Criação de núcleos específicos para determinados problemas de classificação usando máquinas de suporte vetorial (SVM)

Alfredo Roberto Junior

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
OBTENÇÃO DO TÍTULO DE
MESTRE EM CIÊNCIAS

Área de Concentração: Ciência da Computação
Orientador: Prof. Dr. Marco Dimas Gubitoso

São Paulo, maio de 2007

Criação de núcleos específicos para determinados problemas de classificação usando máquinas de suporte vetorial (SVM)

Alfredo Roberto Junior

DISSERTAÇÃO APRESENTADA
AO
INSTITUTO DE MATEMÁTICA E ESTATÍSTICA
DA
UNIVERSIDADE DE SÃO PAULO
PARA
OBTENÇÃO DO TÍTULO DE
MESTRE EM CIÊNCIAS

Área de Concentração: Ciência da Computação
Orientador: Prof. Dr. Marco Dimas Gubitoso

São Paulo, maio de 2007

Criação de mapeamentos específicos para problemas de classificação usando máquinas de suporte vetorial (SVM)

Este exemplar corresponde à redação
final da dissertação defendida
e corrigida por
Alfredo Roberto Junior
e aprovada pela comissão julgadora.

São Paulo, maio abril de 2007

Banca examinadora:

- Prof.Dr. Marco Dimas Gubitoso (orientador) (IME-USP)
- Prof.Dr. André Ponce de Leon Ferreira de Carvalho (ICMSC-USP)
- Prof.Dr. Paulo José da Silva e Silva (IME-USP)

Aos meus pais Alfredo (in memoriam) e Lidia

AGRADECIMENTOS

Primeiramente gostaria de agradecer aos meus pais, que sempre lutaram pelo meu futuro fazendo um esforço sobrenatural para deixarem abertas todas as portas possíveis visando meu crescimento. Pessoas incríveis, exemplos que formaram o alicerce do meu caráter, devo tudo à eles, tudo.

Obrigado Juliana, meu amor, por sua paciência nas horas que deixamos de passar juntos para este trabalho ser completado, por sua compreensão e, principalmente, apoio quando por vezes fraquejei.

Ao professor Marco Dimas Gubitoso devo grande parte dos objetivos aqui alcançados, sempre serei grato pelo seu incentivo e sua ajuda na hora mais difícil vivida por mim no mestrado. Animado e perseverante, deu-me a confiança necesssária para atingir os objetivos propostos.

Obrigado ao professor Ronaldo Fumio e ao João pelo esforço conjunto nos seminários que apresentamos, foram fundamentais para o entendimento do projeto, ajudaram a aprimorar, e muito, minha formação matemática.

Agradecimentos também se fazem necessários ao professor Paulo Silva, por ter me incentivado a buscar o melhor, ao professor André Ponce, ao ter aceitado fazer parte da banca julgadora; ao professor Flávio Corrêa, que me deu a chance de cursar o mestrado; ao professor Ernesto Birgin, pelas palavras sensatas e amigas dirigidas a mim desde a graduação; aos professores Junior Barreira e Yoshiharu Kohayakawa, decisivos na fase final do projeto; a professora Zara Issa, uma grande amiga.

Muito obrigado a todos os funcionários da secretaria da Pós-

Graduação e Graduação, especialmente ao Pinho por sua incomensurável paciência e dedicação, devo à ele parte de tudo que alcancei aqui, tenha certeza que nunca esquecerei de sua ajuda.

Agradeço, desde já, aos amigos sempre presentes nos desafios de minha vida, relações iniciadas dentro e fora da faculdade. Sinto-me abençoado por ter tantas amizades sinceras, íntegras, pessoas que não mediriam e não mediram esforços para me ajudar nos momentos mais difíceis que passei até hoje, mais do que citar nomes, deixo aqui a certeza da presença de todos no meu coração.

Obrigado Associação Atlética Acadêmica da Matemática, você ajudou a construir um novo Alfredo, mais maduro e responsável.

Finalizo agradecendo a todos os professores e colegas que ajudaram na minha vida imeana.

Alfredo Roberto junior

RESUMO

Embora muitos conjuntos de dados sejam linearmente separáveis (um hiperplano consegue classificá-los), este não é um caso geral e classificadores lineares geralmente não apresentam bons resultados uma vez que muitos conjuntos de dados podem estar dispostos de maneira a não permitir que um simples hiperplano consiga definir dois grupos completamente desagregados.

Neste ponto um aspecto apresenta suma importância quanto a redefinir a distribuição espacial dos elementos de uma possível classificação, o *núcleo*. Este último centraliza sua ação no mapeamento do conjunto de dados inicial para um chamado *"espaço de característica"*, onde os dados antes não separáveis linearmente, podem sê-lo. Este fato é de importância ímpar, pois tendo em mãos um núcleo que melhor "mapeie" o espaço amostral dos dados a serem classificados, melhores e mais precisos serão os resultados obtidos e teremos então um classificador especializado para aquele determinado problema.

Este trabalho será voltado para a busca de um mapeamento que traga melhores resultados quanto a classificação de músicas e ruídos, juntamente com o uso de uma biblioteca que implemente a metodologia do SVM.

ABSTRACT

Despite of many data sets can be linearly separable (an hiperplan can classify them), this is not the usual case and linear classifers generaly do not present satisfactory results once many data sets can be distributed on the space of such way that no hiperplan can define two clusters completely separated.

At this point is presented an aspect of singular importance in the process of redefining the spacial distribution of the elements from a possible classification, the *kernel*. This last centralize its action in the mapping of inicial data set to the *"feature space"*, where the data before, not linearly separable, can be. This fact give us a interesting tool, because keep in hands a kernel that makes the best input space "mapping" of the data to be classified, better and more precise will be the results fot that determined problem.

This work will search a mapping that brings better classification results separating music and noise, together with the use of a library wich implements the SVM methodology.

Sumário

Capítulo 1

Introdução

Algoritmos de aprendizado vêm alcançando cada vez mais espaço em outras áreas de pesquisa, principalmente quando o ponto em questão é a classificação de dados, um assunto de grande importância dentro da biologia genética ([18], [19], [21]), reconhecimento de padrões ([20]). Uma relação intrínseca com processamento de informações que possam estar escondidas atrás de milhares de números, células, enfim, a classificação é usada como uma ferramenta para a busca por respostas concretas, resultados práticos para problemas do dia-a-dia.

É nesse contexto que o *SVM (Máquina de Suporte Vetorial)* se coloca como um algoritmo de extrema relevância na classificação de dados. Tal poder tem chamado a atenção de vários pesquisadores, e um deles, *Jean-Philippe Vert* ([1]) ponderou tal importância como a união entre quatro ítens, os quais resumem bem as características do SVM:

- Boa generalização: uma vez que o SVM é colocado junto a um conjunto de treinamento ele é capaz de aprender alguma regra potencialmente boa para classificar um novo objeto;
- Eficiência computacional: o algoritmo pode ser implemen-

tado de maneira eficiente em termos de velocidade e complexidade;

- Robustez: o SVM mostra-se robusto em relação a classificar objetos de grandes dimensões (imagens de genes, por exemplo);
- Origem: o SVM é fruto de uma intensa pesquisa teórica levantada nos anos 60 e 70 por Valpnik e Chervonenkis, a Teoria do Aprendizado Estatístico, teoria esta que nos levará a escolha do melhor classificador dentro de uma família.

O SVM, bem como vários algoritmos de classificação, procura representar no espaço os dados a serem classificados e tal representação é ideal quando o conjunto de dados é linearmente separável (um hiperplano consegue classificá-los). Este não é um caso geral e classificadores lineares, geralmente não apresentam bons resultados uma vez que muitos conjuntos de dados podem estar dispostos de maneira a não permitir que um simples hiperplano consiga definir dois grupos completamente desagregados.

Neste ponto um aspecto apresenta suma importância quanto a redefinir a distribuição espacial dos elementos de uma possível classificação, o *núcleo*. Este último centraliza sua ação no mapeamento do conjunto de dados inicial para um chamado *"espaço de característica"*, onde os dados antes não separáveis linearmente, podem sê-lo. Este fato é de importância ímpar, pois tendo em mãos um *núcleo* que melhor "mapeie" o espaço amostral dos dados a serem classificados, melhores e mais precisos serão os resultados obtidos e teremos então um classificador especializado para aquele determinado problema. Nesse prisma, a busca por um *mapeamento* que traga melhores resultados quanto a classificação dentro de um determinado problema torna-se de

grande importância para um uso consciente e bem estruturado do SVM, sendo este o principal estudo a ser feito neste trabalho, a busca por um *"mapeamento específico"* para a classificação entre músicas e ruídos.

Tal trabalho será realizado obtendo-se um conjunto de arquivos de áudio representativos de alguns estilos musicais ("rock", clássico, "pop") e de ruídos, juntamente com o uso de uma biblioteca que implemente a metodologia do SVM.

Seguiremos, então, nos demais capítulos, com explicações sobre os tópicos que fundamentam este estudo. No capítulo 2, introduziremos a base matemática formadora do alicerce do SVM (Teoria do Aprendizado Estatístico); já no capítulo 3, desenvolveremos a formulação do classificador linear e traremos definições sobre o uso do SVM e suas restrições; no capítulo 4, falaremos sobre classifcadores não lineares, espaço de característica e a definição de núcleo; no capítulo 5, caracterizaremos o ambiente de trabalho bem como uma explicação sobre a biblioteca, LIBSVM, que dará suporte no uso do SVM; no capítulo 6, trataremos do problema de classificação escolhido, no caso a classificação entre músicas e ruídos; no capítulo 7, as conclusões do estudo; e ainda traremos um apêndice com algumas definições pertinentes para um maior entendimento da teoria apresentada.

Capítulo 2

Teoria do Aprendizado Estatístico

2.1 Teoria

A Teoria do Aprendizado Estatístico constitui uma das bases da teoria do SVM, já que ela busca uma forma de encontrar o melhor classificador dentro de um conjunto, aquele capaz de ter o melhor desempenho (poucos erros no processo de classificação), em outras palavras, um classificador que tenha os melhores resultados quando colocado em contato com um conjunto de treinamento (dados que visam treinar o classificador para o tipo de problema ao qual ele será exposto) e com uma nova observação (dados representantes do problema em questão). Cabe ressaltar que apesar da denominação diferente, um conjunto de treinamento e uma observação representam os mesmos tipos de dados por tratarem do mesmo problema em estudo, e sua diferenciação dá-se, somente, pela finalidade de cada um deles (enquanto um deles visa treinar o classificador, o outro é usado para verificar o resultado do treinamento). Será usual também a utilização do termo *classe* para representar um determinado conjunto de

dados unidos em torno de determinadas semelhanças, no nosso caso as *classes* em questão sempre serão duas e representarão determinadas regiões espaciais.

Comecemos a teoria, então, representando por χ o espaço de objetos e x um elemento pertencente a tal espaço; por f chamaremos a regra de classificação (ou classificador) que mapeará cada $\vec{x} \in \chi$ para o espaço de classes $f(\vec{x})$ (cujos valores serão -1 ou 1). Chamaremos ainda de F o conjunto de todos os possíveis classificadores que um determinado algoritmo de aprendizado poderá escolher, sendo S o conjunto de treinamento que este último utilizará para aprender este particular classificador.

Procurando escolher um classificador $f \in F$ por meio de um conjunto de treinamento S, verificar a habilidade de f corretamente classificar os objetos do conjunto de treinamento passa a ser uma medida natural de qualidade. Segue daí que a denominação *risco empírico* de um classificador é dada pela taxa média de classificações errôneas que o mesmo pode realizar sobre S e será representado por $R_{emp}(f)$

$$R_{emp}(f) = \frac{1}{2l} \sum_{i=1}^{l} |y_i - f(x_i)|,$$

onde l representa o tamanho da amostra e y_i a classe de um determinado elemento da amostra. Chamaremos ainda $\frac{1}{2}|y_i - f(x_i)|$ de "perda" de um determinado $x \in \chi$.

Consideraremos um mau classificador aquele que apresentar $R_{emp}(f) = 1$ (comete erro para cada exemplo), enquanto um bom classificador apresentará $R_{emp}(f) = 0$ (classifica corretamente todos os objetos do conjunto de treinamento). Concluímos então, neste ponto, que ao $R_{emp}(f)$ caracterizar a capacidade de uma determinada regra f classificar corretamente

exemplos de um conjunto de treinamento S, os algoritmos de aprendizado procurarão sempre regras f que apresentem pequenos valores no seu $R_{emp}(f)$. Um modo conveniente para definir a generalização de desempenho da regra f é imaginar que os exemplos de treinamento sejam gerados, um a um, de maneira aleatória e independente, respeitando uma distribuição probabilística. Dentro desta suposição um objeto a ser classificado x e sua classe y serão variáveis aleatórias tendo dois possíveis valores (exemplificando -1 ou 1); uma observação (x, y) será regida por uma probabilidade P desconhecida numa primeira análise. Suporemos, então, que um conjunto de treinamento será formado por n variáveis $(x_1y_1, ..., x_Ny_N)$ gerados de acordo com P, e além disso, conseguiremos definir mais precisamente que a "generalização de desempenho" da regra f é a probabilidade de tal regra em cometer erros na amostra gerada aleatoriamente de acordo com a distribução P, em outras palavras:

$$R(f) = \int \frac{1}{2}|Y_i - f(x_i)|dP(x, y)$$

onde $R\ (f)$ é chamado de *risco* do classificador f e quantifica a habilidade da regra de classificação generalizar bem, ou seja, classificar corretamente uma nova observação ($R(f) = 0$, f não cometerá erros). Dada uma nova regra f e um conjunto de treinamento S nós só podemos observar o risco empírico $R_{emp}(f)$, intuitivamente, entretanto, é natural pensarmos que em muitos casos, se $R_{emp}(f)$ é pequeno, então $R(f)$ também será (intuição essa parcialmente verdadeira se utilizarmos um conjunto de treinamento cujo tamanho seja muito extenso, neste caso as chances de termos escolhido uma regra que melhor reflita uma determinada classe são muito maiores e $R_{emp}(f)$ converge para $R(f)$).

Porém, quando analisamos um algoritmo de aprendizado a situação se torna um pouco mais complexa. Como tal algoritmo tem que escolher uma regra $f \in F$ baseado na observação do conjunto de treinamento, é claro neste ponto a sua preferência por um f com o menor risco empírico sobre os dados de treinamento, ou seja, a regra f^e

$$R_{emp}(f^e) = \inf_{f \in F} R_{emp}(f)$$

sendo tal método chamado *minimização de risco empírico (ERM).* Ao mesmo tempo o real objetivo do algoritmo é encontrar a regra f que melhor generalize, isto é, a regra f^*

$$R(f^*) = \inf_{f \in F} R(f).$$

Os principais resultados da *Teoria do Aprendizado Estatístico* relacionam R_{emp} e R procurando responder a pergunta central com a qual acabamos de nos deparar: até que ponto a regra com o menor risco empírico é a aquela com o menor risco? Porém, antes de respondê-la, precisamos colocar aqui mais uma definição (uma primeira), a *dimensão de Vapnik-Chervonenkis.* **Chamada de $V(F)$, ou ainda, *dimensão VC* da classe F, ela se caracteriza por ser o máximo número de pontos com os quais podemos construir configurações no espaço onde algum classificador $f \in F$ consiga separá-los corretamente não importando os rótulos dados a eles (figura (2.1)).**

Escolhendo um η tal que $0 < \eta < 1$, tendo as "perdas" valores 0 ou 1, com probabilidade $1 - \eta$; Vapnik ([17]) chegou a seguinte relação ([7]):

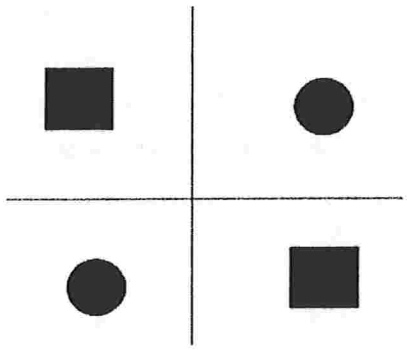

Figura 2.1: Quatro pontos $\Re^2$ que não podem ser separados corretamente por qualquer hiperplano.

$$R(f) \leq R_{emp}(f) + \sqrt{\frac{(V(F)(log(2N/V(F)) + 1) - log(\eta/4))}{N}} \tag{2.1}$$

sendo N o número de exemplos do conjunto de treinamento. Trazemos à tona, então, a possibilidade da regra f^e não se distanciar do melhor risco $R(f^*)$ possível, se duas condições forem satisfeitas:

- o número de observações N do conjunto de treinamento deve ser o maior possível;
- a dimensão VC do conjunto de possíveis regras F deve ser a menor possível.

Assim fica claro o papel central da dimensão VC na generalização de desempenho da regra f selecionada pela minimização do risco empírico: quanto menor a dimensão VC, melhor será a generalização da regra escolhida.

2.2 Dimensão de Vapnik-Chervonenkis

Dado o conjunto de classificadores $F = \{f \in F | \Re^n \rightarrow \{0,1\}\}$ e sendo $k_1, k_2, ..., k_l$ uma amostra de l exemplos onde cada $k_{i,i=1,...,l} \in \Re^n$ temos, então,

$$(k_1, k_2, ..., k_l) \in (\Re^n)^l$$

Se aplicarmos uma determinada regra f para cada uma dos exemplos k_i teremos uma configuração que refletirá uma certa classificação $(f(k_1), f(k_2), ..., f(k_l))$, em outras palavras, podemos ter várias configurações diferentes cada uma representando uma das possíveis classificações da amostra dada. Como o número de exemplos da amostra é l concluímos que no máximo teremos 2^l possíveis configurações de $(f(k_1), f(k_2), ..., f(k_l))$ (2^l subsequências binárias de comprimento l) , ou seja,

$$|(f(k_1), f(k_2), ..., f(k_l)) : f \in F| \leq 2^l$$

Definindo

$$B_F(l) = \max_{(k_1,k_2,...,k_l) \in (\Re^n)^l} |(f(k_1), f(k_2), ..., f(k_l)) : f \in F|$$

diremos que quando $B_F(l) = 2^l$ o conjunto de pontos $(k_1, k_2, ..., k_l)$ será "particionado" por F, estaremos definindo que para quaisquer das possíveis 2^l configurações de classificação sempre teremos uma regra $f \in F$ responsável por tal configuração. A **dimensão de Vapnik-Chervonenkis** será definida, então, como o máximo número l de exemplos que poderão ser *particionados* por F. É bom enfatizar que o fato da dimensão VC ser l significa que pelo menos um conjunto de l exemplos da amostra pode ser *particionado* e geralmente, não garante que *todos* os conjuntos de l exemplos possam ser *particionados* ([16], [11], [2]).

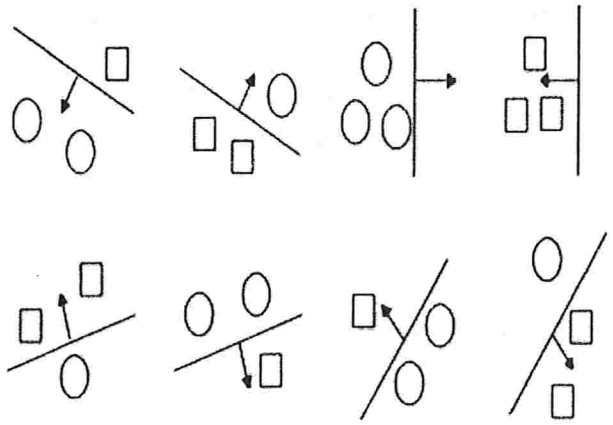

Figura 2.2: Todas as configurações possíveis com 3 exemplos para uma classe de classificadores lineares, ou seja, dimensão VC igual a 3. (Extraído de Burges, 1998 [7]).

2.3 Ligação entre o Aprendizado Estatístico e SVM

Na seção anterior vimos que o principal resultado da *Teoria do Aprendizado Estatístico* relaciona o risco da regra f^e selecionada pelo algoritmo de aprendizado (a partir do conjunto de regras F) usando o método de *minimização de risco empírico*, com o risco da melhor regra pertencente a F usando - $R(f) \leq R_{emp}(f) + \sqrt{\frac{(V(F)(log(2N/V(F))+1)-log(\eta/4))}{N}}$. Tal resultado provê a principal idéia do SVM: *a fim de obter boa generalização de desempenho a dimensão VC deve ser a menos possível.*

Há dois objetivos que se opõem na arquitetura de um algoritmo de aprendizado:

- Escolher um F tão grande quanto possível para pelo menos

uma regra dentro dele ter um risco pequeno;

- Escolher um F tão pequeno quanto possível para se ter certeza que a regra selecionada tenha risco tão pequeno quanto a melhor regra;

Visando resolver este aparente paradoxo, Vapnik propôs considerar a família de regras F a qual é a união crescente de famílias com dimensões VC crescentes, isto é,

$$F_0 \subset F_1 \subset ... \subset F_n \subset ... \subset F$$

com

$$V(F_1) \leq V(F_2) \leq ... \leq V(F_n) \leq ... \leq V(F)$$

Em cada família devemos ser capazes de computar a dimensão VC ou sermos capazes de achar um limite para a mesma. A *minimização do risco esrutural* consite em encontrar uma família F_i que minimiza o limite do risco R, em outras palavras, busca encontrar aquela família F_i que minimiza o risco empírico e que a soma desse último com sua respectiva dimensão VC seja mínima. Cabe ressaltar que o SVM é colocado como uma implementação da minimização do risco estrutural.

Capítulo 3

SVM

3.1 Considerações

Consideraremos neste capítulo um tipo especial de objetos, os quais denominaremos *vetores reais de dimensão finita.* Um vetor real m-dimensional $\vec{x}$ tem m coordenadas as quais representaremos como $\vec{x} = (x_1, x_2, ..., x_m)$, onde cada x_i é um número real ($x_i \in \Re$, para $i = 1, ..., m$) que representará um determinado atributo do objeto, em outras palavras, a característica citada no início deste estudo será formada por cada um deste m elementos do objeto em questão.

3.2 Classificadores lineares

Consideremos, neste caso, $m = 2$, teremos $\vec{x} = (x_1, x_2)$ e nosso ponto poderá ser representado por um ponto no plano. Neste espaço um linha pode ser caracterizada por uma equação linear da forma:

$$\vec{w} \cdot \vec{x} + b = 0 \tag{3.1}$$

sendo $\vec{w} \cdot \vec{x}$ o produto interno entre $\vec{w}$ e $\vec{x}$, isto é,

$$\vec{w} \cdot \vec{x} = w_1.x_1 + w_2.x_2 \tag{3.2}$$

Nós sabemos que dois vetores $\vec{x}_1$ e $\vec{x}_2$ são perpendiculares (ortogonais) se e somente se seu produto interno é igual a zero, ou seja, $\vec{x}_1 \cdot \vec{x}_2 = 0$. Tal propriedade pode ser usada para verificar que o vetor $\vec{w}$ é perpendicular a linha definida pela equação (3.1) da seguinte forma: seja $\vec{x}_1$ e $\vec{x}_2$ dois pontos que satisfazem (3.1), portanto teremos que $\vec{w} \cdot \vec{x}_1 + b = \vec{w} \cdot \vec{x}_2 + b = 0$; subtraindo uma equação da outra teremos que $\vec{w}.(\vec{x}_1 - \vec{x}_2) = 0$, mostrando assim que $\vec{w}$ é ortogonal ao vetor $\vec{x}_1 - \vec{x}_2$, sendo este último precisamente paralelo a reta definida na figura (3.1).

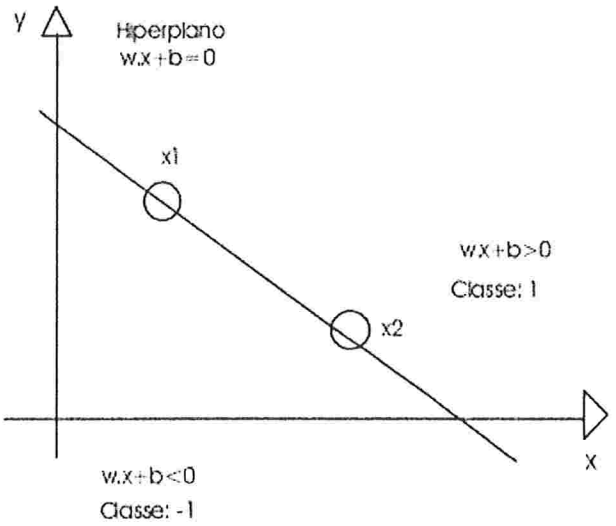

Figura 3.1: Classificador linear em $\Re^2$

Verificamos que a reta definida na figura (3.1) divide o espaço em duas áreas, ou *meio espaços*, um deles definido pelo conjunto de $\vec{x}$ tais que

$$\vec{w} \cdot \vec{x} + b > 0 \tag{3.3}$$

e outro definido pelo conjunto de $\vec{x}$ tais que

$$\vec{w} \cdot \vec{x} + b < 0 \tag{3.4}$$

Tal linha define um *classificador linear*. Ela pode classificar qualquer ponto $\vec{x} \in \Re^2$ dependendo da posição deste em relação a reta: $\vec{x}$ é classificado como +1 se $\vec{w} \cdot \vec{x} + b > 0$, -1 se $\vec{w} \cdot \vec{x} + b < 0$ e indeterminado se estiver posicionado sobre a reta ($\vec{w} \cdot \vec{x} + b = 0$).

A construção que acabamos de definir pode ser generalizada para $\Re^m$ mantendo as mesmas notações e portanto, para qualquer vetor $\vec{w} \in \Re^m$ e $b \in \Re$, o conjunto de pontos $\vec{x}$ que satisfaz

$$\vec{w} \cdot \vec{x} + b = 0$$

é chamado de hiperplano, reponsável por dividir o espaço em duas partes, um classificador linear que classificará qualquer vetor $\vec{x}$ dependendo do sinal de $\vec{w} \cdot \vec{x} + b$.

3.3 Perceptron

O *Perceptron* foi o primeiro algoritmo de aprendizado para classificação linear proposto em 1956 por Frank Rosenblatt. Ele é um algoritmo que em tempo real tenta achar um hiperplano separador para o conjunto de dados em questão, atualizando tanto o vetor $\vec{w}$ quanto o parâmetro b a medida que algum erro de classificação ocorra com os dados de teste. Ou seja, se para

um dado par $(\vec{w}, b)$ o hiperplano separador não conseguir classificar um determinado elemento do problema analisado um novo par $(\vec{w}, b)$ será gerado para cobrir tal classificação errônea na próxima iteração do algoritmo.

O perceptron garante convergir para uma solução caso realmente haja um hiperplano separador que corretamente classifique os dados do problema (dados linearmente separáveis). Para a descrição do algoritmo devemos levar em conta a existência de um conjunto de dados D onde cada elemento terá o formato (x_i, y_i) (x_i o i-ésimo elemento do conjunto e y_i sua classe).

Dado um conjunto de dados D e uma taxa de aprendizado $\eta \in \Re^+$
$w_0 \leftarrow 0;\ b \leftarrow 0;\ m \leftarrow 0;$
$R \leftarrow max_{1 \leq i \leq l} \|x_i\|$
repita até que nenhum erro de classificação ocorra
para $i = 1$ até l
se $y_i(\langle \vec{w_m} \cdot \vec{x_i} \rangle + b_m) \leq 0$ então
$\vec{w}_{m+1} = \vec{w}_m + \eta y_i \vec{x}_i$
$b_{m+1} = b_m + \eta y_i R^2$
$m \leftarrow m + 1$ (número de erros)
final se
final para
retorna $(\vec{w}_m, b_m)$

3.4 SVM linear para um conjunto de dados separáveis

Dado o conjunto de pontos $\vec{x_i} \in \Re^m$ com suas respectivas classes $y_i \in \{-1, +1\}$ para todo i = 1,..., l, onde l é o número de

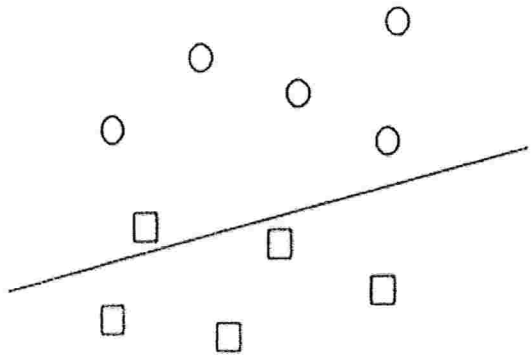

Figura 3.2: Hiperplano gerado pelo algoritmo perceptron classifica errôneamente elementos do conjunto de dados.

exemplos da amostra, conjunto de treinamento S será definido como

$$S = (\vec{x}_1, y_1), ..., (\vec{x}_l, y_l).$$

Nós dizemos que um conjunto de treinamento S é linearmente separável se existe pelo menos um classificador linear, definido pelo par $\vec{w}$ e b, o qual corretamente classifica todos os objetos em S, isto é,

$$\begin{cases} \vec{w} \cdot \vec{x} + b > 0 & \text{se} \quad y_i = +1 \\ \vec{w} \cdot \vec{x} + b < 0 & \text{se} \quad y_i = -1 \end{cases}$$

Porém, podemos ir mais longe e acrescentar que quando um conjunto de treinamento é lineramente separável normalmente existem muitos classificadores lineares capazes de classificar tal conjunto (figura (3.4)).

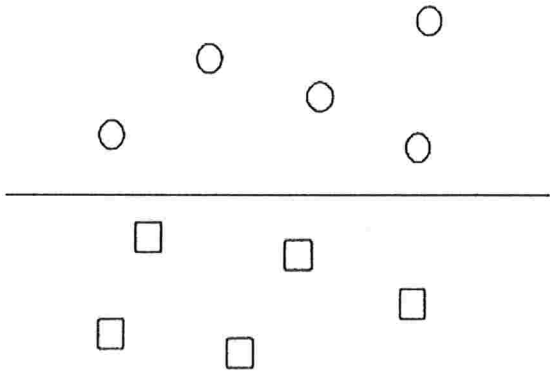

Figura 3.3: O algoritmo perceptron corrige o hiperplano gerado na figura 3.2.

Como fora visto anteriormente a aplicação da *Teoria do Aprendizado Estatístico* para a escolha de um bom classificador envolve dois pontos:

- Escolher um classificador com o menor risco empírico possível (buscando, é claro, aquele que melhor gneralize, que tenha o melhor risco, na classificação final);
- Escolher um classificador de uma família com a menor dimensão VC possível.

Para a primeira condição basta a escolha de um classificador que classifique corretamente todos os elementos do conjunto de treinamento (qualquer um da figura (3.4)); já para a segunda é necessário saber o significado da dimensão VC para uma família de classificadores. Cabe colocarmos aqui mais uma definição, a de *margem*, caracterizadda por ser a menor distância entre um ponto no conjunto de treinamento e o hiperplano separador; será

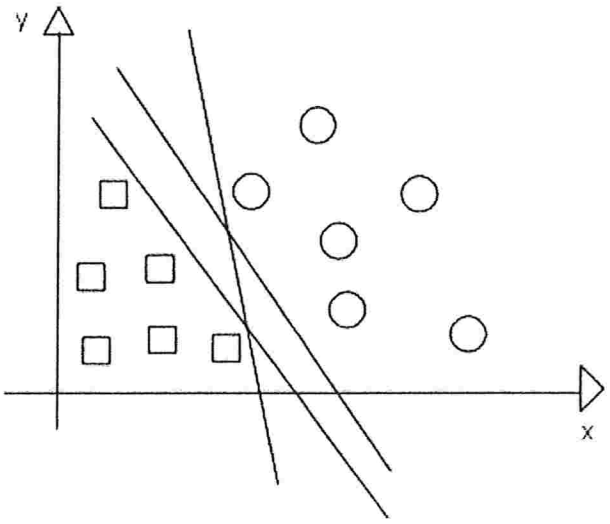

Figura 3.4: Diferentes classificadores lineares que corretamente classificam o conjunto de treinamento

representada aqui por γ (um exemplo pode ser visto na figura (3.5)).

3.4.1 O hiperplano ótimo

Considere o conjunto de pares $(\vec{w}, b)$ que satisfazem as seguintes desigualdades

$$\begin{cases} \vec{w} \cdot \vec{x_i} + b \geq +1 & \text{se} \quad y_i = +1 \\ \vec{w} \cdot \vec{x_i} + b \leq -1 & \text{se} \quad y_i = -1 \end{cases} \tag{3.5}$$

Cada par $(\vec{w}, b)$ define um classificador que classifica corretamente o conjunto de treinamento, sendo sua representação

$$f(x) = \begin{cases} +1 & \text{se} \quad \vec{w} \cdot \vec{x} + b > 0 \\ -1 & \text{se} \quad \vec{w} \cdot \vec{x} + b < 0. \end{cases}$$

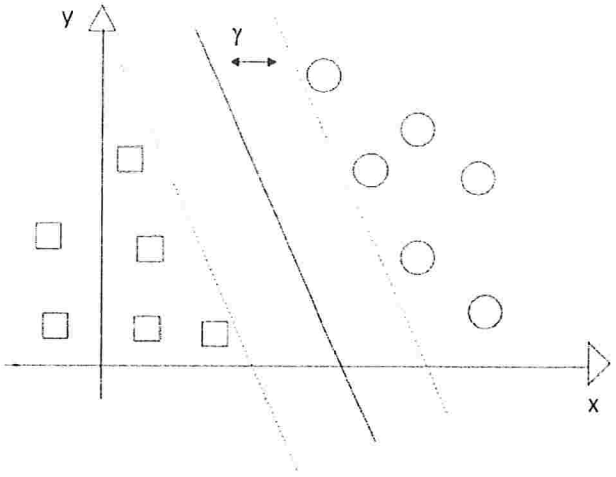

Figura 3.5: A margem λ de um classificador linear

Procurando encontrar um determinado par $(\vec{w}^*, b^*)$ que nos dá a maior margem, precisamos encontrar a margem criada pelo par $(\vec{w}, b)$ que satisfaz as desigualdades de (3.5) e as minimiza.

As desigualdades de (3.5) mostram que não há qualquer ponto entre os hiperplanos definidos por $\vec{w} \cdot \vec{x} + b = 0$ e $\vec{w} \cdot \vec{x} + b = 1$ (veja figura (3.6)); como resultado a margem γ é pelo menos igual a distância entre os hiperplanos definidos pelas equações $\vec{w} \cdot \vec{x} + b = 0$ e $\vec{w} \cdot \vec{x} + b = 1$, isto é, a distância entre $\vec{x}_1$ e $\vec{x}_2$ na figura (3.6). Veja como a computamos,

$$\begin{cases} \vec{w} \cdot \vec{x}_1 + b = 0 \\ \vec{w} \cdot \vec{x}_2 + b = 1 \end{cases}$$

Subtraindo a primeira da segunda obtemos

$$\vec{w}.(\vec{x}_2 - \vec{x}_1) = 1 \tag{3.6}$$

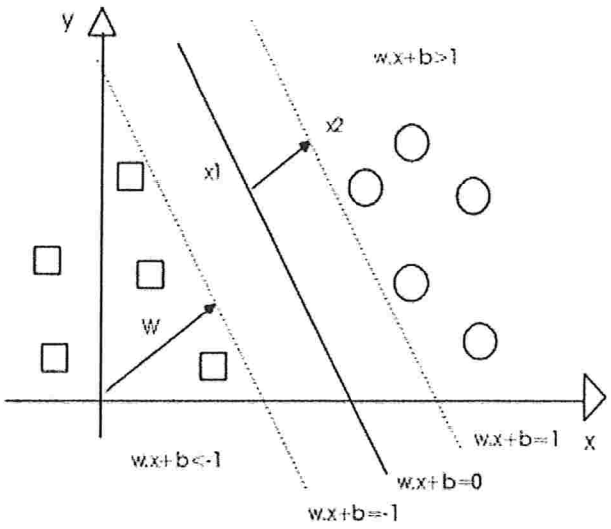

Figura 3.6: Hiperplano separador

Agora como $\vec{x}_2 - \vec{x}_1$ é ortogonal ao hiperplano separador, assim como $\vec{w}$, temos que $\vec{x}_2 - \vec{x_1}$ e $\vec{w}$ são paralelos e portanto seu produto interno satisfaz

$$|\vec{w}.(\vec{x_2} - \vec{x_1})| = \|\vec{w}\|.\|\vec{x_2} - \vec{x_1}\|,$$

onde $\|\vec{w}\|$ representa a norma do vetor $\vec{w}$. Usando esta equação com (3.6) obtemos

$$\|\vec{x_2} - \vec{x}_1\| = \frac{1}{\|\vec{w}\|}$$

Em outras palavras a distância entre o hiperplano definido pelas equações $\vec{w} \cdot \vec{x} + b = 0$ e $\vec{w} \cdot \vec{x} + b = 1$ é igual a $\frac{1}{\|\vec{w}\|}$. Já que a margem do respectivo classificador será sempre maior que esta distância teremos que quanto menor o valor de $\|\vec{w}\|$, maior será a margem (figura (3.6)).

A partir de agora poderemos caracterizar o problema de encontrar um hiperplano ótimo dizendo que tal hiperplano é definido pelo par $(\vec{w}, b)$ que satisfaz

$$\begin{cases} \vec{w} \cdot \vec{x}_i + b \geq +1 & \text{se} \quad y_i = +1 \\ \vec{w} \cdot \vec{x}_i + b \leq -1 & \text{se} \quad y_i = -1, \end{cases}$$

para o qual a norma $\|\vec{w}\|$ é mínima (problema de otimização com restrições).

3.4.2 A resolução do problema de otimização

O problema da seção anterior fica classicamente reformulado da seguinte forma (levando-se em conta que minimizar $\|w\|$ é o mesmo que minimizar $\|w\|^2$ e que as restrições são as mesmas):

Minimize

$$\|\vec{w}\|^2$$

sob as restrições

$$para \quad i = 1, ..., N, \quad y_i(\vec{w} \cdot \vec{x_i} + b) - 1 \geq 0.$$

Vamos inserir nesta abordagem a representação dual tentando expressar a importância de cada exemplo no conjunto de treinamento.

Começaremos, então, a resolução do problema de minimização sob restrições introduzindo o Lagrangiano $\mathcal{L}$, definida pelo vetor $\vec{w}$, o número real b e o vetor dual $\vec{\lambda} = (\vec{\lambda_1}, ..., \vec{\lambda_N})$ da seguinte forma

$$\mathcal{L}(\vec{w}, b, \vec{\lambda}) = \|\vec{w}\|^2 - \sum_{i=1}^{N} \lambda_i [y_i(\vec{w} \cdot \vec{x_i} + b) - 1].$$

Para formular o problema dual, primeiro precisamos calcular, para cada $\vec{\lambda}$ positivo, os valores $(\vec{w_{\vec{\lambda}}}, b_{\vec{\lambda}})$ os quais minimizam o Lagrangiano ($\vec{\lambda}$ fixo). E para isto podemos resolver a diferencial da função de *Lagrange* com respeito a $\vec{w}$ e b, tentando encontrar os valores para os quais eles serão iguais a zero:

$$\frac{\partial L}{\partial \vec{w}}(\vec{w}, b, \vec{\lambda}) = 2\vec{w} - \sum_{i=1}^{N} \lambda_i y_i \vec{x_i} = 0,$$
$$\frac{\partial L}{\partial b}(\vec{w}, b, \vec{\lambda}) = -\sum_{i=1}^{N} \lambda_i y_i = 0.$$

Da primeira equação temos que para qualquer $\vec{\lambda}$, o vetor $\vec{w_{\vec{\lambda}}}$ pode ser expresso como

$$\vec{w_{\vec{\lambda}}} = \frac{1}{2} \sum_{i=1}^{N} \lambda_i y_i \vec{x_i} \tag{3.7}$$

Da segunda, nós não podemos expressar diretamente $b_{\vec{\lambda}}$ em termos de $\vec{\lambda}$, porém nós temos a restrição sobre $\vec{\lambda}$

$$\sum_{i=1}^{N} \lambda_i y_i = 0. \tag{3.8}$$

Para qualquer $\vec{\lambda}$ onde esta restrição não é respeitada, o mínimo do *Lagrangiano* $L(\vec{w}, b, \vec{\lambda})$ é - ∞ ($b = +\infty$ se $\sum_{i=1}^{N} \lambda_i y_i > 0$, $b = -\infty$ se $\sum_{i=1}^{N} \lambda_i y_i < 0$).

Para qualquer $\vec{\lambda}$ que satisfaça (3.8) podemos usar (3.7) para calcular

$$\begin{aligned} inf_{\vec{w},b}L(\vec{w},b,\vec{\lambda}) &= L((\vec{w_{\vec{\lambda}}},b_{\vec{\lambda}},\vec{\lambda}) \\ &= \|\frac{1}{2}\sum_{i=1}^{N}\lambda_i y_i \vec{x_i}\|^2 - \sum_{i=1}^{N}\lambda_i[y_i(\vec{x_i}\sum_{j=1}^{N}\lambda_j y_j \vec{x_j} + b_\lambda) - 1] \\ &= \frac{1}{4}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \vec{x_i}\vec{x_j} - \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \vec{x_i}\vec{x_j} \\ &\quad -b\sum_{i=1}^{N}\lambda_i y_i + \sum_{i=1}^{N}\lambda_i \\ &= -\frac{1}{4}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \vec{x_i}\vec{x_j} + \sum_{i=1}^{N}\lambda_i. \end{aligned}$$

Para qualquer vetor $\vec{\lambda}$ chamaremos, então, $W(\vec{\lambda})$ o valor mínimo do *Lagrangiano* quando $\vec{\lambda}$ for fixado e $(\vec{w},b)$ puder variar sem restrição alguma, ou seja,

$$W(\vec{\lambda}) = inf_{\vec{w},b}L(\vec{w},b,\vec{\lambda}).$$

Partindo dos cálculos acima, temos que $W(\vec{\lambda})$ pode ser expresso, para qualquer vetor $\vec{\lambda}$, da seguinte forma:

$$W(\vec{\lambda}) = \begin{cases} -\frac{1}{4}\Sigma_{i=1}^{N}\Sigma_{j=1}^{N}\lambda_i\lambda_j y_i y_j \vec{x_i}\vec{x_j} + \Sigma_{i=1}^{N}\lambda_i & \text{se} \quad \Sigma_{i=1}^{N}\lambda_i y_i = 0 \\ -\infty & \text{caso contrário.} \end{cases}$$

Definimos, então, o *problema dual como o problema de encontrar o vetor positivo* $\vec{\lambda}$ *que maximiza a função* $W(\vec{\lambda})$.

Montado o problema dual e chamando sua solução de $\vec{\lambda}^*$, podemos encontrar o vetor $\vec{w}^*$ (o qual nos dá a direção do hiperplano ótimo) usando (3.7), agora, b^* por não ser obtido diretamente de $\vec{\lambda}^*$, será obtido através de $\vec{w}^*$ da seguinte forma:

$$b^* = -\frac{1}{2}[min_{i:y_i=+1}(\vec{w}^* \cdot \vec{x_i}) + max_{i:y_i=-1}(\vec{w}^* \cdot \vec{x_i})].$$

Teremos, dessa forma, que a aproximação dual para o problema de encontrar um hiperplano ótimo ficaria assim descrita:

Para qualquer conjunto de treinamento separável

$$S = (\vec{x_1}, y_1), ...(\vec{x_N}, y_N),$$

$\vec{\lambda}^* = (\lambda_1^*, ..., \lambda_N^*)$ *será o vetor solução do problema de otimização com restrições (problema dual) que maximiza*

$$W(\vec{\lambda}) = -\frac{1}{4}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \vec{x_i}\vec{x_j} + \sum_{i=1}^{N}\lambda_i,$$

sob as restrições

$$\begin{cases} \sum_{i=1}^{N}\lambda_i y_i = 0 \\ \vec{\lambda_i} > 0 \qquad \text{para} \quad i = 1, ..., N. \end{cases}$$

E tem o par $(\vec{w}^*, b^*)$ *definidor do hiperplano ótimo caracterizado por*

$$\begin{cases} \vec{w}^* = \sum_{i=1}^{N}\lambda_i^* y_i \vec{x_i} \\ b^* = -\frac{1}{2}[min_{i:y_i=+1}(\vec{w}^* \cdot \vec{x_i}) + max_{i:y_i=-1}(\vec{w}^* \cdot \vec{x_i})]. \end{cases}$$

3.5 SVM para um conjunto de dados não separáveis linearmente

Até agora tratamos de conjunto de dados linearmente separáveis, onde os classificadores lineares não apresentaram qualquer erro na classificação dos elementos do conjunto de treinamento, o que é difícil de ser observado na maioria dos exemplos reais. Mostraremos, nesta seção, como adaptar o SVM para abranger casos linearmente não separáveis e analisaremos os novos resultados desta adaptação.

Um primeiro passo é definirmos porque se daria uma classificação errônea por parte de um classificador linear? Temos pelo menos duas:

- conjunto de treinamento com ruídos;
- duas classes não podem ser separadas por um simples classificador linear, necessitando de formas mais sofisticadas (regras de separação não lineares).

Façamos agora uma análise sobre o par $(\vec{w}, b)$ definidor de um hiperplano separador ótimo. Tal par acaba por representar uma *região de folga* formada pelo conjunto de pontos entre os hiperplanos $\vec{w} \cdot \vec{x} + b = -1$ e $\vec{w} \cdot \vec{x} + b = +1$; tal região não cometerá erros se todos os pontos positivos do conjunto de treinamento satisfizerem $\vec{w} \cdot \vec{x} + b \geq +1$ e todos os pontos negativos satisfizerem $\vec{w} \cdot \vec{x} + b \leq -1$. A fim de fazermos uma medida da "quantidade de erros" que uma região de folga definida por um determinado conjunto de treinamento pode ter, introduziremos as chamadas *variáveis de folga*:

- se $y_i = +1$, temos

$$\zeta_i(\vec{w}, b) = \begin{cases} 0 & \text{se} \quad \vec{w} \cdot \vec{x} + b \geq +1 \\ 1 - (\vec{w} \cdot \vec{x} + b) & \text{se} \quad \vec{w} \cdot \vec{x} + b \leq +1; \end{cases}$$

- se $y_i = -1$, temos

$$\zeta_i(\vec{w}, b) = \begin{cases} 0 & \text{se} \quad \vec{w} \cdot \vec{x} + b \leq -1 \\ 1 + (\vec{w} \cdot \vec{x} + b) & \text{se} \quad \vec{w} \cdot \vec{x} + b \geq -1. \end{cases}$$

Ou seja, a *variável de folga* $\zeta_i(\vec{w}, b)$ valerá zero quando o elemento $(\vec{x_i}, y_i)$ do exemplo estiver localizado fora da *região* ou terá um valor positivo representando a distância entre x_i e a fronteira $\vec{w} \cdot \vec{x} + b = y_i$. Resumindo, baseando-se nos exemplos positivos e negativos, definiremos a *variável de folga* como:

Para i = 1,...,N temos

$$\zeta_i(\vec{w}, b) = \begin{cases} 0 & \text{se} \quad y_i(\vec{w} \cdot \vec{x} + b) \geq +1 \\ 1 - y_i(\vec{w} \cdot \vec{x} + b) & \text{se} \quad y_i(\vec{w} \cdot \vec{x} + b) \leq +1; \end{cases} \tag{3.9}$$

Fixado este novo conceito, partimos agora para a definição do que seria um bom classificador linear. E rapidamente apontamos duas propriedades:

- o número de classificações erradas sobre o conjunto de treinamento, bem como a importância destes erros deve ser a menor possível, em outras palavras, as *variáveis de folga* $\zeta_i(\vec{w}, b)$ precisam ser as menores possíveis para i = 1,..., N;
- a largura da *região de folga* deve ser a maior possível, garantindo uma boa generalização de desempenho, ou seja, $\|\vec{w}\|$ deve ser o menor possível.

Para combinar tais propriedades em um único valor, medindo desta forma a qualidade do classificador linear, temos

$$\varepsilon(\vec{w}, b) = \|\vec{w}\|^2 + C \sum_{i=1}^{N} \zeta_i(\vec{w}, b),$$

onde C é o parâmetro que fará a ponderação entre a largura da região de folga e o número de erros de classificação no conjunto de treinamento. Portanto, quanto menor for o valor de $\varepsilon(\vec{w}, b)$, melhor será a qualidade do classificador, menos erros ele cometerá, obtendo uma melhor desempenho.

3.5.1 Resolvendo o problema de otimização

Temos até este momento que um SVM linear para um conjunto de treinamento qualquer resolve o problema de encontrar o par $(\vec{w}, b)$ o qual minimiza

$$\varepsilon(\vec{w}, b) = \|\vec{w}\|^2 + C \sum_{i=1}^{N} \zeta_i(\vec{w}, b).$$

Podemos reescrever este último problema envolvendo restrições, basta que adicionemos a ele um vetor solução $\vec{\xi} = (\xi_1, ..., \xi_N)$ e, então, teremos:

Encontre o par $(\vec{w}, b)$ e o vetor $\vec{\xi} = (\xi_1, ..., \xi_N)$ os quais minimizam

$$\|\vec{w}\|^2 + C \sum_{i=1}^{N} \vec{\xi_i},$$

sob as restrições

$$\xi_i \geq \vec{\zeta_i}(\vec{w}, b) \quad \text{para i} = 1,...,N. \tag{3.10}$$

Se deixarmos $\vec{\xi}$ variar, podemos substituir 3.9 por

$$\begin{cases} \xi_i \geq 0, \\ \xi_i \geq 1 - y_i(\vec{w} \cdot \vec{x_i} + b) \quad \text{para} \quad i = 1, ..., N. \end{cases} \tag{3.11}$$

Pela definição das *variáveis de folga*, as restrições 3.9 e 3.10 são equivalentes e, dessa forma, montamos o problema de minimização da seguinte forma:

Encontre o par $(\vec{w}, b)$ *e o vetor* $\vec{\xi} = (\xi_1, ..., \xi_N)$ *os quais minimizam*

$$\|\vec{w}\|^2 + C \sum_{i=1}^{N} \xi_i,$$

sob as restrições

$$\begin{cases} \xi_i \geq 0, \\ \xi_i \geq 1 - y_i(\vec{w} \cdot \vec{x_i} + b) \quad \text{para} \quad i = 1, ..., N. \end{cases}$$

A partir deste ponto focalizamos nossos esforços na resolução do problema de otimização propriamente dito. E para tal fato precisamos introduzir as variáveis duais $\vec{\lambda} = (\lambda_1, ..., \lambda_N)$ para cada restrição $\xi_i - 1 + y_i(\vec{w} \cdot \vec{x} + b) \geq 0$, assim como as variáveis duais $\vec{\mu} = (\mu_1, ..., \mu_N)$ para cada restrição $\xi_i \geq 0$ e a partir daí montarmos o *Lagrangiano* como

$$\mathcal{L}(\vec{w}, b, \vec{\xi}, \vec{\lambda}, \vec{\mu}) = \|w\|^2 + C \sum_{i=1}^{N} \xi_i - \sum_{i=1}^{N} \lambda_i[\xi_i - 1 + y_i(\vec{w} \cdot \vec{x_i} + b)] - \sum_{i=1}^{N} \mu_i \xi_i.$$

Para cada vetor dual $(\vec{\lambda}, \vec{\mu})$ nós precisamos minimizar o *Lagrangiano* em função de $(\vec{w}, b, \vec{\xi})$ e para tanto calculamos as derivadas parciais

$$\frac{\partial L}{\partial \vec{w}}(\vec{w}, b, \vec{\xi}, \vec{\lambda}, \vec{\mu}) = 2\vec{w} - \sum_{i=1}^{N} y_i \lambda_i \vec{x_i} = 0,$$
$$\frac{\partial L}{\partial b}(\vec{w}, b, \vec{\xi}, \vec{\lambda}, \vec{\mu}) = \sum_{i=1}^{N} y_i \lambda_i = 0,$$
$$\frac{\partial L}{\partial b}(\vec{w}, b, \vec{\xi}, \vec{\lambda}, \vec{\mu}) = C - \lambda_i - \mu_i = 0, \quad \text{para cada i = 1,...,N.}$$

Usando estas condições podemos seguir os mesmos cálculos como no caso de um conjunto de dados separável, obtendo

$$inf_{\vec{w},b} L(\vec{w}, b, \vec{\xi}, \vec{\lambda}, \vec{\mu}) = -\frac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \lambda_i \lambda_j \vec{x_i} \vec{x_j} + \sum_{i=1}^{N} \lambda_i,$$

se $\sum_{i=1}^{N} y_i \lambda_i = 0$ e $\lambda_i + \mu_i = C$ para i = 1,...,N; -∞ caso contrário.

Agora tal função precisa ser maximizada em $\vec{\lambda} \geq 0$ e $\vec{\mu} \geq 0$, porém $\vec{\mu}$ não aparece na função para ser maximizada e nós precisamos, então, maximizá-la em função de $\vec{\lambda}$ e verificar se existe algum $\mu_i > 0$ para as quais todas as restrições são satisfeitas, ou seja, se e somente se $\lambda_i \leq C$ para i = 1,...,N (no caso em que $\mu_i \geq 0$ tal que $\mu_i + \lambda_i = C$ já estará satisfazendo).

Portanto, agora podemos descrever o SVM linear para qualquer conjunto de treinamento da seguinte forma:

Para qualquer conjunto de treinamento

$$S = (\vec{x_1}, y_1), ...(\vec{x_N}, y_N),$$

$\vec{\lambda}^* = (\lambda_1^*, ..., \lambda_N^*)$ *será o vetor solução do problema de otimização com restrições (problema dual) que maximiza*

$$W(\vec{\lambda}) = -\frac{1}{4}\sum_{i=1}^{N}\sum_{j=1}^{N}\lambda_i\lambda_j y_i y_j \vec{x_i}\vec{x_j} + \sum_{i=1}^{N}\lambda_i,$$

sob as restrições

$$\begin{cases} \Sigma_{i=1}^{N}\lambda_i y_i = 0 \\ 0 \leq \lambda_i \leq C \quad \text{para} \quad i = 1, ..., N. \end{cases}$$

E tem o par $(\vec{w}^*, b^*)$ *definidor do hiperplano ótimo caracterizado por*

$$\begin{cases} \vec{w^*} = \Sigma_{i=1}^{N}\lambda_i^* y_i \vec{x_i} \\ b^* = -\frac{1}{2}[min_{i:y_i=+1}(\vec{w}^* \cdot \vec{x_i}) + max_{i:y_i=-1}(\vec{w}^* \cdot \vec{x_i})], \end{cases}$$

o qual minimiza

$$\varepsilon(\vec{w}, b) = \|\vec{w}\|^2 + C\sum_{i=1}^{N}\zeta_i(\vec{w}, b),$$

3.6 Vetores de suporte

Até agora nosso objetivo foi encontrar um hiperplano ótimo para um conjunto de dados de treinamento separáveis, onde resolvemos tal questão como um problema de otimização sob restrições usando formulação dual.

Como visto na formulação dual nós encontramos o vetor dual $\vec{\lambda}^* = (\lambda_1^*, ..., \lambda_N^*)$ e o usamos para encontrar o hiperplano ótimo $(\vec{w^*}, b^*)$. Cada componente do vetor dual está associado a uma determinada restrição, sendo está representando dois possíveis tipos:

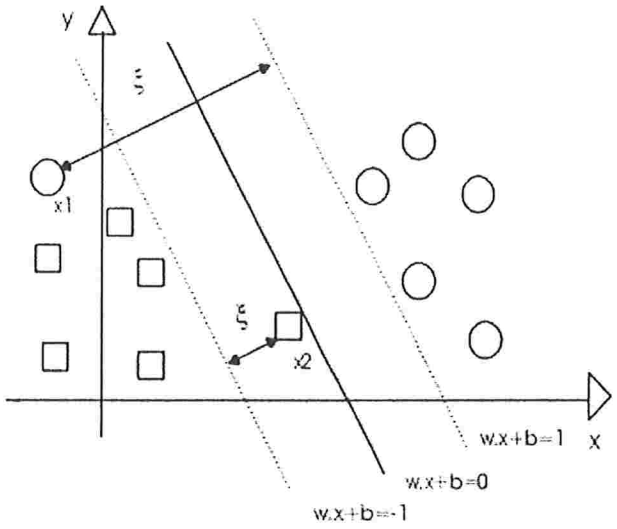

Figura 3.7: Hiperplano com pontos classificados errôneamente.

- restrições nas quais $\lambda_i > 0$, são ativas;
- restrições nas quais $\lambda_i = 0$, são inativas.

Em nosso caso a i-ésima restrição será

$$y_i(\vec{w} \cdot \vec{x_i} + b) - 1 \geq 0$$

e como vimos na figura (3.6) esta restrição nos diz que o ponto $\vec{x_i}$ está no lado "bom" do hiperplano separador e que sua distância deste último é maior que uma dada margem. Entretanto esta restrição é ativa se e somente se sua distância é igual a margem ($y_i(\vec{w}.\vec{x_i}$ + b) - 1 = 0).

Na figura (3.8) fica clara a definição de restrições ativas e inativas: as únicas restrições ativas correspondem aos pontos cuja distância ao hiperplano ótimo é exatamente igual a margem. A estes pontos com restrições ativas damos o nome de *vetores de suporte* (SV) (pontos em destaque na figura (3.8)).

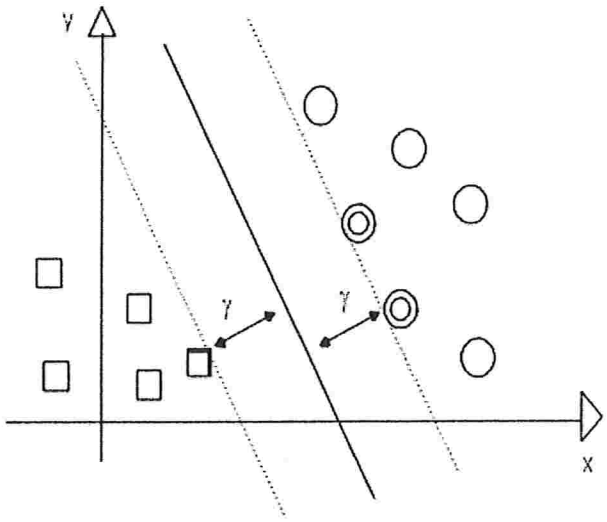

Figura 3.8: Hiperplano ótimo com os vetores de suporte em destaque.

Como visto anteriormente o hiperplano ótimo $(\vec{w}^*, b^*)$ é caracterizado pelo fato do vetor $\vec{w}^*$ ser a conbinação linear de exemplos de treinamento, ou seja,

$$\vec{w}^* = \sum_{i=1}^{N} \lambda_i^* y_i \vec{x}_i, \tag{3.12}$$

porém, descartando os termos nulos de (3.12) o vetor ótimo $\vec{w}^*$ pode ser reescrito como

$$\vec{w}^* = \sum_{x_i \in SV} \lambda_i^* y_i \vec{x}_i$$

Por $\vec{w}^*$ ser a combinação linear de vetores de suporte, teremos as seguintes consequências:

- como o número de vetores de suporte pode ser bem menor que o número de elementos do conjunto de treinamento, $\vec{w}^*$

pode ser a combinação de um pequeno número de vetores, propriedade esta chamada de *esparcialidade*;

- o hiperplano ótimo não é influenciado pelos exemplos de treinamento que não são SV (caso não degenerado).

3.7 Usando SVM para classificação

Através do problema de otimização com restrições chegamos ao par $(\vec{w}^*, b^*)$ definidor do hiperplano ótimo, sendo $\vec{w}^*$ escrito como a combinação linear de vetores de suporte

$$\vec{w}^* = \sum_{x_i \in SV} \lambda_i^* y_i \vec{x}_i$$

Verificamos também neste ponto que o objetivo do aprendizado do SVM é descobrir os parâmetros $\vec{\lambda}^*$ a partir dos quais conseguimos alcançar o hiperplano ótimo.

Uma vez terminado o aprendizado, partimos para a classificação dos elementos de uma nova observação - convém aqui lembrarmos que um classificador linear f definido pelo vetor $(\vec{w}^*, b^*)$ pode classificar qualquer vetor $\vec{x}$ de acordo com a regra

$$f(\vec{x}) = \begin{cases} +1 & \text{se} \quad \vec{w} \cdot \vec{x} + b > 0 \\ -1 & \text{se} \quad \vec{w} \cdot \vec{x} + b < 0 \end{cases}$$

- sendo tal procedimento dependente unicamente do sinal da expressão

$$\vec{w} \cdot \vec{x} + b$$

ou, quando nos referimos ao classificador SVM $(\vec{w}^*, b^*)$

$$\vec{w}^* \cdot \vec{x} + b = \sum_{x_i \in SV} \lambda_i^* y_i \vec{x_i} \cdot \vec{x} + b^*$$

Esta formulação nos mostra que uma vez encontrados λ_i^* e b^*, a classificação de uma nova observação $\vec{x}$ somente requer a computação do produto interno entre $\vec{x}$ e cada vetor de suporte, e suas duas consequências diretas são:

- o número de vetores de suporte é normalmente muito pequeno comparado ao tamanho de todo conjunto de treinamento, desta forma a classificação de uma nova observação é bem rápida dado que a proporção de operações segue a de vetores de suporte, independentemente do tamanho do conjunto de treinamento;
- para classificar uma nova amostra nós somente precisamos conhecer os valores de $\vec{x} \cdot \vec{x_i}$ para cada vetor de suporte $\vec{x_i}$ (propriedade muito usada quando falarmos de núcleos).

3.8 Problema: *Superespecialização*

Quando um conjunto de treinamento é apresentado a um algoritmo de aprendizado, este último tenta encontrar alguma regra que classifique a maioria dos elementos do conjunto corretamente. Algumas vezes o algoritmo pode encontrar uma regra complicada que perfeitamente classifique um conjunto de treinamento, porém ao ser apresentado a uma outra amostra ele tem um desempenho muito abaixo das expectativas, isto por estar intrínsecamente relacionado ao conjunto de treinamento. Dizemos que tal regra não "generaliza bem" e este fenômeno,

chamamos de *superespecialização.*
O trabalho de Vapnik e Chervonenkis buscou uma ligação entre a habilidade do algoritmo de aprendizado buscar uma regra que descreva bem o conjunto de treinamento e de tal regra generalizar bem.

Capítulo 4

Núcleo - SVM não linear

Até agora, trabalhando com um SVM linear tratamos da classificação de um determinado ponto baseados na posição deste em relação ao hiperplano separador. Porém, existem casos onde a disposição, no espaço, dos dados a serem classificados não terá um bom resultado utilizando-se um classificador linear, necessitando, desta forma, de classes de classificadores mais expressivas, que possam vislumbrar distribuições no espaço mais complexas, como, por exemplo, dois grupos de elementos dispostos de forma concêntrica (figura 4.1). É para estes casos que estudaremos neste capítulo uma maneira de generalizar o SVM.

Neste instante fica claro que o ponto a ser explorado é aumentar o poder de exepressividade computacional das máquinas de aprendizado lineares, entrando, a partir daí, o uso de uma poderosa ferramenta: o núcleo. Utilizando núcleos estaremos projetando os dados do espaço de entrada para um *espaço de característica* de dimensão maior, o que nos proporcionará melhores condições para analisar aspectos até então pouco evidentes no espaço de entrada. Convém destacar neste ponto que a *representação dual* detém suma importância neste processo uma vez que além de ser responsável por tal mapeamento se realizar

de forma implícita nos permitir, mesmo aumentando a dimensão do espaço de projeção, não aumentar o número de variáveis do problema.

Característica singular do estudo de núcleos é o fato dele ser totalmente *autocontido*, modular, podendo ser usado não só dentro do escopo de SVM, mas também em outras teorias de aprendizado. Tal estudo dará ainda condições para que dentro das especificidades da área na qual o SVM esteja sendo aplicado, o alvo passe simplesmente para a busca de um núcleo que melhor atenda o problema de classificação em questão. Traremos agora maiores detalhes de como todo este processo se dará e sobre a teoria envolvida.

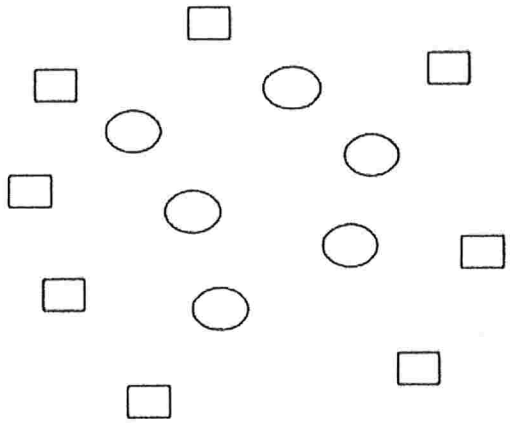

Figura 4.1: Conjunto de treinamanto não separável linearmente

4.1 Espaço de característica

Seja $S = (\vec{x_1}, y_1), ..., (\vec{x_N}, y_N)$ um conjunto de treinamento e suponha que nós possamos definir um conjunto de funções reais

$\phi_1, ..., \phi_M$ sobre o espaço de objetos a serem classificados. A estas funções daremos o nome de *características* e então todo objeto $\vec{x}$ poderá ser mapeado para o vetor $\vec{\phi}(\vec{x})$ com dimensão M da seguinte forma:

$$\vec{x} = (x_1, ..., x_m) \rightarrow \vec{\phi}(\vec{x}) = (\phi_1(\vec{x}), ..., \phi_M). \tag{4.1}$$

As características ϕ_i podem ser quaisquer funções, não tendo a necessidade de serem lineares, além disso, o número de características M pode ser maior ou menor que a dimensão m dos objetos $\vec{x}$.

Depois de mapear todos os pontos do conjunto de treinamento para o espaço de característica, teremos agora novos pontos

$$\vec{\phi}(S) = (\vec{\phi}(\vec{x}_1), y_1), ..., (\vec{\phi}(\vec{x}_N), y_N), \tag{4.2}$$

definidos no espaço de característica $\Re^M$. O fato que desperta o nosso interesse neste instante é este novo conjunto de treinamento $\vec{\phi}(S)$ ter a possibilidade de ser linearmente separável no espaço de característica mesmo o conjunto de treinamento S não o sendo no espaço original.

Durante este mapeamento o controle do número de *características* pode ser feito de diferentes formas, desde escolhendo as *características* que melhor representem informações essenciais do espaço de entrada, até verificando quais *características* são irrelevantes para o estudo em questão. Muitas vezes esta redução na dimensão do espaço de característica pode se dar simplesmente retirando as *características* onde os dados não apresentam variação

Exemplo: Digamos que um elemento no espaço de entrada tem os seguintes atributos

$$\vec{s} = (a_1, a_2, b_1, b_2, m_1) \tag{4.3}$$

e queremos reduzir a dimensão no espaço de característica seguindo o mapeamento $\phi : \Re^5 \rightarrow \Re^3$

$$\vec{s} = (a_1, a_2, b_1, b_2, m_1) \rightarrow \vec{\phi}(\vec{s}) = ((a_1 - a_2), (b_1 - b_2), m_1). \tag{4.4}$$

e, assim, mantendo somente a informação que consideramos essencial.

Em outros casos poderemos, ao invés de diminuir a dimensão do espaço de característica, aumentá-la mediante a necessidade de explicitar alguma informação não visível no espaço de entrada.

Exemplo: Considere um espaço de entrada de dimensão 2, onde nós sabemos que a informação essencial para o problema está contida em monômios de grau 2,

$$\vec{x} = (x_1, x_2) \rightarrow \vec{\phi}(\vec{x}) = (x_1^2, x_1 x_2, x_2^2).$$

Convém lembrarmos que geralmente as pesquisas se prendem à busca de redução de dimensão do espaço de característica, umas vez que quanto maior esta última mais custoso computacionalmente seu problema será, além de maior a probabilidade de *superespecialização* ([4]).

4.2 Mapeamento implícito e a definição de núcleo

Como discutimos anteriormente o ponto fundamental neste parte do estudo é conseguir um mapeamento não linear dos dados de entrada para o *espaço de característica*, onde lá, sim, um classificador linear poderá ser usado. Baseado neste fato a forma de nossos classificadores,

$$f(\vec{x}) = \vec{w} \cdot \vec{x} + b = 0$$

mudará para

$$f(\vec{x}) = \vec{w} \cdot \vec{\phi}(\vec{x}) + b = 0,$$

ou ainda

$$f(\vec{x}) = \sum_{i=1}^{N} \vec{w_i} \cdot \phi_i(\vec{x}) + b = 0,$$

onde $\vec{\phi}$ é um mapeamento não linear do espaço de entrada para o espaço de característica. Ou seja, para criar máquinas de aprendizado não lineares precisamos achar um mapeamento não linear $\vec{\phi}$ e depois classificá-los com um classificador linear no espaço de característica.

Exemplo: Suponha que tenhamos dados representados por pontos bidimensionais (m = 2) $\vec{x} = (x_1, x_2)$, e o seguinte mapeamento

$$\vec{x} = (x_1, x_2) \rightarrow \vec{\phi}(\vec{x}) = (x_1^2, x_1 x_2, x_2^2).$$

O hiperplano linear no espaço de característica é definido pelo vetor $\vec{w} = (w_1, w_2, w_3)$ e o número b da seguinte forma

$$\vec{w} \cdot \vec{\phi}(\vec{x}) + b = 0,$$

ou, de uma maneira mais explícita

$$w_1 x_1^2 + w_2 x_1 x_2 + w_3 x_2^2 + b = 0.$$

Mesmo esta equação sendo linear no espaço de característica, ela corresponde a uma equação polinomial no espaço de entrada $\Re^2$ (em outras palavras, um classificador linear no espaço de característica é de fato um classificador polinomial no espaço de entrada).

Vimos no capítulo anterior que as máquinas de aprendizado lineares podem ser expressas na representação dual, isto signifca que w pode ser escrito como a combinação linear dos pontos do conjunto de treinamento, em outras palavras, nosso classificador poderá ser escrito como o produto interno entre os pontos de teste e os pontos de treinamento:

$$f(\vec{x}) = \sum_{i=1}^{l} \alpha_i y_i \langle \phi_i(\vec{x}_i) \cdot \phi_i(\vec{x}) \rangle + b = 0. \tag{4.5}$$

onde l é o número de amostras de treinamento.

Computar o produto interno $\langle \phi_i(\vec{x}_i) \cdot \phi_i(\vec{x}) \rangle$ no espaço de característica nos dá a capacidade de realizar todos os passos para criar uma máquina de aprendizado de uma só vez, além de viabilizar a seguinte definição:

Definição - Um núcleo $K()$ é uma função tal que para quaisquer pontos $(\vec{x}, \vec{x}')$ no espaço de entrada temos

$$K(\vec{x}, \vec{x}') = \vec{\phi}(\vec{x}) \cdot \vec{\phi}(\vec{x}'),$$

onde $\vec{\phi}$ é um mapeamento para o espaço de característica.

A partir de agora poderemos escrever (4.5) como

$$f(\vec{x}) = \sum_{i=1}^{l} \alpha_i y_i K(\vec{x}_i, \vec{x}) + b = 0. \tag{4.6}$$

4.3 SVM linear no espaço de característica

Seja o espaço de característica

$$\vec{\phi}(S) = (\vec{\phi}(\vec{x}_1), y_1), ..., (\vec{\phi}(\vec{x}_N), y_N)$$

e seguindo a análise mostrada anteriormente, o SVM linear no espaço de característica descobre o vetor dual $\vec{\lambda} = \lambda_1, ..., \lambda_N$ maximizando a função

$$W(\vec{\lambda}) = -\frac{1}{4} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \lambda_i \lambda_j \vec{\phi}(\vec{x}_i) \cdot \vec{\phi}(\vec{x}_j) + \sum_{i=1}^{N} \lambda_i$$

sob as restrições

$$\begin{cases} \Sigma_{i=1}^{N} y_i \vec{\lambda}_i = 0, \\ 0 \leq \vec{\lambda}_i \leq C \quad \text{para} \quad i = 1, ..., N. \end{cases}$$

A partir de $\vec{\lambda}^*$ podemos achar o valor de b pela equação

$$b^* = -\frac{1}{2}\{\ min_{i:y_i=+1}(\vec{w}^*.\vec{x}_i) + max_{i:y_i=-1}(\vec{w}^*.\vec{x}_i)\ \}$$

e o classificador resultante pode classificar qualquer novo exemplo $\vec{x}$ dependendo do sinal da função

$$f(x) = \sum_{i=1}^{N} y_i \vec{\lambda}_i^* \vec{\phi}(\vec{x}_i) \cdot \vec{\phi}(\vec{x}) + b^*.$$

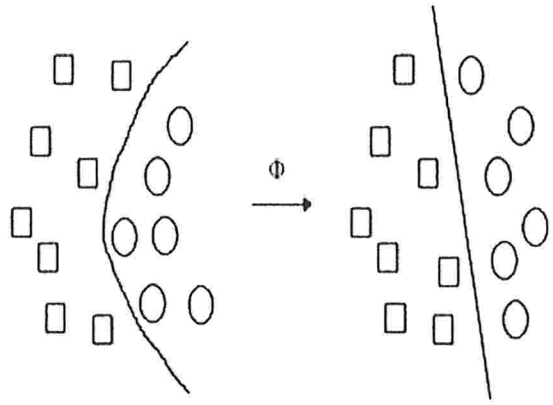

Figura 4.2: Classificador linear no espaço de característica

Dada a definição de núcleo podemos agora reformular o problema do início da seção substituindo o produto interno $\phi(\vec{x}) \cdot \phi(\vec{x}')$ por $k(\vec{x}, \vec{x}')$:

Proposição - Para qualquer conjunto de treinamento

$$S = (\vec{x}_1, y_1), ..., (\vec{x}_N, y_N)$$

e qualquer núcleo K () correspondente ao produto interno no espaço de característica, $\vec{\lambda}^* = (\lambda_1^*, ..., \lambda_N^*)$ será o vetor solução do problema de otimização com restrições, onde maximizaremos

$$W(\vec{\lambda}) = -\frac{1}{4}\sum_{i=1}^{N}\sum_{j=1}^{N} y_i y_j \lambda_i \lambda_j K(\vec{x}_i, \vec{x}_j) + \sum_{i=1}^{N} \lambda_i$$

sob as restrições

$$\begin{cases} \Sigma_{i=1}^{N} y_i \lambda_i = 0, \\ 0 \leq \lambda_i \leq C \qquad \text{para} \quad i = 1, ..., N \end{cases}$$

sendo a função de decisão f para qualquer objeto $\vec{x}$ definida por

$$f(x) = \sum_{i=1}^{N} y_i \lambda_i^* K(\vec{x}_i, \vec{x}) + b^*.$$

Seguiremos mais a frente com a teoria por trás da construção de núcleos bem como suas propriedades.

4.4 Matriz de Gram (Matriz de Núcleo)

Dado um conjunto de vetores $\vec{x_1}, ..., \vec{x_l}$ chamaremos de *matriz de Gram* a matriz G, l x l, cujas entradas são $G_{ij} = \langle \vec{x}_i, \vec{x}_j \rangle$. Se utilizarmos um núcleo K para calcular os produtos internos no espaço de característica com a *função característica* $\vec{\phi}$, a matriz de Gram terá as seguintes entradas

$$G_{ij} = \langle \vec{\phi}(\vec{x}_i).\vec{\phi}(\vec{x}_j) \rangle = K(\vec{x}_i, \vec{x}_j).$$

Neste caso a matriz de Gram será chamada de *matriz de núcleo.*

Proposição 1 - Matrizes de núcleo e matrizes de Gram são positivas semi-definidas.

Prova: Considere, para $i, j = 1, ..., l$, o caso geral de uma matriz de núcleo ser

$$G_{ij} = K(\vec{x}_i, \vec{x}_j) = \langle \vec{\phi}(\vec{x}_i).\vec{\phi}(\vec{x}_j) \rangle.$$

para qualquer vetor v temos

$$\begin{aligned} \vec{v}' G \vec{v} &= \sum_{i,j=1}^{l} \vec{v}_i \vec{v}_j G_{i,j} \\ &= \sum_{i,j=1}^{l} \vec{v}_i \vec{v}_j \langle \vec{\phi}(x_i).\vec{\phi}(x_j) \rangle \\ &= \langle \sum_{i=1}^{l} \vec{v}_i \vec{\phi}(x_i) \cdot \sum_{j=1}^{l} \vec{v}_j \vec{\phi}(x_j) \rangle \\ &= \| \sum_{i=1}^{l} \vec{v}_i \vec{\phi}(x_i) \|^2 \geq 0 \end{aligned}$$

◇

4.5 Caracterização de núcleos

Começaremos colocando uma primeira definição muito importante no decorrer desta seção já que ela nos ajudará a caracte-

rizar núcleos.

Definição - Uma função K: X x $X \rightarrow \Re$ satisfaz a propriedade de ser finitamente positiva semi-definida se ela for uma função simétrica para a qual as matrizes formadas sob qualquer subconjunto finito de X são positivas semi-definidas.

Seguiremos, agora, mostrando um teorema para caracterização de núcleos bem como todo o desenvolvimento dentro desta área para sua comprovação. Uma vez caracaterizado o núcleo ainda apresentaremos o *teorema de Mercer*, usado para construir um espaço de característica a partir de um núcleo válido.

Teorema 1 - Uma função K: X x $X \rightarrow \Re$, seja ela contínua ou com domínio contável, pode ser decomposta

$$K(\vec{x}, \vec{y}) = \langle \vec{\phi}(\vec{x}) \cdot \vec{\phi}(\vec{y}) \rangle$$

em funções característica dentro de um espaço de Hilbert H aplicado sobre seus argumentos e seguido pelo cálculo de seu produto interno dentro de H se e somente se ela satisfaz a propriedade de ser finitamente positiva semi-definida.

O espaço de característica F_K criado a partir de núcleo K é conhecido como *Espaço de Hilbert de um Núcleo de Reprodução (RKHS)*. Dado que no teorema acima conseguimos de um núcleo válido criar um espaço de característica, o próximo teorema que apresentaremos nos dará não um resultado novo, mas uma diferente maneira e chegar ao mesmo, uma vez que definirá um espaço de característica a partir de um explícito vetor de características e não usando um espaço de funções como na construção

do RKHS.

Teorema de Mercer - Seja X um subconjunto compacto de $\Re^n$. Suponha que K é uma função simétrica contínua tal que o operador integral $T_k : L_2(X) \rightarrow L_2(X)$

$$(T_k f)(\cdot) = \int_X K(\cdot, x) f(x) dx,$$

é positivo, isto é

$$\int_{X \times X} K(x, y) f(x) f(y) dx dy \geq 0,$$

para todo $f \in L_2(X)$. Então nós podemos expandir $K(x, y)$ em uma série uniformemente convergente em termos das funções ϕ_j, satisfazendo $\langle \phi_j \cdot \phi_i \rangle = \delta_{ij}$

$$K(x, y) = \sum_{i=1}^{\infty} \phi_i(x) \phi_i(y).$$

Além disso, a série $\Sigma_{i=1}^{\infty} \|\phi_i\|_{L_2}^2$ é convergente ([3]).

4.6 Construção de núcleos

Os passos de caracterização de núcleos bem como de matrizes de núcleos vistos até então nos deram ferramentas não só para decidir sobre um núcleo válido, mas também para justificar uma série de regras usadas na manipulação e combinação de núcleos mais simples a fim de obter outros mais complexos e úteis. Tais regras têm como principal característica o fato de preservarem

no núcleo final a propriedade de ser finitamente positivo semi-definido, além de garantir que continuaremos a preservar os dados no espaço de característica.

Segue então uma proposição que explicíta as operações realizadas sobre núcleos:

Proposição - Sejam K_1 e K_2 núcleos sobre $X \times X$, $X \subseteq \Re^n$, $a \in \Re^+$, $f(\cdot)$ uma função de valor real sobre X, $\vec{\phi} : Z \rightarrow \Re^N$ com K_3 um núcleo sobre $\Re^N \times \Re^N$, e B uma matriz, $n \times n$, simétrica, positiva semi-definida. Temos que as seguintes funções são núcleos ([4]):

- (1) - $K(x,y) = K_1(x,y) + K_2(x,y)$;
- (2) - $K(x,y) = aK_1(x,y)$;
- (3) - $K(x,y) = K_1(x,y)K_2(x,y)$;
- (4) - $K(x,y) = f(x)f(y)$;
- (5) - $K(x,y) = K_3(\vec{\phi}(x), \vec{\phi}(y))$;
- (6) - $K(x,y) = x^{'}By$;

Prova: Seja S um conjunto finito de pontos $x_1, ..., x_l$, seja M_{K_1} e M_{K_2} as matrizes de núcleo correspondentes a restrição dos núcleos K_1 e K_2 sobre estes pontos. Considerando qualquer vetor $\vec{\alpha} \in \Re^l$, diremos que a matriz M será positiva semi-definida se $\vec{\alpha}^{'} M \vec{\alpha} \geq 0$.

- (1) - Temos que

$$\vec{\alpha}^{'}(M_{K_1} + M_{K_2})\vec{\alpha} = \vec{\alpha}^{'} M_{K_1} \vec{\alpha} + \vec{\alpha}^{'} M_{K_2} \vec{\alpha} \geq 0,$$

e desta forma, $M_{K_1}+M_{K_2}$ é positiva semi-definida e K_1+K_2 é um núcleo;

- (2) - De maneira similar ao ítem anterior

$$\vec{\alpha}' aM_{K_1}\vec{\alpha} = a\vec{\alpha}' M_{K_1}\vec{\alpha} \geq 0,$$

verificando que aK_1 é um núcleo;

- (3) - Seja

$$M = M_{K_1} \otimes M_{K_2}$$

o produto (tensor) das matrizes M_{K_1} e M_{K_2} obtido substituindo cada entrada de M_{K_1} por M_{K_2} multiplicado por aquela entrada. O produto (tensor) de duas matrizes positivas semi-definidas é também positivo semi-definido já que os autovalores são todos pares dos produtos dos autovalores dos dois componentes. A matriz correspondente a função K_1K_2 é conhecido como *produto de Schur* H de K_1 e K_2 cujas entradas são os produtos das correspondentes entradas nos dois componentes. A matriz H é a principal submatriz de M por um conjunto de colunas e o mesmo conjunto de linhas. Daí, para qualquer $\vec{\alpha} \in \Re^l$ existe um correspondente $\vec{\alpha}_1 \in \Re^{l^2}$ tal que $\vec{\alpha}' H\vec{\alpha} = \vec{\alpha}_1' M\vec{\alpha}_1 \geq 0$ e então H é positiva semi-definida com o desejávamos demonstrar;

- (4) - Considere a função de característica de dimensão 1

$$\vec{\phi} : x \rightarrow f(x) \in \Re,$$

então seu correspondente núcleo será $K(x, y)$;

- (5) - Desde que K_3 seja um núcleo, a matriz obtida restringindo K_3 sobre os pontos $\vec{\phi}(x_1), ..., \vec{\phi}(x_l)$ é positiva semi-definida como desejávamos;;

- (6) - Considere a diagonalização de $B = V'\Lambda V$ por uma matriz ortogonal V, onde Λ é a matriz diagonal contendo os autovalores não negativos. Seja $\sqrt{\Lambda}$ a matriz diagonal com as raizes quadradas dos autovalores e seja $A = \sqrt{\Lambda}V$, Portanto, temos

$$K(x, y) = x^{'}By = x^{'}V^{'}\Lambda Vy = x^{'}A^{'}Ay = \langle Ax.Ay \rangle,$$

o produto interno usando o mapeamento linear A.

Capítulo 5

Caracterizando o estudo

Neste capítulo traremos informações sobre a plataforma onde os testes se realizaram bem como a biblioteca que implementa o SVM. Ainda apontaremos os principais aspectos que caracterizam não só o funcionamento da biblioteca, mas também os métodos de teste por ela disponibilizados.

5.1 Ambiente de trabalho

Partindo do estudo em relação à um problema de nossa preferência, todo o processo de teste e configuração do ambiente de trabalho será desenvolvido em plataforma UNIX, onde de antemão procuramos uma ferramenta para o uso de SVM. Duas ferramentas foram encontradas, aparentemente são as mais difundidas nesta área de pesquisa: TORCH e LIBSVM.

O trabalho, então, se concentrou na segunda, dada a sua maior facilidade de manipulação, seja quanto a configurar seu ambiente, seja quanto a possibilidade de implementar novas partes (ponto que se mostrou de suma importância dado nosso interesse quanto a utilização de núcleos próprios).

5.2 LIBSVM

A biblioteca LIBSVM ([13], [14]) é caracterizada como uma prática ferramenta para fins educacionais, visando principalmente obter resultados aceitáveis (que realmente contenham um grau razoável de informação) de maneira rápida e simples. Portanto trata-se de uma ferramenta que não busca a resolução de problemas difíceis ou desafiadores, mas sim trazer para uma maior gama de usuários os resultados finais que a aplicação de SVM tem em relação aos problemas estudados.

Procedimentos usuais

Representação dos dados

Cada dado será representado como um vetor de números reais e, portanto, se meu dado em estudo for marcado por atributos como cor, textura, material (por exemplo, um determinado tipo de cerâmica), tais atributos deverão ser mapeados em um vetor onde cada um deles será um determinado número real.

Escalonar os dados

O escalonamento dos dados tenta evitar que atributos com intervalos de valores muito grandes sobreponham àqueles com valores menores; citamos ainda problemas numéricos com os cálculos já que os núcleos dependem do produto interno de vetores de características. É indicado que os valores fiquem compreendidos nos intervalos [-1, 1] ou [0, 1].

Validação cruzada

Neste processo um conjunto de treinamento é dividido em n subconjuntos, a partir daí, sequencialmente, cada subconjunto

será testado com um classificador treinado nos demais *n-1* subconjuntos. A precisão da *validação cruzada* será a porcentagem de dados corretamente classificados. Tal método torna-se uma eficaz arma quanto a simular o trabalho com futuros dados desconhecidos, deixando o classificador mais robusto para a classificação (uma tentativa de evitar a *superespecialização*).

Principais métodos

- svm-scale: realizar o escalonamento dos dados de entrada;
- svm-train: realizar o treinamento do classificador;
- svm-predict: realizar a classificação dos dados.

Núcleos implementados

- núcleo linear: $K(x_i, x_j) = x_i.x_j$;
- núcleo polinomial: $K(x_i, x_j) = (\mu x_i x_j + r)^d, \mu > 0$;
- núcleo radial (RBF): $K(x_i, x_j) = e^{(-\mu \|x_i - x_j\|^2)}, \mu > 0$;
- núcleo sigmóide: $K(x_i, x_j) = tanh(\mu x_i x_j + r)$.

onde μ, d e r são parâmetros do núcleo (tais parâmetros nos testes deste trabalho foram utilizados com seus valores padrão).

Arquivo de dados

Os arquivos de entrada seguem o seguinte formato:

<rótulo> <índice1>:<valor1> <índice2>:<valor2>,

onde <rótulo> representa o valor alvo dos dados de treinamento (representa as classes quando do uso para classificação - n-classes), <índice> um número inteiro iniciando pelo 1 (os índices precisam estar em ordem crescente) e <valor> um número real. Os "índices" podem ser vistos como uma determinada característica da classe estudada e os "valores" a quantificação de tal característica. Daí segue que um arquivo com o seguinte formato

1 1:2 3:4 5:6

2 1:1 2:5 4:5,

seria lido como

- classe 1 com característica 1 tendo valor 2, característica 3 tendo valor 4 e característica 5 tendo valor 6;
- classe 2 com característica 1 tendo valor 2, característica 2 tendo valor 5 e característica 4 tendo valor 5.

Onde cada linha corresponde a um elemento.

Modo de utilização

- transformar os dados do problema para o formato da biblioteca LIBSVM;
- escalonar os dados dentro de um intervalo restrito;
- considerar um dos núcleos já disponíveis (a biblioteca já traz quatro núcleos implementados, porém o usuário pode criar o seu);
- treinar o classificador;
- testar.

5.3 Dinâmica dos testes

Abaixo mostraremos como funcionarão as rotinas da biblioteca LIBSVM na realização dos testes deste trabalho:

./svm-scale -l -1 -u 1 -s fator_escalonamento arquivo_treinamento > arquivo_treinamento.scale

Como verificado anteriormente, o escalonamento do conjunto de dados para valores pequenos facilita o processo de cálculos e melhora o resultado final da classificação e, dessa forma, escalonamos o arquivo "arquivo_treinamento" para que seus dados fiquem inseridos no intervalo $[-1, 1]$ (argumentos -l, limite inferior; -u, limite superior). Os fatores de escalonamento usados neste procedimento são então guardados no arquivo "fator_escalonamento" (argumento "-s") para posterior uso no arquivo que guarda a amostra a ser classificada. O novo arquivo de treinamento, agora escalonado, fica contido em "arquivo_treinamento.scale".

./svm-scale -r fator_escalonamento arquivo_teste > arquivo_teste.scale

O arquivo com os fatores de escalonamento previamente utilizados no arquivo de treinamento, é agora usado no arquivo de teste para que ambos os arquivos, de treinamento e de classificação, sigam o mesmo padrão. O arquivo final terá a extensão *".scale"*.

./svm-train -t número_nucleo arquivo_treinamento.scale

A função svm-train fará o treinamento do classificador em questão sobre o arquivo de treinamento (neste caso, escalonado previamente), utilizando um determinado núcleo. É nesta parte o

ponto chave dos nossos testes, pois ao usarmos um determinado núcleo (argumento "-t" seguido do tipo de núcleo) temos condição de verificar aquele com o melhor resultado final na classificação das amostras (os quatro núcleos já implementados na LIBSVM foram descritos no capítulo anterior - linear, polinomial, radial e sigmóide; representados por 0, 1, 2, 3 respectivamente). Deste processo será criado um arquivo "arquivo_treinamento.scale.model" contendo todos os dados relativos ao treinamento (os principais: núcleo usado, número de vetores de suporte, parâmetros usados nos núcleos - vistos na seção sobre LIBSVM, os próprios vetores de suporte) que serão usados pela função "svm-predict" na classificação da nova amostra.

./svm-predict arquivo_teste.scale arquivo_treinamento.scale.model arquivo_teste.predict

Realiza a classificação sobre a amostra do "arquivo_teste.scale" levando em conta o modelo "arquivo_treinamento.scale.model" criado durante o processo de treinamento. A saída nos dará a porcentagem de acerto na classificação da amostra, bem como jogará no arquivo "arquivo_teste.predict" a classe a que foi atribuído cada elemento da amostra.

Sítio

http://www.csie.ntu.edu.tw/ cjlin/libsvm

Capítulo 6

Problema tratado

O estudo de caso feito neste trabalho buscou resultados significativos quanto a diferenciação entre o que consideramos música e um ruído comum. Tal verificação iniciou-se desde o estudo de possíveis parâmetros para diferenciação dos dois casos, bem como a busca por utilitários que nos dessem uma visão mais clara da própria configuração da onda sonora em questão, uma busca que direcionasse a escolha da melhor abordagem para o caso em questão. Começaremos, então, pela apresentação da teoria por tráz do caso em estudo, passando pelas ferramentas de apoio, até a análise final.

6.1 Onda sonora: música e ruído

O *som* é resultado de vibrações de corpos elásticos, quando estas se encontram em certos limites de frequências ([15]). A tais vibrações damos o nome de vibrações sonoras, as quais se propagam no meio que circunda a fonte sonora (também conhecida como corpo sonoro, corpo emissor da onda) provocando compressões e distenções sucessivas (e transitórias) em todas as direções, ou seja, pequenas variações de pressão que por serem

mecânicas não se propagam no vácuo (Ennio, 1993).

Por ser um movimento material, o som apresenta certa energia e em vista das resistências opostas ao seu deslocamento a restitui para o meio. Temos dois tipos de restituições: quando parte de sua energia é transferida para um obstáculo que também passa a vibrar; ou quando parte da energia cinética se transforma em calor. Na verdade, a partir da fonte sonora, só uma parte da energia dá origem a vibrações sonoras que são transmitidas para as partículas adjacentes até que a energia mecânica disponível, diminuindo, não ocasione qualquer vibração perceptível; o restante é transformada em calor.

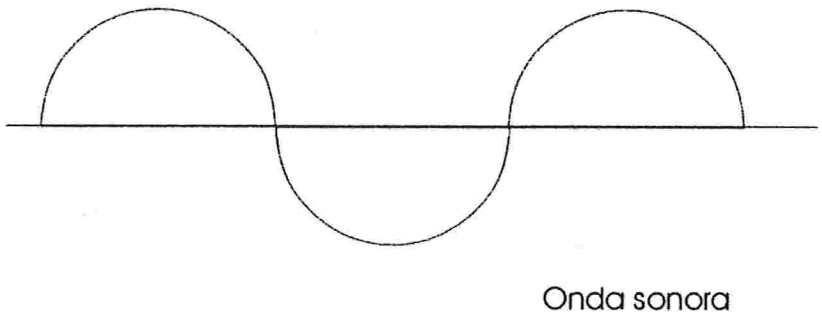

Figura 6.1: Onda sonora

Os elementos do som são:

- **Altura** - relacionada com a sequência das vibrações sonoras, isto é, com a frequência do som;
- **Timbre** - relacionado diretamente com a forma da onda, permite que identifiquemos a procedência da onda sonora (pessoa, instrumento musical);

- **Intensidade** - relacionada com a amplitude da onda, o maior ou menor deslocamento pelas partículas do meio em virtude ds oscilações que a formaram.

Nesse contexto, definimos *ruído* como vibrações completamente desordenadas provenientes de alguma fonte sonora, a soma de um grande número de frequências que acaba fazendo do ruído um gerador de desconforto ao ouvido humano quando audível. Como exemplos de ruídos podemos citar o trovão e um vulcão em erupção. Já a *música*, ao contrário do ruído, organiza o som durante certo espaço de tempo, é marcada por vibrações ordenadas e uma gama limitada de frequências.

Características importantíssimas da música são as escalas musicais ou notas musicais ([9], [10]), em um total de 7, são assim definidas:

Notas musicais								
Nota	dó	ré	mi	fa	sol	lá	si	do
Relação	1	9/8	5/4	4/3	3/2	5/3	15/8	2

Perceba que a cada intervalo de 8 notas elas se repetem, mudando apenas a frequência desta nova nota, que será o dobro da primeira, ou seja, a cada intervalo de oito notas as frequências dobram, sendo este intervalo conhecido como *oitava.*

Um outro conceito importante neste estudo é o de *sensação auditiva ou nível sonoro*, ela mede a maior ou menor impressão causada em nosso ouvido pelo som. Para cada frequência, quando aumentamos a pressão sonora eficaz, cresce a sensação auditiva, variando de zero na linha limite de audibilidade até o máximo na linha limite da dor. A unidade usada para medir a sensação auditiva é o decibel (dB): para uma frequência de 1000Hz o in-

tervalo audível do nível sonoro varia de 0 até 120dB (limite para a dor).

Definidos alguns conceitos básicos, mostraremos agora as ferramentas usadas bem como seu contexto de escolha.

6.2 Formatos de música: MP3

Os formatos de música digital tem gerado uma grande revolução na indústria fonográfica uma vez que possibilitaram manipular grandes quantidades de dados pela Internet e dessa forma popularizaram ainda mais este setor. São formatos conhecidos: WAV, MP3, WMA, entre outros.

O formato *MP3*, ou ainda MPEG Audio Layer III, é um método utilizado para comprimir arquivos de áudio. MPEG são as iniciais para Moving Picture Experts Group, um grupo que desenvolveu sistemas de compressão para dados de vídeo.

No CD (*compact disc*) de música convencional os dados são guardados em um formato de alta resolução, com taxa aproximada de 176.000 bytes por segundo, o que por um período de 4 minutos gera um arquivo de aproximadamente 42Mb, um espaço significativo para que seja manipulado facilmente, principalmente quando imaginamos armazenamento e transferência. O objetivo do formato MP3 é diminuir o tamanho deste arquivo de som sem modificar sensivelmente a qualidade da música, uma vez que compressões podem levar a perdas de qualidade, sendo tal processo resultado do que se espera para um produto final, ou seja, o resultado final da compressão precisa atender os objetivos por tráz deste processo e é exatamente isto que o formato MP3 procura fazer. Baseado em alguns critérios relacionados a percepção humana para a música o processo de compressão MP3

tenta diminuir a quantidade de dados de um arquivo musical e deixá-lo condizente com seus objetivos, são exemplos destes critérios:

- existem sons que o ouvido humano não consegue escutar;
- existem sons que o ouvido humano não consegue diferenciar qualitativamente;
- muitas vezes quando dois sons tocam simultâneamente não podemos distinguir os dois.

Por esta razão a compressão MP3 pode diminir o arquivo em média 10 vezes e transformar a manipulação de arquivos de música em uma atividade muito mais prática e menos custosa. Fatores que nos levaram a trabalhar com tais arquivos em nosso estudo, não só pela mais fácil armazenagem e processamento dos mesmos, mas também pelo maior acesso a este tipo de aquivo, um banco de dados de música de grande dimensão e, ao mesmo tempo, espaço reduzido (comparativamente com outros formatos).

Cabe ressaltarmos que os arquivos de música no formato MP3 usados no estudo são de propriedade privada.

6.3 Audacity

Viu-se necessário, então, o uso de uma ferramenta que possibilitasse um estudo do arquivo de áudio para analisar quais parâmetros poderiam ser usados no processo de classificação, ou melhor, diferenciação entre uma música e um ruído. Nesta análise, o *Audacity* se aproximou do nosso objetivo.

O Audacity é um editor de áudio de código aberto que permite captar, reproduzir e editar sons ([8]). Dentre os vários recursos disponíveis deste programa dois chamaram nossa atenção por terem possibilitado não só a análise dos arquivos de áudio mas também a geração de dados para um posterior processamento.

Como visto na foto abaixo, com o Audacity podemos caracterizar a música pela forma da onda, dando um idéia de como a música se comporta durante seu tempo de duração (fica claro os dois canais do som estéreo).

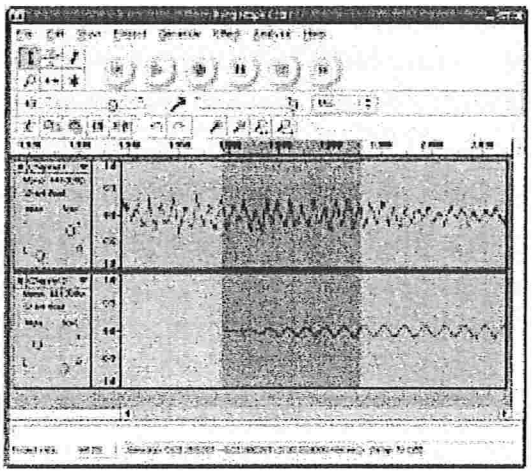

Figura 6.2: Audacity abrindo um arquivo MP3

Além disso o Audacity nos dá uma análise das frequências encontradas em determinado trecho do arquivo de áudio bem como o nível sonoro médio de cada uma delas no período de estudo, possibilitando ainda, exportar tais dados para um arquivo de texto comum.

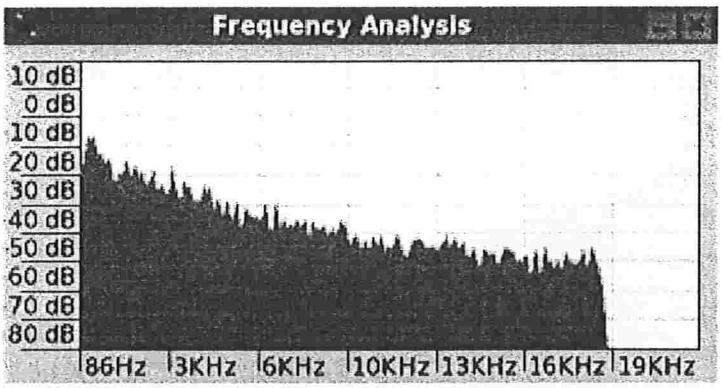

Figura 6.3: Análise de frequências

6.4 Analisando as estruturas dos arquivos de áudio

Como vimos no final da seção anterior, o programa Audacity é capaz de gerar uma saída de dados representando a análise feita sobre as frequências presentes no arquivo de áudio e seus respectivos níveis sonoros médios. Esta análise de frequência trouxe os primeiros dados concretos para verificarmos um possível parâmetro que diferenciasse uma música de um ruído.

Construímos dois repositórios de sons para nossos testes; o primeiro com músicas que variavam entre os estilos "pop", "rock" e clássico; o segundo formado por ruídos comuns (conversas, barulhos do cotidiano) e brancos (gerados pelo próprio Audacity).

Uma vez com os repositórios prontos passamos a estudar os relatórios de frequência dos estilos musicais, bem como dos ruídos. Nas figuras (6.4), (6.5) temos exemplos de músicas (rock e clássica), enquanto na figura (6.6) temos um ruído.

Podemos ver claramente que o nível sonoro, além de apresentar maiores valores nas frequências dos arquivos de música,

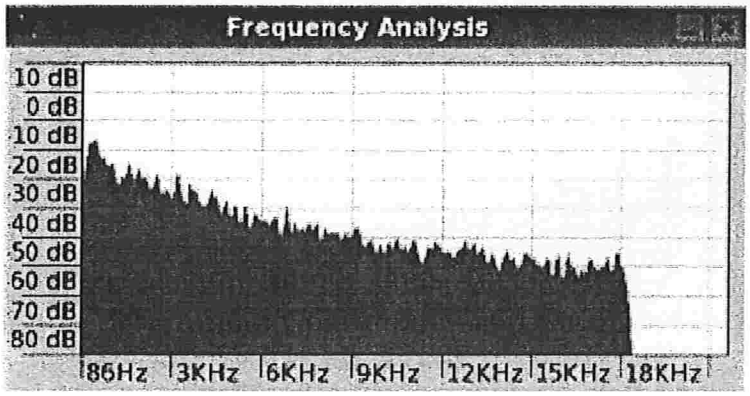

Figura 6.4: Relatório de frequências de um arquivo de "rock".

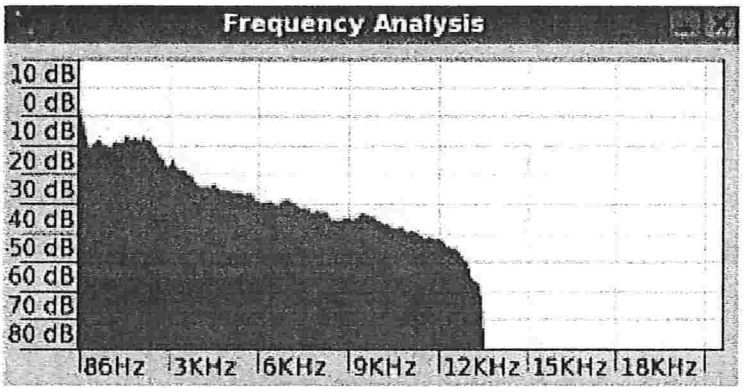

Figura 6.5: Relatório de frequências de um arquivo de música clássica.

também apresenta uma maior consistência, não diminuindo a partir de um determinado patamar (este patamar apresenta-se bem menor no caso dos ruídos). Foram estas observações que direcionaram nosso estudo para determinar o nível sonoro das amostras como o parâmetro a ser utilizado na busca de um melhor núcleo para a classificação das amostras de som.

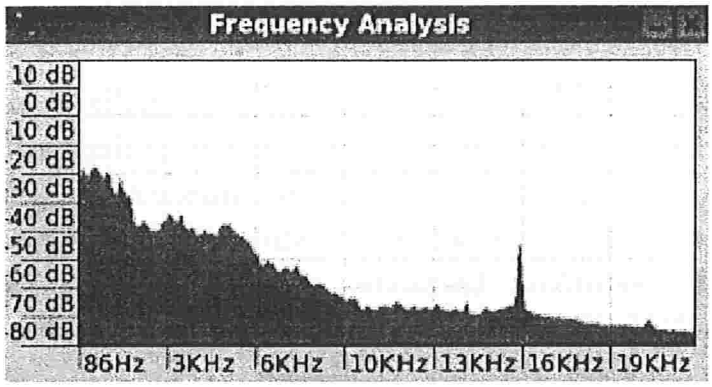

Figura 6.6: Relatório de frequências de um arquivo de ruído

6.5 Busca do mapeamento ideal

Uma vez que escolhemos o nível sonoro das frequências como o parâmetro a ser usado na classificação das amostras de som, passamos a processar estes dados a fim de conseguir uma maior confiabilidade em relação aos resultados obtidos. Tratamos, então, os arquivos gerados pelo Audacity, contendo os dados em questão, como um dado da amostra a ser classificada.

Outro ponto importante para nosso estudo é definir o número de músicas e ruídos usados no treinamento do algoritmo SVM bem como na classificação das amostras. Entre arquivos de música, tanto o arquivo de treinamento do algoritmo quanto o de classificação propriamente dito, apresentavam 340 exemplos cada, enquanto o número de exemplos de ruídos para os mesmos arquivos era de 220. Ressaltamos que os resultados a seguir são reflexo dos testes realizados com os valores descriminados anteriormente, simplesmente por tentarmos ser os mais fiéis possiveis à realidade, uma vez que apesar das variações númericas encontradas com um número de exemplos menores, as tendências dos resultados sempre se mantiveram.

6.5.1 Níveis sonoros normalizados

O primeiro passo foi, para cada amostra, normalizar os valores dos níveis sonoros, uma vez que os mesmos poderiam ter sido gravados com intensidades sonoras diferentes e nossa análise levaria em conta tal diferenciação, causando uma má interpretação do verdadeiro resultado. Tal normalização possibilitou que voltássemos nossa atenção para os intervalos dos níveis sonoros entre as frequências, este sim um dado relevante.

Uma característica das amostras é o fato dos níveis sonoros sempre referenciarem as mesmas frequências, ou seja, todas as amostras geradas sempre nos davam os valores dos níveis sonoros para a mesma faixa de frequências, a saber, o intervalo de 86 até 21963Hz. Este fato define um vetor de características (já no formato do arquivo de entrada para a biblioteca LIBSVM) comum a todas as amostras.

Vetor de características						
-34dB	-30dB	-29dB	-34dB	-34dB	-36dB	-41dB
80Hz	172Hz	258Hz	344Hz	430Hz	516Hz	602Hz

Com as amostras normalizadas fizemos os primeiros testes de classificação variando o tamanho do vetor de características, ou seja, quais frequências seriam analisadas bem como o número delas (dimensão do vetor). Destacamos que o intervalo de frequências usado concentrou-se na região mais intermediária do arquivo de áudio:

Dimensão 50	
Llinear	64.7826%
Polinomial	71.7391%
Radial	28.2609%
Sigmóide	71.7391%

Dimensão 40	
Linear	85.8696%
Polinomial	71.7391%
Radial	28.2609%
Sigmóide	71.7391%

Dimensão 30	
Linear	86.587%
Polinomial	71.7391%
Radial	28.2609%
Sigmóide	71.7391%

Dimensão 20	
Linear	86.413%
Polinomial	71.7391%
Radial	28.2609%
Sigmóide	71.7391%

Dimensão 10	
Linear	86.9565%
Polinomial	71.7391%
Radial	28.2609%
Sigmóide	71.7391%

Percebemos que apesar de pequenas variações na porcentagem de dados classificados, temos que, em média, a análise pura e simples dos níveis sonoros levou a uma taxa de 85% de acerto na classificação executada pelo SVM, um valor que já comprovava a correta direção na análise do problema.

6.5.2 Desvio padrão

Dado o passo anterior voltamos nossa atenção para a melhor forma de analisar as variações de nível sonoro nas frequências, como condensar tal informação na formação de um núcleo que melhor guiasse o SVM na tarefa de classificação.

Como a questão era investigar até que ponto as variações de níveis sonoros nas frequências poderiam direcionar uma classificação, um procedimento que gerasse um primeiro processamento destes dados deveria salientar até que ponto os níveis desviavam de uma determinada média amostral, em outras palavras, até que ponto dado um intervalo de frequências a variação de nível sonoro seguiria um determinado padrão para cada classe.

Neste instante o uso do *desvio padrão* sinalizou como um procedimento de manipulação de dados e geração de um primeiro mapeamento para o SVM:

$$\sigma = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(x_i - \bar{x})}$$

onde $\bar{x}$ é a média da amostra, N o número de amostras e x_i cada elemento da amostra.

O objetivo foi estender o vetor de características, representado pelas frequências dos exemplos, para incluir informações sobre a variação das freqüências, usando o desvio padrão. Isto foi feito utilizando-se uma sequência de intervalos sobrepostos das frequências presentes no arquivo gerado pelo Audacity. A variação das frequências que participariam deste processo variaria conforme fôssemos percorrendo o intervalo de frequências, um índice por vez, como ilustrado na figura 6.7.

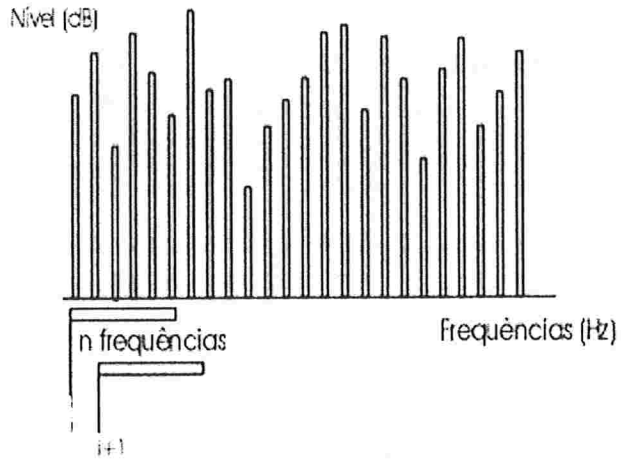

Figura 6.7: Desvio padrão de um determinado intervalo de frequências.

Os desvios padrões assim obtidos formaram um novo vetor de características, que foi processado por um dos quatro outros

núcleos inseridos da biblioteca LIBSVM, com os seguintes resultados:

Dimensão 50	
Desvío padrão e núcleo linear	95.6522%
Desvío padrão e demais núcleos	71.7391%
Dimensão 40	
Desvío padrão e núcleo linear	89.1304%
Desvío padrão e demais núcleos	71.7391%
Dimensão 30	
Desvío padrão e núcleo linear	88.3478%
Desvío padrão e demais núcleos	71.7391%
Dimensão 20	
Desvío padrão e núcleo linear	77.3913%
Desvío padrão e demais núcleos	71.7391%
Dimensão 10	
Desvío padrão e núcleo linear	71.7391%
Desvío padrão e demais núcleos	71.7391%

O número de frequências usadas para gerar cada desvio padrão foi de 200, sendo que para o caso de dimensão 50 a taxa de classificações corretas ultrapassou os 95%. Uma melhora sensível em relação ao melhor caso no estudo dos níveis sonoros, de aproximadamente 9%.

6.5.3 Compondo os métodos

Mesmo alcançando uma taxa de 95% de classificações acertadas com o algoritmo SVM, ainda nos chamava a atenção a possibilidade de analisarmos, juntos, o desvio padrão e os níveis sonoros, incluindo os dois conjuntos de valores no vetor de características. Uma verificação de até que ponto os dois tipos de classificações feitas até então poderiam se relacionar na mesma classificação.

Mantendo os mesmas dimensões considerados neste estudo (50 até 10), variamos quanto o desvio padrão ou o nível sonoro comporiam nosso vetor de características. Sempre buscando qual composição traria uma taxa de acerto superior a 95% na classificação.

Dentre todas as tentativas um caso mostrou um desempenho ainda superior a 95%. Dentro de uma dimensão 50 com uma composição do vetor de características em torno de 70% do parâmetro desvio padrão e 30% do parâmetro nível sonoro, observamos um acrécimo de quase 2% no que então era considerado o melhor caso de classificação nesta pesquisa:

Dimensão 10	
Desvío padrão e núcleo linear	97.2826%
Desvío padrão e demais núcleos	71.7391%

6.5.4 Resultado final: o mapeamento em questão

No estudo do que seria o melhor mapeamento para classificação de músicas e ruídos, na diferenciação entre estes dois tipos de dados, a análise do nível sonoro não só confirmou nossas ex-

pectativas quanto a ser um parâmetro de influência na classificação, como sua manipulação levou a resultados deveras expressivos, com taxas de acerto que superaram o índice de 97%. Neste caminho, ainda verificamos que o núcleo gerador de tão significativa classificação (núcleo linear) agiu sobre um vetor de características resultante de um mapeamento que buscou gerar desvíos padrões sobre intervalos de freqüencias.

Capítulo 7

Conclusão

A pesquisa realizada neste trabalho procurou não só trazer os resultados práticos da aplicação do algoritmo SVM na classificação de música e ruído, na distinção entre estas duas classes; mas também trazer parte do embasamento teórico das *máquinas de suporte vetorial.*

Na primeira parte buscou-se construir uma base para o estudo de caso a ser realizado, uma maneira de englobar os principais tópicos constituintes da teoria; passando pela Teoria do Aprendizado Estatístico, alicerce da busca pelo melhor classificador (*Minimização do Risco Estrutural*); a própria definição de um *classficador linear* e como o SVM implementa a busca pelo melhor classificador dentro de uma família; a definição do que seria um *núcleo* e como através deste, trabalhar no *espaço de característica.*

Na segunda parte iniciamos a descrição de como se dariam os testes de classificação, desde a caracterização do ambiente de trabalho, passando pela sua configuração até alcançar a metodologia usada para a realização dos testes com músicas e ruídos. Houve a preocupação de salientar não só a biblioteca que implementa o algortimo SVM como também todas as demais fer-

ramentas usadas no processamento dos dados em questão.

Tal direção na composição do trabalho visou trazer progressivamente o estudo ao seu objetivo final, elaborando suas atividades de uma forma que trouxesse à tona a própria estrutura da pesquisa, a maneira como foi desenvolvida e compreendida.

Olhando para os resultados práticos alcançados viu-se a possibilidade de analisar separadamente um problema de classificação e direcionar a construção de um *mapeamento* que melhor reflitisse as necessidades deste problema de classificação. Bastando, desta forma, modelar as análises para dentro da teoria do SVM, daí a importância de como o trabalho foi formulado. Desde o primeiro teste, utilizando somente os níveis sonoros normalizados, a porcentagem de acerto localizada em torno de 85% já nos demosntrou quão importante foi aplicar a metodologia do SVM; os resultados subsequentes, passando a taxa de 95% de acerto somente nos motivaram ainda mais na pesquisa, uma comprovação que o caminho traçado havia trazido resultados concretos para o trabalho.

Um ponto de suma importância é o fato dos testes realizados, além de nos darem satisfatórios resultados, abrirem ótimas perspectivas para a evolução deste mesmo trabalho, dependendo somente da análise de outros parâmetros (além de nível sonoro) usados na classificação.

Capítulo 8

Apêndice

Neste capítulo colocaremos uma série de definições que se fazem importantes para um maior entendimento da teoria introduzida neste estudo.

8.1 Problema de otimização

Um problema de otimização geral pode ser colocado como:

Definição (Problema de Otimização Primal) - Dadas as funções f, g_i, onde $i = 1, ..., k$, e $h_i, i = 1, ..., m$, definidas sobre um domínio $U \subseteq \Re^n$,

$$\begin{array}{rll} minimizar & f(x), & w \in U, \\ onde & g_i(\vec{w}) \leq 0, & \text{i=1,...,k,} \\ & h_i(\vec{w}) = 0, & \text{i=1,...,m,,} \end{array}$$

onde $f(\vec{w})$ é chamada *função objetivo* e as demais relações são chamadas *restrições de igualdade e desigualdade.*

8.2 Teoria de Lagrange

A teoria de Lagrange se propõe em caracterizar a solução de um problema de otimização onde inicialmente não existem restrições de desigualdade. Seus principais conceitos são os Multiplicadores de Lagrange e a Função de Lagrange. Convém ressaltar que tal método foi desenvolvido em 1797, por Lagrange, generalisando o trabalho de Fermat, feito em 1629; ainda em 1951 Kuhn e Tucker extenderam o método de Lagrange para restrições de desigualdade.

Teorema 1 (Fermat) - Uma condição necessária para $\vec{w}^*$ ser um mínimo de $f(\vec{w})$, $f \in C^1$ é $\frac{\partial f(\vec{w}^*)}{\partial \vec{w}} = 0$. Esta condição junto à convexidade de f é também uma condição suficiente.

Em problemas com restrições precisamos definir uma função chamada *Lagrangiano*, que incopora informação sobre a função objetivo e a restrições (sua estacionaridade pode ser usada para detectar soluções). Daí, como o *Lagrangiano* é definido como a função objetivo mais uma combinação linear das restrições, aos coeficientes das restrições chamamos *Multiplicadores de Lagrange.*

Definição - Dado um problema de otimização com a função objetivo $f(\vec{w})$ e restrições de igualdade $h_i(\vec{W}) = 0, i = 1, ..., m$, nós definimos o Lagrangiano como

$$L(\vec{w}, \beta) = f(\vec{w}) + \sum_{i=1}^{m} \beta_i h_i(\vec{w})$$

onde os coeficientes β_i são chamados *Multiplicadores de Lagrange.*

Teorema 2 (Lagrange) - Uma condição necessária para um ponto normal $\vec{w}^*$ ser um mínimo de $f(\vec{w})$, $f \in C^1$ com as restrições $h_i(\vec{W}) = 0, i = 1, ..., m$, senfo f e $h_i \in C^1$ é

$$\frac{\partial L(\vec{w}^*, \beta^*)}{\partial \vec{w}} = 0.$$

$$\frac{\partial L(\vec{w}^*, \beta^*)}{\partial \beta} = 0.$$

Consideraremos agora o caso geral onde o problema de otimização terá restrições de igualdade e desigualdade:

Definição - Dado um problema de otimização com domínio $U \subseteq \Re^n$,

$$\begin{array}{rlr} \textit{minimizar} & f(x), & w \in U, \\ \textit{onde} & g_i(\vec{w}) \leq 0, & \text{i=1,...,k,} \\ & h_i(\vec{w}) \leq 0, & \text{i=1,...,m,} \end{array}$$

nós definimos a *função de Lagrange geral* como

$$\begin{aligned} L(\vec{w}, \alpha, \beta) &= f(\vec{w}) + \Sigma_{i=1}^m \alpha_i g_i(\vec{w}) + \Sigma_{i=1}^m \beta_i h_i(\vec{w}) \\ &= f(\vec{w}) + \alpha' g(\vec{w}) + \beta' h(\vec{w}). \end{aligned}$$

Em relação ao problema de otimização primal podemos colocar nossa próxima definição:

Definição - O problema dual de Lagrange é definido como

$$\begin{aligned} maximizar \quad & \theta(\alpha, \beta) \\ onde \quad & \alpha \geq 0 \end{aligned}$$

onde $\theta(\alpha, \beta) = inf_{\vec{w} \in U} L(\vec{w}, \alpha, \beta)$.

Podemos agora finalizar esta seção com o teorema que nos dá condições para achar uma solução ótima para o prblçema geral de otimização:

Teorema (Kuhn-Tucker) - Dado um problema de otimização com um domínio convexo $U \subseteq \Re^n$,

$$\begin{aligned} minimizar \quad & f(x), && w \in U, \\ onde \quad & g_i(\vec{w}) \leq 0, && \text{i=1,...,k,} \\ & h_i(\vec{w}) = 0, && \text{i=1,...,m,,} \end{aligned}$$

com $f \in C^1$ convexo e g_i, h_i funções afins, a condições necessária e suficientes para $\vec{w}$ ser um ótimo é existir α^* e β^* tal que

$$\frac{\partial L(\vec{w}^*, \alpha^*, \beta^*)}{\partial \vec{w}} = 0,$$

$$\frac{\partial L(\vec{w}^*, \alpha^*, \beta*)}{\partial \beta} = 0,$$

$$\alpha_i^* g_i(\vec{w}^*) = 0, i = 1, ..., k,$$

$$g_i(\vec{w}^*) \leq 0, i = 1, ..., k,$$

$$\alpha_i^* >= 0, i = 1, ..., k.$$

8.3 Espaço de Hilbert, autovalores e autovetores

Definição - X é considerado um espaço vetorial se a operação de adição e multiplicação por escalar são definidas sobre X tal que, para $\vec{x}, \vec{y} \in X$ e $\alpha \in \Re$,

$$\begin{aligned} x + y &\in X, \\ \alpha x &\in X, \\ 1x &= X, \\ 0x &= 0, \end{aligned}$$

onde X possui as propriedades comutativas e associativa com identidade 0 na operação de adição, possui propriedade distributiva na multiplicação por escalar:

$$\alpha(x + y) = \alpha x + \alpha y,$$

$$(\alpha + \beta)(x) = \alpha x + \beta x.$$

Completamos dizendo que os elementos pertencentes a X são chamados *vetores* e os números reais, *escalares* ([5]).

Definição Sejam X e Y sobre espaços vetoriais sobre $\Re$. Uma função $F : X \to Y$ é chamada de *Transformação linear* de X em Y se, e somente se

- $F(\vec{x_1} + \vec{x_2}) = F(\vec{x_1}) + F(\vec{x_2}), \forall \vec{x_1} + \vec{x_2} \in X$ e
- $F(\alpha\vec{x}) = \alpha F(\vec{x}), \forall \alpha \in \Re$ e $\forall \vec{x} \in X$.

Caso tenhamos uma transformação linear $F : X \leftrightarrow X$ esta é chamada de *operador linear.*

Definição - Para cada $\vec{x}, \vec{y} \in X$ dizemos que o *produto interno* $\langle \vec{x} \cdot \vec{y} \rangle$ está definido sobre o espaço vetorial X se

- $\langle \vec{x} \cdot \vec{y} \rangle = \langle \vec{y} \cdot \vec{x} \rangle$,
- $\langle \vec{x} \cdot \vec{x} \rangle > 0$, e $\langle \vec{x} \cdot \vec{x} \rangle = 0 \Leftrightarrow \vec{x} = 0$.

Definição - Dado dois vetores $\vec{x}, \vec{y}$, eles serão ortogonais se $\langle \vec{x} \cdot \vec{y} \rangle = 0$. A partir daí podemos dizer também que dado um conjunto $S = \vec{x}_1, ..., \vec{x}_n$ de vetores $\in X$ (espaço vetorial com produto interno definido) é chamado *ortonormal* se $\langle \vec{x}_i \cdot \vec{x}_j \rangle = \delta_{ij}$, onde $\delta_{ij} = 1$ se $i = j$, e 0 caso contrário.

Para um conjunto ortonormal S e um vetor $y \in X$ chamamos $\sum_{i=1}^{n} \langle \vec{x}_i \cdot \vec{y} \rangle \vec{x}_i$ de *Série de Fourier* para y.

Definição - Para um espaço vetorial com produto interno definido, sendo x, y pertencentes a este espaço vale a desigualdade de *Cauchy-Schwarz*

$$\langle x.y \rangle \leq \|x\| \|y\|.$$

Definição - Um espaço H é separável se existe um subconjunto contável $D \subseteq H$, tal que cada elemento de H é o limite de uma sequência de elementos de D.

Definição - Um espaço H é completo se para qualquer $h \in H$ temos

$$\lim_{n\to\infty} sup\|h_n - h_m\| \to 0 \quad onde \quad n < m.$$

Definição - *Espaço de Hilbert* é um espaço completo, separável e com o produto interno definido.

Teorema - Cada espaço de Hilbert tem uma base ortonormal. Seja $S = \vec{x}_1, ..., \vec{x}_n$ uma base ortonormal do Espaço de Hilbert H, podemos dizer que para $\forall \vec{y} \in H$,

$$\vec{y} = \sum_{i=1}^{n} \langle \vec{x}_i \cdot \vec{y} \rangle \vec{x}_i$$

e $\|\vec{y}\| = \Sigma_{i=1}^{n} \mid \langle \vec{y} \cdot \vec{x}_i \rangle \mid^2$.

Este teorema nos mostra que qualquer elemento do espaço de Hilbert pode ser escrito como uma combinação linear dos elementos de sua base ortonormal.

Definição - Dado o operador linear F sobre o espaço de Hilbert H e um vetor $\vec{x} \in H$ tal que $\vec{x} \neq 0$, podemos dizer que dada a expressão $F\vec{x} = \lambda\vec{x}$, λ será chamado *autovalor* de F enquanto $\vec{x}$ seu correspondente *autovetor*.

Definição - Um operador linear $F \in H$(espaço de Hilbert) é limitado se existe um número $\|F\|$ tal que $\|F\vec{x}\| \leq \|F\|\|F\vec{x}\|$, $\forall \vec{x} \in H$.

Definição - Um operador limitado $F \in H$ (espaço de Hilbert) será chamado de *auto-adjunto* se

$$\langle F\vec{x}.\vec{y}\rangle = \langle \vec{x}.F\vec{y}\rangle,$$

com $x, y \in H$. É importante levantarmos aqui que para um espaço finito $\Re^n$ a matriz da transformação linear F, chamada de M_F, n x n, é igual a sua transposta, ou seja, $M_F = M_F^{'}$ e denominada *matriz simétrica.*

Definição - Uma matriz simétrica n x n será *positiva definida* se todos seus autovalores são positivos.

Definição - Uma matriz simétrica n x n será *positiva semi-definida* se todos seus autovalores são não negativos.

Ou ainda,

Definição - Dada uma matriz simétrica M_F, esta será *positiva definida* se e somente se para qualquer $\vec{x} \neq 0$ temos $\vec{x}^{'} M_F \vec{x} > 0$.

Definição - Dada uma matriz simétrica M_F, esta será *positiva semi-definida* se e somente se para qualquer $\vec{x} \neq 0$ temos $\vec{x}^{'} M_F \vec{x} \geq 0$.

8.4 Provas

Seção 4.5 - Prova: Teorema 1: A segunda parte do teorema nós já provamos na seção 4.4 quando provamos que matrizes de núcleo são positivas semi-definidas, restando agora criar um

mapeamento ϕ dentro de um espaço de Hilbert para o qual K é o núcleo.

Sabemos que os elementos do espaço de característica são de fato funções, além de pontos num espaço vetorial. A partir daí poderemos dizer que o espaço de característica será o conjunto de funções que usaremos no problema de aprendizado, o conjunto de funções dentre as quais acharemos aquela que melhor representará o problema em questão. A este conjunto de funções chamaremos F e o definiremos como

$$F = \{\sum_{i=1}^{l} \alpha_i K(\vec{x}_i, \cdot) : l \in N, \vec{x}_i \in X, \alpha_i \in \Re, i = 1, ..., l\}$$

Podemos ver que F é um espaço vetorial (vide Apêndice) e definiremos agora um produto interno sobre F. Com $f, g \in F$ sendo

$$f(\vec{x}) = \sum_{i=1}^{l} \alpha_i K(\vec{x}_i, \vec{x})$$

e

$$g(\vec{x}) = \sum_{i=1}^{n} \beta_i K(\vec{y}_i, \vec{x})$$

definimos

$$\langle f.g \rangle = \sum_{i=1}^{l} \sum_{j=1}^{n} \alpha_i \beta_j K(\vec{x}_i, \vec{y}_j) = \sum_{i=1}^{l} \alpha_i g(\vec{x}_i) = \sum_{j=1}^{n} \beta_i f(\vec{y}_j), \quad (8.1)$$

onde a segunda e terceira igualdade seguem as definições de f e g. Dessas igualdades fica claro que $\langle f.g\rangle = \langle g.f\rangle$, $\langle f.f\rangle > 0$, e $\langle f.f\rangle = 0 \Leftrightarrow f = 0$ para qualquer $f, g \in F$, ou seja, o produto interno está definido em F.

Sabemos ainda que as matrizes de núcleo são positivas semi-definidas desde que

$$\langle f.f\rangle = \sum_{i=1}^{l}\sum_{j=1}^{l} \alpha_i\alpha_j K(\vec{x_i}, \vec{x_j}) = \vec{\alpha}' K\vec{\alpha} \geq 0,$$

onde α é um vetor com entradas α_i, $i = i, ..., l$ e K uma matriz de núcleo construida a partir de $\vec{x_1}, ..., \vec{x_l}$.

Podemos chegar a outra propriedade se levarmos em conta as equações (8.1) mais o fato de $g = K(\vec{x}, \cdot)$

$$\langle f.K(\vec{x}, \cdot)\rangle = \sum_{i=1}^{l} \alpha_i K(\vec{x_i}, \vec{x}) = f(\vec{x})$$

Tal propriedade é conhecida como *propriedade de reprodução* do núcleo, ou seja, dado um determinado núcleo, podemos criar o espaço de Hilbert F_K associado à ele. Falta mostramos que tal espaço é separável e completo.

Omitiremos maiores detalhes técnicos da prova de separabilidade dizendo que esta propriedade decorre do fato do núcleo ser contínuo. Já para a prova de completeza consideraremos uma entrada fixa $\vec{x}$ para um sequencia de *Cauchy* $(f_n)_{n=1}^{\infty}$, daí teremos que

$$(f_n(\vec{x}) - f_m(\vec{x}))^2 = \langle f_n - f_m, K(\vec{x}, \cdot)\rangle^2 \leq |f_n - f_m|^2 K(\vec{x}, \vec{x}).$$

Ou seja, $f_n(\vec{x})$ é limitado por uma sequência de *Cauchy* de números reais.

Dada a construção do espaço de característica F_K especificaremos a imagem de uma entrada x sob o mapeamento ϕ como

$$\vec{\phi} : \vec{x} \in X \rightarrow \vec{\phi}(\vec{x}) = K(\vec{x}, \cdot) \in F_K;$$

e calcularemos o produto interno entre um elemento de F_K e a imagem de uma entrada $\vec{x}$ desta maneira

$$\langle f.K(\vec{x}, \cdot)\rangle = \langle f.K(\vec{x}, \cdot)\rangle = f(\vec{x}).$$

ou seja, f pode ser representado como uma função linear definida sob um produto interno, exatamente o que precisavamos provar.

◇

Seção 4.5 - Prova: Teorema de Mercer: Para a primeira parte do teorema mostraremos que a positividade do operador integral implicará que as todas as submatrizes finitas serão positivas semi-definidas. Suponha que haja uma submatriz finita sobre os pontos $x_1, ..., x_l$ que não seja positiva semi-definida. Para um vetor α tal que

$$\sum_{i,j=1}^{l} K(x_i, y_j)\alpha_i\alpha_j = \epsilon < 0,$$

e seja

$$f_\sigma(x) = \sum_{i=1}^{l} \alpha_i \frac{1}{(2\pi\sigma)^{d/2}} \exp \frac{-\|x - x_i\|^2}{2\sigma^2} \in L_2(X),$$

onde d é a dimensão do espaço X. Temos, então, que

$$\lim_{\sigma \to 0} \int_{X \times X} K(x,y) f_\sigma(x) f_\sigma(y) dx dy = \epsilon.$$

Porém, para algum $\sigma > 0$ a integral será menor que 0, o que vai contra a positividade do operador integral.

Consideraremos, agora, uma base ortonormal $\phi_i(\cdot)$, $i = 1, ...$ de F_K, o RKBS do núcleo K. Teremos, então, a seguinte série de Fourier para $K(x, \cdot)$

$$K(x,y) = \sum_{i=1}^{\infty} \langle K(x,\cdot) \cdot \phi_i(\cdot) \rangle \phi_i(y) = \sum_{i=1}^{\infty} \phi_i(x) \phi_i(y),$$

como queríamos demonstrar.

Terminaremos mostrando que a série $\Sigma_{i=1}^{\infty} \|\phi_i\|_{L_2}^2$ é convergente usando a propriedade do espaço X ser compacto

$$\begin{aligned} \infty \; > \; & \int_X \lim_{n \to \infty} \sum_{i=1}^{n} \phi_i(x) \phi_i(x) dx = \lim_{n \to \infty} \int_X \sum_{i=1}^{n} \phi_i(x) \phi_i(x) dx \\ = \; & \lim_{n \to \infty} \sum_{i=1}^{n} \int_X \phi_i(x) \phi_i(x) dx = \lim_{n \to \infty} \sum_{i=1}^{\infty} \|\phi_i\|_{L_2}^2 \end{aligned}$$

$\diamond$

Referências Bibliográficas

[1] Jean-Philippe Vert, *Introduction to Support Vector Machines and Applications to Computational Biology*, 2001.

[2] Vojislac Kecman, *Learning and Soft Computing*, first Edition, The MIT Press, 2001.

[3] John Shawe-Taylor, Nello Cristianini, *Kernel Methods for Pattern Analysis*, first Edition, Cambridge University Press, 2004.

[4] John Shawe-Taylor, Nello Cristianini, *An Introduction to Support Vector Machines and other kernel-based learning methods*, first Edition, Cambridge University Press, 2000.

[5] Carlos A. Callioli, Hygino H. Domingues, Roberto C. F. Costa, *Álgebra Linear e Aplicações*, sexta edição, Editora Atual S/A, 1998.

[6] Marcos Nascimento Magalhães, Antonio Carlos Pedroso de Lima, *Noções de Probabilidade e Estatística*, primeira edição, IME-USP, 1999.

[7] Christopher J. C. Burges, *A Tutorial on Support Vector Machines for Pattern Recognition*, 1998. Disponível em *http://www.svms.org/tutorials/Burg98.pdf*

[8] AudacityStore.com, *Audacity.* Disponível em *http://audacity.sourceforge.net/* - 20 de janeiro de 2007.

[9] Centro de Divulgação Científica e Cultural, *Ondulatória.* Disponível em *http://www.cdcc.sc.usp.br/ondulatoria/* - 5 de janeiro 2007.

[10] Universidade Federal de Santa Maria - Grupo de Ensino de Física (GEF), *Oscilações e Ondas.* Disponível em *http://www.ufsm.br/gef/* - 5 de janeiro de 2007.

[11] Andrew W. Moore, *VC dimension tutorial*, 2001. Disponível em *http://www.autonlab.org/tutorials/vcdim.html.*

[12] Vojislac Kecman, *Support Vector Machines and Neural Networks - similarities and differences*, 2001. Disponível em *http://www.svms.org/tutorials/Kecman2001.pdf.*

[13] Chih-Chung Chang and Chi-Jen Lin, *LIBSVM: a Library for Support Vector Machines*, 2005.

[14] Chih-Wei Hsuand, Chih-Chung Chang, Chi-Jen Lin, *A Practical Guide to Support Vector Classification*, Department of Computer Science an Information Engineering National Taiwan University, 2005.

[15] Ennio Cruz da Costa, *Acústica Técnica*, primeira edição, Editora Edgard Blucher LTDA, 2003.

[16] Luc Devroye, Laszlo Gyorfi, Gabor Lugosi, *A Probabilistic Theory of Pattern Recognition*, first Edition, New York: Springer, 1996.

[17] Vladimir Naumovich Vapnik, *Statistical Learning Theory*, New York: John-Wiley, 1998.

[18] Sujun Hua and Zhirong Sun, *Support vector machine aproach for protein subcellular localization prediction*, 2001. Disponível em *http://bioinformatics.oxfordjournals.org/cgi/content/abstract/17/8/721.*

[19] Terrence S. Furey, Nello Cristianini, Nigel Duffy, David W. Bednarski, Michèl Schummer and David Haussler *Support vector machine classification and validation of cancer tissue samples using microarray expression data*, 2000. Disponível em *http://bioinformatics.oxfordjournals.org/cgi/content/abstract/16/10/906.*

[20] Simon Tong and Edward Chang, *Support vector machine active learning for image retrieval*, 2001. Disponível em *http://portal.acm.org.*

[21] R. Burbidge, M. Trotter, B. Buxton, S. Holden, *Drug design by machine learning; support vector machines for pharmaceutical data analysis*, 2001. Disponível em *http://www1.elsevier.com/vj/MathWeb/13/16/28/18/article.pdf.*