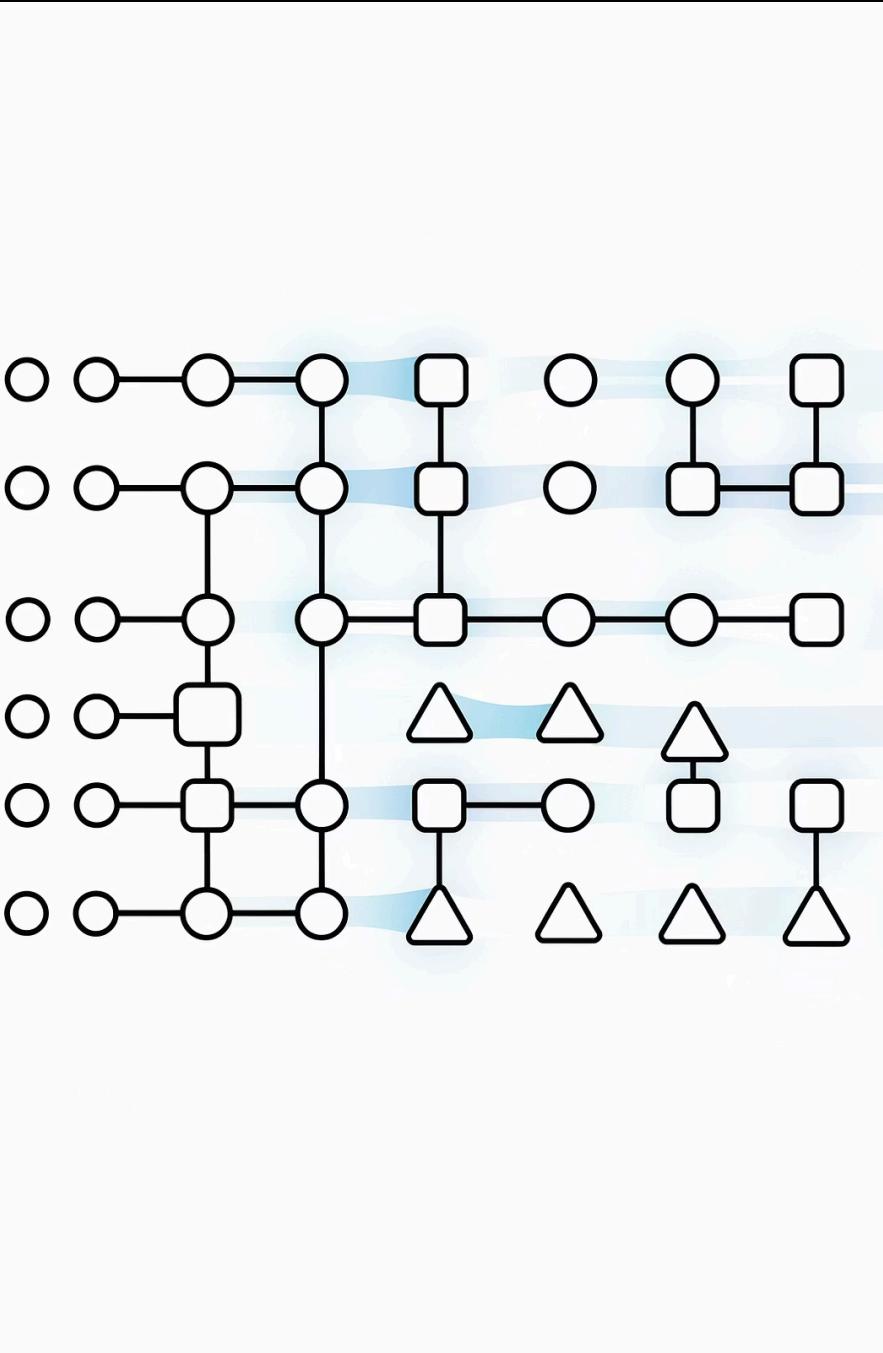


Classificação – Conceitos Avançados

Uma exploração abrangente dos conceitos fundamentais de classificação em ciência de dados e aprendizado de máquina.

Eduardo Ogasawara

eduardo.ogasawara@cefet-rj.br
<https://eic.cefet-rj.br/~eogasawara>



Classificação com Redes Neurais

Classificação por Redes Neurais

As redes neurais artificiais são inspiradas pelos sistemas neurais biológicos, oferecendo uma abordagem poderosa para problemas de classificação complexos. Esses modelos computacionais são capazes de aprender e modelar relacionamentos não-lineares intrincados entre variáveis, tornando-os ideais para tarefas onde métodos tradicionais encontram limitações.



Inspiração Biológica

Modelam o comportamento de neurônios reais através de unidades computacionais interconectadas



Relacionamentos Não-Lineares

Capturam padrões complexos impossíveis para classificadores lineares



Aprendizado por Backpropagation

Algoritmo iterativo que ajusta pesos para minimizar erros de predição



Base do Deep Learning

Arquiteturas modernas evoluíram dessas fundações matemáticas

Desenvolvidas inicialmente por psicólogos e neurobiologistas buscando criar análogos computacionais de neurônios, as redes neurais consistem em conjuntos de unidades de entrada/saída conectadas. Cada conexão possui um peso associado que é ajustado durante a fase de aprendizado, permitindo que a rede preveja corretamente o rótulo de classe das tuplas de entrada. Essa abordagem também é conhecida como aprendizado conexionista devido às conexões entre as unidades.

Neurônio: Unidade da Camada Oculta/Saída

Estrutura Matemática

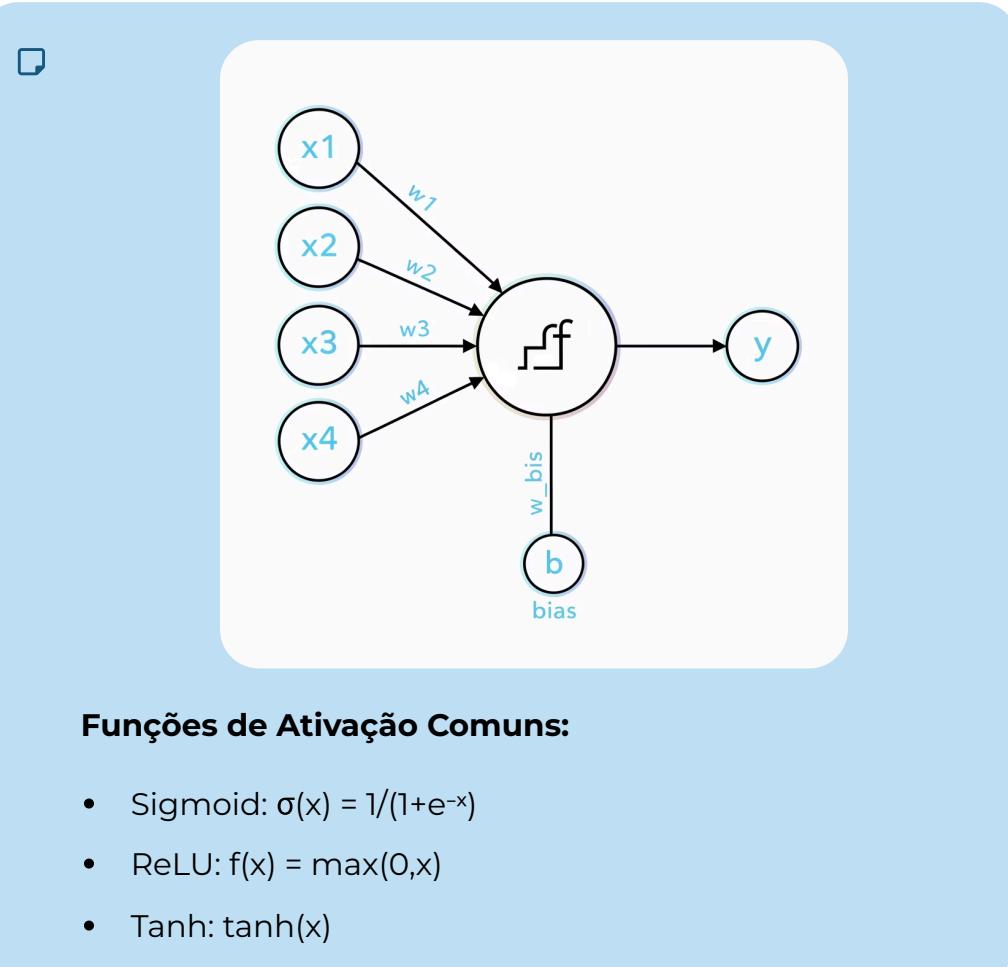
Um neurônio artificial funciona mapeando um vetor de entrada n-dimensional \mathbf{x} em uma variável de saída \mathbf{y} através de duas operações fundamentais:

1. **Produto Escalar Ponderado:** As entradas são multiplicadas por seus pesos correspondentes e somadas
2. **Função de Ativação Não-Linear:** O resultado é transformado por uma função que introduz não-linearidade

As entradas para cada unidade são as saídas da camada anterior. Essas entradas são multiplicadas pelos pesos correspondentes para formar uma soma ponderada, que é então adicionada ao **bias** associado à unidade. Finalmente, uma função de ativação não-linear é aplicada a esse valor acumulado.

Importância do Bias

O termo de bias permite que o neurônio desloque a função de ativação, aumentando a flexibilidade do modelo para se ajustar aos dados de treinamento.



Como Funciona uma Rede Neural Multicamadas

Uma rede neural multicamadas processa informações através de uma arquitetura em camadas, transformando gradualmente as características de entrada em previsões finais. Vamos explorar esse processo em detalhes:



Camada de Entrada

As entradas correspondem aos atributos medidos para cada tupla de treinamento e são alimentadas simultaneamente

Camadas Ocultas

Os dados são ponderados e processados através de uma ou mais camadas ocultas que extraem características

Camada de Saída

As saídas ponderadas da última camada oculta geram a previsão final da rede

Características Arquiteturais

- **Feed-Forward:** A rede é alimentada para frente - nenhum peso retorna para uma unidade de entrada ou para uma unidade de saída de camada anterior
- **Número de Camadas Ocultas:** Embora arbitrário, geralmente apenas uma camada oculta é utilizada para problemas moderadamente complexos
- **Capacidade de Aproximação:** Do ponto de vista estatístico, as redes realizam regressão não-linear

Teorema da Aproximação Universal: Com unidades ocultas suficientes e amostras de treinamento adequadas, redes neurais podem aproximar qualquer função contínua com precisão arbitrária.

Rede Neural Feed-Forward Multicamadas

Vetor de Entrada (X)

Representa os atributos dos dados de entrada, com cada dimensão alimentando um neurônio da camada de entrada.

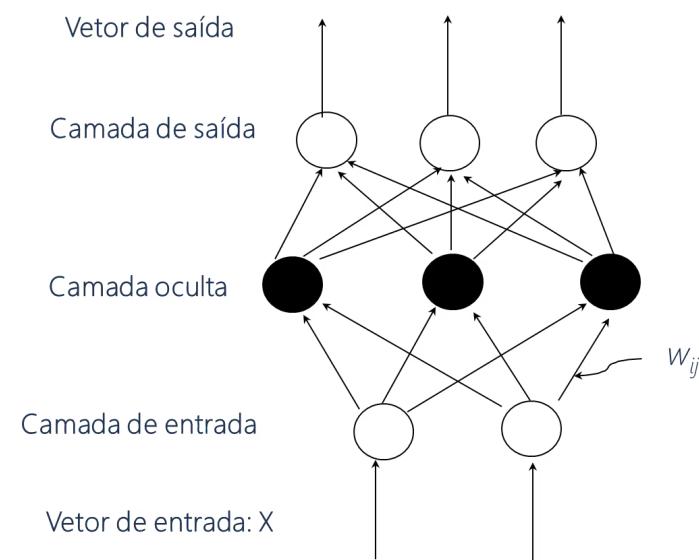
Pesos (w_{ij})

Parâmetros ajustáveis que determinam a força das conexões entre neurônios de camadas adjacentes

Vetor de Saída

Produz as previsões finais da rede, com formato dependendo do tipo de problema (classificação ou regressão)

Esta arquitetura multicamadas ilustra o fluxo de informação através da rede. A **camada de entrada** recebe os dados brutos, as **camadas ocultas** extraem representações cada vez mais abstratas, e a **camada de saída** gera a previsão final. Os pesos **w_{ij}** entre as camadas são otimizados durante o treinamento através do algoritmo de backpropagation.



Definindo a Topologia da Rede

A configuração da topologia da rede neural é uma etapa crítica que influencia diretamente o desempenho do modelo. Decisões arquiteturais inadequadas podem resultar em underfitting ou overfitting dos dados.

01

Especificar a Arquitetura

Defina o número de unidades na camada de entrada, o número de camadas ocultas, unidades em cada camada oculta, e unidades na camada de saída

02

Normalizar Valores de Entrada

Normalize os valores de entrada para cada atributo medido nas tuplas de treinamento para o intervalo [0.0—1.0] para estabilizar o treinamento

03

Configurar Unidades de Entrada

Crie uma unidade de entrada por valor de domínio, cada uma inicializada em 0 para começar o processo de aprendizado

04

Definir Camada de Saída

Para classificação com mais de duas classes, use uma unidade de saída por classe; para regressão, ajuste conforme necessário

05

Iterar e Refinar

Se a precisão for inaceitável após o treinamento, repita com topologia diferente ou conjunto diferente de pesos iniciais

- Dica Prática:** Comece com uma arquitetura simples e aumente gradualmente a complexidade. A validação cruzada ajuda a encontrar a topologia ideal sem overfitting.

Redes Neurais como Classificadores

Fraquezas

Tempo de Treinamento Longo

O processo iterativo de ajuste de pesos pode demandar recursos computacionais significativos, especialmente para redes profundas

Parâmetros Empíricos

Topologia da rede e hiperparâmetros geralmente requerem determinação experimental através de tentativa e erro

Baixa Interpretabilidade

Difícil interpretar o significado simbólico dos pesos aprendidos e das unidades ocultas, funcionando como "caixa-preta"

Forças

Alta Tolerância a Ruído

Robustez excepcional em dados com ruído e imperfeições

Generalização

Capacidade de classificar padrões não vistos durante o treinamento

Valores Contínuos

Adequadas para entradas e saídas com valores contínuos

Sucesso Comprovado

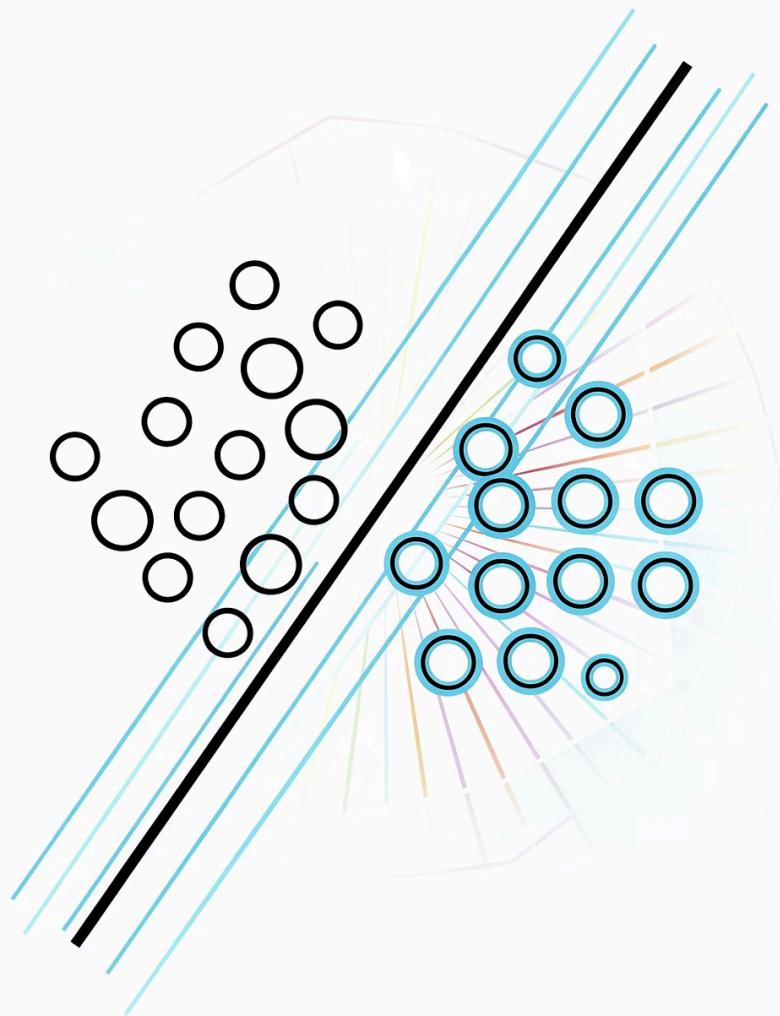
Excelente desempenho em dados reais como reconhecimento de caracteres manuscritos

Paralelização

Algoritmos inherentemente paralelizáveis para processamento eficiente

Extração de Regras

Técnicas recentes permitem extrair regras de redes treinadas



Classificação com Support Vector Machines

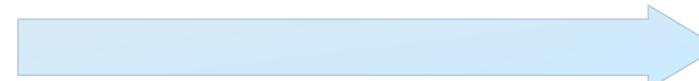
SVM – Support Vector Machines

Support Vector Machines (SVMs) representam um método de classificação relativamente novo, porém extremamente poderoso, capaz de lidar tanto com dados lineares quanto não-lineares. A abordagem fundamental das SVMs é transformar o problema de classificação em um espaço de maior dimensionalidade onde a separação linear se torna possível.



Mapeamento Não-Linear

Utiliza um mapeamento não-linear para transformar os dados de treinamento originais em uma dimensão superior



Busca pelo Hiperplano Ótimo

Na nova dimensão, busca o hiperplano de separação linear ótimo que serve como "fronteira de decisão"



Separação Garantida

Com mapeamento não-linear apropriado para dimensão suficientemente alta, dados de duas classes sempre podem ser separados por um hiperplano

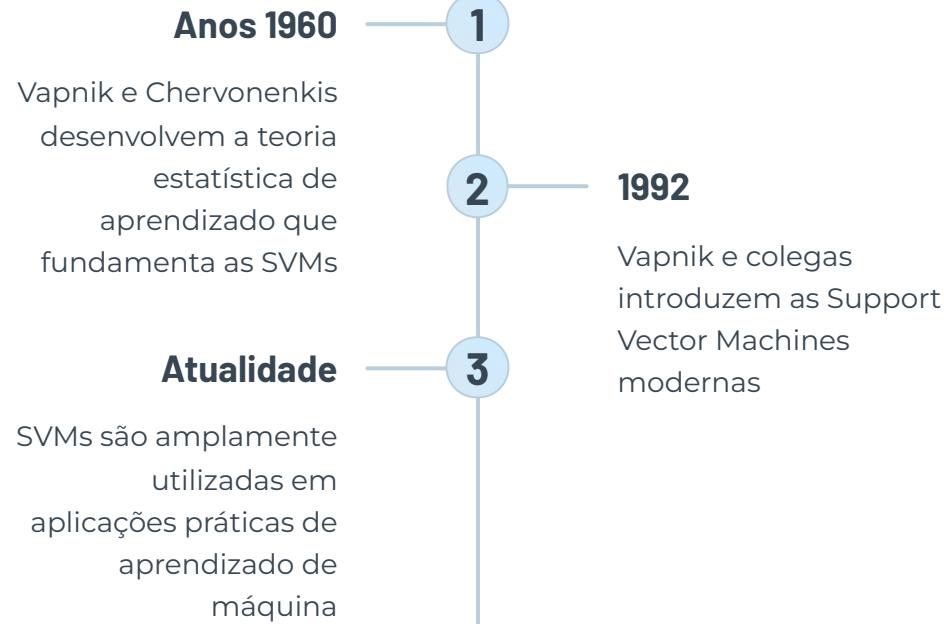
Características Fundamentais

- **Vetores de Suporte:** Tuplas de treinamento "essenciais" que definem a fronteira de decisão
- **Margens:** Distância entre o hiperplano e os vetores de suporte mais próximos, que é maximizada
- **Efetividade em Altas Dimensões:** Mantém bom desempenho mesmo quando o número de características é muito grande

Importante: O treinamento pode ser computacionalmente custoso para grandes conjuntos de dados. Implementações modernas frequentemente utilizam aproximações ou variantes lineares de SVM para dados em larga escala.

SVM – História e Aplicações

Origens e Desenvolvimento



Características e Aplicações

Desempenho

O treinamento pode ser lento, mas a precisão é elevada devido à capacidade de modelar fronteiras de decisão não-lineares complexas através da **maximização de margem**.

Versatilidade

Utilizadas tanto para **classificação** quanto para **predição numérica** (regressão).

Domínios de Aplicação

- Reconhecimento de padrões em imagens
- Classificação de texto e categorização de documentos
- Bioinformática e análise genômica
- Detecção de fraudes financeiras
- Diagnóstico médico assistido

SVM – Filosofia Geral

A filosofia central das Support Vector Machines é encontrar o hiperplano de separação que **maximiza a margem** entre as classes. Diferentemente de outros classificadores que simplesmente buscam qualquer fronteira de decisão, as SVMs buscam a fronteira "ótima" que oferece a maior distância possível entre as classes.

Conceito de Margem Máxima

A margem é a distância perpendicular entre o hiperplano de decisão e os pontos de dados mais próximos de cada classe. Ao maximizar essa margem, a SVM cria um "corredor" de separação o mais amplo possível, o que teoricamente resulta em melhor generalização para novos dados.

Os pontos de dados que se encontram exatamente sobre as bordas desse corredor são chamados de **vetores de suporte** - eles são os únicos pontos que realmente influenciam a posição do hiperplano de decisão.

Vantagens da Abordagem

- **Fundamentação Matemática Rigorosa:** Baseada em teoria de otimização convexa com garantias teóricas
- **Boa Generalização:** A maximização da margem reduz o risco de overfitting
- **Eficiência:** Apenas os vetores de suporte são necessários para definir o classificador final
- **Extensibilidade:** O uso de kernels permite lidar com dados não-linearmente separáveis

SVM – Quando os Dados São Linearmente Separáveis

No caso mais simples, quando os dados são linearmente separáveis, a SVM encontra o hiperplano ótimo que divide perfeitamente as duas classes enquanto maximiza a margem entre elas.

Hiperplano de Separação

Uma fronteira linear (reta em 2D, plano em 3D, hiperplano em n-dimensões) que divide o espaço de características em duas regiões distintas

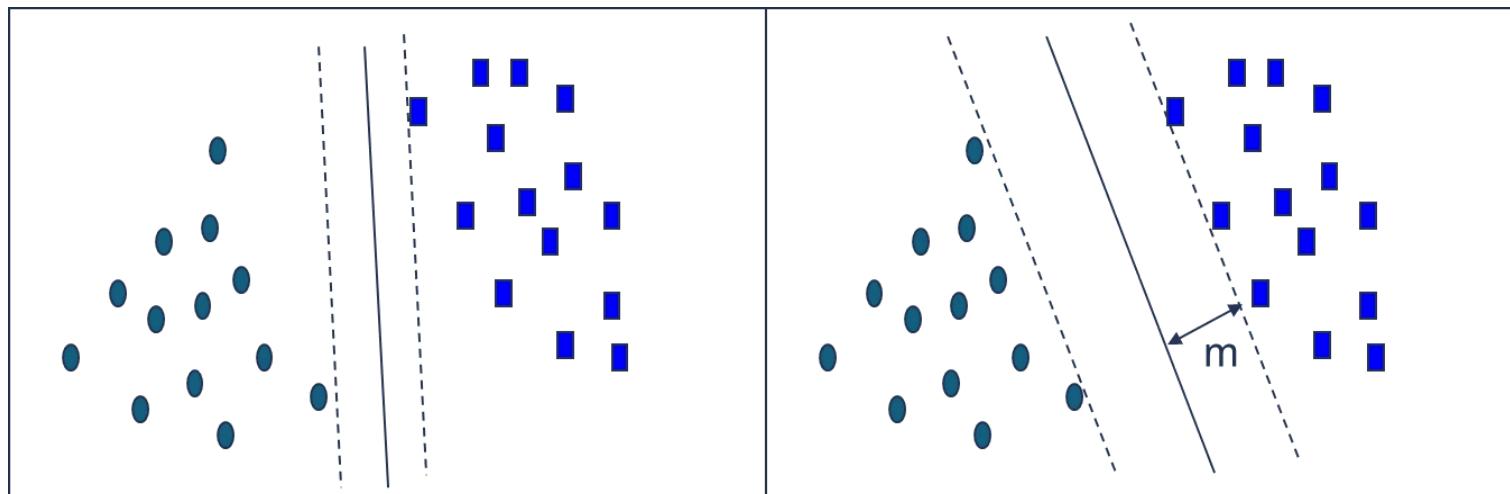
Vetores de Suporte

Os pontos de dados das duas classes que estão mais próximos do hiperplano de decisão e que definem as margens

Margem Máxima

A distância perpendicular entre o hiperplano e os vetores de suporte mais próximos, que é maximizada durante o treinamento

No cenário de separabilidade linear, existe infinitos hiperplanos capazes de separar as classes. A SVM escolhe aquele com a **maior margem**, pois teoricamente oferece melhor capacidade de generalização. Matematicamente, isso é formulado como um problema de otimização convexa que garante encontrar a solução global ótima.



SVM – Linearmente Separável

Formulação Matemática

Para dados linearmente separáveis, o hiperplano de decisão pode ser expresso como:

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

Onde \mathbf{w} é o vetor de pesos (normal ao hiperplano) e b é o bias. Os vetores de suporte satisfazem:

$$|\mathbf{w} \cdot \mathbf{x} + b| = 1$$

A margem total é então $2/\|\mathbf{w}\|$, e o objetivo é minimizar $\|\mathbf{w}\|^2$ sujeito às restrições de classificação correta.

Propriedades Importantes

- **Solução Única:** Para dados linearmente separáveis, existe exatamente um hiperplano de margem máxima
- **Dependência Apenas dos Vetores de Suporte:** Pontos distantes da margem não influenciam a solução
- **Robustez:** Pequenas perturbações nos dados que não afetam os vetores de suporte não alteram o classificador

□ Na prática, dados perfeitamente linearmente separáveis são raros. Por isso, as SVMs utilizam técnicas como **soft margins** e **kernels** para lidar com casos mais realistas.

Por Que SVM É Efetiva em Dados de Alta Dimensionalidade?

Uma das características mais notáveis das Support Vector Machines é sua efetividade em espaços de alta dimensionalidade, desafiando a intuição de que mais dimensões necessariamente aumentam a complexidade do modelo.

1 Complexidade Baseada em Vetores de Suporte

A complexidade do classificador treinado é caracterizada pelo **número de vetores de suporte**, não pela dimensionalidade dos dados. Isso significa que um modelo pode ser simples mesmo em espaços de milhares de dimensões.

2 Vetores de Suporte São Exemplos Críticos

Os vetores de suporte representam os exemplos de treinamento essenciais - aqueles que se encontram mais próximos à fronteira de decisão (Margem Máxima do Hiperplano). Se todos os outros exemplos de treinamento fossem removidos e o treinamento repetido, o mesmo hiperplano de separação seria encontrado.

3 Limite Superior de Erro Independente da Dimensionalidade

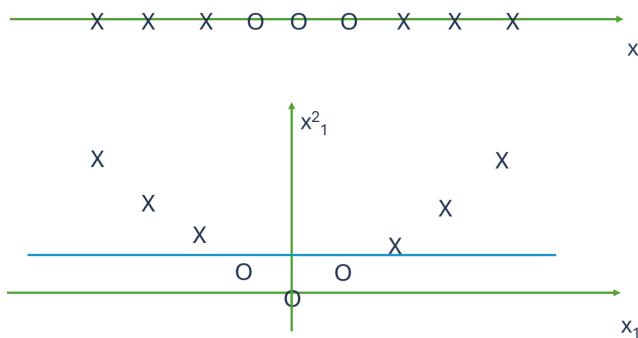
O número de vetores de suporte pode ser usado para calcular um limite superior na taxa de erro esperada do classificador SVM. Este limite é **independente da dimensionalidade dos dados**, fornecendo garantias teóricas sólidas.

4 Boa Generalização em Alta Dimensionalidade

Uma SVM com um número pequeno de vetores de suporte pode ter excelente generalização, mesmo quando a dimensionalidade dos dados é muito alta. Isso contrasta com muitos outros métodos que sofrem com a "maldição da dimensionalidade".

Usando uma Linha para Separar Classes

Esta visualização ilustra o conceito fundamental de separação linear em um espaço bidimensional. Os pontos marcados com **X** representam uma classe, enquanto os pontos marcados com **O** representam outra classe.



Visualização Bidimensional

No espaço definido pelos eixos \mathbf{x}_1 e \mathbf{x}_2 , uma linha reta pode ser traçada para separar completamente as duas classes. Esta linha representa o hiperplano de decisão em duas dimensões.

Cada ponto no espaço pode ser classificado determinando de qual lado da linha ele se encontra. Pontos acima da linha podem ser classificados como uma classe, enquanto pontos abaixo pertencem à outra classe.

Extensão para Dimensões Superiores

O conceito se generaliza naturalmente:

- **2D:** Linha reta
- **3D:** Plano
- **nD:** Hiperplano ($n-1$)-dimensional

A matemática permanece consistente independentemente do número de dimensões, tornando as SVMs escaláveis para problemas com milhares de características.

SVM – Linearmente Inseparável

Na maioria dos problemas do mundo real, os dados não são linearmente separáveis no espaço de características original. As SVMs lidam com essa situação através de duas estratégias principais: **soft margins** (margens suaves) e o **kernel trick** (truque do kernel).

1

Soft Margins (Margens Suaves)

Permite que alguns pontos violem a margem ou mesmo sejam classificados incorretamente, introduzindo um parâmetro de penalidade **C** que controla o trade-off entre maximizar a margem e minimizar erros de classificação. Pontos que violam a margem contribuem para a função de custo.

2

Kernel Trick (Truque do Kernel)

Mapeia implicitamente os dados para um espaço de características de dimensão superior onde a separação linear se torna possível. O kernel permite calcular produtos internos neste espaço de alta dimensão sem explicitamente computar as coordenadas transformadas.

3

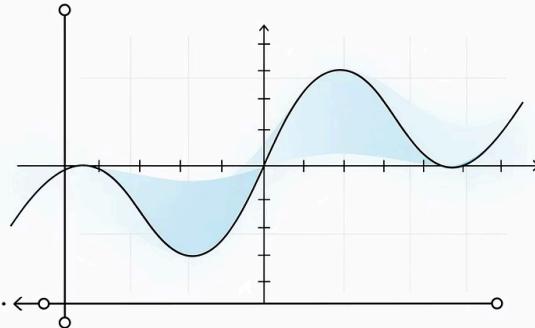
Fronteiras de Decisão Não-Lineares

Embora o hiperplano seja linear no espaço transformado, ele corresponde a uma fronteira de decisão não-linear no espaço original. Isso permite que SVMs capturem relacionamentos complexos entre características.

- **Parâmetro C:** Valores maiores de C resultam em margens mais estreitas com menos violações (risco de overfitting), enquanto valores menores permitem margens mais amplas com mais violações (melhor generalização, mas possível underfitting).

Funções Kernel para Classificação Não-Linear

As funções kernel são o coração da capacidade das SVMs de lidar com dados não-linearmente separáveis. Elas permitem operar implicitamente em espaços de alta dimensão sem o custo computacional de calcular explicitamente as coordenadas transformadas.



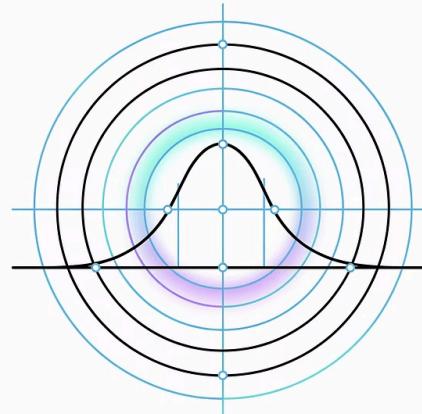
Kernel Polinomial

$$K(x, y) = (x \cdot y + c)^d$$

Mapeia dados para um espaço de dimensão polinomial. O grau d controla a complexidade. Útil quando há interações polinomiais entre características.

Escolhendo o Kernel Apropriado

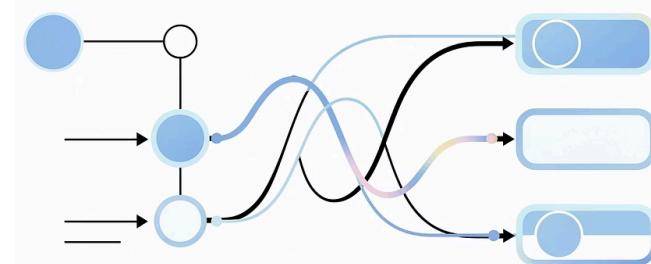
- Linear:** Para dados linearmente separáveis ou de alta dimensão
- RBF:** Boa escolha padrão quando pouco se sabe sobre os dados
- Polinomial:** Quando se suspeita de interações polinomiais específicas



Kernel RBF (Gaussiano)

$$K(x, y) = \exp(-\gamma \|x - y\|^2)$$

O kernel mais popular. Mapeia para espaço infinito-dimensional. O parâmetro γ controla a influência de cada exemplo de treinamento.



Kernel Sígnioide

$$K(x, y) = \tanh(\alpha x \cdot y + c)$$

Similar a redes neurais com uma camada oculta. Nem sempre produz um kernel válido, mas pode ser efetivo em certos domínios.

Validação Cruzada: A escolha do kernel e seus parâmetros deve ser feita através de validação cruzada para evitar overfitting e garantir boa generalização.

SVM vs. Rede Neural

Support Vector Machine

Algoritmo Determinístico

Sempre produz a mesma solução para os mesmos dados, graças à formulação como problema de otimização convexa com solução única

Propriedades de Generalização

Fundamentação matemática sólida com garantias teóricas sobre generalização baseadas na teoria de aprendizado estatístico

Aprendizado em Batch

Difícil de aprender - treinamento realizado em modo batch usando técnicas de programação quadrática, que podem ser computacionalmente intensivas

Funções Complexas via Kernels

Usando kernels, pode aprender funções muito complexas mapeando implicitamente para espaços de alta dimensão

Rede Neural

Algoritmo Não-Determinístico

Diferentes inicializações de pesos podem levar a soluções distintas, pois o algoritmo pode convergir para diferentes mínimos locais

Boa Generalização Empírica

Generaliza bem na prática, mas não possui fundamentação matemática tão forte quanto SVMs para garantias de generalização

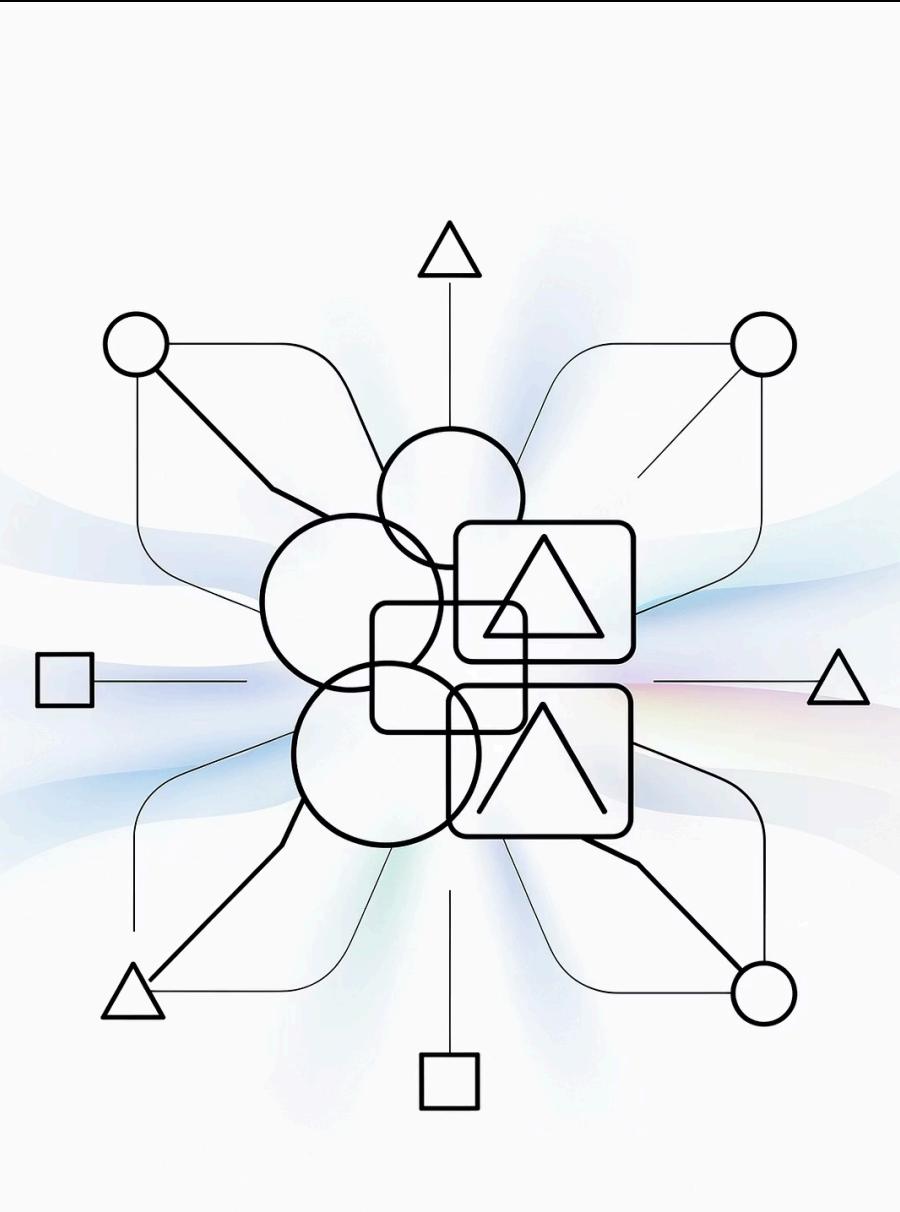
Aprendizado Incremental

Pode ser facilmente aprendida de forma incremental (online), processando exemplos um de cada vez ou em mini-batches

Funções Complexas via Profundidade

Para aprender funções complexas, use perceptron multicamadas (MLP), que é não-trivial de configurar corretamente (número de camadas, neurônios, etc.)

Ambas as abordagens têm seus méritos e a escolha entre SVM e Redes Neurais depende de fatores como tamanho do conjunto de dados, interpretabilidade desejada, recursos computacionais disponíveis, e se o aprendizado incremental é necessário. SVMs tendem a ser preferidas para conjuntos de dados menores com boa fundamentação teórica, enquanto redes neurais (especialmente deep learning) dominam em problemas de larga escala com dados abundantes.



Classificação – Ensembles

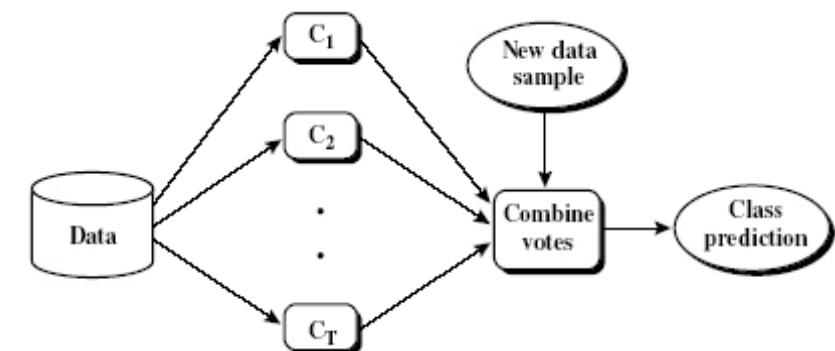
Explorando técnicas avançadas de aprendizado de máquina para aumentar a precisão preditiva através de métodos ensemble e seleção inteligente de atributos.

Métodos Ensemble: Aumentando a Precisão

Os métodos ensemble combinam múltiplos modelos de aprendizado para criar previsões mais robustas e precisas do que qualquer modelo individual. Essa abordagem aproveita a diversidade entre classificadores, reduzindo variância e viés simultaneamente.

A ideia fundamental é que um grupo de "especialistas fracos" pode, quando combinado adequadamente, superar o desempenho de um único "especialista forte". Cada modelo captura diferentes aspectos dos dados, e sua agregação produz resultados mais confiáveis.

Técnicas populares incluem bagging, boosting e random forests, cada uma com características distintas de agregação e construção de modelos base.



Referência: J. Han, J. Pei, and H. Tong, Data Mining: Concepts and Techniques, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022.

Bagging: Bootstrap Aggregation

01

Amostragem Bootstrap

Cria múltiplos subconjuntos de treinamento através de amostragem com reposição do conjunto de dados original. Cada subconjunto tem o mesmo tamanho do original.

02

Treinamento Paralelo

Treina um modelo independente para cada subconjunto bootstrap. Os modelos são construídos em paralelo, sem interação entre si durante o treinamento.

03

Agregação de Previsões

Combina as previsões de todos os modelos através de votação (classificação) ou média (regressão) para produzir a previsão final do ensemble.

O bagging é particularmente eficaz para reduzir a variância de modelos de alta complexidade, como árvores de decisão profundas, tornando-os mais estáveis e menos propensos ao overfitting.

Random Forest

Conceito Principal

Random Forest é um método ensemble onde cada classificador é uma árvore de decisão construída usando seleção aleatória de atributos em cada nó para determinar a divisão. Durante a classificação, cada árvore vota e a classe mais popular é retornada.

Métodos de Construção

Forest-RI

Random Input Selection: Seleciona aleatoriamente F atributos como candidatos para divisão em cada nó. Usa metodologia CART para crescer árvores até o tamanho máximo.

Forest-RC

Random Linear Combinations: Cria novos atributos que são combinações lineares dos existentes, reduzindo a correlação entre classificadores individuais.



- **Vantagens:** Insensível ao número de atributos selecionados em cada divisão e mais rápido que bagging ou boosting tradicional.

Leituras Recomendadas sobre Ensemble

Front. Comput. Sci., 2020, 14(2): 241–258
<https://doi.org/10.1007/s11704-019-8208-z>

REVIEW ARTICLE

A survey on ensemble learning

Xibin DONG¹, Zhiwen YU^{(✉)2}, Wenming CAO², Yifan SHI¹, Qianli MA¹

¹ School of Computer Science and Engineering, South China University of Technology, Guangzhou 510006, China
² Department of Computer Science, City University of Hong Kong, Hong Kong SAR 999077, China

© Higher Education Press and Springer-Verlag GmbH Germany, part of Springer Nature 2019

Abstract Despite significant successes achieved in knowledge discovery, traditional machine learning methods may fail to obtain satisfactory performances when dealing with complex data, such as imbalanced, high-dimensional, noisy data, etc. The reason behind is that it is difficult for these methods to capture multiple characteristics and underlying structure of data. In this context, it becomes an important topic in the data mining field that how to effectively construct an efficient knowledge discovery and mining model. Ensemble learning, as one research hot spot, aims to integrate data fusion, data modeling, and data mining into a unified framework. Specifically, ensemble learning firstly extracts a set of features with a variety of transformations. Based on these learned features, multiple learning algorithms are utilized to produce weak predictive results. Finally, ensemble learning fuses the informative knowledge from the above results to achieve knowledge discovery and better predictive performance via voting schemes in an adaptive way. In this paper, we review the research progress of the mainstream approaches of ensemble learning and classify them based on different characteristics. In addition, we present challenges and possible research directions for each mainstream approach of ensemble learning, and we also give an extra introduction for the combination of ensemble learning with other machine learning hot spots such as deep learning, reinforcement learning, etc.

Keywords ensemble learning, supervised ensemble classification, semi-supervised ensemble classification, clustering ensemble, semi-supervised clustering ensemble

Received June 8, 2018; accepted April 23, 2019

E-mail: shwy@scut.edu.cn; wenmingcaos2@cityu.edu.hk

Survey sobre Ensemble Learning

Dong, X., Yu, Z., Cao, W., Shi, Y., Ma, Q., (2020), "A survey on ensemble learning", *Frontiers of Computer Science*, v. 14, n. 2, p. 241–258.

Revisão abrangente dos métodos ensemble, cobrindo fundamentos teóricos, algoritmos clássicos e aplicações contemporâneas em aprendizado de máquina.

Random Forests - Artigo Seminal

Breiman, L., (2001), "Random forests", *Machine Learning*, v. 45, n. 1, p. 5–32.

O paper original que introduziu Random Forests, apresentando a teoria fundamental, algoritmos e demonstrações empíricas de sua eficácia superior.

Boosting

Boosting é uma técnica de ensemble que constrói modelos sequencialmente, onde cada novo modelo foca em corrigir os erros dos modelos anteriores. Diferentemente do bagging, os modelos não são independentes.



Modelo Inicial

Treina o primeiro classificador fraco com pesos uniformes para todas as instâncias



Ajuste de Pesos

Aumenta pesos das instâncias classificadas incorretamente, diminui pesos das corretas

Iteração

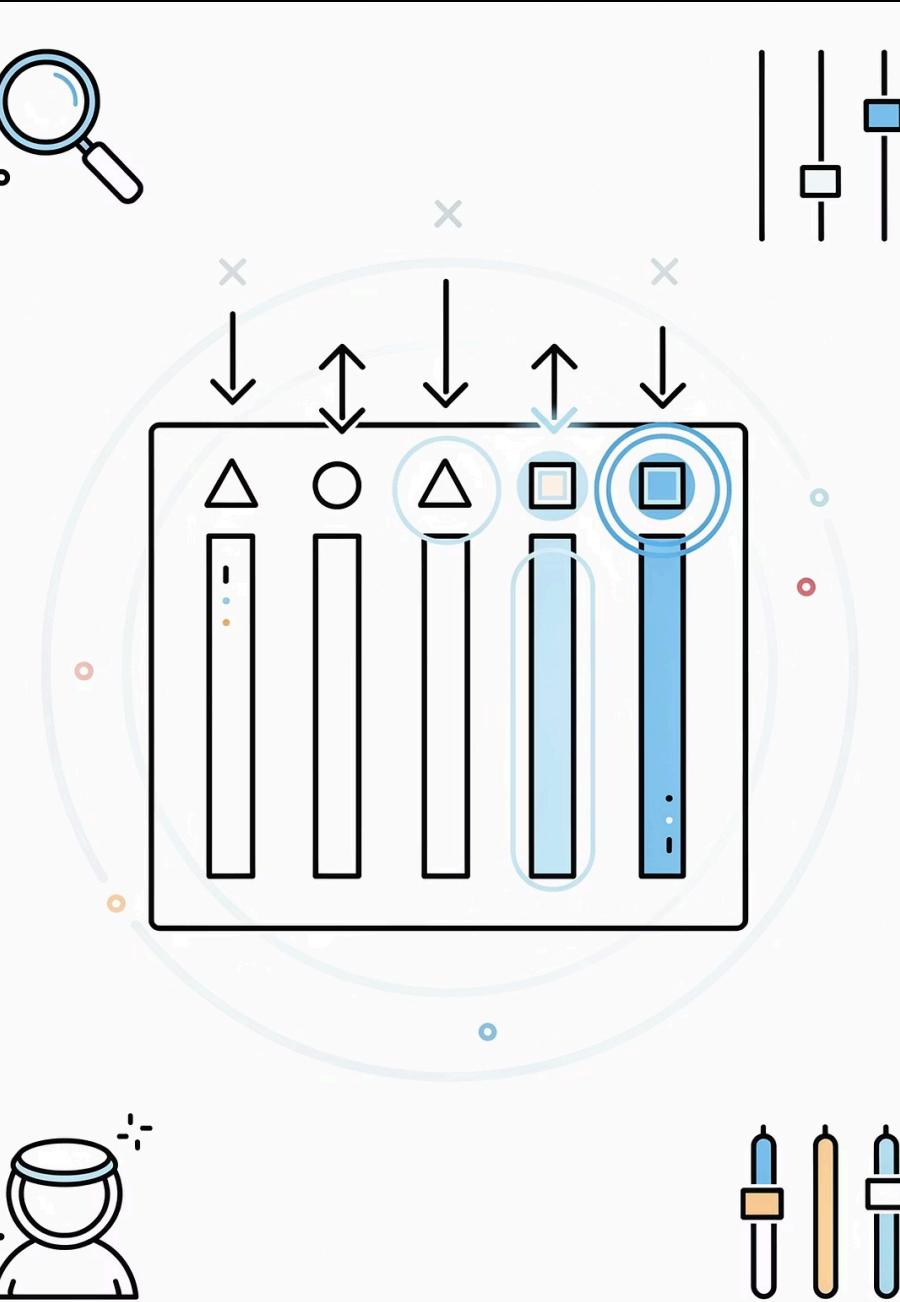
Treina novo modelo focando nas instâncias difíceis, repete o processo



Combinação Final

Agrega todos os modelos com pesos baseados em sua precisão individual

Algoritmos populares incluem AdaBoost, Gradient Boosting e XGBoost. O boosting é particularmente eficaz para reduzir viés, mas pode ser mais suscetível ao overfitting se não calibrado adequadamente.



Classificação - Seleção de Atributos

A seleção de atributos é crucial para construir modelos eficientes e interpretáveis, eliminando redundâncias e irrelevâncias que podem prejudicar o desempenho preditivo.

Maldição da Dimensionalidade

Desafios da Alta Dimensionalidade

Esparsidade dos Dados

À medida que a dimensionalidade aumenta, os dados tornam-se progressivamente mais esparsos no espaço, dificultando a identificação de padrões significativos.

Crescimento Exponencial

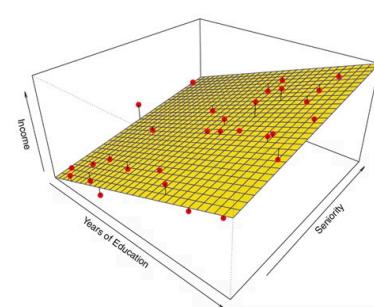
As combinações possíveis de subespaços crescem exponencialmente, tornando a exploração computacionalmente inviável.

Degradiação de Métricas

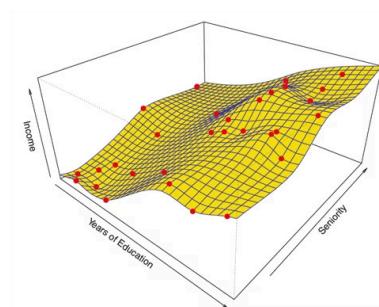
Interfere na densidade e distância entre pontos, afetando agrupamento, predição e análise de outliers de forma significativa.

Exemplo: Modelo Linear vs. Spline

Em baixas dimensões, modelos simples funcionam bem. Conforme a dimensionalidade aumenta, a flexibilidade necessária cresce drasticamente, mas os dados disponíveis se tornam insuficientes para suportar modelos complexos.



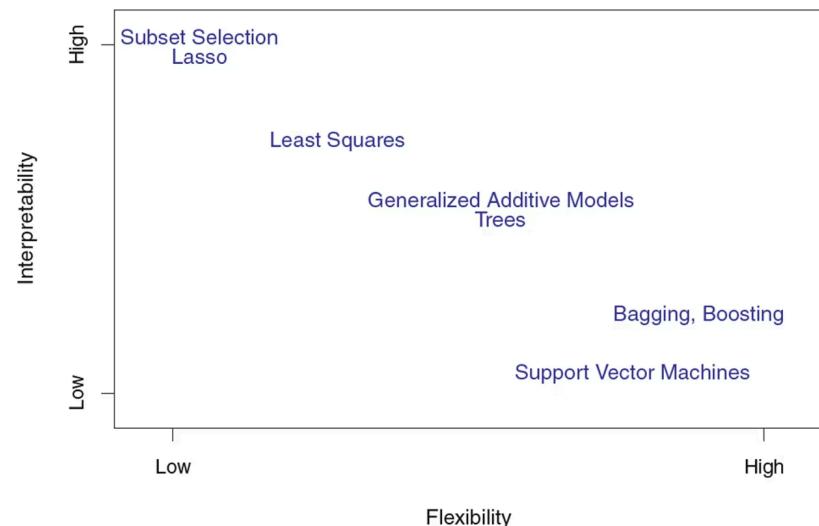
Modelo linear



Modelo spline

Interpretabilidade dos Modelos

Existe um trade-off fundamental entre flexibilidade/precisão e interpretabilidade dos modelos. Modelos mais complexos geralmente oferecem melhor desempenho preditivo, mas sacrificam a transparência e compreensão.



Alta Interpretabilidade

- Regressão linear/logística
- Árvores de decisão simples
- Modelos baseados em regras

Interpretabilidade Moderada

- GAMs (Generalized Additive Models)
- Random forests com poucas árvores
- Modelos com regularização

Baixa Interpretabilidade

- Redes neurais profundas
- Ensembles complexos
- Support Vector Machines com kernels não-lineares

Referência: James, G., Witten, D., Hastie, T., Tibshirani, R., (2013), An Introduction to Statistical Learning: with Applications in R. 1 ed. Springer.

Seleção de Subconjunto de Atributos



Atributos Redundantes

Duplicam informações contidas em um ou mais outros atributos, não agregando valor preditivo adicional ao modelo.

Exemplo: O preço de compra de um produto e o valor do imposto pago sobre essa compra são redundantes, pois o imposto é diretamente calculado a partir do preço.



Atributos Irrelevantes

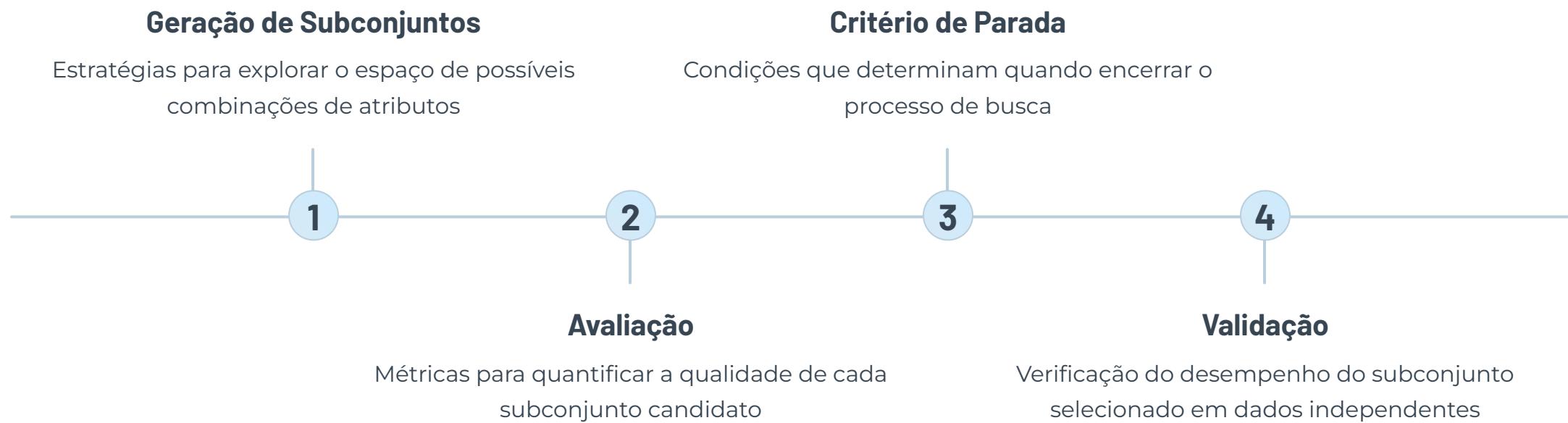
Não contêm informações úteis para a tarefa de mineração de dados em questão, apenas adicionando ruído e complexidade desnecessária.

Exemplo: IDs de estudantes são irrelevantes para a tarefa de prever o GPA (média de notas) dos alunos, pois são apenas identificadores arbitrários sem relação com desempenho acadêmico.

A identificação e remoção desses atributos melhora a eficiência computacional, reduz overfitting e aumenta a interpretabilidade dos modelos resultantes.

Princípios de Seleção de Atributos

A seleção de atributos segue uma arquitetura sistemática que combina diferentes componentes para identificar o subconjunto ótimo de características.



Referência: Dash, M., Liu, H., (1997), "Feature Selection for Classification", Intell. Data Anal., v. 1, n. 3, p. 131–156.

Métodos de Seleção de Atributos

Diversos algoritmos foram desenvolvidos para abordar o problema de seleção de atributos, cada um com características e aplicações específicas.



Information Gain (INFOGAIN)

Baseado em teoria da informação, mede a redução de entropia proporcionada por cada atributo na predição da classe alvo.



Forward Stepwise Selection (FSS)

Abordagem incremental que adiciona atributos um de cada vez, selecionando aquele que mais melhora o desempenho do modelo.



LASSO

Least Absolute Shrinkage and Selection Operator - penaliza coeficientes de regressão, forçando alguns a zero e efetivamente selecionando atributos.



Correlation-based Feature Selection (CFS)

Avalia subconjuntos baseando-se na correlação entre atributos e na correlação dos atributos com a classe.



Recursive Elimination (RELIEF)

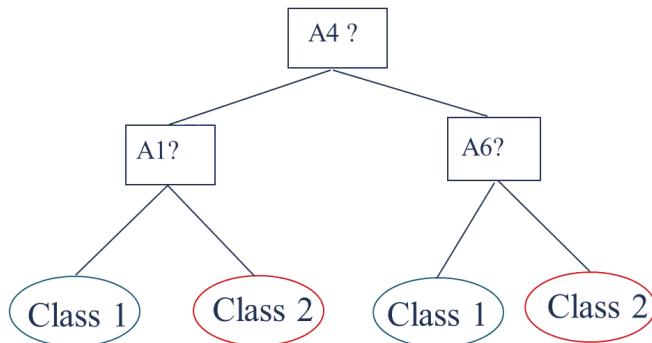
Atribui pesos aos atributos baseando-se em sua capacidade de distinguir instâncias de classes diferentes em vizinhanças próximas.

Information Gain

Conjunto Inicial de Atributos

{A1, A2, A3, A4, A5, A6}

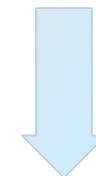
O método baseado em ganho de informação avalia cada atributo quanto à sua capacidade de reduzir a incerteza sobre a classe alvo.



Conjunto Reduzido

{A1, A4, A6}

Após a análise, apenas os atributos com maior ganho de informação são mantidos, otimizando a relação entre complexidade e poder preditivo.



Calcular Entropia Inicial

Mede a impureza do conjunto de dados antes da divisão



Avaliar Atributo A4

Calcula redução de entropia ao dividir por A4



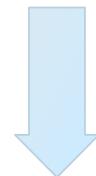
Avaliar Atributo A1

Calcula redução de entropia ao dividir por A1



Avaliar Atributo A6

Calcula redução de entropia ao dividir por A6

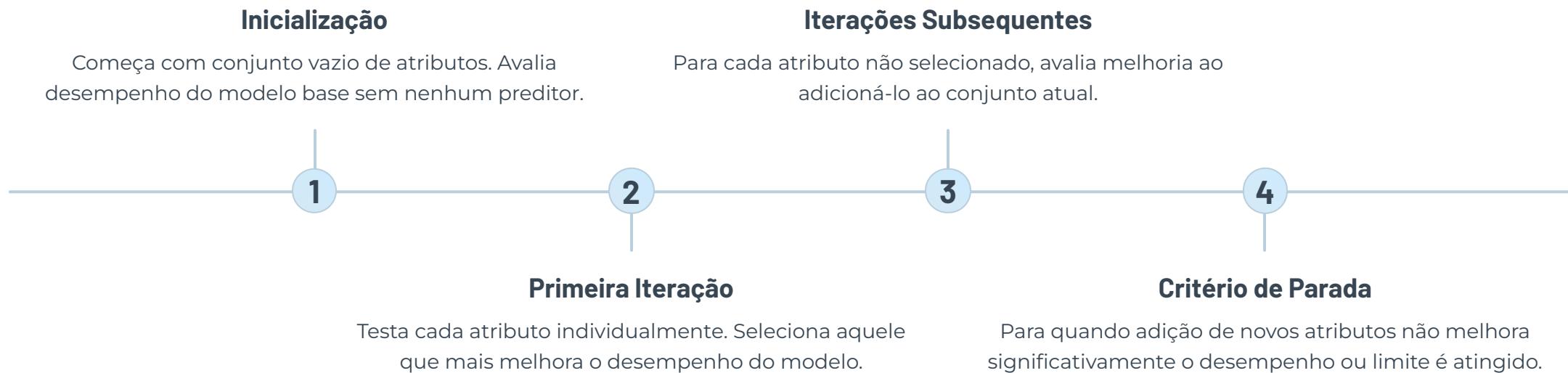


Classe 1 Identificada

Atributos selecionados maximizam discriminação entre classes

Forward Stepwise Selection

Forward Stepwise Selection (FSS) é um método guloso que constrói o conjunto de atributos incrementalmente, começando com conjunto vazio e adicionando um atributo por vez.



Vantagens

- Simples de implementar e entender
- Computacionalmente mais eficiente que busca exaustiva
- Funciona bem com diversos tipos de modelos

Limitações

- Pode não encontrar o subconjunto ótimo global
- Sensível à ordem de seleção inicial
- Não reconsidera atributos previamente adicionados

Least Absolute Shrinkage and Selection Operator (LASSO)

O LASSO opera sobre o modelo de regressão linear padrão, adicionando uma penalidade que impacta a seleção de atributos.

$$y = X\beta + \epsilon$$

Onde:

- $y \in \mathbb{R}^n$: vetor resposta
- $X \in \mathbb{R}^{n \times p}$: matriz de covariáveis
- $\beta \in \mathbb{R}^p$: vetor de coeficientes
- $\epsilon \in \mathbb{R}^n$: erro aleatório

A formulação do LASSO adiciona um termo de penalidade L_1 à função de custo da regressão:

$$\hat{\beta}^{lasso} = \arg \min_{\beta} \left\{ \frac{1}{2n} \|y - X\beta\|_2^2 + \lambda \|\beta\|_1 \right\}$$

Onde $\|\beta\|_1 = \sum_j |\beta_j|$ é a norma L_1 e $\lambda \geq 0$ é o parâmetro de regularização. Uma forma equivalente é expressa como um problema de otimização com restrição:

$$\min_{\beta} \left(\frac{1}{2n} \|y - X\beta\|_2^2 \right) \quad \text{sujeito a: } \|\beta\|_1 \leq t$$

Intuição da Penalização L_1

A penalização L_1 induz esparsidade, zerando coeficientes e combinando regularização com seleção de variáveis. Geometricamente, a solução ocorre nos vértices da região de restrição L_1 .

Propriedades Chave

É um problema convexo, mas não diferenciável em $\beta_j = 0$. A solução é tipicamente encontrada via Coordinate Descent e as variáveis devem ser padronizadas para evitar que a escala influencie a penalização.

Referência: James, G., Witten, D., Hastie, T., Tibshirani, R., (2013), *An Introduction to Statistical Learning: with Applications in R*. 1 ed. Springer.

Correlation-based Feature Selection (CFS)

CFS avalia o mérito de subconjuntos de atributos considerando a capacidade preditiva individual de cada atributo juntamente com o grau de redundância entre eles.



Alta Correlação com Classe

Busca atributos fortemente correlacionados com a variável alvo, maximizando o poder preditivo.

Baixa Correlação Entre Si

Prefere atributos com baixa correlação mútua, minimizando redundância e multicolinearidade.

Função de Mérito

Combina ambos os critérios em uma métrica única que pondera relevância individual versus redundância do conjunto.

Processo de Seleção

1. Calcular correlações atributo-classe
2. Calcular correlações atributo-atributo
3. Avaliar subconjuntos usando função de mérito
4. Selecionar subconjunto com melhor pontuação

Características

- Abordagem baseada em filtro, independente do algoritmo de aprendizado
- Eficiente computacionalmente
- Lida naturalmente com redundância
- Funciona com dados categóricos e numéricos

Recursive Elimination of Features (RELIEF)

RELIEF é um algoritmo baseado em instâncias que atribui pesos aos atributos de acordo com sua relevância para distinguir entre classes diferentes.

01

Amostragem Aleatória

Seleciona aleatoriamente instâncias do conjunto de treinamento para análise iterativa.

02

Identificação de Vizinhos

Para cada instância, identifica o vizinho mais próximo da mesma classe (nearHit) e o vizinho mais próximo de classes diferentes (nearMiss).

03

Cálculo de Score

Calcula o score do atributo baseado no comportamento médio: aumenta se diferencia nearMiss, diminui se diferencia nearHit.

04

Seleção por Threshold

Usuários escolhem atributos estabelecendo um limiar de score mínimo para inclusão no conjunto final.

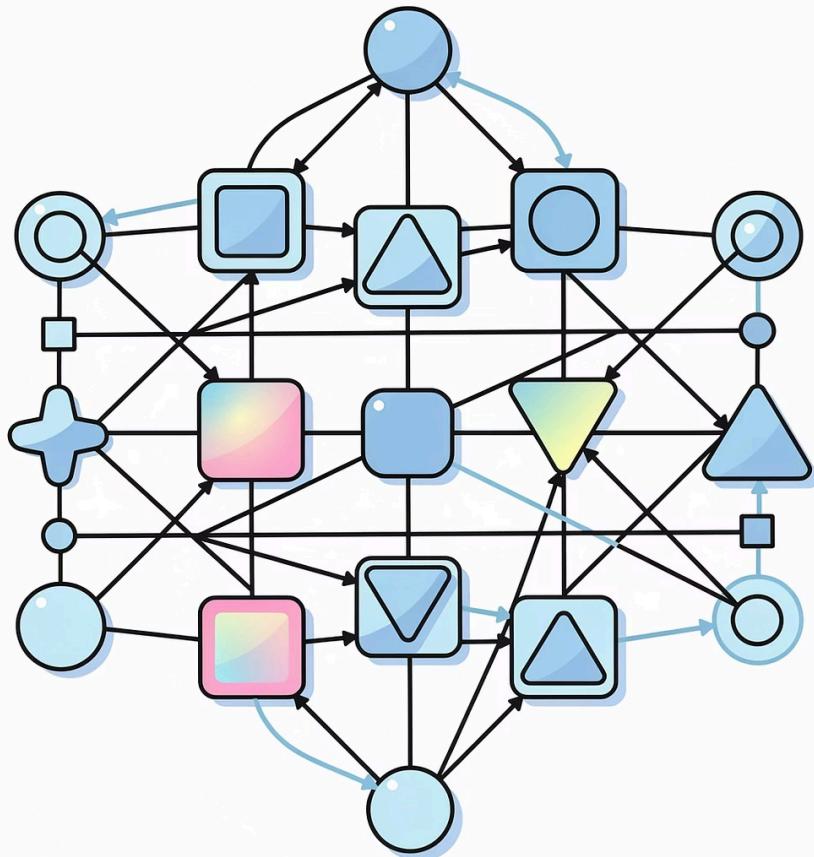
Pontos Fortes

- Detecta interações entre atributos
- Robusto a ruído nos dados
- Não assume independência de atributos
- Funciona bem em dados de alta dimensionalidade

Considerações

- Sensível à escolha da métrica de distância
- Pode ser computacionalmente custoso em grandes conjuntos
- Requer normalização de atributos numéricos
- Threshold de seleção requer ajuste

Referência: Dash, M., Liu, H., (1997), "Feature Selection for Classification", Intell. Data Anal., v. 1, n. 3, p. 131–156.



Tópicos Avançados

Neste módulo, exploraremos conceitos avançados em classificação de aprendizado de máquina. Abordaremos o teorema "No Free Lunch", técnicas para lidar com conjuntos de dados desbalanceados, classificação multiclasse, aprendizado semi-supervisionado e transfer learning. Esses tópicos são essenciais para desenvolver soluções robustas em cenários complexos do mundo real.

No Free Lunch em Classificação

Ausência de Superioridade Universal

Nenhum classificador único apresenta desempenho superior em todos os problemas possíveis. A eficácia de cada algoritmo varia conforme o contexto específico.

Dependência das Características dos Dados

O desempenho está intrinsecamente ligado às características dos dados e à representação de features escolhida para o problema.

Influência das Métricas

As métricas de avaliação selecionadas exercem impacto direto na escolha do modelo mais adequado para cada aplicação.

Seleção Empírica

A seleção de modelos é fundamentalmente empírica e dependente do contexto, exigindo experimentação sistemática e validação rigorosa.

Classificação de Conjuntos de Dados Desbalanceados

O Problema do Desbalanceamento

O problema de desbalanceamento de classes ocorre quando há exemplos positivos raros, mas numerosos exemplos negativos. Isso é comum em diagnóstico médico, detecção de fraudes, vazamento de óleo, detecção de falhas, entre outros.

Métodos tradicionais assumem distribuição balanceada de classes e custos de erro iguais, tornando-se inadequados para dados desbalanceados.



Oversampling

Reamostragem de dados da classe positiva para aumentar sua representação no conjunto de treinamento.

Under-sampling

Eliminação aleatória de tuplas da classe negativa para balancear a distribuição das classes.

Threshold-moving

Move o limiar de decisão para facilitar a classificação de tuplas da classe rara, reduzindo falsos negativos custosos.

Técnicas de Ensemble

Combinação de múltiplos classificadores para melhorar o desempenho geral em dados desbalanceados.

- O problema de desbalanceamento de classes permanece um desafio significativo em tarefas de classificação multiclasse, exigindo abordagens mais sofisticadas.

Classificação Multiclasses

A classificação multiclasses estende os conceitos binários para cenários onde existem três ou mais classes mutuamente exclusivas. Diferentemente da classificação binária, onde temos apenas duas categorias possíveis, a classificação multiclasses requer estratégias específicas para lidar com múltiplas categorias simultaneamente.

1

One-vs-Rest (OvR)

Treina um classificador binário para cada classe contra todas as outras classes. Simples de implementar, mas pode sofrer com desbalanceamento.

2

One-vs-One (OvO)

Treina um classificador binário para cada par de classes. Requer mais modelos, mas cada um enfrenta um problema mais simples.

3

Classificadores Nativos

Alguns algoritmos como redes neurais, árvores de decisão e regressão logística multinomial suportam multiclasses nativamente.

A escolha da estratégia depende do número de classes, da complexidade computacional aceitável e das características específicas do conjunto de dados. Técnicas de ensemble podem combinar múltiplas abordagens para melhorar a robustez.

Classificação Semi-Supervisionada

A classificação semi-supervisionada aproveita tanto dados rotulados quanto não rotulados para melhorar o desempenho do modelo. Esta abordagem é particularmente valiosa quando rotular dados é caro ou demorado, mas dados não rotulados são abundantes.

Dados Rotulados (labeled)

Exemplos com classes conhecidas que fornecem supervisão direta para o modelo. Geralmente limitados devido ao custo de anotação manual.

- Fornecem sinal de aprendizado claro
- Permitem validação precisa
- Requerem esforço humano significativo

Dados Não Rotulados (unlabeled)

Exemplos sem classes conhecidas que ajudam a capturar a estrutura subjacente dos dados e melhorar a generalização.

- Abundantes e de baixo custo
- Revelam estrutura dos dados
- Melhoram fronteiras de decisão



Self-Training

O modelo rotula seus dados mais confiantes e os adiciona ao conjunto de treinamento iterativamente.

Co-Training

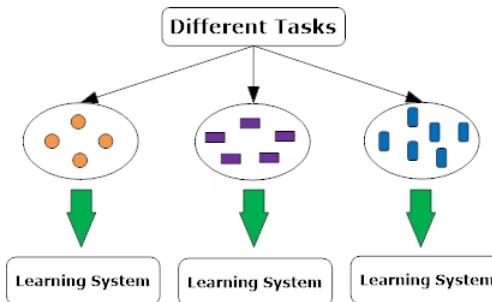
Múltiplos modelos treinados em diferentes views dos dados colaboram para rotular exemplos não supervisionados.

Graph-Based

Propaga rótulos através de grafos baseados em similaridade entre exemplos rotulados e não rotulados.

Transfer Learning: Framework Conceitual

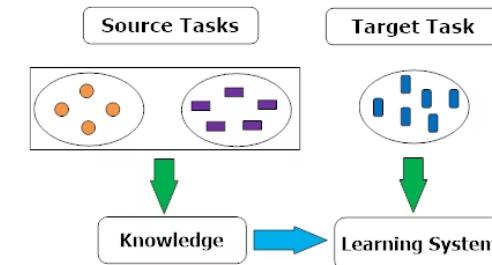
Aprendizado Tradicional



No framework tradicional, um novo classificador é construído do zero para cada nova tarefa. Isso requer grandes quantidades de dados rotulados e tempo computacional para cada problema individual.

- Cada tarefa é tratada isoladamente
- Requer dados abundantes para cada domínio
- Não aproveita conhecimento prévio
- Processo repetitivo e custoso

Framework de Transfer Learning



Transfer learning extrai conhecimento de uma ou mais tarefas fonte e aplica esse conhecimento a uma tarefa alvo. Constrói novos classificadores aplicando conhecimento existente aprendido de tarefas fonte.

- Reutiliza conhecimento entre domínios
- Reduz necessidade de dados rotulados
- Acelera convergência do treinamento
- Melhora desempenho em tarefas relacionadas

- Transfer learning é especialmente valioso quando dados rotulados são escassos na tarefa alvo, mas abundantes em domínios relacionados. Isso permite aproveitar conhecimento prévio para acelerar o aprendizado e melhorar a generalização.

Transfer Learning: Métodos e Aplicações



Transfer Learning Baseado em Instâncias

Reponde a alguns dados das tarefas fonte e os utiliza para aprender a tarefa alvo, permitindo transferência direta de exemplos relevantes.



Transfer AdaBoost

Assume que dados fonte e alvo são descritos pelos mesmos atributos e rótulos, mas com distribuições diferentes. Requer rotular apenas uma pequena quantidade de dados alvo.

Como Transfer AdaBoost Utiliza Dados Fonte

Quando uma tupla fonte é classificada incorretamente, o algoritmo reduz o peso dessas tuplas para que tenham menor efeito nos classificadores subsequentes. Isso permite que o modelo se adapte gradualmente à distribuição da tarefa alvo enquanto aproveita conhecimento das tarefas fonte.

Questões de Pesquisa em Aberto

Transferência Negativa

Ocorre quando o modelo com transferência apresenta desempenho inferior ao modelo sem transferência, geralmente devido a diferenças significativas entre domínios.

Transfer Learning Heterogêneo

Transferência de conhecimento entre espaços de features diferentes ou múltiplos domínios fonte com características distintas.

Transfer Learning em Larga Escala

Desenvolvimento de métodos eficientes para transferir conhecimento com grandes volumes de dados e modelos complexos como deep learning.

Leituras Recomendadas

Para aprofundar seu conhecimento nos tópicos avançados apresentados, recomendamos a leitura dos seguintes artigos fundamentais na área:

Weiss et al. *Big Data* (2016) 3:9
DOI 10.1186/s40537-016-0043-6

Journal of Big Data

SURVEY PAPER **Open Access** 

A survey of transfer learning

Karl Weiss*, Taghi M. Khoshgoftaar and DingDing Wang

Correspondence:
khoshgoftaar@jhu.edu
Florida Atlantic University,
777 Glades Road, Boca Raton,
FL 33401, USA

Abstract
Machine learning and data mining techniques have been used in numerous real-world applications. A common problem in machine learning and data mining is the training data and testing data are taken from the same domain, such that the input feature space and data distribution characteristics are the same. However, in some real-world machine learning scenarios, this assumption does not hold. There are cases where training data is expensive or difficult to collect. Therefore, there is a need to create high-performance learners trained with more easily obtainable data from different domains. This method is referred to as transfer learning. This paper formally defines transfer learning, presents information on current solutions, and reviews applications applied to transfer learning. Lastly, there is information listed on software downloads for various transfer learning solutions and a discussion of possible future research areas. The transfer learning solutions surveyed are independent of data size and can be applied to big data environments.

Keywords: Transfer learning, Survey, Domain adaptation, Machine learning, Data mining

Background
The field of data mining and machine learning has been widely and successfully used in many applications where patterns from past information (training data) can be extracted in order to predict future outcomes [129]. Traditional machine learning is characterized by training data and testing data using the same input features and the same data distribution. When there is a difference in data distribution between the training data and test data, the performance of the learner will be degraded [107]. In some scenarios, obtaining training data that matches the feature space and predicted data distribution characteristics of the test data can be difficult and expensive. Therefore, there is a need to create a high-performance learner for a target domain trained from a related source domain. This is the motivation for transfer learning.

Transfer learning is used to improve a learner from one domain by transferring information from a related domain. We can draw from real-world non-technical experiences to understand why transfer learning is possible. Consider an example of two people who want to learn to play the piano. One person has no previous experience playing music, and the other person has extensive music knowledge through playing the guitar. The person with an extensive music background will be able to learn the piano in a more efficient manner by transferring previously learned music knowledge to the task of learning.

 Springer Open © 2016 The Author(s). This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

Artigos Fundamentais

He, H., Garcia, E. A. (2009). "Learning from imbalanced data". *IEEE Transactions on Knowledge and Data Engineering*, v. 21, n. 9, p. 1263-1284.

Este artigo abrangente apresenta técnicas e desafios para lidar com conjuntos de dados desbalanceados, um problema crítico em aplicações práticas de machine learning.

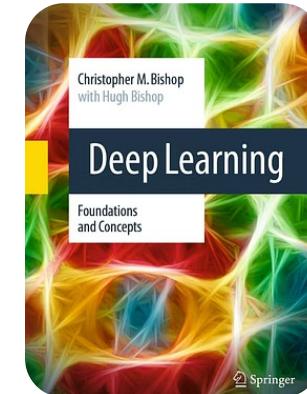
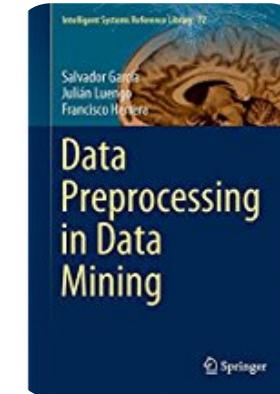
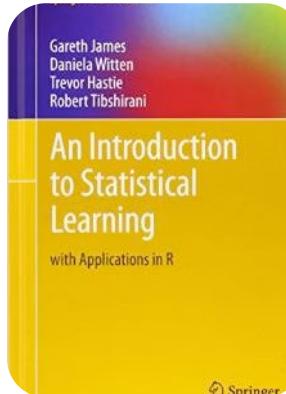
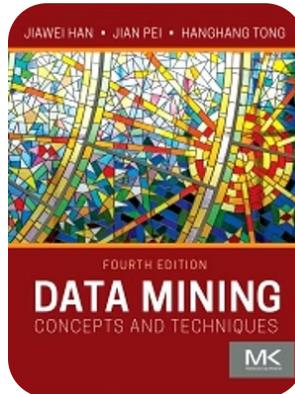
Weiss, K., Khoshgoftaar, T. M., Wang, D. D. (2016). "A survey of transfer learning". *Journal of Big Data*, v. 3, n. 1.

Um survey completo sobre transfer learning, cobrindo métodos, aplicações e direções futuras de pesquisa nesta área em rápida evolução.

- ☐ Estes artigos fornecem uma base sólida para compreender os desafios e soluções em classificação avançada. Recomenda-se também explorar implementações práticas e estudos de caso recentes para consolidar o conhecimento teórico.

Referências Principais

Esta seleção de referências representa os pilares fundamentais para o estudo aprofundado de mineração de dados, cobrindo desde conceitos básicos até técnicas avançadas e aplicações contemporâneas.



1. **J. Han, J. Pei, and H. Tong**, *Data Mining: Concepts and Techniques*, 4th edition. Cambridge, MA: Morgan Kaufmann, 2022.
2. **G. M. James, D. Witten, T. Hastie, and R. Tibshirani**, *An Introduction to Statistical Learning: With Applications in R*. Springer Nature, 2021.
3. **S. Garcia, J. Luengo, and F. Herrera**, *Data Preprocessing in Data Mining*. Springer, 2014.
4. **C. M. Bishop and H. Bishop**, *Deep Learning: Foundations and Concepts*. Springer Nature, 2023.