

# Algoritmos de Classificação e Fronteiras de Decisão

Uma exploração técnica dos algoritmos fundamentais que transformam dados em decisões através de fronteiras matemáticas elegantes

# O Problema Fundamental da Classificação

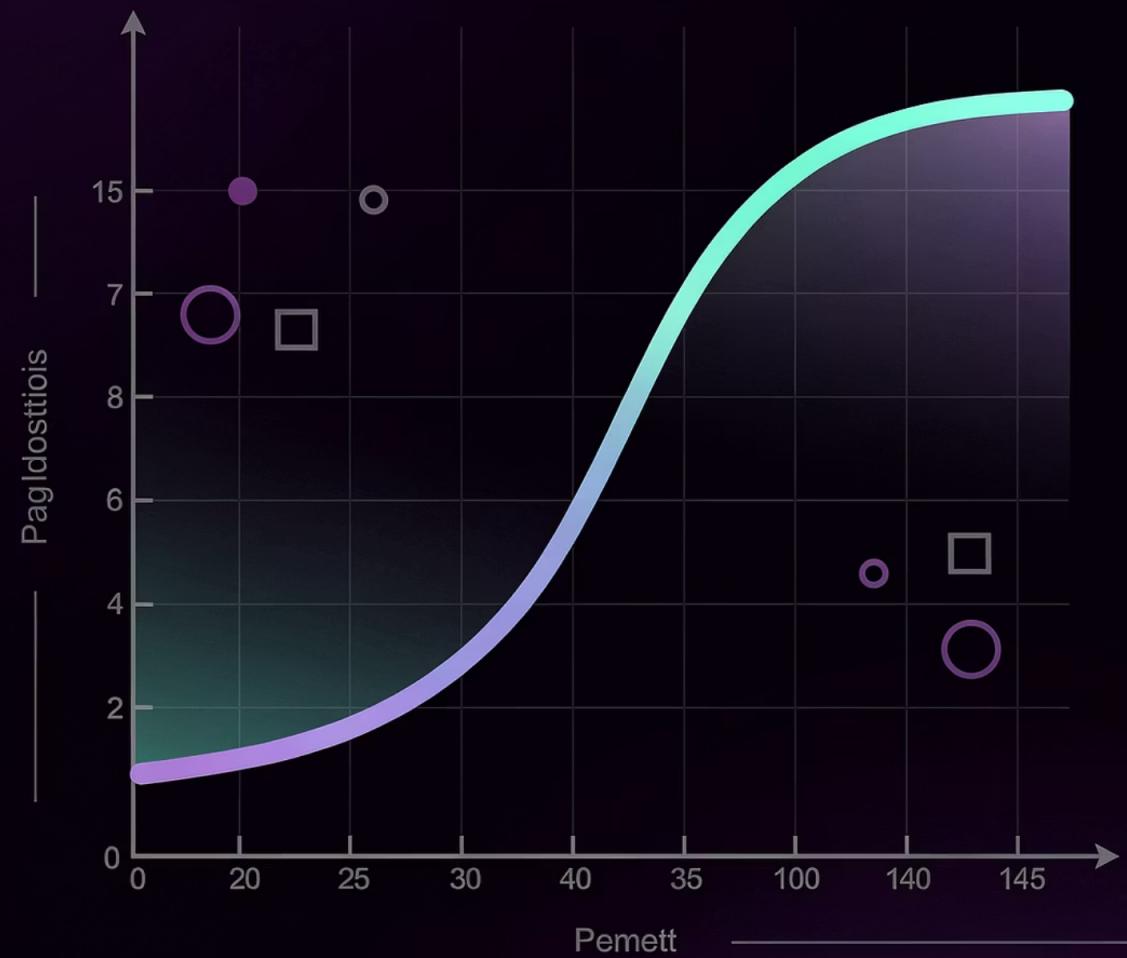
## Desafio Central

Como um algoritmo pode aprender a separar objetos de diferentes categorias em um espaço de características? A classificação busca encontrar fronteiras de decisão que dividam o espaço de dados em regiões correspondentes a cada classe.

Esta aula explora três abordagens fundamentais: Regressão Logística, Support Vector Machines e Árvores de Decisão.

## Objetivos de Aprendizagem

- Compreender a matemática por trás das fronteiras de decisão
- Dominar os conceitos de probabilidade condicional e margem máxima
- Aplicar métricas de entropia para construir árvores
- Visualizar e interpretar separação de classes



# Regressão Logística: Probabilidades e Decisões

A Regressão Logística transforma uma combinação linear de características em uma probabilidade através da função Sísmoide, permitindo decisões probabilísticas elegantes em problemas de classificação binária.

# A Função Sigmoide: Coração da Regressão Logística

## Definição Matemática

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

onde  $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \dots + \beta_n x_n$

A função mapeia qualquer valor real para o intervalo (0,1), interpretável como probabilidade.

## Propriedades Essenciais

- Monotônica: Sempre crescente, preservando ordem
- Suave: Diferenciável em todos os pontos
- Assintótica: Aproxima-se de 0 e 1 nos extremos
- Simétrica: Centro em 0.5 quando  $z = 0$

Essas propriedades tornam a sigmoide ideal para modelar probabilidades condicionais.

# Probabilidade Condisional e Inferência

01

## Combinação Linear

Calcular  $z = \beta^T x$  usando os pesos aprendidos e as características de entrada

02

## Transformação Sísmoide

Aplicar  $\sigma(z)$  para obter  $P(Y=1|X)$ , a probabilidade da classe positiva

03

## Decisão por Threshold

Classificar como classe 1 se  $P(Y = 1 | X) \geq 0.5$ , caso contrário classe 0

A fronteira de decisão ocorre onde  $P(Y=1|X) = 0.5$ , ou seja, onde  $z = 0$ . Esta é uma superfície linear no espaço de características.

# Estimação por Máxima Verossimilhança

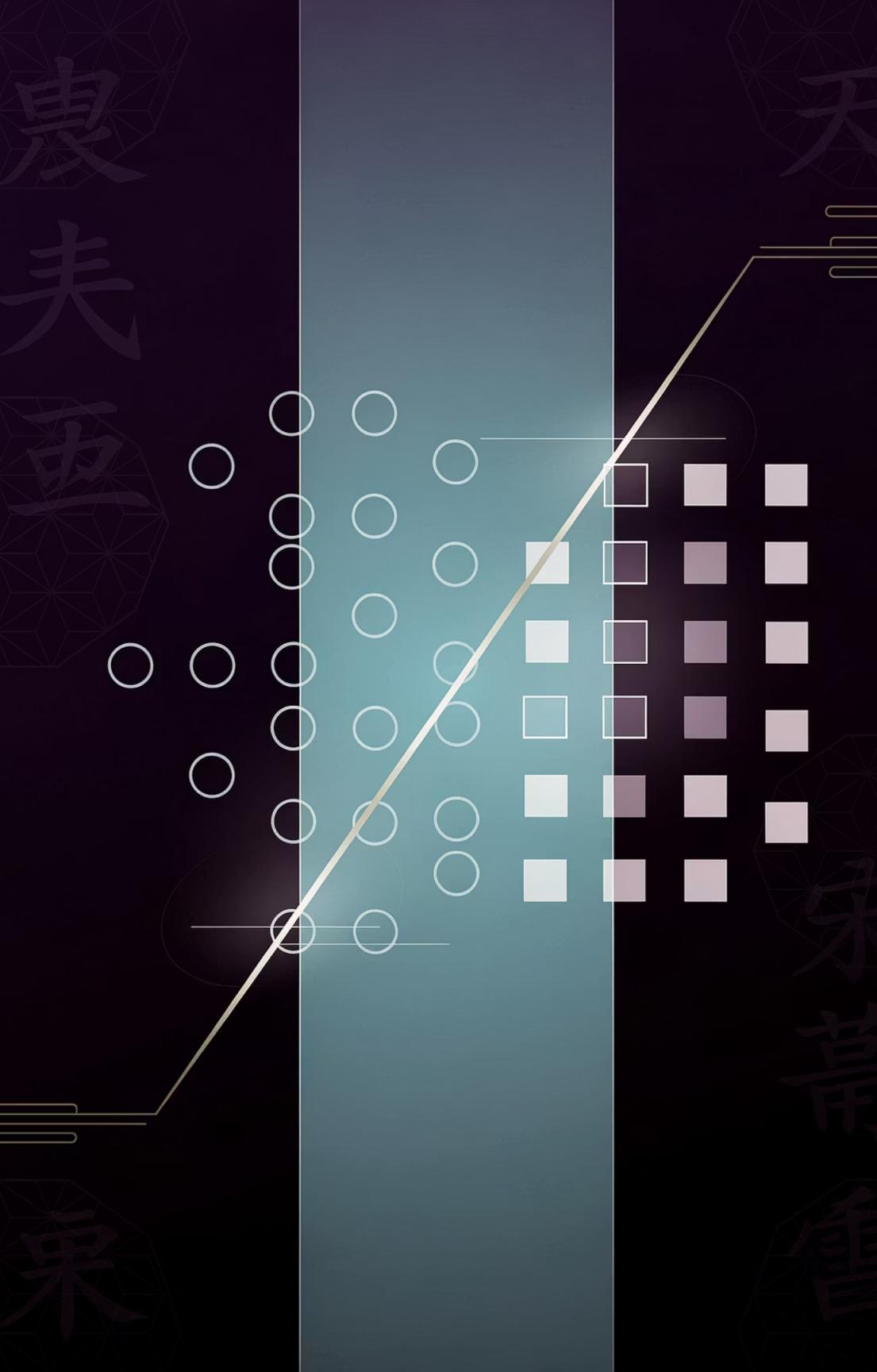
A Regressão Logística utiliza Maximum Likelihood Estimation (MLE) para encontrar os parâmetros ótimos. A função de verossimilhança para n observações é:

$$L(\beta) = \prod_{i=1}^n P(Y_i | X_i)^{y_i} (1 - P(Y_i | X_i))^{1-y_i}$$

Maximizamos o log-likelihood usando gradient descent ou métodos de segunda ordem como Newton-Raphson.

## Vantagens da Abordagem

- Interpretação probabilística direta
- Convergência garantida (função convexa)
- Extensível para múltiplas classes (softmax)
- Base para redes neurais modernas



## Support Vector Machines: Geometria da Margem Máxima

SVM revolucionou o aprendizado de máquina ao formular a classificação como um problema de otimização geométrica: encontrar o hiperplano que maximiza a margem entre as classes, garantindo robustez e generalização superior.

# O Conceito de Margem Máxima

1

## Hiperplano Separador

Uma superfície de decisão  $w^T x + b = 0$  que divide o espaço em duas regiões. O vetor  $w$  é perpendicular ao hiperplano.

2

## Definição de Margem

A distância perpendicular do hiperplano aos pontos mais próximos de cada classe.  
Matematicamente:  $\frac{2}{||w||}$

3

## Problema de Otimização

Maximizar a margem é equivalente a minimizar  $||w||^2$  sujeito a  $y_i(w^T x_i + b) \geq 1$  para todos os pontos

# Vetores de Suporte: Os Pontos Críticos

## Identificação e Papel

Vetores de suporte são os pontos de treinamento que se encontram exatamente na margem, satisfazendo  $y_i(w^T x_i + b) = 1$ . Estes pontos são os únicos que determinam o hiperplano ótimo.

Removendo qualquer outro ponto de treinamento, o hiperplano permanece inalterado. Esta propriedade confere ao SVM sua eficiência e robustez.

## Implicações Práticas

- Esparsidade: Apenas alguns pontos importam
- Robustez: Insensível a outliers distantes
- Memória: Apenas vetores de suporte são armazenados
- Generalização: Foco em casos difíceis

# Formulação Dual e Multiplicadores de Lagrange

A formulação primal do SVM é transformada em sua forma dual usando multiplicadores de Lagrange  $\alpha_i$ . Esta transformação é crucial para o kernel trick:

$$L_D = \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j x_i^T x_j$$

## Condições KKT

As condições de Karush-Kuhn-Tucker garantem que apenas vetores de suporte têm  $\alpha_i > 0$

## Produto Interno

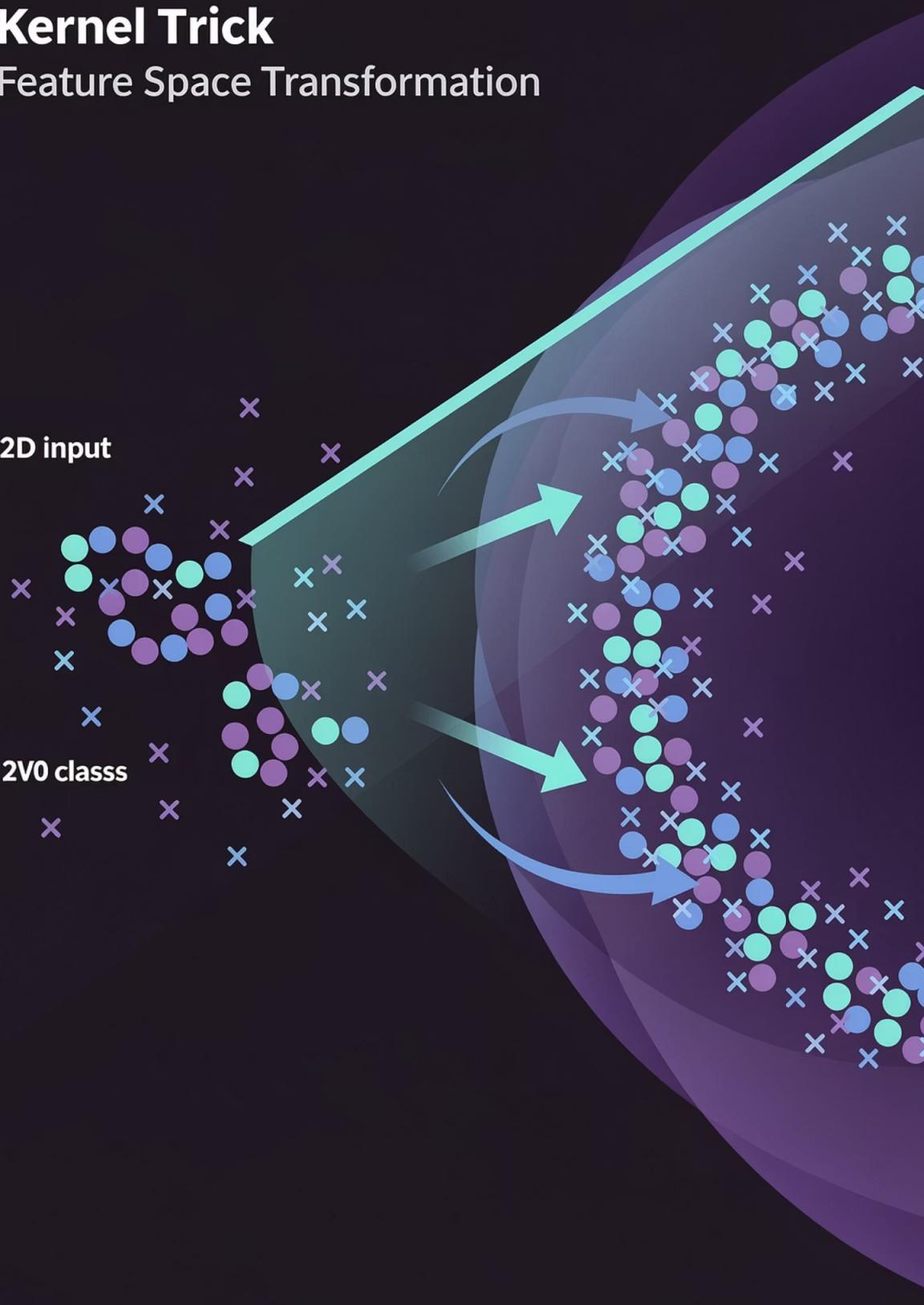
Note que a solução depende apenas de  $x_i^T x_j$ , abrindo caminho para kernels

## Eficiência Computacional

Quadratic programming com restrições lineares, solucionável eficientemente

# Kernel Trick

## Feature Space Transformation



# O Kernel Trick: Mapeando para Dimensões Superiores

## Intuição Central

Dados não linearmente separáveis no espaço original podem tornar-se separáveis em um espaço de características de maior dimensão através de um mapeamento  $\phi(x)$ .

## Kernels Populares

- Linear:  $K(x, z) = x^T z$
- Polinomial:  $K(x, z) = (x^T z + c)^d$
- RBF (Gaussiano):  $K(x, z) = e^{-\gamma ||x-z||^2}$
- Sigmoide:  $K(x, z) = \tanh(\kappa x^T z + c)$

O kernel trick evita computar  $\phi(x)$  explicitamente, calculando apenas

$$K(x_i, x_j) = \phi(x_i)^T \phi(x_j).$$

# Soft Margin e Variável de Folga

Para lidar com dados não perfeitamente separáveis ou ruído, introduzimos variáveis de folga  $\xi_i$  que permitem violações controladas da margem:

$$\min_{w,b,\xi} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^n \xi_i$$

## Parâmetro C

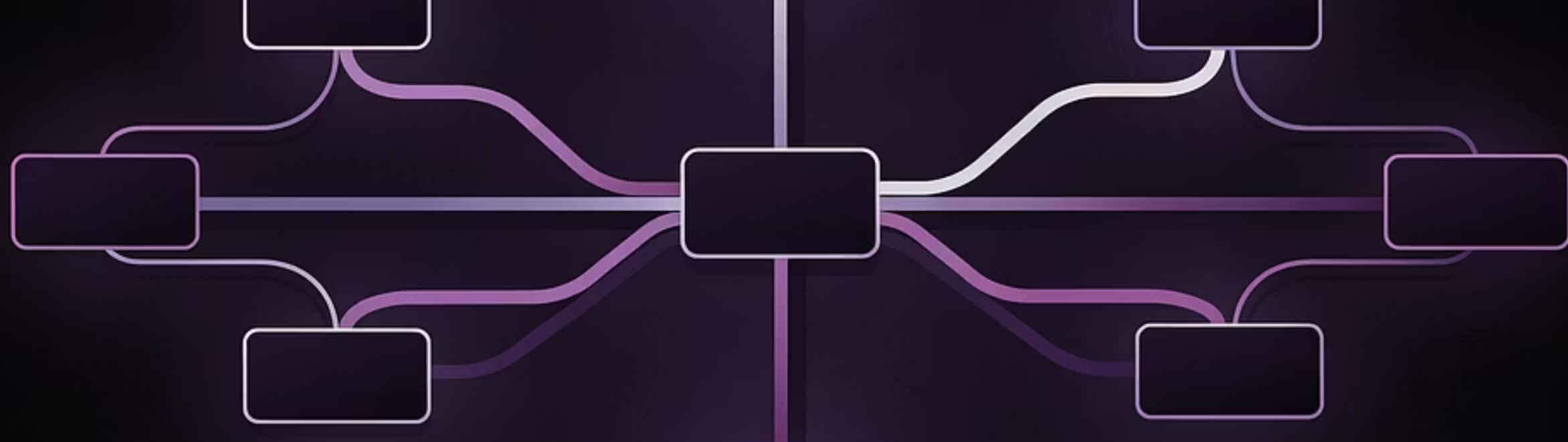
Controla o trade-off entre  
margem máxima e minimização  
de erros de classificação

## C Grande

Penaliza fortemente erros, busca  
classificação perfeita (risco de  
overfitting)

## C Pequeno

Tolera mais erros, prioriza  
margem ampla (melhor  
generalização)



# Árvores de Decisão: Aprendizado Hierárquico

Árvores de Decisão constroem modelos interpretáveis através de uma sequência hierárquica de perguntas sobre as características, dividindo recursivamente o espaço de dados em regiões homogêneas.

# Entropia: Medindo a Impureza

## Definição de Entropia

A entropia de Shannon mede a desordem ou incerteza em um conjunto de dados:

$$H(S) = - \sum_{i=1}^c p_i \log_2(p_i)$$

onde  $p_i$  é a proporção de exemplos da classe  $i$  no conjunto  $S$ , e  $c$  é o número de classes.

## Interpretação e Limites

- $H = 0$ : Conjunto puro (todos da mesma classe)
- $H$  máximo: Distribuição uniforme (máxima incerteza)
- $H \in [0, \log_2(c)]$ : Intervalo possível
- Unidade: bits de informação

Menor entropia indica melhor separação das classes.

# Ganho de Informação (Information Gain)

O Information Gain mede a redução de entropia obtida ao dividir o conjunto de dados por um atributo específico:

$$IG(S, A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} H(S_v)$$

## Entropia Inicial

Calcular  $H(S)$  do conjunto completo antes da divisão

## Média Ponderada

Ponderar cada entropia pela proporção de exemplos em cada subconjunto

## Entropia Condisional

Para cada valor do atributo, calcular  $H(S_v)$  do subconjunto resultante

## Ganho Final

Subtrair a entropia ponderada da entropia inicial para obter o ganho

# Construção da Árvore: Algoritmo ID3

```
function ID3(S, Attributes):
    if all examples in S have same class:
        return leaf node with that class

    if Attributes is empty:
        return leaf with majority class

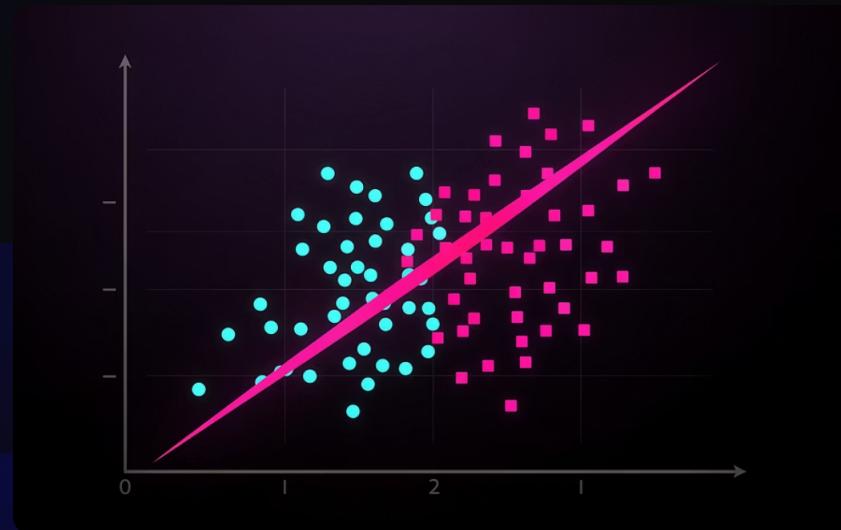
    A = attribute with highest IG(S, A)
    tree = new decision node for A

    for each value v of A:
        S_v = subset of S where A = v
        if S_v is empty:
            add leaf with majority class in S
        else:
            subtree = ID3(S_v, Attributes - {A})
            add branch to tree for A=v with subtree

    return tree
```

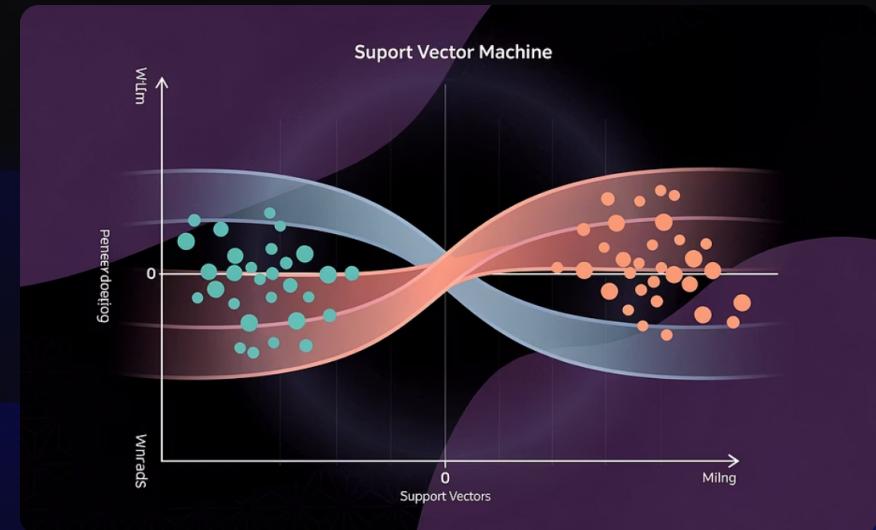
O algoritmo constrói a árvore de forma gulosa, selecionando recursivamente o atributo que maximiza o ganho de informação em cada nó.

# Visualização das Fronteiras de Decisão



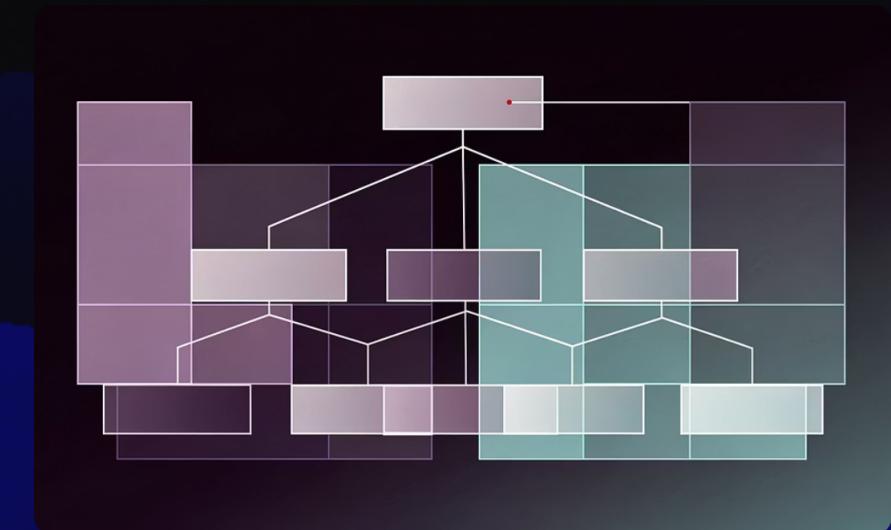
## Regressão Logística

Fronteira linear suave definida por  $w^T x + b = 0$ . Ideal para problemas linearmente separáveis com interpretação probabilística.



## SVM com Kernel

Fronteiras não lineares complexas através do kernel trick. Maximiza margem para robustez e generalização superior.



## Árvore de Decisão

Regiões retangulares alinhadas aos eixos. Divisões hierárquicas criam fronteiras interpretáveis mas potencialmente irregulares.

# Comparação Crítica dos Algoritmos

## Regressão Logística

- Vantagens: Probabilidades calibradas, rápida, convexa
- Limitações: Apenas fronteiras lineares
- Uso ideal: Baseline, interpretação probabilística necessária

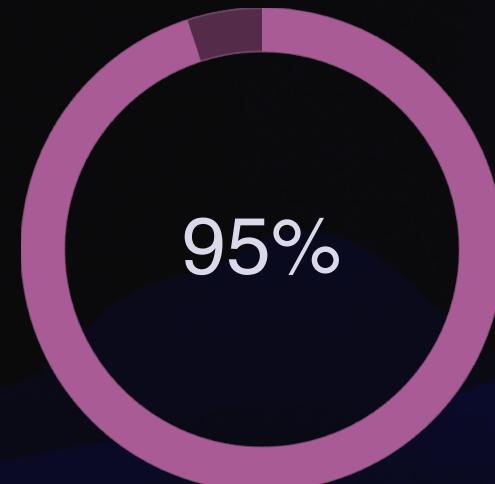
## SVM

- Vantagens: Margem máxima, kernel trick, robusto
- Limitações: Sensível a escala, custo computacional  $O(n^3)$
- Uso ideal: Dados de alta dimensão, fronteiras complexas

## Árvore de Decisão

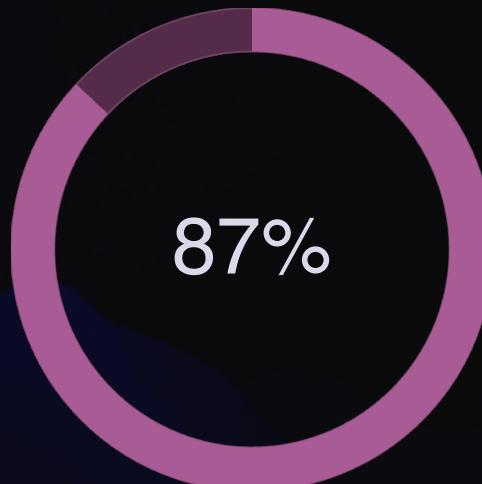
- Vantagens: Interpretável, não paramétrica, multiclasse nativo
- Limitações: Overfitting, instável, fronteiras descontínuas
- Uso ideal: Explicabilidade crítica, características categóricas

## Métricas de Avaliação e Validação



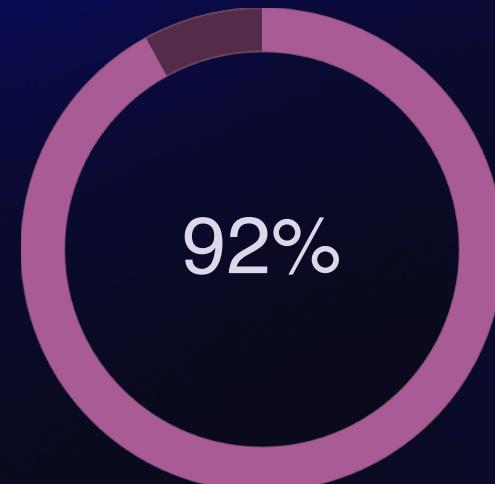
Acurácia

Proporção de previsões corretas sobre o total



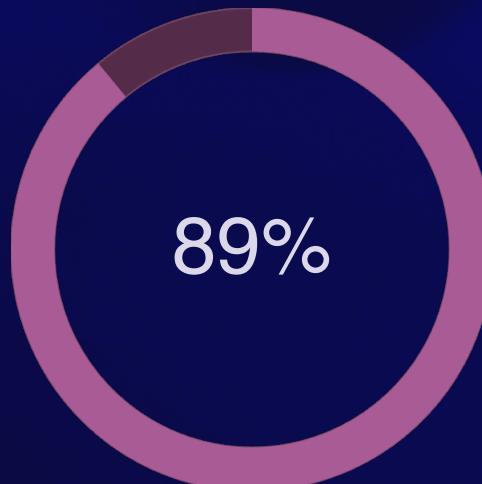
Precisão

Positivos verdadeiros sobre previsões positivas



Recall

Positivos verdadeiros sobre positivos reais



F1-Score

Média harmônica entre precisão e recall

Utilize validação cruzada k-fold para estimativas robustas. Matriz de confusão fornece análise detalhada de erros. Para dados desbalanceados, prefira F1-score ou AUC-ROC à acurácia.

# Conclusões e Próximos Passos

## Fundações Estabelecidas

Dominamos três paradigmas fundamentais de classificação: probabilístico (logística), geométrico (SVM) e baseado em regras (árvores)

## Extensões Avançadas

Ensemble methods (Random Forest, Gradient Boosting) combinam múltiplas árvores. Deep Learning generaliza regressão logística em camadas

## Aplicações Práticas

Diagnóstico médico, detecção de fraude, classificação de documentos, reconhecimento de padrões e sistemas de recomendação

- ❑ Leitura recomendada: Hastie et al. "The Elements of Statistical Learning" (capítulos 4, 9 e 12) oferece tratamento matemático rigoroso destes algoritmos.