos

modelos apresentados por meio de várias ações. Dentre elas, tem-se a reclassificação daqueles exemplos que foram classificados incorretamente pelos modelos, tanto falsos positivos, como falsos negativos. Outra linha de trabalho seria implementar um controle de rejeição, que interromperia a classificação quando ela atingisse um nível pré-estabelecido. Finalmente, uma última oportunidade de trabalho futuro seria testar outras soluções de *Grid Search*, especialmente aquelas que gerem valores de forma automatizada, teste-os e verifique quais são os melhores.

Referências

Desai, V. S., Conway, D. G., Crook, J. N., and Overstreet Jr, G. A. (1997). Credit-scoring models in the credit-union environment using neural networks and genetic algorithms. *IMA Journal of Management Mathematics*, 8(4):323–346.

Fernández, A., del Río, S., Chawla, N. V., and Herrera, F. (2017). An insight into imbalanced big data classification: outcomes and challenges. *Complex & Intelligent Systems*, 3(2):105–120.

Feurer, M. and Hutter, F. (2019). Hyperparameter optimization. In *Automated Machine Learning*, pages 3–33. Springer.

Ling, C. X., Huang, J., and Zhang, H. (2003). Auc: a better measure than accuracy in comparing learning algorithms. In *Conference of the canadian society for computational studies of intelligence*, pages 329–341. Springer.

Ma, L., Zhao, X., Zhou, Z., and Liu, Y. (2018). A new aspect on p2p online lending default prediction using meta-level phone usage data in china. *Decision Support Systems*, 111:60–71.

Maldonado, S., Weber, R., and Famili, F. (2014). Feature selection for high-dimensional class-imbalanced data sets using support vector machines. *Information Sciences*, 286:228–246.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Sadatrasoul, S., Gholamian, M., Shahanaghi, K., and Sadatrasoul, S. M. (2015). Extracting rules from imbalanced data: The case of credit scoring. *Journal of Information Systems and Telecommunication*, 3:22–28.

scikit learn (2019). scikit-learn - Machine Learning in Python. https://scikit-learn.org/stable/. [Online]. Acessado em Nov. 2019.

Yen, S.-J. and Lee, Y.-S. (2009). Cluster-based under-sampling approaches for imbalanced data distributions. *Expert Systems with Applications*, 36(3):5718–5727.

Zhang, W., He, H., and Zhang, S. (2019). A novel multi-stage hybrid model with enhanced multi-population niche genetic algorithm: An application in credit scoring. *Expert Systems with Applications*, 121:221–232.