Pós-Graduação em Ciência da Computação

"CLASSIFICAÇÃO COM EXEMPLOS DE UMA ÚNICA CLASSE BASEADA NA BUSCA PELOS LIMITES DAS CARACTERÍSTICAS DO PROBLEMA"

Por

GEORGE GOMES CABRAL

Tese de Doutorado

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE, SETEMBRO/2014

Universidade Federal de Pernambuco
Centro de Informática
Pós-graduação em Ciência da Computação

George Gomes Cabral

"Classificação com Exemplos de Uma Única Classe Baseada na Busca pelos Limites das Características do Problema"

Trabalho apresentado ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Doutor em Ciência da Computação.

Orientador: *Adriano Lorena Inácio de Oliveira*

Recife
2014

Catalogação na fonte
Bibliotecária Alice Maria dos Santos Costa CRB4-711

C117c Cabral, George Gomes.
Classificação com exemplos de uma única classe baseada na busca pelos limites das características do problema / George Gomes Cabral. – Recife: O Autor, 2014.
109 f.: il., fig., tab.

Orientador: Adriano Lorena Inácio de Oliveira.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIN. Ciência da Computação, 2014.
Inclui referências e apêndices.

1. Inteligência artificial. 2. Sistemas de reconhecimento de padrões. I. Adriano Lorena Inácio de Oliveira (Orientador). II. Titulo.

006.3 CDD (22. ed.) UFPE-MEI 2014-139

Tese de Doutorado apresentada por **George Gomes Cabral** à Pós Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco, sob o título "**Classificação com Exemplos de Uma Única Classe Baseada na Busca pelos Limites das Características do Problema**" orientada pelo **Prof. Adriano Lorena Inácio de Oliveira** e aprovada pela Banca Examinadora formada pelos professores:

_____

Prof. Francisco de Assis Tenório de Carvalho
Centro de Informática / UFPE

_____

Profa. Renata Maria Cardoso Rodrigues de Souza
Centro de Informática / UFPE

_____

Prof. George Darmiton da Cunha Cavalcanti
Centro de Informática / UFPE

_____

Prof. Carmelo Jose Albanez Bastos Filho
Escola Politecnica de Pernambuco / UPE

_____

Prof. Andre Carlos Ponce de Leon Ferreira de Carvalho
Instituto de Ciências Matemáticas e de Computação /USP

Visto e permitida a impressão.
Recife, 05 de agosto de 2014.

_____

Profa. Edna Natividade da Silva Barros
Coordenadora da Pós-Graduação em Ciência da Computação do
Centro de Informática da Universidade Federal de Pernambuco.

Dedico este trabalho aos meus pais José Gomes dos Santos e Juraci Gomes Cabral que fizeram todo o esforço possível para educar seus filhos.

Agradecimentos

Inicialmente, agradeço a Deus pela minha saúde e das pessoas que comigo convivem.

Agradeço a meus pais e irmãs pelo incentivo, apoio e carinho nos momentos difíceis dessa conquista. Agradeço também à minha namorada Tássia pelo apoio. Sem vocês isso não seria possível.

Agradeço ao meu orientador, Adriano Lorena, por ter acreditado e confiado a mim essa imensa responsabilidade. Agradeço também por seu apoio incondicional e amizade.

Agradeço à FACEPE (Fundação de Amparo à Ciência e Tecnologia do Estado de Pernambuco) pelo suporte financeiro sem o qual não seria possível a realização dessa conquista.

Agradeço a todos meus amigos que durante esses quatro anos de alguma forma, mesmo que indiretamente, foram importantes para a obtenção desse sonho.

Por fim, agradeço a todos que comigo conviveram durante esses período, pois direta ou indiretamente, todos contribuíram para o sucesso dessa conquista.

A todos, Obrigado

Resumo

A detecção de novidades é um problema com um grande número de aplicações. Em algumas aplicações, o foco está na prevenção ou detecção de estados indesejados. Em alguns casos, esses estados não são conhecidos durante o treinamento do modelo de classificador; em outros, como monitoramento de máquinas, por exemplo, uma quebra da máquina pode ser bem rara e exemplos desse caso podem ser bastante raros. Nestes casos, a abordagem mais aceita consiste em se modelar o comportamento normal do sistema de forma a, no futuro, se detectar eventos desconhecidos. Esse é o conceito básico de Classificação com Exemplos de uma Única Classe (*One-Class Classification* - OCC).

Esta tese introduz duas versões de um método simples e efetivo para OCC, chamado de FBDOCC (*Feature Boundaries Detector for One-Class Classification*). O FBDOCC funciona analisando cada característica (dimensão) do problema e criando uma representação sintética da classe novidade (desconhecida *a priori*) que engloba os dados da classe normal. Esse trabalho também considera o uso do algoritmo *Particle Swarm Optimization* (PSO) na busca da melhor configuração dos parâmetros do método proposto. Além disso, o presente trabalho introduz também um procedimento para a melhoria do custo computacional durante o treinamento, da técnica proposta, sem que haja a degradação na qualidade da classificação. Entre as motivações por trás deste trabalho, estão a criação de um método com baixo custo computacional e com a mesma ou melhor precisão na classificação que métodos para detecção de novidades do estado da arte.

Vários experimentos foram executados com bases de dados do mundo real e artificiais no intuito de comparar as duas versões desenvolvidas do método proposto com alguns dos mais recentes e efetivos métodos OCC, são eles: *Support Vector Data Description* (SVDD), *One-Class* SVM (OCSVM), *Least Squares One-class* SVM (LSOCSVM), *Kernel Principal Component Analysis* (KPCA), *Gaussian Process Prior* OCC (GP-OCC), *Condensed Nearest Neighbor Data Description* (CNNDD) e *One-class Random Forests* (OCRF). As métricas de desempenho consideradas nos experimentos foram: (i) a area sob a curva ROC (*Area Under the Curve* - AUC); (ii) o coeficiente de correlação de Matthews (*Matthews Correlation Coefficient* - MCC); (iii) o tempo de treinamento; e (iv) a taxa de redução de protótipos. Em relação às métricas AUC e MCC, a primeira versão do método FBDOCC apresentou a melhor média global entre todos os métodos enquanto que a segunda versão do método proposto, FBDOCC2, obteve resultados comparáveis aos melhores métodos em experimentos onde o FBDOCC obteve um baixo desempenho. O FBDOCC obteve os melhores resultados considerando o tempo de treinamento em todas as bases de dados, exceto uma. Em adição, o FBDOCC foi bem mais rápido que todos os métodos baseados em Máquinas de Vetores de Suporte.

Além disso, um estudo de caso foi realizado utilizando dados adquiridos em um hospi-

tal local de renome. Estes dados são compostos de informações não-invasivas sobre as crianças que compareceram ao hospital com sintomas de sopro no coração. Informações como idade, peso, altura, etc., foram usadas para prever se a criança é ou cardiopata. Devido ao elevado grau de desequilíbrio entre as classes (ou seja, o número de pacientes saudáveis foi consideravelmente mais elevado), a abordagem adotada foi a de construir uma descrição dos casos saudáveis deixando casos desconhecidos fora desta descrição. Os resultados mostram que dois dos classificadores OCC aplicados (FBDOCC e OCSVM) obtiveram êxito nesta tarefa, resultando na melhor taxa, entre os métodos investigados, de detecção baseada exclusivamente em dados não-invasivos.

Palavras Chave: Classificação com Exemplos de uma Única Classe. Detecção de Anomalias. Detecção de Novidades. Regra do Vizinho Mais Próximo. Redução de Protótipos.

Abstract

Novelty detection is a problem with a large number of relevant applications. For some applications, the main interest is the prevention or detection of undesired states. In some cases, these undesired states are not known in advance; in others, such as machine monitoring, for instance, machine breakdown may be very rare and examples of this event may be very rare. In such cases, the most widely accepted approach is to model the normal behavior of the system in order to subsequently detect unknown events. This is the basic concept of One-Class Classification (OCC).

This work introduces two versions of an effective and straightforward method for OCC, referred to as FBDOCC (Feature Boundaries Detector for One-Class Classification), which works by analyzing every feature dimension of the problem and building an artificial representation of the unknown (novelty class) which encompasses the normal data. Our work, also considers the use of the Particle Swarm Optimization (PSO) algorithm to find the best parameter configuration of the proposed method. Furthermore, it is also introduced a procedure to improve the training time performance without degrading the quality of the classification. Among the motivations behind this work, are the creation of a method of low computational cost and with the same or better classification quality than state of art novelty detection methods, such as methods based on Support Vector Machines.

A number of experiments were carried out with synthetic and real world datasets aiming at comparing both versions of the FBDOCC method with the most recent and effective OCC methods, namely: Support Vector Data Description (SVDD), One-Class Support Vector Machines (OCSVM), Least Squares One-Class One-Class Support Vector Machines (LSOCSVM), Kernel Principal COmponent Analysis (KPCA), Gaussian Process Prior OCC (GP-OCC), Condensed Nearest Neighbor Data Description (CNNDD) and One-class Random Forests (OCRF). The metrics considered for evaluation of the results of the experiments are: (i) the Area Under the ROC Curve (AUC); (ii) the Matthews Correlation Coefficient (MCC); (iii) the training time; and (iv) the prototype reduction rate. Regarding the metrics AUC and MCC, the first FBDOCC version has presented the best overall results among all the methods whereas the FBDOCC2 ob- tained results comparable to the best methods in some experiments where the standard FBDOCC yielded a poorer performance. FBDOCC was the faster method to train in comparison to the other methods in all but one data set. In addition, FBDOCC was much faster than all methods based on support vector machines.

Additionally, a case study was performed using data acquired in a reputed local hospital. These data are composed of non invasive information about children who attended the hospital with symptons of heart murmur. Information such as age, weight, height, etc. were used to predict whether or not the child is cardiac. Due the high level of unbalancing between the clas-

ses (*i.e.*, the number of healthy patients was considerably higher), the adopted approach was to build a description of the healthy cases leaving unknown cases out of this description. The results show that two of the applied OCC classifiers (FBDOCC and OCSVM) succeeded in this task, yielding a good detection rate, based solely on noninvasive data.

Keywords: One-Class Classification. Anomaly Detection. Novelty Detection. Nearest Neighbor Rule. Prototype Reduction.

Lista de Figuras

Lista de Tabelas

Lista de Acrônimos

AUC	*Area Under the Curve*
CDF	*Cumulative Density Function*
CNN	*Condensed Nearest Neighbor*
CNNDD	*Condensed Nearest Neighbor Data Description*
ENN	*Edited Nearest Neighbor*
FBDOCC	*Feature Boundaries Detector for One-Class Classification*
FN	Falso Negativo
FP	Falso Positivo
GP-OCC	*Gaussian Process for* OCC
IBL	*Instance Based Learning*
IMC	Índice de Massa Corpórea
KS	Kolmogorov-Smirnov
LS-OCSVM	*Least Squares One-Class* SVM
LVQ	*Learning Vector Quantization*
MCC	*Mathews Correlation Coefficient*
NN	*Nearest Neighbor*
NNDD	*Nearest Neighbor Data Description*
NNSRM	*Nearest Neighbor Structural Risk Minimization*
OCC	*One-Class Classification*
OCRF	*One-Class Random Forest*
OCSVM	*One-Class* SVM
PCA	*Principal Component Analysis*
PD	Probabilidade de Detecção

PDF *Probability Density Function*

PFA Probabilidade de Falso Alarme

PNN *Prototypes for Nearest Neighbor*

PRS *Prototype Reduction Scheme*

PSO *Particle Swarm Optimization*

PSS *Prototype Selection Scheme*

RBF *Radial Basis Function*

RISE *Rule Induction from a Set of Exemplars*

RNN *Reduced Nearest Neighbor*

ROC *Receiver Operating Characteristic*

SC Superfície Corpórea

SNN *Selective Nearest Neighbor*

SRM *Structural Risk Minimization*

SVDD *Support Vector Data Description*

SVM *Support Vector Machine*

VN Verdadeiro Negativo

VP Verdadeiro Positivo

Sumário

1

Introdução

Em muitos problemas de classificação não existem regras explícitas para distinção entre objetos, porém exemplos desses objetos podem ser obtidas facilmente. Em reconhecimento de padrões, ou aprendizado de máquinas, tenta-se criar um modelo de classificador a partir de um conjunto limitado de exemplos de treinamento. A meta é obter classificadores capazes de predizer a classe de objetos desconhecidos durante a fase de treinamento.

O paradigma de Aprendizado Baseado em Instâncias (*Instance Based Learning* - IBL) trabalha armazenando instâncias de treinamento na memória como pontos no espaço n-dimensional; esses pontos são definidos pelos n atributos que descrevem o problema. Dessa forma, um modelo de classificador é então representado por um conjunto de pontos. Para que o espaço n-dimensional forneça informações relevantes à distinção entre as classes, o aspecto mais importante consiste na métrica que define a similaridade entre os pontos. As diversas formas de uso do IBL utilizam a técnica do "Vizinho mais Próximo" (*Nearest Neighbor* - NN) (DUDA; HART; STORK, 2001) para classificar novas instâncias (AHA, 1992). Teoricamente, a variante kNN do algoritmo NN resulta em uma solução sub-ótima, porém, bastante popular na prática (THEODORIDIS; KOUTROUMBAS, 2006). Com a sofisticação dos algoritmos dessa família, mecanismos para solucionar alguns dos problemas evidenciados em domínios reais e inerentes ao método kNN básico são incorporados, tais como o tratamento de ruídos nos dados (SAEZ; LUENGO; HERRERA, 2013), a redução de protótipos (TRIGUERO et al., 2012) e a geração de protótipos (TRIGUERO et al., 2011). Desses, os Esquemas de Redução de Protótipos (*Prototype Reduction Scheme* - PRS) concentram a maior parte dos mecanismos investigados para métodos IBL (GARCIA et al., 2012). Reduzir o conjunto de treinamento invocando um esquema de redução de protótipos resulta em pelo menos duas melhorias: (i) redução no número de exemplos no espaço de características (que aumenta a velocidade de busca por protótipos similares) e (ii) minimização do espaço na memória requerido para armazenamento durante a fase de uso do classificador.

O IBL é um paradigma de aprendizado capaz de tratar vários tipos de problemas de reconhecimento de padrões. Entre eles se encontra o problema da detecção de novidades. Na detecção de novidades, um objeto da classe novidade pode ser definido como um objeto sem

nenhuma semelhança com qualquer dos objetos apresentados ao classificador na fase de treinamento. Como exemplo, suponha um classificador que diferencie cavalos de cães. Caso a esse classificador seja apresentado um objeto diferente destes, espera-se que esse objeto seja classificado como uma novidade (*i.e.*, seja rotulado como desconhecido). Um dos problemas dos diversos métodos para classificação investigados é que eles necessitam de instâncias artificiais para a representação da classe novidade durante a fase de treinamento. Já se mostrou que, em problemas de classificação convencionais, a utilização de tais instâncias não é eficiente, pois é difícil selecionar os exemplos mais adequadas para representar o que seriam novidades (OLIVEIRA, 2004). Dessa forma, se faz importante a investigação de métodos de treinamento que não necessitem de instâncias da classe novidade (instâncias positivas) durante o treinamento. Este é o caso da classificação com exemplos de uma única classe (*One-class Classification* - OCC) (OH et al., 2013)(SEO, 2007)(KEMMLER; RODNER; DENZLER, 2013)(CABRAL; OLIVEIRA; CAHÚ, 2009)(KHAN; MADDEN, 2010).

A Figura 1.1 ilustra o exemplo já citado de cavalos e cães. Nela, um classificador convencional (classificador para múltiplas classes) e um classificador para exemplos de uma única classe são aplicados a um conjunto de dados contendo exemplos de ambas as classes (cavalo e cão). A linha sólida representa o classificador convencional que distingue entre as classes, enquanto que a linha vermelha pontilhada seria gerada por um classificador *one-class*, também chamado de método para descrição de dados (*data description*). No exemplo da Figura 1.1, vê-se que essa descrição pode detectar de forma correta o objeto novidade (tubarão), enquanto que o classificador convencional o classificaria como um cão.

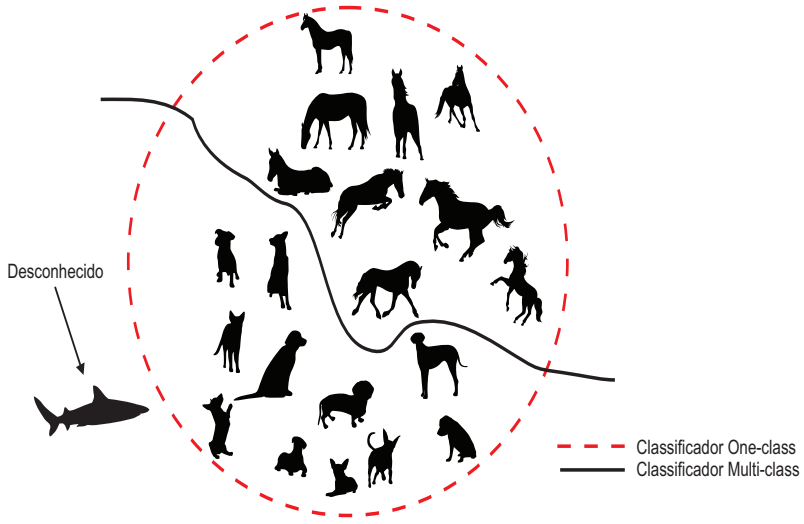

Figura 1.1: Ilustração da classificação com exemplos de múltiplas classes e de uma única classe.

1.1 Motivação

A maioria dos métodos para classificação de padrões baseados no IBL usa medidas de similaridade/dissimilaridade, como a distância Euclidiana, como base para a decisão de armazenar ou não uma instância de treinamento (ver (GARCIA et al., 2012)). Em OCC, a abordagem mais usada consiste na retenção de objetos na borda da classe normal (negativa) no intuito de construir uma descrição que envolve os dados normais conhecidos de antemão (TAX, 2001). Dessa forma, se um objeto z de teste se localiza dentro dessa descrição, é classificado como normal. Caso contrário, é reconhecido como novidade (*i.e.*, pertence à classe positiva que é desconhecida durante a fase de treinamento do classificador). Essa descrição pode ser construída, por exemplo, por um simples método IBL (como o *k-Nearest Neighbor Data Description* - *k*NNDD (TAX, 2001)) ou por um método baseado em estimativa de densidade (como o *Gaussian Process prior OCC*- GP-OCC). Esses métodos geram modelos que armazenam o conjunto de treinamento na íntegra e essa característica pode ser uma importante desvantagem na prática. Por outro lado, métodos baseados em Máquinas de Vetores de Suporte (*Support Vector Machines* - SVMs) armazenam apenas um percentual do conjunto de treinamento onde cada instância armazenada é um vetor de suporte. Porém, seu procedimento de treinamento envolve a realização de uma otimização quadrática que pode se tornar bastante custosa; ou até mesmo inviável dependendo do tamanho do conjunto de treinamento.

De acordo como o nosso conhecimento em relação ao estado da arte da detecção de novidades, métodos IBL usam medidas de similaridade/dissimilaridade sem analisar de forma minuciosa e isolada cada atributo de cada instância do problema por vez. Dessa forma, a investigação de uma abordagem que atue examinando de forma separada cada atributo de cada objeto de treinamento, no intuito de encontrar e armazenar de forma eficiente os limites dos dados normais, pode resultar em uma descrição mais precisa da classe referência. A depender dos parâmetros utilizados, essa abordagem garante que os objetos selecionados estarão na borda da descrição. Além disso, a resolução do problema de forma simples e intuitiva (em comparação com métodos baseados em SVMs que utilizam a otimização quadrática ou ferramentas sofisticadas da algebra linear) é também um aspecto positivo. A criação de uma descrição que envolve os dados da classe normal consiste na mesma ideia apresentada por métodos baseados em SVMs, porém, esses métodos operam no espaço de kernel (*i.e.*, um espaço de alta dimensionalidade) enquanto que a abordagem mencionada atua diretamente no espaço de características.

Teoricamente, todos os problemas envolvendo detecção de novidades podem ser resolvidos através da classificação com exemplos de uma única classe. Esse é mais um fator motivacional à criação de uma nova abordagem para OCC. Exemplos de aplicações de métodos para detecção de novidades com exemplos de uma única classe no mundo real são:

Detecção de fraudes em folhas de pagamento Essa é uma aplicação muito importante pois despesas com funcionários representam uma porção significativa do orçamento de uma

organização, seja ela pública ou privada. Como exemplo, pode-se citar o governo brasileiro que direciona grande parte de sua receita a gastos com pessoal. Esse é um cenário em que um sistema de auditoria de gastos seria de grande importância.

Detecção de fraudes em operações no comércio eletrônico Em sistemas tecnológicos, atividades fraudulentas têm ocorrido em diversas aplicações, tais como comunicação móvel, *on-line banking* e comércio eletrônico. Segundo o portal norte-americano Internet Retailer (INTERNET RETAILER – PORTAL TO E-COMMERCE INTELLIGENCE, ONLINE FRAUD COSTS E-RETAILERS $3.5 BILLION IN 2012, 2013), as fraudes virtuais fizeram com que varejistas americanos e canadenses perdessem, em 2012, US$ 3,5 bilhões; 3% a mais que em 2011 (US$ 3,4 bilhões) e 30% a mais que em 2010 (US$ 2,7 bilhões). Alguns trabalhos da literatura têm reportado a preocupação de empresários com o crescimento de fraudes em operações de comércio eletrônico, por exemplo, (BHATTACHARYYA et al., 2011) (SÁNCHEZ et al., 2009).

Detecção de intrusão O processo de detecção de intrusão (LUO; XIA, 2014) (KUANG; XU; ZHANG, 2014) se caracteriza por identificar e relatar atividades maliciosas agindo em computadores e/ou recursos da rede. Para que o sistema seja apto a determinar o que é um ataque no trafego de dados, ele deve ser previamente ensinado a reconhecer atividades normais do sistema.

Detecção de anomalias clínicas A detecção de anomalias clínicas consiste na identificação de anomalias no contexto de uma população de pacientes. Exemplos de detecção de anomalias são: detecção de câncer de mama (KERHET et al., 2006), detecção do mau funcionamento cardíaco (AVCI, 2009), detecção de diabetes (ACHARYA et al., 2013), etc.

1.2 Objetivos

O objetivo deste trabalho é a proposição, implementação e análise de um novo método para a detecção de novidades baseado no paradigma do Aprendizado Baseado em Instâncias (IBL) e Classificação com Exemplos de uma Única Classe (OCC). Embora já existam métodos populares para a resolução do problema tratado, esses métodos sofrem com o alto custo computacional na fase de modelagem ou com a desvantagem da não seleção das instâncias que melhor discriminam as classes para a composição do modelo (*i.e.*, armazenam o conjunto de treinamento na integra).

O método proposto neste trabalho, chamado FBDOCC (*Features Boundaries Detector for One-Class Classification*), possui apenas dois parâmetros cujos valores podem ser facilmente otimizados com a utilização de uma abordagem metaheurística, como o *Particle Swarm Optimization* (PSO). Adicionalmente, o método proposto deve ser capaz de fornecer a solução

do problema em tempo aceitável, ou seja, compatível ou melhor que os métodos considerados mais populares para essa classe de problemas.

Neste trabalho foram utilizadas bases de dados clássicas dos repositórios *University of California Irvine Machine Learning Repository* (BACHE; LICHMAN, 2013) e StatLib (*Data, Software and News from the Statistics Community*) (CAMPBELL; BENNETT, 2011), além de bases de dados sintéticas e uma base extraída de um renomado hospital situado na cidade de Recife, Brasil [1]. A ideia é utilizar as bases de dados para validar a aplicação do método proposto à tarefa de detecção de novidades e verificar a adequação do mesmo aos problemas tratados; essa verificação se dará considerando vários aspectos de desempenho em comparação a métodos no estado da arte.

1.3 Organização do Documento

Neste Capítulo introdutório, os objetivos e motivações desta tese foram apresentados. Os demais capítulos abordam os seguintes assuntos:

- Capítulo 2: Redução de Protótipos

  O Capítulo 2 oferece uma visão geral da Redução de Protótipos detalhando suas principais características e expondo as vantagens e desvantagens no uso dessas. Além disso, também serão apresentados os modos de representação, seleção e geração de protótipos de forma mais aprofundada através de exemplos de algoritmos dessas naturezas.

- Capítulo 3: Classificação com Exemplos de uma Única Classe

  O Capítulo 3 apresenta o problema da detecção de novidades de acordo com o paradigma da Classificação com Exemplos de uma Única Classe. Definições do paradigma OCC são apresentadas e uma visão geral dos métodos para classificação mais populares é apresentada.

- Capítulo 4: Método Proposto

  O Capítulo 4 introduz os métodos propostos para detecção de novidades neste trabalho. O desenvolvimento dos métodos é detalhado de forma textual e através de pseudo-códigos. Além disso, uma análise da complexidade do algoritmo FBDOCC é fornecida. São também apresentadas heurísticas para melhoria do custo computacional na fase de modelagem e limitações do método.

- Capítulo 5: Experimentos

[1] Real Hospital Português de Beneficência em Pernambuco - Av. Governador Agamenon Magalhães - Paissandu, Recife - PE, 52010-902

O Capítulo 5 apresenta os resultados dos vários experimentos realizados. Experimentos com dados referências foram realizados e uma análise da influência dos parâmetros ajustáveis do método é realizada. Um caso de estudo para o problema da detecção de cardiopatias em crianças com base em informações não invasivas é também desenvolvido.

- Capítulo 6: Conclusão

  No Capítulo 6, a conclusão deste trabalho é apresentada, assim como um resumo da contribuições geradas. Além disso, propostas de trabalhos futuros originadas a partir do presente trabalho são expostas.

2

Redução de Protótipos

O algoritmo do vizinho mais próximo (*Nearest Neighbor* - NN) (DUDA; HART; STORK, 2001) é talvez uma das formas mais simples e intuitivas de se classificar padrões. Seja $T = \{t_1^l, t_2^l, .., t_n^l\}$ um conjunto de treinamento contendo n vetores de tamanho l, e seja X um espaço de Hilbert[1] com uma métrica ρ, tal que ρ mede a similaridade entre quaisquer dois vetores em T, sendo que T está contido em X, é possível realizar a aplicação do kNN a T utilizando ρ como distância. A Figura 2.1 mostra um exemplo onde o espaço de características é Euclidiano, bidimensional (*i.e.*, $l = 2$) e a similaridade entre os objetos (ρ) pode ser medida pela simples distância Euclidiana. Nesse caso, a superfície de decisão é ilustrada pelo espaço de Voronoi. Esse tipo de classificação pode ser estendida de maneira a se considerar os k vizinhos mais próximos e nesse caso a classe mais frequente dentre os k vizinhos é o resultado obtido pelo modelo.

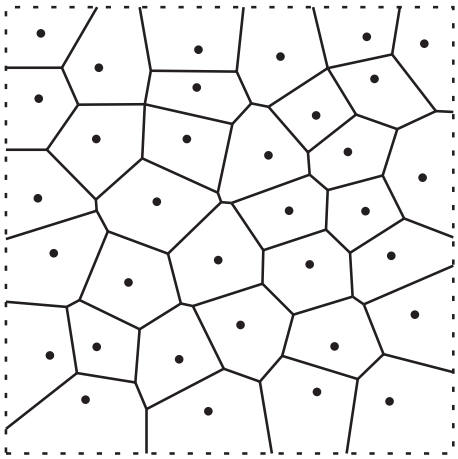

Figura 2.1: Espaço de Voronoi para uma distribuição de dados hipotética.

O algoritmo NN original armazena todas as instâncias de treinamento. Ele aprende muito rápido, pois precisa apenas ler e armazenar o conjunto de treinamento sem a necessidade de um processamento extra, e tem uma generalização precisa para muitas aplicações. Porém, como o algoritmo do vizinho mais próximo básico armazena todas as instâncias de treinamento,

[1] Esse espaço é utilizado devido à sua capacidade de generalizar operações do espaço Euclidiano bi-dimensional para espaços n-dimensionais, sendo $n \gg 2$.

dependendo do problema, pode ser necessário um grande espaço de memória para o armazenamento dessas instâncias. Além disso, o algoritmo deve percorrer todas as instâncias disponíveis para a classificação de um novo vetor de entradas (objeto), ou seja, ele pode se tornar lento durante a classificação. Também, como ele armazena todas as instâncias do conjunto de treinamento, instâncias que representam ruídos (com erro no vetor de entradas ou na classe de saída, ou aquelas não representativas de casos típicos) são armazenadas, o que pode prejudicar a generalização. Dessa forma, se faz necessário armazenamento apenas das instâncias que melhor representam o problema em questão e o descarte de ruídos e instâncias com informação redundante.

O conceito de redução de protótipos (*Data Reduction*) é também conhecido por vários outros termos, como, *editing* (WILSON, 1972), *condensing* (GATES, 1972), *filtering* (LOWE, 1995), etc, dependendo da finalidade da tarefa de redução de protótipos. A tarefa da classificação visa descobrir a classe de um objeto usando uma função discriminante aprendida a partir de um conjunto de instâncias presentes em um conjunto de treinamento. E esse é o que deve ser reduzido, o conjunto de treinamento. Tenta-se representar o conjunto de treinamento completo da forma mais eficaz possível mantendo a precisão na classificação, ou mesmo a aumentando, utilizando-se apenas algumas instâncias. Resumindo, essa classe de algoritmos busca por resultados em que requisitos de memória e tempo de execução sejam reduzidos enquanto a precisão do conjunto de treinamento original seja preservada ou melhore.

A Figura 2.2 mostra um exemplo em que objetos no espaço bi-dimensional de duas classes distintas formam um conjunto de treinamento. O painel da esquerda mostra a distribuição de dados original com o conjunto de treinamento completo. O painel da direita mostra o mesmo conjunto de treinamento, porém, instâncias consideradas redundantes estão em tom de cinza e apenas as instâncias na fronteira entre as classes são armazenadas para representar o modelo discriminante. Esse exemplo descarta instâncias internas à distribuição.

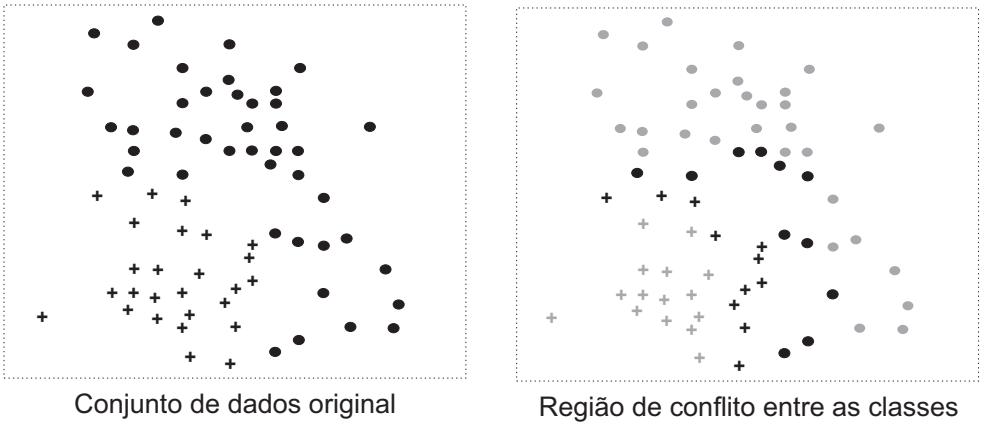

Figura 2.2: Distribuições de dados completas e após a execução da redução de protótipos retendo instâncias na fronteira entre as classes.

2.1 Características de Esquemas para Redução de Protótipos

Algoritmos para redução de protótipos compartilham algumas características que podem ser decisivas na escolha de uma abordagem para determinada tarefa. Os aspectos aqui descritos, de certa forma, categorizam essa classe de algoritmos e devem ser levados em consideração durante o desenvolvimento de novos classificadores. As subseções a seguir apresentam características de algoritmos para redução de protótipos de acordo com a descrição de Wilson e Martinez (WILSON; MARTINEZ, 1997).

2.1.1 Representação

Uma escolha na construção de um algoritmo para redução de protótipos é a decisão entre reter um subconjunto do conjunto de treinamento original ou modificar as instâncias originais (ou até mesmo criar novas), gerando novas representações. Por exemplo, alguns algoritmos (SALZBERG, 1991) (WETTCHERECK; DIETTERICH, 1995) usam hiperretangulos para representar uma coleção de instâncias; instâncias podem ser transformadas em regras (DOMINGOS, 1995) (DOMINGOS, 1996); e protótipos podem ser usados para representar um conjunto de instâncias (CHANG, 1974), mesmo se nenhuma instância original ocorreu onde aqueles protótipos estão localizados. Estes algoritmos são chamados de *criacionais* pelo fato de poderem criar novos protótipos ou ajustarem instâncias da base de dados.

Em contrapartida, muitos algoritmos IBL buscam reter uma fração do conjunto de instâncias original. Um problema com o uso de instâncias em sua forma original é que pode não haver nenhuma instância localizada em um ponto que resultaria na mais precisa descrição daquele problema. Protótipos, por outro lado, podem ser construídos, de forma artificial, para que existam exatamente onde eles são necessários, caso esses locais sejam possíveis de serem determinados. Algoritmos que retêm instâncias ao invés de criá-las ou atualizá-las são chamados de algoritmos seletivos. Exemplos de algoritmos seletivos são o *Condensed Nearest Neighbor* (CNN) (HART, 1968) e o *Nearest Neighbor Structural Risk Minimization* (NNSRM) (KARACALI; KRIM, 2003).

2.1.2 Direção de Busca

Na busca por um subconjunto do conjunto de instâncias, a ser armazenado, existem algumas direções de busca a serem tomadas, sendo as principais a incremental, decremental e *batch*.

Incremental Uma busca incremental começa com um subconjunto S VAZIO do conjunto de treinamento T, e adiciona cada instância de T a S se ela satisfaz alguma pré-condição. Nesse caso, a ordem de apresentação das instâncias pode ser muito importante. Em particular, as primeiras instâncias podem ter probabilidades de serem incluídas em S bem

diferentes caso elas fossem testadas mais tarde.

Em tais esquemas, a ordem de apresentação das instâncias em T ao algoritmo é geralmente aleatória. Por definição, um algoritmo incremental deve ser capaz de tratar novas instâncias à medida que elas se tornem disponíveis sem que todas elas estejam disponíveis no início da fase de aprendizagem.

Uma vantagem do esquema incremental é que, se houver a disponibilidade de instâncias para adição em S, após o treinamento, elas podem continuar a serem adicionadas a S de acordo com o mesmo critério. Outra vantagem é que algoritmos com essa característica podem ser mais rápidos na fase de aprendizado que algoritmos não incrementais, pela possibilidade do descarte de algumas instâncias enquanto outras são adicionadas.

A maior desvantagem é que algoritmos incrementais são sensíveis à ordem de apresentação das instâncias, e suas primeiras decisões são baseadas em muito pouca informação, e são suscetíveis a erro à medida que uma maior quantidade de informação se torna disponível. Alguns algoritmos incrementais usam um pequeno número de instâncias em uma fase *batch* inicial para aliviar esses problemas (SALZBERG, 1991).

Alguns algoritmos adicionam instâncias a S de forma não totalmente incremental. Eles examinam todas as instâncias disponíveis para ajudar na seleção de qual a próxima instância a adicionar (KARACALI; KRIM, 2003). Essa abordagem pode melhorar de forma considerável o desempenho do algoritmo dado que cada decisão, entre reter ou não uma instância, é baseada em uma visão completa dos dados.

Decremental A busca decremental começa com $S = T$ e então procura por instâncias para remover de S. Novamente, a ordem de apresentação das instâncias é importante, porém, diferentemente da busca incremental, todas as instâncias estão disponíveis a qualquer momento. Dessa forma, uma busca pode ser feita para a determinação de qual seria a melhor instância a se remover a cada passo do algoritmo. Exemplos de algoritmos decrementais são o SNN (GATES, 1972), RNN (RITTER et al., 1975) e o ENN (WILSON, 1972). O RISE (DOMINGOS, 1995) pode ser visto como decremental, porém, ao invés de apenas excluir instâncias de S, ele também as transforma em regras. Da mesma forma que o RISE (DOMINGOS, 1995), o PNN (*Prototypes for Nearest Neighbor*) (CHANG, 1974) opera de forma decremental, mas protótipos se fundem ao invés de serem simplesmente removidos.

Uma desvantagem na busca decremental é que ela tem um custo computacional freqüentemente maior que algoritmos incrementais. Por exemplo, na busca por um vizinho mais próximo em T de uma instância z, n cálculos devem ser feitos, sendo n o número de instâncias em T.

De qualquer forma, se a aplicação de um algoritmo decremental puder resultar em uma maior redução no armazenamento, então a computação extra exigida durante a aprendizagem (que é feita apenas uma vez) pode ser justificada pela redução no custo computacio-

nal no uso da aplicação após o treinamento. Melhora na precisão do algoritmo na fase de uso também pode justificar um maior custo computacional no treinamento.

Batch Outra forma de aplicar a redução de protótipos em um conjunto de dados é a *batch*. Nessa forma, decide-se se cada instância satisfaz o critério de remoção antes da remoção de alguma delas. Então, todas as instâncias que satisfazem a aquele critério são removidas de uma só vez. Por exemplo, o All k-NN rule (TOMEK, 1976) opera dessa forma. Essa abordagem pode livrar o algoritmo de ter que constantemente atualizar listas de vizinhos mais próximos e outras informações quando instâncias são individualmente removidas. Porém, também existem grandes desvantagens no processamento *batch*. Assuma que a seguinte regra seja aplicada a um conjunto de instâncias:

> Remova a instância se ela for da mesma classe que seus k vizinhos mais próximos.

Isso pode resultar no desaparecimento de clusters completos se não existirem instâncias de classes diferentes na vizinhança.

2.1.3 Instâncias na Borda x Instâncias Centrais

Outro fator que distingue algoritmos de redução de protótipos é a escolha de instâncias localizadas na borda entre as classes, instâncias localizadas no centro das distribuições das classes ou outros tipos de localização. A Figura 2.3, por exemplo, apresenta casos tanto de escolha de instâncias localizadas na borda da descrição (fronteira entre as classes) quanto protótipos localizados no centro das distribuições.

A justificativa para o armazenamento de instâncias na borda é que instâncias internas não afetam os limites de decisão tanto quanto instâncias na borda e, dessa forma, podem ser removidas sem que a precisão na classificação seja prejudicada.

Em contrapartida, alguns algoritmos ao invés de removerem instâncias na borda, removem instâncias que são ruidosas ou que não condizem com suas vizinhanças. Dessa forma, classificadores com fronteiras entre as classes mais suaves são alcançados. Porém, esses algoritmos não removem instâncias internas, que não agregam informação ao classificador.

Pode ser necessário um grande número de instâncias para definir completamente a borda, então alguns algoritmos armazenam apenas pontos centrais no intuito de usar apenas instâncias mais típicas de uma determinada classe. Isso pode afetar de forma significativa os limites de decisão, porque estes não dependem apenas dos pontos mais típicos de uma dada classe, mas também de onde se localizam as fronteiras de outras classes.

A Figura 2.3 apresenta dois modelos de classificadores. O primeiro gera pontos centrais à distribuição fazendo com que a borda entre as classes (no caso de apenas dois centros) seja uma reta. Para esse caso, o modelo visivelmente não representa de forma fiel a fronteira entre as classes, devido à sua forma complexa. O outro modelo retém instâncias na fronteira entre as

classes descartando instâncias mais centrais. Esse modelo tem um maior poder na descrição de fronteiras mais complexas, como mostra a Figura.

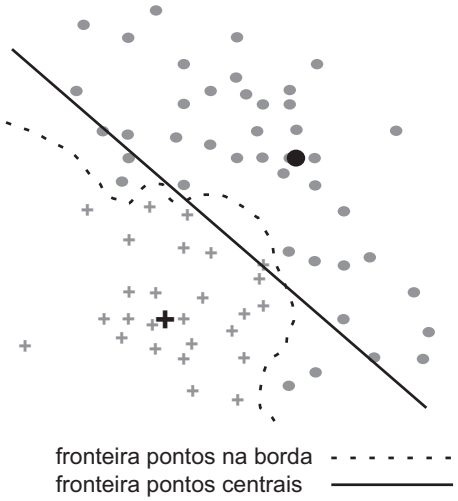

Figura 2.3: Exemplo de fronteira entre as classes para o caso da escolha de pontos centrais e de pontos na borda para a discriminação entre as classes

López *et. al.* (OLVERA-LÓPEZ; CARRASCO-OCHOA; MARTÍNEZ-TRINIDAD, 2010) se basearam em conceitos de métodos de agrupamento para realizar a redução de protótipos. Inicialmente os autores utilizam o algoritmo *c-means* para encontrar grupos e se esses grupos forem homogêneos a instância que representa a média desse grupo é armazenada. Caso o grupo não seja homogêneo, as instâncias em regiões conflitantes entre as classes são armazenadas. Esse é um exemplo de método híbrido, ou seja, o método armazena instâncias tanto na borda da descrição quanto centrais.

2.1.4 Função Distância

A função distância (ou seu complemento, a função de similaridade) é usada na decisão de quais instâncias se localizam mais próximas de um determinado ponto e têm uma maior importância no sistema de aprendizado.

O algoritmo do vizinho mais próximo e suas derivações, geralmente, usam variações da função de distância Euclidiana, que é definida de acordo com a Equação (2.1):

$$D(\vec{x},\vec{y}) = \sqrt{\sum_{i=1}^{m}(x_i - y_i)^2}, \tag{2.1}$$

onde $\overrightarrow{x}$ e $\overrightarrow{y}$ são dois vetores de entradas, m é o número total de atributos (cardinalidade do vetor de entradas) e x_i e y_i são os valores dos atributos de entrada de índice i. Essa função é indicada quando todos os atributos de entrada são numéricos e seus valores variam, aproximadamente, dentro de um mesmo intervalo. Quando os atributos variam dentro de intervalos que diferem de forma significativa, pode se normalizar os vetores de entradas do conjunto de instâncias de

acordo com a Equação (2.2) para normalização dos valores no intervalo [0,1] e a Equação (2.3) para normalização dos valores em um intervalo de valores arbitrários [a,b]:

$$Z'(t) = \frac{Z(t) - min(Z)}{max(Z) - min(Z)}, \tag{2.2}$$

$$Z'(t) = a + \frac{(Z(t) - min(Z)) * (b-a)}{max(Z) - min(Z)}. \tag{2.3}$$

Uma grande variedade de outras distâncias, com indicações para diferentes situações, existem. Algumas delas são:

- Minkowski (BATCHELOR, 1978) - Equação (2.4): A distância de Minkowski é uma generalização da distância Euclidiana (*i.e.*, a distância Euclidiana corresponde à distância de Minkowski com valor de parâmetro p igual a 2). Algoritmos de agrupamento, como o Fuzzy c-means, por exemplo, têm sido uma aplicação regular dessa métrica. A intuição por trás do uso dessa distância é que, variando-se o parâmetro p, ocorre também uma modificação na estrutura geométrica dos grupos na base de dados (GROENEN; KAYMAK; ROSMALEN, 2007).

$$D(\vec{x}, \vec{y}) = \left(\sum_{i=1}^{m} |x_i - y_i|^p \right)^{\frac{1}{p}}, \tag{2.4}$$

- Manhattan (DEZA; DEZA, 2013) - Equação (2.5): A distância de Manhattan computa a distância que seria percorrida de um ponto a outro caso o caminho a ser seguido fosse em forma de grid. Uma possível aplicação dessa métrica é na busca por estratégias em um jogo de xadrez (LINHARES et al., 2012). Essa distância é um caso especial da distância de Minkowski onde o parâmetro p é igual a 1.

$$D(\vec{x}, \vec{y}) = \sum_{i=1}^{m} |x_i - y_i|, \tag{2.5}$$

- Chebyshev (WEBB, 2002) - Equação (2.6): Criada pelo matemático russo Pafnuty Lvovich Chebyshev, essa distância retorna o maior valor entre todas as diferenças das respectivas dimensões dos objetos. Assim como a Manhatan, a distância Chebyshev é um caso especial da Minkowski onde o p tende a infinito e também se aplica à busca por soluções em jogos de xadrez.

$$D(\vec{x}, \vec{y}) = max_{i=1}^{m} |x_i - y_i|, \tag{2.6}$$

- Canberra (LANCE; WILLIAMS, 1967) - Equação (2.7): Trata-se do somatório das diferenças entre as dimensões divididas pelas somas dos valores absolutos das di-

mensões. Um exemplo de aplicação é na detecção de intrusão de computadores (EMRAN; YE, 2002).

$$D(\vec{x},\vec{y}) = \sum_{i=1}^{m} \frac{|x_i - y_i|}{|x_i + y_i|}. \tag{2.7}$$

2.1.5 Votação

Em algoritmos baseados na retenção de instâncias, frequentemente é utilizado o parâmetro k; que indica qual a quantidade de instâncias de treinamento, mais próximas à instância de teste, devem ser consideradas na tomada de decisão.

No algoritmo básico de vizinho mais próximo, a atribuição de um valor maior que 1 ao parâmetro k diminui a sensibilidade do algoritmo a ruídos e tende a suavizar a fronteira entre as classes. Porém, se o conjunto de treinamento foi reduzido ao ponto de haver apenas uma instância representando o que antes era um grupo (*cluster*), talvez $k = 1$ seja mais apropriado que $k > 1$, especialmente se as instâncias ruidosas foram removidas durante o processo de redução de protótipos (WILSON; MARTINEZ, 2000).

2.2 Seleção de Protótipos

Métodos de seleção de protótipos (*Prototype Selection Scheme* - PSS) (GARCIA et al., 2012) pressupõem que o conjunto de treinamento oferece a melhor precisão na classificação pelo simples uso do classificador kNN. Dessa forma, o que se busca é manter ou melhorar discretamente a precisão do kNN a partir do isolamento do menor conjunto de instâncias possível encontrado pelo PSS. Minimizando-se o conjunto de dados é possível reduzir a complexidade espacial, assim como o custo computacional do modelo de classificador a ser utilizado posteriormente. Além disso, a eliminação de ruídos pode beneficiar a generalização.

PSSs podem ser formalmente especificados da seguinte forma: Seja T um conjunto de treinamento formado por N instâncias do tipo $X_p = (X_{p1}, X_{p2}, .., X_{pm}, X_{pc})$, com X_p pertencendo à classe c (representada por X_{pc}). Seja $S \subseteq T$ o subconjunto resultante da execução do PSS sobre T e seja TS um conjunto de instâncias de teste, disjunto de T. Para que o PSS seja considerado eficiente, a precisão da classificação das instâncias do conjunto TS realizadas utilizando-se o subconjunto S, como modelo, deve ser melhor ou igual à aplicação do kNN utilizando o conjunto T original, como modelo. Na Seção 2.2.1, a título de demonstração, é apresentado um exemplo de classificador que utiliza o paradigma da seleção de protótipos.

2.2.1 Nearest Neighbor with Structural Risk Minimization - NNSRM

O algoritmo NNSRM (KARACALI; KRIM, 2003) é um exemplo similar, em objetivo, aos algoritmos da classe dos PSSs, porém, conceitualmente, o NNSRM difere das técnicas

convencionais pois foi criado pela aplicação explicita do princípio da minimização do risco estrutural (*Structural Risk Minimization* - SRM) (VAPNIK; CHERVONENKIS, 1974) a um classificador baseado na regra do vizinho mais próximo.

A proposta do NNSRM é encontrar o conjunto S de instâncias mais significativas (referências) com menor cardinalidade, tal que, para todo $x_i \in T$, a função discriminante, resultante do conjunto S, encontrada é $f(x_i) = y_i$ (*i.e.*, o algoritmo garante acerto de 100% para as instâncias do conjunto de treinamento). Sendo cada objeto de T da forma (x_i, y_i), onde x_i representa o vetor de entradas do objeto e y_i representa a saída (classe) desse objeto. Busca exaustiva é claramente impraticável, pois poderia se chegar a um grande número de subconjuntos de T ($(2^{l_1} - 1) * (2^{l_2} - 1)$ subconjuntos para um problema de apenas duas classes de tamanhos l_1 e l_2, respectivamente); haveria ainda a necessidade de testá-los no conjunto de treinamento completo e selecionar o de menor cardinalidade com erro de treinamento igual a zero. Ao invés disso, Karacali e Krim propuseram o seguinte algoritmo (KARACALI; KRIM, 2003).

Considere todas as distâncias $\rho(x_i, x_j)$, tal que, $y_i = -1$ e $y_j = 1$, e seja d_k uma lista, para $k = 1, .., l_1 l_2$, ordenada de forma crescente, contendo essas distâncias. O Algoritmo 1 descreve o funcionamento do NNSRM para um problema com apenas duas classes diferentes. Inicialmente o conjunto de referências é um conjunto vazio. Enquanto o R_{emp} (risco empírico ou erro de treinamento) for maior que zero, deve-se encontrar o par de instâncias no conjunto de treinamento $\{x_i^1, x_i^{-1}\}$ mais próximas; nesse caso x_i^1 corresponde ao vetor de entradas da amostra de índice i e da classe 1 no conjunto de treinamento e x_i^{-1} a amostra de índice j e da classe -1. O valor da distância entre as instâncias x_i^1 e x_i^{-1} deve ser igual ao valor contido na posição k do vetor de distâncias d. Se esse par de instâncias não estiver contido em S, ambas as instâncias devem ser adicionadas a S. Esse procedimento deve ser repetido até que o erro de treinamento seja igual a zero.

Algoritmo 1: *Nearest Neighbor with Structural Risk Minimization.*

Input: T
Output: S
1 $S = \emptyset$
2 $k = 1$
/* R_{emp} consiste no erro obtido para o conjunto de treinamento, ou seja, risco empírico. */
3 **while** $R_{emp} > 0$ **do**
/* d é um vetor contendo todas as distâncias entre exemplos de treinamento de classes distintas. */
4 encontre $(t_i, t_j) \mid \rho(t_i, t_j) = d_k$ para $y_i = 1$ e $y_j = -1$
5 **if** $(t_i, t_j) \notin S$ **then**
6 $S = S \cup (t_i, t_j)$
7 $k = k + 1$

O procedimento proposto aumenta a inclusão de pares de objetos $\{(x_i, x_j) | y_i = -1, y_j =$

1} mais próximos, e se mantém atualizando o conjunto de referências, S, até que a classificação correta do conjunto de treinamento completo ocorra. A intuição por trás do algoritmo NNSRM é que a maioria dos erros na classificação ocorre em regiões onde objetos de classes diferentes se encontram próximos. Dado um conjunto de treinamento, essas regiões são identificadas pelos pares de objetos de classes diferentes com menores distâncias entre si, e a superfície de separação deve funcionar, primeiramente, para esses objetos situados nos limites entre as classes, ou, regiões de conflito entre as classes (ver Figura 2.2).

Para a classificação com instâncias de múltiplas classes, seja m_i os rótulos das classes, para $i = 1...M$. Como no problema com duas classes, o classificador deve classificar todo o conjunto de treinamento de maneira correta. Seja $X_T = \{x_1^{m_1}, x_2^{m_1}, x_3^{m_1}, ..., x_{l_1}^{m_1}, x_1^{m_2}, ..., x_{l_M}^{m_M}\}$ o conjunto de treinamento, onde l_i denota o número de elementos na classe m_i no conjunto de treinamento. O funcionamento do NNSRM para múltiplas classes é detalhado pelo Algoritmo 2. Nesse caso a lista de distâncias, d, será da forma $d_{k^{mi,mj}}^{mi,mj}$, onde $k^{mi,mj}$ é o índice do valor contido em d e mi e mj são as classes das instâncias de treinamento que resultaram nessa distância de índice $k^{mi,mj}$ no vetor d. O vetor d é novamente ordenado em ordem crescente de valores das distâncias. Seja f_s a função que representa o classificador para um conjunto de referências S, se $f_s(x^{mi}) = mj$, deve-se buscar as instâncias com distância entre si $d_{k^{mi,mj}}^{mi,mj}$ e caso as instâncias que resultaram nessa distância não se encontrem ainda em S, essas devem ser adicionadas a S. O índice $k^{mi,mj}$ é então incrementado.

Algoritmo 2: *Kernel Nearest Neighbor with Structural Risk Minimization.*

```
   Input: T
   Output: S
1  S = k^(mi,mj) = 1
2  while R_emp > 0 do
3      for i = 1 ... #T do
           /* Se aconteceu um erro na classificação.        */
4          if f_s(t_i) == j then
5              encontre (t_i, t_j) | ρ(t_i, t_j) = d^{i,j}_{k^{mi,mj}}
6              k^{mi,mj} = k^{mi,mj} + 1
7          if t_i ∉ S then
8              S = S ∪ t_i
9          else
10             Retorna para a linha 2.
```

O algoritmo NNSRM converge, pois no pior caso todos os objetos no conjunto de treinamento são incluídos em S.

Em (KARACALI; RAMANATH; SNYDER, 2004), Karacali *et. al.* experimentaram a substituição da distância Euclidiana utilizada no NNSRM original, pela medida de distância utilizando o kernel RBF.

$$\|z(x_i) - z(x_j)\|^2 = K(x_i,x_j) + K(x_i,x_j) - 2K(x_i,x_j), \tag{2.8}$$

onde K é a função que caracteriza o *kernel*. O algoritmo em (KARACALI; RAMANATH; SNYDER, 2004) funciona da mesma forma que em (KARACALI; KRIM, 2003), apenas a distância foi substituída pela função de kernel.

O NNSRM é incremental e seletivo. Outra característica dele é o uso de pontos na borda.

2.3 Geração de Protótipos

Algoritmos baseados na geração de protótipos (TRIGUERO et al., 2012) constroem novos exemplos no espaço de características de forma artificial a partir do conjunto de treinamento. A motivação por trás da Geração de Protótipos é a obtenção de um conjunto de protótipos P_s que consiste de r ($r < N$, e sendo N o número de exemplos de treinamento) protótipos ou também instâncias de treinamento. Esses protótipos são gerados de forma a representar de forma eficiente as distribuições das classes e ter um bom nível de discriminação considerando o conjunto de treinamento. A cardinalidade de P_s deve ser suficientemente pequena para reduzir tanto o armazenamento, quanto o tempo de teste de uma nova instância z na utilização do classificador kNN.

Algoritmos de geração de protótipos seguem diferentes heurísticas para a geração do conjunto de protótipos P_s que representa as fronteiras entre as classes. A primeira abordagem publicada na literatura foi o método PNN (*Prototypes for Nearest Neighbor* - PNN) (CHANG, 1974) que funciona através da fusão de instâncias da mesma classe em sucessivas iterações gerando assim centróides. Uma das famílias mais importantes de métodos para geração de protótipos são o métodos baseados no Aprendizado por Quantização Vetorial (*Learning Vector Quantization* - LVQ) (KOHONEN, 1990). A seguir é apresentado um exemplo de classificador que utiliza o paradigma da geração de protótipos.

2.3.1 *Prototypes for Nearest Neighbor* - PNN

O algoritmo PNN (CHANG, 1974) pode ser resumido da seguinte maneira: dado um conjunto de treinamento T, o algoritmo inicia com todos os objetos em T como protótipos. Inicialmente, o conjunto A está vazio e o conjunto B é igual a T. O algoritmo seleciona um objeto de forma arbitrária em B e o adiciona a A. Após isso, os dois protótipos mais próximos em $p \subset A$ e $q \subset B$ da mesma classe são fundidos, sucessivamente, em um novo protótipo, $p*$, se a junção não degradar a generalização em T, onde $p*$ é o peso médio entre p e q. Por exemplo, se p e q são associados com os pesos W_p e W_q, respectivamente, $p*$ é definido como $(W_p p + W_q q)/(W_p + W_q)$ e, inicialmente, p tem peso $(W_p + W_q)$. Quando criado, todo protótipo tem peso 1. O procedimento adotado pelo PNN é ilustrado pelo Algoritmo 3.

Algoritmo 3: *Prototypes for Nearest Neighbor.*

Input: T
Output: P_s

```
1  B = T
2  for q ∈ B do
3      W_q = 1
4  Seleciona-se aleatoriamente uma instância em B e se insere em P_s
5  MERGE = 0
6  while #B > 0 do
7      encontre as instâncias p e q, de P_s e B, respectivamente, mais próximas.
8      if classe de p ≠ classe de q then
9          P_s = P_s ∪ q
10         B = B − q
11     else
12         p* = (W_p * p + W_q * q)/(W_p + W_q)
           /* ε representa o erro empírico.                    */
13         if ε aumentou then
14             P_s = P_s ∪ q
15             B = B − q
16         else
17             P_s = P_s − p
18             B = B − q
               /* p* tem peso (W_p + W_q).                     */
19             P_s = P_s ∪ p*
20             MERGE = MERGE + 1
21 if MERGE = 0 then
22     Retorna P_s
23 else
24     B = T
25     Retorna à linha 4.
```

Esse algoritmo é um bom exemplo de algoritmo que utiliza a forma de representação criacional. Outras características dele são o uso de pontos na borda e direção de busca incremental, mesmo começando removendo elementos do conjunto B.

2.4 Considerações Finais

Atualmente, um grande volume de informação é disponibilizada em meios de comunicação como a Internet, por exemplo. Classificar, entender ou comprimir essa informação é uma tarefa difícil. O mesmo problema é também comum em conjuntos de dados volumosos como os encontrados nos diversos problemas da mineração de dados (como categorização de textos, previsão de séries temporais, etc.). Neste capítulo foram apresentadas algumas técnicas para

a redução de protótipos que visam a redução da complexidade espacial do modelo enquanto, simultaneamente, asseguram que essa redução não afetará de modo significativo superfície de decisão do conjunto de treinamento original. Além disso, os aspectos mais importantes durante a tarefa da redução do protótipos foram discutidos de forma detalhada.

No intuito de se obter uma maior eficiência na redução de protótipos, é importante ter em mãos algum conhecimento *a priori* sobre a distribuição dos dados a serem descritos. Por exemplo, a existência ou não de ruídos no conjunto de treinamento, a existência ou não de uma fronteira bem definida entre as classes, o uso de conjuntos de dados volumosos para a descrição do problema ou a disposição dos dados no espaço de entradas são alguns dos fatores que podem ser decisivos na escolha dentre o grande número de métodos disponíveis na literatura para a tarefa da redução de protótipos.

3

Classificação com Exemplos de uma Única Classe

Este Capítulo detalha várias técnicas para classificação com exemplos de uma única classe (*One-class Classification* - OCC) cuja principal aplicação é a detecção de novidades. A detecção de novidades (*novelty detection*), também conhecida como *anomaly detection* e *data description*, consiste em uma importante tarefa com emprego em muitas áreas do conhecimento. Por exemplo, um sistema de análise de vídeo pode ser planejado de forma a detectar o aparecimento de objetos não pertencentes a uma determinada cena ou pode ser responsável pela detecção de falhas em um equipamento de acordo com o histórico do funcionamento deste equipamento.

Vários métodos foram propostos para a resolução de problemas de classificação com exemplos de uma única classe. Três principais abordagens podem ser identificadas: estimativa de densidade (*density estimation*) (KEMMLER; RODNER; DENZLER, 2013), métodos baseados em fronteiras entre as classes (*boundary methods*) (SCHÖLKOPF et al., 2001) (TAX, 2001) e métodos baseados na reconstrução dos dados (*reconstruction methods*) (HOFFMANN, 2007).

3.1 Considerações Iniciais

Na classificação com exemplos de uma única classe, dois elementos comuns podem ser identificados. O primeiro elemento é a medida de distância $d(z)$, ou semelhança, do objeto de teste à classe alvo. O segundo elemento é um limiar θ para essa distância ou semelhança. Um objeto z de teste é dito pertencer à classe normal se sua distância à classe alvo (classe normal) é menor que o limiar θ_d (Equação 3.1).

$$f(x) = \begin{cases} d(z) \leq \theta_d, & \textit{NORMAL} \\ d(z) > \theta_d & \textit{NOVIDADE} \end{cases} \tag{3.1}$$

ou se sua similaridade ($s(z)$) com a classe alvo é maior que o limiar θ_s (3.2).

$$f(x) = \begin{cases} s(z) \geq \theta_s, & \textit{NORMAL} \\ s(z) < \theta_s & \textit{NOVIDADE} \end{cases} \tag{3.2}$$

Dessa forma, classificadores baseados em OCC diferem basicamente em relação à definição de $d(z)$ (ou $s(z)$), na forma como $d(z)$ (ou $s(z)$) é otimizado e no estabelecimento de um limiar θ em relação ao conjunto de treinamento.

Existem dois tipos de erro na classificação com exemplos de uma única classe, o erro tipo I e erro tipo II. Dado um objeto z de teste e um modelo de classificador $f(x)$, um erro do tipo II acontece quando $f(z)$ atribui z à classe normal (negativa) quando na verdade z é um exemplo da classe novidade (positiva). Um erro do tipo I acontece quando $f(z)$ classifica z como desconhecido quando na verdade z é um exemplo da classe normal. A Figura 3.1 ilustra dois modelos de classificador onde as regiões cinza demarcadas pelas linhas tracejadas denotam as regiões de aceitação do classificador (*i.e.*, regiões onde os objetos contidos nas mesmas são reconhecidos como da classe normal). A regiões demarcadas pelas linhas contínuas representam os conjuntos de treinamentos. O modelo de classificador apresentado em 3.1.a deixa uma parte do conjunto de treinamento fora da descrição, classificando essa parte como novidade. Esse é um caso de erro do tipo I. Esse mesmo modelo não fornece uma descrição justa à distribuição de treinamento classificando exemplos com baixa similaridade a essa distribuição como exemplos normais (*i.e.*, resultando em uma alta taxa de erro do tipo II). O modelo de classificador apresentado em 3.1.b fornece uma descrição mais precisa da distribuição de treinamento resultando em uma menor taxa de erro do tipo II e eliminando o erro do tipo I. A depender da complexidade da superfície normal a ser descrita, ambos os erros podem ser eliminados durante a fase de modelagem.

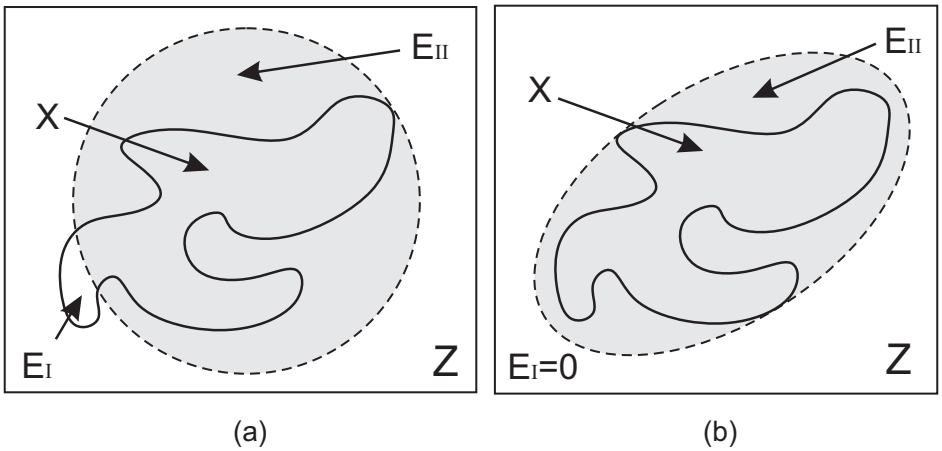

Figura 3.1: Regiões na Classificação com Exemplos de Uma Única Classe. (a) A descrição (linha tracejada) define de forma pouco justa a distribuição X e (b) a descrição se adapta melhor à distribuição X diminuindo o erro tipo II e estinguindo o erro tipo I.

Em alguns domínios, como na medicina, o custo de um erro do tipo II (*i.e.*, classificar um enfermo como saudável) pode ser muito alto, o paciente pode vir a óbito, no pior caso. Em

contrapartida, o custo de se classificar um paciente saudável como enfermo pode ser relativamente baixo. Considerando os diferentes custos dos erros tipos I e II, pode-se ajustar o limiar de detecção θ de acordo com o domínio específico levando o classificador a aumentar a precisão na detecção de novidades ou na aceitação de objetos como normais.

3.2 Métodos de classificação com exemplos de uma única classe

Nessa Seção serão apresentados detalhes de funcionamento de todos os métodos utilizados neste trabalho.

3.2.1 Máquinas de Vetores de Suporte para Classificação com Exemplos de uma Única Classe (*One-Class* SVM)

Máquinas de vetores de suporte (SVMs - *Support Vector Machines*) consistem em um poderoso método de classificação baseado nos princípios da minimização do risco estrutural (SRM - *Structural Risk Minimization*). SVMs estão entre os mais sofisticados métodos supervisionados e não paramétricos existentes. Baseados nas SVMs, Schölkopf *et al.* (SCHÖLKOPF et al., 2001) propuseram o método *one-class* SVM para classificação com exemplos de uma única classe. Exemplos de aplicação desse método são a classificação de documentos (MANEVITZ; YOUSEF, 2001) e a recuperação de imagens (CHEN; ZHOU; HUANG, 2001).

Após o mapeamento dos objetos, via *kernel*, para uma espaço de características, o método *one-class* SVM trata a origem como o único membro da segunda classe. Como mencionado anteriormente, diferentemente do SVDD (que busca pela hipersfera de menor volume), o método *one-class* SVM, ou ν-SVM, está interessado no hiperplano ótimo, com máxima distância da origem, no espaço de características (*feature space*), tal que uma fração pré-definida das amostras de treinamento será separada dos dados normais por esse plano. A meta é maximizar a margem de separação da origem. Da mesma forma que em SVM para múltiplas classes, variáveis de folga, denotadas por $\vec{\xi}_i$, são associadas a cada amostra de dados. Essa associação possibilita que algumas das amostras de treinamento se localizem na face do hiperplano oposta à face que representa sua verdadeira classe (ou seja, sejam classificadas como novidade) quando uma menor distância à origem é escolhida. Uma amostra de treinamento é chamada de vetor de suporte quando ela é incorretamente classificada como novidade durante o treinamento ou se localiza na descrição do hiperplano. As duas abordagens (SVDD e *one-class* SVM) são equivalentes para funções de kernel não lineares, como a função de base radial (RBF) (HOFFMANN, 2007).

No intuito de separar os dados da origem com a margem máxima, deve-se minimizar a Equação (3.3)

$$\frac{1}{2}\|\omega\|^2 - \rho + \frac{1}{\nu\ell}\sum_{i=1}^{l}\vec{\xi}_i \tag{3.3}$$

onde ω é o vetor normal ao hiperplano de separação, ℓ é o número de amostras de treinamento e ρ é o *offset*, sujeito a

$$(\omega \cdot \Phi(x_i)) \geq \rho - \xi_i \quad i = 1,2,...,\ell \quad \xi_i \geq 0 \tag{3.4}$$

onde o operador $\cdot$ representa o produto interno entre dois vetores e $\Phi()$ é a função que mapeia um objeto do espaço original de características para o espaço de *kernel*.

Se ω e ρ resolvem este problema, então (3.5) computa de tal forma que se $f(z) > 0$, o exemplo z é classificado como normal. Caso contrário, z é classificado como novidade.

$$f(z) = sign\,((\omega \cdot \Phi(z))) - \rho) \tag{3.5}$$

Em (3.5), a função $sign()$ retorna 0 ou 1 indicando que uma dada instância z está dentro ou fora de determinada descrição.

Se $\rho > 0$, então o parâmetro $\upsilon \in (0,1)$ é o limite superior na fração de novidades no treinamento (erro de treinamento) e, também, um limite inferior na fração de vetores de suporte. O dual desse problema é:

$$\frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j k(\Phi(x_i),\Phi(x_j)) \tag{3.6}$$

sujeito a

$$0 \leq \alpha_i \leq \frac{1}{\nu\ell} \quad e \quad \sum_i \alpha_i = 1 \tag{3.7}$$

e agora a função discriminante é

$$f(x) = sign\left(\sum_{i=1}^{\ell}\alpha_i k(\Phi(x_i),\Phi(z)) - \rho\right) \tag{3.8}$$

e ρ pode ser calculado de acordo com

$$\rho = \sum_{i=1}^{l}\sum_{j=1}^{l}\alpha_i\alpha_j k(\Phi(x_i),\Phi(x_j)), \tag{3.9}$$

onde $0 \leq \alpha_i, \alpha_j \leq \frac{1}{\nu\ell}$.

Os experimentos realizados nesta tese fizeram uso da biblioteca LibSVM versão 2.84. A LibSVM está disponível em http://www.csie.ntu.edu.tw/ cjlin/libsvm/ (acesso em 20/07/2014).

3.2.2 *Support Vector Data Description* (SVDD)

Dada uma base de dados contendo N objetos x_i, $i = 1,...,N$, o método SVDD (TAX, 2001) tenta encontrar a esfera de menor volume contendo todos (ou quase todos) exemplos de treinamento. A Equação (3.10) representa uma hiper esfera com centro em a e com raio R, minimiza-se o raio onde a variável C consiste no compromisso (*trade-off*) entre a simplicidade (ou volume da esfera) e o número de erros (número de exemplos de treinamento rejeitados).

$$F(R,a,\xi) = R^2 + C\sum_i \xi_i. \tag{3.10}$$

A Equação (3.10) deve ser minimizada de acordo com a seguinte restrição:

$$(x_i - a)^T (x_i - a) \leq R^2 + \xi_i \qquad \forall i, \xi_i \geq 0. \tag{3.11}$$

Incorporando as restrições de (3.11) em (3.10) se constrói a equação de Lagrange:

$$L(R,a,\alpha,\xi) = R^2 + C\sum_i \xi_i - \sum_i \alpha_i\{R^2 + \xi_i - (x_i^2 - 2ax_i + a^2)\} - \sum_i \gamma_i\xi_i. \tag{3.12}$$

Com os multiplicadores de Lagrange $\alpha_i \geq 0$ e $\gamma_i \geq 0$. Igualando as derivadas parciais a zero, novas restrições são obtidas.

$$\sum_i \alpha_i = 1; \tag{3.13}$$

$$a = \frac{\sum_i \alpha_i x_i}{\sum_i \alpha_i} = \sum_i \alpha_i x_i; \tag{3.14}$$

$$C - \alpha_i - \gamma_i = 0 \qquad \forall i. \tag{3.15}$$

Sendo $\alpha_i \geq 0$ e $\gamma_i \geq 0$, pode-se remover γ_i de (3.12) e usar a restrição de que $0 \leq \alpha_i \leq C$, $\forall i$.

Incorporando-se as restrições (3.13), (3.14) e (3.15) em (3.12) e a maximizando com respeito a α_i, tem-se:

$$L = \sum_i \alpha_i (x_i \cdot x_i) - \sum_{i,j} \alpha_i \alpha_j (x_i \cdot x_j). \tag{3.16}$$

A Equação (3.14) mostra que o centro da esfera é dado como uma combinação linear das instâncias escolhidas como vetores de suporte com seus respectivos pesos; definidos pelos multiplicadores de Lagrange (α_i) obtidos durante a otimização. Apenas as instâncias com $\alpha_i > 0$ são reconhecidas como vetores de suporte e serão necessárias para descrição da esfera. O raio

R da esfera pode ser obtido calculando-se a distância do centro da esfera a um vetor de suporte. Objetos com $\alpha_i = C$ atingiram o limite da esfera e estão fora da descrição normal dos dados (*i.e.*, são reconhecidos como novidade).

Para determinar se um objeto z, de teste, se encontra dentro da hiper esfera, sua distância ao centro dessa hiper esfera deve ser calculada. Um objeto de teste é aceito quando sua distância até o centro da hiper esfera é menor que o raio dessa, ou seja, $(z-a)^T(z-a) \leq R^2$. Reescrevendo essa equação em termos de vetores de suporte, objetos são aceitos quando a desigualdade (3.17) é satisfeita.

$$(z \cdot z) - 2\sum_i \alpha_i (z \cdot x_i) + \sum_{i,j} \alpha_i \alpha_j (x_i \cdot x_j) \leq R^2. \tag{3.17}$$

A Figura 3.2 ilustra um exemplo bi-dimensional de como o SVDD e o One-Class SVM mapeiam objetos (instâncias) do espaço de características original para o espaço de kernel. No caso do OCSVM, os objetos mais centrais à distribuição de dados original são mapeados para uma região longe do plano que os separa da origem no espaço de kernel. Esse plano é definido pelos objetos menos centrais à distribuição de treinamento. Objetos localizados mais próximos da origem do que do hiperplano são definidos como da classe novidade e se encontrarão fora da descrição dos dados. No caso do SVDD, os objetos mais centrais à distribuição de dados original são mapeados para o centro de uma esfera de raio R no espaço de kernel. Assim como no OCSVM, essa esfera é definida pelos objetos menos centrais à distribuição de treinamento e objetos localizados fora dessa esfera são reconhecidos como da classe novidade.

3.2.3 Least Squares One-Class SVM

O classificador *Least Squares One-class Support Vector Machines* (LSOCSVM) (CHOI, 2009) consiste em uma reformulação do OCSVM original com a finalidade de substituir o procedimento da otimização quadrática, normalmente adotado por classificadores da família SVM, pela otimização utilizando o método dos mínimos quadrados (*Least Squares*). Essa reformulação visa minimizar a função objetivo apresentada em (3.18) sujeita a $\omega \cdot \phi(x_i) = \rho - \xi_i$, onde ϕ mapeia os objetos para o espaço de kernel, $\omega \cdot \phi(x_i) = \rho$ representa o hiperplano no espaço de kernel, $\| \cdot \|$ denota a norma Euclidiana, $\phi(x_i)$ é a variável de folga para o i-ésimo exemplo de treinamento e o parâmetro C é predefinido e controla a fração de exemplos rejeitados durante o treinamento, de maneira similar ao parâmetro ν em (3.3).

$$\frac{1}{2} \| \omega \|^2 - \rho + \frac{1}{2} C \sum_{i=1}^{n} \xi_i^2. \tag{3.18}$$

Tendo desenvolvido uma formulação adequada da solução usando equações lineares que satisfazem as restrições do OCSVM original, o *offset* (ρ) do hiperplano e os multiplicadores de lagrange (α) podem ser computados de maneira simples utilizando (3.19) e (3.20), respectivamente.

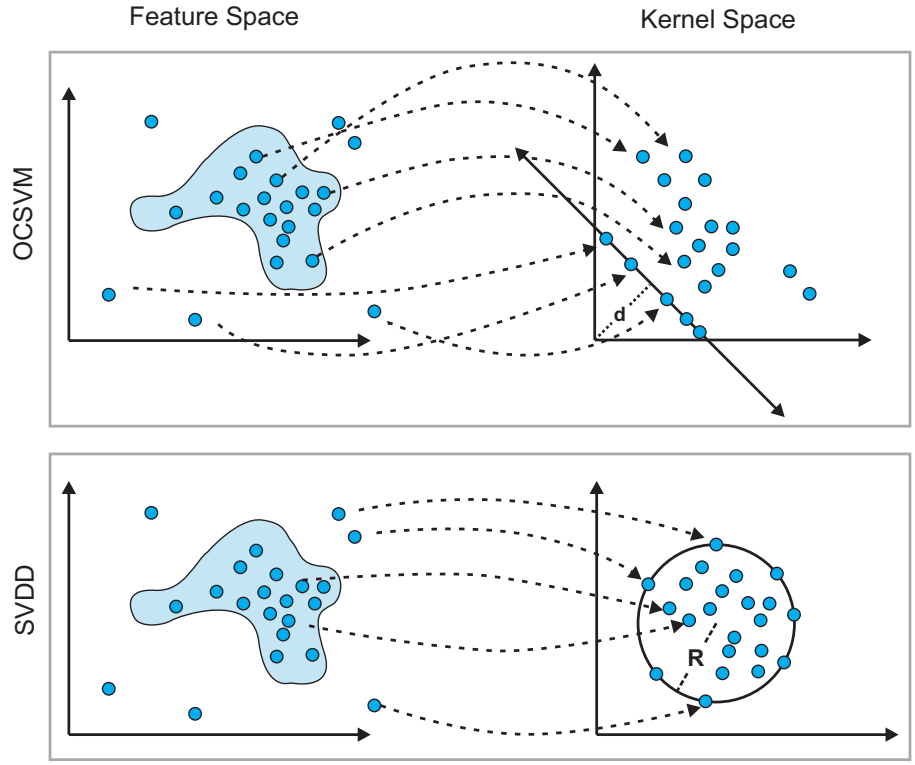

Figura 3.2: Mapeamento de objetos feito pelos métodos OCSVM e SVDD do espaço de características original ao espaço de kernel.

$$\rho = \frac{1}{e'((I/C)+K)^{-1}e}, \tag{3.19}$$

$$\alpha = \frac{((I/C)+K)^{-1}e}{e'((I/C)+K)^{-1}e}. \tag{3.20}$$

Em (3.19) e (3.20), e representa um vetor coluna, *i.e.* $[1, 1, ..., 1]'$, K representa a matriz de kernel com entradas $k_{i,j} = \kappa(x_i, xj)$ e I representa a matriz identidade. Dado um exemplo z de teste, a Equação (3.21) computa a distância entre z e o hiperplano na origem. Em (3.21), k' denota o vetor contendo as saídas da função $k(z, SV)$, onde SV representa o conjunto de vetores de suporte.

$$f(z) = \frac{k'((I/C)+K)^{-1}e - 1}{e'((I/C)+K)^{-1}e}. \tag{3.21}$$

3.2.4 Kernel PCA

Em (HOFFMANN, 2007), Hoffmann propõe um método com a finalidade de fornecer erros de classificação mais baixos devido à geração de uma descrição, em geral, mais justa que a do One-Class SVM. O autor introduz a aplicação de uma versão de kernel do PCA (*Principal Component Analysis*) canônico para resolver o problema da detecção de novidades no contexto da OCC. Em contraste com o PCA original (onde os autovetores e autovalores são gerados a

partir da matriz de covariância gerada pelo simples produto interno entre as características do problema), o kernel PCA computa a matriz de covariância a partir de uma matriz n x n (sendo n o número de exemplos de treinamento) em um espaço de alta dimensionalidade $\mathscr{F}$, onde cada célula dessa matriz é um resultado derivado da função de kernel para dois exemplos de treinamento, ou seja, $\Phi(x_i, x_j)$.

No kernel PCA, um autovetor V da matriz de covariância em $\mathscr{F}$ é uma combinação linear de pontos $\Phi(x_i)$,

$$V = \sum_{i=1}^{n} \alpha_i \tilde{\Phi}(x_i), \tag{3.22}$$

onde Φ mapeia um dado objeto para o espaço de kernel, com

$$\tilde{\Phi}(x_i) = \Phi(x_i) - \frac{1}{n} \sum_{r=1}^{n} \Phi(x_r). \tag{3.23}$$

A Equação (3.23) garante que os pontos no hiper espaço estarão centrados na origem. Dessa forma, é possível se obter a matriz de kernel normalizada $\tilde{K}_{i,j}$ com base na matriz de kernel $K_{i,j}$ (não normalizada), de acordo com a Equação (3.24).

$$\tilde{K}_{i,j} = K_{i,j} - \frac{1}{n} \sum_{r=1}^{n} k_{i,r} - \frac{1}{n} \sum_{r=1}^{n} k_{j,r} + \frac{1}{n^2} \sum_{r,s=1}^{n} k_{r,s} \tag{3.24}$$

Uma vez que o PCA canônico seja computado para a matriz $\tilde{K}_{i,j}$, a medida de novidade de um exemplo z é calculada de acordo com o erro de reconstrução (*reconstruction error*) proposto em (DIAMANTARAS; KUNG, 1996).

3.2.5 Processo Gaussiano Aplicado a OCC (*Gaussian Process for OCC*)

Em (KEMMLER; RODNER; DENZLER, 2013), *Gaussian Process Priors* foram aplicados com sucesso no paradigma OCC. Os autores propõem o uso de um processo gaussiano adequado no intuito de derivar funções de vizinhança eficientes. A Figura 3.3 mostra um exemplo da aplicação de regressão, utilizando-se Gaussian Process, a um problema com uma única dimensão. Pode ser visto que essa utilização leva a uma função de média a posteriori que apresenta altos valores (próximos a y = 1) em áreas de alta densidade (*i.e.*, próximas aos exemplos de treinamento). Observa-se também que os valores dessa função decrescem de forma contínua se a distância aos exemplos de treinamento aumenta. Adicionalmente, a Figura 3.3 mostra também que a variância tem um comportamento oposto, em regiões menos densas ela aumenta. Como conseqüência, a média e variância negativa podem ser usadas como medidas confiáveis para OCC.

Kemmler *et al.* (KEMMLER; RODNER; DENZLER, 2013) experimentaram quatro medidas de novidade diferentes para obter os *scores* que indicam se um objeto é ou não da classe novidade: a média (μ_*), a variância negativa ($-\sigma_*^2$), a probabilidade preditiva (P) e uma

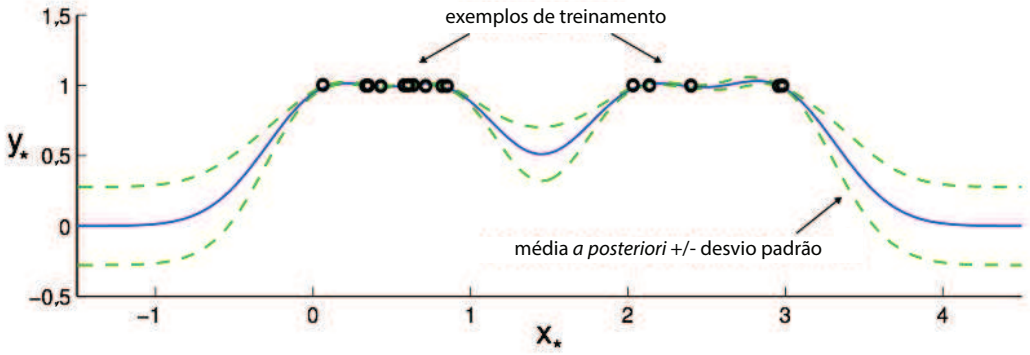

Figura 3.3: Regressão utilizando-se Gaussian Process em um problema no espaço unidimensional. A distribuição de predições (*scores*) é apresentada em termos da função de média e seus respectivos intervalos de confiança. (Figura extraída de (RODNER et al., 2011))

heurística (H). Sendo X um conjunto de treinamento com objetos na forma (x, y), onde x é o vetor de características e y o rótulo da classe, e sendo (x_*, y_*) um objeto de teste, as Equações (3.25), (3.26), (3.27) e (3.28) ilustram cada medida citada, respectivamente.

$$\mu_* = \varepsilon(y_*|X,y,x_*) = k_*^T(K+\sigma_n^2 I)^{-1}y \tag{3.25}$$

$$-\sigma_*^2 = -\nu(y_*|X,y,x_*) = -(k_{**} - k_*^T(K+\sigma_n^2 I)^{-1}y) \tag{3.26}$$

$$P = p(y_* = 1|X,y,x_*) \tag{3.27}$$

$$H = \mu_* \cdot \sigma_*^{-1} \tag{3.28}$$

Nas Equações (3.25) e (3.26), $K = \kappa(X,X)$, $k_* = \kappa(X,x_*)$, $k_{**} = \kappa(x_*,x_*)$ e κ representam o kernel gaussiano.

Durante o treinamento, o único procedimento a ser realizado é a resolução do sistema de equações lineares $(K+\sigma_n^2 I)\alpha = y$ com y sendo um vetor n-dimensional de 1's. A solução desse sistema pode ser encontrada utilizando-se a decomposição de Cholesky. Após o treinamento, o valor estimado da média *a posteriori* é dado por $k_*^T\alpha$.

A Figura 3.4 ilustra os modelos de classificação obtidos pelas quatro diferentes medidas para uma base de dados hipotética. De acordo com essa figura, vê-se que a medida *variância* obtém a região de aceitação mais bem definida (*i.e.*, a intensidade da cor branca representa a força do *score* que indica se a região é reconhecida como da classe normal e para essa medida, essa região tem os limites mais nítidos). As medida *média*, por exemplo, tem a região de fronteira entre as classes mais suave. Com base na Figura 3.4, a variância foi escolhida como medida de novidade a ser utilizada nos experimentos.

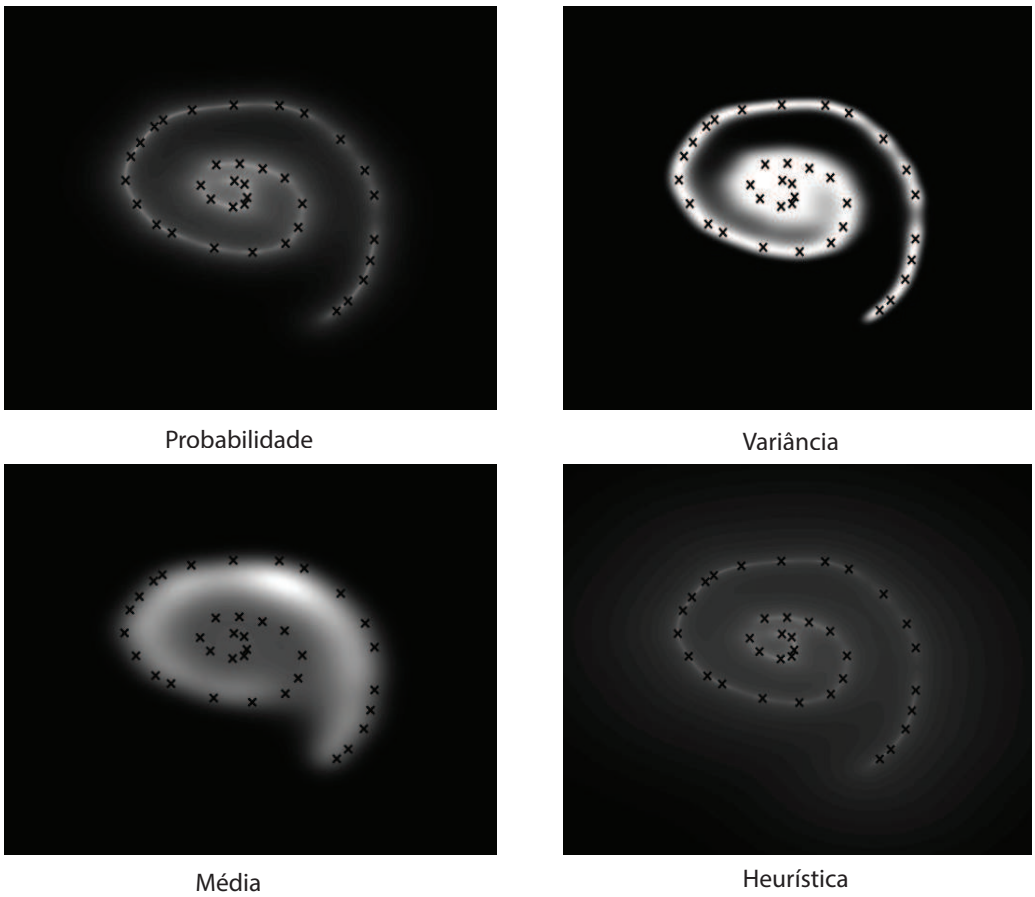

Figura 3.4: Exemplos de modelos gerados utilizando-se as quatro medidas experimentadas. (Extraído de: (KEMMLER; RODNER; DENZLER, 2013))

3.2.6 Condensed Nearest Neighbor Data Description - CNNDD

O algoritmo CNNDD (ANGIULLI, 2007) computa, com apenas duas iterações sobre a base de dados de treinamento, o sub-conjunto de referências mais significativas $RefSet$ de acordo com o algoritmo *Nearest Neighbor Data Description* - NNDD (TAX, 2001). O algoritmo recebe como parâmetros a métrica d, o número k de vizinhos mais próximos considerados, o limiar θ de aceitação (ou rejeição) de uma instância e o fator r da distância de Minkowski utilizada.

O algoritmo possui 3 fases que serão descritas a seguir:

Fase 1: Primeira Passagem pelo Conjunto de Treinamento.

Inicialmente, dois conjuntos auxiliares são utilizados para a criação de $Refset$ (que visa substituir o conjunto de treinamento original $Dataset$), *OutRefSet* e *InRefSet*. Ao final da primeira fase, *OutRefSet* deve conter as instâncias rejeitadas pela regra NNDD (*i.e.*, $NNDD_{OutRefSet \cup InRefSet,d,k,\theta,r}(p) > \theta, \forall p \in OutRefSet$ com base nas instâncias comparadas em $Dataset$) e *InRefSet* deve conter as instâncias que em algum momento foram rejeitadas pela regra NNDD, porém, no fim da primeira fase deixaram de ser (*i.e.*,

$NNDD_{OutRefSet \cup InRefSet,d,k,\theta,r}(p) \leq \theta, \forall p \in RefSet$).

Terminada a fase 1, $Refset = OutRefSet \cup InRefSet$.

Fase 2: Segunda Passagem pelo Conjunto de Treinamento.

A primeira passagem pela base de dados não assegura a consistência do $RefSet$ pois esse conjunto pode conter instâncias erroneamente classificadas com base nele mesmo. Por exemplo, um objeto pertencente a $OutRefSet$ e rejeitado pelo $RefSet$ não é uma novidade. Nessa fase, ainda existe a possibilidade de uma instância p, pertencente a $OutRefSet$, ser considerada normal dado que p pode não ter sido comparada a todas as instâncias de $Dataset$. Dessa forma, uma segunda busca pelos dados pertencetes a $Dataset$ - $RefSet$ é realizada. Sendo assim, a fase 2 atualiza apenas o conjunto $OutRefSet$.

Fase 3: Incremento de $RefSet$.

A terceira fase do algoritmo serve para garantir que o conjunto $RefSet$ tem o mesmo desempenho da regra NNDD. Com essa finalidade, o conjunto $RefSet$ é aumentado com o conjunto $IncrRefSet$ contendo vizinhos mais próximos de todas as instâncias p de $OutRefSet$, tal que p é considerada uma novidade (*i.e.*, $NNDD_{RefSet,d,k,\theta,r}(p) > \theta$). Dessa forma, $NNDD_{RefSet,d,k,\theta,r}(p) > \theta, \forall p \in InRefSet$, ou seja, a regra NNDD classifica corretamente todas as instâncias de treinamento.

3.2.7 Florestas Aleatórias para Classificação com Exemplos de uma Única Classe (*One-class Random Forests* - OCRF)

O método *One-class Random Forests* foi introduzido por Chesner Désir *et al.* (DéSIR et al., 2013) e consiste na conversão de um problema, originalmente, de classificação com exemplos de uma única classe em um problema binário (*i.e.*, com duas classes). A geração de dados da classe novidade (referidos como *outliers*) é feita a partir do histograma complementar de cada dimensão dos dados de treinamento e acontece em espaços de baixa dimensionalidade; no máximo 10 dimensões.

A Figura 3.5 ilustra partes importantes no processo de geração de *outliers* a partir dos dados normais. Inicialmente, (Figura 3.5 - a), dada uma das dimensões do problema, é gerado o histograma dos valores normais dessa dimensão (barras em azul) obtendo-se, também, o histograma complementar (barras em vermelho). Posteriormente, o histograma inicial, e seu complementar, são normalizados (Figura 3.5 - b), em relação ao eixo das ordenadas, e o histograma complementar (Figura 3.5 - c) é usado para a computação das probabilidades da geração de *outliers* em um determinado intervalo de valores (*bin*). Através da soma dos valores correspondentes a todos os *bins* do histograma complementar, e normalização entre 0 e 100 dessa soma, são geradas as probabilidades de cada *bin* (Figura 3.5 - d). Dessa forma, existe uma maior probabilidade de que novos *outliers* sejam atribuídos a intervalos de valores com menor

ocorrência na dimensão em questão (Figura 3.5 - d). Um valor aleatório é gerado e através da técnica de seleção por roleta é atribuído a um *bin*. Esse processo é realizado para todos as dimensões do problema.

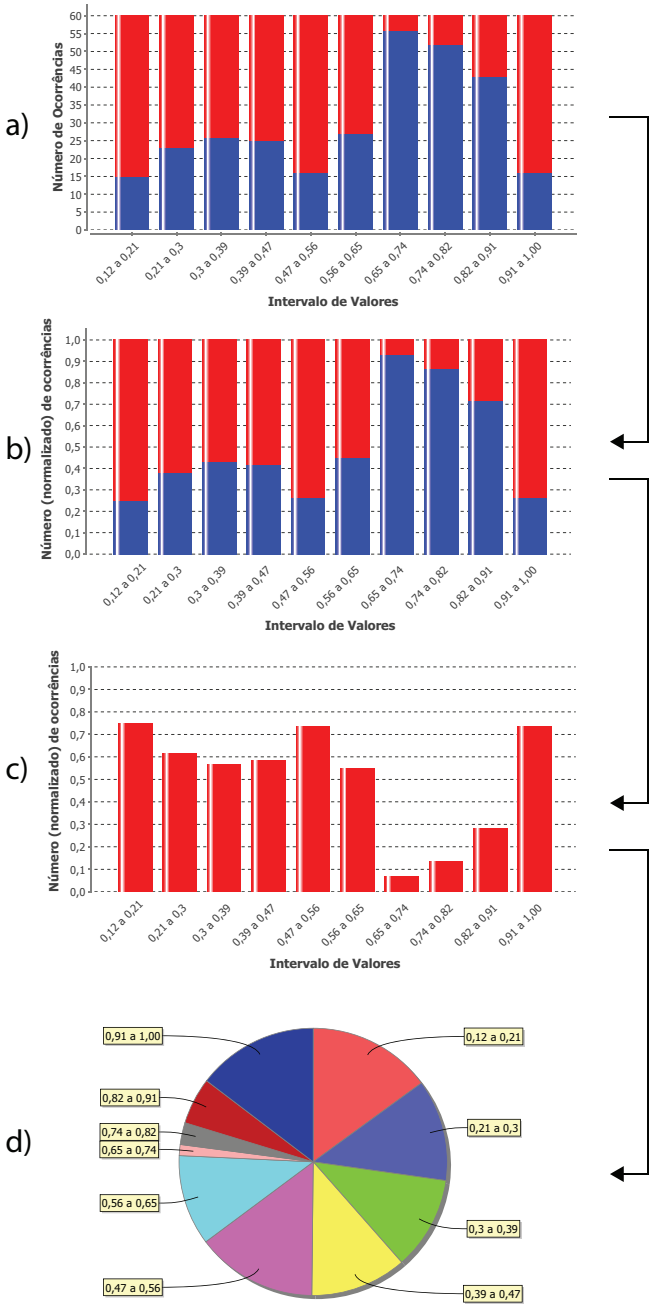

Figura 3.5: Sequência para a geração de outliers do OCRF realizada a partir de uma dimensão (característica) extraída de um problema hipotético. a) Histograma com a frequência de ocorrências para um determinado intervalo de valores (*bin*) normais (azul) e seu histograma complementar (vermelho); b) Histograma com a frequência de ocorrências normalizada entre 0 e 1; c) Histograma complementar ao histograma normal obtido em b); e d) probabilidades de geração de outliers para cada intervalo de valores do histograma.

Finalizada a geração de dados artificiais da classe novidade, o classificador Florestas Aleatórias padrão é então empregado para a resolução do problema binário na base de dados resultante. A ideia principal do OCRF é aplicar cada uma das árvores de decisão da floresta aleatória a um pequeno subconjunto de dimensões do problema, s_i, contendo instâncias da classe normal e novidade. Em (DéSIR et al., 2013), os autores variaram o tamanho de s_i de 2 a no máximo 10 dimensões, a depender da quantidade de dimensões do problema. A criação de *outliers* não previne a geração de ruídos, porém, esses podem ser tratados pelo método das Florestas Aleatórias (BIAU, 2012).

3.3 Características de Métodos para Classificação Com Exemplos de Uma Única Classe

Existem vários aspectos importantes relativos à detecção de novidades. Alguns deles são expressos de acordo com os seguintes princípios (MARKOU; SINGH, 2003):

a) *Princípio da robustez e do trade-off*: a detecção de novidades deve ser capaz de maximizar, no conjunto de teste, a exclusão de amostras da classe novidade e ao mesmo tempo minimizar a exclusão de amostras da classe normal;

b) *Princípio da minimização de parâmetros*: um método para detecção de novidades deve ter o menor número possível de parâmetros ajustáveis pelo usuário;

c) *Princípio da generalização*: o sistema deve ser capaz de generalizar sem confundir a informação generalizada com novidade (classificação de dados normais como novidades) (TAX; DUIN, 1998);

d) *Princípio da independência*: a detecção de novidades deve ser independente do número de atributos e classes disponíveis. O classificador deverá mostrar desempenho razoável no contexto de desbalanceamento de dados, baixo número de amostras e ruído.

e) *Princípio da adaptabilidade*: um sistema que reconhece novidades durante o teste deve ser capaz de usar essa informação para um re-treinamento (SAUNDERS; GERO, 2001).

f) *Princípio da complexidade computacional*: boa parte das aplicações de detecção de novidades são em tempo real, dessa forma, a complexidade computacional de um mecanismo de detecção de novidades deve ser o mais baixa possível.

É certo que existem vários outros métodos para OCC não abordados nesse trabalho. Porém, optou-se por usar apenas este conjunto de métodos por cobrir satisfatoriamente um amplo

universo de estratégias para OCC. Os métodos apresentados representam várias das abordagens disponíveis, porém, esse estudo não tem o intuito de enumerar todas as possibilidades de forma exaustiva. Características importantes de classificadores para OCC foram mencionadas acima. A seguir, cada aspecto introduzido acima é discutido no contexto dos métodos apresentados.

Princípio da robustez e do *trade-off*:

O fato de todos os classificadores trabalharem de forma a criar uma descrição que engloba os dados da classe normal deixando exemplos desconhecidos de fora, favorece a maximização da exclusão de instâncias da classe novidade e ao mesmo tempo minimiza a exclusão de instâncias da classe normal. Além disso, esse princípio é tratado de forma natural por todos os classificadores aqui contemplados, dado que eles compartilham o uso de limiares de aceitação θ das classes normal e novidade. Esses limiares podem ser ajustados de forma a aumentar ou diminuir o rigor na aceitação de exemplos como sendo da classe normal. Dessa forma, todos os classificadores aqui experimentados possuem um bom, e equiparável, nível de robustez.

Princípio da minimização de parâmetros:

Para efeito de poder de discriminação, o limiar de aceitação θ não é considerado um parâmetro dado que ele não influi no principal critério que avalia o poder de discriminação dos métodos utilizados nesse trabalho, a AUC (*Area Under the* (ROC) *Curve*) (FAWCETT, 2006). A AUC é computada com base em todos os valores relevantes (*i.e.,* que afetam o desempenho do modelo) de parâmetros θ de um dado modelo. Dessa forma, para cada classificador, todos os possíves valores para θ são considerados.

- *One-class SVM*: assim como a vasta maioria dos trabalhos que utilizaram OCSVM, esta pesquisa opta por utilizar o kernel Gaussiano. Dessa forma, para o uso desse classificador devem se atribuir os parâmetros σ (largura da função gaussiana) e C (fração de exemplos da classe normal rejeitada pelo classificador). Um maior valor para σ deixa a descrição menos justa à distribuição de treinamento enquanto que um maior valor de C rejeita uma maior quantidade de exemplos de treinamento.

- *SVDD*: o SVDD possui dois parâmetros: C e s. O parâmetro C controla o percentual de instâncias de treinamento a serem rejeitadas pela descrição. O parâmetro s se rerefe à largura do kernel gaussiano. Quanto maior o valor de s, menos justa será a descrição dos dados.

- *Least Squares One-class SVM*: possui os mesmos parâmetros do OCSVM original com os mesmos efeitos na classificação.

- *GPP-OCC* : o GPP-OCC contém apenas o parâmetro σ que define a largura da função Gaussiana. Se esse parâmetro for aumentado ocorrerá uma maior aceitação de exemplos distantes do conjunto de treinamento, se for diminuído demasiadamente pode ocorrer o *overfitting*.

- *kernel PCA*: o kernel PCA utiliza os parâmetros σ (largura do kernel) e o número k de características retornadas pelo PCA. Como o PCA atua sobre uma matriz quadrática de tamanho (n x n), onde n é o número de exemplos de treinamento, k varia de *1* a n. Quanto maior o k, maior o ajuste do modelo à distribuição de treinamento.

- *CNNDD*: o CNNDD possui apenas o parâmetro k, quantidade de vizinhos, a ser ajustado. O aumento no valor desse parâmetro implica na diminuição do efeito de ruídos na classificação, porém, pode resultar em descrições menos justas à distribuição de treinamento. Outros parâmetros do algoritmo (como a métrica d e o fator r de Minkowski) foram fixados assim como em (ANGIULLI, 2007). A atribuição do parâmetro θ não é necessária dado que esse parâmetro, quando utilizado no presente trabalho, é obtido de forma automática.

- *One-class Random Forests*: o OCRF usa como parâmetros o número de árvores a serem combinadas para a tomada de decisão e o quantidade q de *outliers* gerados a partir do conjunto de treinamento. O valor de q é gerado em função da quantidade n de instâncias de treinamento (*i.e.,* $q = k * n$, sendo k um inteiro maior que 1).

É interessante notar que todos os métodos acima citados possuem no máximo 2 parâmetros. Esse fato simplifica a utilização de tais classificadores aumentando assim as chances de serem alcançados os melhores resultados para cada par: classificador x problema.

Princípio da generalização:

Esse princípio é tratado de forma diferente por cada método contemplado nesse estudo. Todos os métodos aqui discutidos têm o mesmo propósito, porém, não existe um método que resolva da melhor maneira todos os problemas. Logo, todos os classificadores abordados possuem um bom nível de generalização, porém, a busca pelo melhor classificador no geral visa nos dar a ideia de quais métodos melhor se adaptam a variadas situações.

Princípio da independência:

O princípio da independência cita os aspectos: atributos e classes disponíveis; desbalanceamento; baixo número de amostras; e ruído. Os aspectos baixo número de classes disponíveis e desbalanceamento são tratados de forma simples pelo paradigma OCC. O baixo número de amostras e atributos afeta todos os métodos da mesma forma, não existe nenhum tipo de tratamento por parte dos métodos. Em se tratando de ruído nos dados, apenas o GPP-OCC não oferece solução ao problema.

Princípio da adaptabilidade:

A adaptabilidade é um item muito importante em sistemas *online*, principalmente. Esse é um aspecto não tratado por nenhum método analisado nesse trabalho. Porém, de interesse para trabalhos futuros.

Princípio da complexidade computacional:

Em relação à complexidade computacional, a diferença básica entre os métodos consiste na redução ou não de protótipos durante o treinamento. Essa redução gera custo computacional extra na fase de treinamento, porém reduz o custo na fase de teste. Os métodos *One-class* SVM, SVDD e CNNDD realizam a redução de protótipos durante o treinamento fazendo com que o custo computacional na fase de teste seja reduzido. O método *kernel* PCA armazena k componentes principais que são vetores coluna de tamanho n (sendo k o número de características retornadas pelo método e n o número de exemplos de treinamento). Dessa forma, o custo computacional da fase de teste do *kernel* PCA varia de acordo com os valores k e n. Os métodos GPP-OCC e *Least Squares One-class* SVM não realizam a redução de protótipos. Isso faz com que a fase de teste desses métodos seja mais custosa.

3.4 Considerações Finais

Métodos para classificação com exemplos de uma única classe se baseiam em três diferentes abordagens: estimativa de densidade, estimativa de fronteira entre as classes e reconstrução dos dados. A estimativa de densidade retorna a mais completa descrição dos dados, mas pode requerer um grande armazenamento de dados. Para uma menor descrição dos dados um método que estima a fronteira entre as classes pode ser mais adequado. O método de reconstrução, por sua vez, tem a habilidade de incorporar conhecimento *a-priori* (extra) do problema.

A maioria dos métodos para classificação, baseados em exemplos de uma única classe, utiliza a abordagem de modelar conjuntos de dados da classe normal e assim estimar a probabilidade de um exemplo de teste pertencer a essa classe. Em classificadores desse tipo, não é preciso especificar ou fazer suposições relativas à natureza dos dados. Em contrapartida, o montante de dados no treinamento e a qualidade dos mesmos se tornam cruciais para a criação de um modelo de bom desempenho.

4

Método Proposto

Este Capítulo apresenta as duas versões do método proposto nesta tese. As duas versões do método proposto são ligeiramente diferentes, porém essa diferença resulta em um desvio comportamental durante a fase de modelagem. Esse desvio traz consigo uma diferença considerável em relação ao custo computacional na prática, esse aspecto será avaliado através de uma análise do custo computacional. Definido o custo computacional do método, uma heurística para a melhoria do tempo de treinamento é proposta. Uma discussão sobre o ajuste de parâmetros será conduzida e alternativas automáticas para a obtenção de valores confiáveis de parâmetros são apresentadas.

4.1 Justificativa

Duas abordagens básicas para a implementação de métodos de detecção de novidades são a *Signature Detection* (Detecção Rotulada) e a Classificação com Exemplos de uma Única Classe. Na detecção rotulada, informações sobre todas as classes são fornecidas durante o treinamento, inclusive a classe novidade (*i.e.*, a detecção rotulada consiste na detectão de novidades implementada por métodos de classificação com múltiplas classes). Essa abordagem é a mais indicada quando a classe a ser detectada tem comportamento conhecido e exemplos desse comportamento estão disponíveis. Alarmes falsos são raros no uso da deteção rotulada, pois o método sabe exatamente as condições em que ocorrem um evento novidade. Por outro lado, a detecção rotulada não tem a capacidade de reconhecer eventos não apresentados durante a fase de modelagem. Embora sistemas baseados em OCC possam produzir mais alarmes falsos do que métodos desenvolvidos a partir de detecção rotulada, eles têm a importante vantagem de serem capazes de detectar novos comportamentos desconhecidos durante a fase de modelagem. Dessa forma, a abordagem OCC se torna mais adequada à maioria dos problemas de detecção de novidades e foi então escolhida para o desenvolvimento do método proposto.

O classificador GP-OCC é um exemplo de método para detecção de novidades que não possui uma fase de modelagem, o modelo do classificador consiste basicamente no armazenamento do conjunto de treinamento. Esse aspecto ganha relevância à medida que o conjunto de

treinamento cresce e a classificação de um objeto z de teste passa a ficar mais custosa devido à comparação com todos os exemplos de treinamento e posterior ordenação desse conjunto. Em aplicações onde o tempo de resposta é decisivo, a utilização de métodos com essa característica pode ser inadequada.

Classificadores baseados em Máquinas de Vetores de Suporte, como o SVDD e *One-class* SVM, aplicados ao problema da detecção de novidades tratam na sua formulação o problema de ruídos nos dados. Outra vantagem destes métodos é a remoção de exemplos não significativos para a distinção entre as classes (*i.e.*, Redução de Protótipos). Porém, o custo computacional desses métodos durante a fase de treinamento pode se tornar impraticável com o aumento do número de exemplos de treinamento. O treinamento dessa classe de métodos consiste na resolução de um problema de otimização quadrática com restrições, onde a cada exemplo de treinamento são associadas variáveis a serem otimizadas assim como restrições ao problema. Métodos de otimização quadrática e mínimos quadrados são usados com sucesso na tarefa de otimização, porém, o custo computacional desses métodos pode se tornar bastante alto com o aumento de exemplos de treinamento.

Assim como citado anteriormente, reafirma-se que os métodos propostos nesta tese, visam a obtenção de um resultado similar ou melhor ao obtido pelos métodos no estado da arte, porém, com uma formulação mais simples e menos custosa computacionalmente.

4.2 O Algoritmo: FBDOCC

O método proposto nesta tese, chamado de FBDOCC (*Feature Boundaries Detector for One-Class Classification*), tem como ideia principal a exploração de todas as dimensões do problema para cada instância em um conjunto de treinamento contendo apenas exemplos da classe normal (*i.e.*, instâncias de classes conhecidas do problema) no intuito de encontrar os limites da classe normal que englobam de forma mais justa a distribuição de dados de treinamento. Seja $T = \{t_1^l, t_2^l, .., t_n^l\}$ um conjunto de treinamento contendo n vetores de tamanho l, o FBDOCC gera $2l$ protótipos artificiais para cada instância de treinamento t_i a uma pequena distância Euclidiana (definida pelo parâmetro r) de t_i.

Cada par (t_j, p_j), tal que, p_j é um protótipo artificial gerado por t_j, tem a finalidade de representar uma porção do limite entre as classes normal e novidade. A ideia é gerar uma hiper esfera[1] (com valor de raio definido pelo parâmetro th) centrada em cada protótipo p_j e checar se existe ou não alguma instância de treinamento (exceto t_i) localizada no interior dessa hiper esfera. Se todas as instâncias de treinamento se localizam fora da hiper esfera definida por p_j (*i.e.*, $\rho(t_i, p_l) > th \; \forall \; t_i \in T$), os seguintes procedimentos serão realizados: (i) informações mínimas, descritas a seguir, sobre p_j são armazenadas com a finalidade de se reproduzir esse

[1] A escolha pelo formato de hiper esferas se dá pelo fato de esse ser um formato que necessita apenas de seu centro e raio como informações para reprodução. Diferente de outros formatos que necessitam de mais informações (como o hiper retângulo, por exemplo).

protótipo como da classe positiva na fase de teste, e (ii) a instância de treinamento responsável pela criação de p_j é adicionada ao conjunto de instâncias negativas. Uma vez que informações sobre um dos $2l$ protótipos artificiais (positivos) são armazenadas os $2l - 1$ restantes são descartados. Os protótipos positivos definem a classe novidade enquanto que o conjunto de instâncias negativas representa a classe normal.

A fase de treinamento do FBDOCC é mostrada no fluxograma da Figura 4.1 e pseudo-código do Algoritmo 4. Vale ressaltar que o algoritmo armazena apenas as instâncias originais da classe normal e informações sobre os protótipos da classe novidade selecionados. Dessa forma o conjunto de protótipos artificiais pode ser recriado na fase de teste a partir do conjunto de instâncias da classe normal.

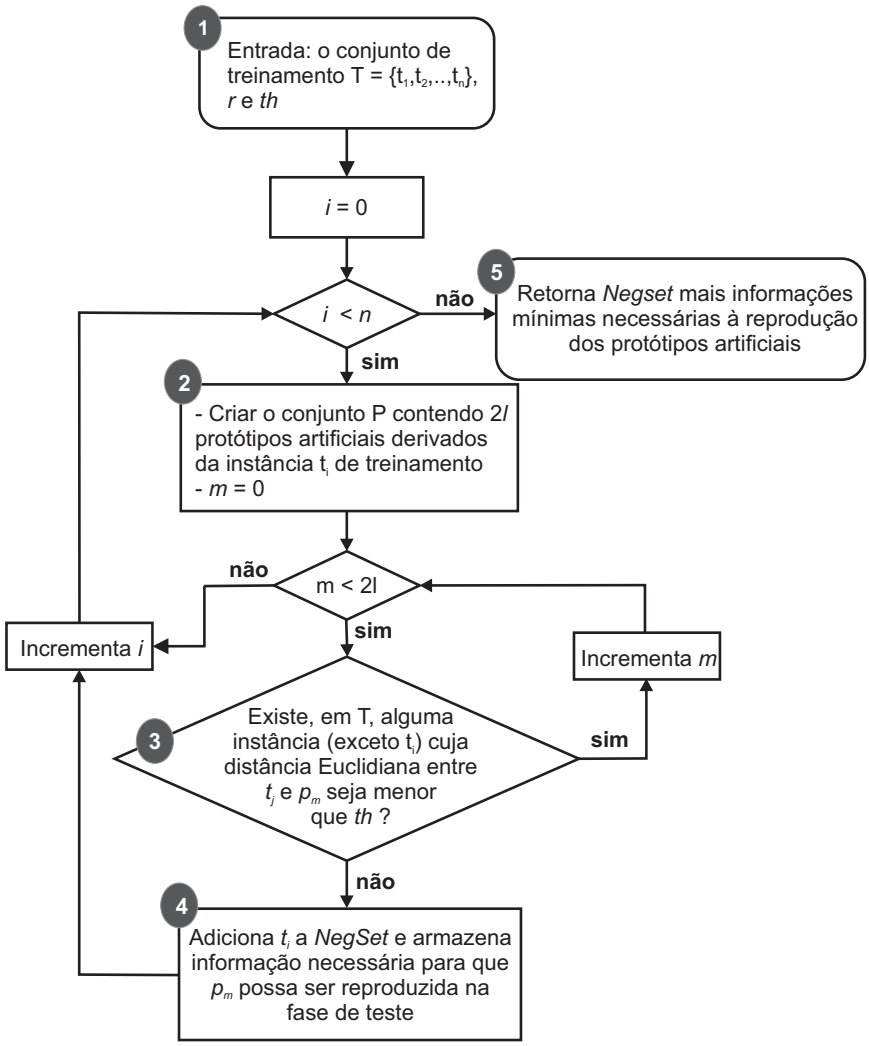

Figura 4.1: Fluxograma de operação do FBDOCC.

Algumas da fases do algoritmo são melhor detalhadas abaixo (ver índices na Figura 4.1):

- **(1)** O algoritmo recebe como entrada os parâmetros r e th e a base de dados T

contendo n instâncias da classe normal com l características, cada.

- **(2)** Para cada instância t_i, $2l$ protótipos da classe positiva serão gerados. Sendo t_i^j o valor da j-esima característica de t_i, dois novos protótipos são criados repetindo-se os valores de todas as características, exceto t_i^j. Os novos objetos são, então, $p_j = (t_i^1, t_i^2, .., t_i^j + r, .., t_i^l)$ e $p_{j+1} = t_i^1, t_i^2, .., t_i^j - r, .., t_i^l$. Esse procedimento é realizado para todas as características da instância t_i, gerando o conjunto P de protótipos para essa instância.

- **(3)** Para cada protótipo $p \in P$, checar se existe alguma instância de treinamento dentro da hiper esfera com centro em p e raio th.

- **(4)** Se existe algum protótipo $p \in P$ tal que nenhuma instância de treinamento está localizada dentro da hiper esfera definida por p com raio th, a instância de treinamento t_i (geradora de p) é adicionada a *NegSet* informações básicas (o índice da característica e o sinal de adição ou subtração) são armazenadas para a reconstrução de p no futuro. p pode ser gerada posteriormente somando-se ou subtraindo-se r do valor da característica cujo índice foi armazenado, as outras características terão valor idêntico aos de t_i.

- **(5)** Após percorrer todas as instâncias de treinamento, o algoritmo retorna o conjunto *NegSet* de instâncias na borda e os dados necessários para a reprodução dos protótipos da classe positiva.

Terminada a fase de treinamento, o modelo consiste do conjunto *NegSet*, que é formado por instâncias da classe normal (negativa), e de uma estrutura de dados contendo informações necessárias à posterior reprodução dos protótipos da classe positiva. O conjunto de protótipos da classe positiva, disponível após a reconstrução dos mesmos, é a partir de agora referido como *PosSet*. Sendo R^{emp} o risco empírico (*i.e.*, erro de treinamento), o problema a ser resolvido pelo FBDOCC pode ser visto como a busca por parâmetros que minimizem o tamanho do conjunto *NegSet* mantendo R^{emp} igual a zero, como formulado abaixo:

$$\begin{array}{ll} \text{minimizar} & \#NegSet, \\ \text{sujeito a} & \rho(t_i, p_i) > th \;\forall\; p_i \in PosSet, \forall\, t_i \in T \\ & R^{emp} = 0. \end{array}$$

Na Figura 4.2 são apresentados exemplos de protótipos gerados a partir de uma instância de treinamento t_i representada pelo ponto central na cor azul. Na Figura 4.2 a), os protótipos são gerados para um problema bidimensional com base nos parâmetros r e th. Nesse caso $l = 2$ resultando na criação de $2l$ protótipos artificiais. A Figura 4.2 b) ilustra um exemplo tridimensional onde cada um dos 6 protótipos gerados define uma esfera. Essa figura ilustra o efeito dos parâmetros r e th na criação do modelo.

Algoritmo 4: FBDOCC (Fase de Treinamento)

```
   Input: T, r, th
   Output: NegSet
1  foreach t_i in T do
       /* P é o conjunto contendo todos os protótipos positivos artificiais gerados
          pela instância de treinamento t_i                                          */
2      P ← {};
       /* x_i representa o valor do i_th atributo de t_i                             */
3      foreach x_i in t_i do
4          foreach op em { +, - } do
5              p = [ x_1, x_2, ... (x_i op th) ..., x_n ] ;
6              P = P ∪ {p};
       /* embaralhar o conjunto P possibilita a obtenção de diferentes soluções para
          uma mesma configuração de parâmetros.                                      */
7      shuffle(P);
8      ctProt = 0;
9      foreach p_j in P do
10         contém = false;
11         foreach q_z em T do
12             if q_z é diferente de t_i then
13                 dist = EuclidDist(p_j, q_z);
14                 if dist < th then
15                     contém = true;
16                     break;
           /* se nenhuma instância de treinamento se encontra dentro da hiper esfera
              com centro em p_i e com raio th                                        */
17         if contém == false then
18             cPtrot = ctProt + 1;
               /* A restrição do próximo comando condicional evita que uma instância
                  de treinamento ruidosa seja adicionada ao conjunto NegSet          */
19             if ctProt + (l - j) <= l then
                   /* Atualiza t_i de forma a armazenar informações necessárias à
                      reprodução do primeiro protótipo p_j de P tal que EuclidDist(t_i, p_j)
                      > th ∀ t_i ∈ T                                                 */
20                 NegSet = NegSet ∪ t_i;
21                 break;
```

A Figura 4.3 ilustra um exemplo dos limites de decisão obtidos por um modelo gerado a partir de uma distribuição bi-dimensional em formato de banana. Os pontos internos à descrição e em azul, representam instâncias negativas armazenadas pelo método no conjunto *NegSet*, os pontos externos em vermelhos representam os protótipos positivos recriados a partir das instâncias negativas e os internos à descrição e em escala de cinza representam as instâncias de treinamento descartadas. Como citado anteriormente, os círculos com linhas tracejadas são gerados para cada protótipo artificial durante a fase de treinamento no intuito de checar se o protótipo contém ou não pelo menos uma instância de treinamento. Nessa figura, a superfície de decisão é gerada por um classificador *1*NN.

4.3 FBDOCC2: Outra versão do método.

Na versão inicial do FBDOCC, apenas um protótipo p da classe novidade pode ser recriado a partir de cada instância do conjunto *NegSet*. Isso significa dizer que p foi criado no intuito de checar apenas uma única região do limite dos dados normais em relação a apenas

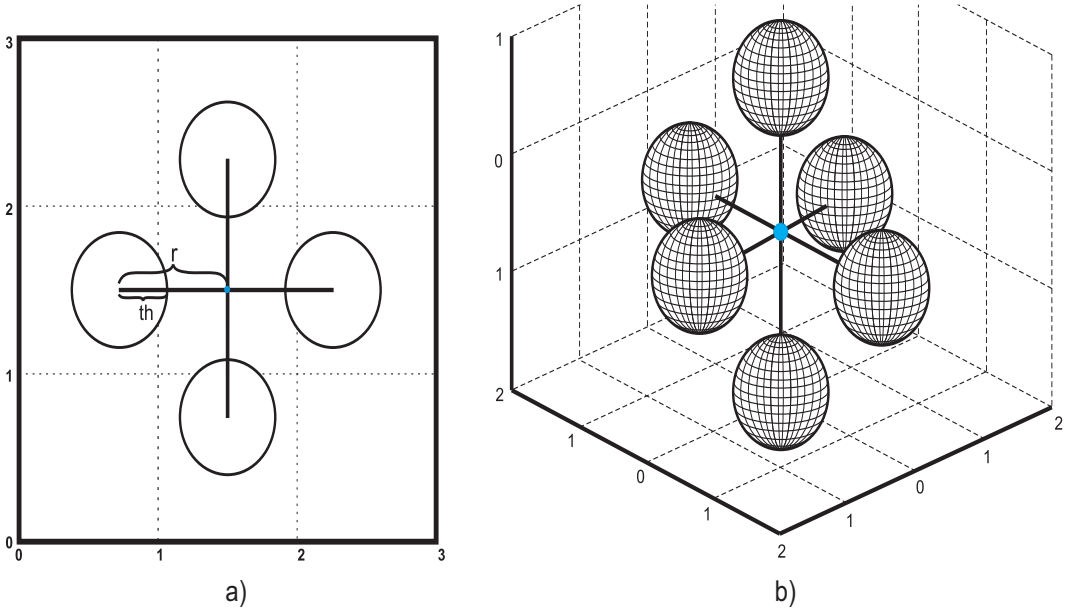

Figura 4.2: Exemplos de protótipos gerados a partir de uma instância de treinamento para problemas bidimensionais e tridimensionais.

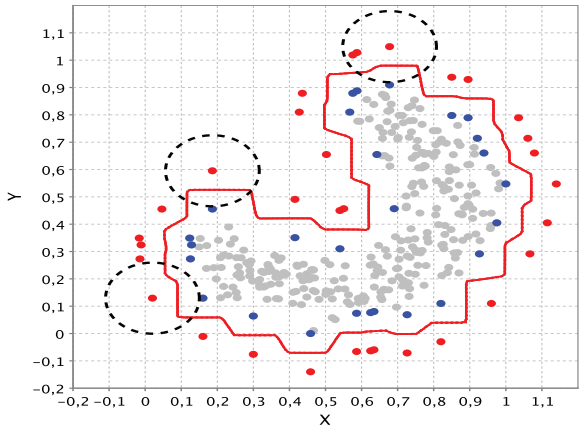

Figura 4.3: Exemplo de superfície de decisão gerada para a base de dados banana.

uma dimensão do problema. Na verdade, cada par (instância negativa, protótipo positivo) define uma reta e todos os pares combinados definem uma supéficie complexa. Pode-se pensar no caso de a instância negativa estar localizada em uma região convexa de forma a possibilitar o armazenamento de mais de um protótipo positivo por instância negativa. A Figura 4.4 apresenta uma distribuição no formato de um quadrado. Para essa distribuição, o FBDOCC padrão escolherá pela retenção de apenas um protótipo, o protótipo à esquerda, à direita, abaixo ou acima da instância localizada na esquina da distribuição. A versão FBDOCC2 é capaz de reter todos os protótipos positivos de uma instância de treinamento cujas respectivas hiper esferas geradas não contenham nenhuma instância de treinamento. As regiões superiores à esquerda de cada descrição mostram a associação entre cada protótipo e sua respectiva instância negativa.

No caso do FBDOCC2, vê-se que uma única instância negativa está associada a dois protótipos positivos.

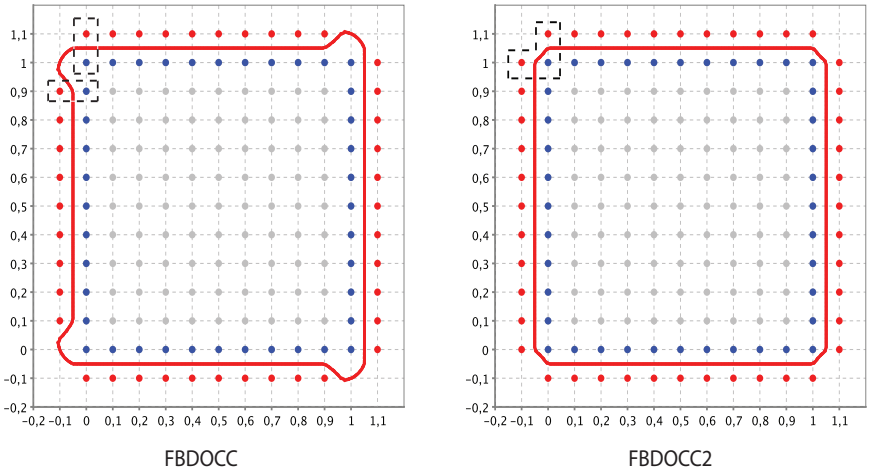

Figura 4.4: Superficies de decisão geradas pelos métodos FBDOCC e FBDOCC2, respectivamente, para uma distribuição no formato de um quadrado.

Em relação ao algoritmo, a diferença entre as duas versões ocorre na fase 4. Na versão FBDOCC2, o algoritmo pode ir da fase 4 para o passo *incrementa m*. Pelo bem da simplicidade na fase de teste, essa versão armazena o conjunto de instâncias negativas *NegSet* e o conjunto de protótipos positivos *PosSet* como modelo. Na Figura 4.4, instâncias negativas nas extremidades da descrição (esquinas) estão associadas a mais de um protótipo da classe positiva, no caso do FBDOCC2. Em comparação com o FBDOCC, essa diferença pode gerar uma superfície de decisão mais bem adaptada à distribuição de dados. Porém, essa característica não necessariamente garante um melhor desempenho na classificação. Quanto ao custo computacional, na prática, essa versão requer um maior esforço computacional; como será visto nos experimentos.

4.4 Fase de Teste

Uma vez que o treinamento tenha sido realizado, o problema em questão deixa de ser *One-class*, torna-se um problema binário e a fase de teste consiste na aplicação do algoritmo *k*NN. Se para um dado exemplo z de teste e um modelo contendo o conjunto reduzido de instâncias *NegSet* e o conjunto *PosSet* de protótipos da classe positiva, gerado a partir de *NegSet*, o resultado de (4.1) for menor que 1, z é classificado como da classe novidade. Caso contráro, z é aceito como normal. A ideia de (4.1) é simples, caso o z esteja mais próximo de protótipos da classe novidade, ele é classificado como tal, caso contrário, é aceito como normal.

$$f(z) = \frac{\sum_{i=1}^{k} DistEuclid(PosSet_i, z)}{\sum_{i=1}^{k} DistEuclid(NegSet_i, z)}. \tag{4.1}$$

O algoritmo *k*NN, por sua vez, gera fronteiras rígidas entre as classes para um valor de k baixo. A Figura 4.5 apresenta as várias descrições geradas conforme o parâmetro k na fase de testes para o mesmo modelo. De acordo com a Figura 4.5, nota-se que o valor do parâmetro k não é de difícil atribuição; dada a semelhança dos modelos descritos para valores de k maiores que 1. Nota-se que à medida que seu valor aumenta a descrição da classe normal se torna sutilmente mais suave, porém, valores muito altos podem distorcer essa descrição. Analisando a Figura 4.5, para esse exemplo, vê-se que o valor $k = 3$ gera uma descrição suave e justa; considerando a distância da fronteira ao conjunto de treinamento. Dessa forma, por simplicidade, esse valor será adotado nos experimentos.

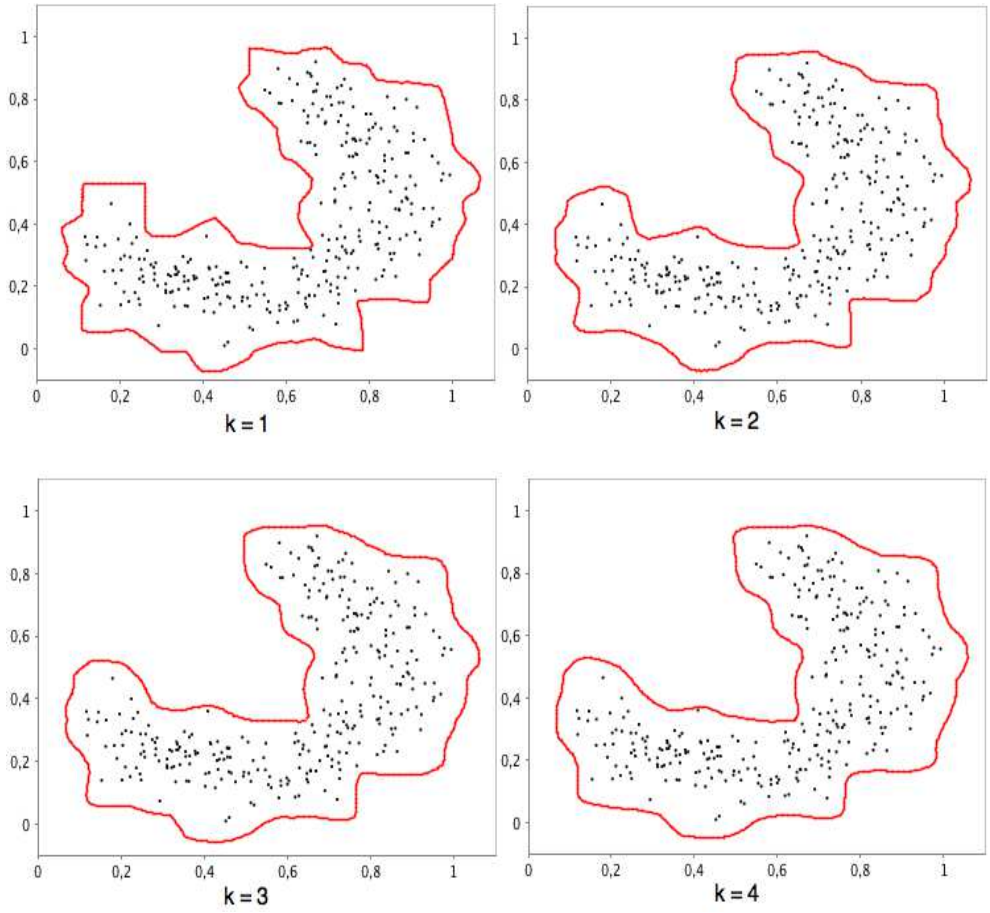

Figura 4.5: Variação das superfícies de decisão de acordo com a variação do k durante a fase de teste.

4.5 Atribuição de Parâmetros

A busca pela melhor configuração dos parâmetros r e th pode ser feita de duas formas: empírica (ou seja, manual) ou através de algum método para otimização de parâmetros. Essa seção ilustra os dois casos.

4.5.1 Atribuição Empírica

A atribuição de parâmetros é uma tarefa que varia de acordo com cada base de dados. A Figura 4.6 ilustra quatro exemplos de configurações de parâmetros para uma base de dados bidimensional. Nela, as instâncias geradoras dos protótipos positivos estão na cor vermelha e os círculos definidos pelos protótipos demarca a área de cobertura de cada protótipo. A lista a seguir apresenta um detalhamento do efeito causado por cada configuração ilustrada na Figura 4.6.

- **Configuração 1**: A situação 1 mostra um caso onde a escolha dos parâmetros r e th geraram protótipos que conseguem detectar de forma correta instâncias na borda da distribuição dos dados.
- **Configuração 2**: Na situação 2, o valor atribuído ao parâmetro r é muito alto fazendo com que instâncias internas à distribuição sejam consideradas como na borda.
- **Configuração 3**: O caso 3 ilustra um exemplo onde o valor do parâmetro th é muito baixo fazendo com que a maior parte das instâncias seja considerada como na borda ou ruído.
- **Configuração 4**: A situação 4 ilustra um caso onde o valor do th é muito grande, fazendo com que a esfera não detecte de forma satisfatória instâncias na borda em regiões côncavas.

A normalização dos valores dos atributos entre 0 e 1 auxilia na atribuição de parâmetros de forma empírica devido à diminuição do espaço de busca por possíveis valores dos parâmetros.

4.5.2 Otimização de Parâmetros do Classificador

Métodos de otimização têm sido utilizados de forma extensiva na busca automática por parâmetros em vários problemas de classificação de padrões. No presente trabalho, foram experimentados os métodos Têmpera Simulada (KIRKPATRICK; GELATT; VECCHI, 1983) e PSO padrão (KENNEDY; EBERHART, 1995) (as técnicas para otimização de parâmetros aqui utilizadas serão detalhadas no Apêndice A). Dentre as duas abordagens, o PSO canônico obteve o melhor rendimento no geral. Dessa forma, o PSO original foi aplicado aos métodos propostos no intuito de encontrar a melhor configuração de parâmetros (r_{best}, th_{best}).

A abordagem de otimização utilizada, proposta na Equação (4.2), incorpora também a complexidade do modelo (*i.e.*, o tamanho do conjunto *NegSet*) como um fator de penalidade. Em outras palavras, f considera mais de um objetivo dado que ela tenta maximizar o valor da AUC de validação minimizando também a redução de protótipos. O fator de penalização C da complexidade do modelo, pode ser variado de acordo com a importância dada ao aspecto da

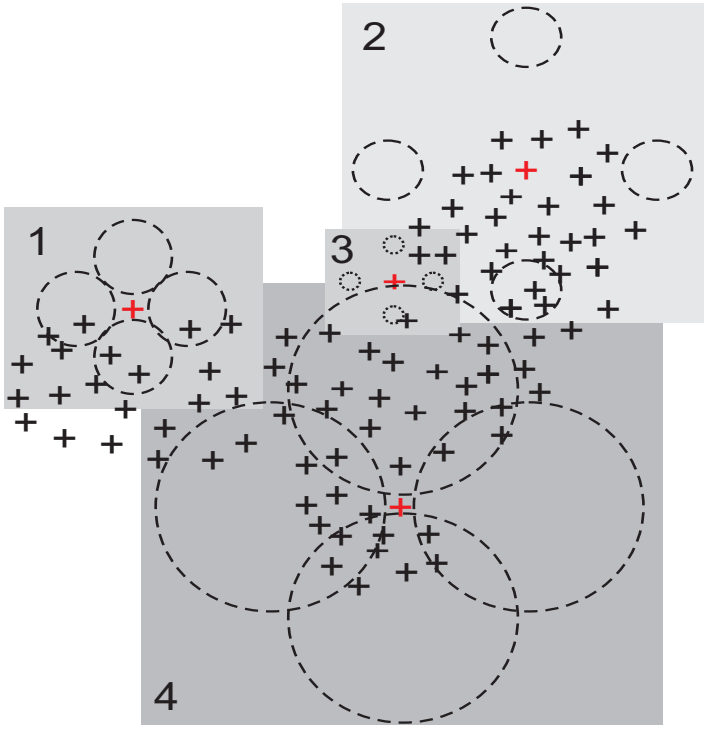

Figura 4.6: Ilustração de quatro casos distintos de combinações de parâmetros.

redução de protótipos. Se esse fator recebe um valor menor que 1, a maximização da AUC tem um maior peso na Equação (4.2) que a redução de protótipos, caso C ultrapasse 1, a redução de protótipos passa a ser mais relevante na otimização. O valor de penalidade máxima ocorre quando o modelo armazena 100% das instâncias de treinamento. Em (4.2), N representa o tamanho do conjunto de treinamento e $\#NegSet$ representa o número de instâncias armazenadas pelo modelo.

$$f = AUC_{val} - (C \cdot \frac{\#NegSet}{N}) \tag{4.2}$$

Nos experimentos, os parâmetros da versão FBDOCC2 foram otimizados também pela função de adaptação (4.2).

Os parâmetros utilizados pelos algoritmos de otimização estão descritos na Seção 5.1.2.

4.6 Análise do Custo Computacional

Em métodos de reconhecimento de padrões, frequentemente o custo computacional é dependente do problema abordado, ou seja, de acordo com a natureza dos dados, o tempo gasto para a construção do modelo pode variar de forma significativa. Em relação a ambas as versões do método proposto, o número de dimensões do problema l é um fator crítico. Com o aumento do número de dimensões, o tempo de treinamento do método tende a aumentar, porém,

a precisão na classificação não necessariamente é afetada. No próximo capítulo (Experimentos), serão apresentadas medições de tempos de treinamentos para vários problemas na prática.

Na teoria, as heurísticas apresentadas na próxima seção não alteram o custo computacional do método. Dessa forma, elas não serão consideradas nessa subseção.

Pior Caso Seja n o número de instâncias no conjunto de treinamento e l o número de dimensões do problema. O cenário onde o algoritmo atinge o pior custo computacional acontece quando todos os protótipos que foram reconhecidos como da classe positiva são os últimos do conjunto de protótipos P para cada instância de treinamento e cada instância de treinamento gera um protótipo. Nesse caso, o protótipo é adicionado apenas quando testado em todo o conjunto de treinamento resultando na análise de $2l$ protótipos sobre cada uma das n instâncias de treinamento.

Dessa forma, o pior caso ocorre quando para cada uma das instâncias de treinamento o laço interno executa $(2l \cdot n)$ operações. Isso resulta no custo de $(n^2 \cdot 2l)$ no pior caso, ou seja, a complexidade é de O$(l \cdot n^2)$.

A complexidade, no pior caso, é compartilhada por ambas as versões do método proposto, FBDOCC e FBDOCC2.

Melhor caso O melhor caso ocorre quando para cada instância no laço mais externo (*i.e.,* a iteração sobre a variável i na Figura 4.1) no algoritmo, todos os $2l$ protótipos são reconhecidos como sendo da classe negativa quando comparados à primeira instância de treinamento. Nesse caso, a complexidade pode ser expressa como $(2l \cdot n)$, ou seja, apenas uma iteração do laço mais externo. Dessa forma, o melhor caso do FBDOCC é O$(l \cdot n)$ que representa também o melhor caso do FBDOCC2. Esse melhor caso gera um modelo inválido, pois nenhuma instância de treinamento é armazenada.

Na teoria, um custo computacional mais elevado do FBDOCC2 em relação ao FBDOCC é esperado, principalmente em problemas de alta dimensionalidade. Na prática, esse custo depende da dificuldade do problema (*i.e.*, o quão complexa é a distribuição de dados de treinamento). Na prática, existe uma baixa probabilidade de ocorrência do pior caso para ambas as versões do método, como mostrado posteriormente no Capítulo de experimentos.

Fase de Teste Na fase de teste, o procedimento mais importante consiste na ordenação do conjunto $NegSet$ em ordem ascendente de distâncias a uma instância z de teste. Seja $d_{NegSet} = dist(z, NN_{NegSet})$ a distância Euclidiana de z a seu vizinho mais próximo em $NegSet$. Seja $PosSet$ o conjunto de protótipos positivos associados com todas as instâncias em $NegSet$ localizadas a uma distância a z inferior a $d_{NegSet} + r$. Sendo $d_{PosSet} = dist(z, NN_{PosSet})$, z é classificado como novidade se $\frac{d_{NegSet}}{d_{PosSet}} > 1$. Nesse caso, a operação mais custosa da fase de teste é a ordenação que tem custo $O(n\ log\ n)$ no caso médio. Esse é o custo do algoritmo Quicksort (HOARE, 1962), utilizado nesse trabalho.

4.7 Método para Redução do Custo Computacional

O custo computacional do método pode ser reduzido evitando-se a realização de computações desnecessárias. Na fase 3 do método, um laço compara um protótipo gerado pela instância t_i com cada instância t_j no conjunto de treinamento T; tal que t_j é diferente de t_i. No entanto, se a distância entre t_i e t_j for maior que a soma $(r + th)$, certamente t_j não estará dentro de nenhuma das hiper esferas definidas pelos protótipos gerados a partir de t_i.

No intuito de previnir comparações entre protótipos e instâncias distantes, antes do início da fase de treinamento, é realizada uma ordenação ascendente de acordo com as distâncias em relação a um ponto localizado na origem do espaço de características. Uma exigência para o uso do método é que a base de dados esteja normalizada de forma a não conter nenhum valor menor que zero. A Figura 4.7 apresenta um exemplo onde os protótipos positivos gerados pela instância de treinamento representada por um círculo preto não precisam ser comparados às instâncias fora da região representada pelo arco em escala de cinza. Note que à medida que $(r + th)$ cresce, menor é a redução do custo computacional e a heurística tende ao mesmo custo do FBDOCC padrão.

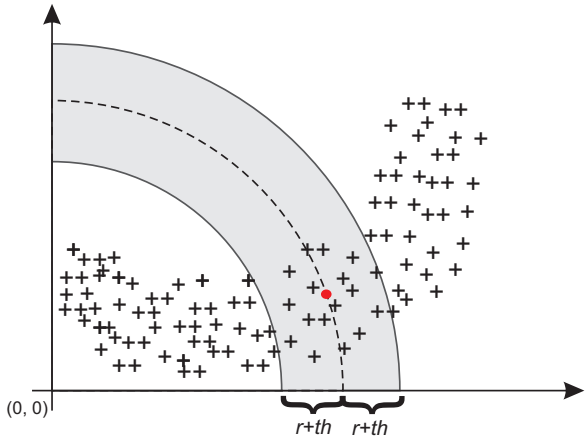

Figura 4.7: As instâncias de treinamento localizadas na região em escala de cinza serão as únicas comparadas aos protótipos gerados pela instância de treinamento representada pelo círculo vermelho.

4.8 Limitações do Método

Nesta subseção, algumas limitações conhecidas do método são listadas.

a) Teoricamente, se espera que o método tenha uma queda no desempenho da classificação quando aplicado a bases de dados com alta dimensionalidade e poucos exemplos de dados (ou seja, quando ocorre o problema da *Maldição da Dimensionalidade*). Nesses casos, o número de protótipos pode crescer de forma significativa

de acordo com o grau de dispersão dos valores dos atributos. Isso pode resultar em um *overfitting* dos dados;

b) Para alguns problemas, encontrar valores adequados para os parâmetros de maneira empírica pode não ser trivial. Porém, o uso de métodos de busca local (como o PSO) podem contornar o problema; e

c) Bases de dados volumosas podem levar a um custo computacional muito alto no treinamento. Se esse caso ocorrer associado a um problema de alta dimensionalidade, a modelagem pode se tornar um problema intratável por nosso método. O método apresentado na Seção 4.7 visa contornar esse problema.

4.9 Avaliação dos métodos propostos considerando a Seção 3.3

Princípio da robustez e do *trade-off* entre aceitação e rejeição de objetos:

Assim como nos outros classificadores experimentados nesse trabalho, esse princípio é tratado de forma natural pelos métodos propostos dado que eles visam a criação de descrições que englobam os dados da classe normal. Adicionalmente, o nível de ajuste da descrição em relação aos dados normais pode ser adaptado de forma a aceitar mais ou menos instâncias da classe novidade. Esse rigor na classificação pode ser manipulado através do uso de limiares de aceitação θ das classes normal e novidade.

Princípio da minimização de parâmetros:

Os métodos propostos, assim como boa parte dos métodos aqui descritos, possuem apenas dois parâmetros. O principal problema em relação aos parâmetros dos métodos propostos é seu alto grau de dependência em relação aos dados do problema. Diferente de outros métodos, como o SVDD e OCSVM, a influência dos parâmetros na classificação se torna muito mais evidente à medida que pequenas variações podem mudar de forma considerável o panorama do desempenho dos métodos. Esse aspecto, como discutido na Seção 4.5, pode ser contornado através do uso de técnicas de busca local.

Princípio da generalização:

O nível de generalização dos métodos propostos é assegurado, assim como para os demais métodos, através da metodologia de experimentos. O uso do conjunto de validação assegura que não ocorra um super ajuste do modelo aos dados de treinamento levando a um fraco desempenho na utilização do modelo em dados não vistos durante o treinamento. Logo, esse princípio é naturalmente considerado pelos métodos propostos e deve ser ajustado durante a criação do modelo.

Princípio da independência:

Dentre os problemas citados nesse princípio (atributos e classes disponíveis; desbalanceamento; baixo número de instâncias; e ruído), os métodos propostos, assim como os demais métodos, podem perder desempenho em casos onde os atributos e classes disponíveis e o baixo número de amostra não representem de forma satisfatória o problema. Por exemplo, no caso de poucas instâncias de treinamento, uma descrição muito grande pode ser gerada de forma a aceitar também objetos desconhecidos. Quanto ao desbalanceamento entre as classes, os métodos propostos não são afetados por esse problema dado que na fase de modelagem todas as classes conhecidas são agrupadas em apenas uma única classe. Em relação a ruídos, a fase de teste dos métodos propostos é realizada pelo algoritmo kNN, logo o parâmetro k, como discutido na Seção 4.4, desempenha um papel importante, não apenas na suavização da curva de decisão, mas também na robustez dos métodos à ruídos.

Princípio da adaptabilidade:

Esse aspecto não é tratado pelos métodos propostos dado que eles foram desenvolvidos para o modo operação *offline*.

Princípio da complexidade computacional:

Esse princípio foi tratado na Seção 4.6 e avaliado de forma criteriosa durante os experimentos. Os métodos propostos visam ser eficientes na fase de modelagem (treinamento) resultando em modelos com baixa complexidade na fase de teste.

4.10 Considerações Finais

Neste capítulo foram introduzidas duas versões do método FBDOCC (*Feature Boundaries Detector for One-Class Classification*). O método proposto tem como foco a resolução do problema da detecção de novidades na classificação com exemplos de uma única classe de forma a obter resultados similares aos métodos considerados no estado da arte em relação à precisão na classificação. Além disso, os métodos propostos visam o alcance de custos computacionais comparáveis ou melhores que estes métodos.

Para que os objetivos desejados fossem alcançados, um novo método para melhoria no tempo de treinamento sem influência no desempenho na classificação foi proposto.

5

Experimentos

Neste capítulo são reportados resultados experimentais obtidos pela aplicação de ambas as versões do método proposto, apresentadas no Capítulo anterior, em diferentes bases de dados obtidas a partir do repositório UCI (BACHE; LICHMAN, 2013), repositório StatLib (CAMPBELL; BENNETT, 2011) e geradas de forma artificial. Os métodos utilizados para comparação com os métodos propostos foram descritos ao longo do Capítulo 3. Os aspectos mais relevantes dos experimentos serão apresentados e analisados com base em métodos estatísticos.

Além dos trinta e três experimentos com bases consideradas *benchmark*, foi realizado um caso de estudo da aplicação de vários métodos para OCC ao problema real da detecção precoce de cardiopatias em crianças com sintomas de sopro cardíaco a partir de dados não invasivos; como peso, frequência cardíaca e idade.

5.1 Metodologia

A partir de cada base de dados original, foram geradas 25 diferentes variações com os exemplos dispostos de forma aleatória[1] e cada uma dessas variações foi particionada nos conjuntos de treinamento, validação e teste. O uso do conjunto de validação foi necessário para a busca pela melhor configuração de parâmetros dos classificadores; tarefa de otimização dos parâmetros. As proporções utilizadas nos conjuntos foram: treinamento - 50% de todos os exemplos normais; validação - 50% de todos os exemplos da classe positiva e 25% de todos os exemplos da classe normal; e teste - mesma proporção utilizada para validação. Para cada uma dessas variações, o desempenho do melhor modelo gerado (pelo método específico de busca de parâmetros descrito em 5.1.2) é armazenado para que possa ser comparado aos outros modelos de classificadores diferentes. Para cada medida de desempenho, o teste estatístico Wilcoxon *signed-rank*, com intervalo de confiança de 95%, foi realizado no intuito de consolidar os resultados. Além de ser encorajado por Demsar (DEMSAR, 2006) e ter sido também utilizado por Wu e Ye em (WU; YE, 2009), esse teste foi escolhio pelo fato de ser não-paramétrico.

[1] O número 25 foi escolhido para que seja possível a aplicação do teste estatístico com um maior grau de confiança

5.1.1 Bases de Dados

A Tabela 5.1 mostra o número de exemplos contidos nos conjuntos de treinamento, validação e teste assim como a quantidade de atributos para cada base de dados. As bases de dados com mais de duas classes estão com a classe escolhida como novidade indicada entre parenteses após seus respectivos nomes.

Tabela 5.1: Número de exemplos e atributos para cada base de dados.

Base de Dados	Conj. de Treinamento	Conj. de Validação	Conj. de Teste	Atributos
Banana	300	300	300	2
Distribicão Gaussiana	225	187	188	2
Banknote	381	495	496	4
Winsconsin Breast Cancer	222	230	231	9
Glass (1)	72	71	71	9
Glass (2)	98	57	59	9
Glass (3)	69	72	73	9
Glass (4)	102	55	57	9
Glass (5)	92	60	62	9
Ionosphere Bad	112	119	120	34
Ionosphere Good	63	143	145	34
Iris (1)	50	50	50	4
Iris (2)	50	50	50	4
Iris (3)	50	50	50	4
Mfeat-morph (0)	900	550	550	6
Mfeat-morph (1)	900	550	550	6
Mfeat-morph (2)	900	550	550	6
Mfeat-morph (3)	900	550	550	6
Mfeat-morph (4)	900	550	550	6
Mfeat-morph (5)	900	550	550	6
Mfeat-morph (6)	900	550	550	6
Mfeat-morph (7)	900	550	550	6
Mfeat-morph (8)	900	550	550	6
Mfeat-morph (9)	900	550	550	6
Seeds (1)	70	70	70	7
Seeds (2)	70	70	70	7
Seeds (3)	70	70	70	7
Soybean (17)	296	193	194	82
Wine (1)	59	58	61	13
Wine (2)	53	61	64	13
Wine (3)	65	56	57	13
Biomed	67	70	72	4
Diabetes	250	259	259	8

Banana - A base artificial Banana, como sugere o nome, é uma base bidimensional em formato de banana. Para os experimentos, foram geradas com o auxílio da biblioteca para Matlab DDTolls (TAX, 2014) 200 instâncias da classe normal e 100 da classe novidade. Tanto o conjunto de validação quanto o conjunto de teste possuem 100 instâncias igualmente divididas entre as classes normal e novidade. Essa base foi também usada em (TAX, 2001) (CABRAL; OLIVEIRA, 2008) (RÄTSCH, 1998) e (RÄTSCH; ONODA; MÜLLER, 1998).

A Figura 5.1 mostra o gráfico de distribuição dos conjuntos de treinamento e teste da base banana. Nesta Figura, pode-se verificar um baixo nível de sobreposição entre os dados das classes normal e novidade; o que torna esse problema relativamente simples.

Gaussiana - Essa base de dados foi também usada por Cao *et al.* (CAO; LEE; CHONG, 2003). A base consiste de 600 instâncias de dados bi-dimensionais geradas a partir de duas distribuições

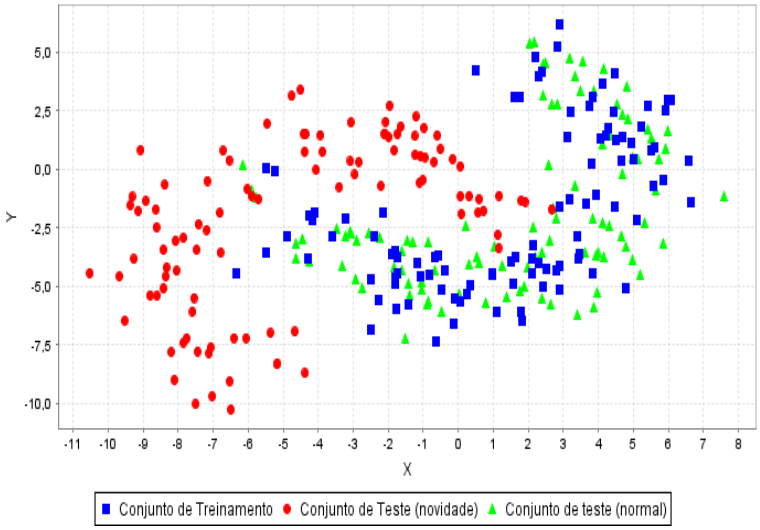

Figura 5.1: Distribuição de dados em formato de banana.

gaussianas. O conjunto de treinamento contém 225 amostras representando a classe normal enquanto que, tanto o conjunto de validação quanto o conjunto de teste, possuem por volta de 188 instâncias igualmente divididas entre as classes normal e novidade. As instâncias da classe normal foram geradas por uma distribuição gaussiana com vetor de médias $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$ e matriz de covariância $\begin{pmatrix} 4 & 0 \\ 0 & 4 \end{pmatrix}$; as instâncias pertencentes à classe novidade foram geradas por uma distribuição gaussiana com vetor de médias $\begin{pmatrix} 4 \\ 4 \end{pmatrix}$ e matriz de covariância $\begin{pmatrix} 4 & 0 \\ 0 & 4 \end{pmatrix}$. A Figura 5.2 mostra a distribuição dos dados da base de dados da distribuição gaussiana.

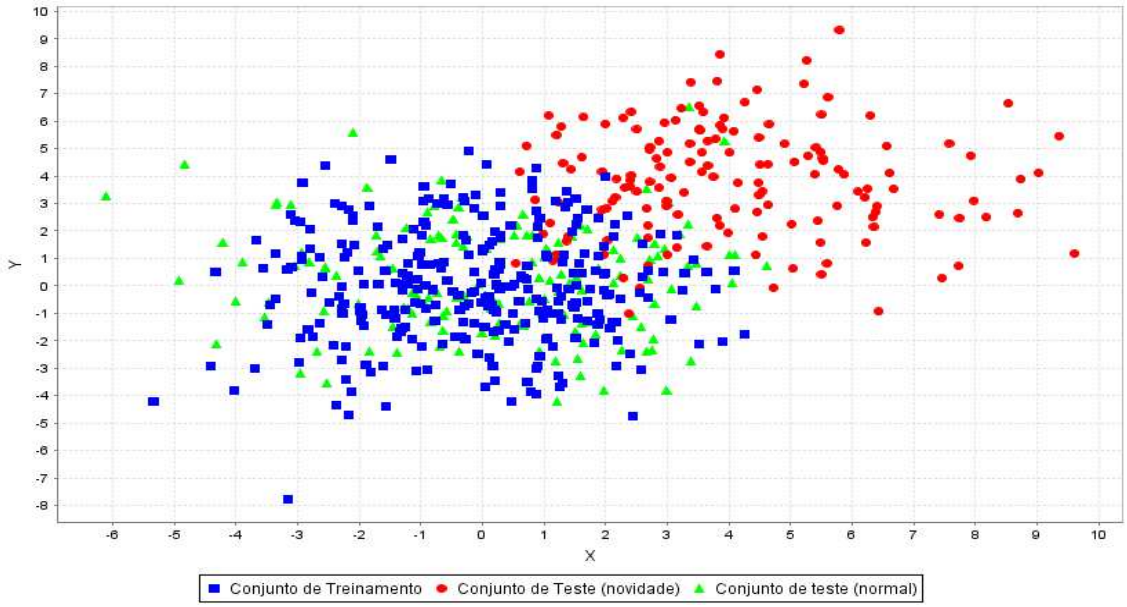

Figura 5.2: Distribuições Gaussianas.

Banknote - Essa base de dados é formada por dados extraídos de imagens de notas promissórias genuínas e forjadas. Para digitalização, uma câmera, geralmente utilizada para inspeção de impressões, foi empregada. As imagens têm 400 x 400 pixels. Para essa base, foi utilizada a transformada de Fourier para a extração de características.

Os seguintes atributos foram utilizados para a montagem da base de dados: variância da transformada wavelet da imagem; assimetria da transformada wavelet da imagem; curtose da transformada wavelet da imagem; e entropia da imagem.

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Wisconsin Breast Cancer - O conjunto de dados Breast Cancer consiste de 683 amostras. Cada amostra é composta por nove atributos de entrada mais o atributo de saída, classe benigna ou maligna. A classe benigna foi selecionada para representar a classe normal e dessa forma o conjunto de treinamento contém 222 amostras da classe normal. Essa base de dados também foi usada em vários outros trabalhos de mineração de dados como (BROWN, 2004) (BENNETT; DEMIRIZ; MACLIN, 2002) e (ANTOS et al., 2001).

A seguinte lista descreve os atributos da base de dados Wisconsin Breast Cancer: espessura do tumor - real de 1 a 10; uniformidade do tamanho da célula - real de 1 a 10; uniformidade do formato da célula - real de 1 a 10; adesão marginal - real de 1 a 10; tamanho de uma célula epitelial - real de 1 a 10; núcleos expostos - real de 1 a 10; cromatina branda - real de 1 a 10; normalidade do nucléolo - real de 1 a 10; mitose - real de 1 a 10; e classe - 2 para benigno e 4 para maligno.

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Glass - A criação dessa base de dados é motivada com base em investigações criminais. Na cena do crime, o tipo de vidro deixado pode ser usado como evidência caso seja corretamente identificado.

As seguintes características foram utilizadas: índice refrativo; sódio; magnésio; alumínio; silicone; potássio; cálcio; bário; e ferro. Exceto o índice refrativo, todas as outras outras características são medidas em relação ao peso percentual do óxido correspondente à cada substância. Essa base de dados possui 7 classes, dessas, 5 foram utilizadas nos experimentos; dado que as outras duas possuiam um número de exemplos insuficiente para modelagem da classe.

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Ionosphere - A base de dados Ionosphere é formada por dados de 16 antenas de alta frequência instaladas na Baía Goose no Canadá. O alvo dessas antenas foram elétrons livres na ionosfera. Exemplos do tipo “Good” são aqueles mostrando evidência de estruturas na ionosfera. Exemplos do tipo “Bad” são aqueles que não apresentam evidências de estruturas; os sinais enviados pelas antenas passam através da ionosfera.

Cada exemplo dessa base de dados é formado por 34 atributos retornados por uma função que tem como argumentos sinais eletromagnéticos complexos.

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Iris - A base de dados Iris foi introduzida por R.A. Fisher (FISHER, 1936) e consiste na classificação de exemplos em três tipos de flores Iris, da família das Iridáceas: Iris Setosa, Iris Versicolour e Iris Virgínica. Cada amostra é representada por quatro diferente atributos: comprimento da sépala; largura da sépala; comprimento da pétala; largura da pétala; e classe (virgínica, versicolor e setosa).

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Mfeat-morph - A base de dados Mfeat-morph (*Multiple Features Data Set*) consiste de características morfológicas de dígitos manuscritos (0 a 9) extraídos de mapas holandeses. 200 exemplos por classe (em um total de 2000 exemplos) foram digitalizados para imagens binárias. Essa base faz parte de um conjunto de 6 bases de dados com dados de diferentes naturezas extraídos de mapas holandeses. Considerando todas as bases disponíveis, cada dígito é representado em termos dos seguintes seis conjuntos de características (seguidos pela quantidade de atributos): coeficientes de Fourier dos formatos dos dígitos (76); correlalação de perfis (216); coeficienes de Karhunen-Love (64); média dos pixels em janelas de 2 x 3 (240); momentos de Zernike (47); e características morfológicas (6). Dessa forma, apenas a última base de dados foi utilizada em experimentos nessa tese. Essa mesma base de dados foi utilizada em (DéSIR et al., 2013).

Para essa base de dados, a cada experimento, um dígito foi utilizado como classe novidade; gerando assim, 10 experimentos distintos para a mesma base de dados.

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Seeds - Essa base de dados consiste de características de sementes pertencentes a três diferentes tipos de trigo: kama, rosa e canadense, 70 exemplos de cada classe. Imagens de alta qualidade da estrutura interna foram obtidas utilizando-se uma técnica de raio-x. Além de ser uma técnica não destrutiva, essa técnica é considerada mais barata que outras mais sofisticadas como tecnologias laser e varredura mcroscópica.

Com base nas imagens coletadas, sete parâmetros geométricos foram medidos: área A; perímetro P; compacidade $C = \frac{4 \cdot \pi \cdot A}{P^2}$; comprimento da semente; largura da semente; coeficiente de assimetria; e comprimento do sulco da semente;

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Soybean - A base de dados Soybean é uma base *benchmark* amplamente usada na mineração de dados. Essa base contém 19 classes, dessa forma, assim como em outros trabalhos (TIAN; GU, 2010) (OLIVEIRA; COSTA; F., 2008), a classe 17 foi escolhida para representar a classe novidade. Após a escolha da classe novidade, todas as amostras dessa classe são retiradas do conjunto de treinamento. Na base de dados soybean, cada amostra é formada por 35 atributos, numéricos e nominais, que quando convertidos para atributos apenas numéricos resultam em 82

atributos.

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Wine - A base de dados Wine é resultado de uma análise química de vinhos produzidos na mesma região na Itália a partir de três diferentes cultivos de uvas. A análise determinou a quantidade de ocorrência de 13 substâncias em cada um dos 3 tipos de vinhos. No contexto da classificação, esse é um problema com classes bem estruturadas no espaço de características. Cada amostra é representada por treze diferente atributos: teor alcoólico; ácido málico; ash (resíduos inorgânicos); alcalinidade do ash; magnésio; fenóis totais; flavonóides; fenóis não-flavonóides; proantocianidinas; intensidade da cor; tonalidade; OD280/OD315 de vinhos diluídos; e prolina.

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

Biomed - A base Biomed foi criada em um estudo para o desenvolvimento de métodos para a identificação de portadores de uma rara desordem genética. Quatro medições, m1, m2, m3 e m4 foram feitas em amostras de sangue. Pela natureza rara da doença, existiam apenas poucos portadores da doença para a disponibilização de dados da classe doença, novidade. A base de dados contém 134 amostras da classe normal (voluntários não portadores da doença) e 75 amostras da classe novidade (voluntários portadores da doença). O conjunto de treinamento foi formado por 67 amostras da classe normal sendo o conjunto de validação formado por 70 amostras e o conjunto de teste formado por 72 amostras.

Essa base de dados está disponível em (CAMPBELL; BENNETT, 2011).

Diabetes - Esta base de dados possui como título original *Pima Indians Diabetes Database* e é de propriedade do *National Institute of Diabetes, Digestive and Kidney Diseases*. Cada amostra consiste em medições realizadas em pacientes do sexo feminino com mais de 21 anos, residentes em Phoenix, Arizona, EUA a fim de se detectar a ocorrência ou não de diabetes. Das 768 amostras disponíveis na base, foram separadas 250 amostras da classe normal (pacientes sem diabetes) para o treinamento.

Cada registro contem 8 atributos de entradas mais a classe. Os atributos representam as seguintes informações: número de vezes que ficou grávida; concentração de glicose no Plasma em teste de tolerância de glicose oral; pressão sanguínea diastólica (mm Hg); dobras na pele do tríceps (mm); aplicações de 2 horas de insulina de soro (mu U/ml); índice de massa corpórea $\left(\frac{peso(kg)}{altura(m^2)}\right)$; função de genealogia de diabete; idade (anos); e classe: 0 - Saudável, 1 - Diabetes.

Essa base de dados está disponível em (BACHE; LICHMAN, 2013).

5.1.2 Atribuição de Parâmetros

No intuito de proporcionar uma maior justiça na seleção de parâmetros entre as técnicas experimentadas, o algoritmo de otimização PSO foi empregado na busca da melhor configura-

ção de parâmetros para: as duas versões do FBDOCC; o SVDD; o OCSVM; o LSOCSVM; o KPCA; e o CNNDD. Para esses métodos, para cada uma das 25 variações de cada base de dados, os parâmetros foram otimizados por uma população de 25 partículas em um máximo de 50 iterações. Como critério de parada, caso a otimização não melhore o valor do *fitness* por dez gerações consecutivas a otimização é finalizada e a melhor partícula é retornada. Para os métodos SVDD, OCSVM, LSOCSVM, KPCA e CNNDD, a função de *fitness* utilizada foi o valor da AUC para o conjunto de validação. A maioria dos trabalhos utilizando o SVDD, incluindo os trabalhos realizados pelo criador do método (TAX, 2001), convergem para o uso da função de kernel gaussiana (LUO; WANG; CUI, 2011)(HANSEN; SJöSTRAND; LARSEN, 2010)(WANG et al., 2013), dessa forma, esse trabalho optou pelo uso dessa função de kernel. Em relação aos métodos GPOCC e OCRF, a seleção de parâmetros não se mostrou uma tarefa decisiva, dessa forma, foram utilizados os parâmetro descritos nos trabalhos conduzidos por seus respectivos autores (BODESHEIM et al., 2012) (DéSIR et al., 2013).

Os parâmetros utilizados por ambas as versões do método FBDOCC foram otimizados com base na função de *fitness* apresentada na Equação (4.2). Os seguintes valores foram testados para o parâmetro c: 0,02; 0,05; 0,1; 0,2; 0,3; 0,4; e 0,5. Teoricamente, o aumento do parâmetro c resulta em modelos com uma maior taxa de redução de protótipos. Dentre todos os valores experimentados para o parâmetro c, o valor que maximiza a redução de protótipos com um nível de generalização satisfatório foi utilizado para comparação com os outros métodos. A Figura 5.3 apresenta o comportamento do método FBDOCC em relação à variação do parâmetro c na função de *fitness*. Pela Figura, percebe-se que o percentual de armazenamento diminui de forma mais acentuada até o valor 0,2, desse valor em diante a taxa de redução de protótipos passa a diminuir de forma mais sutil. Em relação à AUC, o aumento do valor de c resulta em um leve declínio no desempenho. De acordo com os valores utilizados como parâmetro c (0,02; 0,05; 0,1; 0,2; 0,3; 0,4; e 0,5), os resultados apresentados nas curvas da Figura 5.3, são respectivamente, para a AUC e a taxa de armazenamento: 0,903-0,534; 0,884-0,483; 0,879-0,443; 0,874-0,337; 0,860-0,329; 0,862-0,287; e 0,861-0,279. Dadas as boas taxa de armazenamento e AUC, para os próximos experimentos, o valor de c foi fixado em 0,2.

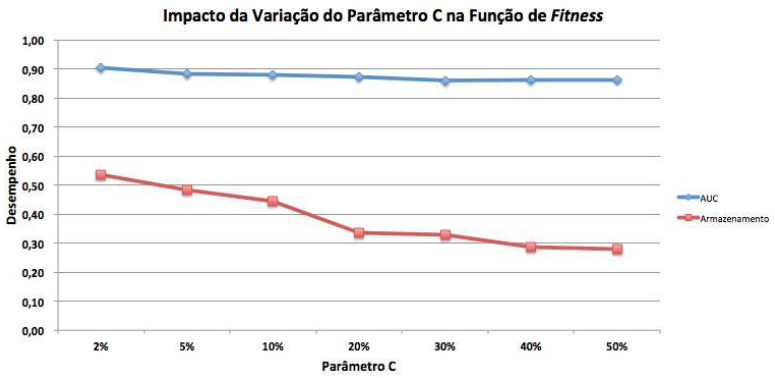

Figura 5.3: Evolução das taxa de armazenamento e AUC com o aumento do valor para o parâmetro c na função de *fitness*

Em relação aos parâmetros específicos do PSO, apresentados no Apêndice A.1, experimentos preliminares mostraram que a variação de valores não influenciou de forma significativa o desempenho da otimização. Dessa forma, esses parâmetros foram fixados em: $w = 0,8$; $\varphi_1 = 0,9$; $\varphi_1 = 0,5$; $\varphi_3 = 0,9$; e tamanho da vizinhança = 4.

Assim como o PSO, a técnica *Simulated Annealing* (KIRKPATRICK; GELATT; VECCHI, 1983) também foi investigada na busca por parâmetros ótimos do método proposto. Dado que o valor 0,2 foi adotado após a investigação utilizando-se o PSO, esse mesmo valor foi experimentado com o *Simulated Annealing*. Diferente do PSO, não foram obtidas taxas satisfatórias de armazenamento e AUC; que foram 0,798 e 0,822, respectivamente. Dessa forma, optou-se por utilizar apenas o PSO como ferramenta de otimização.

5.2 Métricas para Análise de Desempenho

No intuito de comparar os desempenhos dos métodos abordados nesse trabalho, curvas ROC (*Receiver Operating Characteristic*) (FAWCETT, 2006) e suas áreas sob as curvas (AUC - *Area Under the Curve*) foram utilizadas como uma das métricas de desempenho.

Curvas ROC são técnicas para visualização e seleção de classificadores baseadas em seus desempenhos. Um exemplo de aplicação dessa técnica ocorre na teoria da detecção de sinais para mostrar o *trade-off* (compromisso) entre acertos na classe normal e erros na mesma classe (SWETS, 1988) (EGAN, 1975).

Como exemplo, em um teste de diagnóstico médico, existem dois tipos de erro que podem ocorrer na decisão: a escolha de um falso normal (no sentido de declarar uma pessoa enferma como sã) ou a escolha de uma falsa novidade (declarar uma pessoa sã como doente). Para um profissional que tem perante si um dado diagnóstico para uma doença, ao ter que decidir, ele irá preferir uma falsa novidade a um falso normal - principalmente se a doença for contagiosa - pois este tipo de erro conduzirá a "um mal menor" em termos de diagnóstico.

A Figura 5.4 mostra um exemplo de curva ROC onde o classificador consegue distinguir perfeitamente os objetos entre normal e novidade. Na prática esse caso é, na maioria das vezes, irreal, pois problemas do mundo real, geralmente, não são linearmente separáveis (*i.e.*, a sobreposição entre as classes pode impossibilitar a criação de um classificador com 100% de acerto). Na Figura 5.4, *PD* significa probabilidade de detecção e *PFA* significa probabilidade de falso alarme.

Para a obtenção dos pontos da curva ROC, quatro possíveis situações na classificação devem ser consideradas:

a) Verdadeiro Positivo (VP) - quando uma amostra da classe positiva é corretamente classificada como novidade;

b) Verdadeiro Negativo (VN) - quando uma amostra da classe normal é corretamente classificada como normal;

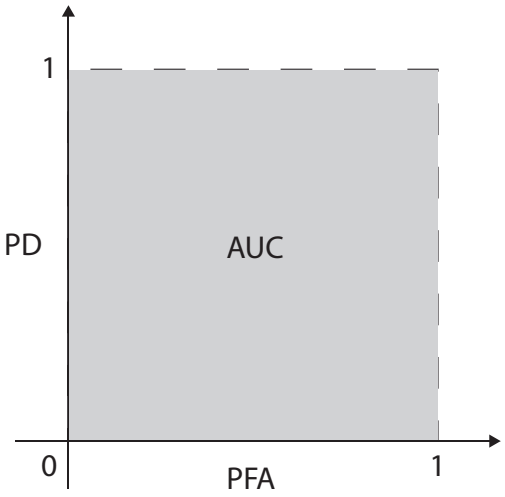

Figura 5.4: Curva ROC perfeita, $AUC = 1$.

c) Falso Positivo (FP) - quando uma amostra da classe normal é incorretamente classificada como novidade; e

d) Falso Negativo (FN) - quando uma amostra da classe novidade é incorretamente classificada como normal.

Levando em conta as quatro possíveis situações acima a Equação (5.1) fornece os valores para cálculo dos pontos da ROC. Em (5.1), P representa o número de amostras da classe positiva enquanto que N representa o número de amostras da classe negativa.

$$PD = \frac{VP}{P} = \frac{VP}{VP+FN}, \qquad PFA = \frac{FP}{N} = \frac{FP}{VN+FP}. \tag{5.1}$$

Com esses valores, pode-se ainda definir a probabilidade de uma instância classificada como positiva ser realmente desta classe (*precision*), e a probabilidade de se classificar corretamente as instâncias considerando as duas classes (*accuracy*). A métrica *precision* trata apenas a capacidade de detecção dos padrões da classe positiva. *Accuracy*, por sua vez, avalia de modo geral a capacidade de generalização de um classificador. A Equação (5.2) ilustra o cálculo dessas duas métricas.

$$precision = \frac{VP}{VP+FP}, \qquad accuracy = \frac{VP+VN}{P+N}. \tag{5.2}$$

A Figura 5.5 consiste em três curvas ROC geradas de maneira hipotética. Segundo essa figura, três diferentes níveis de discriminação de um problema foram alcançados. Classificadores com melhor desempenho produzem curvas mais próximas à curva ideal da Figura 5.4. A diagonal mostra o caso em que o poder de discriminação se iguala ao acaso, ou seja, a discriminação (levando em conta os atributos fornecidos para a resolução do problema) é aleatória.

Classificadores para classificação com exemplos de uma única classe compartilham o modo de operação e podem ser descritos, em alto nível, como uma função $F(x)$, tal que se

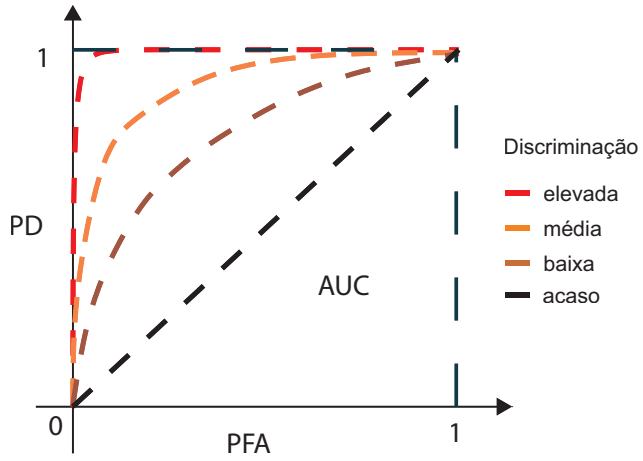

Figura 5.5: Curvas ROC representando três diferentes capacidades de discriminação.

$F(x) < \theta$ o objeto x é dito ser uma novidade, senão o objeto é classificado como normal. De acordo com essa representação, pode-se dizer que classificadores dessa natureza funcionam baseados em limiares de operação, θ. Esse trabalho se refere a esses limiares como pontos de operação. Logo, um classificador pode ser representado como um modelo $F(x)$, gerado pelo treinamento de determinado algoritmo, associado a um ponto de operação. Diferentes pontos de operação resultam em diferentes desempenhos na classificação. Alterando-se um ponto de operação, o classificador pode aumentar sua tolerância a novidades ou ter uma maior taxa de detecção de novidades em detrimento ao erro na classe normal.

A Figura 5.6 ilustra um exemplo de classificador onde um determinado ponto de operação foi escolhido de forma a melhorar a taxa de classificação de instâncias normais em detrimento à detecção de instâncias novidades. O eixo das ordenadas representa o total de instâncias da base de teste. Nessa Figura, à medida que se aumenta o ponto de operação, um maior percentual de instâncias é classificado como novidade (linha tracejada), independente da classe cujo essas instâncias pertencem. Analogamente, à medida que o valor do ponto de operação é diminuído, um maior número de instâncias é aceito como normal (linha contínua).

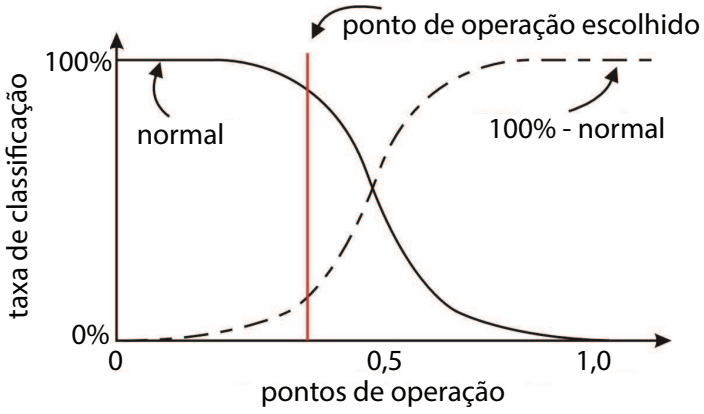

Figura 5.6: Evolução da taxa de classificação de objetos como normais e novidades em relação ao deslocamento do ponto de operação escolhido.

A área sob a curva é uma das métricas mais adotadas para a avaliação do poder de generalização de métodos para OCC. Porém, em caso de dados desbalanceados, o classificador pode produzir um alto valor para a AUC por classificar erroneamente uma grande quantidade de instâncias da classe minoritária. Nesses casos, a métrica MCC (Matthews *Correlation Coefficient*) (BALDI et al., 2000) detecta a ocorrência de erro em uma das classes e diminui o desempenho do modelo. O MCC é obtido escolhendo-se um ponto de operação da curva ROC e avaliando-se a taxa de erro do modelo. Neste trabalho, para todos os experimentos, o ponto de operação escolhido na curva ROC equivale ao ponto mais próximo à coordenada (0,1) no gráfico (*i.e.*, o ponto de operação que resulta na menor taxa de erro). A Equação (5.3) apresenta a fórmula para o cálculo do MCC.

$$MCC = \frac{(VP \times VN) - (FP \times FN)}{\sqrt{(VP+FN)(VP+FP)(VN+FP)(VN+FN)}} \tag{5.3}$$

As outras métricas utilizadas para avaliação nesse trabalho foram o tempo de treinamento em milisegundos e a taxa percentual de redução de protótipos.

5.3 Experimentos Preliminares (Análise Visual)

Algumas bases de dados artificiais bidimensionais foram geradas com a finalidade de avaliar a superfície de decisão do método em distribuições com formatos complexos.

A Figura 5.7 ilustra a região considerada normal pelo método para uma base de dados em forma de espiral. Essa base de dados também foi explorada por Hoffmann (HOFFMANN, 2007) e é definida como o conjunto $\{(x,y)|x = (0,07\varphi + a)cos(\varphi), y = (0,07\varphi + a)sin(\varphi), a \in [0..0,1], \varphi \in [0..6\pi]\}$. Para este conjunto, foram gerados 3000 pontos distribuídos de forma aleatória com igual probabilidade. Essa base artificial é importante devido às suas regiões côncavas e convexas.

A Figura 5.8 apresenta a distribuição de dados RingLineSquare que consiste de um anel ligado por uma linha a um quadrado (base também utilizada em (HOFFMANN, 2007)). Essa base de dados foi gerada pela combinação de um anel de diâmetro interno de 1 centímetro e diâmetro externo de 2 centímetros, um quadrado de com lado 2 centímetros vazado por outro quadrado de lado 1 centímetro e uma reta de comprimento 1,6 e largura 0,2 conectando as duas partes. Esse experimento mostra o desempenho do método proposto aplicado a distribuições não paramétricas. Nesse caso, os interiores nas regiões do círculo e do quadrado criam regiões côncavas de descrição não trivial.

Na Figura 5.9, é mostrada a forma como o método processa bases de dados com múltiplas classes. Nessa Figura uma base de dados formada pela mistura de distribuições contendo três classes diferentes (em escalas de cinza) sendo uma das classes uma distribuição descontínua. As três classes representam eventos normais e foram fundidas para compor o conjunto de exemplos da classe normal. Esse exemplo ilustra o poder do método em representar grupos de

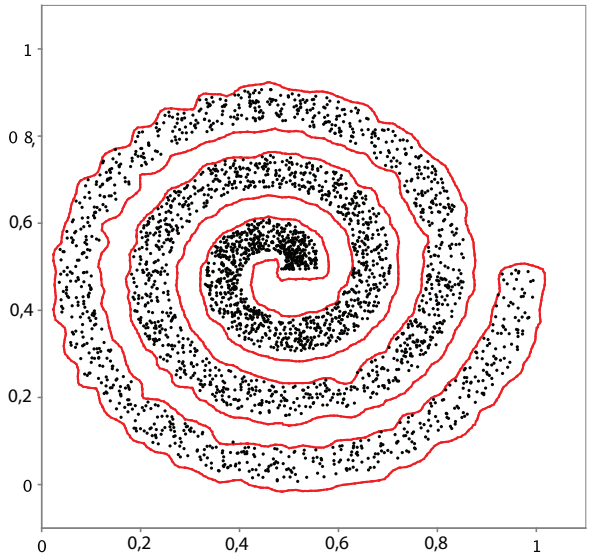

Figura 5.7: Modelo gerado, em vermelho, pelo FBDOCC2 para a base de dados Espiral.

dados descontínuos com a criação de mais de uma descrição para a classe normal.

As Figuras 5.7, 5.8 e 5.9 são exemplos de problemas com espaços de características de baixa dimensionalidade. Esses são casos onde a suposição de modelagem dos dados normais por uma descrição fechada funciona sem grande dificuldade. Para problemas de alta dimensionalidade, construir uma descrição fechada pode ser uma tarefa desafiadora dependendo de quão esparsa seja a distribuição dos exemplos de treinamento no espaço de características e da quantidade de exemplos de treinamento. Esse problema é conhecido como *A Maldição da Dimensionalidade*. Nesse cenário, o método proposto não necessariamente cria uma região totalmente fechada. Os experimentos das próximas seções incluem bases de dado com múltiplas classes e em espaços de alta dimensionalidade.

5.4 Análise da Precisão dos Resultados

A Tabela 5.2 contém as AUCs de todos os experimentos envolvendo as bases de dados descritas na Seção 5.1.1. Cada célula contém a média das 25 AUCs geradas para cada base e respectivo classificador seguida por seu desvio padrão. Nesse caso, um desvio padrão mais alto pode indicar que o método tem um alto grau de adaptação ao conjunto de treinamento (*i.e.*, pode haver ocorrido uma super adaptação do modelo à base de dados em questão), o que não é desejável. As melhores médias, considerando a significância do teste de Wilcoxon *Signed-rank*, para cada base de dados estão destacadas em negrito.

De acordo com a Tabela 5.2, o FBDOCC obteve a melhor média geral das simulações

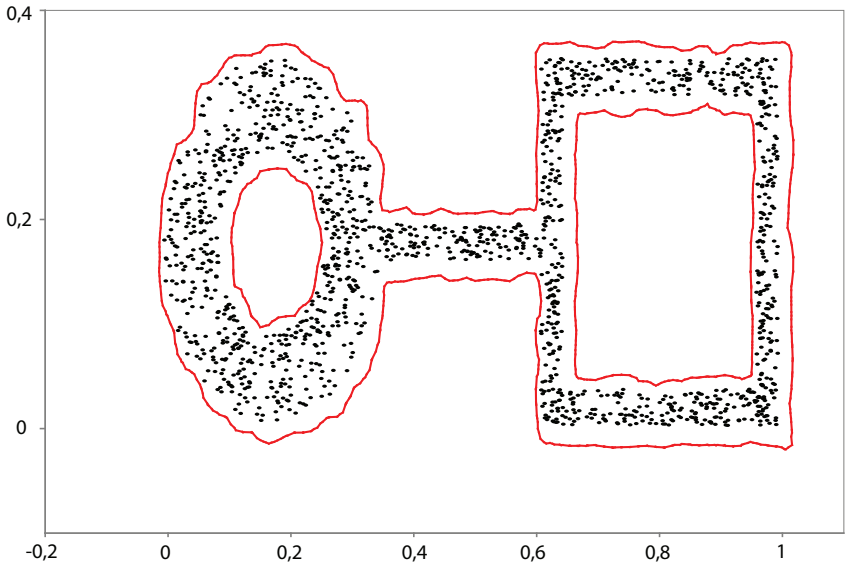

Figura 5.8: Modelo gerado, em vermelho, pelo FBDOCC2 para a base de dados Ring-Line-Square.

assim como os melhores resultados em vinte e um dos trinta e três experimentos. Os resultados apresentados pelo LSOCSVM foram considerados os melhores em dezesseis das ocasiões. O melhor desempenho geral foi alcançado pelo FBDOCC, 0,874. Vale salientar que o método OCRF não está presente na Tabela 5.2 dado que ele retorna um comitê de árvores de decisões binárias; o que impossibilita a computação do *score* necessário para o cálculo da AUC de cada modelo. Dessa forma, a precisão do OCRF será avaliada apenas em termos de seu MCC (Tabela 5.3). A Tabela 5.2 também apresenta uma alta similaridade entre os resultados obtidos pelo SVDD e pelo OCSVM, corroborando com a afirmação feita por Hoffmann (HOFFMANN, 2007) de que quando utilizado um kernel não linear (como é a função de base radial - RBF) esse métodos são equivalentes. Segundo a Tabela 5.2, observa-se também que o FBDOCC2 obteve bons resultados em experimentos onde o FBDOCC não alcançou um bom desempenho, como as bases de dados banana e banknote.

Os experimentos realizados envolvem bases com alta dimensionalidade, como a Soybean. Nesse caso, num espaço com 82 dimensões, a criação de uma região totalmente fechada, que englobe completamente os dados da classe normal, parece ser inviável. Esse aspecto afetou o desempenho apenas da versão FBDOCC2 do método proposto.

A Figura 5.10 ilustra uma comparação par a par entre o método considerado de melhor desempenho nos experimentos, FBDOCC, e os métodos restantes. Para cada gráfico, o eixo vertical concentra o valor das AUCs do FBDOCC e o eixo horizontal representa as AUCs do método a ser comparado. Dessa forma, todo ponto localizado acima da linha que liga as coordenadas (0,0) e (1,1) é interpretado como um experimento onde o FBDOCC teve um melhor desempenho que o método comparado. Analisando visualmente a Figura 5.10, nota-se que

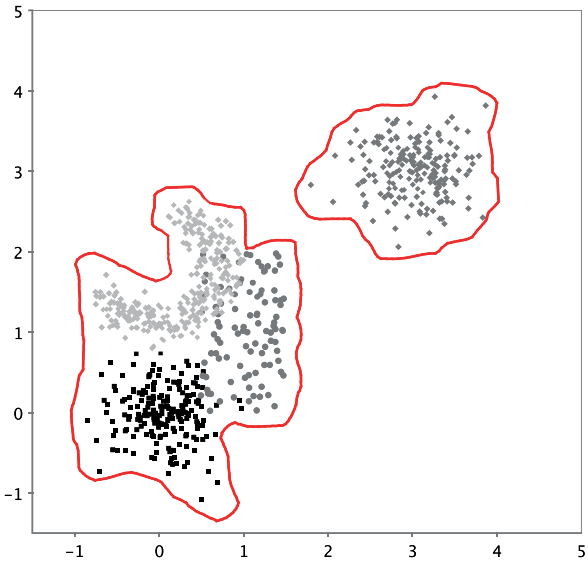

Figura 5.9: Superfície de decisão, em vermelho, para uma base de dados com múltiplas classes e descontinua.

Tabela 5.2: Áreas sob a curva (AUCs) médias para todos os experimentos

Dataset	FBDOCC	FBDOCC2	SVDD	OCSVM	LSOCSVM	ker. PCA	GP-OCC	CNNDD
Banana	0,926 (0,236)	**0,985** (0,099)	0,984 (0,082)	**0,990** (0,065)	**0,989** (0,054)	**0,989** (0,055)	**0,990** (0,051)	0,885 (0,271)
Gaussian Distributions	**0,966** (0,147)	0,952 (0,134)	0,856 (0,301)	0,916 (0,148)	0,925 (0,131)	0,915 (0,139)	0,912 (0,143)	0,932 (0,190)
Banknote	0,968 (0,165)	**0,994** (0,157)	**0,998** (0,036)	0,997 (0,070)	0,946 (0,170)	0,997 (0,038)	0,998 (0,037)	0,812 (0,214)
Winsconsin Breast Cancer	0,971 (0,123)	**0,995** (0,052)	0,991 (0,074)	**0,992** (0,058)	**0,994** (0,049)	0,983 (0,082)	0,981 (0,077)	**0,994** (0,052)
Glass (1)	**0,749** (0,253)	0,540 (0,306)	0,620 (0,302)	0,558 (0,326)	0,451 (0,307)	0,325 (0,335)	0,361 (0,241)	0,404 (0,320)
Glass (2)	**0,706** (0,312)	**0,609** (0,346)	0,537 (0,258)	0,489 (0,321)	**0,656** (0,504)	0,329 (0,240)	0,316 (0,243)	0,448 (0,295)
Glass (3)	**0,699** (0,265)	0,583 (0,294)	0,534 (0,276)	0,544 (0,272)	0,510 (0,289)	0,454 (0,232)	0,461 (0,243)	0,538 (0,318)
Glass (4)	**0,959** (0,250)	**0,931** (0,288)	0,438 (0,379)	0,699 (0,365)	**0,890** (0,361)	0,700 (0,457)	0,847 (0,215)	**0,913** (0,237)
Glass (5)	**0,952** (0,220)	**0,942** (0,210)	0,783 (0,464)	0,879 (0,212)	**0,956** (0,196)	0,802 (0,490)	0,932 (0,191)	0,910 (0,212)
Ionosphere Bad	0,954 (0,144)	0,934 (0,437)	0,920 (0,373)	0,960 (0,135)	**0,985** (0,087)	0,958 (0,159)	0,868 (0,169)	0,916 (0,209)
Ionosphere Good	**0,661** (0,268)	**0,645** (0,275)	0,452 (0,264)	0,450 (0,226)	0,483 (0,304)	0,338 (0,272)	0,509 (0,150)	0,390 (0,286)
Iris (1)	**1,000** (0,018)	**1,000** (0,000)	**1,000** (0,000)	**1,000** (0,000)	**1,000** (0,000)	**0,960** (0,443)	**1,000** (0,000)	**0,999** (0,053)
Iris (2)	0,906 (0,272)	0,957 (0,161)	0,961 (0,171)	0,958 (0,146)	**0,978** (0,154)	0,776 (0,569)	0,953 (0,160)	0,863 (0,381)
Iris (3)	**0,989** (0,138)	**0,989** (0,087)	0,973 (0,126)	0,965 (0,172)	0,988 (0,089)	0,935 (0,440)	0,969 (0,143)	0,953 (0,267)
Mfeat-morph (1)	0,984 (0,075)	**0,986** (0,069)	0,897 (0,353)	0,938 (0,127)	0,953 (0,144)	0,816 (0,180)	0,889 (0,136)	0,815 (0,406)
Mfeat-morph (2)	**0,781** (0,260)	**0,779** (0,304)	0,568 (0,336)	0,404 (0,354)	0,640 (0,305)	0,396 (0,168)	0,342 (0,183)	0,530 (0,362)
Mfeat-morph (3)	0,786 (0,340)	0,624 (0,386)	0,882 (0,124)	0,846 (0,197)	**0,895** (0,126)	0,860 (0,119)	0,860 (0,115)	0,844 (0,165)
Mfeat-morph (4)	0,725 (0,371)	**0,719** (0,331)	0,805 (0,140)	0,795 (0,144)	**0,840** (0,133)	0,810 (0,118)	0,807 (0,122)	0,803 (0,138)
Mfeat-morph (5)	0,782 (0,323)	0,850 (0,203)	0,875 (0,182)	0,799 (0,210)	**0,913** (0,100)	0,792 (0,174)	0,806 (0,145)	0,722 (0,316)
Mfeat-morph (6)	0,918 (0,214)	0,853 (0,361)	0,811 (0,324)	0,939 (0,143)	**0,953** (0,096)	**0,953** (0,087)	0,950 (0,094)	0,931 (0,292)
Mfeat-morph (7)	**0,821** (0,271)	**0,737** (0,295)	0,646 (0,362)	0,513 (0,313)	**0,722** (0,393)	0,323 (0,190)	0,326 (0,155)	0,516 (0,311)
Mfeat-morph (8)	0,764 (0,391)	0,700 (0,391)	**0,887** (0,226)	0,780 (0,263)	**0,888** (0,226)	0,705 (0,286)	0,802 (0,173)	0,616 (0,362)
Mfeat-morph (9)	**0,915** (0,183)	**0,912** (0,171)	0,642 (0,520)	0,808 (0,216)	0,826 (0,150)	0,684 (0,159)	0,459 (0,225)	0,761 (0,379)
Mfeat-morph (10)	**0,771** (0,308)	**0,759** (0,350)	0,624 (0,319)	0,464 (0,295)	**0,787** (0,351)	0,330 (0,185)	0,327 (0,159)	0,478 (0,400)
Seeds (1)	**0,940** (0,206)	**0,904** (0,208)	0,890 (0,200)	0,889 (0,194)	0,895 (0,198)	0,567 (0,603)	0,913 (0,186)	0,752 (0,390)
Seeds (2)	**0,992** (0,126)	0,992 (0,084)	0,972 (0,142)	0,978 (0,105)	0,976 (0,112)	0,548 (0,697)	0,982 (0,103)	0,979 (0,148)
Seeds (3)	**0,976** (0,205)	**0,984** (0,108)	0,939 (0,179)	0,963 (0,144)	0,960 (0,157)	0,644 (0,665)	0,957 (0,146)	**0,952** (0,235)
Soybean (17)	**0,874** (0,214)	0,642 (0,314)	0,635 (0,294)	0,645 (0,195)	**0,923** (0,149)	0,626 (0,197)	0,665 (0,171)	0,267 (0,370)
Wine (1)	**0,985** (0,095)	0,970 (0,129)	0,957 (0,169)	0,976 (0,112)	0,977 (0,110)	0,977 (0,106)	0,974 (0,118)	**0,978** (0,123)
Wine (2)	**0,901** (0,196)	**0,869** (0,187)	0,721 (0,354)	0,763 (0,247)	0,749 (0,248)	0,675 (0,263)	0,649 (0,243)	0,746 (0,365)
Wine (3)	**0,862** (0,272)	**0,762** (0,307)	0,524 (0,292)	0,464 (0,343)	0,627 (0,334)	0,353 (0,309)	0,382 (0,299)	0,488 (0,415)
Biomed	**0,905** (0,193)	**0,899** (0,229)	0,815 (0,249)	0,863 (0,200)	0,880 (0,189)	0,760 (0,532)	0,885 (0,187)	**0,892** (0,193)
Diabetes	0,744 (0,197)	**0,832** (0,356)	**0,722** (0,174)	0,710 (0,162)	0,698 (0,150)	0,697 (0,150)	0,694 (0,160)	0,704 (0,199)
Média	**0,874**	0,843	0,783	0,786	0,844	0,696	0,750	0,748

o método FBDOCC obteve um melhor desempenho médio que os outros métodos dado uma maior concentração de pontos acima das diagonais. A distância do ponto à diagonal também pode ser vista como uma medida do grau de distinção entre os desempenhos dos métodos. Por exemplo, quanto mais distante um ponto estiver acima da diagonal, melhor é o desempenho do

FBDOCC em relação ao método comparado.

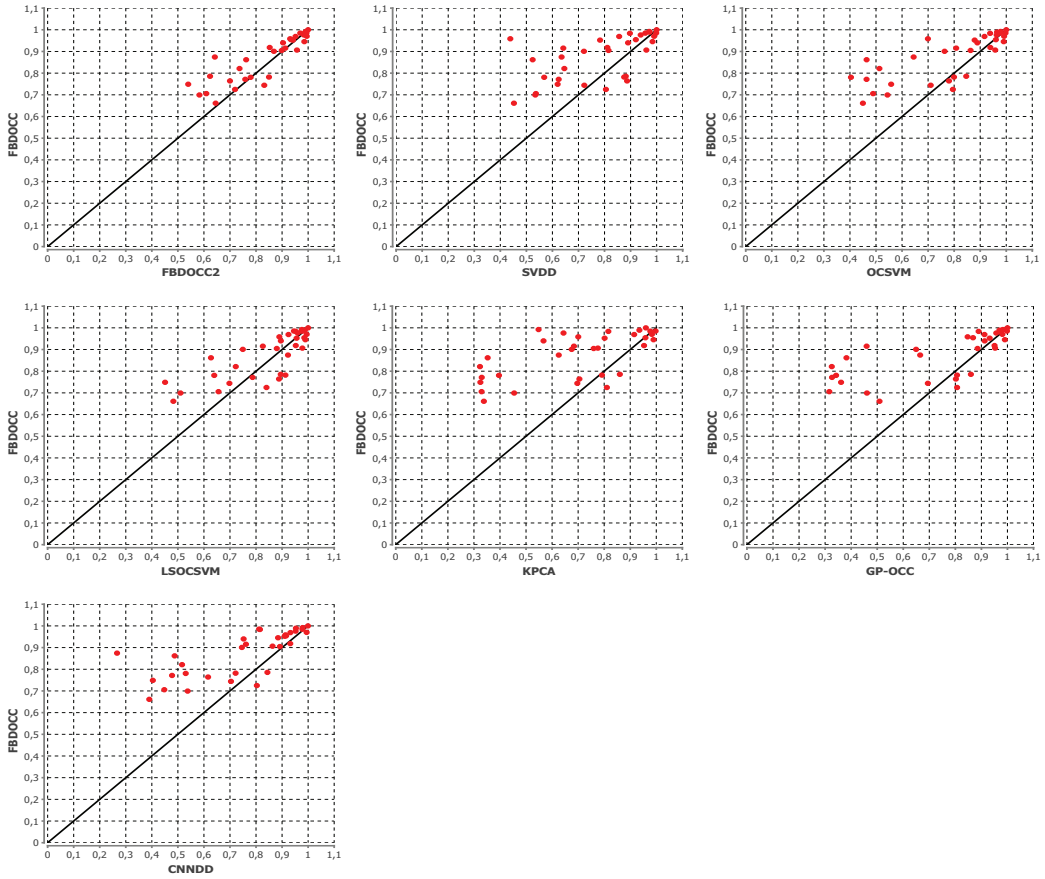

Figura 5.10: Comparação par a par, em termos de AUC, entre o método FBDOCC e o restante dos métodos considerados.

A Tabela 5.3 mostra que o FBDOCC obteve os melhores resultados em vinte e dois dos trinta e três experimentos obtendo também a melhor média geral. Porém, o FBDOCC2 alcançou apenas 13 melhores resultados. O segundo método a obter mais melhores resultados foi o LSOCSVM; 18 de 33.

A Figura 5.11 apresenta um gráfico similar ao da Figura 5.10, porém, considerando a medida MCC. Nessa Figura, pode-se ver uma maior disposição dos pontos por sobre a diagonal, sugerindo um melhor desempenho do método FBDOCC na maioria das comparações. No caso da comparação com o método LSOCSVM, os pontos se distribuem quase simetricamente em volta da diagonal indicando um desempenho similar entre os métodos.

5.5 Análise da Redução de Protótipos

Outro aspecto relevante considerado como métrica de avaliação nos experimentos foi a taxa de redução de protótipos. De acordo com a Tabela 5.4, o método CNNDD obteve a maior

Tabela 5.3: Médias de todos os MCCs para todos os experimentos

Dataset	FBDOCC	FBDOCC2	SVDD	OCSVM	LSOCSVM	ker. PCA	GP-OCC	CNNDD	OCRF
Banana	0,781 (0,332)	0,891 (0,218)	0,893 (0,163)	**0,915** (0,167)	**0,920** (0,124)	**0,920** (0,126)	**0,916** (0,126)	0,562 (0,474)	0,824 (0,214)
Gaussian Distributions	**0,807** (0,234)	0,785 (0,232)	0,683 (0,339)	0,725 (0,200)	0,736 (0,204)	0,723 (0,223)	0,730 (0,219)	**0,697** (0,337)	0,405 (0,227)
Banknote	0,827 (0,326)	**0,975** (0,297)	0,989 (0,089)	0,979 (0,168)	0,802 (0,319)	0,970 (0,136)	**0,986** (0,103)	0,342 (0,339)	0,859 (0,149)
Winsconsin Breast Cancer	0,892 (0,190)	**0,953** (0,137)	**0,938** (0,118)	**0,943** (0,125)	**0,947** (0,117)	0,941 (0,127)	0,941 (0,127)	0,928 (0,230)	**0,943** (0,131)
Glass (1)	**0,474** (0,327)	0,192 (0,384)	0,294 (0,368)	0,222 (0,387)	0,332 (0,659)	-0,006 (0,324)	-0,025 (0,304)	-0,129 (0,448)	-0,182 (0,262)
Glass (2)	**0,278** (0,356)	**0,225** (0,329)	0,120 (0,286)	0,073 (0,328)	**0,376** (0,636)	-0,022 (0,311)	-0,069 (0,254)	-0,050 (0,357)	-0,096 (0,253)
Glass (3)	**0,441** (0,326)	0,260 (0,366)	0,177 (0,334)	0,186 (0,368)	0,162 (0,378)	0,093 (0,334)	0,087 (0,306)	0,020 (0,486)	0,022 (0,334)
Glass (4)	**0,529** (0,334)	**0,470** (0,358)	0,257 (0,448)	0,254 (0,279)	**0,617** (0,556)	0,279 (0,334)	0,335 (0,215)	0,301 (0,343)	-0,035 (0,487)
Glass (5)	**0,787** (0,311)	**0,801** (0,323)	0,598 (0,482)	0,729 (0,271)	**0,829** (0,312)	0,746 (0,255)	**0,809** (0,232)	0,696 (0,463)	**0,808** (0,300)
Ionosphere Bad	0,813 (0,253)	0,851 (0,228)	0,867 (0,239)	0,847 (0,210)	**0,919** (0,188)	0,819 (0,288)	0,737 (0,248)	0,693 (0,338)	0,694 (0,251)
Ionosphere Good	0,338 (0,343)	0,278 (0,301)	**0,679** (0,507)	**0,681** (0,358)	0,464 (0,616)	-0,022 (0,276)	0,013 (0,617)	-0,172 (0,423)	-0,061 (0,326)
Iris (1)	**0,998** (0,088)	**1,000** (0,000)	**1,000** (0,000)	**1,000** (0,000)	**1,000** (0,000)	**1,000** (0,000)	**1,000** (0,000)	**0,944** (0,394)	0,522 (0,302)
Iris (2)	0,745 (0,370)	0,832 (0,275)	0,862 (0,249)	0,844 (0,230)	**0,906** (0,256)	0,523 (0,780)	0,814 (0,245)	0,500 (0,491)	0,681 (0,305)
Iris (3)	**0,945** (0,261)	**0,918** (0,213)	0,860 (0,228)	0,847 (0,259)	**0,905** (0,222)	**0,883** (0,289)	0,847 (0,258)	0,554 (0,546)	0,618 (0,344)
Mfeat-morph (1)	0,841 (0,138)	0,835 (0,148)	0,664 (0,487)	0,754 (0,180)	0,787 (0,287)	0,566 (0,202)	0,622 (0,181)	0,190 (0,626)	**0,879** (0,369)
Mfeat-morph (2)	**0,422** (0,314)	0,330 (0,219)	0,164 (0,372)	0,007 (0,369)	0,247 (0,361)	0,031 (0,181)	-0,026 (0,193)	0,019 (0,427)	0,048 (0,219)
Mfeat-morph (3)	0,463 (0,368)	0,204 (0,442)	0,557 (0,234)	0,513 (0,281)	**0,588** (0,177)	0,562 (0,152)	0,557 (0,146)	0,190 (0,480)	-0,067 (0,437)
Mfeat-morph (4)	**0,345** (0,395)	0,301 (0,350)	0,411 (0,193)	0,408 (0,198)	**0,440** (0,163)	**0,444** (0,157)	**0,441** (0,161)	0,216 (0,419)	0,277 (0,219)
Mfeat-morph (5)	0,436 (0,339)	0,479 (0,297)	0,548 (0,339)	0,433 (0,270)	**0,614** (0,171)	0,415 (0,207)	0,441 (0,192)	0,219 (0,494)	0,058 (0,221)
Mfeat-morph (6)	**0,649** (0,271)	0,404 (0,493)	0,464 (0,416)	**0,666** (0,195)	**0,670** (0,163)	**0,670** (0,163)	0,660 (0,164)	0,493 (0,436)	-0,052 (0,449)
Mfeat-morph (7)	**0,491** (0,274)	0,346 (0,264)	0,213 (0,318)	0,111 (0,286)	**0,355** (0,428)	0,020 (0,295)	0,007 (0,240)	0,043 (0,419)	-0,044 (0,446)
Mfeat-morph (8)	0,443 (0,408)	0,263 (0,424)	0,555 (0,298)	0,388 (0,308)	0,570 (0,275)	0,354 (0,265)	0,483 (0,196)	0,008 (0,480)	**0,959** (0,143)
Mfeat-morph (9)	0,739 (0,283)	0,722 (0,258)	0,399 (0,561)	0,564 (0,285)	0,604 (0,173)	0,438 (0,178)	0,090 (0,322)	0,028 (0,533)	**0,963** (0,131)
Mfeat-morph (10)	**0,423** (0,304)	0,308 (0,240)	0,198 (0,367)	0,065 (0,252)	**0,439** (0,371)	0,013 (0,274)	-0,011 (0,202)	-0,002 (0,499)	-0,015 (0,180)
Seeds (1)	**0,785** (0,339)	**0,711** (0,302)	0,683 (0,266)	0,687 (0,256)	**0,701** (0,295)	0,352 (0,682)	**0,713** (0,267)	0,383 (0,526)	0,609 (0,300)
Seeds (2)	**0,944** (0,259)	**0,942** (0,185)	0,879 (0,226)	0,866 (0,219)	0,865 (0,216)	0,845 (0,371)	0,889 (0,209)	0,801 (0,289)	0,622 (0,301)
Seeds (3)	**0,896** (0,308)	**0,894** (0,233)	0,802 (0,273)	0,829 (0,238)	0,831 (0,241)	0,811 (0,229)	0,810 (0,259)	0,573 (0,491)	0,610 (0,240)
Soybean (17)	**0,584** (0,293)	0,259 (0,288)	**0,496** (0,567)	0,388 (0,502)	**0,681** (0,219)	0,282 (0,448)	0,257 (0,249)	-0,127 (0,495)	-1,000 (0,000)
Wine (1)	**0,884** (0,206)	0,844 (0,232)	0,838 (0,250)	0,854 (0,195)	0,861 (0,216)	0,861 (0,202)	0,849 (0,206)	0,814 (0,278)	0,592 (0,289)
Wine (2)	**0,692** (0,301)	**0,646** (0,301)	0,511 (0,403)	0,548 (0,290)	0,475 (0,319)	0,376 (0,326)	0,369 (0,338)	0,279 (0,500)	0,530 (0,296)
Wine (3)	**0,659** (0,333)	0,482 (0,377)	0,158 (0,346)	0,121 (0,417)	0,282 (0,408)	-0,051 (0,370)	0,003 (0,359)	0,050 (0,526)	**0,603** (0,269)
Biomed	**0,691** (0,288)	**0,595** (0,466)	0,591 (0,308)	**0,636** (0,285)	**0,653** (0,263)	0,634 (0,262)	**0,677** (0,249)	**0,652** (0,290)	0,532 (0,331)
Diabetes	**0,414** (0,217)	**-0,131** (0,812)	**0,379** (0,214)	0,363 (0,204)	0,348 (0,193)	0,356 (0,200)	0,347 (0,197)	0,229 (0,260)	0,274 (0,217)
Média	**0,650**	0,572	0,568	0,559	0,634	0,480	0,494	0,332	0,350

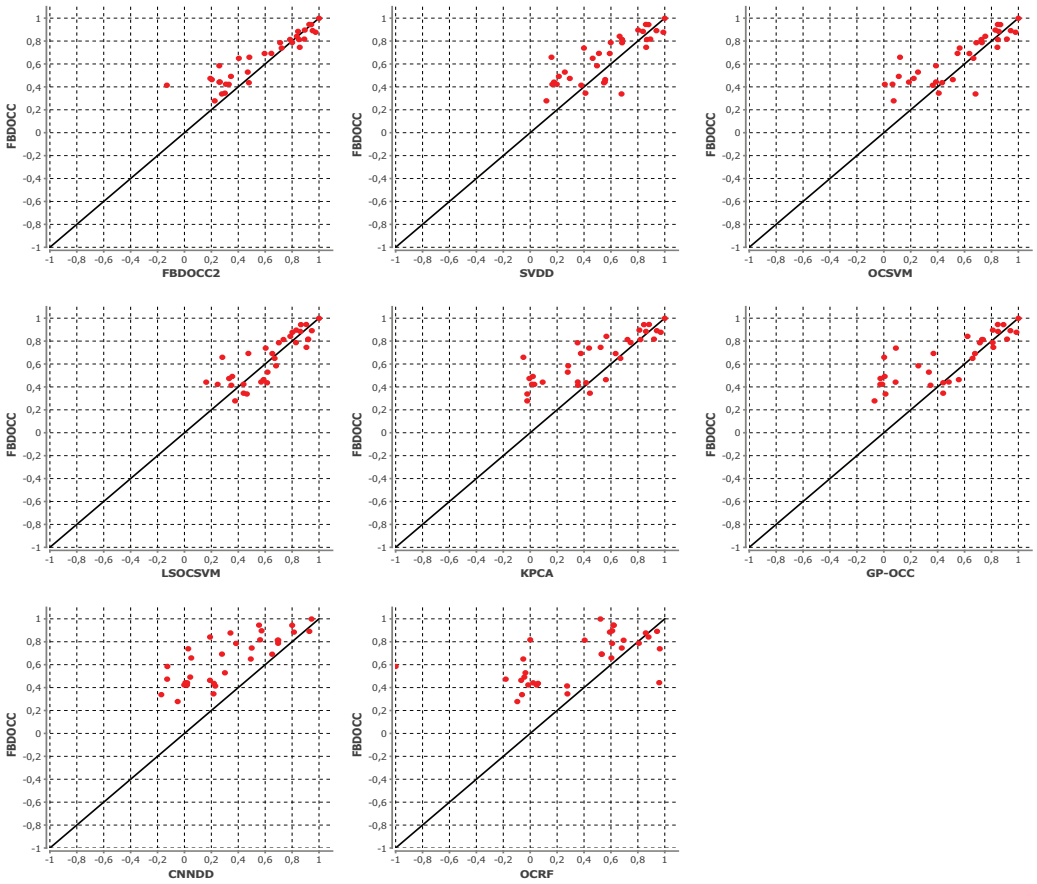

Figura 5.11: Comparação par a par, em termos de MCC, entre o métododo FBDOCC e o restante dos métodos considerados.

taxa de redução de protótipos em trinta em um dos trinta e três experimentos. O FBDOCC armazenou menos protótipos em apenas seis experimentos, enquanto que o SVDD teve o melhor desempenho em 14 bases de dados e o OCSVM em três. O método kernel PCA não pôde ser avaliado nesse aspecto dado que, de fato, ele não retém instâncias do problema mas sim componentes principais; que estão localizados em hiperpaço diferente do das instâncias originais. Em contraste ao OCSVM original, o LSOCSVM não retêm um sub-conjunto de vetores de suporte (*i.e.*, todas as instâncias de treinamento são retidas pelo modelo). É importante enfatizar que, em ambas versões do método proposto, durante a fase de teste, o conjunto de objetos da classe positiva é recriado, ou seja, não existe a necessidade de seu armazenamento pelo modelo.

Tabela 5.4: Percentual de armazenamento do conjunto de treinamento

Dataset	FBDOCC	FBDOCC2	SVDD	OCSVM	LSOCSVM	ker. PCA	GP-OCC	CNNDD	OCRF
Banana	0,18 (0,47)	0,55 (0,51)	0,12 (0,31)	0,37 (0,44)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,09)	–
Gaussian Distributions	**0,03** (0,19)	**0,07** (0,44)	0,16 (0,42)	0,46 (0,48)	100,00 (0,00)	–	100,00 (0,00)	0,04 (0,16)	–
Banknote	0,30 (0,56)	0,89 (0,49)	0,09 (0,18)	0,21 (0,39)	100,00 (0,00)	–	100,00 (0,00)	**0,02** (0,12)	–
Winsconsin Breast Cancer	0,27 (0,53)	**0,06** (0,27)	0,31 (0,30)	0,53 (0,42)	100,00 (0,00)	–	100,00 (0,00)	**0,05** (0,15)	–
Glass (1)	0,34 (0,57)	0,86 (0,57)	**0,09** (0,15)	**0,19** (0,43)	100,00 (0,00)	–	100,00 (0,00)	**0,09** (0,24)	–
Glass (2)	0,47 (0,65)	0,92 (0,51)	**0,06** (0,15)	**0,14** (0,38)	100,00 (0,00)	–	100,00 (0,00)	**0,07** (0,23)	–
Glass (3)	0,60 (0,62)	0,67 (0,67)	**0,14** (0,38)	**0,26** (0,42)	100,00 (0,00)	–	100,00 (0,00)	**0,12** (0,31)	–
Glass (4)	0,25 (0,54)	0,85 (0,58)	**0,09** (0,32)	0,87 (0,44)	100,00 (0,00)	–	100,00 (0,00)	**0,06** (0,20)	–
Glass (5)	**0,18** (0,47)	**0,51** (0,69)	**0,19** (0,43)	0,38 (0,42)	100,00 (0,00)	–	100,00 (0,00)	**0,08** (0,24)	–
Ionosphere Bad	0,62 (0,52)	0,73 (0,47)	0,68 (0,51)	0,81 (0,48)	100,00 (0,00)	–	100,00 (0,00)	**0,07** (0,24)	–
Ionosphere Good	**0,24** (0,45)	0,59 (0,48)	0,93 (0,48)	1,00 (0,06)	100,00 (0,00)	–	100,00 (0,00)	**0,09** (0,22)	–
Iris (1)	0,54 (0,59)	0,67 (0,61)	0,28 (0,30)	0,62 (0,46)	100,00 (0,00)	–	100,00 (0,00)	**0,05** (0,19)	–
Iris (2)	0,59 (0,61)	0,62 (0,58)	0,30 (0,24)	0,53 (0,48)	100,00 (0,00)	–	100,00 (0,00)	**0,12** (0,30)	–
Iris (3)	0,40 (0,58)	**0,33** (0,53)	**0,21** (0,34)	0,53 (0,51)	100,00 (0,00)	–	100,00 (0,00)	**0,16** (0,32)	–
Mfeat-morph (1)	0,06 (0,32)	0,92 (0,51)	**0,01** (0,16)	0,24 (0,35)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,08)	–
Mfeat-morph (2)	0,05 (0,28)	0,92 (0,51)	**0,03** (0,15)	0,21 (0,51)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,07)	–
Mfeat-morph (3)	0,22 (0,59)	0,88 (0,56)	0,04 (0,10)	0,38 (0,60)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,08)	–
Mfeat-morph (4)	0,27 (0,64)	**0,72** (0,67)	0,03 (0,11)	0,21 (0,56)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,08)	–
Mfeat-morph (5)	0,28 (0,60)	0,96 (0,44)	0,04 (0,12)	0,08 (0,21)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,07)	–
Mfeat-morph (6)	0,48 (0,66)	0,72 (0,66)	**0,16** (0,53)	0,73 (0,42)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,08)	–
Mfeat-morph (7)	**0,03** (0,20)	0,81 (0,62)	0,05 (0,13)	0,17 (0,43)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,08)	–
Mfeat-morph (8)	**0,15** (0,56)	0,86 (0,58)	0,04 (0,08)	0,19 (0,47)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,08)	–
Mfeat-morph (9)	0,13 (0,51)	0,92 (0,52)	**0,02** (0,18)	0,36 (0,48)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,07)	–
Mfeat-morph (10)	0,24 (0,61)	0,84 (0,60)	0,04 (0,13)	0,29 (0,44)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,07)	–
Seeds (1)	**0,40** (0,59)	0,79 (0,61)	0,24 (0,21)	0,54 (0,49)	100,00 (0,00)	–	100,00 (0,00)	**0,11** (0,27)	–
Seeds (2)	0,36 (0,56)	**0,51** (0,65)	**0,18** (0,28)	0,57 (0,46)	100,00 (0,00)	–	100,00 (0,00)	**0,13** (0,30)	–
Seeds (3)	0,46 (0,61)	0,81 (0,62)	**0,20** (0,45)	0,57 (0,44)	100,00 (0,00)	–	100,00 (0,00)	**0,14** (0,30)	–
Soybean (17)	0,69 (0,58)	0,87 (0,53)	0,95 (0,10)	0,98 (0,10)	100,00 (0,00)	–	100,00 (0,00)	**0,01** (0,10)	–
Wine (1)	0,72 (0,59)	0,72 (0,64)	**0,19** (0,46)	0,70 (0,51)	100,00 (0,00)	–	100,00 (0,00)	**0,12** (0,29)	–
Wine (2)	0,48 (0,59)	0,96 (0,43)	0,23 (0,45)	0,42 (0,42)	100,00 (0,00)	–	100,00 (0,00)	**0,14** (0,32)	–
Wine (3)	0,33 (0,53)	0,97 (0,40)	0,19 (0,38)	0,38 (0,46)	100,00 (0,00)	–	100,00 (0,00)	**0,09** (0,27)	–
Biomed	0,42 (0,61)	**0,47** (0,68)	**0,09** (0,35)	0,53 (0,52)	100,00 (0,00)	–	100,00 (0,00)	**0,16** (0,29)	–
Diabetes	0,34 (0,65)	**0,18** (0,60)	0,06 (0,25)	0,27 (0,50)	100,00 (0,00)	–	100,00 (0,00)	0,04 (0,14)	–
Média	0,33	0,70	0,20	0,45	100,00	100,00	–	**0,06**	–

5.6 Análise do Custo Computacional

O terceiro item avaliado nos experimentos foi o custo computacional durante o treinamento. Vale salientar que todas as simulações foram realizadas em máquinas com a mesma configuração. Na Tabela 5.5, os tempos médios de treinamento das vinte e cinco execuções de cada experimento são apresentados. Nos experimentos, a heurística para a melhoria do tempo de treinamento proposta na Seção 4.7 foi utilizada por ambas as versões do FBDOCC. Em relação ao tempo de treinamento, o FBDOCC e o CNNDD foram melhores em trinta e dois e vinte e três do total de experimentos, respectivamente (considerando o teste de Wilcoxon *Signed-rank*). O método FBDOCC2, como esperado, obteve tempos de execução muito acima dos obtidos

pela versão original do método. Os tempos obtidos pelos métodos que se utilizam de operações com matrizes (LSOCSVM e kernel PCA) sugerem um alto custo computacional durante a fase de treinamento, especialmente em bases de dados contendo um maior número de instâncias. O OCRF também demonstrou um alto custo computacional em comparação aos outros métodos.

Tabela 5.5: Tempo de treinamento em milisegundos

Dataset	FBDOCC	FBDOCC2	SVDD	OCSVM	LSOCSVM	ker. PCA	GP-OCC	CNNDD	OCRF
Banana	**3,24** (2,41)	22,04 (7,30)	10,60 (4,60)	24,80 (5,51)	86,52 (6,87)	372,64 (9,14)	–	**5,36** (3,69)	1875,20 (9,02)
Gaussian Distributions	**0,68** (1,71)	2,28 (2,22)	1,48 (1,13)	5,60 (2,09)	34,64 (2,19)	322,12 (23,28)	–	1,24 (1,02)	7021,60 (17,24)
Banknote	**2,52** (2,99)	786,80 (19,55)	2,64 (1,29)	6,80 (1,93)	154,76 (2,49)	1495,60 (19,94)	–	2,20 (1,35)	13210,00 (17,59)
Winsconsin Breast Cancer	**0,88** (0,81)	43,88 (10,10)	2,92 (1,15)	6,32 (1,13)	37,40 (1,49)	191,88 (4,69)	–	1,80 (1,05)	7991,20 (15,68)
Glass (1)	**0,08** (0,52)	46,60 (7,64)	**0,16** (0,61)	0,40 (0,70)	2,44 (0,76)	6,56 (1,40)	–	0,44 (0,70)	2150,80 (7,40)
Glass (2)	**0,40** (0,75)	66,68 (8,16)	**0,28** (0,73)	**0,44** (0,70)	9,08 (3,24)	14,24 (1,65)	–	**0,44** (0,70)	3046,00 (9,60)
Glass (3)	**0,28** (0,73)	22,20 (4,36)	**0,16** (0,61)	**0,40** (0,70)	2,16 (0,61)	5,88 (0,98)	–	**0,40** (0,70)	2071,60 (7,91)
Glass (4)	**0,28** (0,73)	91,16 (8,62)	**0,52** (0,80)	1,76 (1,03)	9,92 (3,44)	20,84 (4,28)	–	**0,52** (0,71)	3208,00 (7,94)
Glass (5)	**0,20** (0,63)	36,84 (7,01)	**0,40** (0,70)	1,08 (0,86)	6,88 (2,91)	13,56 (1,82)	–	**0,48** (0,71)	2661,20 (9,89)
Ionosphere Bad	2,00 (0,92)	1291,64 (22,41)	9,72 (3,45)	5,36 (1,25)	12,76 (1,78)	31,24 (3,21)	–	**1,04** (1,71)	9717,60 (14,74)
Ionosphere Good	**1,84** (1,45)	419,60 (14,09)	**1,88** (1,16)	2,60 (1,12)	4,60 (1,74)	12,32 (3,70)	–	**1,56** (1,69)	5639,60 (11,13)
Iris (1)	**0,04** (0,44)	2,08 (1,75)	3,12 (2,69)	**0,28** (0,67)	0,92 (0,52)	2,16 (0,68)	–	0,52 (1,14)	1410,00 (6,54)
Iris (2)	**0,08** (0,52)	3,92 (3,45)	1,24 (1,83)	0,56 (0,76)	0,80 (0,89)	2,40 (0,83)	–	**0,12** (0,57)	1212,80 (5,71)
Iris (3)	**0,08** (0,52)	0,92 (0,92)	**0,24** (0,65)	0,40 (0,70)	1,20 (0,92)	3,04 (1,66)	–	**0,24** (0,65)	1272,40 (6,25)
Mfeat-morph (1)	**2,72** (1,55)	11326,52 (65,09)	**3,32** (1,97)	59,28 (5,39)	3573,08 (7,59)	25842,52 (35,73)	–	**5,32** (2,22)	32232,40 (29,55)
Mfeat-morph (2)	**2,32** (1,57)	10050,28 (61,30)	5,20 (1,98)	42,32 (6,37)	3566,44 (6,14)	25775,68 (50,36)	–	**2,76** (1,71)	33614,40 (21,31)
Mfeat-morph (3)	**2,64** (1,40)	9555,68 (68,24)	6,12 (1,39)	58,04 (6,82)	3586,60 (9,04)	22952,84 (47,05)	–	6,68 (2,23)	32677,20 (22,31)
Mfeat-morph (4)	**2,28** (1,21)	5226,00 (62,83)	5,92 (1,47)	39,64 (6,14)	3563,44 (6,96)	24500,48 (45,79)	–	8,04 (2,38)	32334,40 (23,20)
Mfeat-morph (5)	**2,40** (1,43)	9527,04 (57,95)	6,40 (1,47)	31,60 (4,34)	3561,44 (5,29)	23965,44 (34,09)	–	**3,64** (1,97)	33488,00 (28,64)
Mfeat-morph (6)	**1,60** (0,70)	6959,32 (75,98)	21,04 (5,64)	105,56 (3,43)	3592,44 (10,79)	22927,24 (38,21)	–	8,28 (2,43)	32680,80 (27,33)
Mfeat-morph (7)	**2,12** (1,18)	9537,56 (74,82)	6,52 (1,49)	47,28 (6,45)	4084,68 (26,56)	22249,96 (51,91)	–	**4,32** (2,34)	33162,40 (28,63)
Mfeat-morph (8)	**2,28** (1,17)	7942,08 (68,90)	6,80 (1,23)	41,04 (5,65)	3759,12 (15,28)	21151,20 (29,55)	–	5,00 (2,23)	28838,00 (25,79)
Mfeat-morph (9)	**2,48** (1,22)	11043,16 (65,23)	3,44 (2,34)	66,32 (6,34)	3815,52 (14,82)	25699,84 (26,39)	–	**5,84** (2,05)	31278,40 (24,97)
Mfeat-morph (10)	**2,32** (1,21)	9239,88 (68,27)	6,00 (1,77)	72,92 (6,53)	3693,00 (13,99)	24232,00 (52,63)	–	**3,36** (2,08)	33034,00 (28,53)
Seeds (1)	**0,12** (0,57)	22,60 (4,96)	**0,28** (0,67)	0,60 (0,70)	2,00 (0,53)	6,16 (1,17)	–	**0,40** (0,75)	2363,60 (9,57)
Seeds (2)	**0,24** (0,72)	24,08 (6,76)	**0,40** (0,70)	0,64 (0,69)	2,00 (0,75)	6,44 (2,02)	–	**0,24** (0,65)	2410,40 (11,73)
Seeds (3)	**0,20** (0,63)	53,36 (7,01)	**0,32** (0,68)	0,72 (0,73)	2,04 (0,67)	6,00 (0,83)	–	**0,28** (0,67)	2519,20 (9,09)
Soybean (17)	**30,48** (1,88)	624304,04 (487,94)	53,24 (1,99)	75,84 (1,62)	172,64 (2,69)	402,00 (7,76)	–	**1,56** (1,21)	42534,40 (21,51)
Wine (1)	**0,24** (0,65)	99,84 (8,53)	**0,28** (0,67)	0,64 (0,75)	1,96 (0,88)	4,36 (1,56)	–	**0,76** (1,11)	2494,40 (9,71)
Wine (2)	**0,36** (0,69)	94,80 (5,35)	**0,24** (0,65)	**0,56** (0,70)	1,28 (0,67)	2,96 (0,77)	–	**0,36** (0,69)	2354,80 (7,96)
Wine (3)	**0,28** (0,67)	150,52 (8,47)	**0,36** (0,69)	0,60 (0,75)	2,20 (0,63)	5,52 (1,32)	–	**0,48** (0,71)	2699,60 (8,58)
Biomed	**0,04** (0,44)	2,76 (1,94)	**0,08** (0,52)	0,52 (0,71)	1,68 (0,94)	4,72 (0,73)	–	**0,28** (0,67)	1714,40 (8,47)
Diabetes	**0,88** (1,02)	162,04 (18,63)	**1,16** (0,98)	4,00 (1,67)	52,40 (1,44)	240,64 (3,74)	–	1,92 (1,09)	9347,20 (15,48)
Média	**2,08**	21762,27	4,92	21,36	1133,27	7347,59	–	2,30	13765,64

A Figura 5.12 mostra o percentual do custo computacional do FBDOCC com e sem a Heurística 1 em relação ao pior caso. Os valores apresentados na Figura mostram que ambos os treinamentos têm custo consideravelmente menores que o pior caso. Além disso, o uso da heurística apresentada na Seção 4.7 diminuiu consideravelmente o custo do algoritmo sem o uso da heurística. A redução no custo foi alcançada em todos os experimentos.

5.7 Caso de Estudo: Detecção de Cardiopatias em Crianças com Sintomas de Sopro Cardíaco

A medicina é um domínio complexo e ainda longe de ser entendido em todos seus aspectos. Dessa forma, a aplicação de sistemas inteligentes a problemas médicos é uma área com bastantes problemas a serem explorados(HORN, 2001). Doenças cardíacas são uma das maiores causas das altas taxas de morbidez e mortalidade, não apenas no Brasil, mas também em sociedades mais desenvolvidas (REPORT, 2010). Várias técnicas e ferramentes diferentes estão sendo desenvolvidas de forma contínua com a finalidade de analisar o desempenho do coração. Em várias técnicas, o diagnóstico de uma doença cardíaca se baseia na combinação do histórico do paciente associado a exames físicos. Essa confirmação vem de exames como eletrocardiograma (ECG) e ecocardiograma (ECHO).

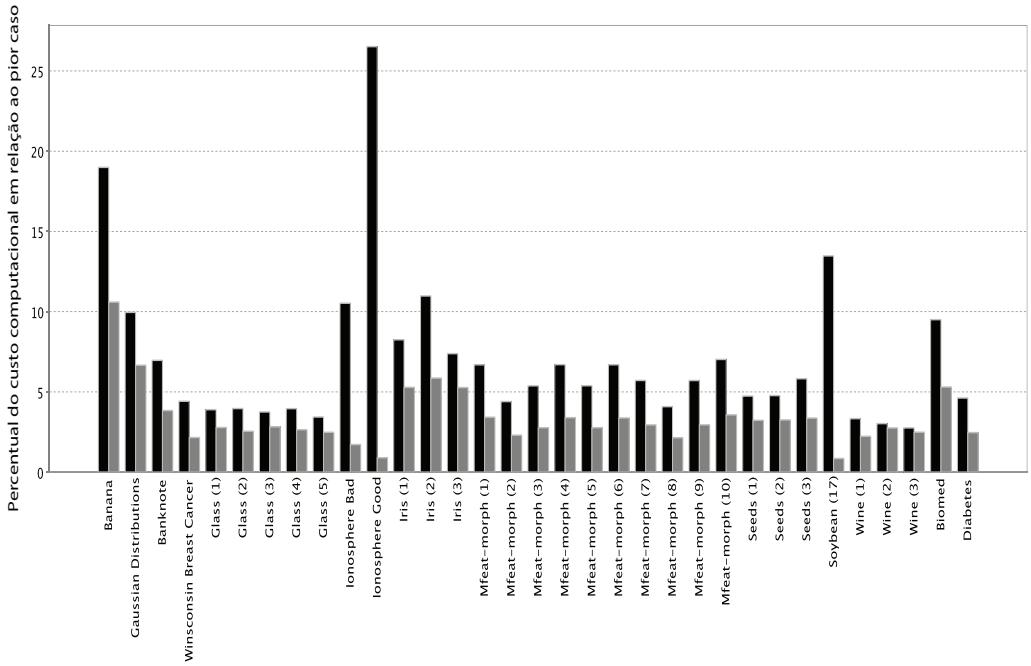

Figura 5.12: Percentual do custo computacional em relação ao pior caso para o FBDOCC com a heurística da Seção 4.7 (colunas em cinza) e sem a heurística (colunas em preto).

Na maioria dos casos, principalmente em crianças, esses exames são necessários antes que se estabeleça um diagnóstico da doença e sua severidade. Porém, esses exames nem sempre estão disponíveis, especialmente em países em desenvolvimento onde existe uma carência de especialistas, como cardiologistas pediátricos, em serviços de emergência de vários centros urbanos. Esse cenário pode levar a atrasos no diagnóstico, e como consequência, uma deterioração na condição clínica do paciente.

Doenças cardíacas congênitas afetam aproximadamente 8 em cada 1000 recém nascidos, e de 25% a 30% são casos graves. Esses casos requerem tratamento médico ou cirúrgico durante o primeiro ano de vida (HOQUE et al., 2008). Quanto mais precoce o diagnóstico e/ou intervenção, melhor o prognóstico dessas crianças.

Um sopro cardíaco é um som gerado pela turbulência do fluxo sanguíneo. Esse sopro pode ocorrer em diferentes regiões do sistema cardiovascular e em diferentes fases do ciclo cardíaco (sistole, diástole ou ambas) e ter diferentes frequências e durações.

O sopro cardíaco é o sinal de alerta mais frequente de um problema cardíaco. Em um estudo conduzido em Israel (REIN; OMOKHODION; NIR, 2000), 86% dos recém nascidos com sopro cardíaco nos primeiros dias de vida demonstraram algum tipo de doença cardíaca estrutural. Outro estudo (AINSWORTH; WYLLIE; WREN, 1999) demonstrou que aproximadamente 44% das deformações podem ser detectadas no período neonatal e que quando um sopro car-

díaco é detectado em um exame físico, as chances de uma lesão estrutural são por volta de 54%. Na verdade, crianças saudáveis podem apresentar sopro cardíaco, geralmente referido como *inocente*. A diferenciação precoce entre um sopro cardíaco patológico e um *inocente* é fundamental em crianças com suspeitas de doenças cardíacas.

Particularmente, o sopro cardíaco quando associado com outros sintomas, é uma indicação para que haja uma melhor investigação no intuito de identificar condições favoráveis a certas doenças. Cardiologistas experientes podem com frequência diferenciar com sucesso um sopro patológico de um "inocente". Porém, essa habilidade requer um longo período de treinamento e prática e por isso se limita a um pequeno grupo de profissionais.

Técnicas de inteligência computacional têm se mostrado efetivas no provimento de um melhor entendimento de dados médicos (SARTAKHTI; ZANGOOEI; MOZAFARI, 2012) (ER; YUMUSAK; TEMURTAS, 2010) (DELEN; WALKER; KADAM, 2005). Sistemas Médicos de Suporte à Decisão são desenvolvidos para dar suporte a médicos e outros profissionais da saúde em problemas como diagnóstico por imagem ou instruções no tratamento clínico. Com essa finalidade, a maioria dos trabalhos emprega técnicas de inteligência computacional para modelar cada problema em particular. Porém, em dados médicos coletados para a construção de um modelo computacional do problema, o número de exemplos de pacientes saudáveis é frequentemente muito maior que a quantidade de pacientes portadores de alguma desordem. Em outras palavras, dados médicos frequentemente apresentam um considerável desbalanceamento entre as classes. Daí a necessidade do uso de técnicas específicas para problemas com essa característica.

Recentemente, Farquad e Bose (FARQUAD; BOSE, 2012) propuseram um método que usa Máquinas de Vetores de Suporte (SVMs) para o pré-processamento dos dados com a finalidade de balancear as classes antes da geração do modelo do classificador. Outra opção para problemas com dados desbalanceados é o uso de métodos que não sofram com essa característica. O uso da classe minoritária durante o treinamento pressupõe que o montante de dados dessa classe representa bem o problema. Na verdade, essa suposição pode não funcionar para a maioria dos problemas, a menos que o problema seja muito simples, o que não é comum em problemas reais. Neste caso de estudo, ao invés de previamente tentar resolver o desbalanceamento, tenta-se criar uma superfície fechada que englobe os exemplos da classe majoritária deixando os exemplos da classe minoritária fora dessa descrição. Com essa finalidade, todos os classificadores já experimentados foram aplicados a esse problema específico.

5.7.1 Base de Dados

A base de dados utilizada nessa pesquisa consiste da coleta de informações (realizada em um renomado hospital pediátrico brasileiro) sobre 1792 pacientes com sintomas de sopro cardíaco. Ao entrar no hospital, pacientes infantis passam por procedimentos padrões para a coleta de informações não invasivas como frequencia cardíaca, altura (em metros) e peso. A

Tabela 5.6 descreve os dados coletados pelo sistema cadastral do hospital.

Tabela 5.6: Característica Inicial dos Dados.

Atributo	Tipo	Estatística
Gênero	nominal	53% masculino
Peso	real	19,5 Kg ($\pm 14,6$)
Altura	real	1,0 m ($\pm 0,32$)
Frequência cardíaca	inteiro	103 ($\pm 25,4$)
Data de Nascimento	nominal	-
Data de atendimento	nominal	-
Cardíaco (classe)	booleano	24,2% sim

Na Tabela 5.6, a coluna Estatística de atributos numéricos apresentam médias seguidas pelos desvios padrões da base de dados para aquele atributo. É possível se observar que a base de dados é desbalanceada devido à discrepância de exemplos da classe cardiopata (classe positiva) e não cardiopata (classe negativa).

Com a finalidade de tornar os dados mais adequados ao uso dos classificadores, foi realizado um pre-processamento nos dados originais. O atributo *altura*, por exemplo, foi coletado nas métricas centímetros e metros. Nesse caso todos os dados foram convertidos para metros. O atributo *Peso* foi convertido de gramas para quilogramas. O atributo *idade* foi processado de acordo com a Equação (5.4). Além dos atributos apresentados na Tabela 5.6, os atributos *faixa etária*, *Gênero Corretamente Preenchido*, *Índice de Massa Corpórea* (IMC) e *Superfície Corpórea* (SC) foram derivados dos dados originais na tentativa de se agregar mais informação ao problema.

$$Idade = \frac{(DatadeAtendimento - DatadeNascimento)}{360}. \tag{5.4}$$

O atributo *Gênero Corretamente Preenchido* é uma *flag* adicionada devido à alta quantidade de valores faltosos no atributo *gênero*. Geralmente, crianças são classificadas em faixas etárias especificadas pela equipe médica. A Tabela 5.7 apresenta o intervalo de idade para cada faixa etária.

O atributo IMC é uma medida internacional usada para classificar um individuo com base em seu peso e altura. A Equação (5.5) mostra como computar esse índice.

$$IMC = \frac{Peso}{Altura^2}. \tag{5.5}$$

A Superfície Corpórea consiste na área da pele em metros quadrados. Esse atributo é um fator importante para o cálculo da dosagem correta de alguns medicamentos. Por exemplo, dosagens de medicamentos quimioterápicos são mais influenciadas pela SC que pelo peso do paciente. A Equação (5.6) calcula o atributo SC.

Tabela 5.7: Faixa etária dos pacientes.

quando	então
menos que 29 dias	Neonato
entre 30 dias e 1 ano	Infante
entre 1 e 6 anos	Pre-escolar
entre 7 e 9 anos	Estudante
entre 10 e 13 anos	Pre-adolescente
entre 14 e 17 anos	Adolescente
mais que 18 anos	Adulto

$$SC = 0,007184 \times Altura^2 \times Peso^{0,425} \tag{5.6}$$

Após a fase de pre-processamento, a base de dados final é composta por nove atributos e o alvo. São eles: (i) Faixa Etária; (ii) Gênero; (iii) IMC; (iv) Frequência Cardíaca; (v) Gênero Corretamente Preenchido; (vi) Idade; (vii) Altura; (viii) SC; e (ix) peso.

5.7.2 Métricas para Avaliação

Por se tratar de um problema médico, o custo do erro Tipo I (*i.e.*, classificar um paciente enfermo como saudável) é maior que o custo do erro Tipo II (*i.e.*, classificar um paciente saudável como enfermo). Dessa forma, foi tomada como principal métrica de avaliação a Sensibilidade (cujo cálculo é ilustrado na Equação (5.7)). A Sensibilidade indica a taxa de acerto na classe positiva.

$$Sensibilidade = \frac{VP}{VP+FN}. \tag{5.7}$$

A Sensibilidade se relaciona apenas com o erro Tipo I, dessa forma ela não pode ser usada como única métrica para a definição de um bom modelo de classificador. A Especificidade, ilustrada na Equação (5.8), define o erro Tipo II.

$$Especificidade = \frac{VN}{VN+FP}. \tag{5.8}$$

A obtenção dos Erros tipos I e II só é possível com a escolha de um ponto de operação adequado na curva ROC. O ponto de operação da curva ROC mais próximo da coordenada (0, 1), visa a obtenção do menor erro considerando o mesmo peso para os erros tipo I e II. A título experimental e em conversas com especialistas da área médica, foi estabelecido para os experimentos que o ponto de operação a ser utilizado deveria resultar em uma Sensibilidade mínima de 81% (i.e., erro Tipo I máximo de 19%).

A última métrica (descrita em detalhes no Apêndice B.2) utilizada para a avaliação desse caso de estudo foi a distância de Kolmogorov-Smirnov (FRIEDMAN, 1977). Essa mé-

trica mede a distinção entre as Funções de Densidade de Probabilidade (FDP) das distribuições negativa e positiva previstas pelo modelo. Uma maior distinção entre as classes acontece quando existe uma maior distância. Além das métricas mencionadas, também foi utilizada a métrica MCC (Equação (5.3)).

5.7.3 Otimização de Parâmetros

Considerando as particularidades de cada método, foi realizado um processo de seleção de parâmetros guiado de forma a maximizar a taxa de acerto do modelo mantendo a Sensibilidade (Equação (5.7)) acima de 81%[2] no conjunto de validação. Dessa forma, tenta-se maximizar o acerto em ambas as classes tendo como restrição um acerto mínimo na classe positiva de 81%. Como a base de dados teve uma de suas primeiras utilizações nesse estudo, durante a otimização dos parâmetros dos métodos FBDOCC e OCSVM[3] foi adicionado outro parâmetro para a seleção do conjunto de características (atributos) do problema que resulta na menor taxa de erro. O conjunto de características indicado por essa busca foi também utilizado pelos outros classificadores.

Para os métodos FBDOCC e OCSVM, o algoritmo PSO foi empregado na otimização dos parâmetros (r, th) para o FBDOCC$_{f2}$ e (γ, ν) para o OCSVM, assim como a seleção de características. Em ambos os classificadores, a otimização convergiu para os seis seguintes atributos do problema: Peso, Altura, Frequência Cardíaca, Idade, SC e IMC. Dessa forma, os atributos restantes não foram utilizados nos experimentos.

Encontrado o conjunto de características a ser utilizado nos experimentos, a otimização ocorreu de forma análoga à descrição da Seção 5.1.2, com a diferença que a função de *fitness* a ser otimizada agora considera apenas o MCC para todos os métodos.

5.7.4 Análise dos Resultados

A Tabela 5.8 compreende todas as medidas citadas na Seção 5.7.2 salvo a distância KS que será apresentada na Figura 5.13. Os modelos de classificadores da Tabela 5.8 foram escolhidos com base no MCC de validação e posteriormente avaliados no conjunto de teste. Cada célula da Tabela contém a média, seguidas por seus desvios padrões, das execuções dos vinte e cinco melhores modelos gerados. Segundo a Tabela 5.8, fica claro o melhor desempenho do método FBDOCC sobre os outros métodos. Os resultados mostram que o FBDOCC obteve o melhor desempenho na classificação seguido pelo SVDD. O método OCRF é uma exceção nos experimentos pelo fato de não ser possível atribuir previamente um valor específico para a sensibilidade; como nos outros experimentos. Isso explica o baixo valor para a sensibilidade e o alto valor para a especificidade.

[2]Esse valor foi definido em conversa com especialistas da área médica.

[3]Esses métodos foram considerados na seleção de características devido ao bom desempenho alcançado por ambos nos experimentos anteriores.

Tabela 5.8: Sumário do desempenho na classificação para a base de doenças cardíacas.

Classificador	AUC	MCC	Sensibilidade	Especificidade
FBDOCC	**0,833** $\pm(0,102)$	**0,491** $\pm(0,207)$	**0,807** $\pm(0,180)$	0,700 $\pm(0,234)$
FBDOCC2	0,812 $\pm(0,137)$	0,441 $\pm((0,224)$	**0,812** $\pm(0,037)$	0,644 $\pm(0,232)$
SVDD	**0,832** $\pm(0,111)$	0,466 $\pm(0,199)$	**0,813** $\pm(0,048)$	0,669 $\pm(0,201)$
OCSVM	0,828 $\pm(0,116)$	0,464 $\pm(0,183)$	**0,812** $\pm(0,052)$	0,668 $\pm(0,187)$
LSOCSVM	0,800 $\pm(0,132)$	0,415 $\pm(0,168)$	**0,812** $\pm(0,045)$	0,615 $\pm(0,175)$
KPCA	0,723 $\pm(0,140)$	0,309 $\pm(0,175)$	**0,813** $\pm(0,053)$	0,499 $\pm(0,185)$
GP-OCC	0,723 $\pm(0,127)$	0,262 $\pm(0,189)$	**0,812** $\pm(0,050)$	0,448 $\pm(0,198)$
CNNDD	0,759 $\pm(0,294)$	0,363 $\pm(0,439)$	0,551 $\pm(0,466)$	0,551 $\pm(0,466)$
OCRF	–	0,252 $\pm(0,262)$	0,183 $\pm(0,229)$	**0,968** $\pm(0,090)$

O teste de Kolmogorov-Smirnov, como mencionado anteriormente, foi usado nesse trabalho como forma de avaliar a distinção entre as classes para os melhores modelos de cada classificador. Na Figura 5.13 as linhas azul e preta representam as funções de distribuição acumuladas (*Cumulative Distribution Function* - CDF) para as classes positiva (cardiopata) e negativa (saudável), respectivamente. O ponto de maior distância entre as linhas representa a distância KS (*i.e.*, o valor máximo da curva vermelha). Como esperado, após a análise dos outros resultados, o desempenho do FBDOCC supera o dos outros classificadores e tem desempenho pouco acima do OCSVM. Os gráficos apresentados na Figura 5.13, servem para apresentar o poder dos classificadores na distinção entre as classes positivas e negativas.

5.8 Considerações Finais

Este capítulo apresentou experimentos para a detecção de novidades em 33 bases de dados diferentes assim como um caso de estudo envolvendo uma base de dados coletada em um hospital local. As 33 bases de dados utilizadas compreendem problemas com duas classes, com múltiplas classes e foram geradas artificialmente ou obtidas em repositórios públicos.

Os primeiros experimentos tiveram a finalidade de avaliar de forma visual a proposta da criação de uma descrição fechada dos dados da classe normal. Para isso, foram utilizadas bases bidimensionais descritas em trabalhos relacionados. Esses experimentos demonstraram que o método proposto é capaz de descrever de forma eficiente dados da classe normal com formatos complexos.

Três aspectos do algoritmo proposto foram avaliados: precisão na classificação, redução de protótipos e tempo de treinamento. Para se obter uma referência nas comparações, outros seis métodos foram utilizados nos experimentos. Desses, três métodos foram implementados e três foram obtidos de pacotes de software disponíveis online. Duas versões do método proposto foram avaliadas sendo que a versão FBDOCC se sobressaiu sobre a FBDOCC2 e sobre os outros métodos. Dentre todos os métodos, essa versão foi melhor em vinte e dois dos trinta e três experimentos considerando à precisão na classificação (métrica MCC). Considerando o

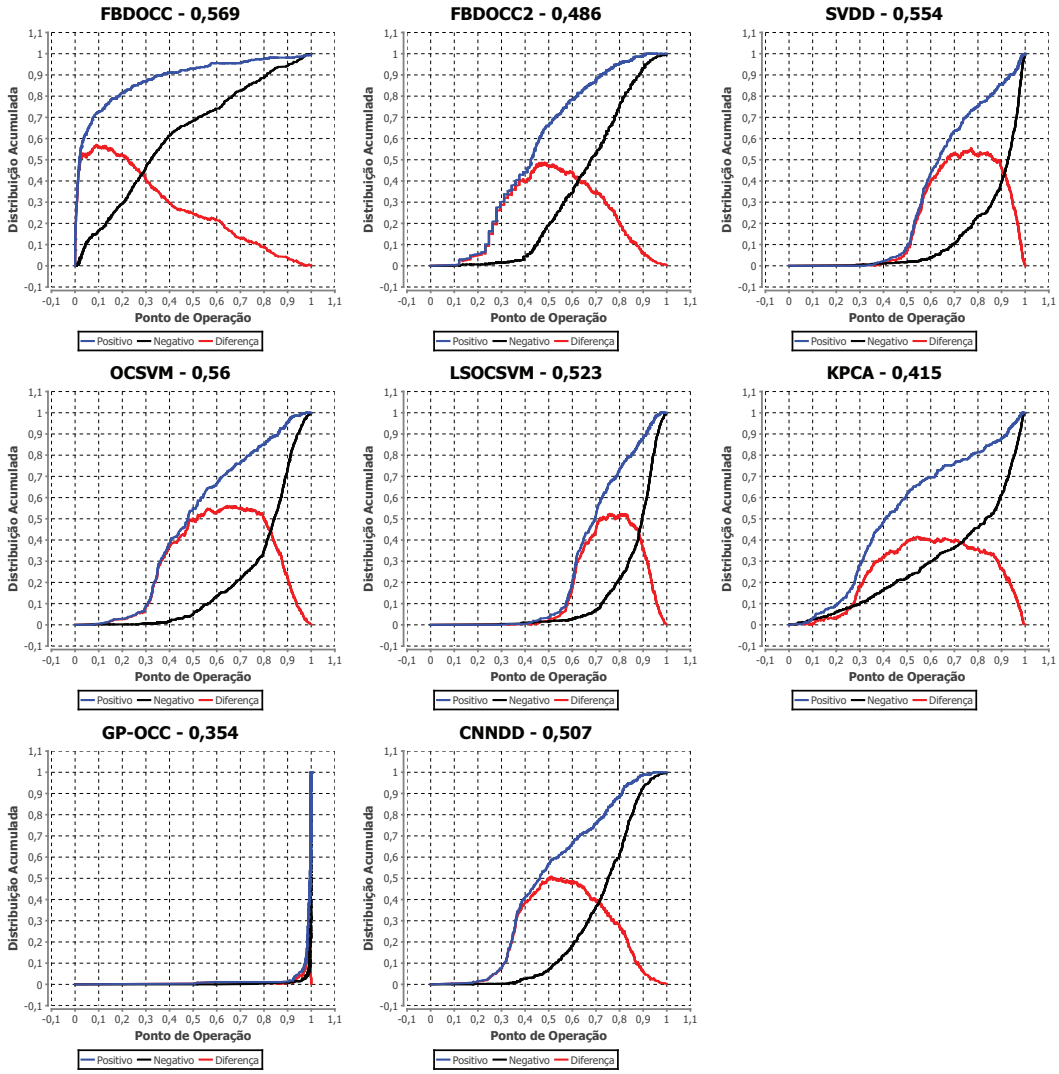

Figura 5.13: Distâncias KS para os melhores modelos de cada método.

desempenho médio dos trinta e três experimentos, a versão FBDOCC também obteve a maior média.

A redução de protótipos é um aspecto de grande relevância no aprendizado baseado em instâncias (IBL). Dentre os métodos que realizam a redução de protótipos, o método CNNDD e o SVDD foram o melhores na média final dos trinta e três experimentos. A versão FBDOCC do método proposto obteve o terceiro melhor resultado e foi melhor em seis dos trinta e três experimentos. Vale salientar que a função de aptidão para a otimização dos parâmetros utilizada nessa versão penaliza modelos com baixas taxas de redução de protótipos.

Uma análise do custo computacional foi realizada no Capítulo 4, porém, essa análise pode não ser realística na prática. O terceiro aspecto abordado nos experimentos foi o tempo de treinamento. Para esse aspecto o método foi analisado com o uso de uma heurística que visa minimizar a realização de comparações desnecessárias. Com o uso dessa heurística, a versão FBDOCC obteve trinta e dois melhores resultados dentre os trinta e três experimentos. A versão

FBDOCC2 se mostrou bastante ineficiente nesse aspecto. Foi avaliada ainda a melhoria no custo computacional do método com e sem o uso dessa heurística em relação ao pior caso. Com a utilização da heurística, o método FBDOCC, em seu pior desempenho, chegou a 11% do pior caso enquanto que sem a heurística ele chegou a 27%.

Foi ainda realizado um caso de estudo para a detecção de cardiopatias em pacientes crianças com sintomas de sopro cardíaco analisando-se dados não invasivos. A motivação por trás desse estudo vem da pouca quantidade de exemplos da classe de cardiopatas, ou seja, a base encontra-se desbalanceada. Dessa forma, é intuitivo o uso de uma abordagem que englobe os dados da classe normal e classifique exemplos fora dessa descrição como desconhecidos pois não se sabe o quão representativos são os exemplos da classe positiva para a fase de modelagem. Nesse estudo, a versão FBDOCC se mostrou bastante eficiente, assim como o método SVDD.

6

Conclusão e trabalhos Futuros

Essa tese introduziu duas versões de um novo método para detecção de novidades com exemplos de uma única classe (*One-class Classification*): *Feature Boundaries Detector for One-class Classification* (FBDOCC). A partir desse método foi derivada uma nova versão (FBDOCC2) capaz de aumentar o poder na geração de descrições fechadas da classe normal. Uma heurística também foi desenvolvida no intuito de diminuir a demanda por recursos computacionais durante a fase de treinamento. Foram realizados vários experimentos aplicando-se os métodos propostos ao problema da detecção de novidades em dados reais e sintéticos e os aspectos precisão na classificação, custo computacional e taxa de redução de protótipos foram avaliados.

O método FBDOCC não faz nenhuma suposição sobre a distribuição de dados (*i.e.*, é um método não paramétrico) assim como não tenta criar descrições em algum formato pré-definido situadas no hiper espaço; como fazem métodos de kernel. Diferentemente de outras abordagens para redução de protótipos, o método FBDOCC trabalha de forma criacional e seletiva. Criacional pois reproduz protótipos artificiais durante a fase de treinamento e teste e seletiva pois armazena as instâncias de treinamento situadas na borda da descrição. As vantagens desse método são sua boa taxa de redução de protótipos, em relação a outros métodos que retêm protótipos na borda, assim como sua simplicidade de implementação. Como desvantagens podemos citar o custo computacional no pior caso e a complexidade na atribuição de parâmetros. O problema do custo computacional foi tratado através da heurística já citada. O problema da atribuição de parâmetros foi contornado com o uso do algoritmo PSO.

Existem inúmeras abordagens para a detecção de novidades que aplicam ou não a tarefa da redução de protótipos. Cada uma dessas técnicas incorpora particularidades que as tornam mais indicadas a problemas específicos em detrimento de seu rendimento em outros casos. A criação de uma técnica reconhecidamente melhor em todos os casos é, na prática, impossível. Dessa forma, a obtenção de métodos OCC capazes de se sobressair em média melhor que a maioria dos classificadores é de considerável importância visto que uma nova gama de problemas antes complexos pode ser melhor resolvida por essa nova abordagem.

Foram realizados vários experimentos em trinta e três bases de dados, reais e sintéticas,

apresentando diferentes características e os resultados foram comparados aos resultados obtidos por outros sete classificadores. Considerando-se o aspecto poder de generalização, o método FBDOCC apresentou o melhor resultado em vinte e dois dos trinta e três experimentos assim como a melhor média geral. Em relação à taxa de redução de protótipos, o método FBDOCC obteve o terceiro melhor resultado, ficando atrás apenas dos métodos CNNDD e SVDD. Considerando o custo computacional, o método FBDOCC obteve melhor resultado em trinta e um dos trinta e três experimentos ficando com a menor média de tempo de treinamento. Em contrapartida, a versão FBDOCC2 resultou em um alto custo computacional na fase de treinamento.

Foi ainda realizado um estudo de caso da aplicação de métodos OCC ao problema da detecção precoce de doenças cardíacas em crianças com sintomas de sopro cardíaco. Esse estudo contou com dados coletados de um renomado hospital local. A partir de informações não-invasivas[1] fornecidas pelo hospital foi montada uma base de dados desbalanceada devido ao número consideravelmente maior de ocorrências normais. A partir dessa base, foi realizado um processo de extração e geração de novas características e as características com maior poder discriminativo foram incorporadas à base de dados final. Como resultado, nosso método, FBDOCC e o método SVDD obtiveram resultados semelhantes obtendo AUCs de 0,833 e 0,832, respectivamente. Essas AUCs foram condicionadas a uma sensibilidade mínima de 81% (ver Tabela 5.8). Em relação à métrica MCC, o método FBDOCC superou o SVDD obtendo o valor de 0,491 contra 0,466.

6.1 Publicações geradas a partir desse trabalho e publicações relacionadas ao tópico em questão

Publicações em revistas:

- G. G. Cabral e A. L. I. Oliveira, One-class Classification based on searching for the problem features limits. Expert Systems with Applications, Vol. 41(16), pp. 7182–7199, 2014.

Publicações em conferências:

- G. G. Cabral e A. L. I. Oliveira, A novel one-class classification method based on feature analysis and prototype reduction, In: IEEE System, Man, and Cybernetics (SMC), pp. 983–988, 2011.
- G. G. Cabral e A. L. I. Oliveira, One-class classification through optimized feature boundaries detection and prototype reduction. In: International Conference on Artificial Neural Networks, Lausanne, pp. 693-700, 2012.

[1](*i.e.*, informações obtidas sem a necessidade de um exame complexo como o ecocardiograma, por exemplo.

- G. P. M. Farias, A. L. I. de Oliveira, G. G. Cabral, Extreme Learning Machines for Intrusion Detection Systems, In: 19th International Conference on Neural Information Processing, Doha, Qatar. V. 7666, pp. 535-543, 2012.

- T. Tavares, A. L. I. Oliveira, G. G. Cabral, S. Mattos e R. Grigorio. Preprocessing Unbalanced Data using Weighted Support Vector Machines for Prediction of Heart Disease in Children, In: International Joint Conference on Neural Networks, pp. 1-8, 2013

- G. G. Cabral e A. L. I. Oliveira, One-class Classification for Heart Disease Diagnosis, In: IEEE System, Man, and Cybernetics (SMC), 2014. (aceito para publicação)

6.2 Proposta de Trabalhos Futuros

Esta Seção sugere alguns temas para pesquisas futuras usando o método FBDOCC. Entre as perspectivas futuras, temos:

a) **Fusão de características do problema**: Os experimentos mostraram que considerar características separadamente se mostrou uma abordagem bastante eficiente, porém, a combinação dessas características pode gerar descrições mais flexíveis Dessa forma, estratégias que contemplem a fusão de características devem ser melhor investigadas.

b) **Investigação de novos espaços através do uso de funções de similaridade**: O método proposto baseia todo seu funcionamento no espaço de características original. A investigação da aplicação de novas funções de similaridades pode transformar esse espaço em um hiperespaço onde os objetos se apresentem de forma mais compacta facilitando assim a descrição dos dados.

c) **Treinamento Online**: Ambientes de redes de computadores com sistemas de detecção de intrusão, precisam se adaptar continuamente a novos tipos de ataques sem que o modelo de classificador faça um novo treinamento com todas as instâncias. Dessa forma é válida a investigação de abordagens capazes de identificar mudanças de conceito nos dados (*Concept Drift*) e incorpora-las no modelo de classificador, assim como descartar conceitos obsoletos.

d) **Incorporação de novas formas de treinamento**: O treinamento original do FBDOCC se mostrou relativamente custoso em alguns problemas. A incorporação da filosofia do *dividir para conquistar*, como aplicado em (GARCIA-PEDRAJAS; CASTILLO; ORTIZ-BOYER, 2010), pode ser uma ideia promissora no que diz respeito à diminuição do custo computacional.

e) **Inclusão de custo na classificação**: Vários problemas apresentam diferentes custos nos erros de classificação das diferentes classes. Por exemplo, classificar um paciente enfermo como saudável é muito mais grave que um paciente saudável como enfermo. Dessa forma, a inclusão de custos na classificação pode gerar um modelo mais adequado a cada tipo de problema.

f) **Inclusão de dados da classe novidade no treinamento**: Geralmente na OCC, a ocorrência de exemplos da classe novidade é mínima ou nenhuma. A inclusão desses exemplos quando disponíveis pode aprimorar o poder de generalização.

g) **Emprego de Otimização Multi-objetivo**: Esse trabalho utilizou o PSO canônico associado a uma função de *fitness* que considerava dois objetivos na tentativa de otimiza-los. O PSO Multi-Objetivo (MOPSO) (NEBRO et al., 2009) pode ser uma alternativa ao uso da função 4.2. Diferente do PSO canônico, o MOPSO retorna não uma única solução, mas um conjunto de soluções. A escolha de uma das soluções no conjunto retornado pelo MOPSO pode ser uma alternativa ao uso do PSO canônico.

Referências

ACHARYA, U. R. et al. Automated identification of normal and diabetes heart rate signals using nonlinear measures. **Computers in Biology and Medicine**, [s.l.], v.43, n.10, p.1523–1529, 2013.

AHA, D. W. Tolerating Noisy, Irrelevant and Novel Attributes in Instance-based Learning Algorithms. **International Journal of Man-Machine Studies**, [s.l.], v.36, p.267–287, 1992.

AINSWORTH, S.; WYLLIE, J. P.; WREN, C. Prevalence and clinical significance of cardiac murmurs in neonates. **Archives of disease in childhood. Fetal and neonatal edition**, [s.l.], v.80, n.1, p.43–45, 1999.

ANGIULLI, F. Condensed nearest neighbor data domain description. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], 2007.

ANTOS, A. et al. Data-Dependent Margin-Based Generalization Bounds for Classification. In: COLT: PROCEEDINGS OF THE WORKSHOP ON COMPUTATIONAL LEARNING THEORY, MORGAN KAUFMANN PUBLISHERS. **Anais...** [s.l.: s.n.], 2001. p.73–98.

AVCI, E. A new intelligent diagnosis system for the heart valve diseases by using genetic-SVM classifier. **Expert Systems with Applications**, [s.l.], v.36, n.7, p.10618–10626, 2009.

BACHE, K.; LICHMAN, M. **UCI Machine Learning Repository**. 2013.

BALDI, P. et al. Assessing the accuracy of prediction algorithms for classification: an overview. **Bioinformatics**, [s.l.], 2000.

BATCHELOR, B. G. **Pattern Recognition**: ideas in practice. [s.l.]: Plenum, 1978.

BENNETT, K. P.; DEMIRIZ, A.; MACLIN, R. Exploiting unlabeled data in ensemble methods. In: INTERNATIONAL CONFERENCE ON KNOWLEDGE DISCOVERY IN DATA-MINING (KDD). **Proceedings...** ACM, 2002. p.289–296.

BHATTACHARYYA, S. et al. Data mining for credit card fraud: A comparative study. **Decision Support Systems**, [s.l.], v.50, n.3, p.602–613, 2011.

BIAU, G. Analysis of a Random Forests Model. **Journal of Machine Learning Research**, [s.l.], v.13, n.1, p.1063–1095, 2012.

BODESHEIM, P. et al. Divergence-Based One-Class Classification Using Gaussian Processes. **British Machine Vision Conference**, [s.l.], 2012.

BROWN, G. **Diversity in Neural Network Ensembles**. 2004. Tese (Doutorado em Ciência da Computação) — School of Computer Science, University of Birmingham.

CABRAL, G. G.; OLIVEIRA, A. L. I. A comparative analysis of one-class structural risk minimization by support vector machines and nearest neighbor rule. In: IN: PROCEEDINGS OF IFIP INTERNATIONAL FEDERATION FOR INFORMATION PROCESSING. **Anais...** [s.l.: s.n.], 2008. v.276, p.245–254.

CABRAL, G. G.; OLIVEIRA, A. L. I.; CAHÚ, C. B. G. Combining Nearest Neighbor Data Description and Structural Risk Minimization for One-Class Classification. **Neural Computing & Applications**, [s.l.], v.18, 2009.

CAMPBELL, C.; BENNETT, K. **StatLib Data, Software and News From the Statistics Community**. Disponível em: http://lib.stat.cmu.edu/datasets.

CAO, L. J.; LEE, H. P.; CHONG, W. K. Modified support vector novelty detector using training data with outliers. **Pattern Recognition Letters**, [s.l.], v.24, n.14, p.2479–2487, 2003.

CHANG, C. L. Finding prototypes for nearest neighbor classifiers. **IEEE Transactions on Computers**, [s.l.], v.23, p.1179–1184, 1974.

CHEN, Y.; ZHOU, X.; HUANG, T. One-Class SVM for learning in Image Retrieval. **In Proceedings of International Conference on Image Processing**, [s.l.], v.1, p.34–37, 2001.

CHOI, Y. S. Least squares one-class support vector machine. **Pattern Recognition Letters**, [s.l.], v.30, n.13, p.1236–1240,, 2009.

DELEN, D.; WALKER, G.; KADAM, A. Predicting breast cancer survivability: a comparison of three data mining methods. **Artificial Intelligence in Medicine**, [s.l.], v.34, n.2, p.113–127, 2005.

DEMSAR, J. Statistical comparisons of classifiers over multiple data sets. **Journal of Machine Learning Research**, [s.l.], v.7, p.1–30, 2006.

DEZA, M. M.; DEZA, E. **Encyclopedia of Distances 2nd ed.** [s.l.]: Springer, 2013.

DIAMANTARAS, K. I.; KUNG, S. Y. **Principal Component Neural Networks**: theory and applications. New York: Wiley, 1996. (Wiley Series on Adaptive and Learning Systems for Signal Processing, Communications, and Control).

DOMINGOS, P. Rule Induction and Instance-Based Learning: A unified approach. In: INTERNATIONAL JOINT CONFERENCES ON ARTIFICIAL INTELLIGENCE. **Proceedings...** [s.l.: s.n.], 1995. p.1226–1232.

DOMINGOS, P. Unifying Instance-Based and Rule-Based Induction. **Machine Learning**, [s.l.], v.24, n.2, p.141–168, 1996.

DUDA, R. O.; HART, P. E.; STORK, D. G. **Pattern Classification**. [s.l.]: John Wiley & Sons, 2001.

DéSIR, C. et al. One Class Random Forests. **Pattern Recognition**, [s.l.], v.46, n.12, p.3490–3506, 2013.

EGAN, J. P. **Signal detection theory and ROC-analysis**. New York: Academic Press, 1975.

EMRAN, S. M.; YE, N. Robustness of Chi-square and Canberra distance metrics for computer intrusion detection. **Quality and Reliability Engineering International**, [s.l.], v.18, n.1, p.19–28, 2002.

ER, O.; YUMUSAK, N.; TEMURTAS, F. Chest diseases diagnosis using artificial neural networks. **Expert Systems with Applications**, [s.l.], v.37, n.12, p.7648–7655, 2010.

FARQUAD, M. A. H.; BOSE, I. Preprocessing unbalanced data using support vector machine. **Decision Support Systems**, [s.l.], v.53, n.1, p.226–233, 2012.

FAWCETT, T. An introduction to ROC analysis. **Pattern Recognition Letters**, [s.l.], v.27, n.8, p.861–874, 2006.

FISHER, R. A. The use of multiple measurements in taxonomic problems. **Annals of Eugenics**, [s.l.], v.7, n.2, p.179–188, 1936.

FRIEDMAN, J. H. A Recursive Partitioning Decision Rule for Nonparametric Classification. **IEEE Transactions on Computers**, [s.l.], v.26, p.404–408, 1977.

GARCIA-PEDRAJAS, N.; CASTILLO, J. A. R.; ORTIZ-BOYER, D. A cooperative coevolutionary algorithm for instance selection for instance-based learning. **Machine Learning**, [s.l.], v.78, n.3, p.381–420, 2010.

GARCIA, S. et al. Prototype Selection for Nearest Neighbor Classification: taxonomy and empirical study. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], v.34, p.417–435, 2012.

GATES, G. W. The Reduced Nearest Neighbor Rule. **IEEE Transactions on Information Theory**, [s.l.], v.18, n.5, p.431–433, 1972.

GROENEN, P. J. F.; KAYMAK, U.; ROSMALEN, J. V. Fuzzy Clustering with Minkowski Distance Functions. **Advances in Fuzzy Clustering and its Applications**, [s.l.], p.53–68, 2007.

HANSEN, M. S.; SJöSTRAND, K.; LARSEN, R. On the regularization path of the support vector domain description. **Pattern Recognition Letters**, [s.l.], n.13, 2010.

HART, P. E. The condensed nearest neighbor rule. **IEEE Transactions on Information Theory**, [s.l.], v.114, p.515–516, 1968.

HOARE, C. A. R. Quicksort. **The Computer Journal**, [s.l.], v.5, n.1, p.10–16, 1962.

HOFFMANN, H. Kernel PCA for novelty detection. **Pattern Recognition**, [s.l.], v.40, n.3, p.863–874, 2007.

HOQUE, M. M. et al. Importance of Cardiac Murmur in Diagnosing Congenital Heart Disease in Neonatal Period. **Bangladesh J. Child Health**, [s.l.], v.32, n.1, p.17–20, 2008.

HORN, W. AI in medicine on its way from knowledge-intensive to data-intensive systems. **Artificial Intelligence in Medicine**, [s.l.], v.23, n.1, p.5–12, 2001.

INTERNET Retailer – Portal to e-Commerce Intelligence, Online fraud costs e-retailers $3.5 billion in 2012. Disponível em: http://www.internetretailer.com/2013/03/28/online-fraud-costs-e-retailers-35-billion-2012. Acesso em: 16 de agosto de 2013.

KARACALI, B.; KRIM, H. Fast minimization of structural risk by nearest neighbor rule. **IEEE Transactions on Neural Networks**, [s.l.], v.14, p.127–137, 2003.

KARACALI, B.; RAMANATH, R.; SNYDER, W. E. A comparative analysis of structural risk minimization by support vector machines and nearest neighbor rule. **Pattern Recognition Letters**, [s.l.], v.25, n.1, p.63–71, 2004.

KEMMLER, M.; RODNER, E.; DENZLER, J. One-class classification with Gaussian processes. **Pattern Recognition**, [s.l.], v.46, n.12, p.3507–3518, 2013.

KENNEDY, J.; EBERHART, R. Particle swarm optimization. In: IEEE INTERNATIONAL CONFERENCE ON NEURAL NETWORKS (ICNN'95). **Anais...** [s.l.: s.n.], 1995. v.4, p.1942–1947.

KERHET, A. et al. A SVM-based approach to microwave breast cancer detection. **Engineering Applications of Artificial Intelligence**, [s.l.], v.19, n.7, p.807–818, 2006.

KHAN, S. S.; MADDEN, M. G. A Survey of Recent Trends in One Class Classification. In: ARTIFICIAL INTELLIGENCE AND COGNITIVE SCIENCE. **Anais...** [s.l.: s.n.], 2010. v.6206, p.188–197.

KIRKPATRICK, S.; GELATT, C. D.; VECCHI, M. P. Optimization by simulated annealing. **SCIENCE**, [s.l.], v.220, n.4598, p.671–680, 1983.

KOHONEN, T. The self-organizing map. In: **Anais...** [s.l.: s.n.], 1990. v.78, n.9, p.1464–1480.

KUANG, F.; XU, W.; ZHANG, S. A novel hybrid KPCA and SVM with GA model for intrusion detection. **Applied Soft Computing**, [s.l.], v.18, p.178–184, 2014.

LANCE, G. N.; WILLIAMS, W. T. Mixed-Data Classificatory Programs I - Agglomerative Systems. **Australian Computer Journal**, [s.l.], v.1, n.1, p.15–20, 1967.

LINHARES, A. et al. Entanglement of perception and reasoning in the combinatorial game of chess: differential errors of strategic reconstruction. **Cognitive Systems Research**, [s.l.], v.13, n.1, p.72–86, 2012.

LOWE, D. G. Similarity Metric Learning for a Variable-Kernel Classifier. **Neural Computation**, [s.l.], v.7, n.1, p.72–85, 1995.

LUO, B.; XIA, J. A novel intrusion detection system based on feature generation with visualization strategy. **Expert Systems with Applications**, [s.l.], v.41, n.9, p.4139–4147, 2014.

LUO, H.; WANG, Y.; CUI, J. A SVDD approach of fuzzy classification for analog circuit fault diagnosis with FWT as preprocessor Original. **Expert Systems with Applications**, [s.l.], n.8, 2011.

MANEVITZ, L. M.; YOUSEF, M. One-Class SVMs for Document Classification. **Journal of Machine Learning Research**, [s.l.], v.2, p.139–154, 2001.

MARKOU, M.; SINGH, S. Novelty detection: a review - part 1: statistical approaches. **Signal Processing**, [s.l.], v.83, n.12, p.2481–2497, 2003.

NEBRO, A. J. et al. SMPSO: a new pso-based metaheuristic for multi-objective optimization. In: IEEE SYMPOSIUM ON COMPUTATIONAL INTELLIGENCE IN MULTI-CRITERIA DECISION-MAKING. **Anais...** [s.l.: s.n.], 2009. p.66–73.

OH, J. et al. Generalized Mean for Feature Extraction in One-class Classification Problems. **Pattern Recognition**, [s.l.], v.46, n.12, p.3328–3340, 2013.

OLIVEIRA, A. L. I. **Neural Networks Forecasting and Classification-Based Techniques for Neural Networks Forecasting and Classification-Based Techniques for Novelty Detection in Time Series**. 2004. Tese (Doutorado em Ciência da Computação) — Federal University of Pernambuco.

OLIVEIRA, A. L. I.; COSTA, F. R. G.; F., C. O. S. Novelty detection with constructive probabilistic neural networks. **Neurocomputing**, [s.l.], v.71, n.4-6, p.1046–1053, 2008.

OLVERA-LÓPEZ, J. A.; CARRASCO-OCHOA, J. A.; MARTÍNEZ-TRINIDAD, J. F. A new fast prototype selection method based on clustering. **Pattern Analysis and Applications**, [s.l.], v.13, n.2, p.131–141, 2010.

RÄTSCH, G. Ensemble Learning Methods for Classification. , [s.l.], 1998. Tese de Doutorado (em alemão).

RÄTSCH, G.; ONODA, T.; MÜLLER, K. R. **Soft Margins for AdaBoost**. [s.l.]: Department of Computer Science, Royal Holloway, University of London, 1998.

REIN, A. J.; OMOKHODION, S. I.; NIR, A. Significance of a cardiac murmur as the sole clinical sign in the newborn. **Clinical Pediatrics**, [s.l.], v.39, p.511–520, 2000.

REPORT, T. world health. **Health systems financing**: the path to universal coverage. [s.l.]: World Health Organization, 2010.

RITTER, G. L. et al. An algorithm for a selective nearest neighbor decision rule. **IEEE Transactions on Information Theory**, [s.l.], v.21, p.665–669, 1975.

RODNER, E. et al. One-Class Classification for Anomaly Detection in Wire Ropes with Gaussian Processes in a Few Lines of Code. In: IAPR INTERNATIONAL CONFERENCE ON MACHINE VISION APPLICATIONS, 12. **Proceedings...** [s.l.: s.n.], 2011. p.219–222.

SAEZ, J. A.; LUENGO, J.; HERRERA, F. Predicting Noise Filtering Efficacy with Data Complexity Measures for Nearest Neighbor Classification. **Pattern Recognition**, [s.l.], v.46, n.1, p.355–364, 2013.

SALZBERG, S. A Nearest Hyperrectangle Learning Method. **Machine Learning**, [s.l.], v.6, p.251–276, 1991.

SÁNCHEZ, D. et al. Association rules applied to credit card fraud detection. **Expert Systems with Applications**, [s.l.], v.36, n.2, p.3630–3640, 2009.

SARTAKHTI, J. S.; ZANGOOEI, M. H.; MOZAFARI, K. Hepatitis disease diagnosis using a novel hybrid method based on support vector machine and simulated annealing (SVM-SA). **Computer Methods and Programs in Biomedicine**, [s.l.], v.108, n.2, p.570–579, 2012.

SAUNDERS, R.; GERO, S. The Importance Of Being Emergent. **Proceedings of Artificial Intelligence in Design**, [s.l.], 2001.

SCHÖLKOPF, B. et al. Estimating the Support of a High-Dimensional Distribution. **Neural Computation**, [s.l.], v.13, n.7, p.1443–1471, 2001.

SEO, K. K. An application of one-class support vector machines in content-based image retrieval. **Expert Systems with Applications**, [s.l.], v.33, n.2, p.491–498, 2007.

SWETS, J. A. Measuring the Accuracy of Diagnostic Systems. **Science**, [s.l.], v.240, p.1285 – 1293, 1988.

TAX, D. M. J. **One-Class Classification Concept-Learning In The Absence Of Counter-Examples**. 2001. Tese (Doutorado em Ciência da Computação) — Technische Universiteit Delft.

TAX, D. M. J. **DDtools, the Data Description Toolbox for Matlab**. version 2.1.1.

TAX, D. M. J.; DUIN, R. P. W. Outlier Detection Using Classifier Instability. **Lecture Notes in Computer Science**, [s.l.], v.1451, p.593–601, 1998.

THEODORIDIS, S.; KOUTROUMBAS, K. **Pattern Recognition, 3rd Edition**. [s.l.]: Academic Press, 2006.

TIAN, J.; GU, H. Anomaly detection combining one-class svms and particle swarm optimization algorithms. **Nonlinear Dynamics**, [s.l.], v.61, p.303–310, 2010.

TOMEK, I. An experiment with the edited nearest-neighbor rule. **IEEE Transactions on Systems, Man, and Cybernetics**, [s.l.], v.6, p.448–452, 1976.

TRIGUERO, I. et al. A Study of the Scaling up Capabilities of Stratified Prototype Generation. In: THIRD WORLD CONGRESS ON NATURE AND BIOLOGICALLY INSPIRED COMPUTING. **Anais...** [s.l.: s.n.], 2011. p.304–309.

TRIGUERO, I. et al. A Taxonomy and Experimental Study on Prototype Generation for Nearest Neighbor Classification. **IEEE Transactions on Systems, Man, and Cybernetics–Part C: Applications and Reviews**, [s.l.], v.42, n.1, p.86–100, 2012.

VAPNIK, V. N.; CHERVONENKIS, A. J. Ordered Risk Minimization (I and II). **Automation and Remote Control**, [s.l.], v.34, p.1226–1235 and 1403–1412, 1974.

WANG, S. et al. A modified support vector data description based novelty detection approach for machinery components. **Applied Soft Computing**, [s.l.], n.2, 2013.

WEBB, A. R. **Statistical Pattern Recognition**. [s.l.]: John Wiley and Sons, 2002.

WETTCHERECK, D.; DIETTERICH, T. G. An Experimental Comparison of the Nearest-Neighbor and Nearest-Hyperrectangle Algorithms. **Machine Learning**, [s.l.], v.19, p.5–27, 1995.

WILSON, D. L. Asymptotic properties of nearest neighbor rules using edited data. **IEEE Transactions on Systems, Man, and Cybernetics**, [s.l.], v.2, p.408–421, 1972.

WILSON, D. R.; MARTINEZ, T. R. Improved Heterogeneous Distance Functions. **J. Artif. Intell. Res. (JAIR)**, [s.l.], v.6, p.1–34, 1997.

WILSON, D. R.; MARTINEZ, T. R. Reduction Techniques for Instance-Based Learning Algorithms. **Machine Learning**, [s.l.], v.38, n.3, p.257, 2000.

WU, M. R.; YE, J. P. A small sphere and large margin approach for novelty detection using training data with outliers. **IEEE Transactions on Pattern Analysis and Machine Intelligence**, [s.l.], v.31, n.11, p.2088–2092, 2009.

Apêndice

A

Técnicas de Otimização

A.1 Particle Swarm Optimization - PSO

O método Particle Swarm Optimization (PSO) foi inicialmente proposto por Kennedy e Eberhart KENNEDY; EBERHART (1995). Essa técnica é geralmente empregada na busca por soluções ótimas em problemas não lineares. Inspirado no comportamento social de um grupo de pássaros, a principal idéia por trás do PSO é a construção de um conjunto (enxame) de partículas para a simulação de movimentos realizados pelos pássaros durante a busca por comida em uma região específica. Essa busca faz uso também da habilidade dos pássaros em se comunicar na busca por uma solução global. Cada partícula no PSO representa uma solução (modelo) em um espaço *n*-dimensional, onde *n* representa o número de parâmetros a serem otimizados.

Os principais conceitos do PSO são bastantes simples. Cada partícula no enxame possui as seguintes características:

- Uma posição, uma velocidade e um valor de adaptação obtido pela função de objetivo;
- Cada partícula conhece seus vizinhos, assim como suas melhores posições obtidas anteriormente e seus valores de adaptação;
- Cada partícula conhece a melhor solução global atual e seu valor de adaptação; e
- Cada partícula lembra sua melhor solução anterior.

Uma vizinhança é definida como um subconjunto do enxame cujas partículas são capazes de estabelecer comunicação. O enxame se move no espaço de busca através das atualizações das velocidades e posições de cada partícula KENNEDY; EBERHART (1995). A cada iteração do algoritmo, o comportamento de uma dada partícula é um compromisso entre quatro possíveis direções: (i) seguir seu próprio caminho; (ii) ir em direção à sua melhor posição anterior; (iii) ir em direção a melhor posição anterior de sua partícula vizinha; e (iv) ir em direção à melhor solução global. A Equação (A.1) mostra como esses componentes se combinam. O

termo wV_i^t representa o componente inércia, $\varphi_1 r_1(P_i - X_i^t)$ representa o componente cognitivo e $\varphi_2 r_2(P_g - X_i^t)$ é o componente social.

$$V_i^{t+1} = wV_i^t + \varphi_1 r_1(P_i - X_i^t) + \varphi_2 r_2(P_n - X_i^t) + \varphi_3 r_3(P_g - X_i^t) \tag{A.1}$$

Em (A.1): $r_1, r_2, r_3 \sim U(0,1)$ são parâmetros estocásticos pertencentes a uma distribuição uniforme; φ_1, φ_2 e φ_3 são constantes de aceleração; e w representa o peso da inércia. Adicionalmente, V_i^t é a velocidade da partícula i no tempo t, P_i é a melhor posição prévia da partícula i, P_n é a melhor posição na vizinhança, P_g é a melhor posição global e X_i^t é a posição atual da partícula i.

A nova posição da partícula, é dessa forma, dada pela Equação (A.2).

$$X_i^{t+1} = X_i + V_i^{t+1} \tag{A.2}$$

A Figura A.1 ilustra a operação do PSO. Inicialmente, um conjunto de partículas geradas aleatoriamente é criada e o valor de *fitness* (adaptação) de cada partícula é computado. A meta é encontrar os parâmetros que maximizam ou minimizam a função de *fitness*. Um critério de parada pode ser definido com a finalidade de finalizar a busca. Geralmente esse critério é o número de iterações. Se o critério não for satisfeito, os valores das velocidades e posições de cada partícula são atualizados, uma nova geração de partículas é criada e o processo retorna ao segundo passo na Figura A.1.

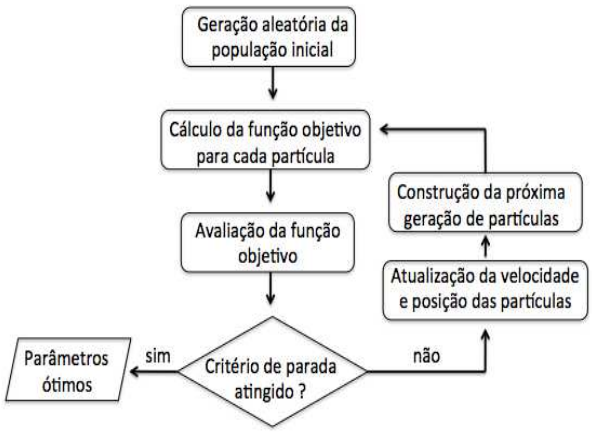

Figura A.1: Funcionamento do algoritmo Particle Swarm Optimization.

A.2 Simulated Annealing

O algoritmo *Simulated Annealing* KIRKPATRICK; GELATT; VECCHI (1983), diferente do PSO, não opera uma população de soluções de forma simultânea. Ao invés disso, apenas um agente reativo simples percorre a superfície de busca. Assim como o PSO, o algo-

ritmo busca por máximos, ou mínimos, globais. Quando a busca fica "presa" em um máximo local, o algoritmo não reinicia a busca aleatoriamente, ele retrocede com certa probabilidade para escapar desse máximo local. Esses retrocessos são chamados de passos indiretos. Apesar de aumentar o tempo de busca, essa estratégia consegue escapar dos máximos locais.

Nas iterações iniciais, o algoritmo não escolhe necessariamente o "melhor" passo, e sim um movimento aleatório: se a situação melhorar, esse movimento será sempre escolhido posteriormente, caso contrário, associa a esse movimento uma probabilidade de escolha menor do que 1. Essa probabilidade depende de dois parâmetros, e decresce exponencialmente com a piora causada pelo movimento. O valor $e^{\frac{\Delta E}{T}}$ representa a probabilidade da escolha por uma solução com pior avaliação, sendo ΔE (ver Equação (A.3)) a aptidão do próximo angente menos a aptidão do agente atual e T a temperatura. Quanto maior for a temperatura, mais próxima de 0 está a expressão $\frac{\Delta E}{T}$. Dessa forma, maior também é a probabilidade de se aceitar uma solução menos apta.

$$\Delta E = fitness(x^{t+1}) - fitness(x^t) \tag{A.3}$$

O algoritmo inicia com um valor alto de temperatura T. Esse valor é decrescido até que o critério de parada seja atingido. Para valores de T próximos a zero, a expressão $\frac{\Delta E}{T}$ cresce fazendo com que a expressão $e^{\frac{\Delta E}{T}}$ tende a zero e a probabilidade de aceitar um valor de "próximo" menos apto que o atual tende a zero. O algoritmo tende a aceitar apenas valores de próximo maiores que a partícula atual.

Para o presente trabalho, a temperatura inicial adotada teve valor 10 e a quantidade máxima de iterações foi de 5000.

B

Ferramentas Estatísticas

B.1 Teste de Wilcoxon

O teste Wilcoxon *signed-rank* é um método não-paramétrico para comparação de duas amostras pareadas. Supondo que tenha-se observações de duas populações A e B de tamanhos idênticos. Deseja-se testar a hipótese nula de que a amostra A é idêntica à B (*i.e.*, H0: A = B), por exemplo. A princípio os elementos de A e B são pareados e os valores numéricos da diferença entre cada par são computados, sendo possível três condições para essa diferença: positiva (+), negativa (-) ou igual (=). Dados que todas as diferenças entre os valores obtidos para cada par de dados foram calculadas, essas diferenças são ordenadas de acordo com seu valor absoluto, substituindo-se então os valores originais das diferenças pelo posto que ocupam na escala ordenada.

A tabela B.1 apresenta um exemplo hipotético desse cálculo. Nesse caso, a soma de postos negativos é igual a 10 enquanto que a soma de postos positivos é igual a 35.

Tabela B.1: Exemplo de populações hipotéticas para o cáculo do teste de Wilcoxon.

A	B	Diferença	Sinal	Posto
0,30	0,12	0,18	+	6
0,56	0,30	0,26	+	9
0,41	0,50	0,09	-	3
0,20	0,15	0,05	+	1
0,31	0,42	0,11	+	4
0,22	0,22	0,00	=	0
0,13	0,32	0,19	-	7
0,43	0,19	0,24	+	8
0,25	0,32	0,07	+	2
0,35	0,21	0,14	+	5

Dependendo da hipótese a ser testada, a estatística T do teste pode ser:

- H_0: A = B ou H_a: A $\neq$ B (bilateral). T é igual à menor dentre as somas de postos positivos ou negativos.

- H_0: A $\leq$ B ou H_a: A > B (unilateral). T é igual à soma de postos negativos.
- H_0: A $\geq$ B ou H_a: A < B (unilateral). T é igual à soma de postos positivos.

Dessa forma, com base no tamanho da amostra, caso a estatítica do teste seja menor que o valor para aquele tamanho de amostra na tabela de valores críticos de referência, a hipótese alternativa é aceita.

B.2 Distância de Kolmogorov-Smirnov

O teste de Kolmogorov-Smirnov (KS) tenta determinar se duas distribuições de dados diferem significativamente. Assim como o teste de Wilcoxon, o teste KS é também não paramétrico, ou seja, não faz nenhum tipo de suposição sobre os dados. A possibilidade de visualização gráfica da forma como os dados estão distribuídos é uma vantagem do teste.

Em um experimento típico, uma distribuição de dados da classe A é comparada a uma distribuição da classe B no intuito de verificar se a primeira distribuição produz resultados, ou *scores*, (quando aplicado um modelo $f(\vec{x})$ de classificador, por exemplo, a cada elemento $\vec{x}$ de cada distribuição) diferentes da segunda. Para essa situação, espera-se que as distribuições resultantes da aplicação do modelo $f(\vec{x})$ aos elementos de A e B sejam também diferentes para que haja a discriminação entre as classes A e B.

Dado o conjunto de *scores* referentes à classe A, esse conjunto é ordenado de acordo com o *score* de cada elemento e o gráfico apresenta a curva contendo o percentual de elementos da classe A à medida que o *score* evolui. A mesma curva é gerada para a classe B e a maior distância vertical entre as duas curvas representa o valor da distância KS. Quanto maior o valor de distância KS maior o poder domodelo na discriminação entre as classes.

A Figura B.1 ilustra um exemplo de curva KS. Nela, a curva azul representa a distribuição acumulada de elementos da classe positiva enquanto que a curva preta representa a distribuição acumulada de elementos da classe negativa. A curva vermelha representa a diferença entre as curvas azul e preta, o ponto em que a curva vermelha tem maior valor indica o melhor ponto de operação a ser usado pelo modelo de classificador, ou seja, o que melhor discrimina entre as classes..

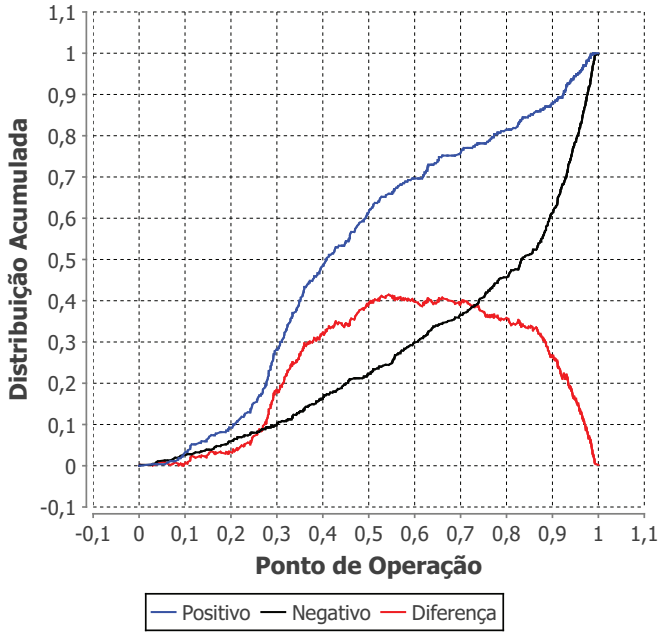

Figura B.1: Ilustração da classificação com exemplos de múltiplas classes e de uma única classe.