Aprendizado de Máquina

Aula 11: Máquinas de Vetores de Suporte

André C. P. L. F de Carvalho
ICMC/USP
andre@icmc.usp.br

Tópicos

- Introdução
- Risco empírico e risco estrutural
- Margens
- Margens suaves
- SVMs
- Kernels
- Multiclasses

Teoria de Aprendizado Estatístico

- Algoritmos de AM
  - Estimam um função (modelo) a partir de um conjunto finito de objetos (exemplos)
  - Função preditiva (classificador ou regressor)
- TAE estabelece princípios para induzir uma função com boa capacidade de generalização
  - Vapnik e Chervonenkis em 1968
  - Base das máquinas de vetores de suporte

TAE

- Sejam
  - h: classificador (hipótese, modelo, função)
  - H: conjunto de todos os classificadores que um algoritmo de AM pode induzir
- Algoritmo de AM utiliza conjunto de dados de treinamento para:
  - Induzir um classificador $\hat{h} \in H$
- Assume que dados são gerados de forma i.i.d. de acordo com *P(x,y)*

TAE

- TAE define condições matemáticas para auxiliar na escolha de uma boa função $\hat{h}$
  - A partir de um conjunto de dados de treinamento
  - Permite escolher $\hat{h}$ com menor risco esperado
    - Para manter bom desempenho com novos dados, avalia:
      - Desempenho preditivo de $\hat{h}$ para dados do conjunto de treinamento
      - Complexidade de $\hat{h}$

Risco esperado (funcional)

- Erro esperado de um classificador para todos os dados de um domínio

$$R(h) = \int c(h(x), y) dP(x, y)$$

C(h(x),y): função de custo relacionando a previsão h(x) à saída desejada y, y ∈ {-1,+1}

$$c(h(x), y) = \frac{1}{2}|y - h(x)|$$

Tipo de função muito empregada em problemas de classificação é a 0-1

- Não é possível minimizar diretamente
  - *P(x,y)* real é desconhecida
  - Alternativa: minimizar risco empírico

Risco empírico

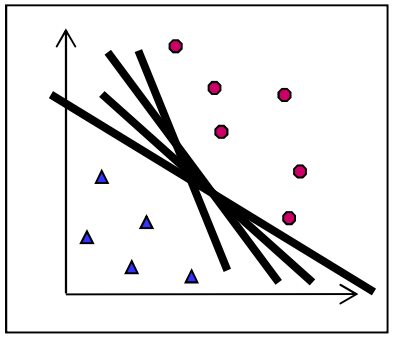
- Algoritmos de AM supervisionado induzem $\hat{h}$ que minimize erro de treinamento
  - Esperando poucos erros para novos dados
- Minimização do risco empírico

$$R_{emp}(h) = \frac{1}{n}\sum_{i=1}^{n} c(h(x_i), y_i)$$

  - *n*: número de exemplos de treinamento
    - Quando $n \rightarrow$ *infinito*, $R_{emp} \rightarrow R$ *(risco esperado)*

Minimização de risco empírico

- Minimização convencional do risco empírico sobre conjunto de treinamento
  - Não implica em boa generalização
  - Diferentes funções podem aproximar bem os dados de treinamento
    - Difícil determinar a função que melhor captura a distribuição real dos dados

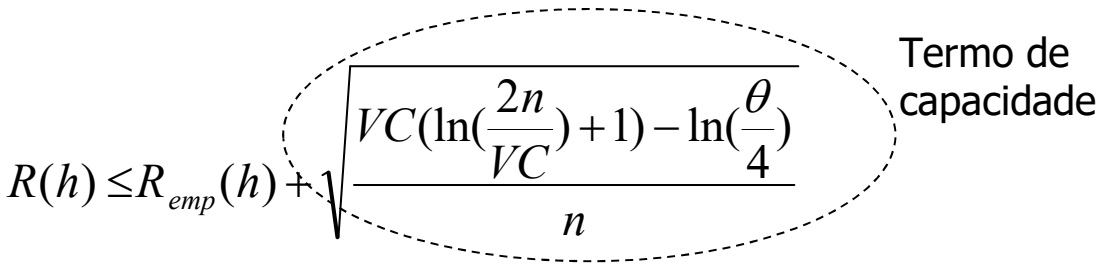


Minimização de risco empírico

- Hipótese $\hat{h}$ pode levar a uma boa estimativa da hipótese verdadeira
  - Mas nem sempre isso ocorre
    - Ex.: Overfitting
- Teoria do aprendizado estatístico provê limites para o risco esperado (funcional)
  - Que podem ser utilizados na escolha de uma hipótese com melhor generalização

Limites no risco esperado

- Limite estabelecido pela TAE para SVMs



  - VC: dimensão Vapnik-Chervonenkis da classe de funções H ($\hat{h} \in H$)
    - Mede a capacidade do conjunto de funções H
  - n: número de exemplos de treinamento
  - Garantido com probabilidade 1- θ ($\theta \in [0,1]$)

Dimensão VC

- Mede a capacidade de um conjunto de funções H
  - Quanto maior seu valor, maior a complexidade das funções que podem ser induzidas
    - Maior chance de *overfitting*
  - Limite estabelecido pela TAE que define o princípio de indução chamado minimização do risco estrutural
    - Busca $\hat{h}$ de menor complexidade possível com baixo erro para os dados de treinamento

Problemas

- Não é fácil computar dimensão VC de uma classe de funções
  - Valor pode ser desconhecido ou infinito
  - Existem resultados alternativos para funções de decisão lineares ($h(x) = w.x$)
    - Vetor *w* é o vetor normal a *h* (*w* é perpendicular ao hiperplano de separação)
    - Relacionam o risco estrutural ao conceito de margens de exemplos

Conceito de margens

- Margem de um exemplo:
  - Relacionada à sua distância à fronteira de decisão induzida no processo de aprendizado
  - Medida de confiança da previsão de um classificador
- Risco (erro) marginal R_ρ
  - Proporção de exemplos de treinamento com margem de confiança inferior a uma constante $\rho > 0$

Risco marginal

$$R_\rho(h) = \frac{1}{n}\sum_{i=1}^{n} I(y_i h(x_i) < \rho)$$

- Onde
  - I(q) = 1 (0) se q for verdadeiro (falso)

Limites no risco esperado

$$R(h) \leq R_{\rho}(h) + \sqrt{\frac{c}{n}\left(\frac{r^2}{\rho^2}\log^2\left(\frac{n}{\rho}\right) + \log\left(\frac{1}{\theta}\right)\right)}$$

Termo de capacidade

- *Onde:*
  - *r* : raio de uma esfera que engloba as funções de H
  - *c* : constante que determina a influência do limite (termo de capacidade)

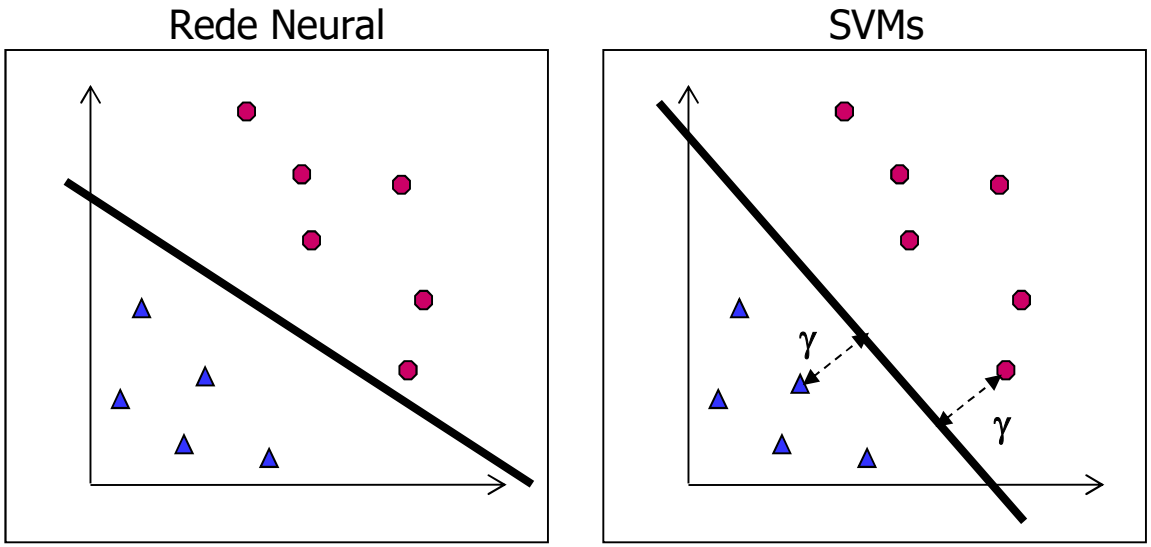
Limites no risco esperado

- Valor de ρ influencia generalização
  - Define a margem
  - Alto, leva a aumento do erro marginal
    - *Undefitting*
  - Baixo, reduz erro marginal, mas aumenta termo de capacidade (complexidade)
    - *Overfitting*
- Objetivo: encontrar hiperplano com margem ρ alta, mas com poucos erros marginais

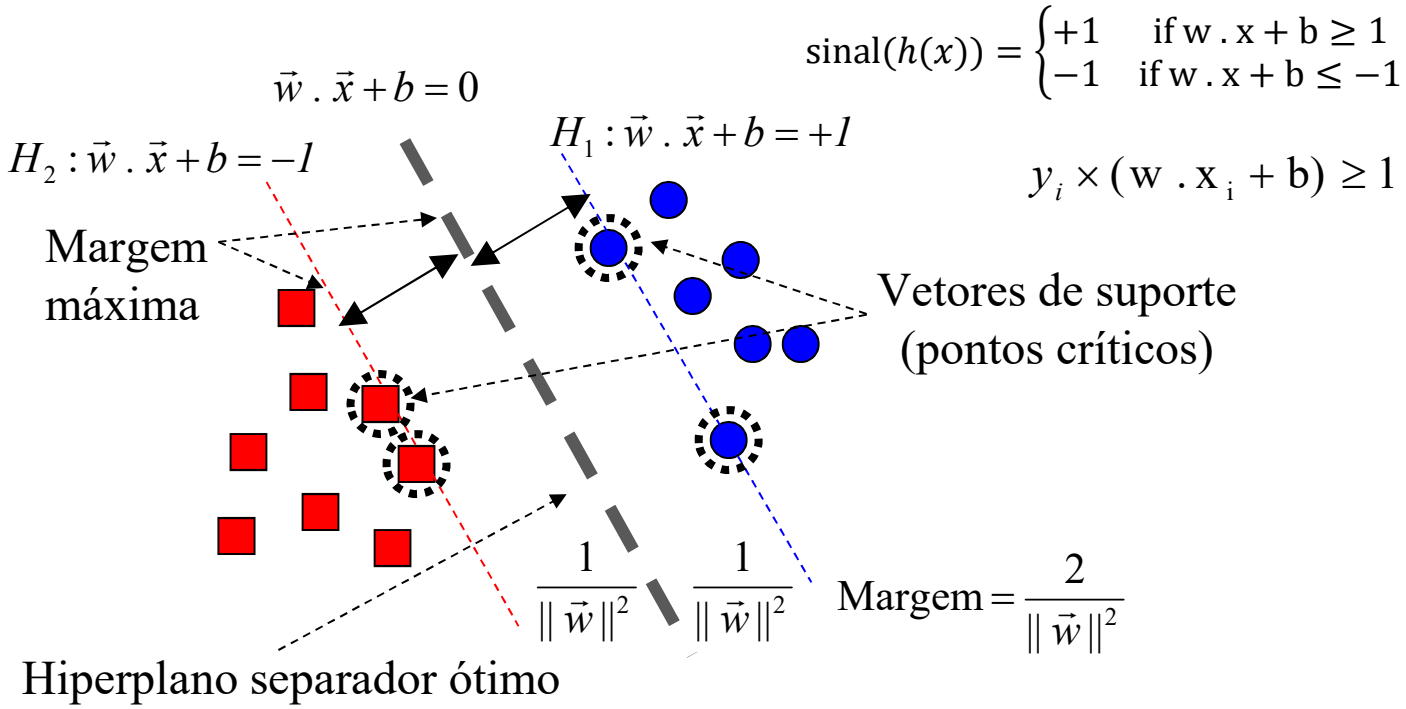
Limites no risco esperado

- Isso é feito pelas máquinas de vetores de suporte (SVMs)
- Estratégia básica
    - Encontrar um hiperplano que maximize margem de separação (margem larga)
        - Distância da fronteira de decisão a um conjunto de “vetores de suporte”
        - Com erro marginal baixo
            - Número mínimo de objetos entre as margens

Máquinas de Vetores de Suporte (SVMs)



Máquinas de Vetores de Suporte (SVMs)



$$\text{sinal}(h(x)) = \begin{cases} +1 & \text{if } w \cdot x + b \geq 1 \\ -1 & \text{if } w \cdot x + b \leq -1 \end{cases}$$

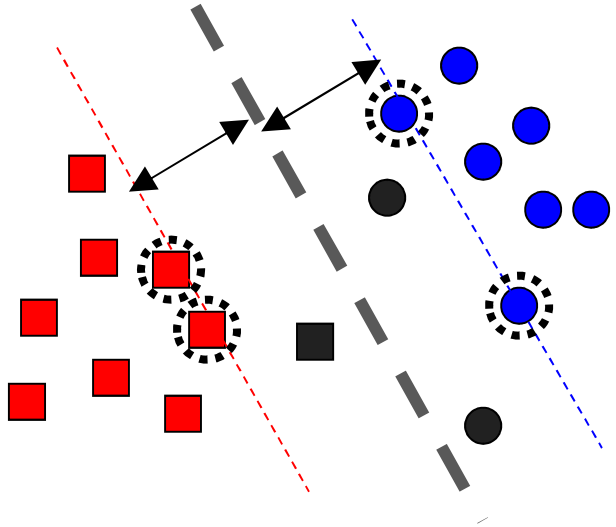
$$y_i \times (w \cdot x_i + b) \geq 1$$

Margens suaves

- Não permitir exemplos entre as margens reduz tamanho da margem
  - Reduz generalização
- SVMs podem ser estendidas para tolerar exemplos dentro das margens
  - Relaxamento de restrições impostas ao problema de otimização
    - Introdução de variáveis de folga

Variáveis de folga

- Slack variables



Linearmente separáveis

- SVMs apresentam bons desempenhos para problemas linearmente separáveis
    - Não conseguem lidar com problemas não linearmente separáveis
- Alguns conjuntos de dados exigem fronteiras mais complexas que lineares
    - Para isso foram propostas alterações baseadas no teorema de Cover

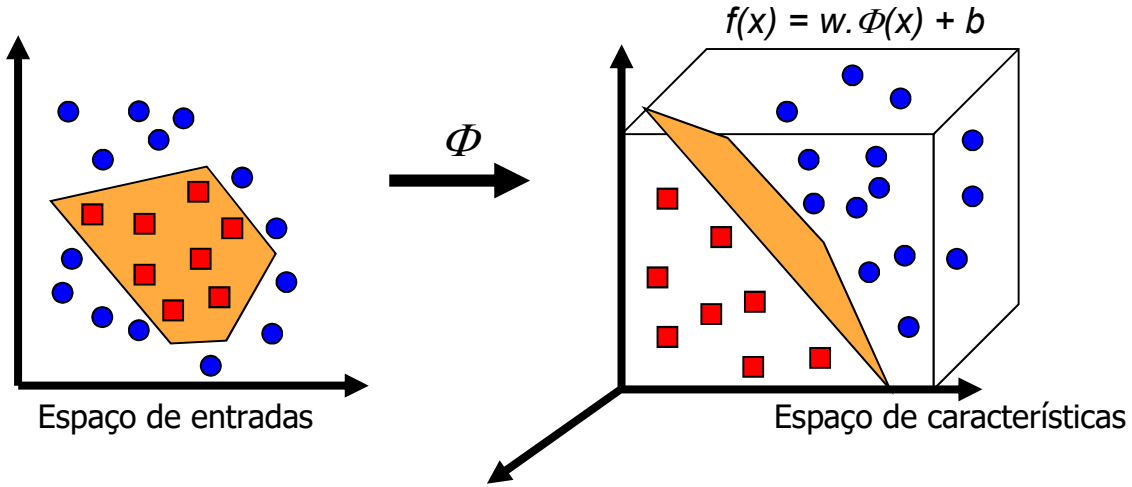
Teorema de Cover

Conjunto de dados não linearmente separáveis em um espaço pode ser transformado para outro espaço em que, com alta probabilidade, se tornam linearmente separáveis

- Condições:
  - Transformação seja não linear
  - Dimensão do novo espaço seja suficientemente alta

Problemas não linearmente separáveis

- Generalização para problemas não lineares
  - Mapeamento de dados de entrada para um espaço de maior dimensão



Exemplo

- Supor conjunto de dados X com 2 atributos preditivos
- Definir 3 pontos de localização no conjunto original
- Usar esses pontos para transformar 2 atributos originais em 3 outros atributos
    - Ex. Distância entre cada exemplo x_i e cada um dos 3 pontos de localização

Fronteiras mais complexas

- Computação da função Φ pode ter custo computacional elevado
  - Informação necessária: cálculo do produto escalar entre objetos
  - Pode simplificada usando funções *kernel (K)*
    - Rcebem 2 pontos no espaço de entradas e calculam produto escalar deles no espaço de características
    - $K(x_i, x_j) \leftrightarrow \Phi(x_i).\Phi(x_j)$

Funções *Kernel*

- Diversas
  - Gaussiana
  - Polinomial
    - Linear
  - Sigmoidal
  - Para aplicações específicas
- Seguem condições estabelecidas pelo teorema de Mercer
- Híper-parâmetros ajustáveis

Funções *Kernel*

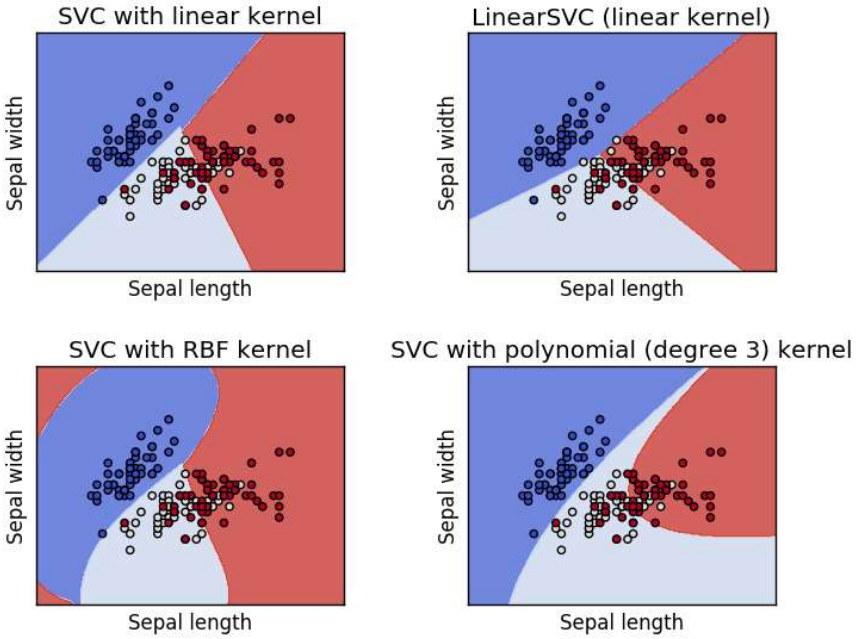
- Em geral, *K* é menos complexa que Φ
  - É comum definir-se a função *K* sem conhecer-se explicitamente Φ

Tipos de Kernel	Função $K(x_i, x_j)$ correspondente
Polinomial	$(x_i^T.x_j + 1)^p$ $(p = 1, linear)$
Gaussiano	$exp(-1/(2\sigma^2)\ ||x_i - x_j||^2)$
Sigmoidal	$tanh(\beta_0 x_i.x_j + \beta_1)$

Funções *Kernel*

- Mede similaridade entre objetos
- Kernel linear:
    - Indicado quando #atributos > #objetos
    - Processamento mais rápido
- Kernel Gaussiano
    - Indicado quando #objetos > #atributos
- Kernels específicos são propostos para algumas aplicações

Funções *Kernel*



Classificação multiclasses

- SVMs podem induzir apenas classificadores binários
    - Outros algoritmos de AM têm a mesma limitação
- Existe um grande número de problemas reais com mais que 2 classes
    - Necessidade de estratégias multiclasses

Estratégias multiclasses

- Duas abordagens têm sido utilizadas:
  - Algoritmo de classificação é internamente adaptado
    - Modificação de parte de suas operações internas
  - Decomposição do problema multiclasses em vários problemas binários
    - Estratégias decomposicionais

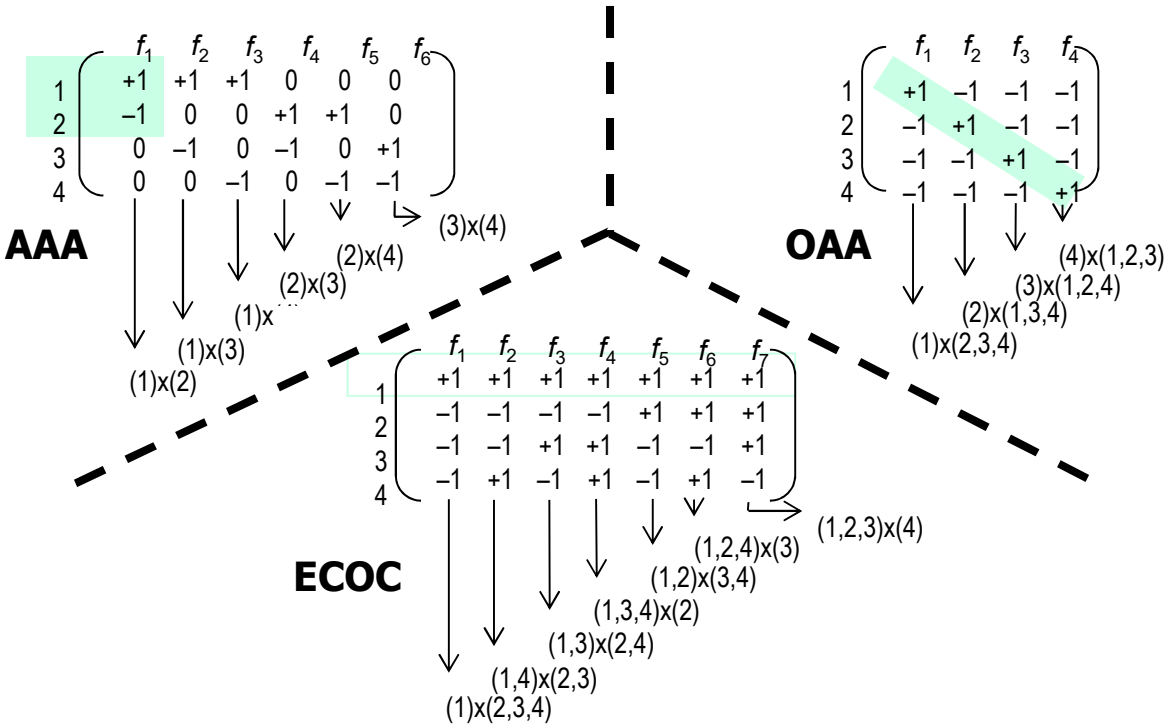
Estratégias decomposicionais

- Etapas
  - Decomposição da tarefa
  - Reconstrução
- Decomposição
  - Geralmente reduz a complexidade da tarefa
  - Permite processamento paralelo
  - Alternativas:
    - Matrizes de códigos (MC)
    - Hierarquias de classificadores

Matrizes de códigos

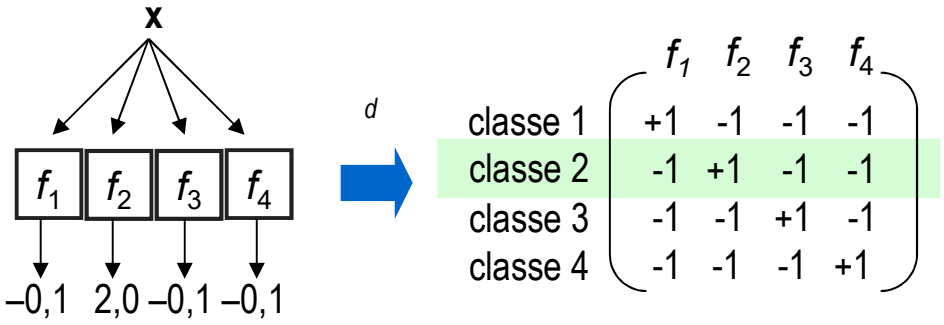
- Um-contra-todos (OAA)
  - Um classificador para cada classe
    - k classificadores para k classes
- Todos contra todos (AAA)
  - Um classificador para cada par de classes
    - k(k-1)/2 classificadores para k classes
- *Error Correcting Output Codes* (ECOC)
  - Um código de correção de erro representando cada classe

Matrizes de códigos



Matrizes de códigos

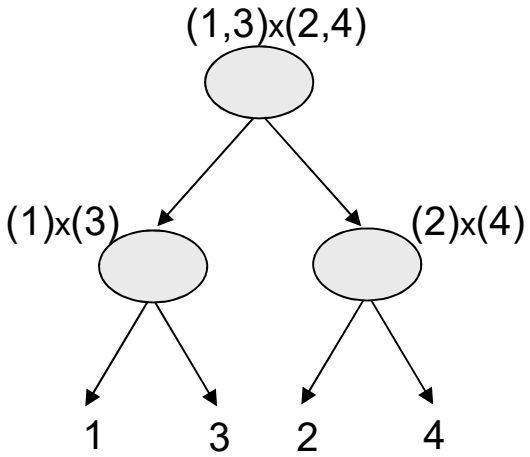
- Reconstrução = decodificação



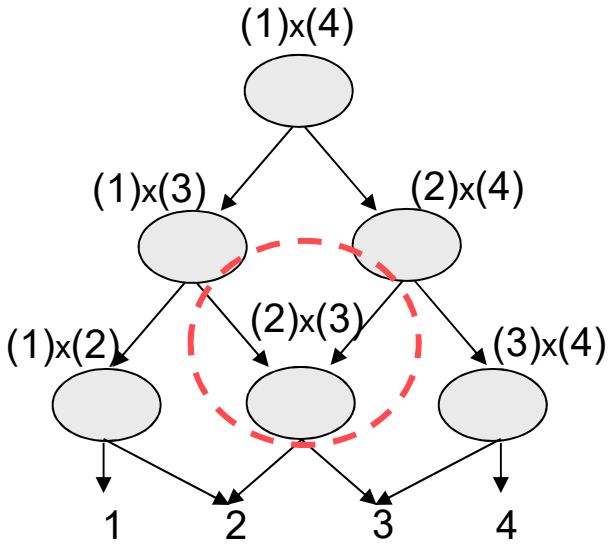
- Função de decodificação *d*
  - *Hamming*
  - Baseada em margens

Estratégias Hierárquicas

- Organizam os preditores hierarquicamente



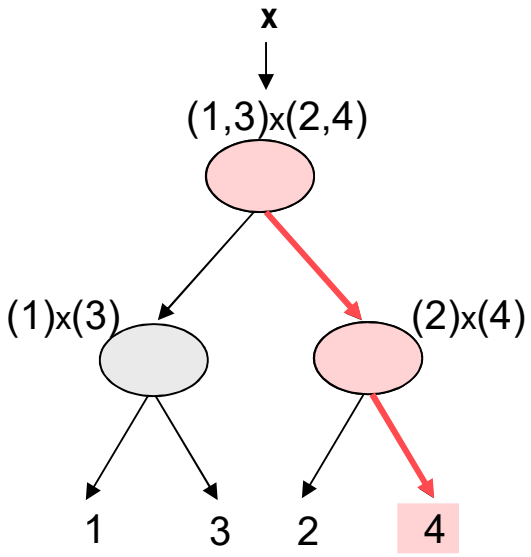
árvore



grafo direcionado acíclico

Estratégias Hierárquicas

- Reconstrução



Conclusão

- Teoria de Aprendizado Estatístico
- SVMs
- Problemas não linearmente separáveis
- Classificação binária e multiclasses
- Regressão

Fim do apresentação

Exercício

- Utilizando do repositório UCI as bases de dados IRIS e GLASS
  - Investigar SVMs
    - Três kernels diferentes
  - Particionar os dados como no exercício de métodos probabilísticos
  - Ajustar parâmetros por tentativa e erro
  - Comparar com resultados do kNN, com os dos métodos probabilísticos e das redes neurais