Aprendizado de Máquina

Máquinas de vetores de suporte

André C. P. L. F. de Carvalho
Posdoutorando: Isvani Frias-Blanco
ICMC-USP

Principais 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 exemplos
  - Função (classificador ou regressor)
- TAE estabelece princípios para induzir função com boa 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)*

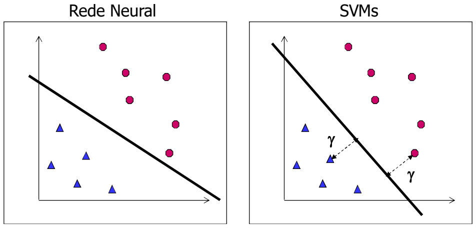
TAE

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

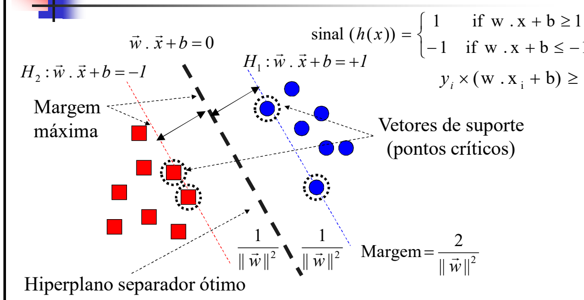
Limites no risco esperado

- 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
      - O mínimo de objetos entre as margens

Máquinas de Vetores de Suporte (SVMs)



SVMs

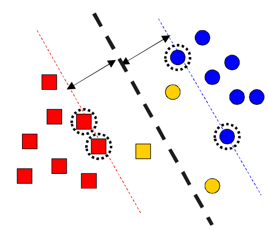


Margens suaves

- 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

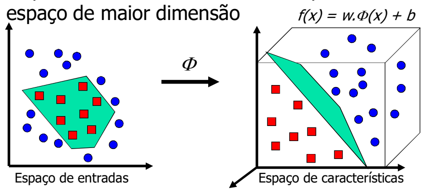
Teorema de Cover

Conjunto de dados não linearmente separáveis em um espaço podem ser transformados 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 lineares

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



Exemplo

- Supor conjunto de dados com dois atributos preditivos
- Definir 3 pontos de localização no conjunto original
- Usar esses pontos para transformar 2 atributos originais em 3 novos atributos
  - Ex. distância entre cada exemplo *x* 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 ser feito por funções *kernel (K)*
    - Função *kernel* recebe dois pontos no espaço de entradas e calcula produto escalar deles no espaço de características
    - $K(x_i, x_j) = \Phi(x_i).\Phi(x_j)$

Funções *Kernel*

- Variações
  - Gaussiana
  - Polinomial
    - Linear
  - Sigmoidal
  - Para aplicações específicas
- Segue condições estabelecidas pelo teorema de Mercer
- Parâmetros ajustáveis

Funções *Kernel*

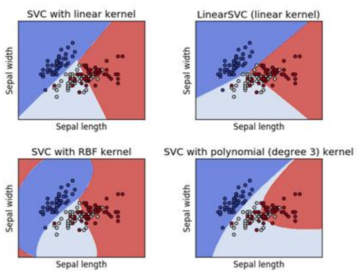
- Em geral, *K* é menos complexa que Φ
  - É comum definir-se a função *K* sem conhecer-se explicitamente Φ

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

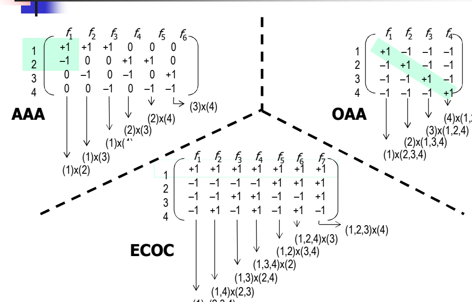
Estratégias decomposicionais

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

Estratégias Baseadas em MC

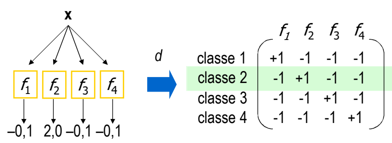
- Um-contra-todos (OAA)
  - Um classificador para cada classe
    - k classificadores para k classes
- Todos contra todos (AAA)
  - Um classificador para cada par de classes
    - k(k-1)/2 classificadores para k classes
- *Error Correcting Output Codes* (ECOC)
  - Um código de correção de erro representando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

Matrizes de códigos



Matrizes de códigos

- Reconstrução = decodificação



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

Conclusão

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

Perguntas

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