Universidade Federal de Minas Gerais
Programa de Pós-Graduação em Engenharia Elétrica

Novos critérios para seleção de modelos neurais em problemas de classificação com dados desbalanceados

Cristiano Leite de Castro
Orientador: Prof. Antônio de Pádua Braga

Tese de doutorado submetida à banca examinadora designada pelo colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais, como requisito parcial à obtenção de título de Doutor em Engenharia Elétrica.

Esta tese é dedicada aos meus pais e irmão

Agradecimentos

Ao orientador Prof. Antônio de Pádua Braga pelos valiosos conselhos e discussões, pela paciência e disponibilidade. Seu conhecimento técnico e humano foram essenciais para minha formação científica nesses vários anos de convívio como aluno de mestrado e doutorado.

Aos demais professores e funcionários do PPGEE.

Aos amigos do LITC pelo entusiasmo na discussão de idéias e trocas de experiência.

À toda minha família. Em especial, agradeço a meu pai Darilo (em memória) por ter me ensinado a lutar por meus sonhos, à minha mãe Ivana pelo apoio incondicional, ao meu irmão Flávio, fiel companheiro, à minha avó Ilda e ao meu avô "Curinga" (em memória). Sou também profundamente grato à Denise Golgher pelo amor, apoio e compreensão durante essa longa jornada.

À todos aqueles que de alguma forma participaram desse trabalho.

À FAPEMIG/CAPES pelo apoio financeiro.

Abstract

Artificial Neural Network learners induced from complex and highly imbalanced data sets tend to yield classification models that are biased towards the overrepresented (majority) class. Although some approaches in the literature address this issue, they are limited in the formalization and theoretical characterization of the problem. Here, a formal analysis of the nature of *class imbalance problem* is described based on Bayesian Decision and Statistical Learning theories. As shown the problem arises as a direct consequence of the minimization of a (general) criteria based on the overall error rate and the level of distribution overlapping (noise). Furthermore, two new learning algorithms for MultiLayer Perceptron topology are designed: WEMLP and AUCMLP. Both are formulated from specific criteria for model selection, which are different from the overall error. The cost function for WEMLP algorithm uses a parameter to assign unequal losses (costs) to the errors of each class. The AUCMLP algorithm optimizes a differentiable approximation of the Wilcoxon-Mann-Whitney statistic, analogous metric to the AUC (Area Under the ROC Curve). In order to incorporate an effective strategy of controlling complexity (flexibility) of models, multiobjective (MOBJ) extensions for WEMLP and AUCMLP formulations are provided. Based on statistical analysis of significance of results on real data our approach shows a significant improvement in the classification ranking quality, and achieves high and balanced accuracy rates for both classes.

Resumo

Redes Neurais Artificiais induzidas por conjuntos de treinamento complexos e altamente desbalanceados tendem a produzir modelos de classificação que favorecem a classe com maior probabilidade de ocorrência (majoritária). Embora na literatura existam soluções propostas para esse problema, apenas uma quantidade limitada de trabalhos tem investigado as suas causas e/ou proposto algum tipo de formalismo. Nesse trabalho, uma análise de cunho formal sobre a natureza do *problema de classes desbalanceadas* é descrita com base nas teorias de Decisão Bayesiana e Aprendizado Estatístico. É demonstrado que o problema surge como uma consequência direta da minimização de um critério baseado no Erro global, tendo como principal atenuante o nível de sobreposição (ruído) das distribuições. Adicionalmente, são desenvolvidos dois novos algoritmos de aprendizado para a topologia *MultiLayer Perceptron*: WEMLP e AUCMLP. Ambos são projetados a partir de critérios específicos para seleção de modelos, os quais são diferentes do Erro global. A função custo proposta para o algoritmo WEMLP utiliza um parâmetro para distinguir as perdas associadas a cada classe. O algoritmo AUCMLP otimiza uma aproximação diferenciável da estatística de *Wilcoxon-Mann-Whitney*. Extensões Multiobjetivo (MOBJ) para as formulações de WEMLP e AUCMLP são também propostas, com o propósito de se incorporar uma estratégia efetiva para o controle de complexidade (flexibilidade) de modelos. Testes estatísticos aplicados aos resultados empíricos obtidos com dados reais mostram a eficiência de nossa abordagem em melhorar o *ranking* de classificação e também, em obter taxas de acerto elevadas e equilibradas para ambas as classes.

Sumário

Abreviaturas e Símbolos

Abreviaturas

ACSVM	Asymmetric Cost Support Vector Machines
AUC	Area Under the ROC Curve
FNr	Taxa de falsos negativos
FPr	Taxa de falsos positivos
KKT	Karush-Kuhn-Tucker
KNN	K-Nearest Neighbours
MER	Minimização Estrutural do Risco
MLP	MultiLayer Perceptron
MOBJ	Aprendizado Multiobjetivo
RBF	Radial Basis Function
RNA	Redes Neurais Artificiais
ROC	Receiver Operating Characteristic
SVM	Support Vector Machines
TPr	Taxa de verdadeiros positivos
TNr	Taxa de verdadeiros negativos
RBoost	RAMOBoost
SMOTE	Synthetic Minority Oversampling Technique
SMTTL	SMOTE e Tomek-Links
WWE	Weighted Wilson's Editing

Símbolos Importantes

α	nível de significância do teste estatístico
β	parâmetro de correção na otimização de *Levenberg-Marquadt*
$\bar{\delta}_k$	valor médio dos gradientes locais para a classe k
$\Delta\mathbf{w}$	termo de atualização de pesos
γ	largura do *kernel* gaussiano
η	taxa de aprendizado
$\phi(\cdot)$	função de ativação
$\phi'(\cdot)$	derivada da função de ativação
κ	parâmetro que controla *ranking* no método AUCMLP
λ	parâmetro de custo do método WEMLP
μ	parâmetro que regula o passo no método de *Levenberg-Marquadt*
Ω	complexidade do espaço de hipóteses (funções)
θ	limiar ou *threshold*
θ_{pd}	limiar padrão
ρ	termo de *momentum*
τ	parâmetro que controla a suavidade no método AUCMLP
$AUC(\hat{f})$	estatística de *Wilcoxon-Mann-Whitney*
$\widehat{AUC}(\mathbf{w})$	aproximação diferenciável para *Wilcoxon-Mann-Whitney*
c	número de classes (grupos) em um problema de classificação
$d(p,q)$	diferença entre as saídas obtidas para os exemplos p e q
$\bar{d}_k$	distância média da classe k à superfície de decisão
D	dimensão do vetor de pesos da rede MLP
$E[\cdot]$	operador de esperança matemática
$e(i)$	erro obtido na saída da rede para o i-ésimo exemplo
$\mathbf{e}_k$	vetor de erros associado à classe k
$\bar{e}_k$	valor médio dos erros obtidos para a classe k
$f_0(\mathbf{x})$	regra de decisão que minimiza o funcional risco esperado
$f(\mathbf{x})$	classificador (regra de decisão) arbitrário
$\hat{f}(\mathbf{x})$	classificador estimado a partir de um conjunto de dados
$\bar{f}_k$	valor médio das saídas da rede para a classe k
$\hat{f}(i)$	saída da rede MLP para o i-ésimo padrão de entrada
$\mathbf{g}$	vetor gradiente
$\mathbf{g}_k$	vetor gradiente associado à classe k

$\mathbf{H}$	matriz Hessiana
$\hat{\mathbf{H}}$	aproximação da matriz Hessiana
$\mathbf{H}_k$	matriz Hessiana associada à classe k
$\mathbf{I}$	matriz Identidade
J	funcional custo
J_k	funcional custo associado à classe k
$K(\cdot,\cdot)$	função de *kernel*
$l(y, f(\mathbf{x}))$	função de perda
n	dimensão da camada de entrada da rede MLP
N	número total de exemplos de treinamento
N_k	número de exemplos da classe k
$p(\mathbf{x}, y)$	função densidade de probabilidade conjunta
$p(\mathbf{x}, y = k)$	função densidade de probabilidade conjunta da classe k
$p(y\|\mathbf{x})$	função densidade condicional dos dados de saída
$p(\mathbf{x}\|y = k)$	função densidade condicional da classe k
$P(y = k)$	probabilidade a priori da classe k
$R[\cdot]$	funcional risco esperado
R_{emp}	funcional risco empírico
$\mathcal{R}_k$	região de decisão associada à classe k
T	conjunto de treinamento
T_k	conjunto de treinamento composto por exemplos da classe k
u_s	saída linear do s-ésimo neurônio da camada escondida
v	saída linear da rede MLP
V_k	v.a. representando o valor linear da saída para a classe k
$\mathbf{w}$	vetor de pesos (parâmetros) da rede MLP
$\mathbf{w}^*$	vetor de pesos que representa um mínimo
w_{sr}	peso entre a unidade escondida s e a entrada r
w_s	peso entre a saída e a unidade escondida s
$\mathbf{x}(i)$	i-ésimo vetor ou exemplo de entrada
$\mathcal{X}$	espaço dos padrões de entrada, normalmente $\mathbb{R}^n$
$y(i)$	saída desejada associada ao i-ésimo exemplo de entrada
$\mathcal{Y}$	espaço dos valores de saída (rótulos)
z_s	saída emitida pelo s-ésimo neurônio da camada escondida
$\mathbf{Z}$	matriz Jacobiana
$\mathbf{Z}_k$	matriz Jacobiana associada à classe k
$\|\cdot\|$	operador que fornece a norma euclidiana

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Um aspecto fundamental em problemas de classificação é a possível desigualdade no número de padrões entre os grupos, que surge principalmente em situações onde informações associadas a determinadas classes são mais difíceis de se obter. Pode-se observar esse comportamento, por exemplo, em um estudo sobre uma doença rara em uma dada população. A proporção de pessoas doentes encontradas é muito menor que a proporção de pessoas sadias. Em problemas dessa natureza, em que os números de exemplos entre as classes no conjunto de treinamento variam significativamente, algoritmos de aprendizado tradicionais têm apresentado dificuldade em distinguir entre os vários grupos. Em geral, a tendência é produzir modelos de classificação que favorecem as classes com maior probabilidade de ocorrência, resultando em baixas taxas de reconhecimento para os grupos minoritários.

O problema de classes desbalanceadas, como é conhecido em aprendizado de máquina e mineração de dados, surge principalmente porque os algoritmos tradicionais assumem diferentes erros como igualmente importantes, supondo que as distribuições são relativamente equilibradas (He & Garcia, 2009; Monard & Batista, 2002). Embora essa estratégia possa produzir modelos com elevadas taxas de acurácia global, ela frequentemente tende a prejudicar a identificação de exemplos pertencentes a grupos raros que, na maioria dos casos, representam os grupos de interesse.

A Figura 1.1, a seguir, ilustra o problema a partir de uma rede neural *MultiLayer Perceptron* (MLP) treinada com o algoritmo *Backpropagation* padrão

(Haykin, 1994; Rumelhart & McClelland, 1986). Os dados de treinamento foram gerados a partir de distribuições gaussianas bidimensionais com matrizes de covariância idênticas. As classes minoritária (cruzes) e majoritária (círculos) apresentam, respectivamente, 24 e 240 exemplos (razão 1 : 10).

Pode ser observado, através da Figura 1.1, que a superfície de decisão (linha pontilhada) estimada pela rede MLP encontra-se desviada em direção à classe com menor número de exemplos. Esse desvio é característico do problema de classes desbalanceadas e provoca uma desproporção entre os desempenhos obtidos: 8 erros em relação à classe minoritária e apenas 1 erro em relação à classe majoritária. A mesma desproporção pode ser esperada para dados pertencentes a um conjunto independente de teste, uma vez que assume-se que eles serão extraídos a partir das mesmas distribuições de probabilidade.

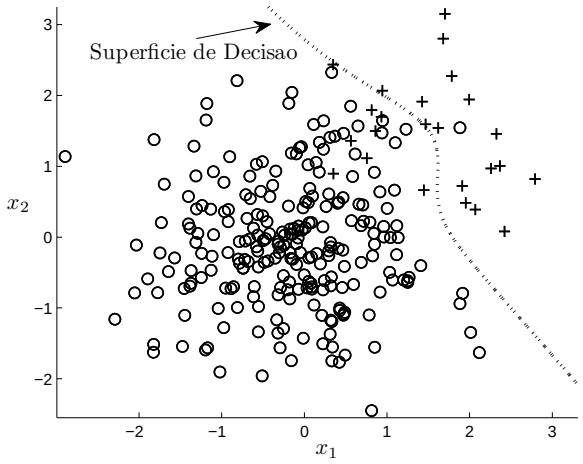


Figura 1.1: Ilustração do problema de classes desbalanceadas a partir do desvio apresentado pela superfície de decisão estimada por uma rede *MultiLayer Perceptron*.

Na maioria das aplicações reais, detectar eventos anormais (ou interessantes) em uma população contendo grande número de eventos comuns é o principal objetivo. Tais aplicações, que normalmente apresentam conjuntos de dados altamente complexos, têm sido reportadas em um grande número de domínios, tais como diagnóstico médico (Braga *et al.*, 2008; Moturu *et al.*, 2010; Silva *et al.*,

2009; Sun *et al.*, 2007), suporte à decisão em unidades de tratamento intensivo (Morik *et al.*, 1999), detecção de fraudes/falhas (Carvalho *et al.*, 2008; Fawcett & Provost, 1997; Gao *et al.*, 2009), categorização de texto (Li & Shawe-Taylor, 2003; Manevitz & Yousef, 2007), reconhecimento de assinaturas (Souza *et al.*, 2010), monitoramento de quebras de eixos automotivos (Hong *et al.*, 2007), identificação de alertas de colisão entre aeronaves (Everson & Fieldsend, 2006b), entre outros.

Dada a relevância do problema, é de fundamental importância o desenvolvimento de algoritmos de aprendizado que considerem o desbalanceamento intrínseco das aplicações. Essa tarefa constitui a base de investigação desse trabalho. Em particular, o estudo aqui realizado busca compreender as dificuldades impostas por classes desbalanceadas no aprendizado de redes *MultiLayer Perceptron* (MLP) e, a partir desse entendimento, sugerir soluções eficientes.

1.1 Objetivos e contribuições

Com o propósito de clarificar as etapas do estudo desenvolvido, são listados a seguir, os objetivos iniciais e as contribuições obtidas. Entre os principais objetivos, destacam-se:

- Estudar o problema de classes desbalanceadas no contexto de redes MLP e fornecer, com base na teoria do Aprendizado Estatístico, melhor entendimento sobre suas causas e consequências. O objetivo desse estudo é obter respaldo teórico para a proposta de uma nova solução para o problema.
- Desenvolver um algoritmo de aprendizado para classificadores binários baseados em redes MLP que possa superar o viés causado por conjuntos de treinamento desbalanceados, melhorando simultaneamente as taxas de acerto para ambas as classes.
- Avaliar o algoritmo desenvolvido em problemas reais desbalanceados comparando os resultados obtidos com métodos similares na literatura.

Entre as contribuições, pode-se citar:

- Análise formal sobre a natureza do problema de classes desbalanceadas a partir das teorias de Decisão Bayesiana (Berger, 1985; Duda *et al.*, 2000) e Aprendizado Estatístico (Vapnik, 1995, 1999). Nessa análise, é demonstrado que o viés imposto pelo desequilíbrio entre as classes surge como uma consequência direta da formulação padrão do aprendizado e também, do nível de incerteza (ruído) associado à tarefa de classificação. Como consequência, o uso da taxa de Erro global como critério (ou função custo) para o treinamento de máquinas de aprendizado é questionado.

  Os resultados provenientes da formalização do problema de classes desbalanceados encontram-se publicados em Castro & Braga (2011a).

- Proposta de dois novos algoritmos de aprendizado para a topologia *Multi-Layer Perceptron*: WEMLP e AUCMLP. Ambos são formulados a partir de critérios específicos e diferentes da taxa de Erro global.

  A função custo proposta para o algoritmo WEMLP possui um parâmetro para penalizar de forma distinta as contribuições dos erros de cada classe. Penalizações com base na proporção dos padrões no conjunto de treinamento permitem obter modelos equilibrados (equidistantes das distribuições), compensando o viés imposto pela classe majoritária. Resultados pré-liminares do algoritmo WEMLP foram publicados como *short-paper* em Castro & Braga (2009). Uma extensão desse trabalho (Castro & Braga, 2011b) foi recentemente submetida para um periódico internacional e, atualmente, encontra-se em fase de revisão.

  A função custo proposta para o algoritmo AUCMLP corresponde a uma aproximação diferenciável da estatística de *Wilcoxon-Mann-Whitney*. Uma restrição na faixa de valores de um dos parâmetros dessa função custo permite obter modelos que otimizam a AUC (*Area Under the ROC Curve*), assim como o equilíbrio entre as taxas de acerto das classes. Resultados teóricos e empíricos do método AUCMLP encontram-se publicados, respectivamente, em Castro & Braga (2008) e Castro & Braga (2010).

- Extensões multiobjetivo (MOBJ) dos algoritmos WEMLP e AUCMLP. As regras de aprendizado dos métodos propostos na tese foram reformuladas

para a incorporação de uma estratégia efetiva de controle de complexidade (flexibilidade) de modelos. Na formulação MOBJ adotada (Teixeira *et al.*, 2000), a norma euclidiana do vetor de pesos da rede MLP (medida de complexidade) é minimizada de forma simultânea aos correspondentes funcionais custo de WEMLP e AUCMLP. O processo de treinamento resulta em um conjunto de soluções eficientes, fornecendo os melhores compromissos entre os funcionais otimizados. O modelo (ou solução) de complexidade apropriada é então selecionado usando um conjunto independente de validação.

Artigos com os resultados das extensões MOBJ dos algoritmos WEMLP e AUCMLP encontram-se em estágio de preparação.

1.2 Visão geral do texto

Para melhor entendimento do leitor, o restante do texto encontra-se estruturado da seguinte forma:

Os dois capítulos iniciais abordam os conceitos teóricos relacionados ao problema de classes desbalanceadas. No Capítulo 2, uma interpretação para a origem do problema é fornecida sob a ótica das teorias do Aprendizado Estatístico e Decisão Bayesiana. No Capítulo 3, são descritas as medidas comumente usadas para avaliar o desempenho de classificadores com distribuições desiguais. Nesse contexto, são também apresentados os fundamentos da análise ROC (*Receiver Operating Characteristic*). A segunda parte do Capítulo 3 traz uma revisão sobre as abordagens propostas para solucionar o problema de classes desbalanceadas, com ênfase naquelas baseadas em modificações de funções custo.

O capítulo 4 fornece os fundamentos teóricos dos algoritmos desenvolvidos na tese. A formulação de WEMLP é apresentada juntamente com uma estratégia, que usa informação a priori, para o ajuste de seu parâmetro de custo. São também discutidos os principais aspectos que motivam o uso da *AUC* (*Area Under the ROC Curve*) como métrica de avaliação de classificadores em domínios desbalanceados. Como consequência, a formulação do algoritmo AUCMLP é descrita. Por último, as extensões multiobjetivo (MOBJ) para o aprendizado de WEMLP

e AUCMLP são apresentadas, com o propósito de se incorporar uma estratégia para controlar a complexidade dos modelos.

No Capítulo 5, a eficiência dos algoritmos propostos é testada em um estudo experimental realizado sobre bases de dados reais. Os desempenhos de WEMLP, AUCMLP e suas respectivas formulações MOBJ são comparados a métodos conhecidos na literatura de classes desbalanceadas. Testes estatísticos de significância são empregados para a análise e discussão dos resultados.

As conclusões gerais do estudo são fornecidas no Capítulo 6, bem como algumas propostas para sua continuidade.

Capítulo 2

O Problema de Classes Desbalanceadas

A maioria dos estudos sobre o problema de classes desbalanceadas foca no desenvolvimento de soluções. Uma quantidade menor tem investigado as suas causas e/ou tentado propor algum tipo de formalismo (Batista *et al.*, 2004; Japkowicz & Stephen, 2002; Khoshgoftaar *et al.*, 2010; Lawrence *et al.*, 1998; Prati *et al.*, 2004; Weiss, 2004; Weiss & Provost, 2003; Wu & Chang, 2003). Nesses trabalhos, a metodologia comumente adotada é a caracterização do problema a partir de observações obtidas com resultados experimentais através de algoritmos de aprendizado específicos.

Nesse capítulo, uma análise de cunho formal sobre a natureza do problema de classes desbalanceadas é fornecida com base nos fundamentos das teorias de Decisão Bayesiana (Berger, 1985; Duda *et al.*, 2000) e Aprendizado Estatístico (Vapnik, 1995, 1999). A metodologia aqui adotada explora as propriedades da solução (ou regra de decisão) ótima que minimiza o valor esperado do erro de classificação. Tal solução pode ser estimada analiticamente em um cenário controlado, onde todas as distribuições de probabilidade são conhecidas. Os aspectos causadores do problema são então discutidos contrastando as características da solução ótima com regras de decisão obtidas por redes *MultiLayer Percetron* (MLP), usando conjuntos de treinamento desbalanceados.

O texto encontra-se organizado da seguinte forma: a Seção 2.1 descreve, sob o ponto de vista estatístico, a formulação geral do problema do aprendizado indu-

tivo e sua especialização na tarefa de classificação contendo somente duas classes (ou grupos). As definições/notações apresentadas nessa seção são essenciais para a análise do problema de classes desbalanceadas a ser detalhada na Seção 2.2. As conclusões oriundas dessa análise são apresentadas na Seção 2.3 e, servem como referência para os desenvolvimentos futuros nos próximos capítulos.

2.1 Formulação do problema do aprendizado

O problema do aprendizado supervisionado (ou indutivo) pode ser formulado como o problema de estimar uma dependência funcional desconhecida (entrada, saída) com base em um conjunto finito de exemplos observados (Vapnik, 1995, 1999). Essa formulação é bem geral e engloba muitas tarefas de aprendizado particulares (ou práticas), tais como regressão, classificação, *clustering* e estimação de densidade (Cherkassky & Mulier, 2007).

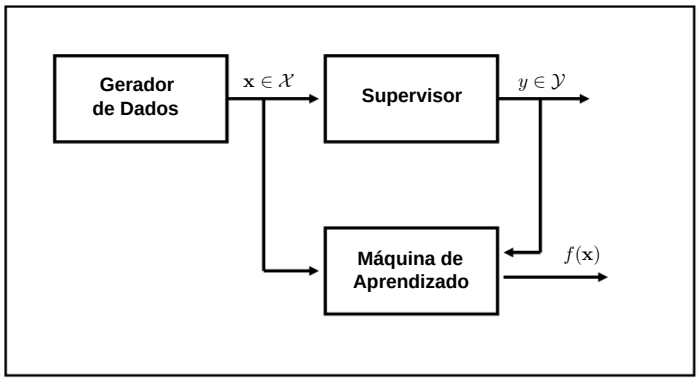


Figura 2.1: Cenário geral de aprendizado envolvendo 3 componentes: um gerador de dados de entrada, um supervisor que retorna uma saída para um dado exemplo de entrada, e uma máquina de aprendizado que estima uma mapeamento desconhecido a partir de dados (entrada, saída) observados.

O cenário geral do aprendizado envolve três componentes, conforme ilustrado pela Figura 2.1. Um **gerador de dados** produz vetores aleatórios $\mathbf{x} \in \mathcal{X}$ obtidos independentemente a partir de uma densidade de probabilidade fixa $p(\mathbf{x})$, que

é desconhecida. Um **supervisor** retorna um valor de saída $y \in \mathcal{Y}$ para todo vetor de entrada $\mathbf{x}$ de acordo com uma densidade condicional $p(y|\mathbf{x})$, também fixa e desconhecida. Para completar o cenário, uma **máquina de aprendizado** é capaz de implementar um conjunto (ou classe) de funções $f : \mathcal{X} \to \mathcal{Y}$, que mapeiam exemplos do espaço de entrada $\mathcal{X}$ para o espaço de saída $\mathcal{Y}$.

Dentre o conjunto de funções fornecido pela máquina de aprendizado, deve-se selecionar aquela que melhor aproxima a resposta do supervisor, baseado em um conjunto finito de exemplos,

$$\{(\mathbf{x}(i), y(i)) \in \mathcal{X} \times \mathcal{Y} \mid i = 1 \dots N\} \tag{2.1}$$

gerados de forma independente e identicamente distribuída (i.i.d.) a partir de uma função densidade de probabilidade conjunta $p(\mathbf{x}, y) = p(y|\mathbf{x})p(\mathbf{x})$.

A qualidade de uma aproximação produzida pela máquina de aprendizado é medida pela perda (ou custo) $l(y, f(\mathbf{x}))$ entre a resposta do supervisor y para um dado vetor de entrada $\mathbf{x}$ e a resposta $f(\mathbf{x})$ fornecida pela máquina. O valor esperado da perda é dado pelo *risco* esperado,

$$R[f] = \int_{\mathcal{X}\times\mathcal{Y}} l(y, f(\mathbf{x}))p(\mathbf{x}, y)dyd\mathbf{x} = E[l(y, f(\mathbf{x}))] \tag{2.2}$$

onde $E[\cdot]$ é o operador de esperança matemática.

Mesmo sem definir uma função de perda particular, o funcional *risco* pode ser visto como um critério de qualidade e o objetivo do aprendizado supervisionado é estimar (escolher) a função que satisfaz esse critério da melhor maneira possível. Formalmente, isso significa, encontrar a função ótima $f_0(\mathbf{x})$ que minimiza (2.2) sobre a classe de funções $f : \mathcal{X} \to \mathcal{Y}$, quando $p(\mathbf{x}, y)$ é desconhecida mas os dados de treinamento (2.1) encontram-se disponíveis.

2.1.1 Classificação binária

Em tarefas de classificação binária (duas classes) é comum considerar que as entradas $\mathbf{x}$ são vetores de características em algum espaço $\mathcal{X} \subseteq \mathbb{R}^n$ e, as saídas assumem somente dois valores simbólicos, ou seja, $y \in \mathcal{Y} = \{0, 1\}$, correspondendo a duas classes. Assim, cada função arbitrária $f(\mathbf{x})$ pertencente à classe de

funções $f : \mathbb{R}^n \rightarrow \{0, 1\}$ torna-se uma regra de decisão, capaz de dividir o espaço de entrada $\mathcal{X}$ em duas regiões disjuntas, denotadas por $\mathcal{R}_0$ e $\mathcal{R}_1$.

A Figura 2.2 ilustra um problema de classificação binária em que as densidades conjuntas das classes $p(x, y = k)$ são representadas por distribuições unidimensionais. Uma regra de decisão $f(x)$ divide o espaço de entrada em duas regiões disjuntas, tal que $\mathcal{X} = \mathcal{R}_0 \cup \mathcal{R}_1$. A decisão sobre a classificação de um dado exemplo de entrada $x(i)$ é feita com base no índice da região $\mathcal{R}_j$ do espaço de entrada em que $x(i)$ está localizado. O limite entre as regiões de decisão é conhecido como superfície de decisão (ou separação) (Cherkassky & Mulier, 2007).

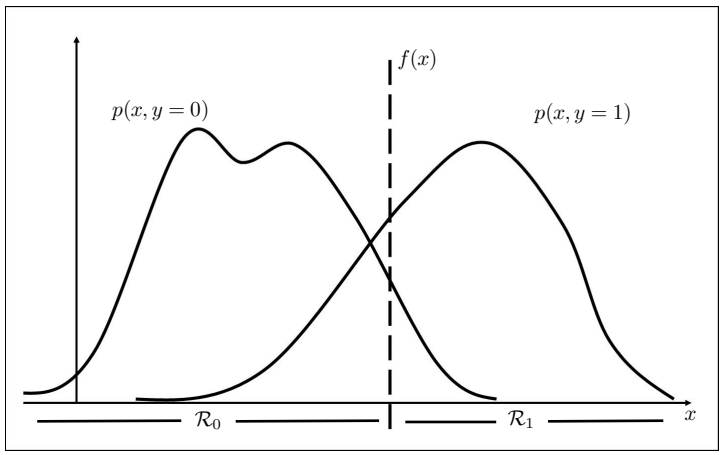


Figura 2.2: Regra de decisão $f(\mathbf{x})$ dividindo o espaço de entrada $\mathcal{X}$ em duas regiões disjuntas $\mathcal{R}_0$ e $\mathcal{R}_1$.

Sem perda de generalidade, é assumido para as seções seguintes desse capítulo que os rótulos (ou índices) 0 e 1 representam, respectivamente, as classes majoritária (ou negativa) e minoritária (ou positiva).

2.2 O problema de classes desbalanceadas

O problema de classes desbalanceadas surge como uma consequência direta da formulação padrão comumente adotada pelas máquinas (algoritmos) de aprendizado

tradicionais. Tal formulação assume custos (perdas) iguais para os diferentes erros de classificação, visando assim à minimização de um critério que corresponde à probabilidade do erro global de classificação (ou taxa de erro esperado) (Bishop, 2006; Vapnik, 1995). Embora a premissa de custos iguais seja mais fiel ao modelo probabilístico adotado, ela tende, em um cenário desbalanceado, a produzir regras de decisão que favorecem a classe com maior probabilidade de ocorrência (majoritária). Essa característica pode não ser adequada para muitos problemas reais em que o objetivo é detectar eventos raros a partir de uma população contendo grande quantidade de eventos comuns.

Essa seção fornece uma caracterização teórica para o problema de classes desbalanceadas. Desde que a sua natureza está intrinsecamente associada à minimização do erro global, a discussão é conduzida através das propriedades da solução ótima $f_0(\mathbf{x})$ para esse critério. Para se definir essa regra de decisão, é preciso, primeiramente, derivar a expressão da probabilidade do erro de classificação a partir da decomposição do funcional *risco* esperado (2.2), em termos das densidades conjuntas $p(\mathbf{x}, y = k)$ de cada classe. Em seguida, $f_0(\mathbf{x})$ pode ser descrita com base nos fundamentos da teoria de Decisão Bayesiana (Berger, 1985; Duda *et al.*, 2000).

Considere então, novamente, a expressão geral do *risco*,

$$\begin{aligned} R[f] &= \int_{\mathcal{X}\times\mathcal{Y}} l(y, f(\mathbf{x}))p(y|\mathbf{x})p(\mathbf{x})dyd\mathbf{x} \\ &= \int_{\mathcal{X}} p(\mathbf{x}) \left[\int_{\mathcal{Y}} l(y, f(\mathbf{x}))p(y|\mathbf{x})dy\right] d\mathbf{x} \end{aligned} \tag{2.3}$$

Seja $p(y|\mathbf{x})$ a densidade condicional dos dados de saída (Bishop, 1995),

$$p(y|\mathbf{x}) = \delta(y)P(y = 0|\mathbf{x}) + \delta(y - 1)P(y = 1|\mathbf{x}) \tag{2.4}$$

onde $P(y = k|\mathbf{x})$ é a probabilidade (condicional) de $y = k$ dado que $\mathbf{x}$ foi observado. Substituindo (2.4) no termo entre colchetes da expressão (2.3), a integral sobre $\mathcal{Y}$ desaparece como consequência da propriedade $\int_t g(t)\delta(t - t_0)\,dt = g(t_0)$ da função delta de *dirac*. A expressão do *risco* pode então ser reescrita como a soma das perdas (custos) esperadas para cada classe,

$$\begin{aligned} R[f] &= \int_{\mathcal{X}} l(y=0, f(\mathbf{x}))P(y=0|\mathbf{x})p(\mathbf{x})d\mathbf{x} \\ &\quad + \int_{\mathcal{X}} l(y=1, f(\mathbf{x}))P(y=1|\mathbf{x})p(\mathbf{x})d\mathbf{x} \end{aligned} \tag{2.5}$$

na qual $\mathcal{X} = \mathcal{R}_0 \cup \mathcal{R}_1$. Finalmente, assumindo custos iguais aos diferentes erros (falsos positivos e falsos negativos), através da *função de perda 0/1* (Vapnik, 1995),

$$l(y=k, f(\mathbf{x})) = \begin{cases} 1 & \text{se } f(\mathbf{x}) \neq k \\ 0 & \text{se } f(\mathbf{x}) = k \end{cases} \tag{2.6}$$

o funcional *risco* esperado (2.5) se reduz à probabilidade do erro global de classificação, dada pela seguinte expressão (Duda *et al.*, 2000),

$$\begin{aligned} R[f] &= P(\mathbf{x} \in \mathcal{R}_1, y=0) + P(\mathbf{x} \in \mathcal{R}_0, y=1) \\ &= \int_{\mathcal{R}_1} p(\mathbf{x}, y=0)\, d\mathbf{x} + \int_{\mathcal{R}_0} p(\mathbf{x}, y=1)\, d\mathbf{x} \end{aligned} \tag{2.7}$$

onde $P(\mathbf{x} \in \mathcal{R}_j, y=k)$ é a probabilidade conjunta de $\mathbf{x}$ ser atribuído à classe j, sendo que sua verdadeira classe é k. Note que para minimizar a probabilidade do erro deve-se atribuir cada vetor de entrada $\mathbf{x}$ à classe (ou região) j para a qual o valor do integrando em (2.7) é mínimo. Isso resulta na seguinte expressão para a regra de decisão ótima (Berger, 1985; Duda *et al.*, 2000),

$$f_0(\mathbf{x}) = \begin{cases} 1 & \text{se } \frac{p(\mathbf{x},y=1)}{p(\mathbf{x},y=0)} \geq 1, \\ 0 & \text{caso contrário.} \end{cases} \tag{2.8}$$

Desde que $p(\mathbf{x}, y=k) = p(\mathbf{x}|y=k)P(y=k)$, a regra (2.8) pode ser reescrita em termos da razão entre as densidades condicionais $p(\mathbf{x}|y=k)$ para cada classe (razão de verossimilhança),

$$f_0(\mathbf{x}) = \begin{cases} 1 & \text{se } \frac{p(\mathbf{x}|y=1)}{p(\mathbf{x}|y=0)} \geq \frac{P(y=0)}{P(y=1)}, \\ 0 & \text{caso contrário.} \end{cases} \tag{2.9}$$

A expressão (2.9) define a solução alvo da formulação padrão adotada pelos algoritmos de aprendizado baseados na minimização do erro global. Uma análise das características apresentadas por essa solução permite entender a natureza do

viés causado pelo desequilíbrio das classes. Observe a partir de (2.9), que a decisão sobre a pertinência de um exemplo arbitrário à classe positiva (minoritária) é diretamente influenciada pela razão entre as probabilidades de ocorrência (a priori) das classes. Assim, para um problema desbalanceado, em que o limiar $\frac{P(y=0)}{P(y=1)}$ é muito maior que 1, a regra de decisão ótima f_0 naturalmente deve favorecer a classe majoritária.

Para fins de ilustração, considere a situação hipotética apresentada na Figura 2.3, onde as densidades condicionais $p(x|y = k)$ são representadas por distribuições gaussianas unidimensionais (conhecidas) possuindo sobreposição e mesma variância; x_0 é a superfície de decisão estimada a partir da regra f_0 que divide o espaço de entrada entre as regiões $\mathcal{R}_0$ e $\mathcal{R}_1$. Note, através dessa figura, que se uma ambiguidade surge na classificação de um exemplo de entrada particular $x(i)$, devido aos valores similares observados para as densidades condicionais, ou seja, $p(x|y = 0) \approx p(x|y = 1)$, f_0 irá atribuir $x(i)$ à classe majoritária, desde que a razão entre as verossimilhanças não excede o limiar imposto por $\frac{P(y=0)}{P(y=1)}$. Analisando a superfície de decisão x_0 no espaço de entrada, um desvio em direção à classe minoritária pode ser verificado.

Como em situações reais não é possível encontrar exatamente f_0, considere $\hat{f}$ como uma estimativa da solução ótima obtida a partir de um conjunto finito de exemplos usando algum método de aprendizado baseado no erro global. Desde que $\hat{f}$ aproxima f_0, é provável em um cenário desbalanceado, que um número maior de erros seja obtido para a classe minoritária. Essa característica é ilustrada no exemplo a seguir, onde $\hat{f}$ é estimada e avaliada, respectivamente, a partir de conjuntos representativos de treinamento e teste gerados (i.i.d.) segundo as densidades conjuntas $p(\mathbf{x}, y = k)$ (conhecidas). A Figura 2.4a, à esquerda, apresenta dados sintéticos (treinamento) obtidos a partir de duas distribuições gaussianas bidimensionais com vetores de média $\mu_0 = (-1, -1)$ e $\mu_1 = (1, 1)$, e matrizes de covariância $\mathbf{\Sigma}_k$ diagonais cujos elementos na diagonal principal são iguais a 1.5. Os círculos pontilhados concêntricos marcam as curvas de nível para as distribuições. A razão entre os números de exemplos da classe majoritária (círculos) e minoritária (cruzes) é 19:1. Duas superfícies de decisão podem ser observadas:

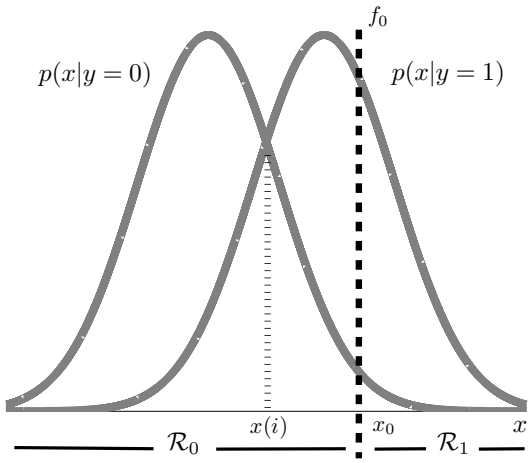
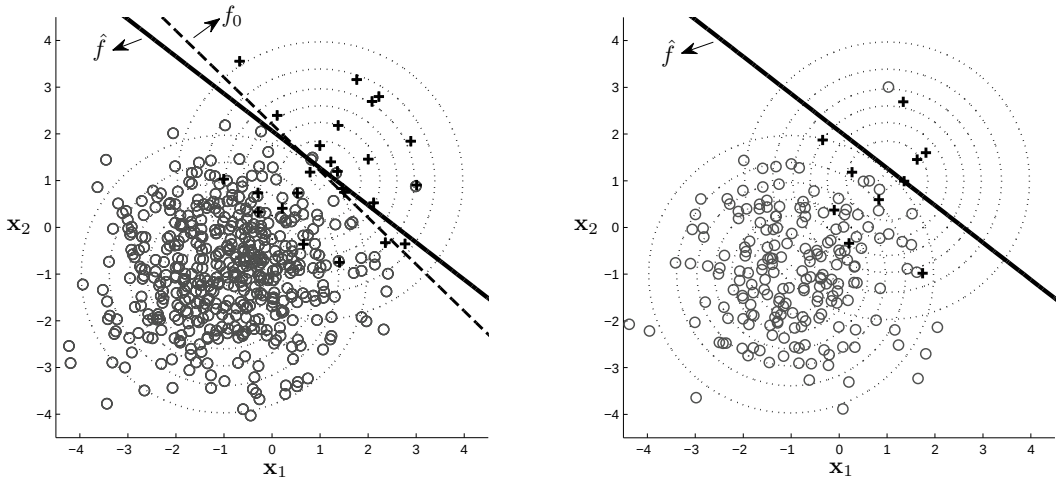


Figura 2.3: Densidades condicionais representadas por gaussianas unidimensionais apresentando sobreposição e mesma variância; x_0 é a superfície de decisão estimada a partir de f_0 para um problema naturalmente desbalanceado. Note que o exemplo arbitrário $x(i)$ é classificado como pertencente ao grupo majoritário, embora os valores observados para as densidades condicionais $p(x|y=0)$ e $p(x|y=1)$ sejam similares.

(i) f_0 (linha tracejada) que corresponde à solução ótima estimada analiticamente[1] a partir das expressões das densidades condicionais $p(\mathbf{x}|y=k)$ e das probabilidades $P(y=k)$; (ii) $\hat{f}$ (linha contínua) que representa a solução estimada por uma rede MLP (topologia 2:1:1) usando o conjunto de treinamento desbalanceado. Note que $\hat{f} \approx f_0$ e, uma vez que os dados são desbalanceados, ambas as superfícies encontram-se desviadas em direção à classe com menor número de exemplos. Na Figura 2.4b, à direita, $\hat{f}$ (linha contínua) foi avaliada em relação ao conjunto de teste. Como esperado, foram observados 6 erros para a classe minoritária (cruzes) e apenas 1 erro para a classe majoritária (círculos).

Além do desequilíbrio entre as distribuições, outro fator determinante para o problema em questão é o nível de incerteza (ruído) associado à tarefa de classificação. Estudos experimentais conduzidos em Japkowicz & Stephen (2002) e Prati *et al.* (2004) mostraram que, para uma razão fixa de desbalanceamento, um

[1]Expressões para funções discriminantes derivadas de distribuições gaussianas multivariadas são fornecidas no Apêndice A.



(a) (b)

Figura 2.4: Ilustração do problema de classes desbalanceadas. Na Figura 2.4a, à esquerda, são apresentadas duas superfícies de separação: (i) f_0 (linha tracejada) derivada analiticamente a partir das distribuições conhecidas $p(\mathbf{x}, y = k)$; (ii) $\hat{f}$ (linha contínua) estimada por uma rede MLP usando o conjunto de treinamento com grau de desbalanceamento 19:1. Note que $\hat{f}$ aproxima f_0 e, ambas as superfícies encontram-se desviadas em direção à classe minoritária (cruzes). Na Figura 2.4b, à direita, $\hat{f}$ (linha contínua) foi avaliada em relação ao conjunto de teste. Como esperado, $\hat{f}$ favorece a classe majoritária (círculos), apresentando um número maior de erros em relação à classe positiva (cruzes).

aumento no nível de sobreposição das classes pode diminuir significativamente o número de classificações positivas corretas. Em trabalho recente, Khoshgoftaar *et al.* (2010) realizaram uma extensa investigação empírica sobre o impacto causado pela combinação "ruído + desbalanceamento" no aprendizado de modelos baseados em redes MLP e RBF (*Radial Basis Function*) (Haykin, 1994). Como resultado da investigação, foi reportado que embora as redes MLP tenham se apresentado mais robustas à presença de "ruído + desbalanceamento" do que as redes RBF, a capacidade de discriminação de ambos os modelos diminui em função do aumento desses fatores.

As conclusões obtidas nos estudos supracitados permitem explicar porque,

para determinadas aplicações, pequenas razões de desbalanceamento podem comprometer mais a capacidade de reconhecimento da classe positiva do que as grandes. Além disso, elas ajudam a compreender a razão da insensibilidade ao desbalanceamento normalmente observada em domínios separáveis, onde as classes representam *clusters* bem definidos no espaço de entrada. Para ilustrar essa idéia, considere o *toy problem* "Duas Luas" na Figura 2.5. Nesse exemplo, devido à separabilidade das distribuições $p(\mathbf{x}|y = k)$, a regra de decisão ótima[1] f_0 (linha tracejada) praticamente não sofre influência do desequilíbrio entre as prioris $P(y = k)$ (razão 5:1). A solução $\hat{f}$ (linha contínua) foi estimada por uma rede MLP treinada com algoritmo MOBJ (Teixeira *et al.*, 2000), usando o conjunto de treinamento formado pelas classes negativa (círculos) e positiva (losangos preenchidos). Note pela Figura 2.5, que apesar do grau de desbalanceamento (razão 5:1), não houve perda na capacidade de reconhecimento da classe de interesse.

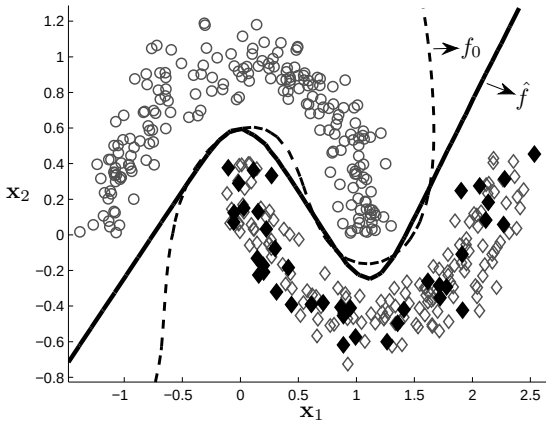


Figura 2.5: "Separabilidade": embora a tarefa de classificação seja desbalanceada (razão 5:1), o grau de separabilidade (não há ruído) das distribuições majoritária (círculos) e minoritária (losangos) assegura que que a solução ótima (linha tracejada) e sua aproximação $\hat{f}$ (linha contínua) apresentem boa capacidade de reconhecimento da classe de interesse.

[1]No exemplo ilustrado pela Figura 2.5, a regra de decisão ótima f_0 foi representada pela superfície de decisão de margem máxima em relação às distribuições alvo.

2.2.1 Análise do problema para a função de perda quadrática

Na seção anterior, o problema de classes desbalanceadas foi analisado no âmbito da formulação tradicional da tarefa de classificação, que usa função de perda 0/1 (veja Equação (2.6)) para penalizar uniformemente os diferentes erros. Nesse caso geral, foi mostrado que o viés imposto pelo grupo dominante é uma consequência direta da minimização de um critério baseado no erro global, tendo como principal atenuante o nível de sobreposição (ruído) das distribuições.

A análise conduzida nessa seção tem como objetivo demonstrar que as conclusões fornecidas para o caso geral são também válidas para a formulação que considera a função de perda quadrática, $l(y, f(\mathbf{x})) = (y - f(\mathbf{x}))^2$, para medir os erros de classificação. Tal formulação, adotada por inúmeras máquinas de aprendizado, incluindo as redes MLP, também assume consequências iguais aos erros de cada classe.

Considere então a definição do *risco* esperado (2.2) através da função de perda quadrática,

$$R[f] = \int_{\mathcal{X}\times\mathcal{Y}} (y - f(\mathbf{x}))^2 \, p(\mathbf{x}, y) dy d\mathbf{x} \tag{2.10}$$

Seja $r(\mathbf{x})$ a função (regressão) que fornece o valor médio de y condicionado a $\mathbf{x}$,

$$r(\mathbf{x}) = E\left[y|\mathbf{x}\right] = \int_{\mathcal{Y}} y \, p(y|\mathbf{x}) dy \tag{2.11}$$

Introduzindo (2.11) no termo entre parênteses de (2.10), pode-se escrever que (Vapnik, 1995),

$$
\begin{aligned}
R[f] &= \int_{\mathcal{X}}\int_{\mathcal{Y}} \left(y - r(\mathbf{x}) + r(\mathbf{x}) - f(\mathbf{x})\right)^2 p(\mathbf{x}, y)dyd\mathbf{x} \\
&= \int_{\mathcal{X}}\int_{\mathcal{Y}} \left(y - r(\mathbf{x})\right)^2 p(\mathbf{x}, y)dyd\mathbf{x} + \int_{\mathcal{X}} \left(f(\mathbf{x}) - r(\mathbf{x})\right)^2 p(\mathbf{x})d\mathbf{x} \\
&\quad +2\int_{\mathcal{X}}\int_{\mathcal{Y}} \left(y - r(\mathbf{x})\right)\left(r(\mathbf{x}) - f(\mathbf{x})\right) p(\mathbf{x}, y)dyd\mathbf{x} \\
&= \int_{\mathcal{X}}\int_{\mathcal{Y}} \left(y - r(\mathbf{x})\right)^2 p(\mathbf{x}, y)dyd\mathbf{x} + \int_{\mathcal{X}} \left(f(\mathbf{x}) - r(\mathbf{x})\right)^2 p(\mathbf{x})d\mathbf{x} \\
&\quad +2\int_{\mathcal{X}} \left(r(\mathbf{x}) - f(\mathbf{x})\right)\left[\int_{\mathcal{Y}} \left(y - r(\mathbf{x})\right) p(y|\mathbf{x})dy\right] p(\mathbf{x})d\mathbf{x} \qquad (2.12)
\end{aligned}
$$

Na expressão (2.12) o terceiro termo se anula, pois,

$$\int_{\mathcal{Y}} \left(y - r(\mathbf{x})\right) p(y|\mathbf{x})dy = \int_{\mathcal{Y}} y\, p(y|\mathbf{x})dy - r(\mathbf{x}) = 0 \qquad (2.13)$$

e assim, o risco esperado pode ser reescrito como (Bishop, 1995; Vapnik, 1995),

$$R[f] = \int_{\mathcal{X}\times\mathcal{Y}} \left(y - r(\mathbf{x})\right)^2 p(\mathbf{x}, y)dyd\mathbf{x} + \int_{\mathcal{X}} \left(f(\mathbf{x}) - r(\mathbf{x})\right)^2 p(\mathbf{x})d\mathbf{x} \qquad (2.14)$$

Note que o primeiro termo em (2.14) é independente de $f(\mathbf{x})$ e representa somente o ruído intrínseco dos dados. Desde que o integrando do segundo termo é não negativo, o mínimo absoluto de $R[f]$ ocorre quando esse termo se anula, o que corresponde à seguinte regra de decisão ótima,

$$f_0(\mathbf{x}) = r(\mathbf{x}) = E\left[y|\mathbf{x}\right] \qquad (2.15)$$

Em outras palavras, a solução que minimiza o *risco* esperado, definido com base na função de perda quadrática, é a própria regressão $E\left[y|\mathbf{x}\right]$.

Para um problema binário de classificação, em que $y \in \{0, 1\}$, considere novamente a expressão da densidade condicional dos dados de saída,

$$p(y|\mathbf{x}) = \delta(y)P(y = 0|\mathbf{x}) + \delta(y - 1)P(y = 1|\mathbf{x}) \qquad (2.16)$$

Substituindo (2.16) em (2.11) pode-se escrever que,

$$\begin{aligned} E\left[y|\mathbf{x}\right] &= \int_{\mathcal{Y}} y\,\left[\delta(y)P(y=0|\mathbf{x}) + \delta(y-1)P(y=1|\mathbf{x})\right] dy \\ &= \int_{\mathcal{Y}} y\,P(y=0|\mathbf{x})\delta(y)dy + \int_{\mathcal{Y}} y\,P(y=1|\mathbf{x})\delta(y-1)dy \\ &= 0\cdot P(y=0|\mathbf{x}) + 1\cdot P(y=1|\mathbf{x}) \\ &= P(y=1|\mathbf{x}) \end{aligned} \tag{2.17}$$

o que nos leva a concluir a partir de (2.15), que a regra de decisão ótima fornece, para cada vetor de entrada $\mathbf{x}$, a sua probabilidade de pertencer à classe positiva (ou minoritária), ou seja,

$$f_0(\mathbf{x}) = E\left[y|\mathbf{x}\right] = P(y=1|\mathbf{x}) \tag{2.18}$$

Colocando $f_0(\mathbf{x})$ na forma de uma função discriminante, com limiar em $1/2$ (padrão), obtém-se a seguinte expressão para a regra de decisão ótima,

$$f_0\left(\mathbf{x}\right) = \begin{cases} 1 & \text{se } P(y=1|\mathbf{x}) \geq \frac{1}{2}, \\ 0 & \text{caso contrário.} \end{cases} \tag{2.19}$$

Desde que a comparação $P(y=1|\mathbf{x}) \geq 1/2$ é equivalente a $P(y=1|\mathbf{x}) \geq P(y=0|\mathbf{x})$ e, fazendo uso da regra do produto da probabilidade $p(\mathbf{x}, y=k) = p(y=k|\mathbf{x})p(\mathbf{x})$, (2.19) pode ser reescrita em termos das densidades conjuntas para cada classe, como segue,

$$f_0\left(\mathbf{x}\right) = \begin{cases} 1 & \text{se } p(\mathbf{x}, y=1) \geq p(\mathbf{x}, y=0), \\ 0 & \text{caso contrário.} \end{cases} \tag{2.20}$$

Observe que (2.20) coincide com a solução ótima (2.8) que minimiza a probabilidade do erro global de classificação (2.7). Esse fato demonstra que, para a tarefa de classificação binária, as formulações baseadas nas perdas quadrática e 0/1 são equivalentes e, portanto, as discussões sobre o problema de classes desbalanceadas apresentadas na Seção 2.2 continuam válidas.

2.3 Conclusões do capítulo

Nesse capítulo uma análise formal sobre a natureza do problema de classes desbalanceadas foi apresentada com base nas teorias de Decisão Bayesiana e Aprendizado Estatístico. Como principal resultado dessa análise, foi demonstrado que o viés induzido pelo desequilíbrio entre as classes é uma consequência direta da formulação padrão do aprendizado e também, do nível de incerteza (ruído) associado à tarefa de classificação.

Note que esse resultado coloca em discussão a habilidade da formulação padrão em lidar com dados altamente desbalanceados e ruidosos, comumente observados em problemas reais; mesmo que embora, essa formulação seja capaz de fornecer "garantias" teóricas para a obtenção da menor taxa de erro global. Em casos extremos de desbalanceamento, tais como aqueles reportados em He & Shen (2007) (razão de 100:1), Kubat *et al.* (1998) (razão de 1000:1) e Pearson *et al.* (2003) (razão de 10000:1), não é incomum que soluções (classificadores) obtidas pela simples minimização do erro global venham a perder toda a sua capacidade de discriminação, classificando todos os exemplos como pertencentes à classe dominante. Essa situação torna-se ainda mais grave quando o interesse do aprendizado está na identificação de exemplos pertencentes ao grupo raro, como ocorre em muitos problemas da área de diagnóstico médico.

Por outro lado, a análise aqui realizada sugere que soluções que não favorecem uma classe em particular ou, de alguma forma priorizam a detecção de exemplos do grupo de interesse, podem ser obtidas modificando-se a formulação comumente adotada pelos algoritmos tradicionais. Do ponto de vista da teoria do Aprendizado Estatístico, isso implica em mudanças no critério escolhido para representar o funcional *risco* esperado, com o objetivo de compensar o efeito causado pelo desbalanceamento. Essas idéias nos levam a considerar a proposta de formulações alternativas ("não-padrão") para o aprendizado de redes MLP que melhor reflitam as necessidades ou requisitos do domínio de aplicação em foco.

Capítulo 3

Aprendizado com dados desbalanceados

Esse capítulo aborda o referencial teórico sobre a pesquisa desenvolvida em aprendizado com dados desbalanceados. Inicialmente, na Seção 3.1, são descritas as medidas comumente usadas para avaliar o desempenho de classificadores sobre classes desiguais. São também fornecidos os principais fundamentos da análise ROC (*Receiver Operating Characteristic*). Em seguida, a Seção 3.2 traz uma revisão sobre as abordagens propostas para solucionar o problema de classes desbalanceadas. Seguindo a discussão teórica sobre a natureza do problema, apresentada no Capítulo 2, uma ênfase nessa revisão é dada ao grupo de métodos que modificam a formulação padrão do aprendizado que é baseada na minimização da taxa de erro global.

Para facilitar o entendimento, os conceitos são descritos com a mesma notação adotada no capítulo anterior: os rótulos (ou saídas) $y = 0$ e $y = 1$ representam, respectivamente, as classes majoritária (ou negativa) e minoritária (ou positiva); $\hat{f}(\mathbf{x})$ corresponde a um classificador (ou regra de decisão) estimado por um conjunto de dados e, $\mathcal{R}_k$ é a região de decisão do espaço de entrada associada à classe k.

3.1 Métricas de avaliação

Uma métrica comumente usada na avaliação e seleção de modelos de classificação é a *acurácia* (ou *taxa de erro*) estimada em relação ao conjunto de validação/teste. Essa metodologia é justificada pela formulação padrão do aprendizado que visa a minimização da probabilidade do erro global. Para problemas desbalanceados, no entanto, a *acurácia* não fornece informação adequada sobre a capacidade de discriminação de um classificador $\hat{f}$ em relação a cada uma das classes em separado. Considere, por exemplo, um conjunto de dados em que a classe minoritária é representada por apenas 2% das observações. Um classificador com *acurácia* de 98% pode ser diretamente obtido, por simplesmente classificar todo exemplo como pertencente à classe majoritária. Apesar da elevada taxa de *acurácia* obtida, tal classificador torna-se inútil se o objetivo principal é a identificação de exemplos raros.

Alguns trabalhos têm chamado a atenção para os problemas causados pelo uso da *acurácia* em cenários desbalanceados (Bradley, 1997; Maloof, 2003; Provost *et al.*, 1998; Sun *et al.*, 2007). Nesse contexto, uma maneira mais eficaz de se avaliar um dado classificador $\hat{f}$ é através da distinção dos erros (ou acertos) cometidos para cada classe. Isso pode ser obtido descrevendo o desempenho de $\hat{f}$ a partir de uma matriz de confusão ou tabela de contingência (vide Tabela 3.1) (Fawcett, 2006). Cada elemento $\epsilon_{k,j}$ dessa matriz fornece o número de exemplos, cuja verdadeira classe era k e que foi atualmente classificado como j. Assim, os elementos ao longo da diagonal principal representam as decisões corretas: número de verdadeiros negativos (TN) e verdadeiros positivos (TP); enquanto os elementos fora dessa diagonal representam os erros cometidos: número de falsos positivos (FP) e falsos negativos (FN).

Tabela 3.1: Matriz de Confusão para um classificador binário.

	predição ($\hat{f} = 0$)	predição ($\hat{f} = 1$)
real ($y = 0$)	TN	FP
real ($y = 1$)	FN	TP

A partir da Tabela 3.1, é possível extrair 4 métricas importantes que diretamente avaliam, de forma independente, o desempenho sobre as classes positiva e negativa,

$$\textit{Taxa de Falsos Positivos: } FPr = \frac{FP}{TN + FP} \tag{3.1}$$

$$\textit{Taxa de Falsos Negativos: } FNr = \frac{FN}{TP + FN} \tag{3.2}$$

$$\textit{Taxa de Verdadeiros Positivos: } TPr = \frac{TP}{TP + FN} \tag{3.3}$$

$$\textit{Taxa de Verdadeiros Negativos: } TNr = \frac{TN}{TN + FP} \tag{3.4}$$

Além das taxas de erro/acerto para cada classe, outras métricas têm sido frequentemente adotadas com o objetivo de fornecer avaliações mais adequadas para aplicações desbalanceadas (He & Garcia, 2009; Sun *et al.*, 2007). Em geral, esses critérios focam na detecção da classe minoritária ou consideram com mesma relevância a discriminação de ambas as classes. Entre as medidas mais usadas, encontram-se:

1. *F-measure*: a métrica *F-measure* considera somente o desempenho para a classe positiva. Ela é calculada a partir de duas métricas adotadas em Recuperação de Informação: *Recall* e *Precision* (Tan *et al.*, 2005). *Recall* (R) é equivalente à taxa de verdadeiros positivos (TPr) e denota a razão entre o número de exemplos positivos corretamente classificados e o número total de exemplos positivos originais,

$$R = TPr = \frac{TP}{TP + FN} \tag{3.5}$$

Precision (P), por sua vez, corresponde à razão entre o número de exemplos positivos corretamente classificados e o número total de exemplos identificados como positivos pelo classificador,

$$P = \frac{TP}{TP + FP} \tag{3.6}$$

Baseado nessas definições, *F-measure* pode ser calculada como,

$$F\text{-}measure = \frac{(1+\beta)\cdot R \cdot P}{\beta^2 \cdot R + P} \tag{3.7}$$

onde β é usado para ajustar a importância relativa entre *Recall* e *Precision*. Tipicamente, $\beta = 1$.

2. *G-mean*: a métrica *G-mean* foi proposta por Kubat *et al.* (1998) e corresponde à média geométrica entre as taxas de verdadeiros positivos (TPr) e verdadeiros negativos (TNr),

$$G\text{-}mean = \sqrt{TPr \cdot TNr} \tag{3.8}$$

G-mean mede o desempenho equilibrado de um classificador em relação às taxas de acertos de ambas as classes (Sun *et al.*, 2007).

3.1.1 Análise ROC

Apesar das métricas apresentadas na Seção 3.1 serem mais eficientes na avaliação de classificadores em cenários desbalanceados, elas não permitem comparar seus desempenhos sobre uma faixa de valores de distribuições a priori. Essa limitação, no entanto, pode ser superada através dos gráficos (curvas) *Receiver Operating Characteristic* (ROC) que foram originalmente desenvolvidos na Teoria de Detecção de Sinais (Egan, 1975; Swets *et al.*, 2000) e, nos últimos anos, têm sido usados pelas comunidades de Aprendizado de Máquina e Mineração de Dados para visualização, avaliação e seleção de modelos (Fawcett, 2006; Prati *et al.*, 2008; Spackman, 1989).

3.1.1.1 Curvas ROC

As curvas ROC possuem propriedades que as tornam especialmente úteis para domínios com classes desbalanceadas. Para compreender seu significado teórico, considere a seguinte regra de decisão expressa através da razão entre as densidades condicionais (razão de verossimilhança),

$$f(\mathbf{x}) = \begin{cases} 1 & \text{se } \frac{p(\mathbf{x}|y=1)}{p(\mathbf{x}|y=0)} > \theta, \\ 0 & \text{caso contrário.} \end{cases} \tag{3.9}$$

Note que essa regra possui forma similar a (2.9) exceto que a razão entre as probabilidades a priori $\frac{P(y=0)}{P(y=1)}$ das classes está implícita no limiar de decisão θ (*threshold*). Assim, variar o limiar θ implica em variar a razão entre as prioris.

Supondo que as distribuições $p(\mathbf{x}|y=k)$ são conhecidas (ou foram estimadas), um valor específico para θ determina as probabilidades de erro/acerto para cada classe: $P(\mathbf{x} \in \mathcal{R}_0|y=1)$ (falsos negativos), $P(\mathbf{x} \in \mathcal{R}_1|y=1)$ (verdadeiros positivos), $P(\mathbf{x} \in \mathcal{R}_1|y=0)$ (falsos positivos) e $P(\mathbf{x} \in \mathcal{R}_0|y=0)$ (verdadeiros negativos); veja Figura 3.1a. Tais probabilidades podem ser calculadas analiticamente por

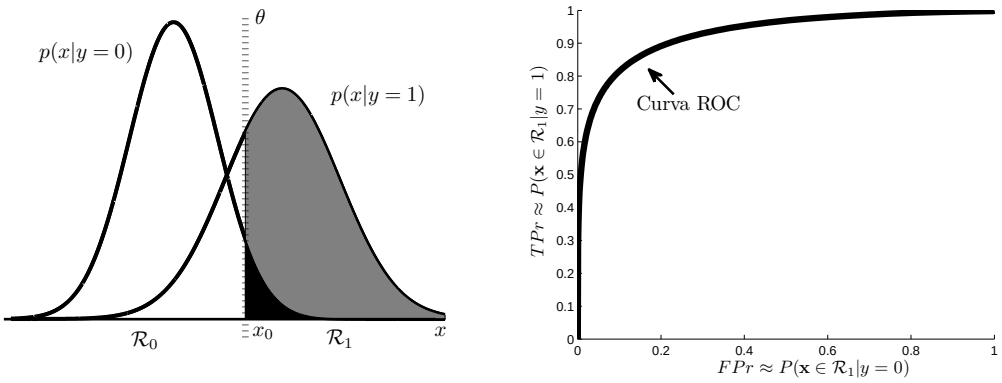
$$P(\mathbf{x} \in \mathcal{R}_j|y=k) = \int_{\mathcal{R}_j} p(\mathbf{x}|y=k)d\mathbf{x}\ , \tag{3.10}$$

com $j, k \in \{0, 1\}$.

A capacidade de discriminação da regra (3.9) sobre toda a faixa de valores do limiar ($0 \leq \theta \leq \infty$) é dada pela curva *Receiver Operating Characteristic* (ROC) (Cherkassky & Mulier, 2007; Duda *et al.*, 2000). Como pode ser visto na Figura 3.1b, uma curva ROC reflete os erros de classificação em termos das probabilidades de detecção $P(\mathbf{x} \in \mathcal{R}_1|y=1)$ (eixo vertical) e falsos alarmes $P(\mathbf{x} \in \mathcal{R}_1|y=0)$ (eixo horizontal) quando θ é variado. Portanto, θ controla a fração de exemplos da classe 1 (positiva) corretamente classificados versus a fração de exemplos da classe 0 (negativa) incorretamente classificados. Esse relacionamento é também conhecido como *trade-off* sensibilidade-especificidade (Lasko *et al.*, 2005).

3.1.1.2 Estimando curvas ROC a partir de conjuntos de dados

Como na prática, as distribuições das classes são desconhecidas, a curva ROC para um classificador $\hat{f}$ é normalmente construída com base nas saídas obtidas por $\hat{f}$ sobre um conjunto de dados particular (Fawcett, 2006). Nesse caso, considere que $\hat{f}$ produz, para cada exemplo $\mathbf{x}(i)$, um *score* que representa o grau de



(a) (b)

Figura 3.1: Significado da Curva ROC. Para fins de ilustração, a Figura 3.1a (esquerda) mostra distribuições $p(x|y=k)$ unidimensionais conhecidas. Note que um valor específico do limiar de decisão (θ) determina as probabilidades de acerto $P(\mathbf{x} \in \mathcal{R}_1|y=1)$ (área em cinza) para a classe positiva e erro $P(\mathbf{x} \in \mathcal{R}_1|y=0)$ (área em preto) para a classe negativa. Na Figura 3.1b, à direita, a curva ROC (para uma regra de decisão) descreve o relacionamento entre as probabilidades de detecção (TPr) e falsos alarmes (FPr) obtidas a partir da variação de θ sobre toda a sua faixa de valores.

pertinência do exemplo à classe positiva. Uma curva ROC pode então ser estimada a partir da variação de um limiar de decisão θ sobre toda a faixa de *scores* (*ranking*) produzida. Cada valor de θ determina valores absolutos para as taxas de detecção (verdadeiros positivos) e falso alarmes (falsos positivos), calculados, respectivamente, por (3.3) e (3.1). A variação de θ sobre toda a faixa de saída de $\hat{f}$ gera uma curva que mostra graficamente o relacionamento (*trade-off*) entre a taxa de verdadeiros positivos (TPr) e a taxa de falsos positivos (FPr). Para um conjunto finito de dados, essas quantidades correspondem, respectivamente, às estimativas para as probabilidades $P(\mathbf{x} \in \mathcal{R}_1|y=1)$ e $P(\mathbf{x} \in \mathcal{R}_1|y=0)$ (veja Figura 3.1b à direita). A curva ROC do classificador "ideal" possui o formato da função de *Heaviside* (*Heaviside step function*) no domínio $0 \leq FPr \leq 1$, indicando que $\hat{f}$ foi capaz de assinalar *scores* mais elevados para os exemplos positivos do que para os exemplos negativos. Isso caracteriza um *ranking* perfeito. Um algoritmo eficiente para computar a curva ROC pode ser encontrado em Fawcett

(2006).

Preferencialmente, um conjunto de teste deve ser usado para a obtenção da curva ROC que fornece uma estimativa da capacidade discriminativa do classificador em termos das probabilidades de erro (Cherkassky & Mulier, 2007). Uma vez estimada, essa curva é útil para a escolha de um ponto de operação θ segundo um critério adotado (Provost & Fawcett, 2001). Por exemplo, pode-se escolher um classificador (ponto de operação) que garanta uma probabilidade muito pequena de erros do tipo falso positivo (*critério de Neyman-Pearson*) (Duda *et al.*, 2000). Cabe ressaltar entretanto, que a acurácia da curva ROC obtida (através do conjunto de teste) é dependente da qualidade da solução estimada $\hat{f}$ usando os dados de treinamento.

Diferentes classificadores podem ser comparados através de suas curvas ROC, contrastando seus desempenhos de detecção TPr para vários valores de θ ou, equivalentemente, FPr. Em alguns casos, as curvas ROC cruzam, indicando que um classificador não fornece melhor desempenho para todos os valores de θ. A *Area Under the ROC Curve* (AUC) (Hanley & Mcneil, 1982) fornece uma medida geral da capacidade de discriminação do classificador que é independente do valor selecionado para θ e, consequentemente, das probabilidades a priori das classes. Detalhes sobre essa importante métrica de avaliação são fornecidos no Capítulo 4.

3.2 Estado da arte das soluções

Essa seção fornece uma breve revisão das abordagens propostas para solucionar o problema de classes desbalanceadas. Seguindo padrão adotado na literatura, essas abordagens foram divididas em duas grandes categorias: *pré-processamento de dados* e *adaptações em algoritmos de aprendizado*. Dentro da segunda categoria, uma maior atenção é dedicada às soluções baseadas em propostas e/ou modificações de funcionais *risco* (ou funções custo) otimizados por algoritmos de aprendizado.

3.2.1 Pré-processamento de dados

Na abordagem de *pré-processamento de dados*, o objetivo é modificar (no sentido de balancear) o conjunto de treinamento através de mecanismos de reamostragem de dados no espaço de entrada, que incluem *sobreamostragem* da classe minoritária, *subamostragem* da classe majoritária ou a combinação de ambas as técnicas.

A *sobreamostragem* é baseada na replicação de exemplos preexistentes (*sobreamostragem com substituição*) ou na geração de dados sintéticos. No primeiro caso, a seleção de exemplos a serem replicados pode ser aleatória (*sobreamostragem aleatória*) ou direcionada (*sobreamostragem informativa*). Com relação à geração de dados sintéticos, a técnica de interpolação é comumente usada. Por exemplo, no conhecido método SMOTE (*Synthetic Minority Oversampling Technique*), proposto em Chawla *et al.* (2002), para cada exemplo positivo $\mathbf{x}(i)$, novos exemplos artificiais são criados entre os segmentos de reta que ligam $\mathbf{x}(i)$ aos seus K vizinhos mais próximos.

A *subamostragem* envolve a eliminação de exemplos da classe majoritária. Os exemplos a serem eliminados podem ser escolhidos aleatoriamente (*subamostragem aleatória*) ou a partir de alguma informação a priori (*subamostragem informativa*). O algoritmo OSS (*One-Sided Selection*), proposto em Kubat & Matwin (1997), é considerado um exemplo de *subamostragem* informativa. Após selecionar um subconjunto representativo da classe majoritária e combiná-lo com todos os exemplos da classe minoritária, o algoritmo OSS usa técnicas de limpeza (*data cleaning*) para obter *clusters* bem definidos para ambas as classes.

Apesar das técnicas de *subamostragem* e *sobreamostragem* possuírem o mesmo propósito, elas introduzem diferentes características ao novo conjunto de treinamento que podem algumas vezes, dificultar o aprendizado (Drummond & Holte, 2003; He & Garcia, 2009; Mease *et al.*, 2007). Por exemplo, no caso de *subamostragem* aleatória, o principal problema é a possível *perda de informação* causada pela eliminação de exemplos representativos da classe majoritária. *Subamostragem* informativa tenta solucionar esse problema por eliminar uma fração menos representativa como, por exemplo, exemplos redundantes, ruidosos e/ou próximos à fronteira de separação entre as classes (*borderlines*). Cabe ressaltar,

entretanto, que a escolha de critérios adequados para selecionar esses exemplos não é uma tarefa fácil. Grande parte dos métodos informativos usa o algoritmo KNN (*K-Nearest Neighbour*) para guiar o processo de subamostragem (Barandela *et al.*, 2004; Batista *et al.*, 2004; Castro *et al.*, 2009; Kubat & Matwin, 1997; Zhang & Mani, 2003). O algoritmo *BalanceCascade*, por sua vez, usa uma estratégia iterativa de geração de um *ensemble* de classificadores para a escolha dos exemplos a serem removidos (Liu *et al.*, 2009).

Com relação a *sobreamostragem*, alguns problemas têm sido reportados. No contexto de árvores de decisão (Breiman *et al.*, 1984), foi observado que o uso de *sobreamostragem com substituição* não melhora de forma significativa o reconhecimento da classe minoritária (Chawla *et al.*, 2002; Mease *et al.*, 2007). Isso ocorre devido à geração de inúmeras cláusulas em um regra para múltiplas cópias do mesmo padrão, tornando a regra muito específica. Outro problema relacionado à *sobreamostragem*, é o aumento da variância (sobreposição) causado por técnicas de geração de dados sintéticos que não consideram a vizinhança entre as classes, como é o caso do método SMOTE (He & Garcia, 2009). Para superar essa limitação, adaptações têm sido propostas para guiar o processo de interpolação adotado (Han *et al.*, 2005; He *et al.*, 2008). Além disso, técnicas de *data cleaning*, tais como *links de Tomek* (Tomek, 1976) e ENN (*Edited Nearest Neighbor rule*) (Wilson, 1972), têm sido sido aplicadas para reduzir o nível de ruído presente nos dados (Batista *et al.*, 2005, 2004).

O efeito das técnicas de reamostragem sobre Redes Neurais Artificiais (RNAs) tem sido investigado. Embora alguns estudos experimentais (Huang *et al.*, 2006; Japkowicz, 2000; Lan *et al.*, 2010; Zhou & Liu, 2006) tenham reportado sucesso em melhorar o desempenho de classificação sobre dados desbalanceados, outros estudos recentes (Alejo *et al.*, 2006; Khoshgoftaar *et al.*, 2010; Mazurowski *et al.*, 2008) têm observado que essa melhora, na maioria dos casos, pode não ser significativa.

Para uma revisão mais detalhada sobre as técnicas de *pré-processamento de dados* existentes, recomenda-se os trabalhos de Weiss (2004) e He & Garcia (2009).

3.2.2 Adaptações em algoritmos de aprendizado

Soluções propostas nessa categoria são baseadas na *adaptação de algoritmos de aprendizado* existentes visando melhorar, ao mesmo tempo, o número de classificações positivas corretas e a acurácia geral do classificador. Em geral, três metodologias principais têm sido usadas no desenvolvimento das soluções, permitindo sua divisão em três grupos. O primeiro grupo considera somente exemplos minoritários durante o processo de aprendizado com o objetivo de reconhecer (ou reconstruir) a classe de minoritária. As principais soluções dessa metodologia baseada em reconhecimento incluem o *autoassociator* (Japkowicz, 2001), cuja arquitetura é baseada em uma rede *MultiLayer Perceptron* (MLP), e *one-class Support Vector Machines* (Raskutti & Kowalczyk, 2004; Schölkopf *et al.*, 2001).

O segundo grupo de soluções é baseado em extensões do algoritmo de *Boosting* (Freund & Schapire, 1997). A maior parte dessas extensões é realizada através da incorporação de diferentes fatores de custo diretamente na função de distribuição, com o objetivo de diferenciar a importância entre as classes e aumentar de forma mais intensa os pesos associados aos exemplos (erros/acertos) da classe minoritária. Baseados nessa metodologia, métodos *cost-sensitive Boosting* foram propostos, tais como AdaCost (Fan *et al.*, 1999), CSB1 e CSB2 (Ting, 2000) e, AdaC1, AdaC2 e AdaC3 (Sun *et al.*, 2007). Outra extensão de *Boosting*, denominada RAMOBoost, foi apresentada recentemente em Chen *et al.* (2010). RAMOBoost combina uma técnica de geração adaptativa de dados sintéticos (RAMO) com um sistema de aprendizado *ensemble* (AdaBoost.M2). Usando redes MLP como *weak-learners*, Chen *et al.* (2010) mostraram a eficácia de RAMOBoost sobre inúmeras bases de dados reais e diferentes métricas de avaliação.

Finalmente, o último grupo de soluções está relacionado a propostas e/ou modificações de funções custo (objetivo). Assumindo a premissa de custos iguais para os erros de classificação, a maioria dos algoritmos de aprendizado existentes é projetada para minimizar o erro global sobre o conjunto de treinamento. Modificações nesse critério, com o objetivo melhorar a detecção da classe minoritária, têm sido propostas de diferentes formas. Uma estratégia comum é introduzir constantes (ou funções) de penalidade para distinguir a importância das classes. Outras estratégias, que envolvem modificações no espaço de características induzido

por *kernels* (Muller *et al.*, 2001), tais como o reposicionamento do hiperplano de separação ou o aumento da resolução espacial dos exemplos positivos, influenciam diretamente o critério de estimação de parâmetros adotado. Além disso, a avaliação individual do desempenho por classe tem permitido uma formulação multiobjetivo para o problema do aprendizado.

Como mencionado anteriormente, as soluções desse último grupo constituem o foco de nossa revisão e assim, uma descrição detalhada dos principais trabalhos propostos no âmbito de máquinas de *kernel* e redes MLP é fornecida nas seções a seguir. Para facilitar o entendimento, são apresentadas aqui as principais notações usadas para descrever os métodos. Seja um conjunto de treinamento $T = \{\mathbf{x}(i), \mathbf{y}(i)\}_{i=1}^{N}$ consistindo de N exemplos pertencentes a duas classes, onde $\mathbf{y}(i) \in \mathcal{Y}$ denota o rótulo para cada vetor de entrada $\mathbf{x}(i) \in \mathbb{R}^n$. A natureza do conjunto $\mathcal{Y}$ é dependente da convenção adotada pelo algoritmo de aprendizado. *Support Vector Machines* (SVMs) e outras máquinas de *kernel*, por exemplo, frequentemente assumem $\mathcal{Y} = \{-1, 1\}$ e, assim, $y(i)$ torna-se uma simples variável simbólica. Quando necessário, a natureza de $\mathcal{Y}$ será especificada durante a descrição do algoritmo. Considere também que existem N_1 exemplos da classe positiva ou minoritária, T_1 e, N_0 exemplos da classe negativa ou majoritária, T_0. Vetores de entrada arbitrários pertencentes às classes positiva e negativa são denotados, respectivamente, por $\mathbf{x}_1$ e $\mathbf{x}_0$.

3.2.2.1 SVMs com Custos Assimétricos

No contexto de SVMs, Lin *et al.* (2002) distinguiram os erros entre as classes positiva e negativa através da introdução de diferentes parâmetros de regularização: C^1 e C^0. Assumindo $y(i) \in \{-1, 1\}$, os autores propuseram a seguinte modificação na função custo do problema primal de SVMs com margens suaves (Cortes & Vapnik, 1995),

$$min_{(\mathbf{w},b,\varepsilon(i))} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C^1 \sum_{i \in T_1} \varepsilon(i) + C^0 \sum_{i \in T_0} \varepsilon(i) \tag{3.11}$$

$$s.a. \quad y(i)\left(\langle \mathbf{w} \cdot \mathbf{x}(i) \rangle + b\right) \geq 1 - \varepsilon(i), \quad \forall\ i \in T$$
$$\varepsilon(i) \geq 0, \quad \forall\ i \in T$$

onde $\mathbf{w}$ e b correspondem aos parâmetros do hiperplano ($\langle \mathbf{w} \cdot \mathbf{x} \rangle + b = 0$) em algum espaço de características $\mathbb{F}$ e as variáveis de folga $\varepsilon(i)$ são introduzidas para permitir erros de classificação. A formulação dual equivalente é dada por,

$$max_{(\alpha)} \quad \sum_{i=1}^{N} \alpha(i) - \frac{1}{2} \sum_{i,j=1}^{N} y(i)y(j)\alpha(i)\alpha(j)K\left(\mathbf{x}(i), \mathbf{x}(j)\right) \tag{3.12}$$

$$s.a. \quad 0 \leq \alpha(i) \leq C^1, \quad \forall \ i \in T_1 \tag{3.13}$$

$$0 \leq \alpha(i) \leq C^0, \quad \forall \ i \in T_0 \tag{3.14}$$

$$\sum_{i=1}^{N} \alpha(i)y(i) = 0 \tag{3.15}$$

onde $K\left(\mathbf{x}, \mathbf{x}'\right)$ representa a função de *kernel*. Resolvendo o problema dual, os multiplicadores de *lagrange* $\alpha(i)$, cujos tamanhos são limitados por C^1 e C^0, são estimados; o parâmetro b pode ser obtido a partir de algum exemplo $\mathbf{x}(i)$ com $\alpha(i)$ não nulo (vetor de suporte). A classificação de um exemplo arbitrário $\mathbf{x}(j)$ é dada pela seguinte regra de decisão (mesma regra da SVM original),

$$\hat{f}\left(\mathbf{x}(j)\right) = \text{sgn}\left(\sum_{i=1}^{N} y(i)\alpha(i)K\left(\mathbf{x}(i), \mathbf{x}(j)\right) + b\right) \tag{3.16}$$

A idéia básica do método é compensar o desbalanceamento do conjunto de dados a partir do ajuste da razão $\frac{C^1}{C^0}$. Segundo Lin *et al.* (2002), se $\frac{C^1}{C^0} > 1$, a estratégia permite aumentar a influência dos vetores de suporte da classe positiva, desde que valores maiores de $\alpha(i)$ são obtidos para os exemplos positivos, conforme condições de *KKT* (*Karush-Kuhn-Tucker*) dadas por (3.13) e (3.14). Isso faz com que a superfície de decisão fique mais distante da classe minoritária e consequentemente, o número de falsos negativos diminua.

Com o objetivo de equilibrar os custos das classes positiva e negativa, Morik *et al.* (1999) e Joachims (2002) propuseram que a razão $\frac{C^1}{C^0}$ seja igual a $\frac{N_0}{N_1}$. Em Lin *et al.* (2002) foi adotada uma estratégia diferente para o ajuste de C^1 e C^0 que além de considerar custos desiguais (para falsos positivos e falsos negativos) também considera *viés de amostragem*. Segundo os autores, *viés de amostragem* ocorre quando os exemplos não são amostrados de uma maneira completamente aleatória, fazendo com que as proporções de positivos e negativos no conjunto de

treinamento não correspondam às atuais proporções na população alvo. Assim, a seguinte técnica para o ajuste de C^1 e C^0 foi proposta,

$$\begin{aligned} C^1 &= l_1\,\hat{\pi}_0\,\pi_1 \\ C^0 &= l_0\,\hat{\pi}_1\,\pi_0 \end{aligned}$$

onde l_1 e l_0 representam os custos associados aos erros das classes positiva e negativa, respectivamente, $\hat{\pi}_1$ e $\hat{\pi}_0$ correspondem, respectivamente, às proporções de exemplos positivos e negativos no conjunto de treinamento e π_1 e π_0 são essas proporções (probabilidades a priori) na população alvo, na qual a SVM deve ser aplicada.

Críticas à eficiência das AC-SVM (*SVMs com Custos Assimétricos*) foram feitas em Wu & Chang (2003). Baseados nas condições de *KKT*, os autores argumentaram que a restrição (3.15) impõe equilíbrio na influência total dos vetores de suporte de cada classe. Para que a restrição seja satisfeita, um aumento nos valores de $\alpha(i)$ para exemplos positivos também deve acarretar um aumento nos valores de $\alpha(i)$ para exemplos negativos. Apesar disso, a estratégia tem apresentado excelentes resultados em aplicações reais desbalanceadas (Akbani *et al.*, 2004; Tang *et al.*, 2009).

3.2.2.2 SVMs com Margens Desiguais

Em Karakoulas & Shawe-Taylor (1999) foi proposta uma estratégia para diferenciar o tamanho das margens (positiva e negativa) no treinamento de SVMs. Isso pode ser obtido a partir da incorporação do parâmetro τ nas restrições de desigualdade referentes aos exemplos da classe positiva. A formulação do problema primal de SVMs com margens suaves é dada por,

$$min_{(\mathbf{w},b,\varepsilon(i))} \quad \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\varepsilon(i) \tag{3.17}$$

$$\begin{aligned} s.a. \quad & y(i)\,(\langle \mathbf{w}\cdot\mathbf{x}(i)\rangle + b) \geq \tau - \varepsilon(i),\ \forall\, i \in T_1 \\ & y(i)\,(\langle \mathbf{w}\cdot\mathbf{x}(i)\rangle + b) \geq -1 + \varepsilon(i),\ \forall\, i \in T_0 \\ & \varepsilon(i) \geq 0,\ \forall\, i \in T \end{aligned}$$

onde $\tau > 1$ corresponde à razão entre a margem positiva e negativa, ou seja, $\tau = \frac{\rho_1}{\rho_0}$. O efeito obtido com o método é o deslocamento paralelo do hiperplano obtido no espaço de características de forma que a margem positiva fique τ vezes maior que a margem negativa.

Os autores também mostraram que o mesmo efeito pode ser obtido a partir da solução do problema original proposto para SVMs (Cortes & Vapnik, 1995), seguido de uma simples mudança no cálculo do parâmetro b (*threshold*) do hiperplano de separação,

$$b = \frac{1}{1+\tau}[\langle \mathbf{w} \cdot \mathbf{x}_1 \rangle + \tau \langle \mathbf{w} \cdot \mathbf{x}_0 \rangle] \tag{3.18}$$

onde $\mathbf{x}_1$ e $\mathbf{x}_0$ correspondem, respectivamente, a vetores de suporte arbitrários da classe positiva e negativa.

Idéia similar foi apresentada em Li & Shawe-Taylor (2003) porém, a incorporação do parâmetro τ ocorre nas restrições de desigualdade referentes aos exemplos da classe negativa. Nesse trabalho, os autores mostraram a eficiência do método em problemas de categorização de textos que, em geral, são altamente desbalanceados.

3.2.2.3 Mudanças no *Kernel*

Ainda no contexto de SVMs, Wu & Chang (2003, 2005) sugerem duas abordagens para modificar o *kernel* empregado considerando a distribuição dos dados como informação a priori. O primeiro algoritmo, *Adaptive Conformal Transformation* (ACT) (Wu & Chang, 2003), modifica a função de *kernel* K no espaço de entrada $\mathbb{I}$ e, portanto, depende que os dados possuam uma representação vetorial de dimensão fixa. O segundo, denominado *Kernel Boundary Alignment* (KBA) (Wu & Chang, 2005), modifica diretamente a matriz de *Kernel* $\mathbf{K}$ no espaço de características $\mathbb{F}$, podendo lidar com dados de diferentes dimensões (sequências de DNA, vídeos de monitoramento, etc.).

A idéia básica em ambos os métodos é aumentar o valor da métrica de *Riemann* para dados próximos à fronteira de separação entre as classes. Segundo os autores, a métrica de *Riemann* associada à função de *kernel* $K(\mathbf{x}, \mathbf{x}')$, mede

como uma área local ao redor de $\mathbf{x}$ em $\mathbb{I}$ é aumentada em $\mathbb{F}$ a partir do mapeamento imposto por $\Phi(\mathbf{x})$. No algoritmo ACT, isso é obtido através de uma transformação *conformal* da função de *kernel* $K(\mathbf{x}, \mathbf{x}')$,

$$\tilde{K}(\mathbf{x}, \mathbf{x}') = D(\mathbf{x})D(\mathbf{x}')K(\mathbf{x}, \mathbf{x}') \tag{3.19}$$

onde $D(\mathbf{x})$ é uma função positiva definida que deve ser escolhida para que a nova métrica de *Riemann* associada à nova função $\tilde{K}(\mathbf{x}, \mathbf{x}')$ possua valores maiores em regiões próximas à fronteira de decisão entre as classes. Além disso, para obter uma superfície de decisão mais distante da classe minoritária, os autores propõem que a métrica de *Riemann* seja aumentada de forma mais intensa em regiões próximas à margem da classe positiva. Para isso, eles sugerem o uso de uma família de funções gaussianas para $D(\mathbf{x})$,

$$D(\mathbf{x}) = \sum_{k=1}^{N_{vs}} \exp\left(-\frac{\|x - x(k)\|}{\sigma^2(k)}\right) \tag{3.20}$$

na qual N_{vs} representa o número total de vetores de suporte (*v.s.*) e o parâmetro de largura $\sigma^2(k)$, deve ser calculado para cada *v.s.* segundo a distribuição espacial de sua vizinhança no espaço de características $\mathbb{F}$. Para detalhes de como esse cálculo é feito veja Wu & Chang (2003). Diferentes fatores são então multiplicados ao parâmetro $\sigma^2(k)$, dependendo se k corresponde a um *v.s.* positivo ou negativo,

$$\begin{cases} \sigma^2(k) \Leftarrow \eta_1\, \sigma^2(k) & \text{se } k \text{ é um } v.s. \text{ positivo,} \\ \sigma^2(k) \Leftarrow \eta_0\, \sigma^2(k) & \text{se } k \text{ é um } v.s. \text{ negativo} \end{cases} \tag{3.21}$$

onde $\eta_1 = \frac{N_{vs}^0}{N_{vs}^1}$ e $\eta_0 = \frac{N_{vs}^1}{N_{vs}^0}$ com, N_{vs}^1 e N_{vs}^0, representando os números de vetores de suporte das classes positiva e negativa, respectivamente. Esse ajuste intensifica a resolução espacial em regiões próximas aos *v.s.* positivos. Após a obtenção da função transformada $\tilde{K}(\mathbf{x}, \mathbf{x}')$, um novo treinamento permite estimar uma regra de decisão com melhor capacidade discriminativa.

No algoritmo KBA, os autores adotam a estratégia de aumentar a resolução espacial junto a um hiperplano de separação considerado "ideal". Eles partem da hipótese de que, quando o conjunto de dados é desbalanceado, o hiperplano de margem máxima obtido pelas SVMs é desviado em direção à classe minoritária. Assim, a superfície de separação "ideal" deve ficar entre esse hiperplano (central)

e o hiperplano representado pela margem da classe majoritária. A localização de um exemplo arbitrário no hiperplano "ideal" é obtida a partir do seguinte procedimento de interpolação, que considera um *v.s.* positivo $\Phi(\mathbf{x}_1)$ e um *v.s.* negativo $\Phi(\mathbf{x}_0)$ no espaço de características $\mathbb{F}$,

$$\Phi\left(\mathbf{x}(b)\right) = (1-\beta)\,\Phi(\mathbf{x}_1) + \beta\,\Phi(\mathbf{x}_0),\;\; \frac{1}{2} \leq \beta \leq 1 \tag{3.22}$$

O parâmetro β fornece indiretamente a localização do hiperplano "ideal" em $\mathbb{F}$. Seu valor ótimo é obtido a partir da minimização de uma função custo que mede a perda causada por falsos positivos e falsos negativos (veja Wu & Chang (2005) para detalhes). O próximo passo é aumentar a métrica de *Riemann* ao redor do hiperplano "ideal". Para tanto, os autores sugerem $D(\mathbf{x})$ como uma família de gaussianas,

$$D(\mathbf{x}) = \frac{1}{N_B}\sum_{b=1}^{N_B} \exp\left(-\frac{\|\Phi(\mathbf{x}) - \Phi\left(\mathbf{x}(b)\right)\|}{\sigma^2(b)}\right) \tag{3.23}$$

onde $\sigma^2(b)$ representa a largura da gaussiana associada a um dado *v.s.* interpolado $\Phi\left(\mathbf{x}(b)\right)$ e, N_B corresponde ao número de *v.s.* interpolados ao longo do hiperplano "ideal". Para um exemplo arbitrário $\mathbf{x}$, $D(\mathbf{x})$ é calculado como a média dessas gaussianas. Desde que o mapeamento $\Phi(\mathbf{x})$ é desconhecido, $\|\Phi(\mathbf{x}) - \Phi\left(\mathbf{x}(b)\right)\|$ pode ser obtido por,

$$\begin{aligned}
\|\Phi(\mathbf{x}) - \Phi\left(\mathbf{x}(b)\right)\| &= \|\Phi(\mathbf{x}) - (1-\beta)\,\Phi(\mathbf{x}_1) - \beta\,\Phi(\mathbf{x}_0)\| \\
&= k_{xx} + (1-\beta)^2\,k_{x^1x^1} + \beta^2\,k_{x^0x^0} \\
&\quad - 2(1-\beta)\,k_{xx^1} - 2\beta\,k_{xx^0} \\
&\quad + 2\beta\,(1-\beta)\,k_{x^1x^0}
\end{aligned}$$

onde $k_{xx'}$ é extraído diretamente da matriz de *kernel* $\mathbf{K}$. Cada elemento de $\mathbf{K}$ é então modificado a partir da transformação conformal descrita a seguir,

$$\tilde{k}_{ij} = D\left(\mathbf{x}(i)\right)\; D\left(\mathbf{x}(j)\right)\; k_{ij} \tag{3.24}$$

A nova matriz $\tilde{\mathbf{K}}$ obtida é novamente usada pelo algoritmo de treinamento original. Em Wu & Chang (2005), os autores testaram a eficiência de seus métodos em bases desbalanceadas do repositório UCI e obtiveram bons resultados.

Outro algoritmo baseado em modificação do *kernel* foi proposto em Kandola & Shawe-Taylor (2003). Os autores sugeriram uma extensão do algoritmo *Kernel Target Alignment* (Cristianini *et al.*, 2002), atribuindo *targets* de alinhamento de $\frac{1}{N^0}$ para exemplos positivos e $-\frac{1}{N^1}$ para exemplos negativos. Foi observado, no entanto, que o algoritmo não foi eficiente para conjuntos de dados com elevado grau de desbalanceamento.

3.2.2.4 *Orthogonal Forward Selection*

Hong *et al.* (2007) apresentaram um novo método para a construção de classificadores binários baseados em *kernels* que, segundo resultados empíricos, tem mostrado bom desempenho em aplicações desbalanceadas. Para tanto, os autores propuseram modificações nos critérios de estimação de parâmetros e seleção de modelos do algoritmo *Orthogonal Forward Selection* (OFS) (Chen *et al.*, 2006).

A cada passo do algoritmo OFS, o método *Regularized Orthogonal Weighted Least Squares* (ROWLS) é usado para estimar os parâmetros dos modelos candidatos através de uma nova função custo que distingue os erros obtidos para cada classe,

$$J = \gamma \sum_{i \in T_0} e^2(i) + \sum_{i \in T_1} e^2(i) \tag{3.25}$$

onde o erro obtido na saída do classificador para um exemplo arbitrário $\mathbf{x}(i)$ é dado por $e(i) = y(i) - \hat{f}(i)$, com $y(i) \in \{-1, 1\}$. O parâmetro de custo $\gamma > 1$, que deve ser escolhido pelo usuário, é usado para atribuir maior peso aos exemplos da classe minoritária; γ tem o efeito de mover o hiperplano para longe da classe minoritária, garantindo que os modelos candidatos sejam apropriados para aplicações desbalanceadas.

Para a seleção do melhor modelo entre os candidatos, os autores propuseram o critério *Leave-One-Out Area Under the ROC Curve* (LOO-AUC). Segundo esse critério, para um dado modelo candidato, os parâmetros são estimados com os

$N-1$ exemplos do conjunto de treinamento, e o exemplo restante é usado como validação, sendo a saída do classificador para esse exemplo denotada por $\hat{f}^{(-i)}(i)$. A AUC é então calculada através das saídas de validação obtidas a partir do *LOO-crossvalidation*, através da seguinte Equação,

$$AUC^{(-)} = \frac{1 + TP^{(-)} - FP^{(-)}}{2} \tag{3.26}$$

onde,

$$\begin{aligned} TP^{(-)} &= \frac{1}{N_1} \sum_{i=1}^{N} IdT\left(\hat{f}^{(-i)}(i) \times y(i),\ y(i)\right) \\ FP^{(-)} &= \frac{1}{N_0} \sum_{i=1}^{N} IdF\left(\hat{f}^{(-i)}(i) \times y(i),\ y(i)\right) \end{aligned}$$

na quais as funções indicadoras $IdT(u,v)$ e $IdF(u,v)$ são definidas por,

$$IdT(u,v) = \begin{cases} 1 & \text{se } u \geq 0 \text{ e } v = 1, \\ 0 & \text{caso contrário.} \end{cases} \tag{3.27}$$

$$IdF(u,v) = \begin{cases} 1 & \text{se } u \leq 0 \text{ e } v = -1, \\ 0 & \text{caso contrário.} \end{cases} \tag{3.28}$$

Segundo os autores, o critério LOO-AUC não é caro computacionalmente, uma vez que o método ROWLS possui fórmulas recursivas que algebricamente implementam *LOO-crossvalidation* sem a necessidade de dividir o conjunto de treinamento.

3.2.2.5 Rede MLP sensível ao custo

Aprendizado sensível ao custo pode ser usado como solução alternativa ao problema de classes desbalanceadas (Elkan, 2001). Essa metodologia tem por base a definição de uma matriz de custo L, cujos elementos fora da diagonal principal, l_{kj} com $k \neq j$, descrevem a penalidade associada ao se classificar um exemplo arbitrário $\mathbf{x}(i)$ à classe j, quando sua verdadeira classe é k.

Para a obtenção de redes MLP sensíveis ao custo, Kukar & Kononenko (1998) propuseram uma modificação na função custo original para que o algoritmo *Back-propagation* (Rumelhart & McClelland, 1986) passe a minimizar o custo total dos

erros de classificação (Elkan, 2001). Os autores consideram uma topologia de rede MLP com codificação 0 de $c-1$ na camada de saída, onde c é o número de unidades (classes); nessa codificação, dado um vetor de entrada $\mathbf{x}(i)$ pertencente à classe T_k, o rótulo $\mathbf{y}(i)$ associado é um vetor cujo j-ésimo componente $y_j(i) = \delta_{jk}$, onde δ_{jk} é o símbolo delta de *kronecker* definido como: $\delta_{jk} = 1$ se $k = j$ e $\delta_{jk} = 0$ se $k \neq j$, para $k, j = 0, \ldots, c-1$. Com base nessa notação, a mudança proposta no funcional somatório dos erros quadráticos é a incorporação do fator $\zeta(k, j)$, com k representando a classe desejada (correta) para o i-ésimo exemplo de treinamento e j a classe atual,

$$J = \frac{1}{2} \sum_{i=i}^{N} \sum_{j=0}^{c-1} \Big(\Big[y_j(i) - \hat{f}_j(i) \Big] \ \zeta(k, j) \Big)^2 \tag{3.29}$$

onde $y_j(i)$ e $\hat{f}_j(i)$ correspondem, respectivamente, às saídas desejada e obtida no j-ésimo neurônio de saída devido à apresentação do exemplo $\mathbf{x}(i)$; A definição do fator $\zeta(k, j)$ é baseada nos custos l_{kj} extraídos da matriz L e, depende de dois aspectos:

- se o neurônio de saída j corresponde à classe correta k do i-ésimo exemplo de treinamento, então a diferença $y_j(i) - \hat{f}_j(i)$ pode ser interpretada como a probabilidade de classificar o exemplo $\mathbf{x}(i)$ em qualquer uma das $c-1$ classes incorretas. Essa probabilidade deve ser ponderada pelo custo esperado do erro para a classe k, descrito pela Equação (3.30) a seguir.

- para os demais neurônios j, que não correspondem à classe correta k do i-ésimo exemplo, a diferença $y_j(i) - \hat{f}_j(i)$ pode ser interpretada como a probabilidade de classificar o exemplo $\mathbf{x}(i)$ na classe j dado $\mathbf{x}(i)$ pertence à k. Nesse caso, ela deve ser ponderada pelo custo l_{kj}.

$$\zeta(k, j) = \begin{cases} \frac{1}{1-\pi_k} \sum_{\forall\, j \neq k} \pi_j \, l_{kj} & \text{se } k = j, \\ \\ l_{kj} & \text{se } k \neq j. \end{cases} \tag{3.30}$$

  na qual π_k é a probabilidade a priori da classe k.

Usando bases de dados do repositório UCI cuja informação da matriz de custo L encontra-se disponível, os autores mostraram que o método proposto possui desempenho próximo a outras abordagens sensíveis ao custo da literatura. A métrica usada para avaliação foi o custo médio esperado (Kukar & Kononenko, 1998).

3.2.2.6 Extensão do Algoritmo *BackPropagation*

Limitando o escopo a problemas contendo somente 2 classes, trabalho recente de Oh (2011) considera redes MLP com duas unidades de saída, tal que o vetor de saída desejada $\mathbf{y}(i)$ associado a um dado exemplo de entrada $\mathbf{x}(i)$ tem seus componentes codificados como: $y_k(i) = +1$ se $\mathbf{x}(i) \in T_k$ e, $y_k(i) = -1$ se $\mathbf{x}(i) \notin T_k$. Usando essa notação, o autor propôs uma extensão do algoritmo *BackPropagation* padrão através da definição de uma nova função custo,

$$J = -\sum_{i=i}^{N} \left[\int \frac{(y_0(i))^{m+1} \left(y_0(i) - \hat{f}_0(i)\right)^m}{2^{m-2}\left(1 - (\hat{f}_0(i))^2\right)} d\hat{f}_0(i) + \int \frac{(y_1(i))^{n+1} \left(y_1(i) - \hat{f}_1(i)\right)^n}{2^{n-2}\left(1 - (\hat{f}_1(i))^2\right)} d\hat{f}_1(i) \right] \tag{3.31}$$

que distingue os sinais de erro obtidos pelas saídas $\hat{f}_k$ referentes a cada classe,

$$\delta_k(i) = \frac{\partial J}{\partial \hat{f}_k(i)} = \begin{cases} (y_0(i))^{m+1} \left(y_0(i) - \hat{f}_0(i)\right)^m / 2^{m-1} & \text{p/ } k = 0, \\ (y_1(i))^{n+1} \left(y_1(i) - \hat{f}_1(i)\right)^n / 2^{n-1} & \text{p/ } k = 1. \end{cases} \tag{3.32}$$

De acordo com Oh (2011), $n < m$ implica em $|\delta_1(i)| > |\delta_0(i)|$ para $-1 < \hat{f}_k(i) < +1$. Logo, um sinal mais forte de erro será gerado para o neurônio alvo da classe positiva. Como, no algoritmo *BackPropagation*, os pesos são atualizados de forma proporcional a $\delta_k(i)$, essa estratégia permite intensificar a atualização de pesos com relação a $\hat{f}_1$ e enfraquecer com relação a $\hat{f}_0$. Por último, desde que em uma época de treinamento, o neurônio $\hat{f}_1$ é selecionado como alvo $(+1)$ N_1 vezes, e $\hat{f}_2$ é selecionado como alvo N_0 vezes, a seguinte modificação foi proposta para os sinais de erro obtidos,

$$\delta_k(i) = \begin{cases} \omega\delta_k(i) & \text{se } (k = 0 \text{ e } y_k(i) = 1) \text{ ou } (k = 1 \text{ e } y_k(i) = -1)\ , \\ \\ \delta_k(i) & \text{caso contrário.} \end{cases} \tag{3.33}$$

com $\omega = N_1/N_0$. Com relação aos valores para os demais parâmetros, foi sugerido $n = 2$ e $3 < m < 10$. Em experimentos conduzidos com dois problemas desbalanceados da área de diagnóstico médico, o método proposto obteve melhores resultados em termos das métricas *G-mean*, TPr e $|TPr - TNr|$.

3.2.2.7 Abordagem Multiobjetivo

Com o objetivo de otimizar a curva ROC para classificadores binários baseados em redes MLP, alguns trabalhos na literatura (Everson & Fieldsend, 2006a; Graening *et al.*, 2006; Kupinski & Anastasio, 1999; Sanchez *et al.*, 2005), formularam o problema do aprendizado como um problema de otimização multiobjetivo, da seguinte forma,

$$\arg_{\mathbf{w}} \ \max\,(\min) \begin{cases} J_0(\mathbf{w}) \\ J_1(\mathbf{w}) \end{cases} \tag{3.34}$$

onde $\mathbf{w}$ é conjunto de parâmetros (pesos) e as funções custo $J_0(\mathbf{w})$ e $J_1(\mathbf{w})$ correspondem a métricas extraídas da matriz confusão que medem o desempenho obtido pela rede para as classes T_0 e T_1, respectivamente. Em Kupinski & Anastasio (1999), os autores usaram $J_1(\mathbf{w}) = TPr(\mathbf{w})$ e $J_0(\mathbf{w}) = TNr(\mathbf{w})$; em Sanchez *et al.* (2005) e Everson & Fieldsend (2006a) foram adotados $J_1(\mathbf{w}) = FNr(\mathbf{w})$ e $J_0(\mathbf{w}) = FPr(\mathbf{w})$; e, no trabalho de Graening *et al.* (2006), foram sugeridos $J_1(\mathbf{w}) = TPr(\mathbf{w})$ e $J_0(\mathbf{w}) = FPr(\mathbf{w})$.

Em todos os trabalhos, algoritmos evolucionários multiobjetivo foram escolhidos para solucionar o problema (3.34). Ao final do processo de aprendizado, os algoritmos retornam uma estimativa para o conjunto de soluções não dominadas[1] denominado conjunto Pareto-ótimo. Todas as soluções são equivalentes na

[1]Considerando um problema de otimização multiobjetivo em que todos os funcionais J_k, com $k \in \{0, 1\}$, devem ser minimizados, uma solução $\mathbf{w}^*$ é dita ser não dominada, se não existe qualquer outra solução $\mathbf{w}$ tal que $\mathbf{J}(\mathbf{w}) \leq \mathbf{J}(\mathbf{w}^*)$ e $J_k(\mathbf{w}) < J_k(\mathbf{w}^*)$ para no mínimo um índice k.

ausência de qualquer informação referente aos objetivos $J_0(\mathbf{w})$ e $J_1(\mathbf{w})$ e podem ser interpretadas como pontos de operação de uma curva ROC ótima.

Nos trabalhos supracitados não foi proposta nenhuma estratégia de decisão para a escolha de uma solução (ou ponto de operação) no conjunto Pareto-ótimo. Os autores deixam a cargo do usuário escolher a solução cujo desempenho seja mais apropriado para a tarefa de aprendizado em questão.

3.3 Conclusões do capítulo

Esse capítulo forneceu uma revisão bibliográfica sobre a pesquisa desenvolvida no âmbito do aprendizado com dados desbalanceados. Foi visto que em cenários apresentando graus elevados de desequilíbrio entre as classes, a melhor estratégia é avaliar modelos usando critérios que dissociam o desempenho por classe, com base na matriz de confusão. Foi também recomendado o uso de medidas alternativas que focam na detecção da classe menos representativa (de interesse) ou consideram com mesma importância a discriminação de ambas as classes, tais como *F-measure* e *G-mean*.

Ainda no escopo de métricas de avaliação, foram descritos os princípios da análise ROC (*Receiver Operating Characteristic*). Foi visto que os gráficos ROC são ferramentas úteis em cenários desbalanceados, desde que eles permitem comparar modelos contrastando seus desempenhos de detecção (taxas de verdadeiros positivos) sobre uma faixa de distribuições a priori (ou valores de limiar). Além disso, eles podem ser usados para a seleção de pontos de operação, segundo um critério adotado pelo usuário. Foi também chamada a atenção para a independência das curvas ROC em relação às prioris das classes: se a proporção entre positivos e negativos muda na população alvo, sobre a qual o classificador é aplicado, a curva ROC não muda (Fawcett, 2006). Isso faz com que a métrica AUC, que corresponde à área abaixo da Curva ROC, leve vantagem sobre uma série de medidas, tais como acurácia (taxa de erro) e *precision*, que são diretamente influenciadas por mudanças nas distribuições das classes.

Na segunda parte do capítulo foi abordado o estado da arte das soluções propostas para o problema de classes desbalanceadas. Tais soluções foram divididas em duas categorias: *pré-processamento de dados* e *adaptações em algoritmos de*

aprendizado. Dentro dessa última categoria, foram descritos os principais trabalhos cujas propostas foram baseadas em modificações de funções custo, no contexto de máquinas de *kernel* e redes MLP.

Capítulo 4

Algoritmos Propostos

A análise do problema de classes desbalanceadas realizada no Capítulo 2 demonstrou que o viés imposto pela classe majoritária está principalmente relacionado ao critério otimizado no processo de aprendizado. Como a maioria dos algoritmos é projetada para minimizar uma aproximação empírica da probabilidade do erro global de classificação, o problema surge intrinsecamente.

Nesse capítulo, novos algoritmos de aprendizado são propostos com o objetivo de melhorar o desempenho de redes *MultiLayer Perceptron* (MLP) em aplicações desbalanceadas. A metodologia adotada no desenvolvimento dos algoritmos consistiu em reformular o treinamento padrão de MLPs, partindo-se da idéia de se considerar os desempenhos individuais por classe. Com base nesse ponto vista, critérios específicos para seleção de modelos puderam ser projetados com o propósito de atender as necessidades do problema em foco. São priorizadas taxas de acerto elevadas e equilibradas para ambas e classes, bem como a melhoria da qualidade do *ranking* de classificação.

O primeiro algoritmo de aprendizado, denominado WEMLP (Ponderação de Erros), otimiza uma função custo que permite ponderar as importâncias das classes no treinamento. A motivação teórica por trás de WEMLP está na violação da premissa de custos (perdas) iguais assumida rede pela MLP tradicional (baseada no erro global). Adicionalmente, a possibilidade de incorporação de informação a priori a partir de um parâmetro de custo, como na formulação do *risco* Bayesiano (Duda *et al.*, 2000), pode levar a modelos que não favorecem uma classe em particular (*unbiased*), melhorando a detecção da classe minoritária.

O segundo algoritmo de aprendizado, denominado AUCMLP (Otimização da *AUC*), aposta no conceito da maximização da "distância" entre as densidades das saídas (*scores*) produzidas pela MLP para as classes em separado. Formalmente, isso equivale à otimizar a métrica *AUC* (*Area Under the ROC Curve*). As principais motivações teóricas de AUCMLP estão na independência em relação às distribuições a priori apresentada pela *AUC*, bem como sua relação com a qualidade do *ranking* de classificação. Em contraste com a taxa de erro global, tais propriedades podem proporcionar vantagens práticas em problemas com níveis elevados de desbalanceamento e sobreposição entre as classes.

Na parte final do capítulo, as formulações propostas para WEMLP e AUCMLP são estendidas com o propósito de se controlar a complexidade (flexibilidade) dos modelos selecionados no processo de aprendizado. Isso é feito seguindo a metodologia do aprendizado multiobjetivo (MOBJ) para MLPs, proposta inicialmente em Teixeira *et al.* (2000).

O texto encontra-se organizado da seguinte forma: a Seção 4.1 descreve a topologia de rede MLP assim como a notação adotada na descrição dos algoritmos. Os fundamentos teóricos de WEMLP e AUCMLP são fornecidos, respectivamente, nas Seções 4.2 e 4.3. As extensões multiobjetivo são apresentadas na Seção 4.4. Por último, a Seção 4.5 traz as conclusões do capítulo.

4.1 Rede MLP

Como o escopo dos algoritmos propostos na tese é limitado a problemas binários (contendo somente duas classes), considere uma rede MLP com n entradas, uma camada escondida com h unidades (neurônios) e uma camada de saída contendo uma única unidade, conforme ilustrado pela Figura 4.1.

O valor de saída obtido na unidade escondida s da rede, como consequência da apresentação de um vetor de entrada $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, é dado pela seguinte expressão,

$$z_s = \phi\left(u_s\right) = \phi\left(\sum_{r=0}^{n} w_{sr}\, x_r\right) \tag{4.1}$$

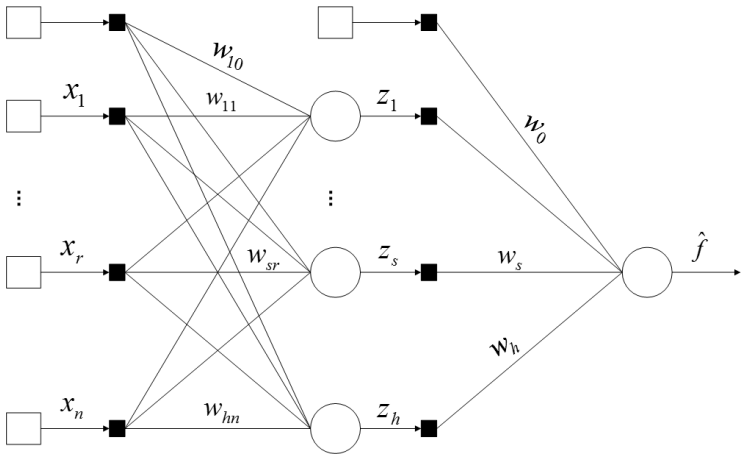


Figura 4.1: Topologia de rede *MultiLayer Perceptron* comumente adotada em problemas de classificação binária.

onde w_{sr} representa um peso entre a unidade escondida s e a unidade de entrada r; $\phi(\cdot)$ é a função de ativação. Similarmente, o valor obtido na unidade de saída da rede, é calculado com base nas saídas emitidas pelas unidades escondidas,

$$\hat{f} = \phi(v) = \phi\left(\sum_{s=0}^{h} w_s \, z_s\right) \tag{4.2}$$

na qual w_s representa um peso entre o neurônio de saída e a unidade escondida s. O termo *bias* foi considerado como uma unidade (entrada/escondida) extra com valor igual a 1. A classificação do exemplo $\mathbf{x}$ é obtida pelo sinal de $\hat{f}$.

Dado o conjunto de dados $T = \{(\mathbf{x}(i), y(i)) \mid i = 1 \ldots N\}$, com $y(i) \in \{+1, -1\}$ denotando o rótulo (saída desejada) para cada vetor de entrada $\mathbf{x}(i) \in \mathbb{R}^n$, a expressão geral do sinal de erro (estimado na saída da rede) para o i-ésimo exemplo de treinamento é definida como $e(i) = y(i) - \hat{f}(i)$.

4.2 Aprendizado por ponderação de erros

Essa seção fornece os fundamentos teóricos do algoritmo de aprendizado por Ponderação de Erros (WEMLP). Sua formulação é baseada na proposta de uma

função custo conjunta que leva em conta as contribuições individuais dos erros de cada classe. Os pesos dessas contribuições podem ser ajustados através de um único parâmetro incorporado à nova função custo. Uma relação entre o ajuste desse parâmetro e as propriedades da superfície de decisão obtida no espaço de características (induzido pela camada escondida da rede) é também apresentada.

A regra de atualização de pesos do método WEMLP é uma extensão da regra de *Levenberg-Marquadt* (Hagan & Menhaj, 1994), o que assegura sua eficiência computacional. Cabe ressaltar que, no contexto de redes MLP, o princípio da divisão dos erros entre as classes já havia sido proposto em Anand *et al.* (1993). Naquele trabalho, no entanto, o objetivo foi acelerar a velocidade de convergência do algoritmo *BackPropagation* em problemas desbalanceados.

4.2.1 Função custo conjunta

O conjunto de dados T pode ser reescrito como a união das classes positiva (ou minoritária) e negativa (ou majoritária), isto é, $T = \{T_1 \cup T_2\}$, com $T_1 = \{(\mathbf{x}(p), y(p)) \mid p = 1 \ldots N_1\}$ e $T_2 = \{(\mathbf{x}(q), y(q)) \mid q = 1 \ldots N_2\}$; Os rótulos são definidos como: $y(p) = +1 \ \forall \ \mathbf{x}(p) \in T_1$ e, $y(q) = -1 \ \forall \ \mathbf{x}(q) \in T_2$.

Para a obtenção de modelos sensíveis à importância de cada classe, uma nova função custo é proposta para a estimação dos parâmetros da rede MLP. Sua expressão é definida como a soma ponderada dos funcionais J_1 e J_2 que representam, respectivamente, o somatório dos erros quadráticos para os conjuntos T_1 e T_2,

$$J = \lambda \, J_1 + (1 - \lambda) \, J_2 \tag{4.3}$$

onde J_1 e J_2 são descritos pelas seguintes expressões,

$$J_1 = \sum_{p=1}^{N_1} e^2(p) \quad \forall \, \mathbf{x}(p) \in T_1 \tag{4.4}$$

$$J_2 = \sum_{q=1}^{N_2} e^2(q) \quad \forall \, \mathbf{x}(q) \in T_2 \tag{4.5}$$

e, o parâmetro $0 \leq \lambda \leq 1$ é usado para ponderar as contribuições de J_1 e J_2 na composição de J. Note que quando λ assume valor igual a $1/2$, o funcional J se torna equivalente ao somatório dos erros quadráticos sobre todos os exemplos de

treinamento (taxa de erro global). Caso contrário, se $\lambda \neq 1/2$, custos (perdas) desiguais são assinalados aos erros de cada classe. Essa estratégia ($\lambda \neq 1/2$) modifica a formulação padrão do aprendizado.

Como será mostrado na Seção 4.2.3, λ influencia a localização da superfície de decisão estimada durante o treinamento. Assim, caso N_2 seja maior que N_1, esse parâmetro pode ser usado para compensar o desbalanceamento das classes e obter uma superfície de decisão equilibrada.

4.2.2 Atualização dos pesos

A regra de aprendizado (otimização de parâmetros) de WEMLP é baseada no algoritmo *Levenberg-Marquadt*, que constitui uma aproximação para o método de *Newton* (Hagan & Menhaj, 1994; Luenberger, 1984). Para derivar essa regra, considere os parâmetros adaptativos da rede (pesos e bias) agrupados em um único vetor $\mathbf{w}$ D-dimensional com componentes $\{w_0, w_1, \ldots, w_D\}^T$ e o funcional (4.3) descrito na forma vetorial, como segue,

$$J(\mathbf{w}) = \lambda \left[\mathbf{e}_1^T(\mathbf{w})\mathbf{e}_1(\mathbf{w})\right] + (1-\lambda) \left[\mathbf{e}_2^T(\mathbf{w})\mathbf{e}_2(\mathbf{w})\right] \tag{4.6}$$

onde $\mathbf{e}_k(\mathbf{w}) = \{e(1), e(2), \ldots, e(N_k)\}^T$ é o vetor de erros para a classe T_k, obtido em relação ao vetor de pesos corrente $\mathbf{w}$. Assumindo uma aproximação quadrática local para o funcional (4.6) ao redor de $\mathbf{w}$, o termo de atualização de pesos usando o método de *Newton* é dado pela seguinte equação,

$$\Delta\mathbf{w} = -\left[\mathbf{H}(\mathbf{w})\right]^{-1}\mathbf{g}(\mathbf{w}) \tag{4.7}$$

na qual $\mathbf{H}(\mathbf{w})$ e $\mathbf{g}(\mathbf{w})$ correspondem, respectivamente, à matriz Hessiana e ao vetor gradiente. Note que essas grandezas podem também ser expressas como somas ponderadas, através das Hessianas e dos gradientes para os funcionais $J_1(\mathbf{w})$ e $J_2(\mathbf{w})$,

$$\mathbf{H}(\mathbf{w}) = \lambda\,\mathbf{H}_1(\mathbf{w}) + (1-\lambda)\,\mathbf{H}_2(\mathbf{w}) \tag{4.8}$$

$$\mathbf{g}(\mathbf{w}) = \lambda\,\mathbf{g}_1(\mathbf{w}) + (1-\lambda)\,\mathbf{g}_2(\mathbf{w}) \tag{4.9}$$

Desde que os funcionais $J_k(\mathbf{w})$ são somas de quadrados de funções não lineares (vide (4.4) e (4.5)), as Hessianas $\mathbf{H}_k(\mathbf{w})$ e os gradientes $\mathbf{g}_k(\mathbf{w})$ podem ser definidos da seguinte forma,

$$\mathbf{H}_k(\mathbf{w}) = \mathbf{Z}_k^T(\mathbf{w})\,\mathbf{Z}_k(\mathbf{w}) + \mathbf{S}_k(\mathbf{w}) \tag{4.10}$$

$$\mathbf{g}_k(\mathbf{w}) = \mathbf{Z}_k^T(\mathbf{w})\,\mathbf{e}_k(\mathbf{w}) \tag{4.11}$$

onde $\mathbf{Z}_k(\mathbf{w})$ é a matriz Jacobiana para a classe T_k,

$$\mathbf{Z}_k(\mathbf{w}) = \begin{bmatrix} \frac{\partial e(1)}{\partial w_0} & \frac{\partial e(1)}{\partial w_1} & \cdots & \frac{\partial e(1)}{\partial w_D} \\ \frac{\partial e(2)}{\partial w_0} & \frac{\partial e(2)}{\partial w_1} & \cdots & \frac{\partial e(2)}{\partial w_D} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial e(N_k)}{\partial w_0} & \frac{\partial e(N_k)}{\partial w_1} & \cdots & \frac{\partial e(N_k)}{\partial w_D} \end{bmatrix} \tag{4.12}$$

e cada elemento da matriz $\mathbf{S}_k(\mathbf{w})$ é dado por,

$$[\mathbf{S}_k(\mathbf{w})]_{lm} = \sum_{i\,\in\,T_k} e(i)\frac{\partial^2 e(i)}{\partial w_l \partial w_m} \tag{4.13}$$

Para o método *Gauss-Newton*, assume-se que $\mathbf{S}_k(\mathbf{w}) \approx 0$ e então, o cálculo da matriz Hessiana (4.10) passa a depender somente da Jacobiana $\mathbf{Z}_k(\mathbf{w})$. Note que essa aproximação para a matriz Hessiana é fácil de computar, uma vez que as derivadas parciais do erro em relação aos pesos da rede $\left(\frac{\partial e(i)}{\partial w_l}\right)$ podem ser obtidas usando a formulação do algoritmo *BackPropagation*[1] padrão (Rumelhart & McClelland, 1986). O termo de atualização de pesos (4.7) torna-se portanto,

$$\Delta\mathbf{w} = -\left[\hat{\mathbf{H}}(\mathbf{w})\right]^{-1}\mathbf{g}(\mathbf{w}) \tag{4.14}$$

onde, de acordo com (4.8), (4.9), (4.10) e (4.11),

$$\hat{\mathbf{H}}(\mathbf{w}) = \lambda\,\left[\mathbf{Z}_1^T(\mathbf{w})\mathbf{Z}_1(\mathbf{w})\right] + (1-\lambda)\,\left[\mathbf{Z}_2^T(\mathbf{w})\mathbf{Z}_2(\mathbf{w})\right] \tag{4.15}$$

[1] A expressão geral para o cálculo da derivada parcial do erro em relação a um peso arbitrário da rede $\left(\frac{\partial e(i)}{\partial w_l}\right)$ é fornecida no Apêndice B.

$$\mathbf{g}(\mathbf{w}) = \lambda \left[\mathbf{Z}_1^T(\mathbf{w})\mathbf{e}_1(\mathbf{w})\right] + (1-\lambda) \left[\mathbf{Z}_2^T(\mathbf{w})\mathbf{e}_2(\mathbf{w})\right] \tag{4.16}$$

Finalmente, a regra de aprendizado de *Levenberg-Marquardt* é obtida como uma modificação do método de *Gauss-Newton*,

$$\mathbf{w}_{new} = \mathbf{w}_{old} - \left[\hat{\mathbf{H}}(\mathbf{w}_{old}) + \mu\mathbf{I}\right]^{-1} \mathbf{g}(\mathbf{w}_{old}) \tag{4.17}$$

onde $\mathbf{I}$ é a matriz identidade e o parâmetro $\mu > 0$ deve ser variado apropriadamente durante o processo de minimização. Se o funcional $J(\mathbf{w})$ decresce a partir de um novo passo calculado por (4.17), o novo vetor de pesos torna-se válido, o valor de μ é dividido por algum fator (β) e uma nova iteração (época) é processada. Caso contrário, se $J(\mathbf{w})$ aumenta, então μ é multiplicado por β, o vetor de pesos antigo é mantido e um novo termo de atualização é computado. Isso é repetido até que um decréscimo em $J(\mathbf{w})$ seja obtido. Para valores pequenos do parâmetro μ, o algoritmo comporta-se como *Gauss-Newton* enquanto para valores grandes de μ, são gerados passos muito pequenos na direção negativa do vetor gradiente (gradiente descendente). Uma estratégia comum é usar $\mu = 0.1$ como ponto de partida e $\beta = 10$ (Bishop, 1995).

4.2.3 Análise do parâmetro λ

Essa seção fornece uma análise formal da influência do parâmetro de custo λ no algoritmo WEMLP. Cabe ressaltar que as idéias aqui discutidas constituem uma parte central de nossa contribuição para o problema do aprendizado indutivo com classes desbalanceadas.

A análise do parâmetro λ se baseia nas propriedades de um mínimo local ou global da função custo J (4.3) para o qual o algoritmo de treinamento deve convergir. Seja $\mathbf{w}^*$ o vetor de pesos que representa um mínimo local ou global. Uma condição necessária para que $\mathbf{w}^*$ seja um mínimo é que o vetor gradiente $\mathbf{g}(\mathbf{w})$ em relação ao vetor de pesos $\mathbf{w}$ seja próximo de zero em $\mathbf{w} \approx \mathbf{w}^*$. Consequentemente, para todo componente do vetor $\mathbf{g}(\mathbf{w})$ tem-se que,

$$\frac{\partial J}{\partial w_l} \approx 0\,,$$

$$\lambda \frac{\partial J_1}{\partial w_l} + (1-\lambda) \frac{\partial J_2}{\partial w_l} \approx 0\,,$$

$$\frac{\frac{\partial J_1}{\partial w_l}}{\frac{\partial J_2}{\partial w_l}} \approx -\frac{(1-\lambda)}{\lambda} \tag{4.18}$$

onde w_l representa um peso arbitrário da rede e a expressão geral para as derivadas parciais $\frac{\partial J_1}{\partial w_l}$ e $\frac{\partial J_2}{\partial w_l}$ é dada pela Equação (4.19), a seguir. Detalhes sobre o cálculo de $\frac{\partial e(i)}{\partial w_l}$ em (4.19) são apresentados no Apêndice B.

$$\frac{\partial J_k}{\partial w_l} = 2 \sum_{i \,\in\, T_k} e(i) \, \frac{\partial e(i)}{\partial w_l} \quad \forall\, \mathbf{x}(i) \in T_k \tag{4.19}$$

Considerando o componente do vetor $g(\mathbf{w})$ em relação ao termo de polarização (*bias*) do neurônio de saída, pode-se escrever que,

$$\frac{\sum_{p=1}^{N_1} (y(p) - \hat{f}(p)) \phi'\left(v(p)\right)}{\sum_{q=1}^{N_2} (y(q) - \hat{f}(q)) \phi'\left(v(q)\right)} \approx -\frac{(1-\lambda)}{\lambda} \tag{4.20}$$

Desde que $y(p) = +1 \;\forall\; \mathbf{x}(p) \in T_1$ e $y(q) = -1 \;\forall\; \mathbf{x}(q) \in T_2$, tem-se que,

$$\frac{\sum_{p=1}^{N_1} (1 - \hat{f}(p)) \phi'\left(v(p)\right)}{\sum_{q=1}^{N_2} (1 + \hat{f}(q)) \phi'\left(v(q)\right)} \approx \frac{1-\lambda}{\lambda} \tag{4.21}$$

A partir da Equação (4.21), considere a análise da influência de λ sob duas situações distintas: (i) o neurônio de saída possui função de ativação linear e, (ii) o neurônio de saída possui função de ativação não linear (sigmóide).

1. **Função de ativação linear no neurônio de saída**:

   Inicialmente, considere que o neurônio de saída da rede possui função de ativação linear e portanto, $\phi'\left(v(i)\right) = 1 \;\forall\; \mathbf{x}(i) \in T$. Nesse caso, uma superfície de decisão (linear) na forma de um hiperplano (dado por $\sum_s w_s \, \phi(\sum_r w_{sr} \, x_r) = 0$) é definida no espaço de características de dimensão h. Sob essa premissa, a expressão (4.21) torna-se,

$$\frac{\sum_{p=1}^{N_1} (1 - \hat{f}(p))}{\sum_{q=1}^{N_2} (1 + \hat{f}(q))} \approx \frac{1-\lambda}{\lambda} \tag{4.22}$$

Sejam $\mathbf{e}_1(\mathbf{w}) = \{1 - \hat{f}(p) \mid p = 1 \ldots N_1\}$ e $\mathbf{e}_2(\mathbf{w}) = \{1 + \hat{f}(q) \mid q = 1 \ldots N_2\}$, os vetores que armazenam os sinais de erro (em relação ao vetor solução $\mathbf{w}$) para os conjuntos T_1 e T_2, respectivamente. Tomando os valores médios desses vetores, denotados por $\bar{e}_1$ e $\bar{e}_2$, a Equação (4.22) pode ser reescrita da seguinte forma,

$$\frac{N_1}{N_2}\frac{\bar{e}_1}{\bar{e}_2} \approx \frac{1-\lambda}{\lambda},$$

$$\frac{\bar{e}_1}{\bar{e}_2} \approx \frac{N_2}{N_1}\frac{(1-\lambda)}{\lambda} \tag{4.23}$$

A partir de (4.23), pode-se verificar que se $\lambda = 1/2$, situação em que a função custo J corresponde ao erro global para o conjunto T, a razão entre os valores médios dos sinais de erro das classes é aproximadamente o inverso da razão entre os números de exemplos das mesmas. Assim, para um problema desbalanceado, em que $N2 > N1$, a seguinte condição é estabelecida,

$$\bar{e}_1 > \bar{e}_2 \quad \text{se} \quad N2 > N1 \tag{4.24}$$

Desde que $\bar{e}_1 = 1 - \bar{f}_1$ e $\bar{e}_2 = 1 + \bar{f}_2$, pode-se reescrever a condição (4.24) em termos de $\mathbf{f}_1(\mathbf{w})$ e $\mathbf{f}_2(\mathbf{w})$, vetores que armazenam os valores de saída para os conjuntos T_1 e T_2, respectivamente,

$$\bar{f}_1 < -\bar{f}_2 \quad \text{se} \quad N2 > N1 \tag{4.25}$$

Sabe-se que $\hat{f}(i)$, o valor obtido na saída da rede para o i-ésimo vetor de entrada, é diretamente proporcional à distância algébrica (medida no espaço de características) do vetor $\mathbf{x}(i)$ ao hiperplano de decisão (Duda *et al.*, 2000), ou seja,

$$\hat{f}(i) \propto d(i) \;\; \forall \;\; \mathbf{x}(i) \in T \tag{4.26}$$

onde $d(i)$ corresponde à distância algébrica do i-ésimo vetor de entrada ao hiperplano de decisão; $d(i)$ é positivo se $\mathbf{x}(i)$ estiver do lado positivo

do hiperplano e negativo se estiver do lado negativo; se $d(i) = 0$, $\mathbf{x}(i)$ se encontra exatamente no hiperplano de separação.

Analisando a condição (4.25) através de (4.26), pode-se afirmar que, para um conjunto de treinamento desbalanceado, o hiperplano de decisão deve se encontrar mais próximo dos exemplos de T_1 (classe minoritária) do que dos exemplos de T_2 (classe majoritária). Essa conclusão se apóia na seguinte relação,

$$\bar{f}_1 < -\bar{f}_2 \Rightarrow \bar{d}_1 < -\bar{d}_2 \tag{4.27}$$

na qual $\bar{d}_1$ e $\bar{d}_2$ representam as distâncias médias dos exemplos de treinamento ao hiperplano de decisão, calculadas para os conjuntos T_1 e T_2, respectivamente.

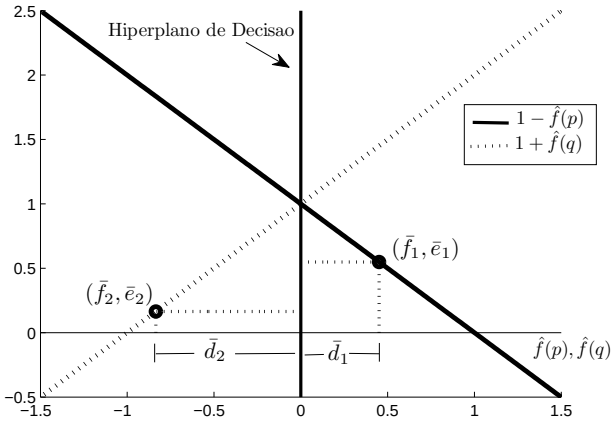


Figura 4.2: Sinais de erro $e(p) = 1 - \hat{f}(p)$ (linha contínua) e $e(q) = 1 + \hat{f}(q)$ (linha pontilhada) em função de $\hat{f}(p)$ e $\hat{f}(q)$, para $-1.5 \leq \hat{f}(p), \hat{f}(q) \leq 1.5$. Os pontos marcados nos gráficos, $(\bar{f}_1, \bar{e}_1)$ e $(\bar{f}_2, \bar{e}_2)$, correspondem respectivamente aos valores médios dos sinais de erro para os conjuntos T_1 e T_2, quando $N_2 > N_1$.

A Figura 4.2 ilustra os gráficos que representam os sinais de erro $e(p) = 1 - \hat{f}(p)$ (linha contínua) e $e(q) = 1 + \hat{f}(q)$ (linha pontilhada) em função de $\hat{f}(p)$ e $\hat{f}(p)$, para $-1.5 \leq \hat{f}(p), \hat{f}(q) \leq 1.5$. Considerando que $N2 > N1$, valores hipotéticos representando as médias dos sinais de erro, $\bar{e}_1$ e $\bar{e}_2$, foram

marcados nos gráficos juntamente com as respectivas distâncias médias, $\bar{d}_1$ e $\bar{d}_2$, em relação ao hiperplano, representado pelo eixo vertical $\hat{f}(i) = 0$. A simetria dos gráficos confirma a relação estabelecida em (4.27) e mostra que, para se obter um hiperplano equilibrado no espaço de características, situação na qual $\bar{d}_1 = -\bar{d}_2$, é necessário que $\bar{e}_1 \approx \bar{e}_2$. De acordo com (4.23), essa condição é satisfeita com a incorporação da seguinte informação a priori ao ajuste do parâmetro λ,

$$\lambda = \frac{N_2}{N_1 + N_2} \tag{4.28}$$

o que leva à seguinte relação,

$$\bar{e}_1 \approx \bar{e}_2 \Rightarrow \bar{f}_1 \approx -\bar{f}_2 \Rightarrow \bar{d}_1 \approx -\bar{d}_2 \tag{4.29}$$

2. **Função de ativação não linear no neurônio de saída:**

   Considere agora que o neurônio de saída possui função de ativação não linear como, por exemplo, a função sigmóide tangente hiperbólica[1], descrita pela Equação (4.30). Nesse caso, o intervalo dos valores de saída é $-1 \leq \phi\left(v(i)\right) \leq 1$; uma função de decisão não linear com *threshold* em 0 é então definida no espaço de características de dimensão h (número de unidades escondidas).

$$\phi\left(v(i)\right) = \frac{\exp\left(v(i)\right) - \exp\left(-v(i)\right)}{\exp\left(v(i)\right) + \exp\left(-v(i)\right)} \quad \forall \ \mathbf{x}(i) \in T \tag{4.30}$$

   e,

$$\phi'\left(v(i)\right) = 1 - \hat{f}^{\,2}(i) \quad \forall \ \mathbf{x}(i) \in T \tag{4.31}$$

   Substituindo (4.31) em (4.21), obtém-se a seguinte expressão,

[1] No projeto de classificadores baseados em redes MLP, funções de ativação do tipo sigmóide logística e sigmóide tangente hiperbólica são comumente usadas (Haykin, 1994).

$$\frac{\sum_{p=1}^{N_1}\left(1-\hat{f}(p)\right)\left(1-\hat{f}^{\,2}(p)\right)}{\sum_{q=1}^{N_2}\left(1+\hat{f}(q)\right)\left(1-\hat{f}^{\,2}(q)\right)} \approx \frac{1-\lambda}{\lambda} \tag{4.32}$$

De forma equivalente a (4.23), pode-se expressar (4.32) a partir dos valores médios de $\delta_1(\mathbf{w})$ e $\delta_2(\mathbf{w})$, vetores que armazenam os gradientes locais (do neurônio de saída) para os conjuntos T_1 e T_2, respectivamente. A seguinte Equação é obtida,

$$\frac{\bar{\delta}_1}{\bar{\delta}_2} \approx \frac{N_2}{N_1}\frac{(1-\lambda)}{\lambda} \tag{4.33}$$

onde, $\bar{\delta}_1$ e $\bar{\delta}_2$ correspondem, respectivamente, aos valores médios dos vetores $\delta_1(\mathbf{w}) = \{(1-\hat{f}(p))(1-\hat{f}^{\,2}(p)) \mid p = 1 \ldots N_1\}$ e $\delta_2(\mathbf{w}) = \{(1+\hat{f}(q))(1-\hat{f}^{\,2}(q)) \mid q = 1 \ldots N_2\}$.

Analisando (4.33), para o caso em que a função de custo J equivale ao erro global, i.e., $\lambda = \frac{1}{2}$, a seguinte condição pode ser estabelecida para um conjunto de treinamento desbalanceado,

$$\bar{\delta}_1 > \bar{\delta}_2 \quad \text{se} \quad N2 > N1 \tag{4.34}$$

Considere a Figura 4.3 a seguir, que plota os gradientes locais $\delta(p) = (1-\hat{f}(p))(1-\hat{f}^{\,2}(p))$ (linha contínua) e $\delta(q) = (1+\hat{f}(q))(1-\hat{f}^{\,2}(q))$ (linha pontilhada) em função de $\hat{f}(p)$ e $\hat{f}(q)$, para $-1.0 \leq \hat{f}(p), \hat{f}(q) \leq 1.0$. Observe, através dessa figura, que a seguinte relação,

$$\delta(p) > \delta(q) \Rightarrow \hat{f}(p) < -\hat{f}(q) \tag{4.35}$$

é válida nos intervalos $I_1 = \{-0.335 \leq \hat{f}(p) \leq 1\}$ e $I_2 = \{-1 \leq \hat{f}(q) \leq 0.335\}$, cujos limites são representados pelas linhas tracejadas na Figura 4.3.

Desde que a função sigmóide tangente hiperbólica é monotônica crescente sobre toda a faixa de seu argumento e tomando o seu inverso, i.e.,

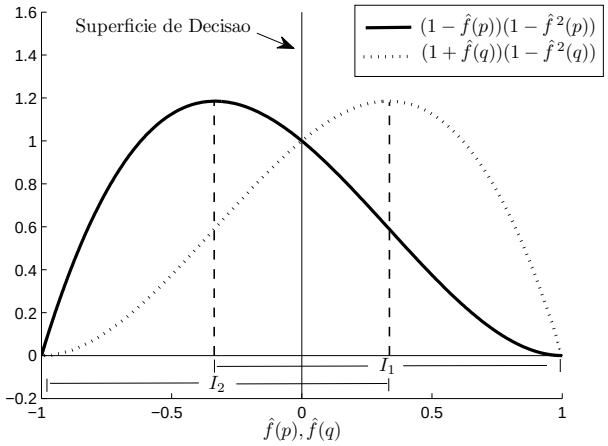


Figura 4.3: Gradientes locais $\delta(p) = (1 - \hat{f}(p))(1 - \hat{f}^2(p))$ (linha contínua) e $\delta(q) = (1 + \hat{f}(q))(1 - \hat{f}^2(q))$ (linha pontilhada) em função de $\hat{f}(p)$ e $\hat{f}(q)$, para $-1.0 \leq \hat{f}(p), \hat{f}(q) \leq 1.0$. A relação $\delta(p) > \delta(q) \Rightarrow \hat{f}(p) < -\hat{f}(q)$ é válida para valores de $\hat{f}(p) \in I_1$ e $\hat{f}(q) \in I_2$.

$\phi^{-1}(\hat{f}(i)) = v(i)$, os intervalos I_1 e I_2 podem ser redefinidos em termos dos valores lineares $v(p)$ e $v(q)$ obtidos na saída da rede, ou seja, $I_1 = \{-0.347 \leq v(p) \leq 5.3\}$ e $I_2 = \{-5.3 \leq v(q) \leq 0.347\}$. Os limites, superior para I_1 e inferior para I_2, correspondem aos valores de $v(i)$ que causam a saturação da unidade de saída. A relação (4.35) pode então ser reescrita em termos de $v(p)$ e $v(q)$, para os novos intervalos I_1 e I_2,

$$\delta(p) > \delta(q) \Rightarrow v(p) < -v(q) \tag{4.36}$$

Sejam v_1 e v_2 as saídas lineares (abscissas) correspondentes aos valores médios dos gradientes locais $\bar{\delta_1}$ e $\bar{\delta_2}$. Assumindo que $v_1 \in I_1$ e $v_2 \in I_2$ e retomando a condição (4.34), que considera um conjunto desbalanceado com $N_2 > N_1$, a seguinte relação é válida,

$$\bar{\delta_1} > \bar{\delta_2} \Rightarrow v_1 < -v_2 \tag{4.37}$$

Considere novamente $d(i)$, como a distância algébrica do i-ésimo vetor de entrada à superfície de decisão estimada no espaço de características; como,

por definição, $v(i) \propto d(i)\ \forall\ \mathbf{x}(i) \in T$, a análise da relação (4.37) sugere que, quando $N_2 > N_1$, a superfície de separação deve se encontrar mais próxima dos exemplos de T_1 (classe minoritária) do que dos exemplos de T_2 (classe majoritária).

Além disso, considerando a simetria das curvas obtidas para $\delta(p)$ e $\delta(q)$ em relação ao eixo vertical $\hat{f}(i) = 0$ (vide Figura 4.3), pode-se estabelecer a seguinte condição válida dentro dos intervalos I_1 e I_2,

$$v_1 = -v_2 \quad \text{se} \quad \bar{\delta}_1 = \bar{\delta}_2 \tag{4.38}$$

E assim, pode-se tentar obter uma superfície de decisão mais equilibrada, balanceando os gradientes locais médios $\bar{\delta}_1$ e $\bar{\delta}_2$. De acordo com a Equação (4.33), isso pode ser alcançado através do seguinte ajuste no parâmetro λ,

$$\lambda = \frac{N_2}{N_1 + N_2} \tag{4.39}$$

o que leva à seguinte relação,

$$\bar{\delta}_1 \approx \bar{\delta}_2 \Rightarrow v_1 \approx -v_2 \Rightarrow d_1 \approx -d_2 \tag{4.40}$$

Conforme esperado, o ajuste sugerido para λ no caso da rede MLP possuir ativação não linear no neurônio de saída (4.39) coincide com o resultado obtido anteriormente para o caso de ativação linear (4.28).

4.2.4 Exemplos com dados sintéticos

Com o objetivo de ilustrar a análise descrita na seção anterior e mostrar o efeito causado por λ na localização da superfície de decisão, experimentos foram conduzidos com um conjunto de dados sintético gerado a partir de distribuições gaussianas bidimensionais, com vetores de média $[0, 0]^T$ e $[2, 2]^T$ e matrizes de covariância equivalentes à matriz identidade. A razão entre os números de exemplos negativos (círculos) e positivos (cruzes) é 10:1.

A Figura 4.4 mostra as superfícies de decisão estimadas por uma rede MLP com topologia 2:3:1 e saída não linear (tangente hiperbólica). A linha pontilhada

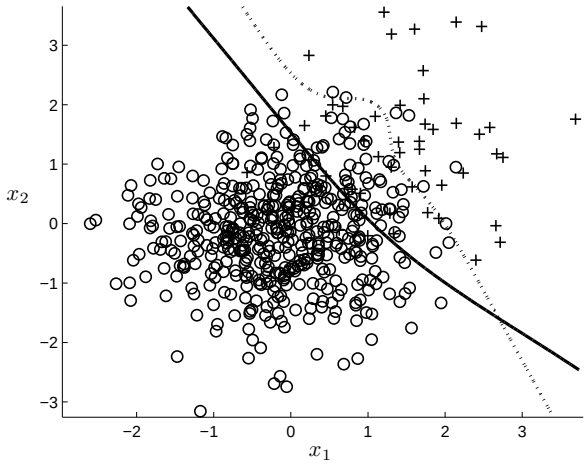
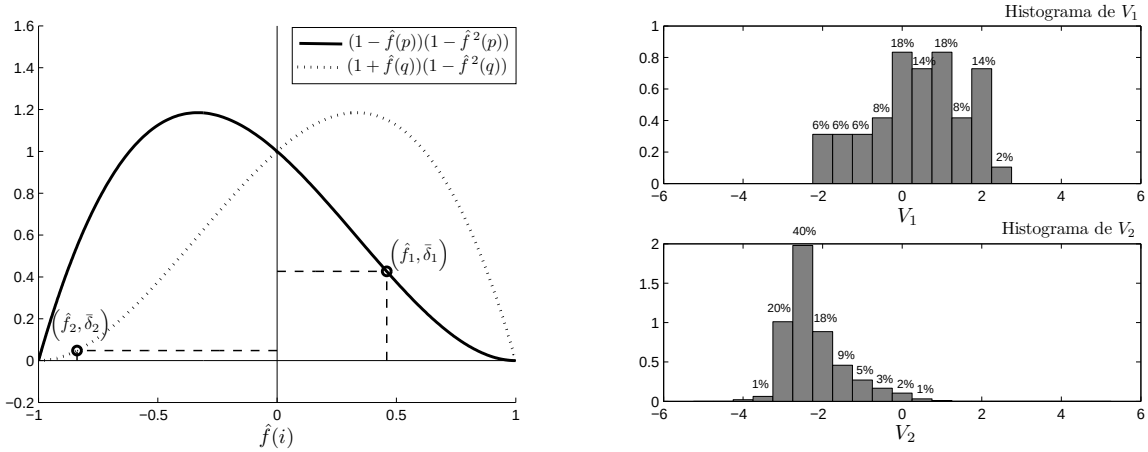


Figura 4.4: Efeito causado por λ nas superfícies de decisão estimadas por uma rede MLP com topologia 2:3:1. Os ajustes foram: (i) solução padrão (linha pontilhada) $\Rightarrow \lambda = 1/2$; (ii) solução balanceada (linha contínua) $\Rightarrow \lambda = N_2/(N_1 + N_2)$. Em ambos os casos, o aprendizado foi inicializado a partir do mesmo vetor de parâmetros.

corresponde à solução padrão que minimiza a taxa de erro global, com λ igual a $1/2$; o desempenho obtido sobre o conjunto de treinamento foi: $TPr = 0.56$ e $TNr = 0.99$. A linha contínua corresponde à solução balanceada obtida através do ajuste de λ conforme indicado pela Equação (4.39); o desempenho obtido foi: $TPr = 0.90$ e $TNr = 0.89$. Em ambas as situações, o treinamento foi inicializado a partir do mesmo vetor de parâmetros.

Considere também a Figura 4.5a que mostra os valores médios dos gradientes locais, obtidos ao final do aprendizado, para a solução padrão (linha pontilhada na Figura 4.4). Conforme descrito na Seção 4.2.3, a diferença entre $\bar{\delta}_1$ e $\bar{\delta}_2$, causada pelo desbalanceamento das classes, implica na obtenção de uma superfície de decisão desviada em direção à classe minoritária.

Esse fato pode ser comprovado a partir dos histogramas (vide Figura 4.5b) que ilustram as densidades de V_1 e V_2, variáveis que representam os valores lineares de saída da rede para T_1 e T_2, respectivamente. A área de cada retângulo nos histogramas, corresponde à frequência relativa (dada em porcentagem) da respectiva faixa de valores da variável cuja amplitude é de 0.5. Note que V_1

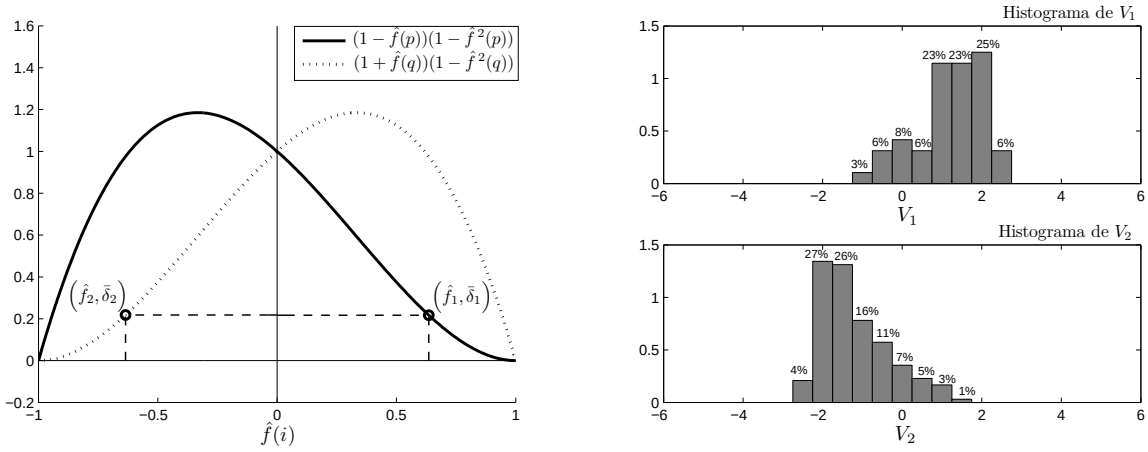


(a) (b)

Figura 4.5: Características apresentadas pela solução padrão (linha pontilhada na Figura 4.4) após o aprendizado. A Figura 4.5a (à esquerda) mostra os valores médios obtidos para os gradientes locais com suas respectivas abscissas: $(\hat{f}_1, \bar{\delta}_1)$ e $(\hat{f}_2, \bar{\delta}_2)$. A diferença apresentada entre $\bar{\delta}_1$ e $\bar{\delta}_2$ é devido à minimização do erro global a partir de um conjunto de treinamento desbalanceado. A Figura 4.5b (à direita) traz os histogramas referentes às distribuições de V_1 (superior) e V_2 (inferior), valores lineares de saída da rede para T_1 e T_2, respectivamente. A área de cada retângulo corresponde à frequência relativa (dada em porcentagem) da respectiva faixa de valores da variável cuja amplitude é de 0.5. As diferenças entre as distribuições comprovam o desvio da superfície de decisão em direção à classe minoritária.

(histograma superior) concentra aproximadamente 66% de seus valores entre as faixas $-0.75 \leq V_1 \leq 1.75$, apresentando uma considerável porcentagem (cerca de 44%) abaixo do *threshold* de decisão $V_1 = 0$. Por outro lado, V_2 (histograma inferior) concentra suas observações (aproximadamente 78%) entre as faixas $-3.25 \leq V_2 \leq -1.75$, com apenas cerca de 1% acima de $V_2 = 0$. As médias e variâncias obtidas para V_1 e V_2 foram aproximadamente: $\bar{v}_1 = 0.39$, $\bar{v}_2 = -2.20$, $\sigma^2_{v_1} = 1.43$ e $\sigma^2_{v_2} = 0.56$.

Diferente da solução padrão, a solução balanceada (linha contínua na Figura 4.4), por igualar os valores médios dos gradientes locais (vide Figura 4.6a), apresenta uma superfície de decisão mais equilibrada, conforme ilustrado pelos his-



(a) (b)

Figura 4.6: Características apresentadas pela solução balanceada (linha contínua na Figura 4.4) após o aprendizado. A Figura 4.6a (à esquerda) mostra equilíbrio entre os valores médios dos gradientes locais obtidos através do ajuste $\lambda = N_2/(N_1 + N_2)$. A Figura 4.6b (à direita) mostra os histogramas referentes às distribuições de V_1 (superior) e V_2 (inferior), valores lineares de saída da rede para T_1 e T_2, respectivamente. Como pode ser observado, V_1 e V_2 possuem valores distribuídos em faixas similares, porém simétricas em relação ao *threshold* de decisão $(V_1, V_2 = 0)$. Isso evidencia o equilíbrio da superfície de separação estimada.

togramas de V_1 e V_2 na Figura 4.6b. Observe que V_1 e V_2 possuem valores distribuídos em faixas similares, porém simétricas em relação ao *threshold* de decisão $(V_1, V_2 = 0)$. Assim, a solução balanceada foi capaz de diminuir as discrepâncias entre as médias e as variâncias obtidas para V_1 e V_2: $\bar{v}_1 = 1.21$, $\bar{v}_2 = -1.17$, $\sigma^2_{v_1} = 0.72$ e $\sigma^2_{v_2} = 0.76$. Além disso, a porcentagem de erros (cerca de 10%) foi aproximadamente a mesma para cada classe.

Usando o mesmo conjunto de dados, o experimento a seguir mostra que é possível controlar o aprendizado de redes MLP no espaço ROC através do ajuste de λ. A Figura 4.7 apresenta os valores de TPr e TNr coletados ao final do treinamento, em função de λ, para o seguinte intervalo $0.50 \leq \lambda \leq 0.975$. Cada par $(TPr,\ TNr)$ representa o valor médio sobre 7 execuções. O desvio padrão correspondente é apresentado em forma de barra vertical. Para cada execução, o

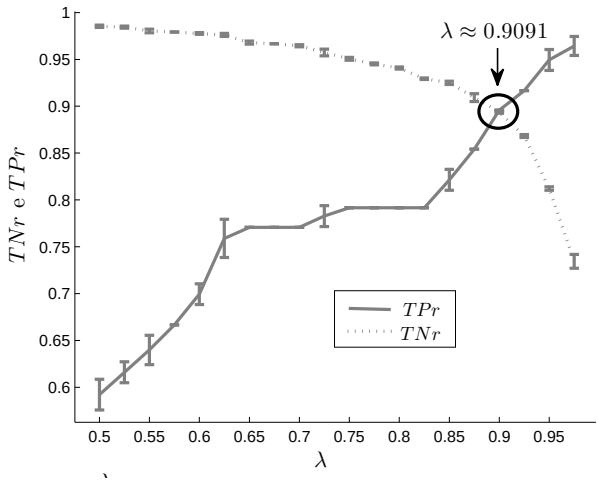


Figura 4.7: Valores de TPr e TNr em função de λ para o seguinte intervalo $0.50 \leq \lambda \leq 0.975$. Cada par ($TPr$, TNr) representa o valor médio sobre 7 execuções. O desvio padrão correspondente é apresentado em forma de barra vertical. Para cada execução, o treinamento foi inicializado a partir do mesmo vetor de parâmetros.

treinamento foi inicializado a partir do mesmo vetor de parâmetros. A topologia de rede MLP utilizada nesse experimento foi 2:3:1.

Conforme ilustrado pela Figura 4.7, soluções com valores elevados de TPr e TNr foram obtidas para os valores extremos do intervalo $0.50 \leq \lambda \leq 0.975$. Como esperado, superfícies de decisão mais equilibradas são alcançadas quando $\lambda \approx N_2/(N_1 + N_2)$, ou seja, desde que $N_2 = 480$ e $N_1 = 48$, o equilíbrio ocorre em torno de 0.9091.

4.3 Otimização da AUC

Classificadores binários baseados em redes MLP comumente usam função de ativação do tipo sigmóide na unidade (neurônio) de saída. Com base nas saídas contínuas (*scores*) obtidas para os padrões de entrada, um *threshold* é selecionado para definir a pertinência de um exemplo a uma dada classe. Essa abordagem permite a contabilização do número médio de erros/acertos, métricas comumente usadas para se determinar o desempenho do classificador.

Para muitas aplicações reais, no entanto, a taxa de erro não é a métrica mais pertinente para se avaliar um classificador. Critérios como ordenação ou *ranking* são mais apropriados. Considere, por exemplo, uma lista de documentos retornada por uma máquina de busca a partir de uma consulta específica. A lista pode conter muitos documentos mas, na prática, somente aqueles que estão no topo devem ser examinados pelo usuário. Nesse caso, ordenar os documentos segundo seus *scores* é mais crítico do que simplesmente avaliar a taxa de erro global sobre as classificações de todos os documentos como relevantes ou não.

A AUC (*Area Under the ROC Curve*) é uma métrica robusta que avalia o desempenho geral do classificador sem considerar um *threshold* de decisão específico. Na literatura de aprendizado de máquina, a AUC tem sido frequentemente usada para medir a qualidade do *ranking* de classificação (Hanley & Mcneil, 1982). Além disso, por ser independente do *threshold*, e consequentemente das probabilidades a priori das classes (vide Seção 3.1.1.1 do Capítulo 3), obtém vantagem em relação à taxa de erro quando aplicada a problemas com elevadas desproporções entre as classes (Bradley, 1997).

Em geral, a função custo otimizada pela maioria dos algoritmos de aprendizado é a taxa de erro e não a AUC. Entretanto, como será descrito mais adiante na Seção 4.3.2, otimizar o erro em determinados casos, não garante a maximização da AUC. Dessa forma, é necessário um algoritmo que diretamente otimize a AUC. Alguns trabalhos na literatura visam ao tratamento desse problema. Um método para otimizar a AUC localmente foi proposto no contexto de Árvores de Decisão (Ferri *et al.*, 2002). Outros algoritmos foram desenvolvidos para maximizar aproximações globais da AUC (Herschtal & Raskutti, 2004; Herschtal *et al.*, 2006). Estudo de Cortes & Mohri (2004) mostrou que, sob certas condições, a função otimizada pelo algoritmo *RankBoost* (Freund *et al.*, 2003) é exatamente a AUC. Joachims (2005) chamou a atenção para a dificuldade computacional na otimização de medidas de desempenho não lineares e multivariadas, como é o caso da AUC, e apresentou um método baseado em vetores de suporte para a otimização daquela métrica.

Nessa seção, os fundamentos teóricos de um novo algoritmo de aprendizado (AUCMLP) que diretamente otimiza a AUC são descritos. AUCMLP é baseado

em uma função custo que corresponde a uma aproximação diferenciável da estatística de *Wilcoxon-Mann-Whitney*. Enquanto os estudos supracitados maximizam a AUC com o objetivo de melhorar a qualidade do *ranking* em aplicações específicas como, por exemplo, em Recuperação de Informação, nosso intuito é, principalmente, avaliar a AUC como função custo alternativa para melhorar o desempenho de redes MLP sobre aplicações desbalanceadas.

4.3.1 *Area Under the ROC Curve*

A AUC associada a um classificador $\hat{f}$, avaliado sobre o conjunto T, pode ser expressa como a probabilidade $P\left(\hat{f}(\mathbf{X}^+) > \hat{f}(\mathbf{X}^-)\right)$, onde $\hat{f}(\mathbf{X}^+)$ e $\hat{f}(\mathbf{X}^-)$ correspondem, respectivamente, às densidades (pdfs) das saídas (*scores*) estimadas por $\hat{f}$ para os exemplos positivos e negativos. A expressão dessa probabilidade para o caso discreto é equivalente à estatística de *Wilcoxon-Mann-Whitney* (Mann & Whitney, 1947; Wilcoxon, 1945), e descrita como segue,

$$AUC(\hat{f}) = \frac{1}{N_1 N_2}\left(\sum_{p=1}^{N_1}\sum_{q=1}^{N_2} G\left(\hat{f}(\mathbf{x}(p)) - \hat{f}(\mathbf{x}(q))\right)\right) \tag{4.41}$$

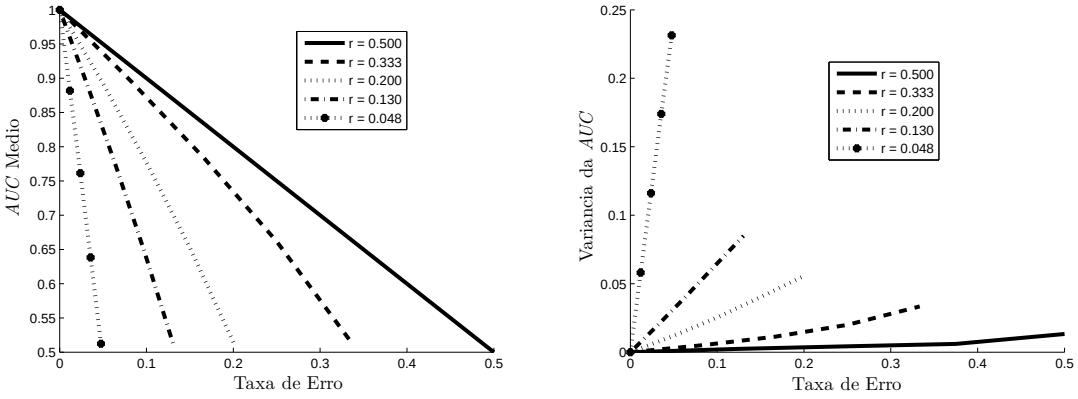
onde o funcional $G(t)$ é definido por,

$$G(t) = \begin{cases} 0 & \text{se } t < 0 \\ 0.5 & \text{se } t = 0 \\ 1 & \text{se } t > 0 \end{cases} \tag{4.42}$$

A AUC pode ser vista como uma medida baseada em comparações par a par entre classificações de ambas as classes. Com um *ranking* perfeito, todas os exemplos da classe positiva possuirão *scores* mais elevados que os da classe negativa e assim, $AUC\left(\hat{f}\right) = 1$.

4.3.2 Propriedades da *AUC*

Dois classificadores podem apresentar a mesma taxa de erro mas valores diferentes de AUC. De fato, para um dado *threshold* de classificação (θ), uma reordenação arbitrária dos exemplos com saídas maiores que θ claramente não afeta a taxa de erro mas leva a diferentes valores de AUC. Similarmente, pode-se reordenar os exemplos com saída menores que θ sem alterar a taxa de erro.



(a) (b)

Figura 4.8: Análise da média (à esquerda) e da variância (à direita) da AUC em função da taxa de erro para diferentes razões de desbalanceamento.

Em outras palavras, pode-se dizer que geralmente não há, na comparação entre as métricas erro e AUC, uma correspondência um para um entre seus valores. Dado um valor fixo k para o número de erros, é possível obter *rankings* (ou classificações) distintos, os quais podem produzir valores desiguais de AUC.

Levando em conta os aspectos acima mencionados, Cortes & Mohri (2004) formalizaram as correspondentes expressões do valor médio (esperado) e variância da AUC sobre todas as classificações que produzem um número fixo de erros (k). Cada expressão foi derivada como uma função dos parâmetros k, N_1 e N_2; o que permitiu que elas fossem usadas para comparar o comportamento assumido pelas métricas AUC e taxa de erro sob diversos cenários.

A Figura 4.8a ilustra o valor médio da AUC em função da taxa de erro para diferentes razões de desbalanceamento $r = N_1/(N_1 + N_2)$. Cada curva, representando uma dada razão r, foi obtida decrementando-se a taxa de erro a partir de $N_1/(N_1 + N_2)$ até 0. A curva $r = 0.5$ (linha contínua) nos mostra que o valor médio da AUC coincide com a Acurácia (1− erro) quando as distribuições são balanceadas ($N_1 = N_2$). O mesmo não pode ser dito para as demais curvas em que $N_1 \neq N_2$. Verifica-se, no entanto, para todas as curvas, que o valor médio da AUC cresce monotonicamente com a Acurácia na faixa entre 0.5 e 1.0. Com base nessas observações, parece não haver vantagem em se desenvolver

algoritmos específicos para maximizar a AUC, já que um algoritmo que minimiza o erro indiretamente otimiza a AUC (Cortes & Mohri, 2004).

Essa hipótese, no entanto, pode ser contrariada a partir da análise da variância da AUC em função da taxa de erro, ilustrada pela Figura 4.8b. Nota-se nessa figura que a variância da AUC cresce com o número de erros e também com o grau de desproporção (r) das classes. Esse comportamento exibido pela variância demonstra que em cenários muito desbalanceados, classificadores com a mesma taxa de erro podem apresentar valores bem distintos de AUC; o que sugere que algoritmos projetados para diretamente otimizar a AUC devem produzir melhores resultados do que aqueles que o fazem indiretamente através da minimização do erro global (Cortes & Mohri, 2004).

Os principais conceitos discutidos em Cortes & Mohri (2004) podem ser ilustrados numericamente a partir do seguinte exemplo. Considere 4 conjuntos de dados de tamanhos iguais ($N = 200$) mas com diferentes proporções entre positivos e negativos, $N_1 = \{100, 75, 50, 25\}$ e $N_2 = \{100, 125, 150, 175\}$, o que produz as seguintes razões de desbalanceamento, $r = \{0.5, 0.375, 0.25, 0.125\}$. Fixando-se o número de erros em $k = 20$, os valores de AUC em função da variação do número de falsos positivos ($0 \leq FP \leq k$) foram calculados[1] para cada conjunto de dados. Desde que k é fixo, o aumento de FP provoca intrinsecamente o decréscimo do número de falso negativos (FN). A Figura 4.9 apresenta as curvas de AUC obtidas em função de FP para cada conjunto representando uma dada razão r.

Observe pela curva $r = 0.5$ (linha contínua) que se as classes são balanceadas ($N_1 = N_2$) e o número de erros é pequeno ($k = 20$) em relação ao tamanho total do conjunto ($N = 200$), os valores de AUC tendem a ser iguais, fazendo com que a variância seja praticamente nula. Entretanto, quando $N_1 \neq N_2$, os valores de AUC estimados para uma dada razão r são distintos e decrescem monotonicamente com a diminuição do número de falsos positivos. Nesses casos, as variâncias da AUC são não nulas e crescem de forma inversa com o grau de desbalanceamento (r).

[1] O valor correspondente de AUC para um número fixo de falsos positivos (FP) foi obtido segundo expressão fornecida em Cortes & Mohri (2004).

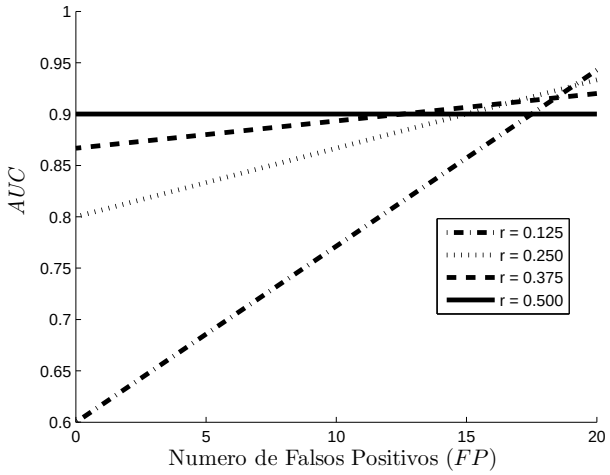


Figura 4.9: Valores de AUC em função do número de falsos positivos para diferentes razões de desbalanceamento.

O exemplo também mostra que, para conjuntos desbalanceados ($N_1 \neq N_2$), a AUC é sensível aos diferentes tipos de erros. Note a partir da Figura 4.9 que os valores de AUC decrescem à medida que o número de falsos positivos diminui e, consequentemente, o número de falsos negativos aumenta. Por outro lado, a taxa de erro se mantém constante e igual a 10% para todos os conjuntos de dados, uma vez que o número total de erros (k) e exemplos ($N_1 + N_2$) foi mantido invariável.

Essa sensibilidade da AUC em função do número de falsos positivos sugere que, em cenários muito desbalanceados, um classificador que diretamente otimize a AUC deve priorizar *scores* mais altos para os exemplos da classe positiva, melhorando a qualidade do *ranking* e consequentemente, o número de classificações corretas da classe minoritária. Além disso, por ser uma medida geral de desempenho do classificador, independente do limiar (*threshold*) de classificação, a AUC não deve sofrer do viés imposto pelo grupo majoritário, como ocorre na minimização do erro global.

4.3.3 Definição da função custo

Desde que a expressão original da estatística de *Wilcoxon-Mann-Whitney*, apresentada em (4.41), é não diferenciável, Yan *et al.* (2003) propõem uma estratégia

de suavização através da substituição da função degrau $G(t)$, vide Equação (4.42), pela função $R(t)$ definida por,

$$R(t) = \begin{cases} (-(t-\kappa))^{\tau} & \text{se } t < \kappa, \\ 0 & \text{caso contrário.} \end{cases} \tag{4.43}$$

para $0 < \kappa \leq d_{max}$ e $\tau > 1$. Considere $\hat{f}(p)$ e $\hat{f}(q)$, respectivamente, os *scores* estimados pela rede MLP para o p-ésimo exemplo positivo e o q-ésimo exemplo negativo; Seja $d(p,q) = \hat{f}(p) - \hat{f}(q)$, a diferença entre esses *scores*; d_{max} é o maior valor que pode ser obtido para $d(p,q)$; por exemplo, para uma rede MLP com saídas no intervalo, $-1 \leq \hat{f}(i) \leq 1$, $d_{max} = 2$. A Figura 4.10 ilustra gráficos de $G(t)$ (linha contínua) e $R(t)$ (linha pontilhada) em função da diferença $d(p,q)$, para o intervalo $-2 \leq d(p,q) \leq 2$. Para a curva $R(t)$, os seguintes parâmetros foram usados: $\kappa = 1.2$ e $\tau = 2$.

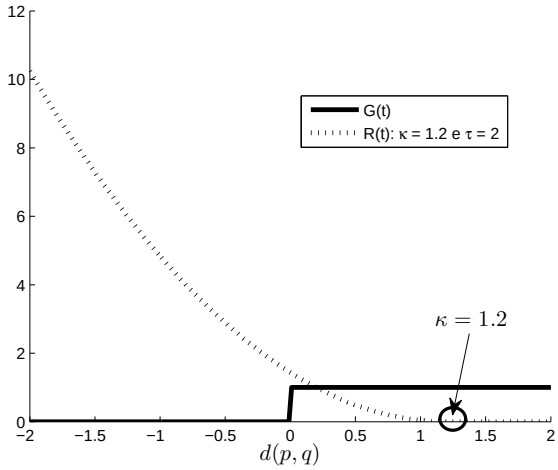


Figura 4.10: Curvas $G(t)$ e $R(t)$ (com $\kappa = 1.2$ e $\tau = 2$) em função da diferença $d(p,q)$, para o intervalo $-2 \leq d(p,q) \leq 2$.

A partir da Equação (4.43), uma aproximação diferenciável para o funcional $AUC(\hat{f})$ é definida, conforme Equação (4.44) a seguir (Yan *et al.*, 2003),

$$\widehat{AUC}(\mathbf{w}) = \frac{1}{N_1 N_2} \left(\sum_{p=1}^{N_1} \sum_{q=1}^{N_2} R\left(d(p,q)\right) \right) \tag{4.44}$$

com $\mathbf{w}$ representando o vetor de pesos da rede MLP. É importante mencionar que na substituição da função degrau $G(t)$ por $R(t)$, a aproximação obtida (4.44) deve ser minimizada ao invés de maximizada.

Note que minimizar o funcional $\widehat{AUC}(\mathbf{w})$ implica em buscar soluções cujos valores $d(p, q)$ sejam maiores ou iguais a κ para todos os pares de exemplos.

Como mencionado anteriormente, Yan *et al.* (2003) sugerem que κ assuma valores maiores que 0, o que supostamente assegura que os *scores* obtidos para os exemplos positivos sejam maiores que os negativos. Sua estratégia tem por princípio obter um bom *ranking*, maximizando a AUC sem a necessidade de se considerar um *threshold* de decisão.

Com o objetivo de garantir não somente a qualidade do *ranking*, mas também que os *scores* estimados para as diferentes classes fiquem de lados opostos em relação ao *threshold* padrão ($\theta_{pd} = (\hat{f}_{max} + \hat{f}_{min})/2$), uma nova faixa de valores para κ é aqui sugerida: $(d_{max}/2) < \kappa \leq d_{max}$.

Considere novamente, por exemplo, uma rede MLP com saídas contínuas no intervalo $-1 \leq \hat{f}(i) \leq 1$. Nesse caso, κ deve ser maior que 1, na tentativa de que a solução encontrada na otimização da $\widehat{AUC}(\mathbf{w})$ produza $\hat{f}(p) \geq \theta_{pd}$ e $\hat{f}(q) < \theta_{pd}$, com $\theta_{pd} = 0.0$. Ao mesmo tempo, θ_{pd} pode ser usado para estimar métricas extraídas da matriz de confusão como, por exemplo, TPr (sensibilidade) e TNr (especificidade).

O parâmetro τ influencia somente a inclinação da função $R(t)$. Em testes empíricos, foi observado que os melhores resultados foram obtidos para $\tau = 2, 3$ e, $1.2 \leq \kappa \leq 1.5$. Para valores elevados de τ, problemas numéricos no processo de otimização foram notados para alguns conjuntos de dados.

4.3.4 Formulação do problema de aprendizado

Formalmente, o seguinte problema de otimização deve ser resolvido,

$$\mathbf{w}^* = \arg_{\mathbf{w}} \min \ \widehat{AUC}(\mathbf{w}) \tag{4.45}$$

O objetivo do aprendizado é obter o vetor de pesos ótimo, $\mathbf{w}^*$, que minimiza a função custo $\widehat{AUC}(\mathbf{w})$ e, consequentemente, maximiza a curva ROC.

4.3.5 Vetor gradiente

Seja $\mathbf{g}(\mathbf{w})$ o vetor gradiente da função custo $\widehat{AUC}(\mathbf{w})$ em relação ao vetor de pesos corrente $\mathbf{w}$. Cada componente do vetor $\mathbf{g}(\mathbf{w})$ é dado pela derivada parcial de $\widehat{AUC}(\mathbf{w})$ em relação a um peso arbitrário da rede w_l, conforme descrito pela Equação (4.46) a seguir,

$$\frac{\partial \widehat{AUC}}{\partial w_l} = \frac{1}{N_1 N_2} \sum_{p=1}^{N_1} \sum_{q=1}^{N_2} \frac{\partial R\left(d(p,q)\right)}{\partial w_l} \tag{4.46}$$

onde $\frac{\partial R(d(p,q))}{\partial w_l}$ corresponde ao escalar gradiente devido à apresentação do par de exemplos (p, q). Para um dado peso w_s da camada de saída, esse termo é obtido pela Equação (4.47) a seguir,

$$\frac{\partial R\left(d(p,q)\right)}{\partial w_s} = \tau \left(-\hat{f}(p) + \hat{f}(q) - \kappa\right)^{\tau-1} \left[-\phi'\left(v(p)\right) z_s(p) + \phi'\left(v(q)\right) z_s(q)\right] \tag{4.47}$$

Similarmente, para um peso w_{sr} da camada escondida, o escalar gradiente devido à apresentação do par de exemplos (p, q), é dado por,

$$\begin{aligned} \frac{\partial R\left(d(p,q)\right)}{\partial w_{sr}} \quad = \quad & \tau \left(-\hat{f}(p) + \hat{f}(q) - \kappa\right)^{\tau-1} \left[-\phi'\left(v(p)\right) w_s \, \phi'\left(u_s(p)\right) x_r(p)\right. \\ & \left. + \phi'\left(v(q)\right) w_s \, \phi'\left(u_s(q)\right) x_r(q)\right] \end{aligned} \tag{4.48}$$

4.3.6 Atualização dos pesos

A regra de aprendizado é baseada no método do gradiente descendente (Luenberger, 1984). Os pesos da rede são inicializados com valores aleatórios e atualizados, a cada iteração (época), na direção oposta do vetor gradiente, conforme as Equações (4.49) e (4.50) a seguir,

$$\Delta \mathbf{w} = -\eta \, \mathbf{g}(\mathbf{w}) \tag{4.49}$$

$$\mathbf{w}_{new} = \mathbf{w}_{old} + (1 - \rho) \, \Delta \mathbf{w}_{old} + \rho \, \Delta \mathbf{w}_{old-1} \tag{4.50}$$

onde η é uma constante positiva (taxa de aprendizado) que indica o tamanho do termo de atualização ($\Delta\mathbf{w}$) aplicado a cada época sobre o vetor de pesos ($\mathbf{w}$). O termo de *momentum*, $0 \leq \rho \leq 1$, é usado para acelerar a velocidade de convergência do método, especialmente em regiões onde a função custo apresenta *plateaus*, e evitar que o mesmo alcance mínimos locais rasos (Haykin, 1994).

4.4 Extensões multiobjetivo para os algoritmos propostos

De acordo com a teoria do Aprendizado Estatístico (Vapnik, 1995, 1999), uma maneira de se obter modelos eficientes (que generalizam bem) é controlar a complexidade da classe de hipóteses fornecida pela máquina de aprendizado. Essa propriedade, no entanto, não foi contemplada nas formulações originais propostas para os algoritmos WEMLP e AUCMLP. Seus funcionais custo consideram somente as perdas associadas aos padrões treinamento (*risco* empírico), o que, em um senso teórico, poderia levar à seleção de modelos sub-ótimos.

Nessa seção, as formulações de WEMLP e AUCMLP são estendidas com o objetivo de se incorporar uma técnica efetiva para controle de complexidade dos modelos. Isso é feito com base no aprendizado multiobjetivo (MOBJ) de MLPs (Teixeira *et al.*, 2000), o qual pode ser considerado como uma implementação do princípio indutivo de minimização estrutural do *risco* (SRM), descrito em Vapnik (1995, 1999).

4.4.1 Controlando a complexidade com o aprendizado MOBJ

O princípio indutivo de minimização estrutural do *risco* (SRM) define um limite superior para o funcional *risco* esperado ($R[f]$) de uma máquina de aprendizado (Vapnik, 1998, 1995). Com probabilidade $1-\delta$ sobre N padrões de treinamento, o *risco* esperado é limitado pela função $\zeta(\cdot)$, que é uma função inversa de N e, uma função direta dos funcionais *risco* empírico (R_{emp}) e complexidade (Ω), conforme Equação (4.51),

$$R[f] \leq \zeta\left(N, R_{emp}, \Omega\right) \tag{4.51}$$

O funcional R_{emp} corresponde a uma estimativa de $R[f]$ em relação aos dados de treinamento e, Ω é uma medida da complexidade (capacidade) da classe de funções $f : \mathbb{R}^n \rightarrow \{0, 1\}$ fornecida pela máquina de aprendizado. Note a partir de (4.51) que ao se decrementar $\zeta(\cdot)$, decrementa-se também o limite superior do *risco* esperado. Isso pode ser feito com um aumento do número de padrões N e/ou pela minimização simultânea de dois objetivos conflitantes: R_{emp} e Ω (Vapnik, 1998, 1995).

O princípio SRM foi originalmente formalizado com o *risco* esperado representando a probabilidade do erro global de classificação (ou erro de generalização). Ele nos mostra que a obtenção de soluções eficientes para o problema do aprendizado não depende somente do erro sobre o conjunto de treinamento (R_{emp}) mas também, de um controle efetivo na complexidade dos modelos (Ω) (Vapnik, 1998, 1995).

A formulação MOBJ para o aprendizado de MLPs, proposta em Teixeira *et al.* (2000), foi concebida segundo o princípio SRM. Ela faz o controle da complexidade dos modelos minimizando, ao mesmo tempo, a magnitude dos pesos da rede (Ω) e o erro global de treinamento (R_{emp}). Dado o conjunto de dados $T = \{(\mathbf{x}(i), y(i)) \mid i = 1 \ldots N\}$, o problema de aprendizado MOBJ é,

$$\min \begin{cases} J_1 = \frac{1}{2}\sum_{i=1}^{N}\left(y(i) - \hat{f}(\mathbf{x}(i))\right)^2 \\ J_2 = \|\mathbf{w}\| \end{cases} \tag{4.52}$$

onde $\mathbf{w}$ é o vetor de pesos da rede e, $\|\cdot\|$ é o operador que fornece a norma euclidiana de um vetor. Embora existam na literatura outras formas de se medir a complexidade das funções fornecidas por uma MLP, tais como a magnitude da transformada de *Fourier* da saída da rede (Barron, 1993) e o número de parâmetros livres ou derivadas contínuas (Girosi *et al.*, 1995), a escolha da norma dos pesos como medida de complexidade pode ser justificada pelo estudo de Bartlett (1997), onde foi mostrado que redes com um elevado número de pesos se comportam como sistemas menos complexos quando a magnitude dos mesmos é reduzida (ou restringida).

A proposta de incorporação de uma técnica de controle de complexidade aos métodos propostos na tese pode ser vista como uma extensão da metologia MOBJ. O procedimento adotado consistiu em modificar o problema de aprendizado, descrito em (4.52), para que o funcional J_1 (R_{emp}), fosse substituído pelas correspondentes funções custo de WEMLP e AUCMLP. A norma dos pesos $\|\mathbf{w}\|$ segue como um segundo objetivo (J_2) a ser minimizado, com o intuito de limitar a flexibilidade das funções de decisão.

As novas formulações, denominadas WEMOBJ e AUCMOBJ, foram implementadas através do método ε-restrito, conforme descrito em Teixeira *et al.* (2000). Esse método permite reescrever o problema de aprendizado biobjetivo, descrito em (4.52), como um problema restrito de otimização mono-objetivo. Para tanto, o funcional J_1 (R_{emp}) é otimizado, enquanto a norma dos pesos (J_2) torna-se uma restrição limitada por ε. Um problema restrito de otimização é então resolvido[1] para cada diferente valor de ε. A formulação de WEMOBJ com base na técnica ε-restrito é descrita como segue,

$$\begin{aligned} \min \quad & J_1 = \lambda \sum_{p=1}^{N_1} e^2(p) + (1-\lambda) \sum_{q=1}^{N_2} e^2(q) \\ s.a. \quad & J_2 = \|\mathbf{w}\| \leq \varepsilon \end{aligned} \tag{4.53}$$

onde $e(p)$ e $e(q)$ representam os erros obtidos na saída da rede para padrões arbitrários das classes positiva e negativa, respectivamente. A formulação de AUCMOBJ surge intrinsecamente de (4.53), com a simples substituição da função custo de WEMLP (J_1) por aquela de AUCMLP, apresentada em (4.44).

A variação paramétrica de ε possibilita a obtenção de um conjunto de soluções eficientes (estimativa do conjunto pareto-ótimo), fornecendo os melhores compromissos entre os funcionais J_1 e J_2. O número de soluções geradas é determinado pelos seguintes parâmetros que devem ser fornecidos a priori: valor inicial (ε_0) e máximo (ε_{max}) para a norma dos pesos e também, seu intervalo de variação (δ_ε). Em relação aos parâmetros da rede MLP, Teixeira *et al.* (2000) sugerem um

[1] Nas implementações de WEMOBJ e AUCMOBJ, o algoritmo elipsoidal (Bland *et al.*, 1980) foi usado para resolver cada problema restrito de otimização mono-objetivo.

valor elevado para o número de neurônios (h) da camada escondida e, que o vetor inicial de pesos ($\mathbf{w}_0$) seja gerado de forma aleatória segundo uma distribuição gaussiana com média zero e variância muito pequena.

Uma vez determinado o conjunto de soluções eficientes, a próxima etapa do aprendizado MOBJ consiste na escolha da solução final, a qual deve apresentar a complexidade adequada para a tarefa de aprendizado em questão. Em outras palavras, espera-se que o modelo selecionado constitua uma boa aproximação para o mínimo absoluto do *risco* esperado. Para as formulações WEMOBJ e AUCMOBJ, a escolha do modelo final foi feita através do *decisor* por validação, proposto em Teixeira *et al.* (2000). Nesse *decisor*, um conjunto de validação é apresentado a todas os modelos pertencentes ao conjunto pareto-ótimo. O modelo que apresentar o menor valor segundo o critério de WEMLP (ou AUCMLP) é escolhido como solução final. A regra de decisão é dada como segue,

$$\mathbf{w}^* = \underset{\mathbf{w}\in\mathcal{W}}{\arg\min}\, J_1^v \tag{4.54}$$

onde J_1^v é o funcional custo avaliado em relação ao conjunto de validação, e $\mathcal{W}$ representa o conjunto de soluções eficientes.

4.5 Conclusões do capítulo

Novos algoritmos de aprendizado para lidar com o problema de classes desbalanceadas no contexto de redes MLP foram apresentados: WEMLP (Ponderação de Erros) e AUCMLP (Otimização da AUC). Ambos possuem critérios que consideram os desempenhos individuais por classe, visando melhorar o equilíbrio entre as acurácias, assim como superar o viés imposto pelo grupo dominante na presença de dados complexos e desbalanceados.

WEMLP é baseado na otimização de uma função objetivo conjunta que usa um parâmetro de custo (λ) para penalizar de forma distinta as contribuições dos erros de cada classe. Uma análise formal sobre o papel de λ na função custo proposta mostrou que superfícies de decisão equilibradas no espaço de características podem ser obtidas através da incorporação de uma simples informação a priori, ou seja, através do ajuste de λ segundo a proporção de exemplos negativos no

conjunto de treinamento ($N_2/(N_1 + N_2)$). Além disso, por assinalar diferentes perdas (custos) aos erros de classificação, λ permite alcançar soluções distintas no espaço ROC.

É importante ressaltar que a estratégia adotada por WEMLP para a obtenção de modelos equidistantes das classes é válida para dados separáveis, bem como não linearmente separáveis. Essa propriedade é garantida pelo simples fato das relações estabelecidas na Seção 4.2.3 considerarem todos os padrões de entrada, incluindo aqueles que se encontram do lado oposto do hiperplano em relação à sua classe de origem.

Na segunda parte do capítulo, os aspectos que motivam o desenvolvimento de um método para maximizar diretamente a *Area Under the ROC Curve* (*AUC*) foram discutidos, a partir de uma análise das expressões e observações descritas em Cortes & Mohri (2004). Como consequência, os fundamentos teóricos de AUCMLP foram apresentados, a partir da otimização de uma função custo que corresponde a uma aproximação diferenciável da estatística de *Wilcoxon-Mann-Whitney*. Uma adaptação na faixa de valores do parâmetro κ foi sugerida para essa função custo. Além de priorizar *ranking*, a adaptação é uma tentativa de fazer com que os *scores* estimados para as diferentes classes fiquem de lados opostos em relação ao *threshold* padrão.

Na parte final do capítulo, extensões bi-objetivo (MOBJ) para as formulações de WEMLP e AUCMLP foram apresentadas com o propósito de se incorporar uma estratégia de controle de complexidade a seus processos de seleção de modelos. O procedimento adotado consistiu em reformular o problema original do aprendizado para que a norma dos pesos da rede (medida de complexidade) fosse minimizada de forma simultânea aos correspondentes funcionais custo propostos. Essa abordagem foi motivada, principalmente, pelo sucesso empírico obtido com a metodologia MOBJ no escopo de MLPs tradicionais (baseadas no erro global).

Capítulo 5

Experimentos e Resultados

Nesse capítulo, a eficiência dos algoritmos propostos na tese, WEMLP e AUCMLP, é colocada à prova em um estudo experimental realizado sobre bases de dados reais com diferentes razões de desbalanceamento. Os desempenhos de WEMLP e AUCMLP são comparados a métodos conhecidos na literatura de classes desbalanceadas. Foram selecionados métodos das categorias de *pré-processamento de dados* e *adaptação em algoritmos de aprendizado*, nas linhas de *Boosting* e modificação de função custo. Testes estatísticos de significância são empregados para a análise dos resultados e, posterior discussão sobre as propriedades dos algoritmos usados no estudo.

5.1 Metodologia

Os parágrafos a seguir descrevem a metodologia geral adotada na condução dos experimentos. Metodologia similar foi usada em Wu & Chang (2005) e Tang *et al.* (2009).

Foram selecionadas, ao todo, 17 bases de dados do repositório UCI (Asuncion & Newman, 2007) com diferentes graus de desbalanceamento. Tais bases encontram-se listadas na Tabela 5.1 juntamente com suas características: número de atributos (#atributos), número de exemplos positivos (N_1), número de exemplos negativos (N_2), razão de desbalanceamento ($N_1/(N_1 + N_2)$) e porcentagem de dados sintéticos (%sintéticos) a serem gerados por um método de so-

breamostragem particular. Essa última característica será detalhada adiante na Seção 5.2.

Tabela 5.1: Características das bases de dados usadas nos experimentos: número de atributos (#atributos), número de exemplos positivos (N_1), número de exemplos negativos (N_2), razão de desbalanceamento ($N_1/(N_1+N_2)$) e porcentagem de dados sintéticos (%sintéticos) gerados por sobreamostragem.

Base de dados	*alias*	#atributos	N_1	N_2	$N_1/(N_1+N_2)$	%sintéticos
Ionosphere	iono	34	126	225	0.359	100%
Pima Indians Diabetes	pid	08	268	500	0.349	100%
German Credit	gmn	24	300	700	0.300	100%
WP Breast Cancer	wpbc	33	47	151	0.237	200%
Vehicle (4 vs. all)	veh	18	199	647	0.235	200%
SPECTF Heart	hrt	44	55	212	0.206	300%
Segmentation (1 vs. all)	seg	19	30	180	0.143	500%
Glass (7 vs. all)	gls7	10	29	185	0.136	500%
Euthyroid (1 vs. all)	euth	24	238	1762	0.119	600%
Satimage (4 vs. all)	sat	36	626	5809	0.097	800%
Vowel (1 vs. all)	vow	10	90	900	0.091	900%
Abalone (18 vs. 9)	a18-9	08	42	689	0.057	1000%
Glass (6 vs. all)	gls6	10	9	205	0.042	2000%
Yeast (9 vs. 1)	y9-1	08	20	463	0.041	2000%
Car (3 vs. all)	car	06	69	1659	0.040	2500%
Yeast (5 vs. all)	y5	08	51	1433	0.034	2500%
Abalone (19 vs. all)	a19	08	32	4145	0.008	10000%

Todas as bases de dados passaram pelos seguintes estágios de pré-processamento: atributos categóricos foram expandidos para os correspondentes vetores binários e em seguida, cada atributo (numérico ou binário) foi normalizado para o intervalo $[-1, 1]$.

Bases de dados contendo $c > 2$ classes foram reduzidas à classificação binária usando um dos procedimentos a seguir: (i) escolha de um dos rótulos para representar a classe positiva (ou de interesse) e união dos demais rótulos para compor a classe negativa. Por exemplo, *Vehicle* 4 ("van") vs. all ("opel", "saas" e "bus"); (ii) seleção de apenas 2 rótulos entre todos os c rótulos. Para a base de dados *Yeast*, por exemplo, as observações com rótulos 9 ("pox") e 1 ("cyt") foram esco-

lhidas para representarem as classes positiva e negativa, respectivamente. Ambos os procedimentos foram aplicados seguindo sugestões da literatura (Chen *et al.*, 2010; Tang *et al.*, 2009; Wu & Chang, 2005).

Visando a obter representatividade nos resultados dos algoritmos testados, foram geradas 20 permutações aleatórias para cada base de dados. Cada permutação foi então dividida em subconjuntos de treinamento (2/3) e teste (1/3) de uma maneira estratificada, garantindo em cada um deles a mesma razão de desbalanceamento da base de dados original. Observe que com esse procedimento, foram produzidos 20 diferentes casos de treinamento/teste para cada base de dados. Dessa forma, para uma algoritmo particular, o desempenho médio e desvio-padrão foram calculados sobre 20 execuções (casos de treinamento/teste), com métricas comumente usadas na avaliação de problemas desbalanceados. São elas:

- *G-mean* $= \sqrt{TPr \cdot TNr}$, onde TPr e TNr representam as taxas de verdadeiros positivos (sensibilidade) e verdadeiros negativos (especificidade), respectivamente (Kubat *et al.*, 1998); valores elevados de *G-mean* refletem taxas de acerto elevadas e equilibradas para ambas as classes.

- *AUC* (*Area Under the ROC Curve*), obtida através do algoritmo descrito em Fawcett (2006), que soma sucessivas áreas de trapézios formados na construção da Curva ROC.

- Curvas ROC médias estimadas com a técnica *threshold averaging*, também apresentada em Fawcett (2006).

5.2 Experimento 1

O objetivo do primeiro experimento foi testar a eficiência dos algoritmos propostos, WEMLP e AUCMLP, comparando-os com métodos conhecidos na literatura que podem ser combinados com redes MLP, ou seja, que podem usar MLP como classificador base. São eles: *Smote + Tomek-Links* (SMTTL) (Batista *et al.*, 2004), *Weighted Wilson's Editing* (WWE) (Barandela *et al.*, 2004) e *RAMOBoost* (RBoost) (Chen *et al.*, 2010). Além disso, uma rede MLP pura (MLP),

isto é, sem qualquer estratégia para lidar com dados desbalanceados, foi também testada dentro das mesmas condições dos algoritmos acima mencionados[1].

O método SMTTL combina uma estratégia de geração de dados sintéticos (*Smote*) com uma técnica de limpeza (*Tomek-Links*) para a obtenção de distribuições bem definidas no espaço de entrada (Batista *et al.*, 2004). WWE intensifica a regra de edição de Wilson (*Wilson's Editing*) eliminando um número maior de exemplos da classe majoritária cujo rótulo difere da maioria de seus k vizinhos mais próximos (Barandela *et al.*, 2004). RAMOBoost, por sua vez, mescla uma estratégia inteligente de geração de dados sintéticos (RAMO) com um sistema de aprendizado *ensemble* (AdaBoost.M2) (Chen *et al.*, 2010).

A rede MLP usada como classificador base possui topologia $n : h : 1$ e função de ativação do tipo tangente hiperbólica em todas as unidades. Com exceção de WEMLP e AUCMLP que modificam a função custo original, todos os outros algoritmos foram associados a MLPs baseadas na minimização do erro global. AUCMLP possui regra de aprendizado baseada no *Gradiente com termo de Momentum*, com parâmetros $\rho = 0.9$ e $\eta = 0.1$ (vide Seção 4.3.6). Os demais algoritmos, incluindo WEMLP, usam *Levenberg-Marquadt* para atualização dos pesos, com parâmetros $\mu = 0.1$ e $\beta = 10$ (vide Seção 4.2.2).

Na execução de um algoritmo particular sobre um dado caso de treinamento/teste, o procedimento de busca em *grid* com *stratified* 7-*fold crossvalidation* (Van Gestel *et al.*, 2004) foi empregado (sobre o subconjunto de treinamento) para obtenção do número ótimo de neurônios h^* na camada escondida da rede. O conjunto inicial de parâmetros candidatos foi $h_0 = \{1 : 3 : 13\}$. A busca em *grid* contou com apenas um refinamento ao redor do parâmetro ótimo h_0^* selecionado na iteração 0. O conjunto de parâmetros candidatos na iteração 1 foi $h_1 = \{h_0^* - 2 : 1 : h_0^* + 2\}$. Para os algoritmos SMTTL, WWE e RBoost a busca em *grid* foi aplicada após os dados de treinamento terem sido modificados por suas respectivas estratégias de sobreamostragem/subamostragem.

[1]Este procedimento é padrão na literatura de aprendizado com classes desbalanceadas. Quando uma nova solução é proposta, seu desempenho é então colocado à prova na comparação com outros métodos, usando o mesmo classificador base. Além disso, é comum considerar o desempenho obtido com o classificador base puro para servir como referência (*baseline*) nas comparações (Khoshgoftaar *et al.*, 2010).

Os parâmetros dos algoritmos foram configurados seguindo sugestões da literatura. Para RBoost, os números de vizinhos mais próximos usados no ajuste das probabilidades de amostragem dos exemplos minoritários e na geração de dados sintéticos foram configurados como 5 e 10, respectivamente; o número de iterações de *Boosting* e o coeficiente de escalonamento foram ajustados para 20 e 0.3, respectivamente (Chen *et al.*, 2010). Para SMTTL, o número de vizinhos mais próximos foi configurado como 5 (Batista *et al.*, 2004), enquanto que o valor 3 foi escolhido para WWE (Barandela *et al.*, 2004; Khoshgoftaar *et al.*, 2010). A última coluna da Tabela 5.1 mostra as porcentagens de exemplos sintéticos gerados para cada base de dados na execução dos algoritmos SMTTL e RBoost. Para a base de dados *Vehicle*, por exemplo, o número de exemplos sintéticos gerados a cada execução é igual a 200% do número original de exemplos positivos (N_1). Os mecanismos para estimação de vizinhos mais próximos e/ou síntese de dados dos algoritmos SMTTL, WWE e RBoost foram estendidos para a manipulação de atributos categóricos conforme sugerido em Chawla *et al.* (2002). Para cada base de dados, o parâmetro λ do algoritmo WEMLP foi configurado como $\frac{N_2}{N_1+N_2}$ e mantido constante em todas as execuções. O mesmo foi feito para os parâmetros $\kappa = 1.4$ e $\tau = 2$ que regulam a função custo do método AUCMLP.

5.2.1 Resultados

As Tabelas 5.2 e 5.3 listam, respectivamente, os valores de *G-mean* e AUC obtidos pelos algoritmos MLP, SMTTL, WWE, RBoost, WEMLP e AUCMLP sobre as 17 bases de dados. As médias e desvios-padrão foram calculados sobre 20 diferentes casos de teste, conforme descrito na Seção 5.1. Os melhores valores encontram-se em negrito. As correspondentes curvas ROC médias estimadas pelos algoritmos sobre para as 10 bases de dados mais desbalanceadas (gls7, euth, sat, vow, a18-9, gls6, y9-1, car, y5, a19) encontram-se disponíveis no Apêndice C.

Observe a partir das Tabelas 5.2 e 5.3 que, na maioria dos casos (base de dados), considerando a pequena diferença numérica entre as médias e a magnitude dos desvios, fica difícil distinguir os algoritmos testados. A aplicação de um teste estatístico faz-se então necessária para se obter conclusões sobre os desempenhos dos mesmos.

Tabela 5.2: Comparação entre os valores de *G-mean* (em %) obtidos pelos algoritmos MLP, SMTTL, WWE, RBoost, WEMLP e AUCMLP sobre as 17 base de dados. Os melhores valores encontram-se em negrito.

Base de dados	MLP	SMTTL	WWE	RBoost	WEMLP	AUCMLP
iono	84.33 ± 4.18	85.20 ± 5.31	85.21 ± 4.60	$\mathbf{86.39 \pm 4.03}$	85.60 ± 5.49	85.15 ± 4.65
pid	69.18 ± 3.86	66.81 ± 3.58	72.61 ± 3.79	72.12 ± 2.96	$\mathbf{72.82 \pm 3.02}$	72.63 ± 2.40
gmn	57.60 ± 14.63	61.05 ± 2.06	67.42 ± 3.76	65.86 ± 2.76	$\mathbf{70.00 \pm 1.68}$	67.97 ± 1.76
wpbc	62.07 ± 16.85	61.33 ± 7.96	61.72 ± 8.38	$\mathbf{66.63 \pm 8.55}$	63.47 ± 9.29	63.39 ± 7.77
veh	95.41 ± 1.97	95.91 ± 1.51	96.00 ± 1.81	$\mathbf{97.07 \pm 1.14}$	96.84 ± 1.26	96.76 ± 1.06
hrt	57.67 ± 15.45	61.21 ± 10.60	$\mathbf{67.70 \pm 5.51}$	64.93 ± 6.60	66.22 ± 5.97	65.50 ± 6.07
seg	94.54 ± 4.37	95.96 ± 3.71	95.94 ± 2.68	96.53 ± 2.58	96.45 ± 2.63	$\mathbf{96.98 \pm 2.51}$
gls7	89.69 ± 11.29	89.49 ± 8.01	92.36 ± 5.97	92.25 ± 6.59	$\mathbf{93.34 \pm 4.91}$	92.77 ± 5.18
euth	87.39 ± 3.79	87.29 ± 2.60	88.64 ± 2.35	87.55 ± 1.98	$\mathbf{90.40 \pm 1.96}$	89.34 ± 1.86
sat	69.90 ± 3.68	76.86 ± 2.27	76.88 ± 2.97	77.65 ± 2.31	84.82 ± 1.74	$\mathbf{85.77 \pm 0.75}$
vow	98.14 ± 1.71	98.36 ± 2.72	97.83 ± 3.96	$\mathbf{99.90 \pm 0.38}$	99.01 ± 1.56	98.37 ± 2.74
a18-9	70.36 ± 6.85	67.08 ± 8.95	69.51 ± 10.47	70.39 ± 12.11	82.40 ± 5.36	$\mathbf{83.04 \pm 6.15}$
gls6	85.29 ± 14.46	78.13 ± 11.86	81.37 ± 23.15	81.95 ± 13.27	$\mathbf{95.56 \pm 3.90}$	87.26 ± 13.27
y9-1	63.27 ± 20.05	66.93 ± 20.39	71.76 ± 14.19	$\mathbf{73.26 \pm 12.36}$	70.54 ± 11.82	70.53 ± 11.56
car	87.39 ± 21.20	96.02 ± 4.13	97.41 ± 0.65	95.96 ± 5.00	$\mathbf{98.31 \pm 0.84}$	97.56 ± 0.97
y5	42.69 ± 16.48	61.49 ± 9.75	67.80 ± 9.34	62.76 ± 7.97	78.98 ± 6.19	$\mathbf{79.63 \pm 5.23}$
a19	0.00 ± 0.00	21.79 ± 17.22	0.00 ± 0.00	68.14 ± 7.06	$\mathbf{76.12 \pm 7.96}$	75.09 ± 6.90

Tabela 5.3: Comparação entre os valores de AUC (em %) obtidos pelos algoritmos MLP, SMTTL, WWE, RBoost, WEMLP e AUCMLP sobre as 17 base de dados. Os melhores valores encontram-se em negrito.

Base de dados	MLP	SMTTL	WWE	RBoost	WEMLP	AUCMLP
iono	90.29 ± 4.75	91.36 ± 4.85	91.14 ± 4.51	$\mathbf{91.47 \pm 3.71}$	90.67 ± 4.81	89.76 ± 4.05
pid	81.46 ± 3.52	74.49 ± 3.93	79.06 ± 5.16	79.13 ± 2.39	$\mathbf{82.15 \pm 2.56}$	81.73 ± 1.84
gmn	72.97 ± 3.61	67.85 ± 2.24	73.63 ± 3.59	73.91 ± 1.94	$\mathbf{75.68 \pm 2.10}$	74.51 ± 2.10
wpbc	$\mathbf{73.84 \pm 6.40}$	71.34 ± 7.15	68.38 ± 6.25	73.52 ± 7.61	72.51 ± 6.50	73.48 ± 6.25
veh	99.17 ± 0.69	$\mathbf{99.46 \pm 0.33}$	98.99 ± 0.68	99.39 ± 1.11	99.28 ± 0.42	99.29 ± 0.36
hrt	73.60 ± 6.06	75.02 ± 4.85	76.71 ± 5.05	76.55 ± 4.52	77.70 ± 3.36	$\mathbf{79.05 \pm 3.32}$
seg	98.15 ± 2.19	98.52 ± 2.21	99.14 ± 1.23	98.42 ± 2.14	99.34 ± 0.77	$\mathbf{99.70 \pm 0.46}$
gls7	92.43 ± 12.12	96.74 ± 3.46	$\mathbf{98.14 \pm 1.79}$	97.09 ± 4.52	97.78 ± 3.09	97.75 ± 3.37
euth	93.04 ± 2.92	94.11 ± 1.51	93.77 ± 1.93	94.60 ± 1.72	$\mathbf{95.98 \pm 1.14}$	95.27 ± 1.19
sat	87.07 ± 2.63	90.90 ± 1.13	91.22 ± 1.40	91.07 ± 2.18	92.57 ± 1.25	$\mathbf{93.58 \pm 0.35}$
vow	99.86 ± 0.25	99.84 ± 0.51	99.59 ± 1.24	$\mathbf{99.99 \pm 0.04}$	99.84 ± 0.22	99.84 ± 0.44
a18-9	81.51 ± 6.88	84.70 ± 5.42	84.15 ± 6.13	85.28 ± 9.09	$\mathbf{92.47 \pm 3.17}$	92.35 ± 3.69
gls6	97.35 ± 5.08	97.18 ± 1.26	96.19 ± 5.91	$\mathbf{98.38 \pm 4.69}$	96.27 ± 5.25	98.26 ± 2.87
y9-1	76.87 ± 9.81	75.16 ± 10.31	77.59 ± 9.65	$\mathbf{80.10 \pm 7.94}$	77.94 ± 11.10	75.67 ± 10.68
car	96.73 ± 9.47	99.60 ± 0.48	98.81 ± 0.58	$\mathbf{99.75 \pm 0.36}$	99.33 ± 0.30	98.59 ± 0.94
y5	82.41 ± 5.72	82.89 ± 5.43	82.42 ± 7.77	84.00 ± 6.07	87.33 ± 3.99	$\mathbf{88.00 \pm 4.17}$
a19	68.24 ± 12.71	62.91 ± 12.23	67.44 ± 9.67	81.93 ± 7.15	$\mathbf{85.79 \pm 3.57}$	84.02 ± 5.23

5.2.2 Teste de significância

Estudo recente de Demsar (2006) analisou teórica e empiricamente inúmeros testes estatísticos para comparação de dois ou mais classificadores sobre múltiplas bases de dados. Com base nas propriedades estatísticas desses testes e no conhecimento sobre os dados comumente usados em problemas de aprendizado de máquina, o estudo concluiu que testes não-paramétricos de *ranking*, tais como *Wilcoxon*, para comparação de dois classificadores, e *Friedman*, para comparação de múltiplos classificadores, devem ser preferidos aos conhecidos testes paramétricos *Paired t-test* e *ANOVA*. Segundo Demsar (2006), os testes não-paramétricos são mais seguros que seus correlatos paramétricos, desde que eles não assumem distribuições normais ou homogeneidade de variância para os *scores* obtidos sobre inúmeras bases de dados. Isso faz com que eles possam ser aplicados sobre diferentes métricas de avaliação, incluindo tempos computacionais. Resultados empíricos também sugeriram que os testes não-paramétricos são mais fortes, apresentando maior probabilidade de rejeição da hipótese nula.

Seguindo recomendação de Demsar (2006), o teste não-paramétrico de *Friedman* (Friedman, 1937, 1940) foi selecionado para verificar se os algoritmos MLP, SMTTL, WWE, RBoost, WEMLP e AUCMLP são significativamente diferentes. Foram considerados para aplicação do teste os valores médios obtidos com as métricas *G-mean* e *AUC*.

No teste de *Friedman*, dados L algoritmos avaliados sobre M bases de dados, com seus *ranks*[1] médios R_t, $t = 1, \ldots, L$, assume-se a hipótese-nula (H_0) de que todos os algoritmos são equivalentes, tal que seus *ranks* médios devem ser iguais. Nesse caso, a estatística,

$$F_F = \frac{(M-1)\chi_F^2}{M(L-1) - \chi_F^2} \tag{5.1}$$

é distribuída segundo a distribuição F com $L-1$ e $(L-1)(M-1)$ graus de liberdade, onde,

[1] O teste de *Friedman* "ranqueia" os algoritmos para cada base de dados separadamente. O algoritmo de melhor *score* recebe o *rank* 1, o segundo melhor o *rank* 2 e, assim por diante. Em caso de empates, *ranks* médios são atribuídos.

$$\chi_F^2 = \frac{12M}{L(L+1)}\left(\sum_t R_t^2 - \frac{L(L+1)^2}{4}\right) \tag{5.2}$$

Em nosso experimento $M = 17$ e $L = 6$ e então, 5 e 80 são os graus de liberdade para o numerador e o denominador, respectivamente. De acordo com a tabela de valores críticos[1] da distribuição F, H_0 é rejeitada, a um nível de significância (α) de 5%, quando $F_F > 2.329$.

A Tabela 5.4 lista os *ranks* médios obtidos pelos algoritmos para as métricas *G-mean* e *AUC*. As duas últimas colunas da tabela, mostram as correspondentes estatísticas F_F e p-valor[2] calculadas para o teste de *Friedman*. De acordo com os valores dessas estatísticas, a hipótese nula (H_0) de *ranks* iguais pode ser rejeitada a um nível significância de 5%; o que indica que os algoritmos são significativamente diferentes para ambas as métricas analisadas.

Tabela 5.4: *Ranks* médios obtidos pelos algoritmos MLP, SMTTL, WWE, RBoost, WEMLP e AUCMLP para as métricas *G-mean* e *AUC*. As duas últimas colunas da tabela mostram os correspondente valores das estatísticas F_F e p-valor referentes ao teste de *Friedman*.

	MLP	SMTTL	WWE	RBoost	WEMLP	AUCMLP	F_F	p-valor
G-mean	5.32	5.06	3.74	2.88	1.71	2.29	26.20	3.6×10^{-10}
AUC	4.71	4.35	4.29	2.65	2.41	2.59	7.44	5.7×10^{-5}

Em seguida, conforme sugerido em Demsar (2006), o teste *post-hoc* de *Nemenyi* (Nemenyi, 1963) pode ser aplicado para comparar os algoritmos no formato "um-contra-um". No teste de *Nemenyi*, se os *ranks* médios de dois algoritmos diferem de pelo menos,

$$CD = q_\alpha \sqrt{\frac{L(L+1)}{6M}} \tag{5.3}$$

[1] O valor crítico de $F(5, 80)$ para $\alpha = 0.05$ é 2.329. Veja tabela em Sheskin (2007).

[2] A estatística p-valor para o teste de *Friedman* foi calculada com o auxílio do *software* MatLab (MATLAB, 2010).

a diferença é considerada significativa com nível de confiança de $1-\alpha$. De acordo com a tabela de valores críticos para o teste de *Nemenyi*, disponível em Demsar (2006), a constante $q_{0.05}$ para $L = 6$ é 2.850, tal que CD é 1.828. Esse valor é o mesmo para ambas as métricas, *G-mean* e AUC.

Os resultados do teste *post-hoc* para as métricas *G-mean* e AUC encontram-se visualmente representados nos diagramas de Diferença Crítica (DC) (Demsar, 2006) das Figuras 5.1 e 5.2, respectivamente. No eixo superior de cada diagrama DC, os *ranks* médios dos algoritmos foram marcados em ordem decrescente, com o melhor algoritmo ficando mais à direita. Além disso, os grupos de algoritmos que não são significativamente diferentes (para $\alpha = 0.05$), isto é, cujas diferenças entre *ranks* médios não superam $CD = 1.828$, foram conectados por traços horizontais.

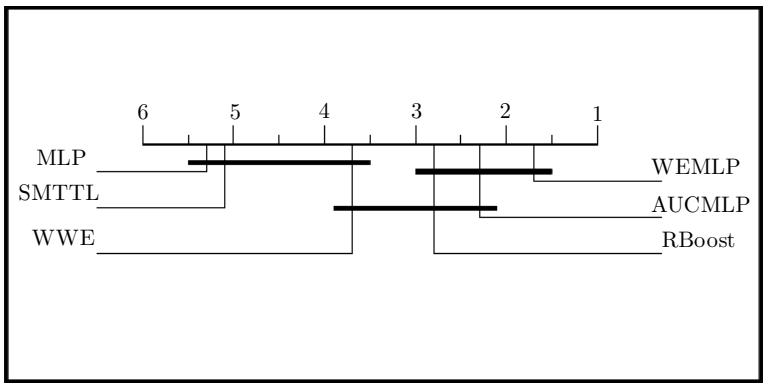


Figura 5.1: Diagrama DC representando os resultados do teste *post-hoc* de *Nemenyi* para a métrica *G-mean*. Os grupos de algoritmos que não são significativamente diferentes (para $\alpha = 0.05$) encontram-se conectados por traços horizontais.

Analisando os resultados do teste *post-hoc* para a métrica *G-mean* (vide Figura 5.1), pode-se concluir que MLP e SMTTL são significativamente piores que RBoost, WEMLP e AUCMLP, que aparentam possuir desempenhos equivalentes. Além disso, os dados não são suficientes para concluir se WWE possui desempenho equivalente à MLP e SMTTL ou à RBoost e AUCMLP. Pode-se afirmar apenas que WWE é significativamente pior que WEMLP.

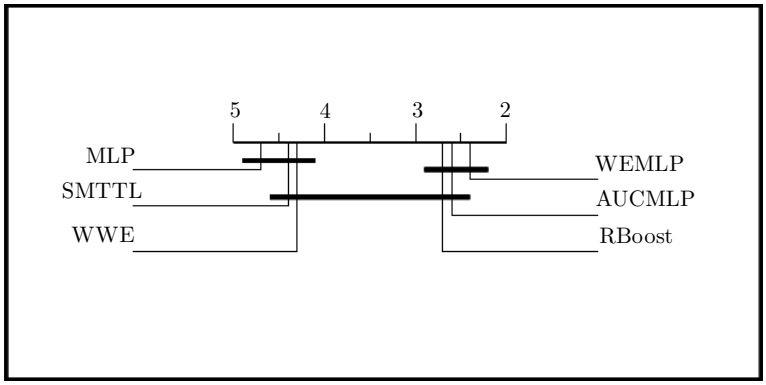


Figura 5.2: Diagrama DC representando os resultados do teste *post-hoc* de *Nemenyi* para a métrica *AUC*. Os grupos de algoritmos que não são significativamente diferentes (para $\alpha = 0.05$) encontram-se conectados por traços horizontais.

Com relação à métrica AUC (vide Figura 5.2), o teste *post-hoc* nos permite concluir que MLP é significativamente pior que RBoost, WEMLP e AUCMLP que, novamente parecem possuir desempenhos iguais. Os dados também não são capazes de indicar se SMTTL e WWE possuem desempenhos equivalentes à MLP pura ou à RBoost e AUCMLP. Cabe ressaltar, no entanto, que as diferenças entre os *ranks* médios de SMTTL e AUCMLP ($4.35 - 2.59 = 1.76$) e, WWE e AUCMLP ($4.29 - 2.59 = 1.70$) são muito próximas do valor crítico $CD = 1.828$ para $\alpha = 0.05$. O mesmo é válido para SMTTL e RBoost ($4.35 - 2.65 = 1.70$) e, WWE e RBoost ($4.29 - 2.65 = 1.64$). Isso sugere que diferenças significativas entre os grupos SMTTL + WWE e RBoost + AUCMLP poderiam ser encontradas para valores maiores de significância do teste *post-hoc*, tal como $\alpha = 0.10$. Por último, pode-se concluir através da Figura 5.2 que ambos, SMTTL e WWE, são significativamente piores que WEMLP.

5.2.3 Discussão

Os testes estatísticos oriundos do Experimento 1 mostram que os algoritmos WEMLP e AUCMLP, propostos na tese, são igualmente eficientes, sendo capazes

de melhorar significativamente o desempenho médio de MLPs puras sobre dados desbalanceados. Os ganhos de desempenho em termos de *G-mean* (Tabela 5.2) revelam que ambos, WEMLP e AUCMLP, podem ser usados para obter taxas de acerto elevadas e equilibradas para ambas as classes. Além disso, eles são capazes de otimizar a Curva ROC, conforme indicado pelos valores de AUC (Tabela 5.3) e, pelas correspondentes curvas ROC médias ilustradas no Apêndice C.

Os resultados dos testes de significância também mostram que WEMLP possui desempenho superior aos tradicionais métodos de reamostragem (SMTTL e WWE) para ambas as métricas analisadas. Embora o mesmo não possa ser afirmado sobre AUCMLP, o teste *post-hoc* para *G-mean* indica que ele é superior a SMTTL a um nível de significância de 5%. Além disso, as diferenças elevadas entre os *ranks* médios de AUCMLP e SMTTL e, AUCMLP e WWE (vide Tabela 5.4) para a métrica AUC, sugerem superioridade de nosso método (AUCMLP) a um nível maior de significância do teste *post-hoc* como, por exemplo, $\alpha = 0.10$. É possível especular também que a superioridade de AUCMLP sobre tais métodos de reamostragem possa ser alcançada através de um aumento do número de bases de dados no experimento.

Adicionalmente, os testes estatísticos apontam que WEMLP e AUCMLP possuem desempenho análogo à RBoost, cuja eficiência sobre inúmeras bases de dados reais e métricas de avaliação foi recentemente demonstrada em Chen *et al.* (2010), também usando MLP como classificador base.

Por último, os resultados numéricos (média/dispersão) das Tabelas 5.2 e 5.3 sugerem que nossos métodos podem ser, em média, estatisticamente significativos quando aplicados às bases de dados com elevadas razões de desbalanceamento, tais como y5 (razão = 0.034) e a19 (razão = 0.008). Particularmente para essas bases (vide Apêndice C), as correspondentes curvas ROC médias de AUCMLP e WEMLP praticamente "dominam" SMTTL, WWE e RBoost sobre todos os possíveis valores de limiar (ou FPr).

5.3 Experimento 2

O objetivo do segundo experimento foi comparar WEMLP e AUCMLP com o conhecido método *Asymmetric Cost Support Vector Machines*[1] (ACSVM) (Joachims, 2002; Lin *et al.*, 2002).

Estudo experimental recente conduzido em Tang *et al.* (2009), avaliou inúmeras estratégias baseadas em SVMs (classificador base) sobre dados reais desbalanceados. Como um dos resultados, o estudo apontou ACSVM como a melhor alternativa entre os métodos testados, quando os conjuntos de dados disponíveis não são muito grandes. Uma outra motivação para comparar o método ACSVM com os métodos propostos na tese, é que ACSVM foi desenvolvido com o mesmo princípio de WEMLP e AUCMLP, ou seja, ACSVM modifica a função custo original proposta para SVMs com margens suaves (Cortes & Vapnik, 1995). Para detalhes sobre a formulação desse método, veja Seção 3.2.2.1 do Capítulo 3.

A metodologia consistiu em aplicar o algoritmo ACSVM sobre as 8 bases de dados mais desbalanceadas listadas na Tabela 5.1 (sat, vow, a18-9, gls6, y9-1, car, y5, a19), usando os mesmos casos de treinamento/teste do Experimento 1. Os resultados obtidos com ACSVM puderam então ser comparados aos resultados de WEMLP e AUCMLP dentro das mesmas condições.

A implementação de ACSVM usada no experimento foi extraída da biblioteca LIBSVM (Chang & Lin, 2011). Optou-se pelo uso da formulação padrão C-SVC com *kernel* gaussiano $K(\mathbf{x}, \mathbf{x}') = \exp\left(-\gamma \left\|\mathbf{x} - \mathbf{x}'\right\|^2\right)$ (Cortes & Vapnik, 1995).

O procedimento de busca em *grid* com *stratified 7-fold crossvalidation* (Van Gestel *et al.*, 2004) foi empregado para configurar os hiperparâmetros C, que controla a força da regularização na formulação C-SVC e, γ, que corresponde à largura do *kernel* gaussiano. Sequências exponencialmente crescentes da forma $C = \{2^{-5}, 2^{-3}, \ldots, 2^{15}\}$ e $\gamma = \{2^{-15}, 2^{-13}, \ldots, 2^{3}\}$ foram escolhidas para compor o *grid* inicial (bi-dimensional) de hiperparâmetros candidatos. A busca em *grid* contou com apenas um refinamento na vizinhança da combinação ótima (C^0, γ^0) selecionada na iteração 0. As sequências de hiperparâmetros candidatos na iteração 1 foram $C = \{(C^0 \times 2^{-2}) : (C^0 \times 2^2)\}$ e $\gamma = \{(\gamma^0 \times 2^{-2}) : (\gamma^0 \times 2^2)\}$, ambas apresentando passo multiplicador de $2^{0.25}$.

[1] ACSVM é também conhecido na literatura como SVM-WEIGHT (Tang *et al.*, 2009).

Adicionalmente, conforme sugerido em Joachims (2002) e Tang *et al.* (2009), os parâmetros ω_1 e ω_2 que regulam os custos associados aos exemplos positivos e negativos, respectivamente, foram configurados como N_2 e N_1 e mantidos constantes em todas as execuções. Na implementação de ACSVM fornecida na biblioteca LIBSVM, ω_i é usado para ponderar o parâmetro regularizador para a classe i, ou seja, $C^i = \omega_i \times C$. O efeito obtido com essa implementação é o mesmo daquele descrito em Lin *et al.* (2002) e, apresentado na Seção 3.2.2.1 do Capítulo 3.

5.3.1 Resultados e testes de significância

As Tabelas 5.5 e 5.6 listam, respectivamente, os valores de *G-mean* e AUC obtidos pelos algoritmos WEMLP, AUCMLP e ACSVM sobre as 8 bases de dados mais desbalanceadas da Tabela 5.1. As médias e desvios-padrão foram calculados sobre 20 diferentes casos de teste. Os melhores valores encontram-se em negrito.

Tabela 5.5: Comparação entre os valores de *G-mean* (em %) obtidos pelos algoritmos WEMLP, AUCMLP e ACSVM sobre as 8 base de dados mais desbalanceadas. Os melhores valores encontram-se em negrito.

Base de dados	WEMLP	AUCMLP	ACSVM
sat	84.82 ± 1.74	85.77 ± 0.75	$\mathbf{89.78 \pm 0.92}$
vow	99.01 ± 1.56	98.37 ± 2.74	$\mathbf{99.74 \pm 1.15}$
a18-9	82.40 ± 5.36	83.04 ± 6.15	$\mathbf{84.23 \pm 6.24}$
gls6	95.56 ± 3.90	87.26 ± 13.27	$\mathbf{96.58 \pm 5.44}$
y9-1	70.54 ± 11.82	70.53 ± 11.56	$\mathbf{73.11 \pm 12.37}$
car	98.31 ± 0.84	97.56 ± 0.97	$\mathbf{98.78 \pm 1.33}$
y5	78.98 ± 6.19	79.63 ± 5.23	$\mathbf{81.84 \pm 5.88}$
a19	76.12 ± 7.96	75.09 ± 6.90	$\mathbf{77.59 \pm 8.06}$

Com o objetivo de se obter conclusões sobre os resultados, o teste de *Friedman* foi aplicado considerando os valores médios obtidos com as métricas *G-mean* (Tabela 5.5) e AUC (Tabela 5.6). A idéia, com o teste de *Friedman*, é verificar

Tabela 5.6: Comparação entre os valores de AUC (em %) obtidos pelos algoritmos WEMLP, AUCMLP e ACSVM sobre as 8 base de dados mais desbalanceadas. Os melhores valores encontram-se em negrito.

Base de dados	WEMLP	AUCMLP	ACSVM
sat	92.57 ± 1.25	93.58 ± 0.35	$\mathbf{95.76 \pm 0.61}$
vow	99.84 ± 0.22	99.84 ± 0.44	$\mathbf{100.00 \pm 0.00}$
a18-9	92.47 ± 3.17	92.35 ± 3.69	$\mathbf{93.64 \pm 2.57}$
gls6	96.27 ± 5.25	98.26 ± 2.87	$\mathbf{99.41 \pm 0.99}$
y9-1	77.94 ± 11.10	75.67 ± 10.68	$\mathbf{81.04 \pm 9.52}$
car	99.33 ± 0.30	98.59 ± 0.94	$\mathbf{99.86 \pm 0.32}$
y5	87.33 ± 3.99	$\mathbf{88.00 \pm 4.17}$	87.92 ± 4.18
a19	$\mathbf{85.79 \pm 3.57}$	84.02 ± 5.23	84.84 ± 4.53

se os algoritmos WEMLP, AUCMLP e ACSVM são significativamente diferentes, em termos de seus *ranks* médios, a um nível de confiança de $1 - \alpha$.

Desde que nesse experimento, M = 8 (bases de dados) e L = 3 (algoritmos), a estatística F_F é distribuída com graus de liberdade 2 (para o numerador) e 14 (para o denominador). Dessa forma, a hipótese nula (H_0), que afirma que os algoritmos são equivalentes, deve ser rejeitada quando $F_F > 3.739$ (para $\alpha = 0.05$). A tabela de valores críticos para a distribuição F (ou F_F) pode ser encontrada em Sheskin (2007).

A Tabela 5.7 lista os *ranks* médios obtidos pelos algoritmos WEMLP, AUCMLP e ACSVM para as métricas *G-mean* e *AUC*. As duas últimas colunas da tabela mostram as correspondentes estatísticas F_F e p-valor referentes ao teste de *Friedman*. Com base nos valores dessas estatísticas, a hipótese nula (H_0) pode ser rejeitada a um nível significância de 5%; o que indica que os algoritmos são significativamente diferentes para ambas as métricas analisadas.

Em seguida, comparações "um-contra-um" entre os algoritmos foram realizadas com o teste *post-hoc* de *Nemenyi*. Considerando $\alpha = 0.05$, a constante q_α para $L = 3$ é igual a 2.343 e, a correspondente diferença crítica (CD) é 1.171. A tabela de valores críticos (q_α) para o teste de *Nemenyi* encontra-se disponível em

Tabela 5.7: *Ranks* médios obtidos pelos algoritmos WEMLP, AUCMLP e ACSVM para as métricas *G-mean* e AUC. As duas últimas colunas da tabela mostram os correspondente valores das estatísticas F_F e p-valor referentes ao teste de *Friedman*.

	WEMLP	AUCMLP	ACSVM	F_F	p-valor
G-mean	2.375	2.625	1.000	22.87	0.022
AUC	2.375	2.375	1.250	5.11	0.034

Demsar (2006). Vale lembrar que nesse teste, dois algoritmos são considerados significativamente diferentes se seus *ranks* médios diferem de pelo menos o valor de CD.

Com base nos valores dos *ranks* médios da Tabela 5.7, pode-se concluir, em termos de *G-mean*, que ACSVM é significativamente melhor que WEMLP e AUCMLP que, novamente, mostram-se equivalentes (para $\alpha = 0.05$).

Com relação à métrica AUC, observe pela Tabela 5.7 que as diferenças entre o melhor algoritmo (ACSVM) e os demais algoritmos (WEMLP e AUCMLP), que possuem *ranks* equivalentes, são menores que o valor crítico $CD = 1.171$. Logo, para $\alpha = 0.05$, pode-se concluir que o teste *post-hoc* não foi poderoso o suficiente para distinguir significativamente os algoritmos testados. Nota-se, no entanto, que as diferenças entre os *ranks* médios ($2.375 - 1.250 = 1.125$) são muito próximas de $CD = 1.171$.

Para $\alpha = 0.10$, o valor de CD torna-se 1.026, uma vez que a constante $q_{0.10}$ para $L = 3$ é igual a 2.052 (segundo tabela disponível em Demsar (2006)). Nesse caso, os resultados do teste *post-hoc* para a métrica AUC indicam que ACSVM é significativamente melhor que o grupo formado por WEMLP e AUCMLP, a um nível de significância de 10%.

5.3.2 Discussão

Os testes estatísticos de significância mostram superioridade no desempenho do método ACSVM sobre WEMLP e AUCMLP para ambas as métricas, *G-mean* e

AUC. Esse resultado aponta para a existência de melhores soluções (modelos), as quais provavelmente não puderam ser alcançadas por nossos métodos devido à limitações em suas formulações. Além disso, ele nos permite especular sobre os motivos que tornam ACSVM mais robusto.

O método ACSVM, diferentemente de WEMLP e AUCMLP, possui incorporada, ao seu processo de aprendizado, uma técnica para o controle efetivo da complexidade dos modelos (ou funções de decisão). Essa propriedade foi herdada da formulação original das SVMs (Cortes & Vapnik, 1995) sobre a qual o método ACSVM foi projetado.

No aprendizado de ACSVM, o funcional custo a ser otimizado controla a complexidade dos modelos, através da imposição de restrições na construção do hiperplano de separação. Busca-se pelo hiperplano que maximiza a margem de separação entre as classes, garantindo-se, ao mesmo tempo, que as perdas (desvios em relação à margem) associadas aos erros sejam minimizadas (Lin *et al.*, 2002). Soluções eficientes, em um senso teórico, podem ser obtidas realizando-se um treinamento com a combinação ótima dos parâmetros regularizador (C) e de *kernel* (γ, no caso particular do *kernel* gaussiano), que regulam o equilíbrio entre a suavidade do modelo e a magnitude dos erros de classificação.

Os métodos propostos na tese, por outro lado, não fornecem controle intrínseco da complexidade dos modelos em suas formulações. Eles consideram somente as perdas associadas aos padrões de treinamento (*risco* empírico) no processo de aprendizado, o que intuitivamente poderia explicar a razão de seus desempenhos terem sido significativamente inferiores àquele apresentado por ACSVM.

A próxima seção desse capítulo (Experimento 3) se destina à verificação dessa hipótese. O procedimento adotado foi contrastar os resultados obtidos por ACSVM com aqueles alcançados pelas extensões multiobjetivo de WEMLP e AUCMLP, apresentadas na Seção 4.4. Tais extensões, denominadas WEMOBJ e AUCMOBJ, também possuem embutidas em seus funcionais custo uma estratégia para controle de complexidade dos modelos, que se dá com a imposição de restrições à magnitude do vetor de pesos da rede MLP.

Adicionalmente, uma vez que todos os métodos (ACSVM, WEMOBJ e AUCMOBJ) contemplam o controle de complexidade em suas formulações, a comparação de seus desempenhos, a ser realizada no experimento a seguir, permite

isolar um aspecto mais importante que é a estratégia adotada por cada um deles para lidar com aplicações desbalanceadas.

5.4 Experimento 3

Esse experimento comparou os resultados obtidos com os métodos WEMOBJ, AUCMOBJ e ACSVM sobre as 8 bases de dados mais desbalanceadas, que se encontram listadas na Tabela 5.1. O objetivo foi verificar se as limitações de desempenho apresentadas por WEMLP e AUCMLP no Experimento 2, realmente podem ser explicadas pela falta do uso de uma técnica de controle de complexidade em suas formulações. Para uma comparação justa com ACSVM, os mesmos casos de treinamento/teste foram utilizados por WEMOBJ e AUCMOBJ.

A rede MLP usada como classificador base possui topologia $n : 20 : 1$ e função de ativação do tipo tangente hiperbólica em todas as unidades. A configuração de parâmetros para o aprendizado MOBJ foi: norma inicial (ε_0) igual a 0.5, normal final (ε_{max}) igual a 10 e, intervalo de variação (δ_ε) igual a 0.5 (vide Seção 4.4). Com relação aos parâmetros que regem as funções custo, WEMOBJ teve seu parâmetro λ ajustado como $\frac{N_2}{N_1+N_2}$, enquanto AUCMOBJ foi aplicado com $\kappa = 1.4$ e $\tau = 2$. Essas configurações foram mantidas constantes para todas as bases de dados.

Em cada execução de WEMOBJ ou AUCMOBJ, um conjunto de validação foi extraído do conjunto de treinamento de forma aleatória e estratificada. Tal conjunto tinha 15% do número total de dados de treinamento e foi usado para a escolha do modelo final, conforme regra de decisão descrita pela Equação (4.54).

5.4.1 Resultados e testes de significância

As Tabelas 5.8 e 5.9 listam, respectivamente, os valores de *G-mean* e AUC obtidos pelos algoritmos ACSVM, WEMOBJ e AUCMOBJ sobre as 8 bases de dados mais desbalanceadas da Tabela 5.1. As médias e desvios-padrão foram calculados sobre 20 diferentes casos de teste. As correspondentes curvas ROC médias encontram-se disponíveis no Apêndice C.

Tabela 5.8: Comparação entre os valores de *G-mean* (em %) obtidos pelos algoritmos ACSVM, WEMOBJ e AUCMOBJ sobre as 8 base de dados mais desbalanceadas. Os melhores valores encontram-se em negrito.

Base de dados	ACSVM	WEMOBJ	AUCMOBJ
sat	$\mathbf{89.78 \pm 0.92}$	87.75 ± 0.93	84.83 ± 1.56
vow	$\mathbf{99.74 \pm 1.15}$	99.05 ± 1.40	97.51 ± 2.64
a18-9	84.23 ± 6.24	81.40 ± 4.36	$\mathbf{85.45 \pm 4.64}$
gls6	96.58 ± 5.44	$\mathbf{99.33 \pm 5.90}$	95.62 ± 5.66
y9-1	73.11 ± 12.37	73.98 ± 8.54	$\mathbf{74.15 \pm 9.25}$
car	98.78 ± 1.33	$\mathbf{99.27 \pm 0.79}$	96.41 ± 2.77
y5	81.84 ± 5.88	$\mathbf{84.09 \pm 4.65}$	81.23 ± 4.83
a19	77.59 ± 8.06	76.75 ± 4.31	$\mathbf{79.78 \pm 3.15}$

Tabela 5.9: Comparação entre os valores de AUC (em %) obtidos pelos algoritmos ACSVM, WEMOBJ e AUCMOBJ sobre as 8 base de dados mais desbalanceadas. Os melhores valores encontram-se em negrito.

Base de dados	ACSVM	WEMOBJ	AUCMOBJ
sat	$\mathbf{95.76 \pm 0.61}$	93.94 ± 0.79	93.45 ± 0.93
vow	$\mathbf{100.00 \pm 0.00}$	99.82 ± 0.27	99.88 ± 0.19
a18-9	93.64 ± 2.57	93.55 ± 3.27	$\mathbf{94.33 \pm 2.75}$
gls6	99.41 ± 0.99	$\mathbf{99.66 \pm 0.58}$	99.53 ± 0.88
y9-1	81.04 ± 9.52	82.14 ± 7.88	$\mathbf{83.37 \pm 7.50}$
car	$\mathbf{99.86 \pm 0.32}$	99.49 ± 0.63	99.73 ± 0.19
y5	87.92 ± 4.18	$\mathbf{88.27 \pm 4.63}$	87.72 ± 4.01
a19	84.84 ± 4.53	86.52 ± 3.44	$\mathbf{87.10 \pm 2.60}$

Devido à proximidade numérica dos resultados, o teste de *Friedman* foi aplicado para checar significância em termos dos *ranks* médios obtidos pelos algoritmos ACSVM, WEMOBJ e AUCMOBJ. Novamente, foram considerados na aplicação do teste, os desempenhos médios estimados com as métricas *G-mean*

(Tabela 5.8) e AUC (Tabela 5.9).

Assim como no Experimento 2, a estatística F_F é distribuída com graus de liberdade 2 (para o numerador) e 14 (para o denominador), o que implica que a hipótese nula (H_0) de *ranks* iguais deve ser rejeitada, quando $F_F > 3.739$, para um nível de significância (α) de 0.05.

A Tabela 5.10 lista os *ranks* médios obtidos pelos algoritmos ACSVM, WEMOBJ e AUCMOBJ para as métricas *G-mean* e AUC. As duas últimas colunas da tabela mostram os correspondentes valores das estatísticas F_F e p-valor referentes ao teste de *Friedman*.

Tabela 5.10: *Ranks* médios obtidos pelos algoritmos ACSVM, WEMOBJ e AUCMOBJ para as métricas *G-mean* e AUC. As duas últimas colunas da tabela mostram os correspondentes valores das estatísticas F_F e p-valor referentes ao teste de *Friedman*.

	ACSVM	WEMOBJ	AUCMOBJ	F_F	p-valor
G-mean	1.875	1.875	2.250	0.344	0.687
AUC	2.000	2.125	1.875	0.111	0.882

Com base nos valores das estatísticas F_F e p-valor obtidos para *G-mean* e AUC, pode-se afirmar, a um nível de significância de 5%, que os algoritmos ACSVM, WEMOBJ e AUCMOBJ são estatisticamente equivalentes.

5.4.2 Discussão

Os testes estatísticos dos resultados confirmam a hipótese levantada na Seção 5.3.2. Pode-se afirmar que o controle de complexidade embutido no classificador base (SVMs) foi o diferencial a favor do algoritmo ACSVM no Experimento 2. A incorporação de técnica análoga (MOBJ) às formulações de WEMLP e AUCMLP possibilitou a busca por soluções mais robustas e, a consequente equivalência de seus desempenhos com ACSVM no Experimento 3.

A validação dessa hipótese também aponta para a importância de se adotar técnicas efetivas para controle de complexidade (ou generalização) nas soluções

propostas para lidar com dados desbalanceados. Pode-se especular que os resultados de uma solução particular serão melhores, caso uma estratégia de suavização (regularização, maximização da margem de separação ou restrição da magnitude de parâmetros) esteja embutida em seu classificador base.

Adicionalmente, a equivalência de desempenho, apontada pelo teste de *Friedman*, com um método cuja eficácia é conhecida na literatura (ACSVM), reforça o conceito de que as formulações (funções custo) propostas na tese são eficientes para equilibrar as taxas de acerto entre as classes e otimizar a Curva ROC.

Por último, os resultados em termos das curvas ROC médias ilustradas no Apêndice C, segurem superioridade de WEMOBJ e AUCMOBJ para as bases de dados contendo as mais elevadas razões de desbalanceamento: y5 (razão = 0.034) e a19 (razão = 0.008). Vale lembrar que esse fato já havia sido notado no Experimento 1.

5.5 Conclusões do capítulo

Nesse capítulo, um estudo empírico foi conduzido para avaliar a eficiência dos algoritmos propostos na tese em lidar com dados reais desbalanceados. Os resultados mostraram que WEMLP e AUCMLP possuem desempenhos similares e podem ser usados para melhorar o desempenho de redes MLPs puras, no que diz respeito ao equilíbrio entre as taxas de acerto das classes (*G-mean*) e, à otimização da curva ROC (AUC). Foi também observado que nossos algoritmos foram superiores, na maioria dos casos, a métodos tradicionais de reamostragem de dados, tais como SMTTL e WWE. Além disso, eles se mostraram estatisticamente equivalentes a RBoost e ACSVM, métodos oriundos de *adaptações em algoritmos de aprendizado* e, cuja eficiência havia sido comprovada em estudos experimentais recentes (Chen *et al.*, 2010; Tang *et al.*, 2009).

Os resultados do estudo empírico também apontaram a importância de se incorporar uma técnica efetiva de controle de complexidade no âmbito do aprendizado com classes desbalanceadas. Esse fato pôde ser confirmado com a extensão multiobjetivo (MOBJ) das formulações de WEMLP e AUCMLP, a qual possibilitou soluções mais robustas para nossos métodos e, a consequente equivalência de seus desempenhos com um método (ACSVM) baseado em SVMs.

Adicionalmente, os resultados sugerem que nossos algoritmos podem produzir melhores resultados quando aplicados a conjuntos de dados que apresentam os mais elevados graus de desbalanceamento. Essa conclusão foi principalmente motivada pelo "domínio" exercido pelas curvas ROC médias de WEMLP e AUCMLP (vide Apêndice C), ao se considerar, por exemplo, as bases de dados y5 (razão = 0.034) e a19 (razão = 0.008).

Capítulo 6

Conclusões e Propostas de Continuidade

Essa tese abordou alguns aspectos teóricos e práticos para o problema do aprendizado indutivo com classes desbalanceadas. Primeiramente, foi mostrado que o viés induzido pelo desequilíbrio das distribuições surge intrinsecamente da minimização de um critério baseado na taxa de Erro global, tendo como fator atenuante o nível de incerteza dos dados (presença de ruído). Embora apareçam na literatura alguns estudos empíricos destinados à investigar as causas/consequências do problema (Japkowicz & Stephen, 2002; Khoshgoftaar *et al.*, 2010; Lawrence *et al.*, 1998; Prati *et al.*, 2004; Weiss, 2004), nesse trabalho esses conceitos foram explorados com base nos fundamentos teóricos do aprendizado de máquina, contribuindo, portanto, no sentido de suprir uma carência por abordagens formais. As idéias aqui discutidas fornecem subsídios para compreensão dos princípios que regem as soluções até então propostas para o problema e, servem como guia para o desenvolvimento de novos métodos de aprendizado.

Torna-se mais claro, a partir da caracterização teórica apresentada, que soluções promissoras para o problema de classes desbalanceadas devem considerar critérios alternativos para seleção de modelos, os quais devem refletir as necessidades do domínio de aplicação em foco. Essa observação ajuda a entender o sucesso empírico das soluções que customizam funcionais custo na abordagem de *adaptações em algoritmos de aprendizado*. Adicionalmente, ela permite explicar a eficácia de alguns métodos da abordagem de *pré-processamento*

de dados, os quais provocam readaptações indiretas na função critério, ao modificarem as distribuições de probabilidade (a prioris) a partir de suas estratégias de reamostragem de dados.

As idéias conceituais provenientes da formalização do problema foram então aplicadas ao desenvolvimento de novos algoritmos de aprendizado para a topologia *MultiLayer Percetron*: WEMLP e AUCMLP. A essência de tais algoritmos está nos seus critérios para seleção de modelos, os quais foram propostos com o objetivo de priorizar taxas de acerto elevadas e equilibradas para as classes e a melhoria da qualidade do *ranking* de classificação.

O critério proposto para o método WEMLP utiliza um parâmetro de custo para distinguir as perdas associadas a cada classe. Foi demonstrado que a incorporação de informação a priori, através do parâmetro de custo, permite obter superfícies de decisão equidistantes das classes. O critério proposto para o algoritmo AUCMLP corresponde a uma aproximação diferenciável da estatística de *Wilcoxon-Mann-Whitney*. Uma restrição imposta na faixa de valores de um dos parâmetros desse funcional permite a seleção de modelos que priorizam a qualidade do *ranking* de classificação, assim como a separabilidade das classes a partir do limiar (*threshold*) padrão.

Várias das propriedades teóricas previstas para WEMLP e AUCMLP foram confirmadas no estudo experimental realizado, como a capacidade de ambos em melhorar a taxa de reconhecimento do grupo minoritário, conseguindo um maior equilíbrio entre as acurácias individuais das classes. Além disso, as vantagens da métrica AUC (*Area Under the ROC Curve*) sobre o Erro global em cenários desbalanceados foram comprovadas, a partir dos melhores resultados apresentados por AUCMLP em relação a redes MLP tradicionais (baseadas na minimização do Erro). Foi também observado que WEMLP (com $\lambda = N_2/(N_1+N_2)$) e AUCMLP possuem desempenhos similares, o que sugere que os funcionais custo propostos para esses algoritmos encontram-se de alguma forma relacionados; seus processos de otimização produzem modelos (soluções) com propriedades semelhantes.

Uma outra contribuição desse trabalho foi mostrar a importância da adoção de uma estratégia para controle efetivo de complexidade de modelos no âmbito do aprendizado com classes desiguais. Embora esse conceito já esteja bem sedimentado nas formulações baseadas no Erro global (Geman *et al.*, 1992; Vapnik,

1995), ele ainda não tinha sido apontado como fator fundamental nas formulações propostas para lidar com dados desbalanceados. Uma possível explicação para o fato da "questão complexidade" não ter ainda vindo à tona, pode ser dada a partir da metodologia comumente usada para testar novas soluções para o problema de classes desbalanceadas. Na maioria dos casos, os testes ocorrem usando o mesmo classificador base, com uma configuração padrão para seus parâmetros. O uso dessa abordagem tende a *mascarar* a influência da complexidade nos desempenhos dos algoritmos testados.

Ao se considerar a "questão complexidade", o problema do aprendizado deve ser visto como um problema bi-objetivo, com a minimização de um funcional risco empírico (R_{emp}), medindo as perdas sobre os padrões de treinamento, e a minimização de um funcional complexidade (Ω), que reflete a flexibilidade dos modelos fornecidos por uma máquina de aprendizado. No caso particular desse trabalho, extensões bi-objetivo (MOBJ) para as formulações de WEMLP e AUCMLP foram apresentadas. A flexibilidade dos modelos é controlada com a imposição de restrições à magnitude (norma euclidiana) do vetor de pesos da rede. Os problemas de aprendizado originalmente propostos para WEMLP e AUCMLP foram então reformulados para que a norma dos pesos (Ω) fosse minimizada de forma simultânea aos seus correspondentes funcionais custo (R_{emp}). A eficiência da abordagem MOBJ foi comprovada com a obtenção de soluções mais robustas em contraste com as soluções produzidas pela abordagem mono-objetivo (R_{emp}) associada à estratégia *k-fold crossvalidation*.

Por fim, espera-se que os resultados do presente estudo, em termos dos conceitos teóricos e práticos apresentados, possam ser aplicados em problemas reais desbalanceados, bem como possam ser aproveitados para o projeto de novos algoritmos de aprendizado.

6.1 Propostas de Continuidade

Sugere-se como propostas de continuidade desse trabalho, investir nos seguintes problemas relacionados ao tema:

- Projeto de novos algoritmos de aprendizado para outras topologias de redes, tais como RBF e ANFIS, usando os funcionais custo desenvolvidos na tese.

- Investigação de métodos de otimização mais robustos com o objetivo de melhorar a velocidade de convergência e a estabilidade dos algoritmos WEMLP, AUCMLP e suas extensões multiobjetivo (MOBJ).

- Em se tratando dos funcionais critério propostos na tese, seu relacionamento pode ser investigado com o objetivo de explicar os fatores que fazem com que os modelos selecionados por WEMLP, com $\lambda = N_2/(N_1 + N_2)$, sejam similares àqueles selecionados por AUCMLP. Um possível ponto de partida para essa investigação pode estar em um dos resultados teóricos apresentados em Rudin & Schapire (2009). Nesse trabalho, é demonstrado, no escopo de algoritmos de *Boosting*, que otimizar uma função custo onde os exemplos positivos e negativos contribuem igualmente (pesos iguais) é aproximadamente equivalente a minimizar a probabilidade de *misranking*. Esse resultado foi usado para explicar o sucesso empírico do algoritmo *AdaBoost* (Freund & Schapire, 1997) em otimizar a AUC, embora ele não tenha sido originalmente projetado para essa tarefa.

- As implicações teóricas por trás das extensões multiobjetivo (MOBJ) também abrem caminhos para investigações futuras. Limites na capacidade de generalização (*generalization bounds*) de máquinas de aprendizado podem ser estudados tomando como base os critérios propostos para WEMLP e AUCMLP. Tais limites são importantes para mostrar formalmente que soluções robustas (que generalizam bem) são mais prováveis de serem obtidas a partir de um equilíbrio entre os funcionais risco empírico (R_{emp}) e a complexidade do espaço de funções (hipóteses) (Ω). Tentativas nessa direção para a função custo baseada na estatística de *Wilcoxon-Mann-Whitney* (AUC) foram feitas, respectivamente, em Agarwal *et al.* (2005); Rudin & Schapire (2009).

- Extensão dos algoritmos propostos na tese para problemas de classificação envolvendo mais de duas classes (multiclasse).

Uma alternativa simples e direta é considerar a decomposição de um problema de classificação com $c > 2$ classes dentro de múltiplos problemas com duas classes. As abordagens mais comuns para efetuar essa decomposição são *one-against-all* e *one-against-one*. Tais abordagens são independentes do algoritmo de aprendizado, podendo assim, serem aplicadas com WEMLP, AUCMLP e suas correspondentes extensões MOBJ. Para detalhes sobre o funcionamento dessas estratégias recomenda-se os trabalhos de Bishop (2006); Vapnik (1998).

Outra possibilidade seria estender diretamente as formulações de WEMLP e AUCMLP para contemplar problemas multiclasse. Um possível caminho para a reformulação de AUCMLP seria considerar generalizações multiclasse da estatística de *Wilcoxon-Mann-Whitney* propostas na literatura. Dentre elas, pode-se citar o VUS (*Volume Under the ROC Surface*) (Hand & Till, 2001), que é baseado na agregação de valores de AUC para todos os pares de classes e, uma extensão do coeficiente *Gini* (medida análoga à AUC), apresentada em Everson & Fieldsend (2006b).

No caso particular de WEMLP, os parágrafos, a seguir, apresentam nossas idéias para a extensão de sua formulação para problemas com $c > 2$ classes. Para melhor compreensão das idéias, uma breve análise sobre as propriedades da solução teórica buscada pela formulação original de WEMLP, quando da introdução de informação a priori a partir do parâmetro λ, é aqui fornecida.

Conforme visto no Capítulo 2, o objetivo da formulação padrão do aprendizado para classificação binária é a minimização da probabilidade do erro global de classificação. Introduzindo o parâmetro λ, e seu complemento $(1-\lambda)$, diretamente à expressão dessa probabilidade, tem-se

$$\begin{aligned} R[f] &= \lambda\, P(\mathbf{x} \in \mathcal{R}_0, y = 1) + (1-\lambda)\, P(\mathbf{x} \in \mathcal{R}_1, y = 0) \\ &= \lambda \int_{\mathcal{R}_0} p(\mathbf{x}|y=1)P(y=1)\, d\mathbf{x} + \\ &\quad (1-\lambda) \int_{\mathcal{R}_1} p(\mathbf{x}|y=0)P(y=0)\, d\mathbf{x} \end{aligned} \tag{6.1}$$

onde $P(\mathbf{x} \in \mathcal{R}_j, y = k)$ é a probabilidade conjunta de $\mathbf{x}$ ser atribuído à classe j, sendo que sua verdadeira classe é k. Substituindo em (6.1) as probabilidades a priori $P(y = k)$ pelas proporções de exemplos N_k/N no conjunto de treinamento e, incorporando a informação a priori $\lambda = N_0/N$ e $(1 - \lambda) = N_1/N$, conforme sugerido em WEMLP; é possível mostrar que a solução ótima f_0 que minimiza o funcional (6.1) é aquela que atribui cada exemplo de entrada $\mathbf{x}$ à classe k cujo valor de densidade condicional $p(\mathbf{x}|y = k)$ é maior, ou seja

$$f_0(\mathbf{x}) = \begin{cases} 1 & \text{se } \frac{p(\mathbf{x}|y=1)}{p(\mathbf{x}|y=0)} > 1, \\ 0 & \text{caso contrário.} \end{cases} \tag{6.2}$$

Observe a partir de (6.2) que a estratégia adotada por WEMLP no caso binário (contendo somente duas classes), busca uma solução teórica (f_0) que desconsidera a influência das probabilidades a priori, confiando somente na informação associada às características observadas, ou seja, nas verossimilhanças $p(\mathbf{x}|y = k)$.

Mudando o ponto de vista para o caso geral de c classes, o objetivo da formulação padrão do aprendizado torna-se a maximização da probabilidade de um exemplo ser corretamente classificado[1], dada pela seguinte expressão (Duda *et al.*, 2000)

$$\begin{aligned} R[f] = P(\text{Correto}) &= \sum_{k=1}^{c} P(\mathbf{x} \in \mathcal{R}_k, y = k) \\ &= \sum_{k=1}^{c} P(\mathbf{x} \in \mathcal{R}_k | y = k) P(y = k) \\ &= \sum_{k=1}^{c} \int_{\mathcal{R}_k} p(\mathbf{x}|y = k) P(y = k) d\mathbf{x} \end{aligned} \tag{6.3}$$

Partindo-se do mesmo princípio adotado no caso binário, um caminho inicial para generalização da formulação de WEMLP seria a incorporação de

[1] A definição do funcional *risco* em termos das probabilidades de acerto das classes é mais simples, uma vez que para o caso multiclasse existem mais formas de se errar do que de se acertar (Duda *et al.*, 2000).

parâmetros distintos (λ_k) para cada termo (classe) do somatório (6.3), de forma que os efeitos induzidos pelas a prioris ($P(y = k)$) possam ser anulados e, consequentemente, a solução alvo (teórica) do aprendizado confie somente nas informações associadas às verossimilhanças das classes $p(\mathbf{x}|y = k)$. A aproximação empírica desse funcional considerando um número limitado de exemplos, levaria à uma nova função custo para WEMLP, a qual poderia ser diretamente aplicada a problemas multiclasse.

Apêndice A

Esse apêndice fornece expressões analíticas para funções discriminantes derivadas de distribuições gaussianas multivariadas. Para facilitar o entendimento, os conceitos são apresentados com a mesma notação do Capítulo 2. Para mais detalhes sobre o assunto, veja Duda *et al.* (2000).

A.1 Funções discriminantes

Seja a regra de decisão (ou classificador) que minimiza a probabilidade do erro global de classificação

$$f_0(\mathbf{x}) = \begin{cases} 1 & \text{se } p(\mathbf{x}|y=1)P(y=1) \geq p(\mathbf{x}|y=0)P(y=0), \\ 0 & \text{caso contrário.} \end{cases} \tag{A.1}$$

Uma representação alternativa para (A.1) pode ser obtida em termos das funções discriminantes para cada classe, $g_k(\mathbf{x})$ com $k = 0, 1$. Fazendo $g_k(\mathbf{x}) = p(\mathbf{x}|y=k)P(y=k)$, a regra de decisão $f_0(\mathbf{x})$ deve atribuir um vetor arbitrário $\mathbf{x}$ à classe k se

$$g_k(\mathbf{x}) > g_j(\mathbf{x}) \quad \text{para } j \neq k \tag{A.2}$$

A escolha da função discriminante não é única. Pode-se, por exemplo, multiplicar/somar todas as funções discriminantes pela mesma constate positiva sem influenciar a decisão. Em outras palavras, se $g_k(\mathbf{x})$ for substituída por $\zeta(g_k(\mathbf{x}))$, onde $\zeta(\cdot)$ é uma função monotonicamente crescente, o resultado da classificação

não se altera. Essa observação pode levar a simplificações analíticas significativas. Em particular, para a regra de decisão $f_0(\mathbf{x})$, que minimiza a taxa de erro global, a seguinte representação para funções discriminantes tem sido comumente adotada

$$\begin{aligned} g_k(\mathbf{x}) &= \ln\left(p(\mathbf{x}|y=k)P(y=k)\right) \\ &= \ln\left(p(\mathbf{x}|y=k)\right) + \ln\left(P(y=k)\right) \end{aligned} \tag{A.3}$$

onde $\ln(\cdot)$ denota o logaritmo natural.

A.2 Funções discriminantes para a densidade gaussiana

A expressão geral para a densidade normal (gaussiana) multivariada (n-dimensional) é dada por

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{\frac{n}{2}}\,|\mathbf{\Sigma}|^{\frac{1}{2}}} \exp\left[-\frac{1}{2}(\mathbf{x}-\mu)^T\mathbf{\Sigma}^{-1}(\mathbf{x}-\mu)\right] \tag{A.4}$$

onde $\mathbf{x}$ é um vetor coluna de dimensão n, μ é o vetor de médias de dimensão n, $\mathbf{\Sigma}$ é a matriz de covariância de dimensão n x n, sendo $|\mathbf{\Sigma}|$ e $\mathbf{\Sigma}^{-1}$, respectivamente, o determinante e a inversa de $\mathbf{\Sigma}$.

Considere, novamente, a expressão geral das discriminantes para o classificador de mínimo erro global,

$$g_k(\mathbf{x}) = \ln\left(p(\mathbf{x}|y=k)\right) + \ln\left(P(y=k)\right) \tag{A.5}$$

Assumindo que as densidades condicionais são distribuições normais multivariadas, ou seja, $p(\mathbf{x}|y=k) \sim N(\mu_k, \mathbf{\Sigma}_k)$, pode-se escrever a partir de (A.4) que

$$
\begin{aligned}
g_k(\mathbf{x}) &= \ln\left(\exp\left[-\frac{1}{2}(\mathbf{x}-\mu_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\mu_k)\right]\right) + \ln\left(\frac{1}{(2\pi)^{\frac{n}{2}}\,|\boldsymbol{\Sigma}_k|^{\frac{1}{2}}}\right) + \ln\left(P(y=k)\right) \\
&= -\frac{1}{2}(\mathbf{x}-\mu_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\mu_k) + \ln 1 - \ln\left((2\pi)^{\frac{n}{2}}\,|\boldsymbol{\Sigma}_k|^{\frac{1}{2}}\right) + \ln\left(P(y=k)\right) \\
&= -\frac{1}{2}(\mathbf{x}-\mu_k)^T\boldsymbol{\Sigma}_k^{-1}(\mathbf{x}-\mu_k) - \frac{n}{2}\ln 2\pi - \frac{1}{2}\ln|\boldsymbol{\Sigma}_k| + \ln\left(P(y=k)\right) \qquad \text{(A.6)}
\end{aligned}
$$

As seções a seguir analisam (A.6) para alguns casos especiais.

A.2.1 Caso 1: $\boldsymbol{\Sigma}_k = \sigma^2\mathbf{I}$

O caso mais simples ocorre quando os atributos (características) são estatisticamente independentes e cada atributo possui a mesma variância σ^2. Nesse caso, as matrizes de covariância $\boldsymbol{\Sigma}_k$ são idênticas e diagonais, sendo iguais a σ^2 vezes a matriz identidade $\mathbf{I}$. Geometricamente, isso corresponde à situação em que as classes são *clusters* hiperesféricos, cada qual centrado no seu correspondente vetor de médias μ_k.

O determinante e a inversa de $\boldsymbol{\Sigma}_k$ são dados, respectivamente, por $|\boldsymbol{\Sigma}_k| = \sigma^{2n}$ e $\boldsymbol{\Sigma}_k^{-1} = (1/\sigma^2)\,\mathbf{I}$. Pelo fato de ambos os termos $|\boldsymbol{\Sigma}_k|$ e $\frac{n}{2}\ln 2\pi$ na Equação (A.6) serem independentes de k, eles são considerados como aditivos constantes e podem ser ignorados. Então, as seguintes funções discriminantes simplificadas são obtidas

$$
g_k(\mathbf{x}) = -\frac{\|\mathbf{x}-\mu_k\|}{2\sigma^2} + \ln\left(P(y=k)\right) \qquad \text{(A.7)}
$$

onde $\|\cdot\|$ denota a norma euclidiana, dada por

$$
\|\mathbf{x}-\mu_k\| = (\mathbf{x}-\mu_k)^T(\mathbf{x}-\mu_k) \qquad \text{(A.8)}
$$

A expansão da forma quadrática $(\mathbf{x}-\mu_k)^T(\mathbf{x}-\mu_k)$ leva à seguinte expressão

$$
g_k(\mathbf{x}) = -\frac{1}{2\sigma^2}\left[\mathbf{x}^T\mathbf{x} - 2\mu_k^T\mathbf{x} + \mu_k^T\mu_k\right] + \ln\left(P(y=k)\right) \qquad \text{(A.9)}
$$

a qual parece ser uma função quadrática de $\mathbf{x}$. Entretanto, desde que o termo quadrático $\mathbf{x}^T\mathbf{x}$ é o mesmo para todo k, ele também pode ser ignorado. Assim, a

expressão geral para $g_k(\mathbf{x})$ torna-se uma função linear de $\mathbf{x}$, podendo ser reescrita na seguinte forma

$$g_k(\mathbf{x}) = \omega_k^T \mathbf{x} + b_{k0} \tag{A.10}$$

onde

$$\omega_k = \frac{1}{\sigma^2}\mu_k \tag{A.11}$$

e

$$b_{k0} = -\frac{1}{2\sigma^2}\mu_k^T \mu_k + \ln\left(P(y=k)\right) \tag{A.12}$$

O termo b_{k0} é conhecido como *bias* associado à classe k. Uma vez que as discriminantes $g_k(\mathbf{x})$ são lineares, as superfícies de decisão (separação) assumem o formato de hiperplanos, definidos pelas equações $g_k(\mathbf{x}) = g_j(\mathbf{x})$, com $j \neq k$.

A.2.2 Caso 2: $\boldsymbol{\Sigma}_k = \boldsymbol{\Sigma}$

Um outro caso especial surge quando as matrizes de covariância $\boldsymbol{\Sigma}_k$ para as duas classes são idênticas, porém arbitrárias. Geometricamente, isso corresponde à situação na qual as classes possuem o formato de hiperelipsóides de tamanhos e formas iguais, sendo que k-ésima classe encontra-se centrada no vetor de médias μ_k. Pelo fato de ambos os termos $|\boldsymbol{\Sigma}_k|$ e $\frac{n}{2}\ln 2\pi$ na Equação (A.6) serem independentes de k, eles podem ser ignorados. Essa simplificação leva à seguinte expressão geral para as funções discriminantes

$$g_k(\mathbf{x}) = -\frac{1}{2}(\mathbf{x}-\mu_k)^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mu_k) + \ln\left(P(y=k)\right) \tag{A.13}$$

A expansão da forma quadrática $(\mathbf{x}-\mu_k)^T\boldsymbol{\Sigma}^{-1}(\mathbf{x}-\mu_k)$ resulta em uma soma envolvendo um termo quadrático $\mathbf{x}^T\boldsymbol{\Sigma}^{-1}\mathbf{x}$ que é também independente de k. Excluindo esse termo da expressão (A.13), as funções discriminantes resultantes podem ser novamente escritas na forma linear

$$g_k(\mathbf{x}) = \omega_k^T \mathbf{x} + b_{k0} \tag{A.14}$$

onde

$$\omega_k = \mathbf{\Sigma}^{-1}\mu_k \tag{A.15}$$

e

$$b_{k0} = -\frac{1}{2}\mu_k^T\mathbf{\Sigma}^{-1}\mu_k + \ln\left(P(y=k)\right) \tag{A.16}$$

Desde que as funções discriminantes são lineares, as superfícies de decisão resultantes são novamente hiperplanos.

A.2.3 Caso 3: $\mathbf{\Sigma}_k$ = arbitrária

No caso geral, as matrizes de covariância das classes $\mathbf{\Sigma}_k$ são diferentes. Consequentemente, o único termo que pode ser excluído da Equação (A.6) é $\frac{n}{2}\ln 2\pi$. As funções discriminantes resultantes assumem a seguinte forma quadrática

$$g_k\left(\mathbf{x}\right) = \mathbf{x}^T\mathbf{W}_k^T\mathbf{x} + \omega_k^T\mathbf{x} + b_{k0} \tag{A.17}$$

onde

$$\mathbf{W}_k = -\frac{1}{2}\mathbf{\Sigma}_k^{-1} \tag{A.18}$$

$$\omega_k = \mathbf{\Sigma}_k^{-1}\mu_k \tag{A.19}$$

e

$$b_{k0} = -\frac{1}{2}\mu_k^T\mathbf{\Sigma}_k^{-1}\mu_k - \frac{1}{2}\ln\left|\mathbf{\Sigma}_k\right| + \ln\left(P(y=k)\right) \tag{A.20}$$

No caso de duas classes (binário), as superfícies de decisão resultantes são hiperquádricas, podendo assumir as seguintes formas: hiperplanos, pares de hiperplanos, hiperesferas, hiperelipsóides, hiperparabolóides e hiperhiperbolóides.

Apêndice B

Esse apêndice fornece a formulação básica do algoritmo *BackPropagation* padrão (Rumelhart & McClelland, 1986) em modo *batch*, na qual os parâmetros (pesos e bias) da rede *MultiLayer Perceptron* (MLP) são atualizados somente após a apresentação de todos os exemplos do conjunto de dados, ou seja, a cada época de treinamento. Para facilitar o entendimento, a formulação é apresentada com a mesma notação adotada no Capítulo 4, que descreve os métodos de aprendizado propostos na tese.

B.1 Algoritmo *BackPropagation*

Seja uma rede MLP com n entradas, uma camada escondida com h unidades (neurônios) e uma camada de saída contendo uma única unidade, conforme ilustrado pela Figura B.1. O valor de saída obtido na unidade escondida s da rede, devido à apresentação de um vetor de entrada $\mathbf{x} = \{x_1, x_2, \ldots, x_n\}$, é dado pela seguinte expressão,

$$z_s = \phi\left(u_s\right) = \phi\left(\sum_{r=0}^{n} w_{sr}\, x_r\right) \tag{B.1}$$

onde w_{sr} denota um peso entre a unidade escondida s e a unidade de entrada r; $\phi(\cdot)$ é a função de ativação. Similarmente, o valor obtido na unidade de saída da rede, é calculado com base nas saídas emitidas pelas unidades escondidas,

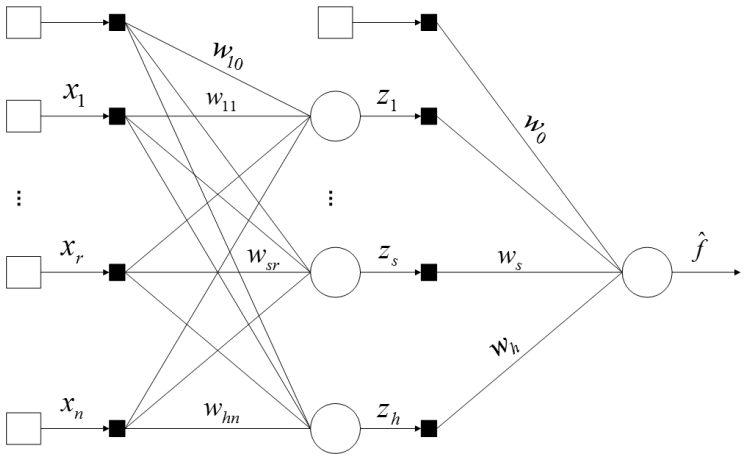


Figura B.1: Exemplo de Rede *MultiLayer Perceptron*.

$$\hat{f} = \phi(v) = \phi\left(\sum_{s=0}^{h} w_s \, z_s\right) \tag{B.2}$$

na qual w_s representa um peso entre o neurônio de saída e a unidade escondida s. O termo *bias* foi considerado como uma unidade (entrada/escondida) extra com valor igual a 1.

Dado o conjunto de dados $T = \{(\mathbf{x}(i), y(i)) \mid i = 1 \ldots N\}$, com $y(i)$ denotando o rótulo (saída desejada) para cada vetor de entrada $\mathbf{x}(i) \in \mathbb{R}^n$, o sinal de erro (estimado na saída da rede) para o i-ésimo exemplo de treinamento é dado por $e(i) = y(i) - \hat{f}(i)$. Com base nessa expressão, a função custo somatório dos erros quadráticos sobre o conjunto T pode ser definida como segue,

$$J(\mathbf{w}) = \frac{1}{2} \sum_{i=1}^{N} e^2(i) \quad \forall \, \mathbf{x}(i) \in T \tag{B.3}$$

com $\mathbf{w}$ representando o vetor que armazena todos os parâmetros (pesos e bias) da rede.

B.1.1 Regra de atualização dos pesos

A regra de aprendizado do algoritmo *BackPropagation* padrão é baseada no método do gradiente descendente (Luenberger, 1984). Os parâmetros da rede são inicializados com valores aleatórios e atualizados, a cada iteração (época), na direção oposta do vetor gradiente, conforme as Equações (B.4) e (B.5) a seguir,

$$\Delta \mathbf{w} = -\eta\, \mathbf{g}(\mathbf{w}) \tag{B.4}$$

$$\mathbf{w}_{new} = \mathbf{w}_{old} + \Delta \mathbf{w}_{old} \tag{B.5}$$

onde $\mathbf{g}(\mathbf{w})$ é o vetor gradiente da função custo (B.3) em relação ao vetor de pesos corrente $\mathbf{w}$ e, η é uma constante positiva (taxa de aprendizado) que indica o tamanho do termo de atualização (B.4) aplicado a cada época de treinamento.

B.1.2 Vetor gradiente

Cada componente do vetor gradiente $\mathbf{g}(\mathbf{w})$ é dado pela derivada parcial da função custo (B.3) em relação a um peso arbitrário da rede w_l, conforme Equação (B.6) a seguir,

$$\begin{aligned} \frac{\partial J}{\partial w_l} &= \frac{1}{2}\sum_{i=1}^{N} \frac{\partial e^2(i)}{\partial w_l} \\ &= \sum_{i=1}^{N} e(i)\frac{\partial e(i)}{\partial w_l} \end{aligned} \tag{B.6}$$

onde a expressão para o cálculo da derivada parcial $\frac{\partial e(i)}{\partial w_l}$, devido à apresentação do i-ésimo vetor de entrada, é definida da seguinte forma:

- se w_l corresponde a um peso arbitrário da camada de saída, ou seja, w_l representa w_s na Fig. (B.1), o escalar gradiente, devido ao i-ésimo exemplo de treinamento, é obtido pela seguinte regra da cadeia,

$$\begin{aligned}\frac{\partial e(i)}{\partial w_s} &= \frac{\partial e(i)}{\partial \hat{f}(i)}\frac{\partial \hat{f}(i)}{\partial v(i)}\frac{\partial v(i)}{\partial w_s} \\ &= -\phi'\left(v(i)\right)\, z_s(i) \end{aligned} \tag{B.7}$$

- similarmente, se w_l é um peso arbitrário da camada escondida, ou seja, w_l corresponde a w_{sr} (vide Fig. (B.1)), o escalar gradiente é obtido por,

$$\begin{aligned}\frac{\partial e(i)}{\partial w_{sr}} &= \frac{\partial e(i)}{\partial z_s(i)}\frac{\partial z_s(i)}{\partial u_s(i)}\frac{\partial u_s(i)}{\partial w_{sr}} \\ &= -\phi'(v(i))\, w_s\, \phi'(u_s(i))\, x_r(i) \end{aligned} \tag{B.8}$$

Apêndice C

Esse apêndice apresenta as curvas ROC médias obtidas nos experimentos conduzidos no Capítulo 5. Para um algoritmo particular, uma curva ROC média foi estimada com a aplicação da técnica *threshold averaging* sobre 20 diferentes subconjuntos de teste extraídos de uma dada base de dados. Detalhes sobre a técnica *threshold averaging* podem ser encontrados em Fawcett (2006).

Para facilitar a visualização na comparação dos algoritmos, alguns gráficos ROC são apresentados entre as faixas $[0.0, 0.4]$ para FPr e $[0.6, 1.0]$ para TPr. Isso foi necessário, principalmente, para aquelas bases de dados que apresentavam menor grau de dificuldade de aprendizado. Para tais bases, as curvas ROC médias produzidas pelos algoritmos tendem a ser muito próximas entre si, o que dificulta sua comparação quando todo o espaço ROC é considerado.

C.1 Gráficos ROC referentes ao Experimento 1

As figuras a seguir ilustram as curvas ROC médias estimadas pelos algoritmos MLP, SMTTL, WWE, RBoost, WEMLP e AUCMLP para as 10 bases mais desbalanceadas do Experimento 1. São elas: gls7, euth, sat, vow, a18-9, gls6, y9-1, car, y5, a19. Detalhes sobre o Experimento 1 podem ser encontrados na Seção 5.2 do Capítulo 5.

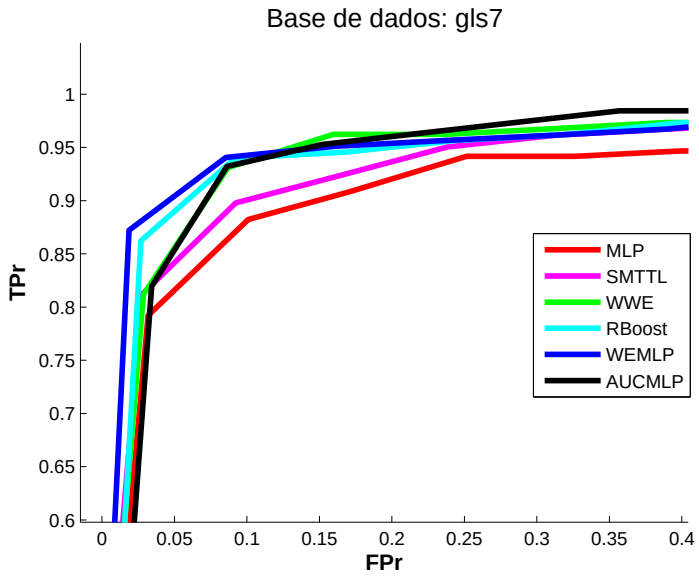


Figura C.1: Curvas ROC médias para a base de dados gls7.

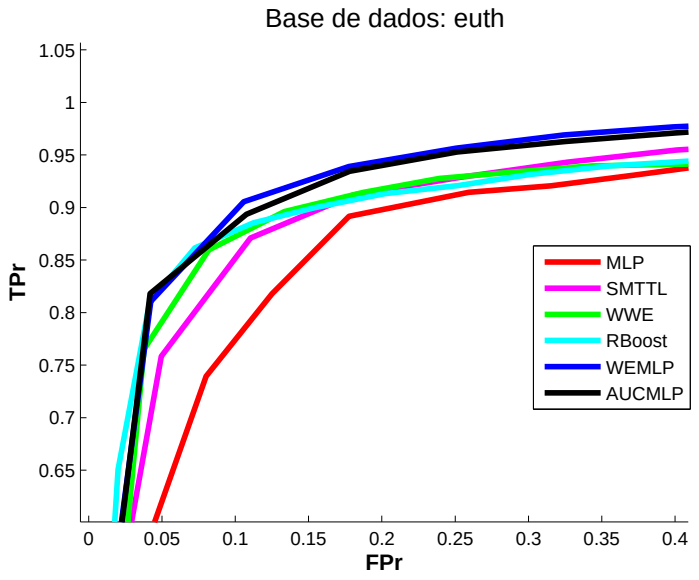


Figura C.2: Curvas ROC médias para a base de dados euth.

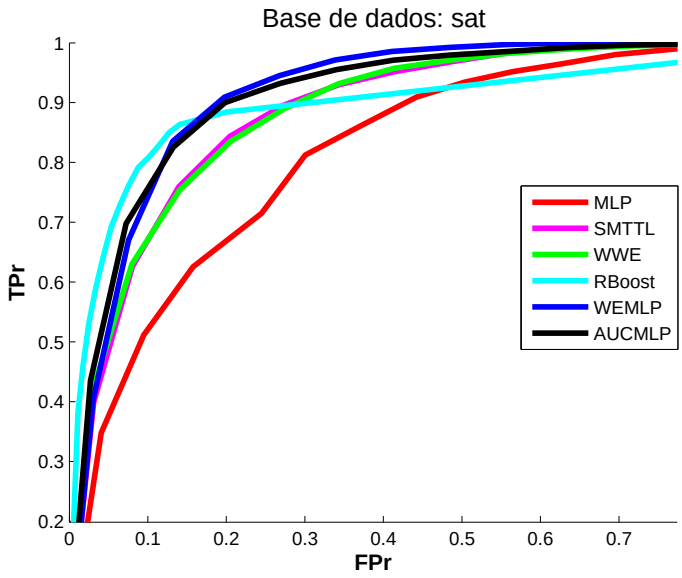


Figura C.3: Curvas ROC médias para a base de dados sat.

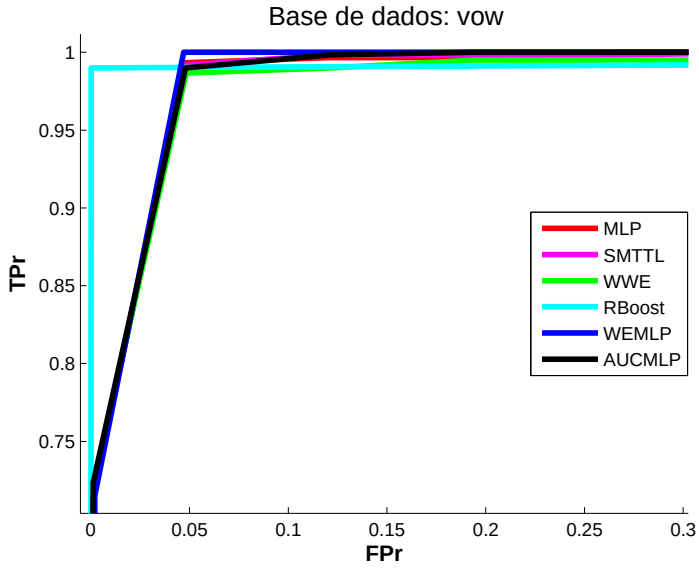


Figura C.4: Curvas ROC médias para a base de dados vow.

C.2 Gráficos ROC referentes ao Experimento 3

As figuras a seguir ilustram as curvas ROC médias estimadas pelos algoritmos ACSVM, WEMOBJ e AUCMOBJ para as bases de dados usadas no Experimento

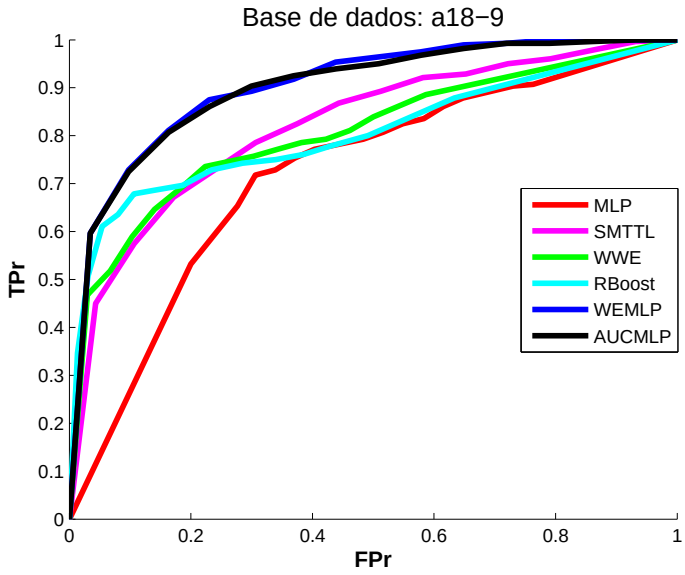


Figura C.5: Curvas ROC médias para a base de dados a18-9.

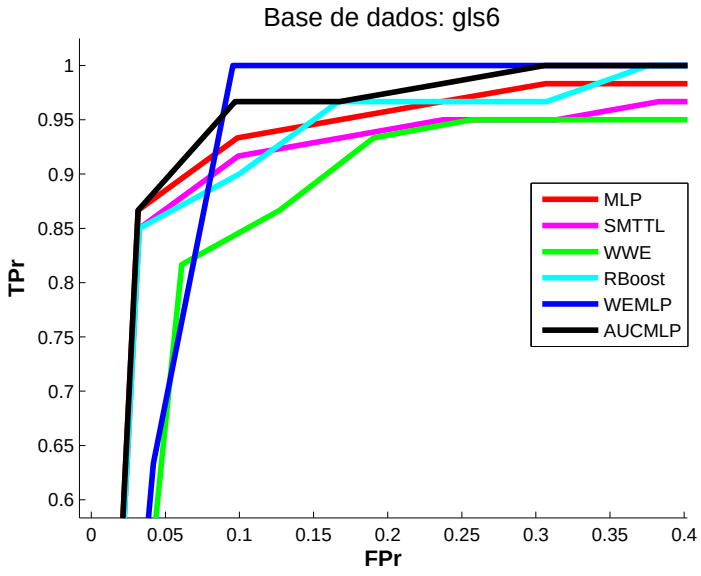


Figura C.6: Curvas ROC médias para a base de dados gls6.

3: sat, vow, a18-9, gls6, y9-1, car, y5, a19. Detalhes sobre o Experimento 3 podem ser encontrados na Seção 5.4 do Capítulo 5.

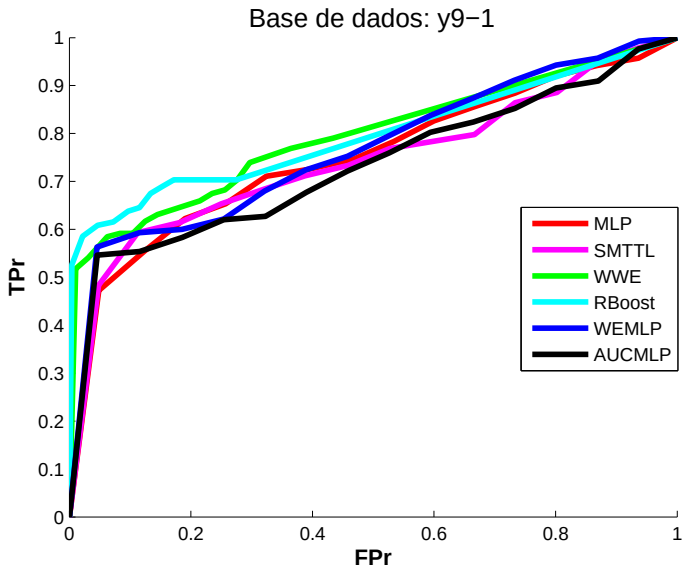


Figura C.7: Curvas ROC médias para a base de dados y9-1.

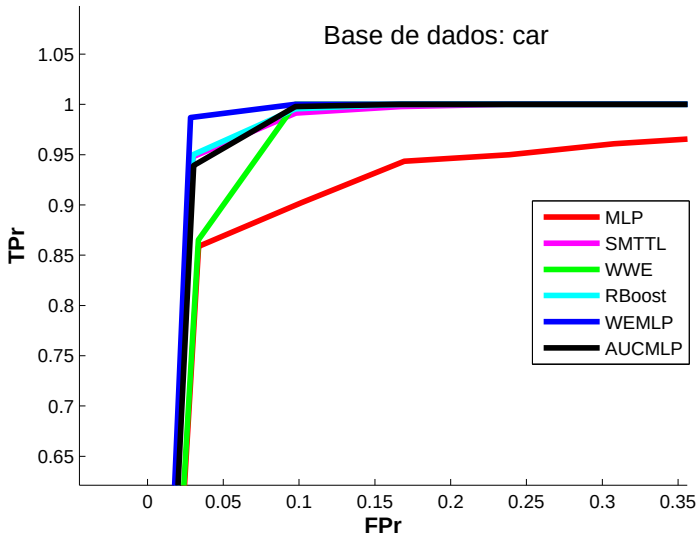


Figura C.8: Curvas ROC médias para a base de dados car.

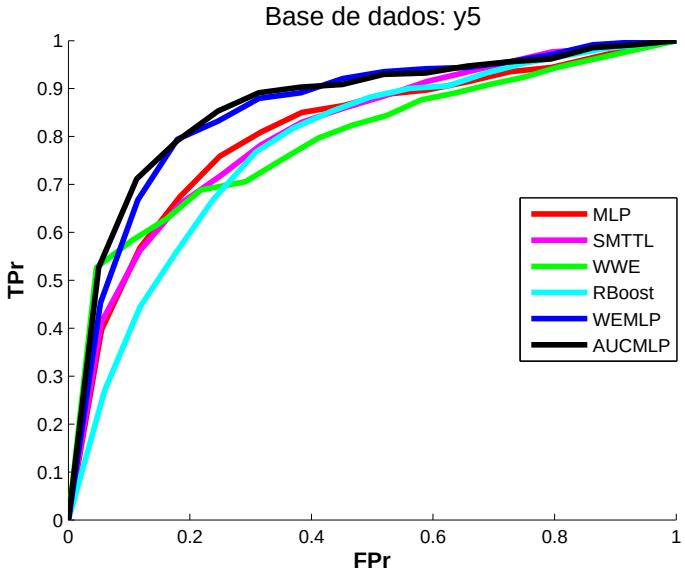


Figura C.9: Curvas ROC médias para a base de dados y5.

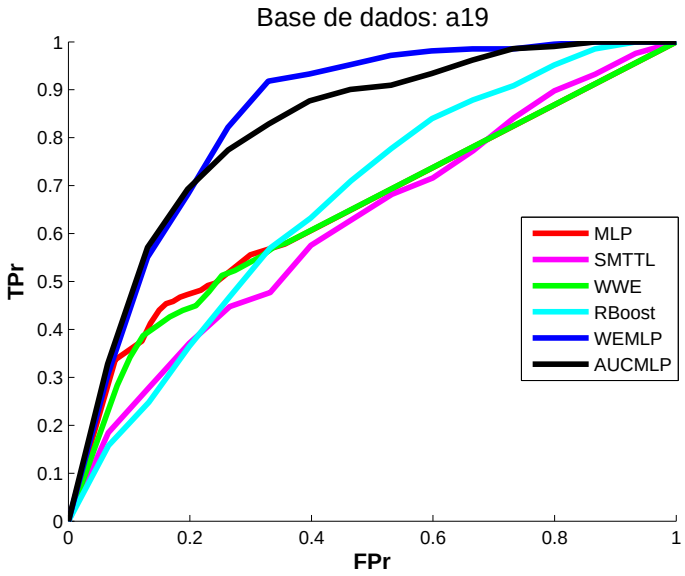


Figura C.10: Curvas ROC médias para a base de dados a19.

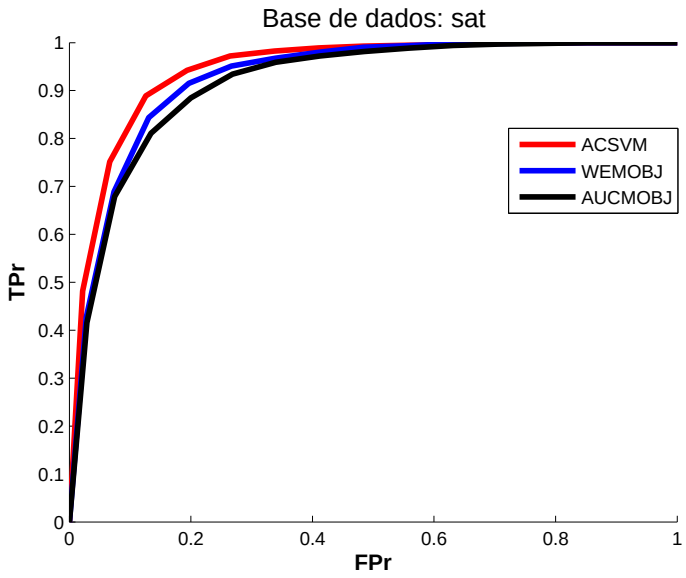


Figura C.11: Curvas ROC médias para a base de dados sat.

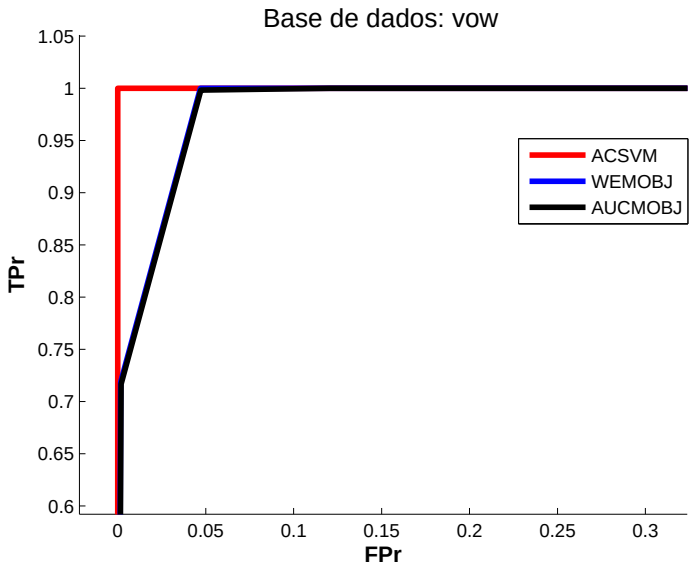


Figura C.12: Curvas ROC médias para a base de dados vow.

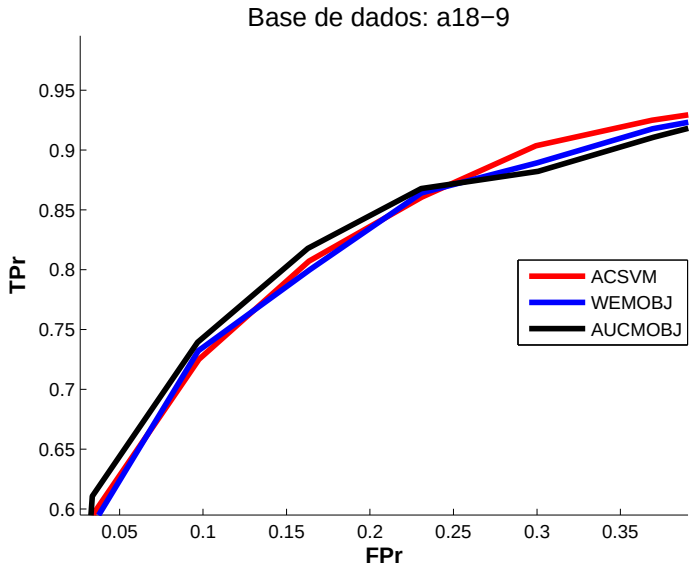


Figura C.13: Curvas ROC médias para a base de dados a18-9.

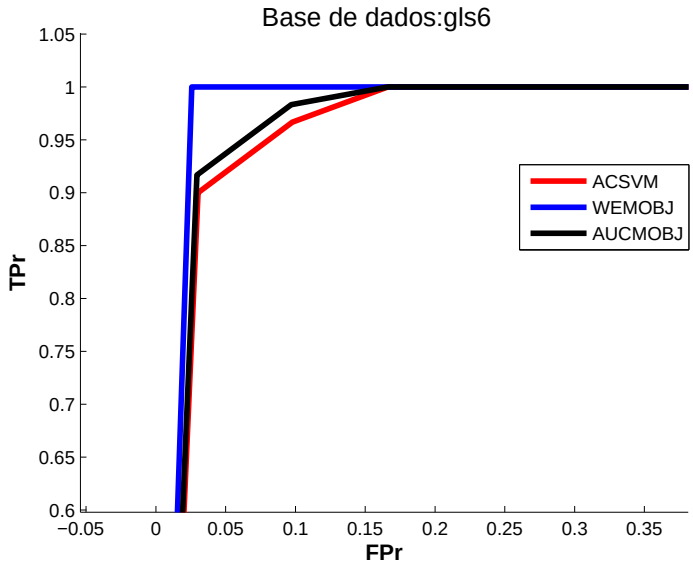


Figura C.14: Curvas ROC médias para a base de dados gls6.

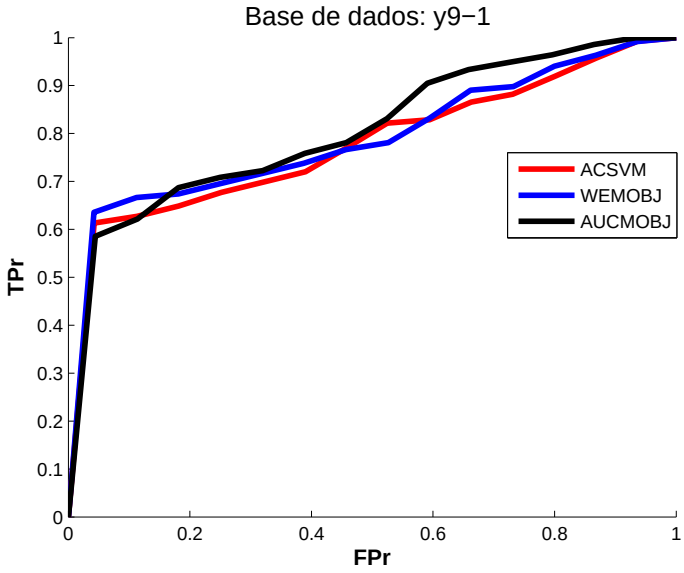


Figura C.15: Curvas ROC médias para a base de dados y9-1.

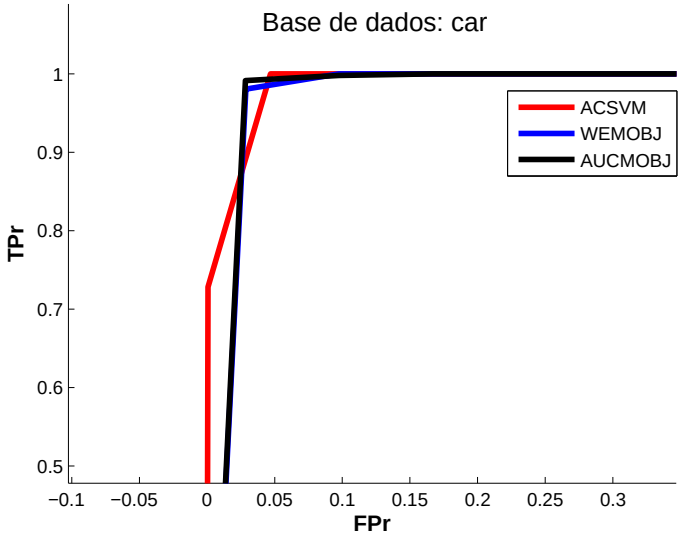


Figura C.16: Curvas ROC médias para a base de dados car.

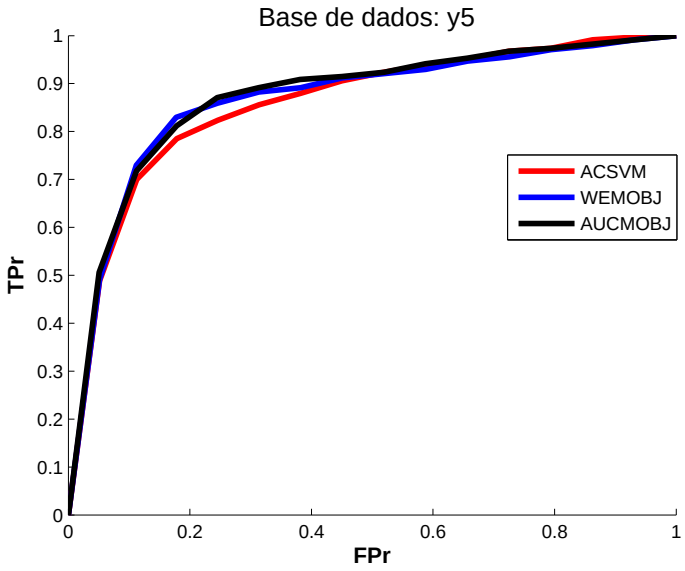


Figura C.17: Curvas ROC médias para a base de dados y5.

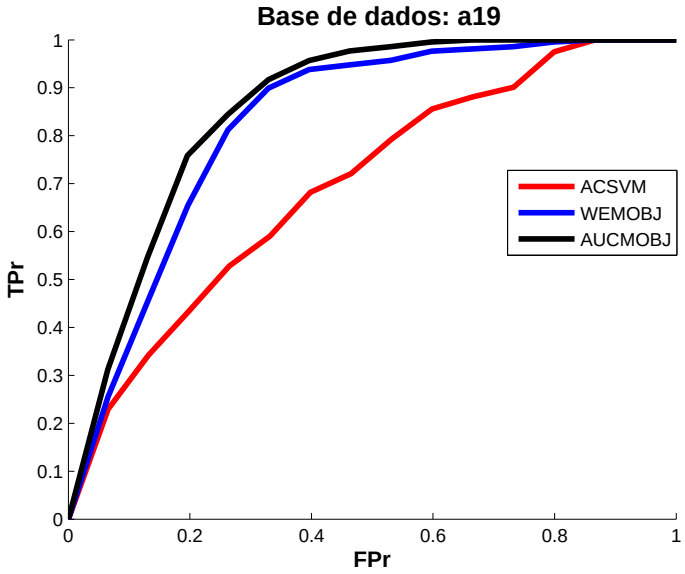


Figura C.18: Curvas ROC médias para a base de dados a19.

Referências Bibliográficas

Agarwal, S., Graepel, T., Herbrich, R., Har-Peled, S. & Roth, D. (2005). Generalization bounds for the area under the roc curve. *J. Mach. Learn. Res.*, **6**, 393–425. 100

Akbani, R., Kwek, S. & Japkowicz, N. (2004). Applying support vector machines to imbalanced datasets. In *Proceedings of European Conference on Machine Learning*, 39–50. 33

Alejo, R., Garcia, V., Sotoca, J., Mollineda, R. & Sánchez, J. (2006). Improving the classification accuracy of rbf and mlp neural networks trained with imbalanced samples. In *Intelligent Data Engineering and Automated Learning IDEAL 2006*, vol. 4224 of *Lecture Notes in Computer Science*, 464–471, Springer Berlin / Heidelberg. 29

Anand, R., Mehrotra, K., Mohan, C. & Ranka, S. (1993). An improved algorithm for neural network classification of imbalanced training sets. *IEEE Transactions on Neural Networks, 6(4):962-969*. 47

Asuncion, A. & Newman, D. (2007). UCI machine learning repository. 75

Barandela, R., Valdovinos, R.M., Sánchez, J.S. & Ferri, F.J. (2004). The imbalanced training sample problem: Under or over sampling? In *Structural, Syntactic, and Statistical Pattern Recognition*, vol. 3138 of *Lecture Notes in Computer Science*, 806–814, Springer Berlin / Heidelberg. 29, 77, 78, 79

Barron, A. (1993). Universal approximation bounds for superpositions of a sigmoidal function. *IEEE Transactions on Information Theory*, **39**, 930–945. 71

Bartlett, P.L. (1997). For valid generalization the size of the weights is more important than the size of the network. In *Advances in Neural Information Processing Systems*, vol. 9. 71

Batista, G.E., Prati, R.C. & Monard, M.C. (2005). Balancing strategies and class overlapping. In *Advances in Intelligent Data Analysis VI*, vol. 3646 of *Lecture Notes in Computer Science*, 24–35, Springer Berlin / Heidelberg. 29

Batista, G.E.A.P.A., Prati, R.C. & Monard, M.C. (2004). A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explor. Newsl.*, **6**, 20–29. 7, 29, 77, 78, 79

Berger, J. (1985). *Statistical Decision Theory and Bayesian Analysis*. Springer, 2nd edn. 4, 7, 11, 12

Bishop, C.M. (1995). *Neural Networks for Pattern Recognition*. Oxford University Press, 1st edn. 11, 18, 50

Bishop, C.M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer. 11, 101

Bland, R.G., Goldfarb, D. & Todd, M.J. (1980). The ellipsoid method: A survey. Tech. rep., Ithaca, NY, USA. 72

Bradley, A.P. (1997). The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, **30**, 1145–1159. 22, 62

Braga, A.P., Horta, E.G., Natowicz, R., Rouzier, R., Incitti, R., Rodrigues, T.S., Costa, M.A., Pataro, C.D.M. & Çela, A. (2008). Bayesian classifiers for predicting the outcome of breast cancer preoperative chemotherapy. In *ANNPR*, vol. 5064 of *Lecture Notes in Computer Science*, 263–266, Springer. 2

Breiman, L., Friedman, J., Stone, C.J. & Olshen, R.A. (1984). *Classification and Regression Trees*. Chapman & Hall/CRC. 29

Carvalho, A., Pozo, A., Vergilio, S. & Lenz, A. (2008). Predicting fault proneness of classes trough a multiobjective particle swarm optimization algorithm. In *Proceedings of the 2008 20th IEEE International Conference on Tools with Artificial Intelligence - Volume 02*, 387–394, IEEE Computer Society. 3

Castro, C. & Braga, A. (2008). Optimization of the area under the roc curve. In *Proceedings of the 10th Brazilian Symposium on Neural Networks (SBRN '08)*, 141–146, IEEE Computer Society, Washington, DC, USA. 4

Castro, C. & Braga, A. (2009). Artificial neural networks learning in roc space. In *Proc. of the International Joint Conference on Computational Intelligence (IJCCI'09)*, 484–489. 4

Castro, C. & Braga, A. (2010). Aprendizado de redes mlp através da otimização da área abaixo da curva roc. In *XVIII Congresso Brasileiro de Automática (CBA2010)*, 4445–4451. 4

Castro, C. & Braga, A. (2011a). Supervised learning with imbalanced data sets. *Revista Controle & Automacao, "accepted for publication"*. 4

Castro, C. & Braga, A. (2011b). Using prior information to improve the mlps performance on imbalanced data. *"submitted to journal"*. 4

Castro, C.L., Carvalho, M.A. & Braga, A.P. (2009). An improved algorithm for svms classification of imbalanced data sets. In *Engineering Applications of Neural Networks*, vol. 43 of *Communications in Computer and Information Science*, 108–118, Springer Berlin Heidelberg. 29

Chang, C.C. & Lin, C.J. (2011). LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, **2**, 27:1–27:27, software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`. 87

Chawla, N.V., Bowyer, K.W. & Kegelmeyer, P.W. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, **16**, 321–357. 28, 29, 79

Chen, S., Wang, X., Hong, X. & Harris, C. (2006). Kernel classifier construction using orthogonal forward selection and boosting with fisher ratio class separability measure. *IEEE Transactions on Neural Networks*, **17**, 1652–1656. 37

Chen, S., He, H. & Garcia, E.A. (2010). Ramoboost: ranked minority oversampling in boosting. *IEEE Trans. on Neural Networks*, **21**, 1624–1642. 30, 77, 78, 79, 86, 95

Cherkassky, V. & Mulier, F. (2007). *Learning from data*. John Wiley and Sons, 2nd edn. 8, 10, 25, 27

Cortes, C. & Mohri, M. (2004). Auc optimization vs. error rate minimization. In *Advances in Neural Information Processing Systems 16*, MIT Press, Cambridge, MA. 62, 64, 65, 74

Cortes, C. & Vapnik, V. (1995). Support-vector networks. *Mach. Learn.*, **20**, 273–297. 31, 34, 87, 91

Cristianini, N., Kandola, J., Elisseeff, A. & Shawe-Taylor, J. (2002). On kernel-target alignment. In *Advances in Neural Information Processing Systems 14*, vol. 14, 367–373. 37

Demsar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, **7**, 1–30. 82, 83, 84, 90

Drummond, C. & Holte, R. (2003). C4.5, class imbalance, and cost sensitivity: Why under-sampling beats over-sampling. In *Working Notes of the ICML Workshop Learning from Imbalanced Data Sets*. 28

Duda, R.O., Hart, P.E. & Stork, D.G. (2000). *Pattern Classification (2nd Edition)*. Wiley-Interscience. 4, 7, 11, 12, 25, 27, 44, 52, 102, 104

Egan, J.P. (1975). *Signal Detection Theory and ROC Analysis*. Academic Press. 24

Elkan, C. (2001). The foundations of cost-sensitive learning. In *Proceedings of the Seventeenth International Joint Conference on Artificial Intelligence, IJCAI*, 973–978. 38, 39

Everson, R.M. & Fieldsend, J.E. (2006a). Multi-class roc analysis from a multi-objective optimisation perspective. *Pattern Recogn. Lett.*, **27**, 918–927. 41

Everson, R.M. & Fieldsend, J.E. (2006b). Multi-objective optimisation for receiver operating characteristic analysis. In *Multi-Objective Machine Learning*, 533–556. 3, 101

Fan, W., Stolfo, S.J., Zhang, J. & Chan, P.K. (1999). Adacost: misclassification cost-sensitive boosting. In *Proceedings of IEEE International Conference on Machine Learning*, 97–105, Morgan Kaufmann. 30

Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recogn. Lett.*, **27**, 861–874. 22, 24, 25, 26, 42, 77, 113

Fawcett, T. & Provost, F. (1997). Adaptive fraud detection. *Data Min. Knowl. Discov.*, **1**, 291–316. 3

Ferri, C., Flach, P.A. & Hernández-Orallo, J. (2002). Learning decision trees using the area under the roc curve. In *ICML '02: Proceedings of the Nineteenth International Conference on Machine Learning*, 139–146, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 62

Freund, Y. & Schapire, R.E. (1997). A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, **55**, 119–139. 30, 100

Freund, Y., Iyer, R., Schapire, R.E. & Singer, Y. (2003). An efficient boosting algorithm for combining preferences. *J. Mach. Learn. Res.*, **4**, 933–969. 62

Friedman, M. (1937). The use of ranks to avoid the assumption of normality implicit in the analysis of variance. *Journal of the American Statistical Association*, **32**, 675–701. 82

Friedman, M. (1940). A comparison of alternative tests of significance for the problem of m rankings. *The Annals of Mathematical Statistics*, **11**, 86–92. 82

Gao, Y., Wang, S. & Liu, Z. (2009). Automatic fault detection and diagnosis for sensor based on kpca. In *Proceedings of International Symposium on the Computational Intelligence and Design*, 135–138, IEEE Computer Society. 3

Geman, S., Bienenstock, E. & Doursat, R. (1992). Neural networks and the bias/variance dilemma. *Neural Computation*, **4**, 1–58. 98

Girosi, F., Jones, M. & Poggio, T. (1995). Regularization theory and neural networks architectures. *Neural Comput.*, **7**, 219–269. 71

Graening, L., Jin, Y. & Sendhoff, B. (2006). Generalization improvement in multi-objective learning. In *International Joint Conference on Neural Networks*, 9893–9900, IEEE Press. 41

Hagan, M.T. & Menhaj, M.B. (1994). Training feedforward networks with the marquardt algorithm. *IEEE Trans. on Neural Networks*, **5**, 989–993. 47, 48

Han, H., Wang, W.Y. & Mao, B.H. (2005). Borderline-smote: A new over-sampling method in imbalanced data sets learning. In *Advances in Intelligent Computing*, vol. 3644 of *Lecture Notes in Computer Science*, 878–887, Springer Berlin, Heidelberg. 29

Hand, D. & Till, R. (2001). A simple generalisation of the area under the roc curve for multiple class classification problems. *Mach. Learn.*, **45**, 171–186. 101

Hanley, J.A. & Mcneil, B.J. (1982). The meaning and use of the area under a receiver operating characteristic (roc) curve. *Radiology*, **143**, 29–36. 27, 62

Haykin, S. (1994). *Neural Networks: A Comprehensive Foundation*. Macmillan, New York. 2, 15, 54, 70

He, H. & Garcia, E.A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, **21**, 1263–1284. 1, 23, 28, 29

He, H. & Shen, X. (2007). A ranked subspace learning method for gene expression data classification. In *Proceedings of the 2007 International Conference on Artificial Intelligence, ICAI 2007, Volume I, June 25-28, 2007, Las Vegas, Nevada, USA*, 358–364. 20

He, H., Bai, Y., Garcia, E.A. & Li, S. (2008). Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *Proceedings of the International Joint Conference on Neural Networks, IJCNN 2008*, 1322–1328. 29

Herschtal, A. & Raskutti, B. (2004). Optimising area under the roc curve using gradient descent. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, 49, ACM, New York, NY, USA. 62

Herschtal, A., Raskutti, B. & Campbell, P.K. (2006). Area under roc optimisation using a ramp approximation. In *Proceedings of the Sixth SIAM International Conference on Data Mining*, 1–11. 62

Hong, X., Chen, S. & Harris, C. (2007). A kernel-based two-class classifier for imbalanced data sets. *IEEE Transactions on Neural Networks*, **18**, 28–41. 3, 37

Huang, Y.M., Hung, C.M. & Jiau, H.C. (2006). Evaluation of neural networks and data mining methods on a credit assessment task for class imbalance problem. *Nonlinear Analysis: Real World Applications*, **7**, 720 – 747. 29

Japkowicz, N. (2000). Learning from imbalanced data sets: A comparison of various strategies. In *AAAI Conference on Artificial Intelligence*, 10–15, AAAI Press. 29

Japkowicz, N. (2001). Supervised versus unsupervised binary-learning by feedforward neural networks. *Mach. Learn.*, **42**, 97–122. 30

Japkowicz, N. & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intell. Data Anal.*, **6**, 429–449. 7, 14, 97

JOACHIMS, T. (2002). *Learning to Classify Text Using Support Vector Machines: Methods, Theory and Algorithms*. Kluwer Academic Publishers, Norwell, MA, USA. 32, 87, 88

JOACHIMS, T. (2005). A support vector method for multivariate performance measures. In *ICML '05: Proceedings of the 22nd international conference on Machine learning*, 377–384, ACM, New York, NY, USA. 62

KANDOLA, J. & SHAWE-TAYLOR, J. (2003). Refining kernels for regression and uneven classification problems. In *Proceedings of International Conference on Artificial Intelligence and Statistics*, Springer-Verlag, Berlin Heidelberg. 37

KARAKOULAS, G. & SHAWE-TAYLOR, J. (1999). Optimizing classifiers for imbalanced training sets. In *Proceedings of Conference on Advances in Neural Information Processing Systems II*, 253–259, MIT Press, Cambridge, MA, USA. 33

KHOSHGOFTAAR, T.M., HULSE, J.V. & NAPOLITANO, A. (2010). Supervised neural network modeling: An empirical investigation into learning from imbalanced data with labeling errors. *IEEE Trans. on Neural Networks*, **21**, 813–830. 7, 15, 29, 78, 79, 97

KUBAT, M. & MATWIN, S. (1997). Addressing the curse of imbalanced training sets: one-sided selection. In *Proc. 14th International Conference on Machine Learning*, 179–186, Morgan Kaufmann. 28, 29

KUBAT, M., HOLTE, R.C. & MATWIN, S. (1998). Machine learning for the detection of oil spills in satellite radar images. *Machine Learning*, **30**, 195–215. 20, 24, 77

KUKAR, M. & KONONENKO, I. (1998). Cost-sensitive learning with neural networks. In *Proceedings of the 13th European Conference on Artificial Intelligence (ECAI-98)*, 445–449, John Wiley and Sons. 38, 40

KUPINSKI, M.A. & ANASTASIO, M.A. (1999). Multiobjective genetic optimization of diagnostic classifiers with implications for generating receiver operating

characterisitic curves. *IEEE Transactions on Medical Imaging*, **18**, 675–685. 41

Lan, J., Hu, M.Y., Patuwo, E. & Zhang, G.P. (2010). An investigation of neural network classifiers with unequal misclassification costs and group sizes. *Decis. Support Syst.*, **48**, 582–591. 29

Lasko, T.A., Bhagwat, J.G., Zou, K.H. & Ohno-Machado, L. (2005). The use of receiver operating characteristic curves in biomedical informatics. *Journal of Biomedical Informatics*, **38**, 404–415. 25

Lawrence, S., Burns, I., Back, A.D., Tsoi, A.C. & Giles, C.L. (1998). Neural network classification and prior class probabilities. In *Neural Networks: Tricks of the Trade, this book is an outgrowth of a 1996 NIPS workshop*, 299–313, Springer-Verlag, London, UK. 7, 97

Li, Y. & Shawe-Taylor, J. (2003). The svm with uneven margins and chinese document categorization. In *Proceedings of the 17th Pacific Asia Conference on Language, Information and Computation*, 216–227. 3, 34

Lin, Y., Lee, Y. & Wahba, G. (2002). Support vector machines for classification in nonstandard situations. *Mach. Learn.*, **46**, 191–202. 31, 32, 87, 88, 91

Liu, X.Y., Wu, J. & Zhou, Z.H. (2009). Exploratory undersampling for class-imbalance learning. *IEEE Trans. on Sys. Man Cyber. Part B*, **39**, 539–550. 29

Luenberger, D. (1984). *Linear and Nonlinear Programming*. Addison-Wesley, Reading, 2nd edn. 48, 69, 111

Maloof, M.A. (2003). Learning when data sets are imbalanced and when costs are unequal and unknown. In *Proceedings of the International Conf. Machine Learning, Workshop on Learning from Imbalanced Data Sets II*. 22

Manevitz, L. & Yousef, M. (2007). One-class document classification via neural networks. *Neurocomput.*, **70**, 1466–1481. 3

Mann, H.B. & Whitney, D.R. (1947). On a test wheter one of two random variables is stochastically larger than the other. *Annals of Math. Statistics, 18, pgs. 50 - 60.*. 63

MATLAB (2010). *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts. 83

Mazurowski, M.A., Habas, P.A., Zurada, J.M., Lo, J.Y., Baker, J.A. & Tourassi, G.D. (2008). Training neural network classifiers for medical decision making: The effects of imbalanced datasets on classification performance. *Neural Networks*, **21**, 427–436. 29

Mease, D., Wyner, A.J. & Buja, A. (2007). Boosted classification trees and class probability/quantile estimation. *J. Mach. Learn. Res.*, **8**, 409–439. 28, 29

Monard, M. & Batista, G. (2002). Learning with skewed class distribution. In *Advances in Logic, Artificial Intelligence and Robotics*, 173–180, IOS Press. 1

Morik, K., Brockhausen, P. & Joachims, T. (1999). Combining statistical learning with a knowledge-based approach - a case study in intensive care monitoring. In *Proceedings of the Sixteenth International Conference on Machine Learning*, 268–277, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 3, 32

Moturu, S.T., Johnson, W.G. & Liu, H. (2010). Predictive risk modelling for forecasting high-cost patients: a real-world application using medicaid data. *International Journal of Biomedical Engineering and Technology*, **2**, 114–132. 2

Muller, K.R., Mika, S., Ratsch, G., Tsuda, K. & Scholkopf, B. (2001). An introduction to kernel-based learning algorithms. *IEEE Trans. on Neural Networks*, **12**, 181–201. 31

Nemenyi, P. (1963). *Distribution-free multiple comparisons*. Ph.D. thesis, Princeton University. 83

Oh, S.H. (2011). Error back-propagation algorithm for classification of imbalanced data. *Neurocomputing*, **74**, 1058–1061. 40

Pearson, P., Goney, G. & Shwaber, J. (2003). Imbalanced clustering for microarray time-series. In *Proc. 20th International Conference on Machine Learning (ICML'03)*. 20

Prati, R., Batista, G. & Monard, M. (2008). Evaluating classifiers using roc curves. *Latin America Transactions, IEEE (Revista IEEE America Latina)*, **6**, 215 –222. 24

Prati, R.C., Batista, G.E.A.P.A. & Monard, M.C. (2004). Class imbalances versus class overlapping: An analysis of a learning system behavior. In *MICAI 2004: Advances in Artificial Intelligence, Third Mexican International Conference on Artificial Intelligence*, vol. 2972 of *Lecture Notes in Computer Science*, 312–321, Springer. 7, 14, 97

Provost, F. & Fawcett, T. (2001). Robust classification for imprecise environments. *Mach. Learn.*, **42**, 203–231. 27

Provost, F.J., Fawcett, T. & Kohavi, R. (1998). The case against accuracy estimation for comparing induction algorithms. In *ICML '98: Proceedings of the Fifteenth International Conference on Machine Learning*, 445–453, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 22

Raskutti, B. & Kowalczyk, A. (2004). Extreme re-balancing for svms: a case study. *SIGKDD Explor. Newsl.*, **6**, 60–69. 30

Rudin, C. & Schapire, R.E. (2009). Margin-based ranking and an equivalence between AdaBoost and RankBoost. *Journal of Machine Learning Research*, **10**, 2193–2232. 100

Rumelhart, D.E. & McClelland, J.L. (1986). *Parallel distributed processing: Explorations in the microstructure of cognition*, vol. 1: Foundations. MIT Press. 2, 38, 49, 109

Sanchez, M.S., Ortiz, M.C., Sarabia, L.A. & Lleti, R. (2005). On pareto-optimal fronts for deciding about sensitivity and specificity in class-modelling problems. *Analytica Chimica Acta*, **544**, 236 – 245. 41

Schölkopf, B., Platt, J.C., Shawe-Taylor, J.C., Smola, A.J. & Williamson, R.C. (2001). Estimating the support of a high-dimensional distribution. *Neural Comput.*, **13**, 1443–1471. 30

Sheskin, D.J. (2007). *Handbook of Parametric and Nonparametric Statistical Procedures*. Chapman & Hall/CRC, 4th edn. 83, 89

Silva, C., Silva, A., Netto, S., Paiva, A., Junior, G. & Nunes, R. (2009). Lung nodules classification in ct images using simpsons index, geometrical measures and one-class svm. In *Machine Learning and Data Mining in Pattern Recognition*, vol. 5632 of *Lecture Notes in Computer Science*, 810–822, Springer Berlin / Heidelberg. 2

Souza, M.R.P., Cavalcanti, G.D.C. & Tsang, I.R. (2010). Off-line signature verification: An approach based on combining distances and one-class classifiers. In *Proceedings of the 22nd IEEE International Conference on Tools with Artificial Intelligence, ICTAI 2010, Arras, France*, 7–11, IEEE Computer Society. 3

Spackman, K.A. (1989). Signal detection theory: valuable tools for evaluating inductive learning. In *Proceedings of the sixth international workshop on Machine learning*, 160–163, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. 24

Sun, Y., Kamel, M.S., Wong, A.K.C. & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition*, **40**, 3358–3378. 3, 22, 23, 24, 30

Swets, J.A., Dawes, R.M. & Monahan, J. (2000). Better decisions through science. *Scientific American*, **283**, 82–87. 24

Tan, P.N., Steinbach, M. & Kumar, V. (2005). *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA. 23

Tang, Y., Zhang, Y.Q., Chawla, N.V. & Krasser, S. (2009). Svms modeling for highly imbalanced classification. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, **39**, 281–288. 33, 75, 77, 87, 88, 95

Teixeira, R., Braga, A., Takahashi, R. & Saldanha, R. (2000). Improving generalization of mlps with multi-objective optimization. *Neurocomputing*, **35**, 189–194. 5, 16, 45, 70, 71, 72, 73

Ting, K.M. (2000). A comparative study of cost-sensitive boosting algorithms. In *Proceedings of the 17th International Conference on Machine Learning*, 983–990, Morgan Kaufmann. 30

Tomek, I. (1976). Two modifications of cnn. *IEEE Trans. Systems, Man, and Cybernetics*, **6**, 769–772. 29

Van Gestel, T., Suykens, J.A.K., Baesens, B., Viaene, S., Vanthienen, J., Dedene, G., De Moor, B. & Vandewalle, J. (2004). Benchmarking least squares support vector machine classifiers. *Mach. Learn.*, **54**, 5–32. 78, 87

Vapnik, V. (1998). *Statistical Learning Theory*. Wiley-Interscience. 70, 71, 101

Vapnik, V.N. (1995). *The nature of statistical learning theory*. Springer-Verlag New York, Inc. 4, 7, 8, 11, 12, 17, 18, 70, 71, 98

Vapnik, V.N. (1999). An overview of statistical learning theory. *IEEE Trans. on Neural Networks*, **10**, 988–999. 4, 7, 8, 70

Weiss, G.M. (2004). Mining with rarity: a unifying framework. *SIGKDD Explor. Newsl.*, **6**, 7–19. 7, 29, 97

Weiss, G.M. & Provost, F. (2003). Learning when training data are costly: The effect of class distribution on tree induction. *Journal of Artificial Intelligence Research*, **19**, 315–354. 7

WILCOXON, F. (1945). Individual comparisons by ranking methods. *Biometrics, 1, pages 80 - 83.*. 63

WILSON, D. (1972). Asymptotic properties of nearest neighbor rules using edited data. *IEEE Trans. Systems, Man, and Cybernetics*, **2**, 408–421. 29

WU, G. & CHANG, E.Y. (2003). Adaptive feature-space conformal transformation for imbalanced-data learning. In *Proceedings of IEEE International Conference on Machine Learning*, 816–823. 7, 33, 34, 35

WU, G. & CHANG, E.Y. (2005). Kba: Kernel boundary alignment considering imbalanced data distribution. *IEEE Transactions on Knowledge and Data Engineering*, **17**, 786–795. 34, 36, 37, 75, 77

YAN, L., DODIER, R.H., MOZER, M. & WOLNIEWICZ, R.H. (2003). Optimizing classifier performance via an approximation to the wilcoxon-mann-whitney statistic. In *ICML '03: Proceedings of the twenty international conference on Machine learning*, 848–855. 66, 67, 68

ZHANG, J. & MANI, I. (2003). Knn approach to unbalanced data distributions: A case study involving information extraction. In *Proceedings of the ICML'2003 workshop on learning from imbalanced datasets*. 29

ZHOU, Z.H. & LIU, X.Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering*, **18**, 63–77. 29