Avaliação empírica de classificadores e métodos de balanceamento para detecção de fraudes em transações com cartões de créditos

Victor Gomes de Oliveira Martins Nicola[1], **Marcelo de Souza Lauretto**[1], **Karina Valdivia Delgado**[1]

[1]Escola de Artes, Ciências e Humanidades – Universidade de São Paulo (USP)
Rua Arlindo Bettio, 1000 – 03828-000 – São Paulo – SP – Brasil

victor.nicola@usp.br, marcelolauretto@usp.br, kvd@usp.br

Abstract. *Machine learning algorithms are widely used in credit card fraud detection systems due to their ability to distinguish between legitimate and fraudulent transactions. A known problem in this field is the high imbalance usually found in the classes, which can compromise the performance of the classifiers. The empirical studies found in the literature apply, at most, two sampling techniques. This article presents a comparative study of five classification models under five different methods of balancing the training sets. The best performance was obtained by random forest, which in addition to having the highest average F-score (*$= 0.867$*), proved to be considerably more robust than the other classifiers in relation to the choice of the balancing technique and attribute selection.*

Resumo. *Algoritmos de aprendizado de máquina são amplamente utilizados em sistemas para detecção de fraudes em cartões de crédito devido à capacidade de distinguir entre transações legítimas e fraudulentas. Um problema reconhecido nesta área é o alto desbalanceamento usualmente encontrado nas classes, que pode comprometer o desempenho dos classificadores. Os estudos empíricos encontrados na literatura aplicam, no máximo, duas técnicas de amostragem. Este artigo traz um estudo comparativo de cinco modelos de classificação sob cinco diferentes métodos de balanceamento dos conjuntos de treinamento. O melhor desempenho foi obtido pela random forest, que além de ter apresentado o maior F-score médio (*$= 0,867$*), mostrou-se consideravelmente mais robusta do que os demais classificadores em relação à escolha da técnica de balanceamento e à seleção de atributos.*

1. Introdução

A detecção de fraudes em cartões de crédito não é um problema novo e estudos como [Chan et al. 1999] mostram que já era uma preocupação desde o início do comércio eletrônico. Entretanto, o relatório da revista Nilson (do inglês, *News and Statistics for Card and Mobile Payment Executives*[1]) estima que as fraudes reportadas por operadoras de cartões de crédito totalizaram 19.21 bilhões de dólares, em 2018, um aumento de 14.48% em relação a 2017. Isto mostra que o problema ainda não foi resolvido e que a detecção desse tipo de fraudes é de grande interesse do mercado.

[1]https://nilsonreport.com/mention/407/1link/

Os problemas de escalabilidade, eficiência e desbalanceamento do conjunto de dados também são conhecidos há muito tempo para este tipo de detecção de fraudes. Por isso, é possível encontrar estudos que comparam modelos treinados sob conjuntos de dados desbalanceados ou que utilizam alguma técnica de balanceamento para detecção de fraudes em cartões de crédito. Todavia, os estudos encontrados concentram-se quase exclusivamente na comparação entre classificadores, sem considerar métodos de balanceamento ou, no máximo, limitando-se a um ou dois métodos. Por exemplo, em [Khatri et al. 2020] é realizado um estudo comparativo de diferentes técnicas de aprendizado de máquina, porém sob o conjunto original desbalanceado. Já em [Sahin and Duman 2011], [Dhankhad et al. 2018], [Mishra and Ghorpade 2018], [Niu et al. 2019] e [Varmedja et al. 2019], utiliza-se uma única técnica de balanceamento, enquanto que em [Awoyemi et al. 2017] é utilizada uma combinação de duas técnicas.

Visto que alguns classificadores são sensíveis ao desbalanceamento dos dados de treinamento, uma questão adicional é decidir a adequação de se usar técnicas de rebalanceamento, bem como a escolha da melhor técnica. A contribuição deste trabalho é realizar um estudo comparativo não apenas de classificadores, mas de diferentes técnicas de balanceamento, utilizando cinco classificadores sob seis diferentes configurações de balanceamento dos dados, para o problema de classificar transações de cartão de crédito como legítimas ou fraudulentas. O objetivo é analisar as diferenças entre os modelos obtidos com o conjunto original e os conjuntos balanceados, analisando o impacto do rebalanceamento no desempenho dos classificadores.

As técnicas de aprendizado de máquina avaliadas foram o *k Nearest Neighbors* (k-NN), [Altman 1992], *naïve Bayes* [Maron 1961], *Random Forest* [Breiman 2001], *Regressão Logística* [Neter et al. 1996] e *Support Vector Machine* (SVM) [Cortes and Vapnik 1995].

Tais modelos foram treinados tanto sobre o conjunto de dados original (desbalanceado) como também aplicando-se cinco diferentes abordagens de balanceamento – duas de subamostragem, duas de sobreamostragem e uma híbrida. As técnicas de subamostragem escolhidas foram *Random Undersampling* [Kuhn and Johnson 2013] e Geração de Protótipos com k-means [Ren and Yang 2019]. As técnicas escolhidas de sobreamostragem foram *Random Oversampling* [Dupret and Koda 2001] e *Synthetic Minority Oversampling Technique* (SMOTE) [Bowyer et al. 2011]. A abordagem híbrida considerada foi SMOTE seguido de ligações de TOMEK [Batista et al. 2004], que consiste na combinação de sobreamostragem seguida de subamostragem.

Todas essas técnicas foram avaliadas no conjunto de dados formado por transações reais feitas com cartões de crédito fornecido pelo ULB *Machine Learning Group* e fornecido por meio da plataforma kaggle[2].Os experimentos foram realizados por meio da validação cruzada *K-fold*. Sendo que, em cada iteração, usou-se o conjunto de teste original (sem balanceamento), para uma avaliação mais realista e próxima das condições reais de aplicação.

O artigo está organizado da seguinte forma. Na seção 2 é feita a análise de trabalhos correlatos, tendo a maioria utilizado o mesmo conjunto de dados deste estudo. A seção 3 apresenta a metodologia do trabalho, incluindo a descrição do conjunto de da-

[2]https://www.kaggle.com/mlg-ulb/creditcardfraud

dos, o protocolo seguido nos experimentos, as técnicas de balanceamento utilizadas e a calibração dos modelos. Por fim, a seção 4 discute os resultados, comparando os diferentes modelos.

2. Trabalhos correlatos

Esta seção apresenta estudos que compararam técnicas de aprendizado supervisionado para classificar transações como fraudulentas ou não, geralmente aplicando algum método de rebalanceamento dos dados.

No trabalho [Khatri et al. 2020], foram comparados modelos obtidos com árvore de decisão, k-NN, regressão logística, *random forest* e naïve Bayes em termos de tempo e precisão, utilizando o mesmo conjunto de dados que o presente trabalho. Foram obtidas taxas superiores com o k-NN, mas o elevado tempo de processamento fez com que o melhor modelo escolhido fosse o de árvore de decisão. Porém, nenhuma técnica de balanceamento foi aplicada.

Enquanto isso, a maioria dos trabalhos aplica apenas uma técnica de balanceamento, muitas vezes treinando e testando os modelos sobre o conjunto de dados balanceado. O trabalho [Sahin and Duman 2011] utilizou um conjunto de dados diferente para comparar sete modelos de máquinas de vetor de suporte e árvores de decisão, utilizando amostragem estratificada para balancear os dados e mostrou que as abordagens com árvore de decisão trouxeram melhores resultados que as SVM. O trabalho [Dhankhad et al. 2018] também utilizou os dados do *ULB Machine Learning Group*, utilizado nesse trabalho, mas aplicou uma técnica de subamostragem. Nele, foram comparados dez diferentes algoritmos e observou-se que *random forest* foi o melhor, isoladamente, mas indicou que o desempenho deste algoritmo poderia ser superado por uma pilha de classificadores. Já o artigo de [Mishra and Ghorpade 2018] realizou testes neste mesmo conjunto de dados, balanceando com a técnica de *random undersampling* e utilizando modelos de regressão logística, SVM, *Gradient Boosting*, *random forest* e uma pilha de classificadores. Ele também mostrou que o desempenho da *random forest* foi superior aos demais tanto no conjunto desbalanceado (original) quanto no conjunto rebalanceado.

Em [Varmedja et al. 2019] também foi utilizado o mesmo conjunto de dados. Foi aplicada a técnica SMOTE para sobreamostragem e feita uma seleção de atributos para comparar os algoritmos de regressão logística, Naïve Bayes, *random forest* e *multilayer perceptron*. Foi encontrado que a *random forest* teve os melhores resultados. O trabalho [Niu et al. 2019] fez uma comparação entre algoritmos de aprendizado supervisionado e não-supervisionado sob os mesmos dados, mas balanceados com uma técnica de subamostragem. Esse estudo afirma que a abordagem supervisionada trouxe resultados melhores, mas é dependente dos rótulos dos dados e que uma abordagem não-supervisionada é promissora e não sofre dessa desvantagem. O artigo [Awoyemi et al. 2017] rebalanceia os dados com uma técnica de sobreamostragem combinada com uma de subamostragem, utilizando o mesmo conjunto de dados que o usado aqui. Neste estudo foram comparados os modelos de naïve Bayes, k-NN e regressão logística, encontrando que o k-NN obteve o melhor desempenho.

3. Metodologia

Esta seção descreve os experimentos realizados e está divida em cinco partes. A primeira subseção descreve o conjunto de dados utilizado. A segunda subseção apresenta as técnicas de rebalanceamento aplicadas. A terceira mostra o algoritmo de eliminação recursiva, que foi utilizado para realizar a seleção de atributos, bem como o conjunto de atributos selecionados. A quarta subseção fala sobre a calibração dos parâmetros dos classificadores. Por fim, a quinta subseção descreve os testes de validação cruzada e significância.

A implementação foi realizada por meio da linguagem Python (versão 3.8) e toda a implementação dos classificadores e técnicas de rebalanceamento foi feita com a biblioteca scikit-learn [Pedregosa et al. 2011]. Os testes foram realizados em um ambiente em nuvem disponibilizado pela Google gratuitamente, o Google Colaboratory, que disponibiliza uma máquina com um processador Intel(R) Xeon(R) CPU @ 2.30GHz e 12 Gb de memória RAM.

3.1. Conjunto de dados

O conjunto de dados escolhido foi fornecido pelo ULB *Machine Learning Group* e contém 284.807 transações que ocorreram em dois dias de setembro de 2013, realizadas na Europa. O conjunto é altamente desbalanceado, com apenas 0,172% de transações fraudulentas. Possui 28 atributos numéricos contínuos, derivados das variáveis originais por meio da análise de componentes principais. A descrição dos atributos foi omitida e os rótulos são genéricos - V1, V2, V3, ... V28. Além disso, possui um atributo numérico que indica a ordem em que as transações ocorreram no tempo e outro atributo numérico com o valor original da transação - totalizando 30 atributos. Outros detalhes sobre os registros, incluindo as variáveis originais, não são fornecidos.

3.2. Técnicas de balanceamento

A seguir, são discutidas brevemente as cinco técnicas de balanceamento utilizadas neste estudo.

Random undersampling [Kuhn and Johnson 2013]: A subamostragem é feita por sorteio simples de instâncias da classe majoritária até atingir o balanceamento desejado, podendo ser feita com ou sem reposição. Neste trabalho, utilizamos a amostragem sem reposição, dada a alta prevalência de transações legítimas.

Protótipos gerados com clusterização [Ren and Yang 2019]: Realiza a subamostragem, substituindo um conjunto de instâncias da classe majoritária por um centroide, por meio de um algoritmo de clusterização. Para isto, foi escolhido o *k-means*.

Random oversampling [Dupret and Koda 2001]: Esta técnica consiste em selecionar elementos da classe minoritária aleatoriamente e duplicá-los para atingir a proporção desejada.

Synthetic Minority Oversampling Technique (SMOTE) [Bowyer et al. 2011]: Esta técnica consiste em aumentar o número de instâncias de treinamento da classe minoritária, criando dados sintéticos. De maneira simplificada, para cada exemplo da classe minoritária, são encontrados seus k vizinhos mais próximos (de mesma classe) no espaço vetorial de características. Alguns desses k vizinhos são então

sorteados. Finalmente, gera-se aleatoriamente um novo ponto (ou seja, um novo vetor de características sintético) dentro de cada segmento de reta, ligando o exemplo considerado e seus vizinhos sorteados.

SMOTE combinado com ligações de TOMEK [Batista et al. 2004]: Esta abordagem combina a técnica de sobreamostragem (SMOTE) seguida de outra para subamostragem (ligações de TOMEK [Tomek 1976]). Pode-se dizer que uma ligação de Tomek existe entre duas instâncias se elas forem as vizinhas mais próximas uma da outra. A subamostragem por ligação de Tomek consiste em identificar instâncias que possuem essas ligações e eliminar uma das duas observações, repetindo o processo até atingir o balanceamento desejado. Nessa abordagem combinada, a classe minoritária é sobreamostrada e em seguida os elementos com ligação de TOMEK em ambas as classes são removidos.

3.3. Seleção de atributos

Como comentado anteriormente, foram realizados experimentos com todos os 30 atributos para descrever uma transação e com 9 atributos selecionados pela eliminação recursiva. A eliminação recursiva consiste em utilizar um estimador que atribua pesos a diferentes atributos para selecionar as características de maior relevância, e reavaliar para um conjunto de atributos menor, de maneira recursiva.

Para este trabalho, o estimador escolhido foi o de regressão logística e a validação cruzada também foi realizada dez vezes em cada combinação de atributos. A melhor combinação de atributos escolhida pelo modelo foi: [V4, V5, V8, V9, V10, V11, V12, V14, V16].

3.4. Calibração de parâmetros

Os parâmetros foram explorados por meio de diversos testes para k-NN e a *random forest* e detalhados nas subseções a seguir. Para os demais classificadores, foi adotada a parametrização de estudos descritos na literatura ou mantendo os valores padrão da biblioteca utilizada.

3.4.1. Parâmetros do k-NN

Uma vez que o valor de k pode impactar muito no resultado final do modelo, foi necessário calibrar esse parâmetro. Assim, foram realizados testes com $1 \leq k \leq 20$ tanto para o conjunto desbalanceado, quanto para o conjunto de dados balanceado com cada uma das 5 técnicas de balanceamento, também utilizando a validação cruzada de 10 partes. Foi possível observar que as abordagens de subamostragem trouxeram resultados expressivamente inferiores e que os melhores resultados foram obtidos com *random oversampling*. Em todos os casos, a partir de $k = 2$, há perda no desempenho do classificador. Como em cinco (dos seis conjuntos de dados testados) o valor $k = 2$ foi o melhor, esse valor foi escolhido para os testes finais.

3.4.2. Parâmetros da *Random Forest*

Na tabela 1 estão os nove parâmetros com os valores testados em cada um. Optou-se por variar cada parâmetro de maneira independente, fixando os outros, pois o número

Parâmetro	Valores
Bootstrap	*True* ou *False*
Critério de divisão dos nós	gini e *entropy*
Profundidade máxima	$[1, 2, 4, 8, 16, 32, None]$
Número máximo de atributos	$[2, 4, 5, 6, 8, 10, 12, 15, 20, 25, 30]$
Número máximo de nós terminais	$[10, 100, 500, 1000, 2000, 10000, None]$
Mínimo decaimento da impureza	$[0, 0, 00005, 0, 0001, 0, 0005, 0, 001, 0, 005, 0, 01, 0, 05, 0, 1, 0, 2, 0, 3, 0, 4, 0, 5, 0, 6, 0, 7, 0, 8, 0, 9]$
Número mínimo de amostras em um nó terminal	$[1, 2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 9192]$
Número mínimo de amostras para dividir um nó	$[2, 4, 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 9192]$
Número de estimadores	$[100, 150, 200, 250, 500, 750, 1000, 1500]$

Tabela 1. Valores de parâmetros testados para *Random Forest*

Parâmetro	Original	Random Undersampling	Protótipos	Random Oversampling	SMOTE	SMOTE Tomek	Final
Bootstrap	*True*	*True*	*True*	*True*	*True*	*True*	*True*
Critério	*entropy*	*entropy*	*entropy*	*entropy*	*entropy*	*entropy*	*entropy*
Prof. Max.	*None*	*None*	*None*	4	*None*	4	*None*
Max. Atr.	8	5	30	8	5	4	8
Max. Folhas	*None*	*None*	*None*	*None*	*None*	*None*	*None*
Decaimento	0	0,2	0,2	0	0	0	0
Amostras por folha	1	128	128	1	1	1	1
Amostras para divisão	2	2	2	2	2	2	2
Estimadores	150	150	250	250	100	250	250

Tabela 2. Melhor valor dos parâmetros do *Random Forest* em cada conjunto de dados. A última coluna representa os valores dos parâmetros selecionados para os testes finais

de combinação de parâmetros torna a busca exaustiva inviável para a configuração da máquina utilizada.

O desempenho foi analisado em termos do F-Score médio das validações. A tabela 2 mostra o melhor valor dos parâmetros em cada um dos conjuntos de dados. A combinação escolhida para os testes finais (última coluna da tabela) foi obtida por meio de voto da maioria. Por exemplo, o decaimento da impureza com valor zero trouxe os melhores resultados nos testes em quatro dos seis conjuntos e por isso esse foi o valor escolhido para os testes finais.

3.4.3. Parâmetros dos demais classificadores

Para a SVM, foi escolhido um kernel linear e tal escolha foi baseada em [Mishra and Ghorpade 2018] e [Sahin and Duman 2011]. Em [Mishra and Ghorpade 2018] foi usado o mesmo conjunto de dados e foi considerado que devido a alta dimensionalidade, um kernel linear seria melhor. Em [Sahin and Duman 2011] foram comparados diferentes kernels aplicados ao problema de detecção de fraudes e foi demonstrado que todos possuem um desempenho similar. O primeiro trabalho utilizou c = 10 para o kernel linear. No entanto, no segundo foi feita uma busca exaustiva e foi encontrado que o melhor parâmetro de regularização foi c =

100. Dado que o conjunto de dados utilizado no segundo fora o mesmo, foi decidido que esse seria o valor utilizado nos experimentos deste trabalho.

Para a regressão logística, decidiu-se aplicar a função de regularização L2 para prevenir o *overfitting*. O solucionador escolhido foi o *Stochastic Average Gradient* (SAG) [Schmidt et al. 2017], pois costuma ser mais eficiente para conjuntos de dados com muitas instâncias. Os outros parâmetros da função foram mantidos conforme o padrão disponibilizado pela biblioteca. Para o algoritmo de Naïve Bayes, foi considerada a distribuição Gaussiana, pois os atributos utilizados não são categóricos nem binários, todos são valores numéricos.

3.5. Validação cruzada e teste de significância

Todos os testes foram realizados por meio da validação cruzada 10-*fold*, na qual o treinamento era feito agrupando nove subconjuntos e deixando um para teste.

Dois cuidados importantes foram tomados:

- Cada técnica de balanceamento foi aplicada sobre os mesmos subconjuntos da validação cruzada, e cada algoritmo foi avaliado sobre os mesmos conjuntos de treinamento (balanceados ou não). Em outras palavras, as observações dos experimentos foram todas pareadas, para evitar que variações nos particionamentos dos dados originais pudessem interferir na comparação entre os resultados;
- Os balanceamentos foram aplicados somente sobre os subconjuntos de treinamento, e os desempenhos foram medidos sobre os conjuntos de testes originais (sem balanceamento), para permitir uma avaliação mais próxima das condições reais de aplicação dos classificadores.

Para a comparação entre os desempenhos de cada classificador sobre cada uma das configurações de balanceamento e de seleção de atributos, foi aplicado um procedimento de teste de permutação [Hesterberg et al. 2003]. Aqui apresentamos brevemente o procedimento básico.

Considere que o problema seja avaliar a significância das diferenças entre os valores observados de F-Score sob duas configurações A e B, onde cada configuração corresponde a um elemento do conjunto cartesiano $\{\text{classificadores}\} \times \{\text{métodos de balanceamento}\} \times \{\text{conjunto completo/reduzido de atributos}\}$. Para essa tarefa, a hipótese nula de interesse é de que as médias das distribuições populacionais (desconhecidas) do F1-Score sob as condições A e B são iguais.

Uma vez que os desempenhos observados são pareados (ou seja, são baseados nas mesmas partições do conjunto de dados original), basta analisar as diferenças entre os F-Scores obtidos pelas configuração A e B em cada uma das 10 iterações da validação cruzada; essas diferenças são denotadas por $d_1, d_2, \ldots d_{10}$. A estatística de interesse é a média dessas diferenças, $\bar{d} = \sum_{i=1}^{10} d_i \, /10$.

O passo seguinte é construir a distribuição que a estatística $\bar{d}$ teria sob a hipótese nula, ou seja, se não houvesse diferença entre as médias das distribuições populacionais do F1-Score sob as configurações A e B. No teste de permutação, essa distribuição é construída a partir dos próprios resultados observados. Isso é feito com base no princípio de que, se é verdadeira a hipótese de que os os desempenhos médios teóricos sob as configurações A e B são equivalentes, então valores positivos ou negativos entre

$d_1, d_2, \ldots d_{10}$ são meramente devidos ao acaso. Portanto, alguns sinais de $d_1, d_2, \ldots d_{10}$ poderiam ser trocados, o que seria equivalente a intercambiar alguns valores do F-Score entre as configurações A e B.

O parágrafo acima fornece a motivação para o seguinte procedimento:

1. Para cada subconjunto B do conjunto de índices $\{1, 2, \ldots, 10\}$
   a) Defina $d'_1, d'_2, \ldots, d'_{10}$ como:
   $d'_i = (-1)\ d_i$ se $i \in B$; caso contrário, $d'_i = d_i$;
   b) Calcule a média $\bar{d}^B = \sum_{i=1}^{10} d'_i\ /10$
2. Calcule o *p-valor* do teste:

$$pv = \frac{\sum_B I(|\bar{d}^B| \geq |\bar{d}|)}{2^{10}},$$

em que I denota a função indicadora: $I(p) = 1$ se p é verdadeiro e $I(p) = 0$ caso contrário

No passo 1, são calculadas todas as possíveis médias $\bar{d}^B$ que podem ser obtidas com as trocas de sinais em $d_1, \ldots, d_{10}$. No passo 2, a evidência a favor da hipótese nula (*p-valor*) é computada pela proporção das médias $\bar{d}^B$ com valores absolutos iguais ou mais extremos do que a estatística original $\bar{d}$. Se essa evidência é baixa ($pv < 0,1$), considera-se que a diferença observada entre os desempenhos sob as configurações A e B é significante, e portanto rejeita-se a hipótese de equivalência entre os desempenhos.

4. Resultados e discussão

Esta seção apresenta os resultados dos experimentos e está dividida em duas partes. A primeira subseção analisa a sensibilidade de cada classificador em relação ao balanceamento do conjunto de treino e à seleção dos atributos. Na segunda subseção, os classificadores são comparados entre si sob as diferentes técnicas de balanceamento. Nesta seção, as configurações de balanceamento serão abreviadas por siglas: DB - desbalanceado (original); PK - protótipos com k-means; RU - *random undersampling*; RO - *random oversampling*; SM - SMOTE, ST - SMOTE seguido de ligações de Tomek.

4.1. Análise de sensibilidade dos classificadores

A tabela 3 apresenta as médias (e os desvios padrão) do F-score em cada uma das configurações de classificador, balanceamento e seleção de atributos (completo/reduzido, ver subseção 3.3). Para cada classificador, é indicada a configuração (balanceamento, seleção de atributos) com o maior F-Score médio (células em negrito). São apresentadas também as configurações cujo F-Score médio não apresentou diferença significante ($pv > 0,1$) em relação ao F-Score ótimo (células sublinhadas).

O k-NN teve seus melhores desempenhos nos balanceamentos DB e RO, sob o conjunto reduzido de atributos. Na sequência, aparecem o SM e o ST, também sobre o conjunto reduzido de atributos, com desempenho 11% inferior ao das configurações ótimas. Observa-se, de forma geral, três resultados importantes: (i) os desempenhos obtidos sob o uso dos atributos completos foram muito baixos para todas as configurações de balanceamento, o que sugere que este classificador é bastante sensível à alta dimensionalidade dos dados ou a atributos com baixo poder discriminante; (ii) quando combinados

Técnicas de balanceamento	k-NN		Naïve Bayes	
	Completo	Reduzido	Completo	Reduzido
Desbalanceado (DB)	0,17 (0,012)	**0,83 (0,011)**	0,24 (0,010)	0,20 (0,005)
Random undersampling (RU)	0,01 (0,000)	0,15 (0,008)	0,21 (0,010)	0,16 (0,013)
Protótipos com kmeans (PK)	0,01 (0,001)	0,01 (0,000)	**0,37 (0,013)**	0,09 (0,008)
Random oversampling (RO)	0,30 (0,022)	0,83 (0,012)	0,21 (0,007)	0,15 (0,003)
SMOTE (SM)	0,08 (0,002)	0,72 (0,012)	0,25 (0,007)	0,16 (0,004)
SMOTE + Tomek (ST)	0,08 (0,003)	0,72 (0,012)	0,25 (0,007)	0,16 (0,004)

Técnicas de balanceamento	Random Forest		Regressão Logística	
	Completo	Reduzido	Completo	Reduzido
Desbalanceado (DB)	**0,87 (0,011)**	0,86 (0,008)	0,00 (0,000)	**0,71 (0,015)**
Random undersampling (RU)	0,12 (0,008)	0,08 (0,006)	0,01 (0,000)	0,12 (0,008)
Protótipos com kmeans (PK)	0,00 (0,000)	0,01 (0,000)	0,00 (0,000)	0,02 (0,000)
Random oversampling (RO)	0,87 (0,010)	0,85 (0,010)	0,13 (0,004)	0,12 (0,002)
SMOTE (SM)	0,87 (0,010)	0,72 (0,009)	0,23 (0,009)	0,17 (0,003)
SMOTE + Tomek (ST)	0,86 (0,009)	0,70 (0,010)	0,23 (0,009)	0,17 (0,003)

Técnicas de balanceamento	SVM	
	Completo	Reduzido
Desbalanceado (DB)	0,14 (0,030)	**0,71 (0,052)**
Random undersampling (RU)	0,05 (0,040)	0,14 (0,013)
Protótipos com kmeans (PK)	0,15 (0,043)	0,01 (0,000)
Random oversampling (RO)	0,12 (0,041)	0,14 (0,008)
SMOTE (SM)	0,15 (0,052)	0,18 (0,007)
SMOTE + Tomek (ST)	0,16 (0,037)	0,16 (0,004)

Legenda: Célula em negrito: F1-Score ótimo; célula sublinhada; F1-Score similar ao ótimo

Tabela 3. Médias (e desvios padrão) do F-score obtidas em cada configuração de classificador, balanceamento e seleção de atributos

com o conjunto reduzido de atributos, o uso do conjunto original (DB) ou das técnicas de sobreamostragem ou mista tendem a apresentar resultados ótimos ou, pelo menos, aceitáveis; (iii) a existência de apenas uma configuração ter apresentado F-Score estatisticamente equivalente ao ótimo sugere uma alta sensibilidade do classificador em relação às configurações de balanceamento e seleção de atributos adotadas.

O naïve Bayes teve seu F-Score ótimo na configuração PK sob o conjunto completo de atributos. Porém, seu desempenho ótimo foi muito inferior aos observados nos demais classificadores. Adicionalmente, nenhuma outra configuração teve desempenho comparável com o ótimo, o que pode sugerir uma alta sensibilidade do modelo em relação à escolha das configurações. De forma geral, em todas as configurações de balanceamento, o uso do conjunto completo de atributos neste classificador apresentou resultados melhores do que usando-se o conjunto reduzido.

A *Random Forest* teve seu melhor desempenho na configuração DB e conjunto completo de atributos. As configurações que apresentaram F-Score médios estatisticamente equivalentes foram: RO, SM, ST sob o conjunto de atributos completo; e DB sob o conjunto reduzido. Em sexto lugar, aparece RO com conjunto reduzido, o qual, embora não tenha sido considerado como estatisticamente equivalente, apresentou desempenho praticamente comparável com os primeiros. É válido notar que essas seis configurações apresentaram desempenhos superiores aos F-Scores ótimos de todos os demais classifica-

dores. Nas configurações SM e ST sob o conjunto de atributos reduzido, obteve desempenhos similares àqueles obtidos pela k-NN nas mesmas configurações. Esses resultados sugerem que, de forma geral, a *Random Forest* mostra-se menos sensível à parametrização entre configuração de amostragem e do conjunto de atributos, exceto pelas configurações de subamostragem (RU e PK), nas quais seu desempenho é significativamente menor. Além disso, o conjunto completo de atributos também é mais apropriado para este classificador.

A regressão logística e a SVM tiveram comportamentos muito similares. Ambas obtiveram seus melhores desempenhos praticamente idênticos, sob a configuração DB e conjunto reduzido de atributos. Todavia, seus valores ótimos foram inferiores àqueles obtidos pelo k-NN e pela *random forest*. Adicionalmente, todas as demais configurações apresentam desempenhos muito inferiores em relação ao F-Score ótimo, o que sugere que esses classificadores também são bastante sensíveis à escolha da configuração de balanceamento e conjunto de atributos.

4.2. Comparação geral entre os classificadores

A tabela 4 apresenta as médias (e desvios padrão) do F-score para cada classificador e técnica de balanceamento. Para cada classificador, foi escolhida a melhor configuração dos atributos (conjunto completo/reduzido) identificada na subseção anterior. A tabela apresenta em negrito o maior F-Score médio entre todas as configurações (classificador, balanceamento). Também são apresentadas as configurações cujo F-Score médio não apresentou diferença significante ($pv > 0,1$) em relação ao F-Score ótimo (células sublinhadas).

Balanceamento	k-NN (Reduzido)	Naïve Bayes (Completo)	Random Forest (Completo)	Regressao Logística (Reduzido)	SVM (Reduzido)
	Média (d. p.)	Média (d. p.)	Média (d. p.)	Média (d. p.)	Média (d. p.)
DB	0,83 (0,011)	0,24 (0,010)	**0,87 (0,011)**	0,71 (0,015)	0,71 (0,052)
RU	0,15 (0,008)	0,21 (0,010)	0,12 (0,008)	0,12 (0,008)	0,14 (0,013)
PK	0,01 (0,000)	0,37 (0,013)	0,00 (0,000)	0,02 (0,000)	0,01 (0,000)
RO	0,83 (0,012)	0,21 (0,007)	0,87 (0,010)	0,12 (0,002)	0,14 (0,008)
SM	0,72 (0,012)	0,25 (0,007)	0,87 (0,010)	0,17 (0,003)	0,18 (0,007)
ST	0,72 (0,012)	0,25 (0,007)	0,86 (0,009)	0,17 (0,003)	0,16 (0,004)

Legenda: Célula em negrito: F1-Score ótimo; célula sublinhada; F1-Score similar ao ótimo

Tabela 4. Médias e desvios padrões de F-score obtidos com os classificadores em cada conjunto balanceado.

Como já observado na subseção anterior, a *Random Forest* foi o classificador que atingiu o maior F-score, e foi bastante consistente em atingir esse patamar em conjuntos de treino balanceamentos com sobreamostragem. Além disso, nenhum dos demais classificadores atingiu desempenhos estatisticamente equivalentes ao F-Score ótimo da *Random Forest*. Entre os demais classificadores, o único que teve F-Score médio comparável foi o k-NN; por outro lado, esse classificador parece pouco robusto em manter um bom desempenho sob outras configurações de balanceamento e seleção de atributos. A regressão logística e a SVM apresentaram desempenhos ótimos similares, porém inferiores aos da *Random Forest* e do k-NN. Em último lugar, o naïve Bayes teve desempenho muito inferior ao dos demais.

5. Conclusões e trabalhos futuros

Este trabalho apresentou um estudo comparativo de cinco técnicas de aprendizado supervisionado com seis diferentes configurações de balanceamento do conjunto de treinamento. Os melhores resultados foram obtidos com modelos de *random forest* no conjunto desbalanceado e nos balanceados baseados em sobreamostragem ou híbridos. Uma vantagem adicional desse classificador foi sua maior robustez em relação à escolha das configurações de balanceamento e seleção de atributos.

Foi observado que muitos trabalhos treinam e testam os modelos no conjunto de dados balanceado, o que pode mascarar o real impacto do balanceamento na capacidade de generalização dos modelos. Ao treinar com conjuntos balanceados e testar com o conjunto desbalanceado, notou-se uma tendência de degradação no desempenho em quase todos os casos quando o balanceamento foi feito por técnicas de subamostragem. A redução do numero de atributos foi positiva para três dos 5 classificadores e neutra para *random forest*, mostrando que este algoritmo é mais robusto do que os demais para lidar com a alta dimensionalidade.

Como sugestão de trabalhos futuros é possível: testar outros classificadores ou pilhas de classificadores; aprofundar os conhecimentos teóricos sobre as técnicas de balanceamento e os impactos na distribuição de probabilidades dos dados; testar outras técnicas de balanceamento e combinações de técnicas; e testar com um conjunto de dados de cartões de crédito diferente.

Referências

Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *The American Statistician*, 46(3):175–185.

Awoyemi, J. O., Adetunmbi, A. O., and Oluwadare, S. A. (2017). Credit card fraud detection using machine learning techniques: A comparative analysis. In *2017 International Conference on Computing Networking and Informatics (ICCNI)*, pages 1–9.

Batista, G. E. A. P. A., Prati, R. C., and Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.*, 6(1):20–29.

Bowyer, K. W., Chawla, N. V., Hall, L. O., and Kegelmeyer, W. P. (2011). SMOTE: synthetic minority over-sampling technique. *CoRR*, abs/1106.1813.

Breiman, L. (2001). Random forests. *Mach. Learn.*, 45(1):5–32.

Chan, P. K., Fan, W., Prodromidis, A. L., and Stolfo, S. J. (1999). Distributed data mining in credit card fraud detection. *IEEE Intelligent Systems and their Applications*, 14(6):67–74.

Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, 20(3):273–297.

Dhankhad, S., Mohammed, E., and Far, B. (2018). Supervised machine learning algorithms for credit card fraudulent transaction detection: A comparative study. In *2018 IEEE International Conference on Information Reuse and Integration (IRI)*, pages 122–125.

Dupret, G. and Koda, M. (2001). Bootstrap re-sampling for unbalanced data in supervised learning. *European Journal of Operational Research*, 134(1):141 – 156.

Hesterberg, T., Monaghan, S., Moore, D., Clipson, A., and Epstein, R. (2003). *Bootstrap Methods and Permutation Tests: Companion Chapter 18 to the Practice of Business Statistics*. W.H.Freeman and Company, New York.

Khatri, S., Arora, A., and Agrawal, A. P. (2020). Supervised machine learning algorithms for credit card fraud detection: A comparison. In *2020 10th International Conference on Cloud Computing, Data Science Engineering (Confluence)*, pages 680–683.

Kuhn, M. and Johnson, K. (2013). *Applied Predictive Learning*. Springer, New York, NY, USA.

Maron, M. E. (1961). Automatic indexing: An experimental inquiry. *J. ACM*, 8(3):404–417.

Mishra, A. and Ghorpade, C. (2018). Credit card fraud detection on the skewed data using various classification and ensemble techniques. In *2018 IEEE International Students' Conference on Electrical, Electronics and Computer Science (SCEECS)*, pages 1–5.

Neter, J., Kutner, M. H., Nachtsheim, C. J., and Wasserman, W. (1996). *Applied Linear Statistical Models*. Irwin.

Niu, X., Wang, L., and Yang, X. (2019). A comparison study of credit card fraud detection: Supervised versus unsupervised. *CoRR*, abs/1904.10604.

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.

Ren, H. and Yang, B. (2019). Clustering-based prototype generation for imbalance classification. In *2019 International Conference on Smart Grid and Electrical Automation (ICSGEA)*, pages 422–426.

Sahin, Y. and Duman, E. (2011). Detecting credit card fraud by decision trees and support vector machines. *IMECS 2011 - International MultiConference of Engineers and Computer Scientists 2011*, 1:442–447.

Schmidt, M., Le Roux, N., and Bach, F. (2017). Minimizing finite sums with the stochastic average gradient. *Math. Program.*, 162(1–2):83–112.

Tomek, I. (1976). Two modifications of cnn. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-6(11):769–772.

Varmedja, D., Karanovic, M., Sladojevic, S., Arsenovic, M., and Anderla, A. (2019). Credit card fraud detection - machine learning methods. pages 1–5.