JISTEM - Journal of Information Systems and Technology Management
Revista de Gestão da Tecnologia e Sistemas de Informação
Vol. 13, No. 1, Jan/Abr., 2016 pp. 61-80
ISSN online: 1807-1775
DOI: 10.4301/S1807-17752016000100004

SMOTE_EASY: UM ALGORITMO PARA TRATAR O PROBLEMA DE CLASSIFICAÇÃO EM BASES DE DADOS REAIS

SOMOTE_EASY: AN ALGORITHM TO TREAT THE CLASSIFICATION ISSUE IN REAL DATABASES

Hugo Leonardo Pereira Rufino
Instituto Federal do Triângulo Mineiro - Campus Avançado Uberaba Parque Tecnológico, Uberaba, Minas Gerais, Brasil
Antônio Cláudio Paschoarelli Veiga
Universidade Federal de Uberlândia, Uberlândia, Minas Gerais, Brasil
Paula Teixeira Nakamoto
Instituto Federal do Triângulo Mineiro - Campus Avançado Uberaba Parque Tecnológico, Uberaba, Minas Gerais, Brasil

ABSTRACT

Most classification tools assume that data distribution be balanced or with similar costs, when not properly classified. Nevertheless, in practical terms, the existence of database where unbalanced classes occur is commonplace, such as in the diagnosis of diseases, in which the confirmed cases are usually rare when compared with a healthy population. Other examples are the detection of fraudulent calls and the detection of system intruders. In these cases, the improper classification of a minority class (for instance, to diagnose a person with cancer as healthy) may result in more serious consequences that incorrectly classify a majority class. Therefore, it is important to treat the database where unbalanced classes occur. This paper presents the SMOTE_Easy algorithm, which can classify data, even if there is a high level of unbalancing between different classes. In order to prove its efficiency, a comparison with the main

Manuscript first received/*Recebido em:* 07/08/2012 Manuscript accepted/*Aprovado em*: 17/03/2016

Address for correspondence / Endereço para correspondência

Hugo Leonardo Pereira Rufino Instituto Federal do Triângulo Mineiro - Campus Avançado Uberaba Parque Tecnológico, End. Av. Doutor Florestan Fernandes, 131 - Bairro: Univerdecidade - CEP: 38064-190 - Uberaba/MG. Tel.: (34) 33261400. Fax: (34)3326-1011. E-mail: hugo@iftm.edu.br. Graduado em Ciência da Computação pela Universidade Federal de Goiás (2000), Doutor em Ciências pela Universidade Federal de Uberlândia (2011).

Antônio Cláudio Paschoarelli Veiga Universidade Federal de Uberlândia, Departamento de Engenharia Elétrica, End. Av. João Naves de Ávila, 2121, Santa Monica, 38408-100, Uberlândia, MG. Tel.: (34) 32394165, ramal: 4717. Fax: (34) 3239-4704. E-mail: acpveiga@ufu.br Possui doutorado em Engenharia Elétrica pela Universidade Estadual de Campinas (2002). Atualmente é professor adjunto 4 da Universidade Federal de Uberlândia.

Paula Teixeira Nakamoto Instituto Federal do Triângulo Mineiro - Campus Avançado Uberaba Parque Tecnológico, End. Av. Doutor Florestan Fernandes, 131 - Bairro: Univerdecidade - CEP: 38064-190 - Uberaba/MG. Tel.: (34) 33261400. Fax: (34)3326-1011. E-mail: paula@iftm.edu.br. Graduada em Ciência da Computação pela Universidade Federal de Goiás (2001), Doutora em Ciências pela Universidade Federal de Uberlândia (2011).

algorithms to treat classification issues was made, where unbalanced data exist. This process was successful in nearly all tested databases
Keywords: Machine learning, Data Classification, Support-Vector Machines, Machine Committee, Unbalanced Classes.

RESUMO

A maioria das ferramentas de classificação assume que a distribuição dos dados seja balanceada ou com custos iguais, quando classificados incorretamente. Mas, na prática, é muito comum a ocorrência de bases de dados onde existam classes desbalanceadas, como no diagnóstico de doenças, no qual os casos confirmados são geralmente raros quando comparados com a população sadia. Outros exemplos são detecção de chamadas fraudulentas, detecção de intrusos em redes. Nestes casos, a classificação incorreta de uma classe minoritária (ex. diagnosticar uma pessoa portadora de câncer como sadia) pode resultar em consequências mais graves que classificar de forma incorreta uma classe majoritária. Por isso, é importante o tratamento de bases de dados em que ocorram classes desbalanceadas. Este artigo apresenta o algoritmo *SMOTE_Easy*, que é capaz de efetuar a classificação de dados, mesmo com uma alta taxa de desbalanceamento entre as diferentes classes. Para provar sua eficácia, foi feita uma comparação com os principais algoritmos para tratar problemas de classificação onde existam dados desbalanceados. Obteve-se êxito em praticamente todas as bases de dados testadas.

Palavras-Chave: Aprendizado de Máquina, Classificação de Dados, Máquinas de Vetores de Suporte, Comitê de Máquinas, Classes Desbalanceadas.

1. INTRODUÇÃO

O aprendizado de máquina é o campo da Inteligência Artificial responsável pelo desenvolvimento de modelos gerados a partir de dados, e que automaticamente aperfeiçoa-se com a experiência. Existem diversas aplicações para aprendizado de máquina, tais como processamento de linguagem natural, detecção de fraudes, análise de imagens ou reconhecimento de padrões (Mitchell, 1997).

Por volta da metade da década de 1990, uma técnica de aprendizado de máquina que ganhou destaque foram as "Máquinas de Vetores de Suporte", por seu alto desempenho de classificação em vários domínios, incluindo reconhecimento de padrões, mineração de dados e bioinformática. As Máquinas de Vetores de Suporte possuem um forte embasamento teórico e uma ótima capacidade de generalização (Vapnik, 1998). Ainda na década de 1990, comitês de máquinas foram criados e popularizados através de *bagging* e *boosting*. O sucesso dos comitês de máquinas se deve ao fato de serem mais precisos que qualquer um dos classificadores que o compõem (Breiman, 1996), (Freund e Schapire, 1996), (Schapire, 1990). *Bagging* e *boosting* foram criados para trabalhar com classificadores fracos (classificador que classifica dados com uma precisão de, no mínimo 50% (Hulley e Marwala, 2007). Mas, logo, pesquisadores perceberam que também é possível construir comitês de máquinas utilizando um classificador forte (Kim, Pang, Je, Kim e Bang, 2003), (Lima, Neto e Melo, 2009).

Com isso, passaram a incluir nos experimentos as Máquinas de Vetores de Suporte, onde obtiveram sucesso em suas aplicações. Desde então, têm surgido novas metodologias para o aprimoramento das técnicas de classificação. Um classificador que

vem se destacando desde a sua criação é o *AdaBoost* (detalhes em (Freund e Schapire, 1997)). Continua destacando porque outras estratégias de aprimoramento do processo de classificação tais como algoritmos para lidar com bases de dados que possuem classes desbalanceadas ou algoritmos que conseguem lidar com treinamento de dados que são inseridos em lotes (algoritmos de aprendizado incremental) têm como base este classificador. Este artigo dará ênfase à primeira estratégia de aprimoramento. Foi criado um algoritmo que tem como base outros dois já conhecidos, que são os algoritmos *SMOTE* e *EasyEnsemble*. Os detalhes de cada um deles serão apresentados nas próximas seções.

2. OBJETIVOS

O destaque que as Máquinas de Vetores de Suporte e as técnicas utilizadas para classificar bases de dados desbalanceadas vêm recebendo na comunidade científica, motivaram o estudo das Máquinas de Vetores de Suporte e dos algoritmos que lidam com classes desbalanceadas. O objetivo deste trabalho é a criação de um novo algoritmo que tem como base outros dois já conhecidos, que são os algoritmos *SMOTE* e *EasyEnsemble*. Este novo algoritmo, que será chamado de *SMOTE_Easy* possui um desempenho melhor que os dois nos quais ele é baseado, conforme pode ser visto nos resultados que serão mostrados adiante.

3. O PROBLEMA DE BASES DE DADOS DESBALANCEADAS

A maioria dos algoritmos de classificação assume que a distribuição dos dados seja balanceada ou com custos iguais quando classificados incorretamente. Entretanto, quando apresentados a conjuntos de dados desbalanceados, estes algoritmos falham e favorecem apenas a classe dominante (também chamada de classe majoritária ou classe negativa. A classe dominada também pode ser identificada como minoritária ou positiva. No restante deste trabalho será utilizada a denominação negativa e positiva).

As dificuldades em se trabalhar com problemas de classificação na qual existem classes desbalanceadas e sua ocorrência em aplicações práticas de aprendizado de máquina têm atraído consideráveis interesses na área científica (Chawla, Japkowicz, e Kotcz, 2004) (Japkowicz, 2000) (F. Provost, 2000).

Outra considerável demonstração de interesse na área de classificação de dados com classes desbalanceadas pode ser vista na Figura 1 (He e Garcia, 2009), onde é mostrado uma estimativa do número de publicações em problemas de aprendizado de dados desbalanceados no período de 1997 a 2007, baseado no *Institute of Electrical and Electronics Engineers* (IEEE) e *Association for Computing Machinery* (ACM).

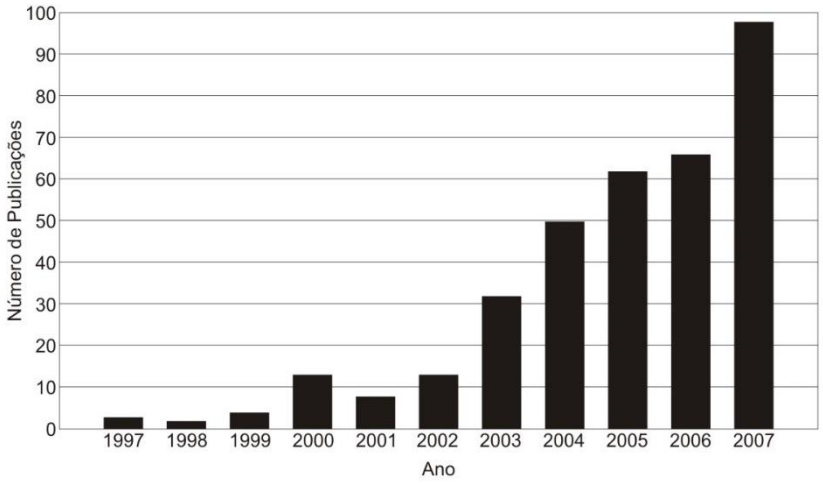

Figura 1 - Número de publicações relacionadas a dados desbalanceados.

Fonte: (He e Garcia, 2009).

O problema que envolve classes desbalanceadas é caracterizado quando existem mais instâncias de uma classe que em outras (pode ser na ordem de 1:10, 1:100 ou mais) (Chawla et al., 2004). Na maioria das aplicações, a classificação correta de um dado pertencente à classe positiva, frequentemente, tem mais valor que no caso contrário. Por exemplo, em problemas de diagnóstico de doenças, onde a quantidade de casos de doenças é bem menor quando comparada com a população total, o objetivo do reconhecimento é detectar pessoas doentes. Então, um bom modelo de classificação é aquele que fornece uma alta taxa de identificação para instâncias que pertencem a esta categoria. Para ilustrar melhor esta situação, considere um conjunto de dados de imagens de mamografias. Analisando as imagens, podem-se obter classes "positivas" e "negativas", onde representam pacientes com câncer e saudáveis, respectivamente. Existem 11.183 imagens, das quais 10.923 pertencem à classe negativa e 260 à classe positiva. Um classificador ideal seria aquele que alcançasse uma taxa de acerto de 100% para ambas as classes. Mas, na prática, existem classificadores que podem gerar uma taxa de acerto próxima de 100% para a classe negativa e de 0 a 10% para a classe positiva (He e Garcia, 2009). Desta taxa de acerto, pode-se concluir que 234 imagens são incorretamente classificadas como pessoas saudáveis (pertencentes à classe negativa). Na área médica, isto gera consequências mais sérias que classificar um paciente saudável como portador de câncer. Outras áreas onde são utilizadas classes desbalanceadas são em detecção de fraude, detecção de intrusos em redes e derramamento de óleo, que podem ser encontrados em (Kubat, Holte, e Matwin, 1998), (Rao, Krishnan, e Niculescu, 2006), (Chan, Fan, Prodromidis, e Stolfo, 1999).

Logo, é importante diferenciar os dois tipos de erros que podem aparecer: um alarme falso (classificar uma classe negativa como positiva) não é tão grave quanto a ausência de um alarme correto (classificar uma classe positiva como negativa). Por exemplo, em condições de monitoramento de máquinas, falsos alarmes ocasionais são menos catastróficos que deixar de emitir um alarme correto de uma falha em uma máquina (Veropoulos, Campbell, e Cristianini, 1999).

Geralmente, o desempenho dos classificadores em bases de dados desbalanceadas é fraco, pois eles são projetados para generalizar a partir dos dados de treinamento e geram o classificador que melhor classifica estes dados, baseados no

princípio da Navalha de Occam. Em dados desbalanceados, a hipótese mais simples, muitas vezes é aquela que classifica todas as instâncias como negativas (Akbani, Kwek, e Japkowicz, 2004).

4. REVISÃO TEÓRICA

Na tarefa de classificação de dados, os comitês de máquinas foram muito importantes, pois aumentaram a precisão da classificação. Neste trabalho, o classificador utilizado para trabalhar nos comitês foram as Máquinas de Vetores de Suporte. Nesta seção serão apresentados estes conceitos principais (Máquinas de Vetores de Suporte e Comitês de Máquinas) para que, logo em seguida, sejam introduzidos os algoritmos-base de *SMOTE_Easy*.

4.1. Máquinas de Vetores de Suporte

Nos últimos anos, o uso de Máquinas de Vetores de Suporte se tornou bastante popular (Wang, 2008). Isto pode ser justificado pela mudança do critério de otimização. Geralmente, algoritmos de classificação procuram definir uma superfície de decisão que minimize o erro de classificação. Nas Máquinas de Vetores de Suporte procura-se definir uma superfície de decisão que maximize a margem, definida como a distância entre a superfície de decisão e os vetores de suporte (instâncias de determinada classe que definem sua linha de fronteira).

A formulação mais simples das Máquinas de Vetores de Suporte é aquela que lida com problemas linearmente separáveis (Boser, Guyon, e Vapnik, 1992). Em problemas não lineares, é feito um mapeando do conjunto de treinamento de seu espaço original para um novo espaço de maior dimensão, chamado de espaço de características (Hearst, 1998).

A tarefa de classificação, usualmente envolve dados de treinamento e teste que consiste de algumas instâncias de dados. Cada instância no conjunto de treinamento contém um valor objetivo (rótulo da classe) e vários atributos (características). O objetivo das Máquinas de Vetores de Suporte é produzir um modelo que prediz o valor objetivo das instâncias no conjunto de teste, que possuem somente os atributos.

Algumas características que motivam o uso das Máquinas de Vetores de Suporte são:

- Possui uma boa capacidade de generalização, ou seja, uma vez que são apresentadas ao conjunto de treinamento, são capazes de aprender a regra que pode classificar corretamente outro conjunto de objetos (Smola et al., 2000);
- O uso das chamadas funções *kernel* torna possível a construção de um hiperplano de separação ótimo no espaço de características (também chamado de espaço de alta dimensão), sem ter que considerar o próprio espaço de características de forma explícita (Haykin, 1999). Isto torna o algoritmo eficiente do ponto de vista computacional (Burges, 1998);
- É eficiente em termos de velocidade e complexidade. É equivalente a procurar por um mínimo de uma função convexa, isto é, sem mínimos locais. Isto elimina muitos dos problemas que ocorrem quando são utilizadas outras

metodologias como redes neurais e árvores de decisão (Bennett e Campbell, 2000);

- Possui uma base teórica bem estabelecida na matemática e estatística. Isso faz com que sejam são robustas quando lidam com objetos de grandes dimensões, como imagens (Smola et al., 2000);
- As MVS foram aplicadas com sucesso em várias áreas (Fan, Zhang, e Xia, 2008), (Li, Fu, e Yang, 2008), (Osuna, Freund, e Girosi, 1997), (Tao, Tang, e Li, 2008) e (Zien et al., 2000). Isto prova que são robustas a ruídos e têm um bom desempenho em várias tarefas.

Hiperplano de Separação para Classes Linearmente Separáveis

O objetivo das Máquinas de Vetores de Suporte é estimar uma função $f : \mathbf{R}^d \to \{\pm 1\}$ utilizando os dados de treinamento compostos de n exemplos de dimensão d, e o rótulo da classe (y) que cada um pertence,

$$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n),\ \mathbf{x} \in \mathbf{R}^d, y \in \{+1, -1\} \tag{1}$$

de tal forma que f classifique corretamente as novas instâncias $(\mathbf{x}, y)$, que são geradas pela mesma distribuição de probabilidade $P(\mathbf{x}, y)$ dos dados utilizados no treinamento.

As Máquinas de Vetores de Suporte são baseadas nos hiperplanos

$$(\mathbf{w} \cdot \mathbf{x}) + b = 0 \text{ com } \mathbf{w} \in \mathbf{R}^d,\ b \in \mathbf{R} \tag{2}$$

que correspondem à função de decisão

$$f(\mathbf{x}) = sign((\mathbf{w} \cdot \mathbf{x}) + b) \tag{3}$$

O hiperplano de separação ótimo é aquele que maximiza a margem (distância entre o hiperplano e o ponto mais próximo de cada classe – Figura 2).

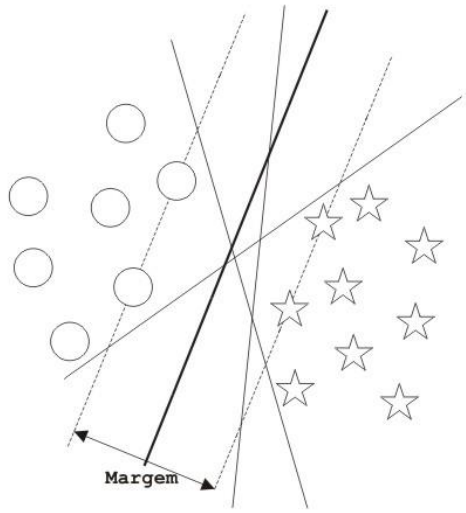

Figura 2 - Hiperplano de Separação Ótimo.

Fonte: Adaptado de (Gunn, 1998).

Este hiperplano pode ser construído através da solução do seguinte problema de otimização quadrática: encontre $\mathbf{w}$ e b de forma que maximize a margem e assegure que todas as instâncias de treinamento são corretamente classificadas,

$$\begin{aligned} &\underset{w,b}{\text{minimizar}} \ \frac{1}{2}\|\mathbf{w}\|^2 \\ &\text{sujeito a} \quad y_i\left[\left(\mathbf{w}\cdot\mathbf{x}_i\right)+b\right]\geq 1, \quad i=1,\ldots,n. \end{aligned} \tag{4}$$

o fator $1/2$ no problema (4) é incluído por conveniência (Bishop, 2006). A restrição é para assegurar que não haja dados de treinamento entre a margem de separação das classes. Utilizando os multiplicadores de Lagrange pode-se encontrar o valor de

$$\mathbf{w} = \sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i, \quad \lambda_i \geq 0, \quad i=1,\ldots,n. \tag{5}$$

Veja na equação (5) que $\lambda_i \geq 0$. Todos os valores de $\mathbf{x}_i$ para os quais $\lambda_i > 0$, são chamados de vetores de suporte (Schölkopf, 1997). Eles carregam toda informação relevante sobre o problema de classificação.

Substituindo a equação (5) na equação (3), obtêm-se

$$f(\mathbf{x}) = sign\left(\sum_{i=1}^{n} \lambda_i y_i (\mathbf{x}_i \cdot \mathbf{x}) + b\right) \tag{6}$$

Para calcular o parâmetro b, usa-se a condição para um padrão de entrada ser um vetor de suporte. Dado qualquer vetor de suporte $(\mathbf{x}_s, y_s)$, este satisfaz a equação (Chekassky e Mulier, 2007)

$$y_s\left[\left(\mathbf{w}\cdot\mathbf{x}_s\right)+b\right]=1 \tag{7}$$

Hiperplano de Separação para Classes Não Linearmente Separáveis

Para tratar estes dados não separáveis linearmente, um novo conjunto de valores escalares não negativos $\{\xi_i\}_{i=1}^{n}$ é introduzido na definição do hiperplano de separação (Cortes e Vapnik, 1995). Estes valores são chamados de variáveis de relaxamento e medem o desvio de um dado da borda da margem de separação. Se um dado estiver localizado na região da margem ou no lado errado do hiperplano de separação, seu correspondente ξ_i será maior que zero. Com estas considerações, o problema de otimização (4) é reformulado como (Cristianini e Shawe-Taylor, 2000).

$$\begin{aligned} &\underset{\xi,w,b}{\text{minimizar}} \ \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{n}\xi_i \\ &\text{sujeito a} \quad y_i\left[\left(\mathbf{w}\cdot\mathbf{x}_i\right)+b\right] \geq 1-\xi_i, \quad i=1,\ldots,n \\ &\text{e} \quad \xi_i \geq 0, \quad i=1,\ldots,n \end{aligned} \tag{8}$$

De acordo com a forma dual de Lagrange, o problema de minimização (8) pode ser escrito como:

$$\begin{aligned} &L(\boldsymbol{\lambda}) = \quad \sum_{i=1}^{n}\lambda_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n}\lambda_i\lambda_j y_i y_j k(\mathbf{x}_i,\mathbf{x}_j) \\ &\text{sujeito a} \quad 0 \leq \lambda_i \leq C, \quad i=1,\ldots,n \\ &\text{e} \quad \sum_{i=1}^{n}\lambda_i y_i = 0 \end{aligned} \tag{9}$$

onde $k(\cdot,\cdot)$ é uma função *kernel*, que mapeia de forma implícita os dados de entrada em um espaço de características, que possui uma dimensão maior e com isso pode efetuar a separação dos dados por um hiperplano.

4.2. Comitês de Máquinas

Um comitê de máquinas (comitê de classificadores) representa a agregação de mais de uma máquina de aprendizado na produção de uma solução computacional para um determinado problema (Haykin, 1999). Além de estarem motivados pela busca da maximização da capacidade de generalização, os comitês de máquinas são motivados por pelo menos outras duas razões (Moraes Lima, 2004):

- Grande disponibilidade de recursos computacionais para possibilitar a síntese de múltiplas propostas de soluções para todo ou parte do problema sendo tratado;
- A demonstração, através do teorema "*no free lunch*" (Wolpert e Macready, 1997), de que não existem modelos genéricos de aprendizado de máquina que, em média, apresentem melhor desempenho que qualquer outro modelo para uma classe de problemas quaisquer.

Uma condição necessária e suficiente para um comitê de máquinas ser mais preciso que qualquer um de seus membros (classificador individual) é que cada um possua uma taxa de erro melhor que uma previsão randômica, e que cada membro tenha erros diferentes para novos dados (Hansen e Salamon, 1990). Na Figura 3 é apresentada a estrutura de um comitê de máquinas.

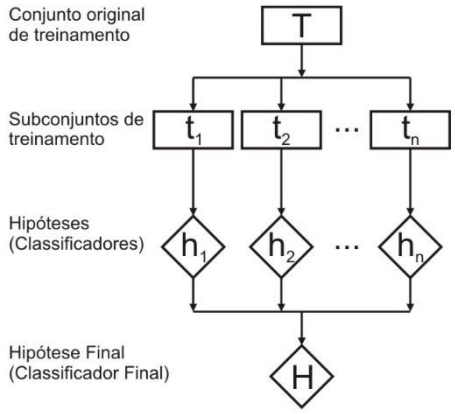

Figura 3 - Estrutura Geral de um Comitê de Máquinas

Comitês de máquinas, tal como *boosting* e *bagging*, constroem múltiplos classificadores através da amostragem do conjunto de treinamento (espaço de entrada): *boosting* trabalha inserindo pesos nas amostras e *bagging* reamostrando subconjuntos das instâncias de treinamento.

A melhoria no desempenho que aparece ao utilizar comitês de máquinas, geralmente é o resultado da redução da variância. A variância mede o quanto as suposições do algoritmo de aprendizado variam para diferentes conjuntos de treinamento. Portanto está associada com superajuste, isto é, modelos (classificadores) mais eficientes são ajustados a diferentes conjuntos de dados obtidos de diferentes distribuições do problema (Kuncheva, 2004). *Boosting* e *Bagging* são capazes de reduzir a variância, logo são imunes ao problema de superajuste.

5. TRABALHOS CORRELATOS

5.1. SMOTE (Synthetic Minority Over-sampling Technique)

Desenvolvido por Chawla et al. (Chawla et al., 2002), o método denominado *SMOTE* efetua superamostragem da classe positiva, criando novas instâncias "sintéticas", ao invés da seleção de amostras. Sendo mais específico, tem-se que $|S| = |S_{pos}| + |S_{neg}|$, onde S representa um dado conjunto de treinamento com m instâncias, ou seja, ($|S| = m$). S_{pos} e S_{neg} representam os conjuntos das classes positivas e negativas, respectivamente. Para o subconjunto S_{pos}, considere os K-vizinhos mais próximos para cada instância $x_i \in S_{pos}$, para um dado valor de K. Os K-vizinhos mais próximos são definidos como os K elementos de S_{pos} cuja distância euclidiana entre si e a instância x_i possui o menor valor. Para criar uma nova instância "sintética", randomicamente selecione um dos K-vizinhos mais próximos, subtraia a instância x_i de seu vizinho mais próximo, multiplique esta diferença por um número randômico entre 0 e 1 e, adicione-a ao valor da instância (x_i).

5.2. Veropoulos

Em classes desbalanceadas, o hiperplano de separação está mais próximo da classe positiva. Isto faz com que os algoritmos de classificação não tenham um bom desempenho neste tipo de problema. Veropoulos (Veropoulos et al., 1999) propôs diferentes penalidades para as diferentes classes, fazendo com que os erros nas instâncias positivas sejam mais penalizados que os das instâncias negativas. Esta alteração altera o comportamento do algoritmo de tal forma que o hiperplano se afaste mais da classe positiva. Especificamente, Veropoulos sugeriu alterar a forma primal de Lagrange do problema (8) para:

$$L(\boldsymbol{\lambda},\boldsymbol{\mu}) = \frac{1}{2}\|\mathbf{w}\|^2 + C^+ \sum_{\{i|y_i=+1\}}^{n_+} + C^- \sum_{\{j|y_j=+1\}}^{n_-} \xi_j - \sum_{i=1}^{n} \lambda_i \left[y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b \right) - 1 + \xi_i \right] - \sum_{i=1}^{n} \mu_i \xi_i \qquad (10)$$

onde $\lambda_i \geq 0$ e $\mu_i \geq 0$. A forma dual é a mesma da equação (9), mas com a restrição de λ_i alterada para:

$$0 \leq \lambda_i \leq C^+ \quad \text{se} \quad y_i = +1 \quad \text{e} \quad 0 \leq \lambda_i \leq C^- \quad \text{se} \quad y_i = -1 \qquad (11)$$

5.3. SDC (SMOTE with Differents Costs)

O algoritmo *SDC* foi proposto por Akbani et at. (2004). A ideia principal de seu funcionamento é unir a técnica *SMOTE*, apresentada na seção 5.1 com custos diferentes para a classe negativa e positiva 5.2. Para isto, utiliza as seguintes estratégias:

- não efetuar subamostragem da classe negativa, pois isto leva a perda de informações;
- utilizar custos diferenciados para erros de classificação de classes diferentes para afastar o hiperplano de separação da classe positiva;
- utilizar *SMOTE* para tornar as instâncias positivas distribuídas de uma forma mais densa, para que o hiperplano seja melhor definido.

Segundo Akbani et at. (2004), em bases de dados desbalanceadas, as instâncias positivas ficam mais afastadas do hiperplano de separação em relação às instâncias negativas. Isso gera um hiperplano de separação mais próximo às classes positivas. Também é mostrado que, apesar do uso de subamostragem na classe negativa melhorar o desempenho do classificador de Máquina de Vetores de Suporte, existe uma inerente perda de informações importantes neste processo. Essa melhoria é justificada pelo aumento da distância dos vetores de suporte e o hiperplano de separação, pois o número de instâncias de treinamento é reduzido. E a perda de informação é justificada através do aumento do ângulo entre o hiperplano ideal e o hiperplano aprendido.

5.4. AdaC1, AdaC2 e AdaC3

Sun, Kamel, Wong, e Wang (2007) utilizaram o algoritmo *AdaBoost* e inseriram um fator de custo em sua função de distribuição, que faz a atualização dos pesos associados a cada instância utilizada no treinamento do algoritmo. Este fator de custo foi inserido de três formas - dentro da exponencial, fora da exponencial e em ambos: fora e dentro da exponencial - gerando três algoritmos denominados AdaC1, AdaC2 e AdaC3. Com estas alterações, a função de distribuição será, respectivamente:

$$D_{t+1}(i) = \frac{D_t(i)\exp(-\beta_t C_i h_t(x_i) y_i)}{Z_t} \tag{12}$$

$$D_{t+1}(i) = \frac{C_i D_t(i)\exp(-\beta_t h_t(x_i) y_i)}{Z_t} \tag{0.13}$$

$$D_{t+1}(i) = \frac{C_i D_t(i)\exp(-\beta_t C_i h_t(x_i) y_i)}{Z_t} \tag{0.14}$$

5.5. EasyEnsemble

Como o uso de subamostragem da classe negativa pode gerar perda de informações, Liu, Wu, e Zhou (2009) desenvolveram o método denominado *EasyEnsemble* (Figura 4), que é um sistema baseado em um comitê de máquinas que cria vários subconjuntos independentes da classe negativa e desenvolve múltiplos classificadores com a combinação destes subconjuntos com a classe positiva.

Entrada:
conjunto dos exemplos da classe minoritária P;
conjunto dos exemplos da classe majoritária N;
o número de subconjuntos T a serem amostrados de N;
o número de iterações s_i para treinar o comitê de máquinas (AdaBoost) H_i;

Saída:
o comitê de máquinas
$H(x) = sgn(\sum_{i=1}^{T}\sum_{j=1}^{s_i}\alpha_{i,j}h_{i,j}(x) - \sum_{i=1}^{T}\Theta_i)$;

i = 0;
repita
  i = i + 1;
  Randomicamente selecione um subconjunto N_i de N, de forma que $|N_i| = |P|$;
  Obtenha H_i utilizando P e N_i. H_i é o comitê de máquinas *Adaboost* com s_i classificadores $h_{i,j}$ e os correspondentes pesos $\alpha_{i,j}$. Θ_i representa o limiar do comitê de máquinas
  $H_i(x) = sgn(\sum_{j=1}^{s_i}\alpha_{i,j}h_{i,j}(x) - \Theta_i)$;
até i = T;

Figura 4 - Algoritmo EasyEnsemble

5.6. Boundary Elimination and Domination (BED)

A ideia principal do algoritmo *BED* (Castro et al., 2009) é melhorar a representatividade da classe positiva através do pré-processamento dos dados no espaço de entrada. Este pré-processamento é feito através da subamostragem das instâncias negativas consideradas ruidosas e da superamostragem da classe positiva. Estes processos são guiados por uma "pontuação de credibilidade" dada a cada amostra do conjunto de treinamento. Esta pontuação é calculada através dos k-vizinhos mais próximos de cada amostra, através da equação

$$cs(x_i^c) = \frac{nC}{nC^+ + nC^-} \tag{15}$$

Onde:

$cs(x_i^c)$ - i-ésima instância de treinamento da classe C;

nC - quantidade de instâncias da classe C presente nos k-vizinhos mais próximos;

nC^+ - quantidade de instâncias que pertencem à classe positiva que estão presentes nos k-vizinhos mais próximos;

nC^- - quantidade de instâncias que pertencem à classe negativa que estão presentes nos k-vizinhos mais próximos.

Se uma instância pertencente à classe negativa tiver uma "pontuação de credibilidade" menor que *max_tol* (valor a ser fornecido, como parâmetro), ele é eliminado do conjunto de treinamento. Se a instância pertencer à classe positiva, e sua pontuação for maior ou igual que *min_tol* (valor a ser fornecido, como parâmetro), um novo elemento é criado, segundo os critérios:

- para cada atributo contínuo m, seu novo valor é calculado baseando na média dos nC^+ vizinhos mais próximos e a amostra x_i^+;
- para os atributos nominais, será assumido o valor que tiver a maior frequência nos nC^+ vizinhos mais próximos e a amostra x_i^+;

6. METODOLOGIA

Para facilitar o entendimento, todos os testes de classificação foram feitos utilizando problemas binários (com duas classes). Pois o problema de classificação pode ser restrito à análise de um problema binário sem perda de generalidade (Gunn, 1998).

As métricas utilizadas serão *G-Mean*, *F-Measure* e *AUC* (apresentadas com mais detalhe, logo adiante). As bases de dados utilizadas no experimento foram extraídas do Repositório UCI (Frank e Asuncion, 2010). São 6 bases de dados com diferentes níveis de desbalanceamento (Tabela 1). Nesta tabela também contém os valores dos parâmetros utilizados no uso dos algoritmos *SMOTE* e *EasyEnsemble*. Para cada base de dado, foram feitas 7 repetições de validação cruzada estratificada com fator 7. Logo em seguida foram calculadas as médias de *G-Mean*, *F-Measure* e AUC. Foi utilizado o

classificador Máquina de Vetores de Suporte, implementado na ferramenta LibSVM (Chang e Lin, 2001), com o *kernel* RBF e Polinomial para todas as bases de dados.

Tabela 1 - Bases de Dados utilizadas no experimento. Para aquelas com mais de duas classes, os valores das classes entre parênteses indicam a classe positiva escolhida. As outras foram unidas e transformadas em classes negativas.

Base de Dado	***Quant. +***	***Quant. -***	***% superamostragem***	***i***
Abalone (19)	32	4145	1000	5
Breast	47	151	200	0
Car (3)	69	1659	1000	0
Diabetes	268	500	100	0
Heart	55	212	300	0
Yeast (5)	51	1433	1000	0

Para avaliar o desempenho, as métricas tradicionais tais como taxa de acerto global ou taxa de erro não são suficientes, pois não fornecem informações adequadas nos casos de aprendizado de classes desbalanceadas. Por exemplo, na seção 3, a classe positiva é representada por aproximadamente 2,3% dos dados de treinamentos. Uma estratégia simples de classificação pode classificar todas as classes do conjunto de treinamento, como sendo pertencente à classe negativa. Com isto, terá uma taxa de acerto de 97,5%. Portanto são necessárias outras formas de métricas para avaliação, tais como *F-measure* (Lewis e Gale, 1994), *G-mean* (Kubat et al., 1998) e área abaixo da curva ROC (do inglês *Area Under the ROC Curve* (AUC)) (Bradley, 1997).

Para formular critérios de desempenho dos sistemas de reconhecimento de padrões, estatísticos trabalham com a matriz de confusão, apresentada na Tabela 2, cujos campos mostram o comportamento de um dado sistema (Kubat e Matwin, 1997).

Tabela 2 - Matriz de confusão: as colunas representam as classes preditas pelo classificador; as linhas representam as verdadeiras classes.

	Classe predita positiva	Classe predita negativa
Classe positiva	VP (Verdadeiro Positivo)	FN (Falso Negativo)
Classe negativa	FP (Falso Positivo)	VN (Verdadeiro Negativo)

- Taxa de Verdadeiro Positivo: $VP_{taxa} = Recall = \frac{VP}{VP + FN}$

- Taxa de Verdadeiro Negativo: $VN_{taxa} = \frac{VN}{VN + FP}$
- Taxa de Falso Positivo: $FP_{taxa} = \frac{FP}{VN + FP}$
- Valor Predito Positivo: $PP_{valor} = \text{Precisão} = \frac{VP}{VP + FP}$
- $F - measure = \frac{(1+\beta^2) \cdot Recall \cdot \text{Precisão}}{\beta^2 \cdot Recall \cdot \text{Precisão}}$

onde β corresponde a importância relativa de precisão vs. *recall* e é usualmente escolhido com o valor 1. O principal objetivo dos algoritmos de aprendizado é aprimorar o *recall* sem sacrificar a precisão (Chawla, Lazarevic, Hall, e Bowyer, 2003). Conforme mostrado, *F-measure* engloba tanto a precisão quanto o *recall*. Isso permite medir a "aceitação" do algoritmo de aprendizado para a classe minoritária.

Quando o desempenho de ambas as classes é importante, tanto a Taxa de Verdadeiro Positivo VP_{taxa} quanto a Taxa de Verdadeiro Negativo VN_{taxa} são esperados ter um alto valor. Kubat (Kubat et al., 1998) sugeriu a métrica: $G{-}mean = \sqrt{VP_{taxa} \cdot VN_{taxa}}$ Esta medida possui a propriedade diferenciada de ser independente da distribuição dos exemplos entre as classes. Portanto, é robusta nas circunstâncias onde a distribuição pode alterar com o tempo (ou ser diferentes nos conjuntos de treinamento e teste). Outra propriedade importante é que uma mudança no valor de VP_{taxa} (ou VN_{taxa}) tem diferentes efeitos em *G-mean*, dependendo do valor de VP_{taxa}: quanto menor o valor de VP_{taxa}, maior a alteração de *G-mean*. Isto significa que classificar incorretamente uma classe positiva faz com que o "custo" seja mais elevado (Kubat et al., 1998).

A área abaixo da curva *ROC* fornece uma medida do desempenho do classificador para avaliar qual é melhor, na média. Para obter a área abaixo da curva *ROC*, é necessário obter a curva *ROC*. Ela é obtida usando os pares (FP_{taxa}, VP_{taxa}). A figura 4.2 ilustra uma curva *ROC*. A parte sombreada representa a área abaixo da curva *ROC*. Os algoritmos para criar a curva e calcular área abaixo da curva podem ser encontrados em Fawcett, 2004.

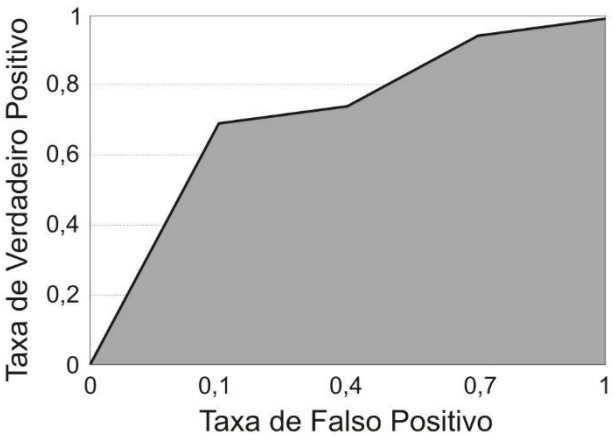

Figura 5 - Representação ROC.

Fonte: (X.-Y. Liu, Wu, e Zhou, 2009).

7. O ALGORITMO *SMOTE_EASY*

Conforme mostrado na seção 5, existem diferentes formas para lidar com o tratamento de bases de dados, onde existe um desbalanceamento entre as classes. O algoritmo *SMOTE_Easy* é a união do algoritmo *SMOTE* com o *EasyEnsemble*. Quanto foi feita esta junção, houve necessidade de efetuar uma mudança no algoritmo *EasyEnsemble*, pois em dois casos (bases de dados "*Heart*" e "*Diabetes*"), após a aplicação do *SMOTE*, a classe positiva ficou com mais instâncias que a classe negativa. Portanto, o trecho do algoritmo que seleciona um subconjunto da classe negativa, baseado no tamanho da classe positiva, foi alterado para que selecionasse um subconjunto da maior classe em relação à menor.

Desta criação, surgiram melhores resultados de classificação, tanto em relação ao *SMOTE* quanto ao *EasyEnsemble*, conforme mostrado na seguinte seção.

8. RESULTADOS

Nas tabelas que seguem é feita uma comparação utilizando as métricas *G-Mean*, *F-Measure* e *AUC* nos algoritmos Máquinas de Vetores de Suporte, *SMOTE*, *EasyEnsemble*, *BED* e *SMOTE_Easy*. O algoritmo *SMOTE_Easy* obteve grande parte dos melhores resultados em todas as bases de dados. O valor da *AUC* do algoritmo *BED* não foi mostrado na

Tabela 5, pois não constava no artigo do autor (Castro et al., 2009).

Tabela 3 - Comparação dos valores do *G-Mean* das bases de dados avaliadas. Os valores presentes na coluna referente ao algoritmo *BED* foram obtidos do artigo do autor (Castro et al., 2009). Máquinas de Vetores de Suporte foi abreviada como MVS.

G-Mean	***MVS***	***SMOTE***	***EasyEnsemble***	***SMOTE_Easy***	**BED**
Abalone (19)	0,0	0,703± 0,011	0,646 ± 0,028	**0,927 ± 0,003**	0,66 ± 0,12
Breast	0,619 ± 0,028	0,858 ± 0,013	0,745 ± 0,032	**0,876 ± 0,013**	0,71 ± 0,04
Car (3)	0,932 ± 0,015	0,933 ± 0,014	0,969 ± 0,002	**0,997 ± 0**	0,95 ± 0,02
Diabetes	0,685 ± 0,009	0,767 ± 0,004	0,745 ± 0,009	**0,771 ± 0,005**	0,75 ± 0,06
Heart	0,640 ± 0,031	0,899 ± 0,007	0,724 ± 0,012	**0,918 ± 0,012**	0,76 ± 0,02
Yeast (5)	0,478 ± 0,028	**0,957 ± 0,002**	0,826 ± 0,018	**0,957 ± 0,004**	0,68 ± 0,04

Tabela 4 - Comparação dos valores do *F-Measure* das bases de dados avaliadas. Os valores presentes na coluna referente ao algoritmo BED foram obtidos do artigo do autor (Castro et al., 2009). Máquinas de Vetores de Suporte foi abreviada como MVS.

F-Measure	***MVS***	***SMOTE***	***EasyEnsemble***	***SMOTE_Easy***	**BED**
Abalone (19)	0,0	0,0	0,043 ± 0,003	**0,581 ± 0,006**	0,33 ± 0,01
Breast	0,465 ± 0,028	0,867 ± 0,012	0,474 ± 0,033	**0,884 ± 0,011**	0,55 ± 0,05
Car (3)	0,895 ± 0,016	0,892 ± 0,022	0,585 ± 0,018	**0,995 ± 0,001**	0,97 ± 0,01
Diabetes	0,608 ± 0,011	**0,783 ± 0,004**	0,672 ± 0,011	0,780 ± 0,005	0,68 ± 0,04
Heart	0,469 ± 0,036	0,914 ± 0,007	0,524 ± 0,014	**0,915 ± 0,008**	0,56 ± 0,05
Yeast (5)	0,276 ± 0,029	**0,930 ± 0,003**	0,247 ± 0,008	0,913 ± 0,006	0,38 ± 0,07

Tabela 5 - Comparação dos valores do AUC das bases de dados avaliadas.

AUC	*MVS*	*SMOTE*	*EasyEnsemble*	*SMOTE_Easy*
Abalone (19)	0,707 ± 0,041	0,969 ± 0,001	0,823 ± 0,020	**0,972 ± 0,001**
Breast	0,745 ± 0,018	**0,975 ± 0,013**	0,728 ± 0,032	0,950 ± 0,008
Car (3)	0,998 ± 0,001	0,998 ± 0,0	0,994 ± 0,002	**1,0 ± 0,0**
Diabetes	0,825 ± 0,005	0,847 ± 0,003	0,820 ± 0,007	**0,849 ± 0,003**
Heart	0,784 ± 0,019	**0,994 ± 0,006**	0,812 ± 0,014	0,977 ± 0,009
Yeast (5)	0,846 ± 0,007	**0,988 ± 0,001**	0,908 ± 0,009	**0,988 ± 0,001**

9. CONCLUSÃO

Este artigo mostrou que a classificação em bases de dados desbalanceadas é muito comum, e que se a ferramenta para executar esta tarefa não for bem elaborada, poderá não conseguir bons resultados ou conseguirá falsos resultados, através da classificação de todos os dados como pertencentes somente a uma classe, conforme visto no exemplo de análise de imagens de mamografia. Em vista disso, foi criado um algoritmo *SMOTE_Easy* que apresenta características das propostas já existentes, mas com melhorias. Analisando os resultados obtidos na seção anterior, percebe-se a superioridade do *SMOTE_Easy*.

REFERÊNCIAS

Akbani, R., Kwek, S., e Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. In *Machine learning: ECML 2004, 15th European Conference on Machine Learning, Pisa, Italy, September 20-24, 2004, Proceedings* (pp. 39–50). Springer.

Bennett, K. P., e Campbell, C. (2000). Support vector machines: Hype or hallelujah? *SIGKDD Explorations*.

Bishop, C. M. (2006). *Pattern recognition and machine learning*. Springer.

Boser, B. E., Guyon, I. M., e Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *COLT: Proceedings of the Workshop on Computational Learning Theory*. Morgan Kaufmann.

Bradley, A. P. (1997). The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition, 30*(7), 1145–1159.

Breiman, L. (1996, Ago.). Bagging predictors. Machine Learning, 24, 123–140.

Burges, C. J. C. (1998). A tutorial on support vector machines for pattern recognition. In U. Fayyad (Ed.), *Data mining and knowledge discovery* (pp. 121–167). Kluwer Academic.

Castro, C. L., Carvalho, M. A., e Braga, A. P. (2009). An improved algorithm for svms classification of imbalanced data sets. In D. Palmer-Brown, C. Draganova, E. Pimenidis, e H. Mouratidis (Eds.), *Engineering applications of neural networks* (Vol. 43, p. 108-118). Springer Berlin Heidelberg.

Chan, P., Fan, W., Prodromidis, A., e Stolfo, S. (1999). Distributed data mining in credit card fraud detection. *IEEE Intelligent Systems, 14*, 67–74.

Chang, C.-C., e Lin, C.-J. (2001). LIBSVM: a library for support vector machines [Computer software manual]. (Recuperado em 15 de agosto, 2010 de http://www.csie.ntu.edu.tw/~cjlin/libsvm)

Chawla, N. V., Bowyer, K. W., Hall, L. O., e Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. J. Artif. Intell. Res. (JAIR), 16, 321–357.

Chawla, N. V., Japkowicz, N., e Kotcz, A. (2004). Editorial: special issue on learning from imbalanced data sets. *SIGKDD Explorations*, 6(1), 1–6.

Chawla, N. V., Lazarevic, A., Hall, L. O., e Bowyer, K. W. (2003). Smoteboost: Improving prediction of the minority class in boosting. In *Principles and practice of knowledge discovery in databases* (pp. 107–119). Springer.

Chekassky, V., e Mulier, F. (2007). *Learning from data - concepts, theory, and methods* (2nd ed.). Wiley.

Cortes, C., e Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297.

Cristianini, N., e Shawe-Taylor, J. (2000). *An introduction to support vector machines and other kernel-based learning methods*. Cambridge, U.K.: Cambridge University Press.

Fan, X., Zhang, G., e Xia, X. (2008). Performance evaluation of SVM in image segmentation. In *IWSCA '08: Proceedings of the 2008 IEEE International Workshop on Semantic Computing and Applications.*

Fawcett, T. (2004). *Roc graphs: Notes and practical considerations for researchers.*

Frank, A., e Asuncion, A. (2010). *UCI machine learning repository*. Recuperado em 26 de setembro, 2010, de http://archive.ics.uci.edu/ml.

Freund, Y., e Schapire, R. E. (1996). Experiments with a new boosting algorithm. In *International Conference on Machine Learning* (pp. 148–156).

Freund, Y., e Schapire, R. E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences, 55*(1), 119–139.

Gunn, S. R. (1998). *Support vector machines for classification and regression* (Tech. Rep.). University of Southampton.

Hansen, L., e Salamon, P. (1990). Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10), 993–1001.

Haykin, S. (1999). *Neural networks: A comprehensive foundation* (2nd ed.). Prentice Hall.

He, H., e Garcia, E. (2009, Set.). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9), 1263–1284.

Hearst, M. A. (1998). Trends & controversies: Support vector machines. *IEEE Intelligent Systems*, 13(4), 18–28.

Hulley, G., e Marwala, T. (2007). Evolving classifiers: Methods for incremental learning. *Computing Research Repository, abs/0709.396*5.

Japkowicz, N. (2000). Learning from imbalanced data sets: A comparison of various strategies. In *Proceedings of Learning from Imbalanced Data Sets, Papers from the AAAI workshop, Technical Report ws-00-05* (pp. 10–15). AAAI Press.

Kubat, M., Holte, R. C., e Matwin, S. (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine Learning, 30*, 195–215.
Kubat, M., e Matwin, S. (1997). Addressing the curse of imbalanced training sets: one-sided selection. In *Proc. 14th International Conference on Machine Learning* (pp. 179–186).
Kuncheva, L. I. (2004). *Combining pattern classifiers - methods and algorithms*. John Wiley & Sons.
Lewis, D. D., e Gale, W. A. (1994). A sequential algorithm for training text classifiers. In *SIGIR '94: Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval* (pp. 3–12). Springer-Verlag New York, Inc.
Li, S., Fu, X., e Yang, B. (2008). Nonsubsampled contourlet transform for texture classifications using support vector machines. In *ICNSC '08: IEEE International Conference on Networking, Sensing and Control*.
Liu, X.-Y.,Wu, J., e Zhou, Z.-H. (2009, Abr.). Exploratory undersampling for class-imbalance learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics), 39*(2), 539–550.
Mitchell, T. M. (1997). *Machine learning*. New York: McGraw-Hill.
Moraes Lima, C. A. de. (2004). *Comitê de máquinas: Uma abordagem unificada empregando máquinas de vetores-suporte*. Tese de doutorado, Universidade Federal de Campinas.
Osuna, E., Freund, R., e Girosi, F. (1997). Training support vector machines: An application to face detection. In IEEE Conference on Computer Vision and Pattern Recognition.
Provost, F. (2000). *Machine learning from imbalanced data sets 101*. (Invited paper for the AAAI'2000 Workshop on Imbalanced Data Sets).
Rao, R. B., Krishnan, S., e Niculescu, R. S. (2006). Data mining for improved cardiac care. *SIGKDD Explorations*, 8(1), 3–10.
Schapire, R. E. (1990). The strength of weak learnability. *Machine Learning, 5*, 197–227.
Schölkopf, B. (1997). *Support vector learning*. Unpublished doctoral dissertation, Technische Universität Berlin.
Smola, A. J., Bartlett, P. L., Schölkopf, B., e Schuurmans, D. (2000). Introduction to large margin classifiers. In A. J. Smola, P. L. Bartlett, B. Schölkopf, & D. Schuurmans (Eds.), *Advances in large margin classifiers* (pp. 1–29). The MIT Press.
Sun, Y. M., Kamel, M. S., Wong, A. K. C., e Wang, Y. (2007, Dez). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition, 40*(12), 3358–3378.
Tao, D., Tang, X., e Li, X. (2008). Which components are important for interactive image searching? *IEEE Trans. Circuits Syst. Video Techn*, 18(1).
Vapnik, V. N. (1998). *Statistical learning theory*. John Wiley & Sons, Inc.
Veropoulos, K., Campbell, C., e Cristianini, N. (1999). Controlling the Sensitivity of Support Vector Machines. In *Proceedings of the International Joint Conference on AI* (pp. 55–60).
Wang, G. (2008, Set.). A survey on training algorithms for support vector machine classifiers. In J. Kim, D. Delen, J. Park, F. Ko, e Y. J. Na (Eds.), *International conference on networked computing and advanced information management* (Vol. 1, pp. 123–128). IEEE Computer Society.
Wolpert, D., e Macready, W. (1997, Abr.). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1(1), 67–82.

Zien, A., Rätsch, G., Mika, S., Schölkopf, B., Lengauer, T., e Muller, K. R. (2000). Engineering support vector machine kernels that recognize translation initiation sites. *BIOINF: Bioinformatics*, 16.

