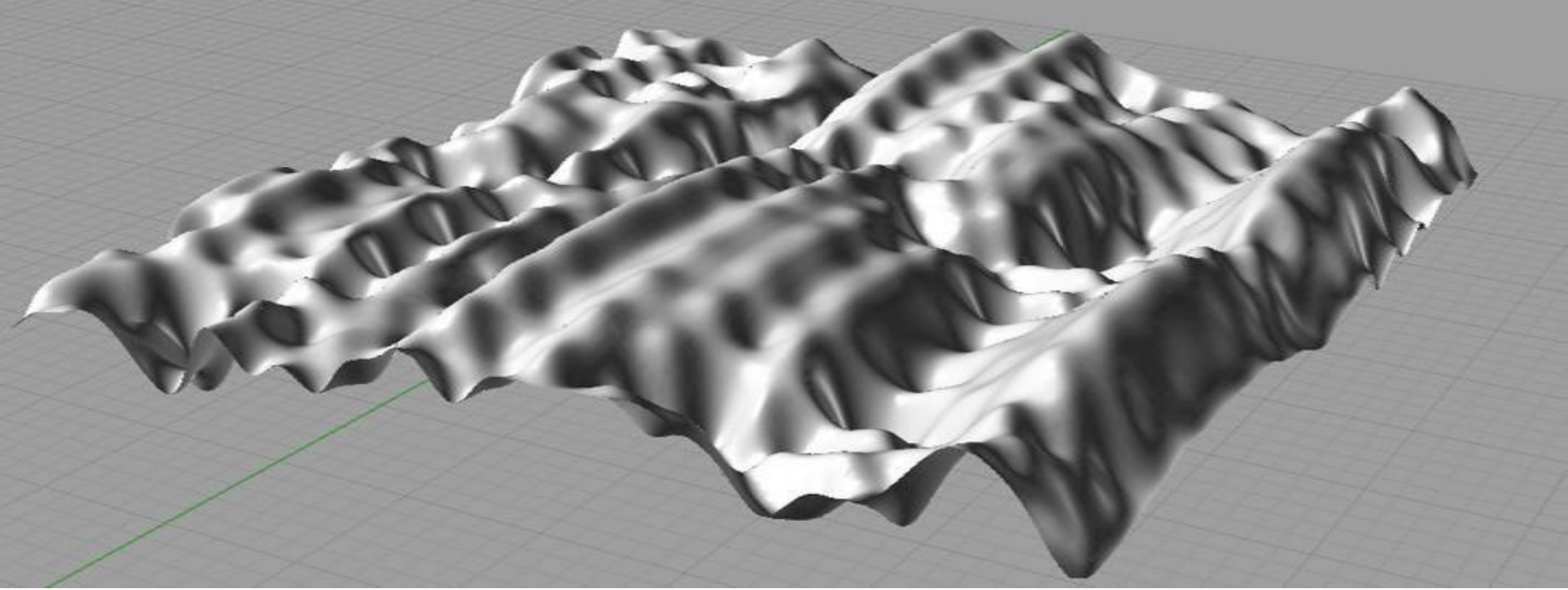


CLASSIFICADORES NÃO-PARAMÉTRICOS

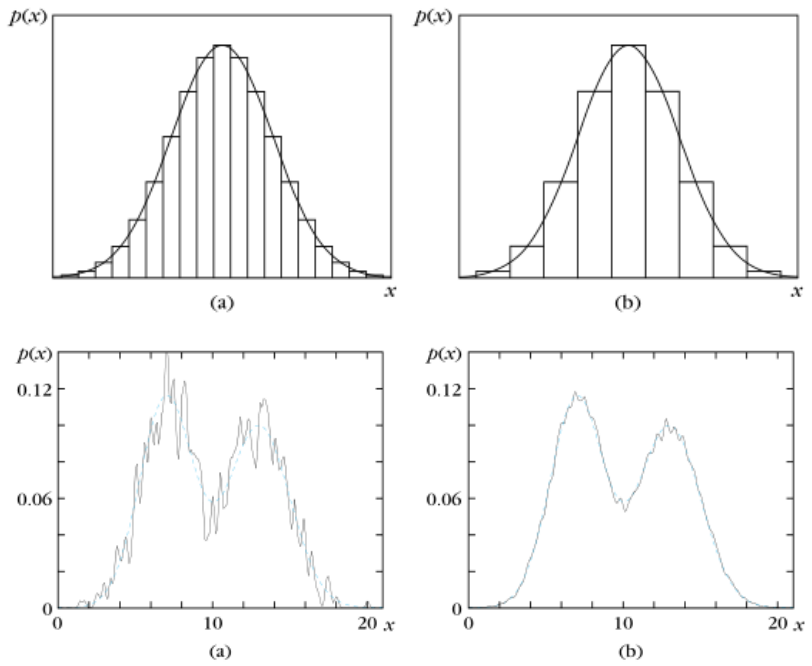
André Gustavo Adami
Daniel Luis Notari

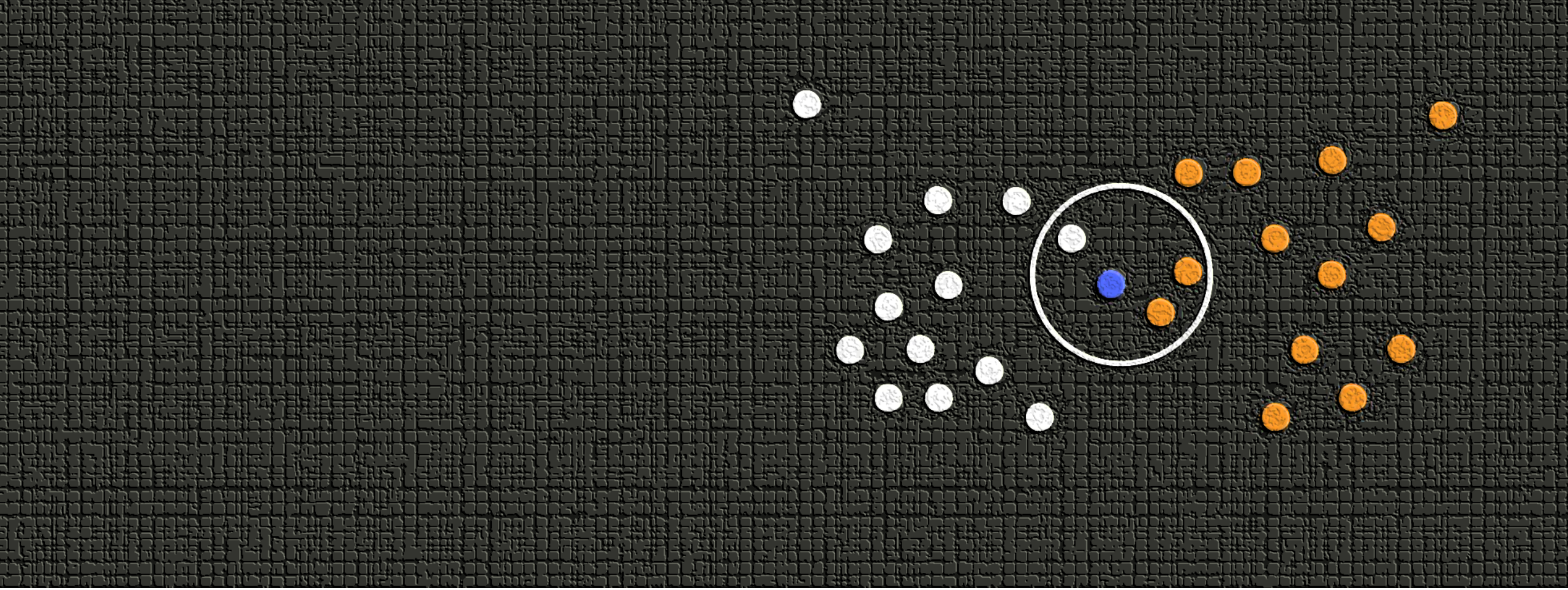
MÉTODOS NÃO-PARAMÉTRICOS

Uma abordagem diferente é não fazer nenhuma suposição explica sobre o tipo de função $f(\cdot)$

Estes métodos buscam estimar $f(\cdot)$ que aproxime ao máximo às amostras sem ser muito superficial ou detalhista

- Histograma
- Janelas de Parzen
- Árvores de Decisão
- Máquinas de Vetor de Suporte
- K Vizinhos Mais Próximos (k-Nearest Neighbors)



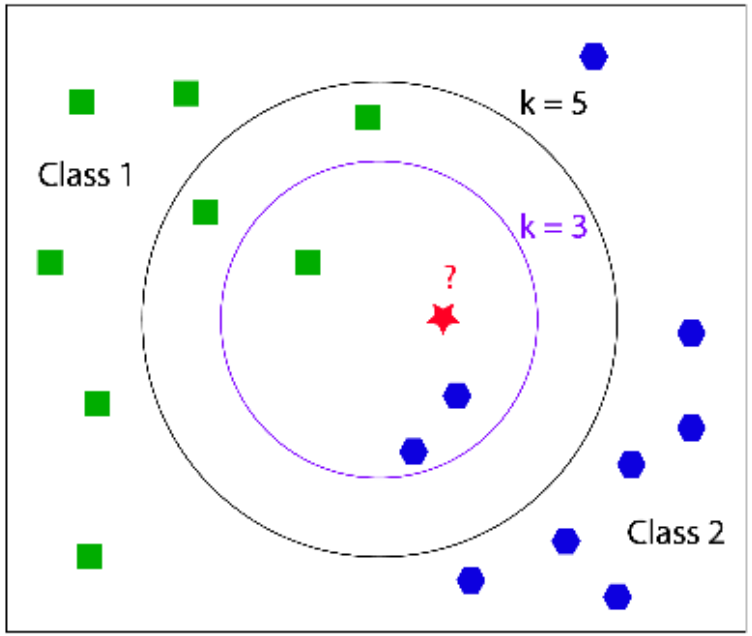


K NEAREST NEIGHBOR – K-NN

K NEAREST NEIGHBOR

K-Nearest Neighbor (k-NN) - k Vizinhos mais Próximos - consiste em classificar x atribuindo a ele o rótulo representado mais frequentemente (contagem de votos) dentre as k amostras mais próximas

- O algoritmo k-NN assume que todas as instâncias correspondem a pontos em um espaço n–dimensional
- Os "vizinhos mais próximos" de uma instância são definidos em termos da distância Euclidiana

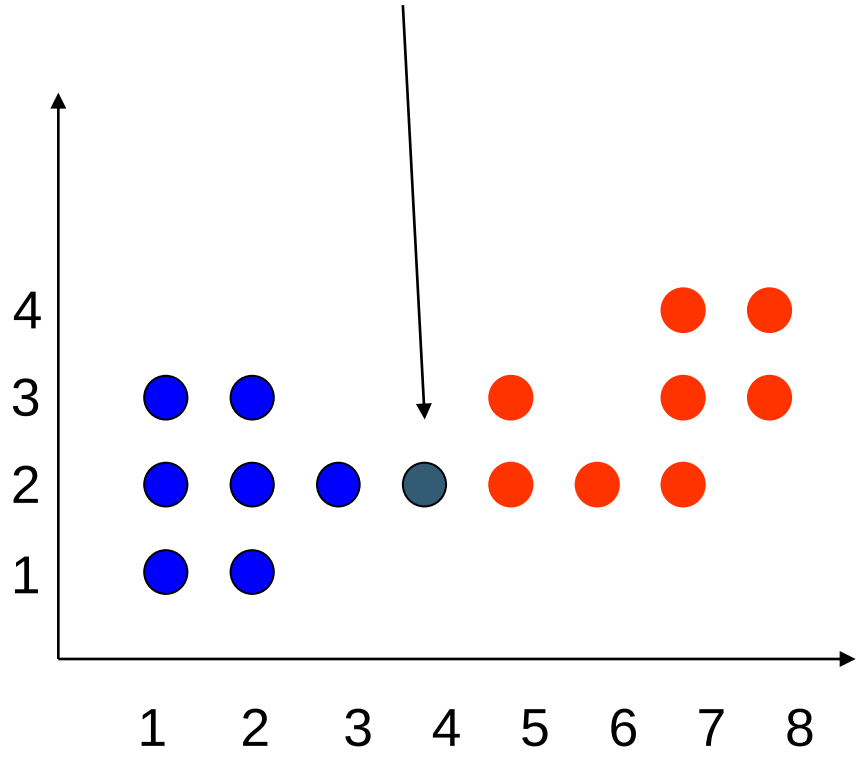


Armazena todas as amostras de referência (= amostras de treinamento)

COVER, T.; HART, P. Nearest neighbor pattern classification. IEEE Transactions on Information Theory, 13(1):21–27, 1967.

K NEAREST NEIGHBOR: UM EXEMPLO

A qual classe pertence este ponto? Azul ou vermelho?



Calcule para os seguintes valores de k:

k=1 não se pode afirmar

k=3 vermelho – 5,2 - 5,3

k=5 vermelho – 5,2 - 5,3 - 6,2

k=7 azul – 3,2 - 2,3 - 2,2 - 2,1

A classificação pode mudar de acordo com a escolha de *k*

K NEAREST NEIGHBOR

O volume da distribuição é uma função dos dados de treinamento, ou também conhecidos por dados de referência

Para estimar $p(\boldsymbol{x})$ de *n* amostras de treinamento, é possível centralizar uma região em $\boldsymbol{x}$ e deixá-la crescer até que ela capture k_n amostras, onde k_n é alguma função de n.

- Estas amostrar são os $\boldsymbol{k_n}$ vizinhos de $\boldsymbol{x}$
- Se a densidade é alta perto de $\boldsymbol{x}$, então a região é relativamente pequena (boa resolução)
- Se a densidade é baixa perto de $\boldsymbol{x}$, então a região crescerá até uma região de alta densidade

$$p_n(\boldsymbol{x}) = \frac{k_n/n}{V}$$

onde V é o volume da região que circunda x

K NEAREST NEIGHBOR

Como a probabilidade a posteriori é estimada da seguinte maneira

$$P_n(\omega_i|\boldsymbol{x}) = \frac{p_n(\boldsymbol{x}, \omega_i)}{p_n(\boldsymbol{x})}$$

e a probabilidade conjunta $p(\boldsymbol{x}, \omega_i)$ pode ser estimada através

$$p_n(\boldsymbol{x}, \omega_i) = \frac{k_i/n}{V}$$

e

$$p_n(\boldsymbol{x}) = \sum_{\boldsymbol{j=1}}^{\boldsymbol{c}} p_n(\boldsymbol{x}, \omega_j)$$

K NEAREST NEIGHBOR

Então

$$P_n(\omega_i|\boldsymbol{x}) = \frac{p_n(\boldsymbol{x}, \omega_i)}{\sum_{j=1}^{c} p_n(\boldsymbol{x}, \omega_j)} = \frac{k_i}{k}$$

Estimar a probabilidade a posteriori de que ω_i é a classe de $\boldsymbol{x}$ é meramente a fração de amostras dentro da região que são rotuladas como ω_i

Consequentemente, para uma taxa de mínimo erro, precisamos selecionar a classe mais frequentemente representada dentro da região

K NEAREST NEIGHBOR: REGRA

1. A partir dos N vetores de referência/treinamento, identificar os k vizinhos mais próximos, desconsiderando os rótulos das classes

2. Das k amostras selecionadas, identificar o número de vetores, k_i , que pertencem a classe i, onde $i = 1, 2, ..., M$
   - $\sum_i k_i = k$

3. Atribuir $\boldsymbol{x}$ a classe i com o maior número k_i de amostras

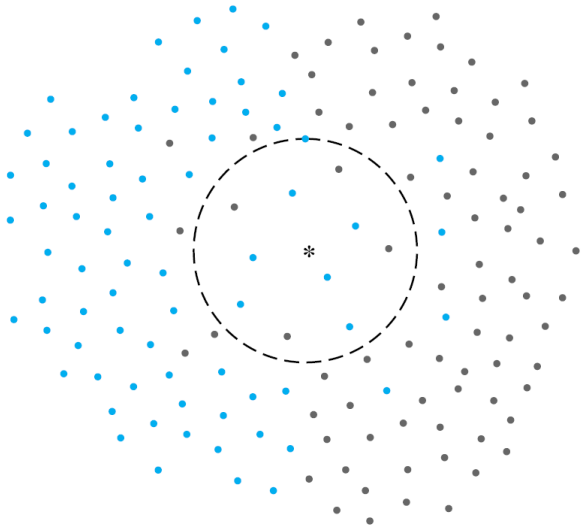
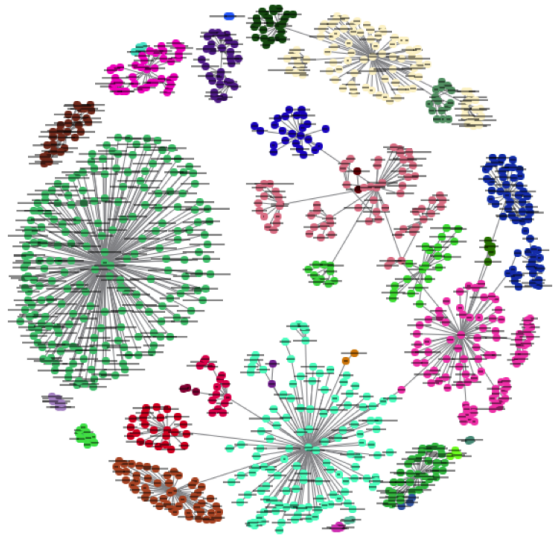


FIGURE 2.25

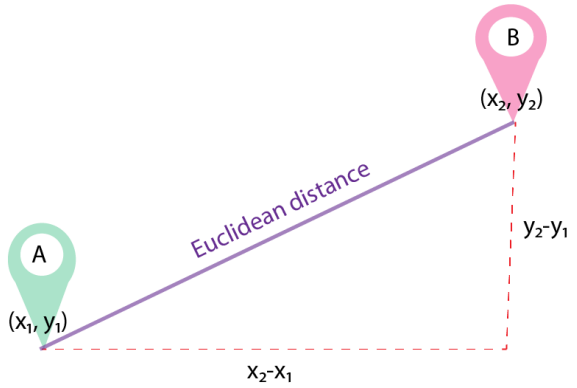
Using the 11-NN rule, the point denoted by a "star" is classified to the class of the red points. Out of the eleven nearest neighbors seven are red and four are black. The circle indicates the area within which the eleven nearest neighbors lie.

K NEAREST NEIGHBOR

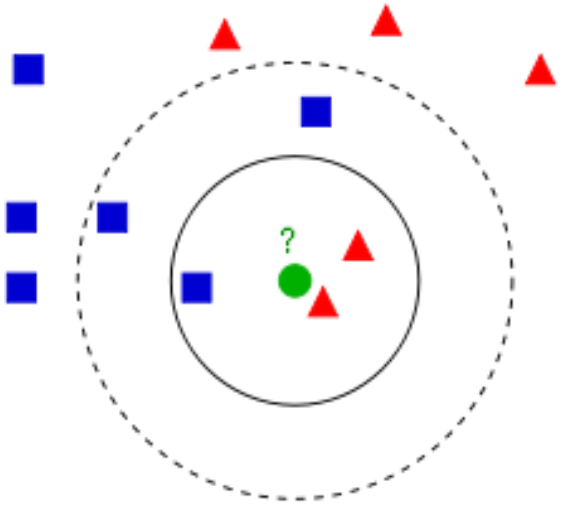
Para utilizar o k-NN é necessário



Dados de referência
(= treinamento aprendizado supervisionado)



Medida de Distância
(volume da distribuição)



Número de Vizinhos K

K NEAREST NEIGHBOR: SELEÇÃO DA DISTÂNCIA

A distância mais utilizada é a Euclideana

$$d = \sqrt{\sum_{i=1}^{x} (x_i - y_i)^2}$$

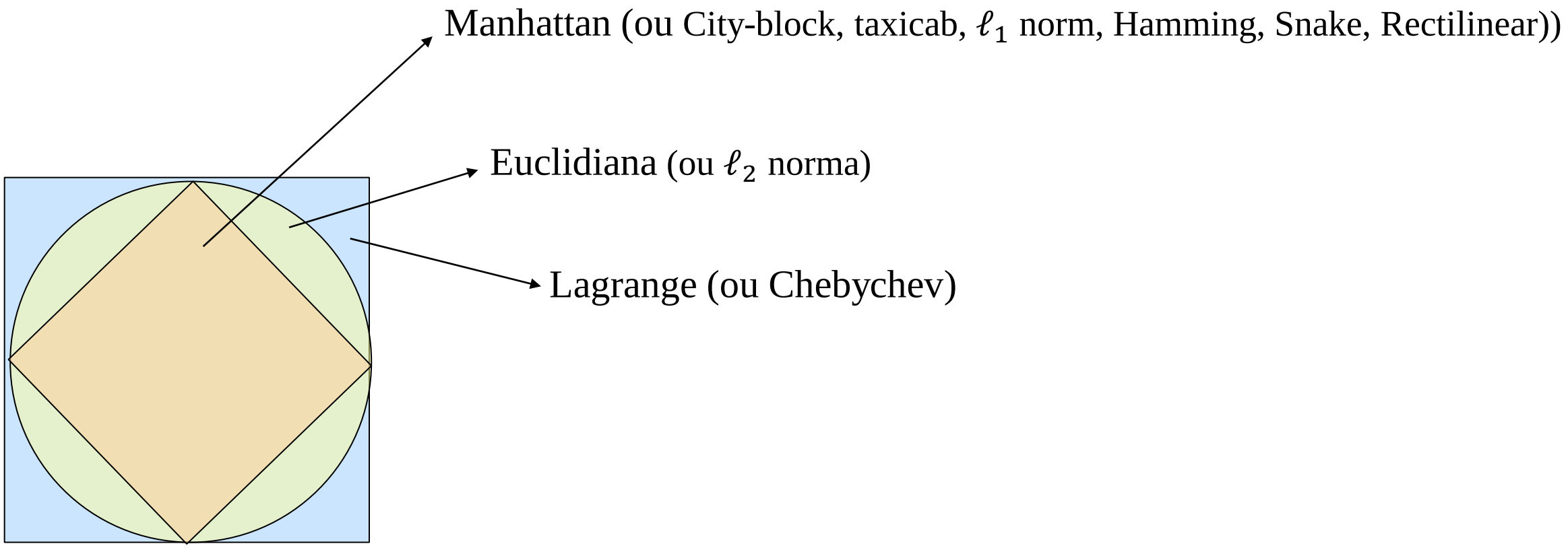
Como esta distância trata todas as variáveis igualmente (mesma importância), é importante realizar uma normalização nas características a fim de a regra k-NN é independente de unidades de medida

K NEAREST NEIGHBOR: MEDIDAS DE DISTÂNCIAS

Table 4.1 Distance metrics assessed by Todeschini.

Cosine metric	$d(\boldsymbol{x},\boldsymbol{y}) = \frac{\boldsymbol{x}.\boldsymbol{y}}{\lvert\boldsymbol{x}\rvert\lvert\boldsymbol{y}\rvert} = \frac{\sum_{i=1}^{d} x_i y_i}{\sqrt{\sum_{i=1}^{d} x_i^2}\sqrt{\sum_{i=1}^{d} y_i^2}}$
Canberra metric	$d(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{d}\sum_{i=1}^{d} \frac{\lvert x_i - y_i\rvert}{x_i + y_i}$
Euclidean metric	$d(\boldsymbol{x},\boldsymbol{y}) = \lvert\boldsymbol{x}-\boldsymbol{y}\rvert = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$
Lagrange metric	$d(\boldsymbol{x},\boldsymbol{y}) = \max_{i=1,\ldots,d} \lvert x_i - y_i\rvert$
Lance–Williams metric	$d(\boldsymbol{x},\boldsymbol{y}) = \frac{\sum_{i=1}^{d} \lvert x_i - y_i\rvert}{\sum_{i=1}^{d}(x_i + y_i)}$
Manhattan metric	$d(\boldsymbol{x},\boldsymbol{y}) = \sum_{i=1}^{d} \lvert x_i - y_i\rvert$

K NEAREST NEIGHBOR: VOLUME DAS MEDIDAS DE DISTÂNCIAS



K NEAREST NEIGHBOR: NORMALIZAÇÃO

Como a variável dominante (maior intervalo) pode não ser a variável mais discriminante, pode-se utilizar alguma normalização dos dados

Table 4.2 Standardisation approaches assessed by Todeschini. μ_j and σ_j are the training data estimates of the mean and standard deviation of the jth variable. U_j and L_j are the upper and lower bounds of the jth variable. $y_{i,j}$ is the jth variable of the ith training data example.

Autoscaling	$x'_j = \frac{x_j - \mu_j}{\sigma_j}$
Maximum scaling	$x'_j = \frac{x_j}{U_j}$
Range scaling	$x'_j = \frac{x_j - L_j}{U_j - L_j}$
Profiles	$x'_j = \frac{x_j}{\sqrt{\sum_{i=1}^{n}\left(y_{i,j}^2\right)}\sqrt{\sum_{j=1}^{d}\left(x_j^2\right)}}$

K NEAREST NEIGHBOR: SELEÇÃO DO K

Não é um problema trivial

- k deve ser grande para minimizar o erro (k muito pequeno leva a fronteiras ruidosas)
- k deve ser pequeno para que somente exemplos próximos sejam incluídos

Encontrar o balanço não é uma coisa trivial

- Base de validação
  - Divide os dados de "treinamento" em 2 grupos: referência e validação
  - Realiza diversos experimentos para diferentes k (usando referência)
  - Seleciona o k que produzir o melhor resultado nos dados de validação

Algumas orientações

- K deve ser ímpar para um problema de duas classes
- Não deve ser um número múltiplo do M classes

K NEAREST NEIGHBOR: VANTAGENS

K-NN é muito fácil de entender e de implementar

Não existe modelo propriamente dito ou treinamento

- O treinamento consiste em armazenar todas as amostras de treinamento, isto é, o modelo são as próprias amostras

Capacidade de trabalhar com dados categóricos

Permite evoluir com a adição de novos dados

K NEAREST NEIGHBOR: DESVANTAGENS

Considerando que precisamos de um *n* grande para o k-NN funcionar bem, a complexidade (ineficiência) torna-se problema

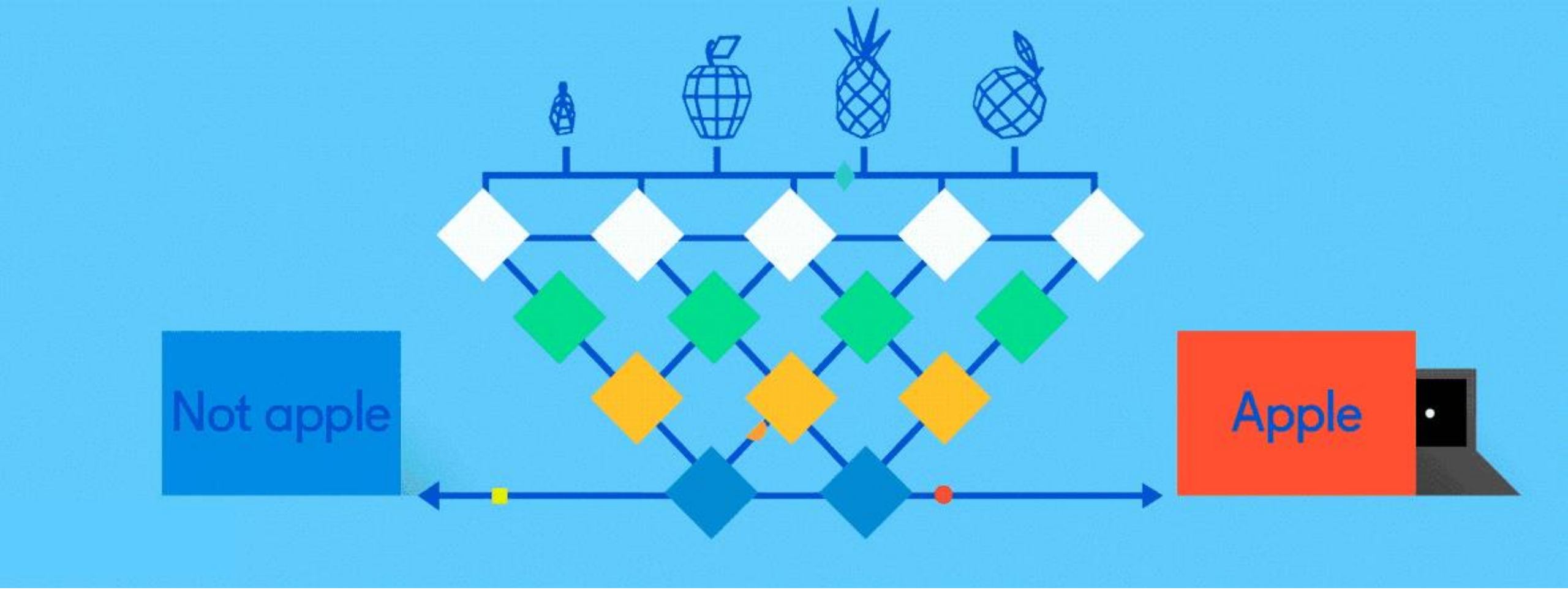
- O(n) e a complexidade para encontrar o vizinho mais próximo
- O(nk) complexidade para encontrar k exemplos mais próximos

Muito sensível aos atributos irrelevantes, faltantes e ruído (dependência forte dos dados de treinamento)

Depende muito da função de distância

A constante k usada para definir o número de vizinhos é obtida por tentativa e erro

Não trabalha muito bem com dados com alta dimensionalidade (cálculo da distância)

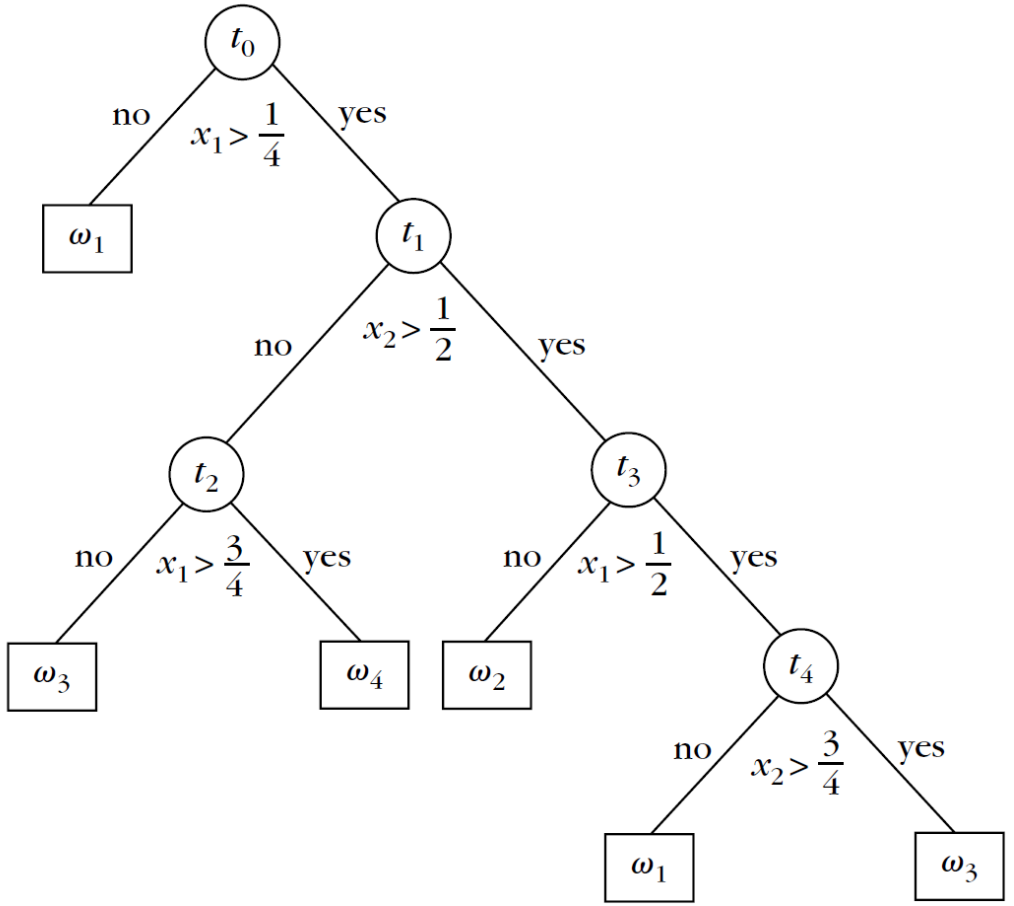


ÁRVORES DE DECISÃO

ÁRVORES DE DECISÃO

Árvores de decisão são modelos muito simples que representam a regra de decisão como uma árvore

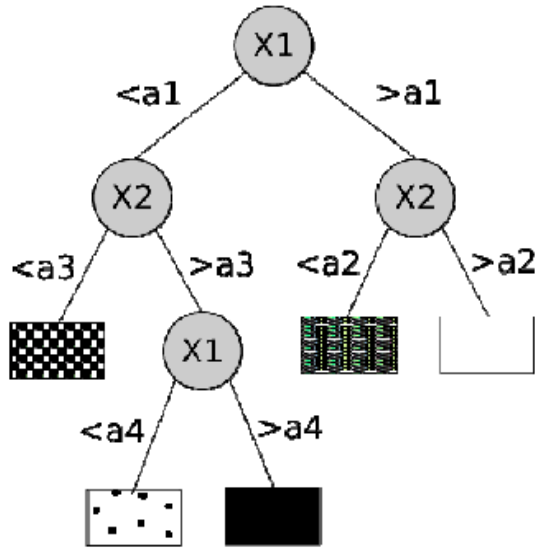
- Nodos rotulados como características
- Ramos (ou arestas) rotuladas como valores (ou conjunto de valores) das características
- Folhas rotuladas como as classes



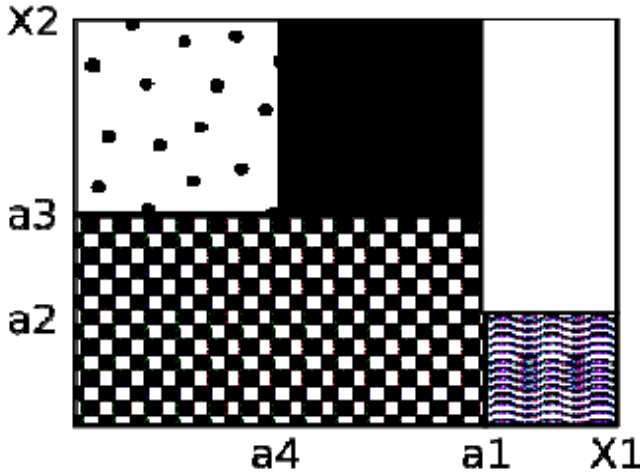
ÁRVORES DE DECISÃO

Realiza um processo de decisão multiestágio, i.e., em vez de utilizar o conjunto completo de características de uma vez só para decidir, diferentes subconjuntos de características são utilizados em diferentes níveis da árvore

- Classes são rejeitadas sequencialmente até que obtenha-se uma única classe aceita



O espaço de características é dividido em regiões únicas, as quais correspondem às classes, de maneira sequencial



ÁRVORES DE DECISÃO

Ao receber um vetor de características, a busca pela região a qual o vetor será atribuído é obtido via uma sequência de decisões por um caminho de nodos de uma árvore

- Vantajoso para problemas com muitas classes

O número de decisões necessárias para classificar um padrão depende do padrão

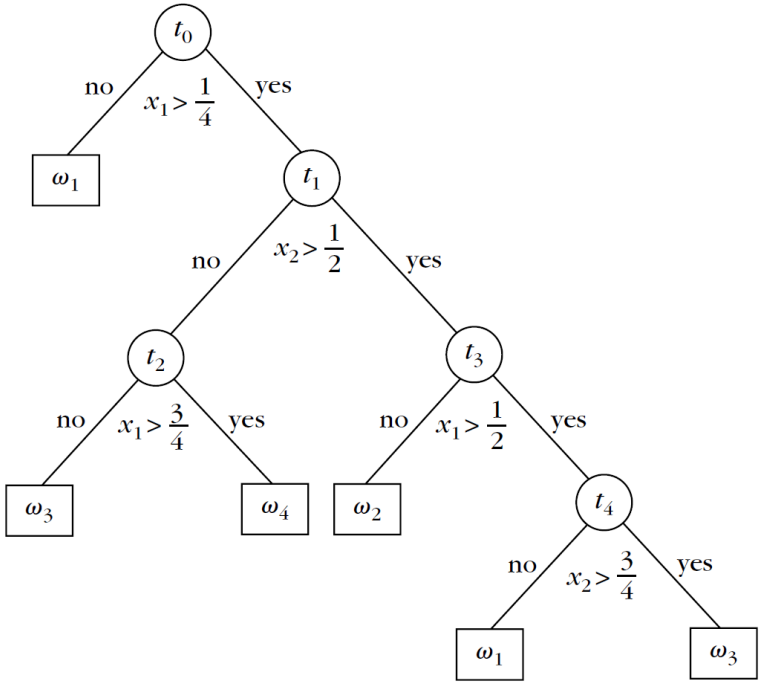
As regras da árvore determinam automaticamente aquelas características importantes para a classificação

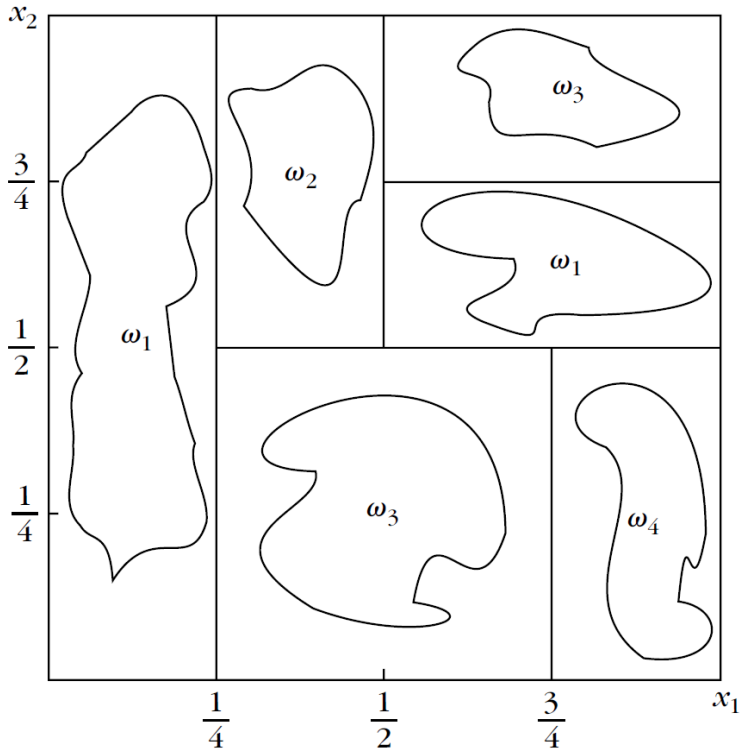
São capazes de modelar fronteiras de decisão não-linear complexas

ÁRVORES DE DECISÃO

As árvores de decisão mais populares são aquelas que dividem o espaço em hiperplanos paralelos aos eixos

- Questões do tipo "a característica $x_i \leq \alpha$?", onde α é um limiar





ÁRVORES DE DECISÃO: CONSTRUÇÃO

A construção da árvore tem por objetivo determinar os testes de determinadas características nos nodos internos e a classe a ser atribuída o vetor de características nos nodos folhas

A construção envolve 3 passos

1. **Selecionar uma regra de divisão para cada nodo interno:** determinar as características junto com um método para particionar os valores que aquelas características podem assumir

2. **Determinar quais nodos são nodos terminais (folhas):** para cada nodo, deve-se definir se continua dividindo ou torna o nodo como terminal e atribui uma classe a ele

3. **Atribuir rótulos das classes aos nodos terminais:** rótulos são atribuídos aos nodos terminais minimizando o erro de classificação

ÁRVORES DE DECISÃO: SELECIONAR REGRA DE DECISÃO

O objetivo é dividir sucessivamente os dados em subconjuntos que são mais puros, i.e., os subconjuntos produzidos pela divisão são mais homogêneos comparado com o conjunto original

Uma regra de divisão é uma prescrição para decidir qual variável, ou combinações de variáveis , devem ser utilizadas em um nodo para dividir as amostras de dados em subconjuntos e para decidir como os valores que a variável pode assumir devem ser particionados

Depende da natureza da característica

- Binária: existem dois possíveis valores
- Nominal: produz um nodo filho para cada valor, produzindo assim um caminho para cada valor na árvore
- Ordinal: semelhante ao nominal, mas valores podem ser agrupados (mantendo a ordem)
- Contínua: utiliza-se um limiar para dividir ($x_1 \leq 4.0$ ou $x_2 + x_4 \leq 2.0$)

ÁRVORES DE DECISÃO: SELECIONAR REGRA DE DECISÃO

Para achar a melhor divisão dos dados que estão no subespaço $u(t)$ no nodo t, utiliza-se medidas de impureza do nodo, as quais são representadas em termos de $p(\omega_j|\boldsymbol{x} \in u(t))$, i.e., probabilidade de um padrão $\boldsymbol{x}$ pertencer a classe ω_j, dados que ele está no nodo t

$$p(\omega_j|\boldsymbol{x} \in u(t)) = p(\omega_j|t) \backsim \frac{N_j(t)}{N(t)}$$

onde $N_j(t)$ é o número de amostras para as quais $\boldsymbol{x}_i \in u(t)$ e $y_i = \omega_j$ e $N(t)$ é o numero de amostras no nodo t

Funções de Impureza do Nodo

- **Gini** $\mathcal{I}(t) = 1 - \sum_i [p(\omega_i|t)]^2$
- Entropia $\mathcal{I}(t) = -\sum_i p(\omega_i|t) ln\big(p(\omega_i|t)\big)$
- Erro da Classificação $\mathcal{I}(t) = 1 - \max_i [p(\omega_i|t)]$

ÁRVORES DE DECISÃO: TERMINAR O PROCESSO DE DIVISÃO

Se continuar dividindo, certamente cada nodo terminal terá somente 1 classe, mas é provável que obterá uma árvore grande que descreve nos mínimos detalhes dos dados (*overfitting*) afetando assim a generalização

Abordagens

- Regra de parada: não divide o nodo se a mudança na função de impureza é menor do que um limiar pré-definido (mas como especificar o limiar? A divisão pode resultar em uma redução muito grande da impureza)
- Poda: a árvore é construída até os nodos terminais serem puros (ou quase puro) e depois podar os nodos (substituído pela classe do novo nodo terminal) utilizando alguma medida (como por exemplo, erros de classificação)

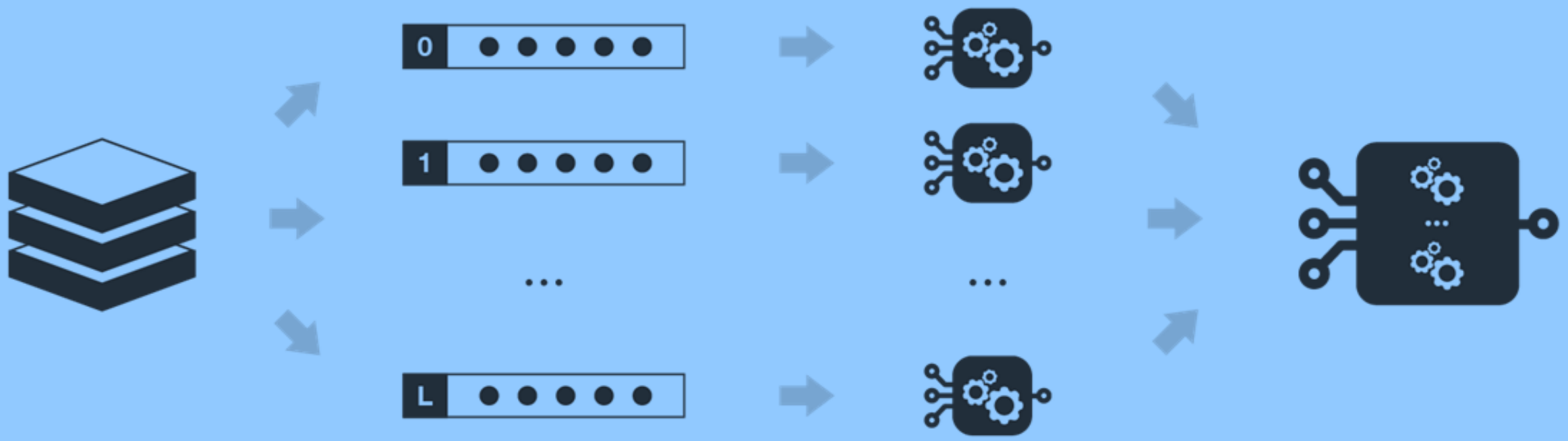
ÁRVORES DE DECISÃO: VANTAGENS X DESVANTAGENS

Vantagens

- Árvores são muito fáceis de serem explicadas
- Alguns alegam que são mais próximas ao processo de decisão de humanos
- Árvores podem ser mostradas graficamente e facilmente interpretadas por uma pessoa leiga
- Árvores podem lidar com variáveis qualitativas sem a criação de representações numéricas (dummy)

Desvantagem

- Árvores geralmente não possuem o mesmo nível de acurácia que os outros classificadores vistos até agora



MÉTODOS ENSEMBLE

MÉTODOS ENSEMBLE

O método ensemble é uma técnica de aprendizado de máquina que combina o resultado de múltiplos modelos em busca de produzir um melhor modelo preditivo

O método combina modelos mais fracos, que quando aplicados separadamente não apresentam o resultado desejado

A combinação dos modelos reduzem a variância de um único modelo

MÉTODOS ENSEMBLE

Bagging: utilizando bootstrap, diversas árvores são construídas com base nos conjuntos de treinamentos criados. O resultado é obtido através da média das previsões (reduz a variância)

Random Forest: semelhante ao bagging, mas na divisão nem todas as variáveis são consideradas (é como se fizesse uma descorrelação das árvores)

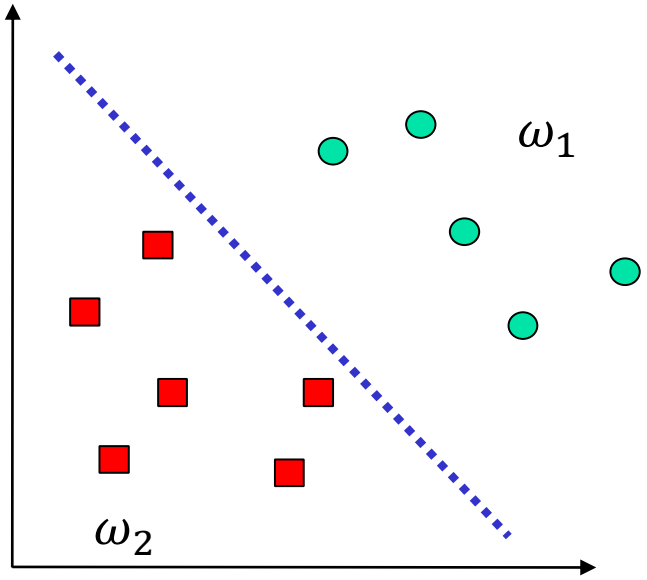
Boosting: as árvores são construídas com base no resultado da árvore anterior (erros de classificação)

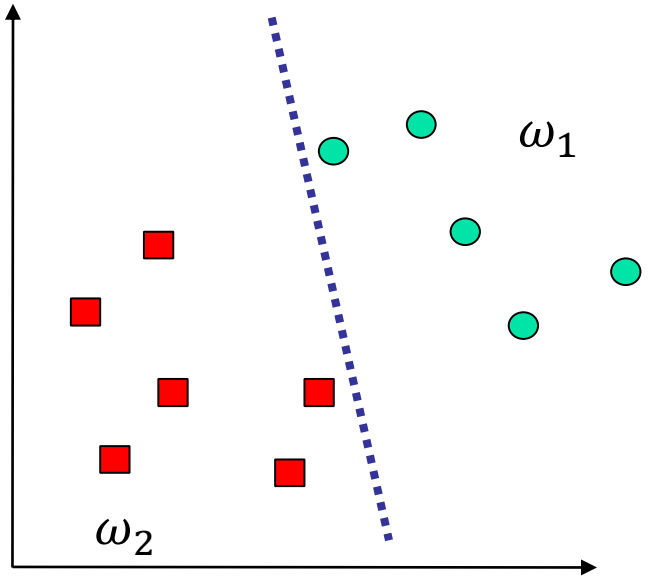
MÁQUINA DE VETOR DE SUPORTE

MÁQUINA DE VETOR DE SUPORTE

Considere a tarefa de classificação binária de 2 classes ω_1 e ω_2 linearmente separáveis

Existem diversas fronteiras que podem ser utilizadas para separá-las





Qual delas escolher?

MÁQUINA DE VETOR DE SUPORTE

Podemos resolver o problema utilizando a função discriminante linear

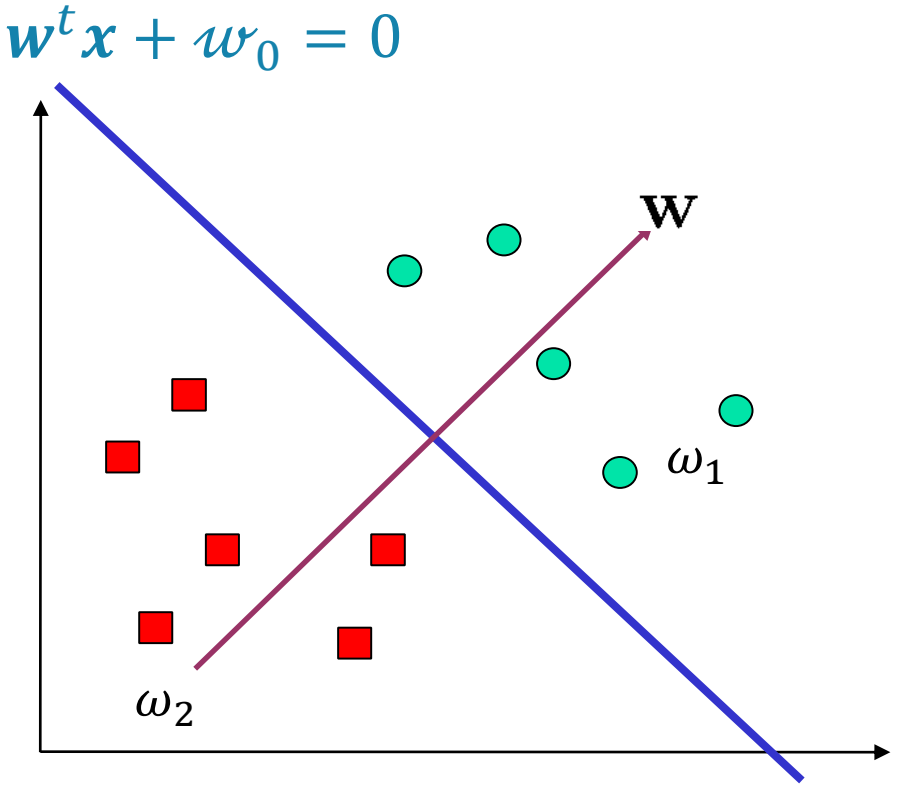
$$g(\boldsymbol{x}) = \boldsymbol{w}^t \boldsymbol{x} + w_0 = 0$$

com a seguinte regra de decisão

$$\boldsymbol{w}^T \boldsymbol{x} + w_0 \begin{cases} > 0 \\ < 0 \end{cases} \Rightarrow \boldsymbol{x} \in \begin{cases} \omega_1, y_i = +1 \\ \omega_2, y_i = -1 \end{cases}$$

Assim todos os vetores são classificados corretamente se

$$y_i(\boldsymbol{w}^T \boldsymbol{x} + w_0) > 0 \quad \text{para todo } i$$

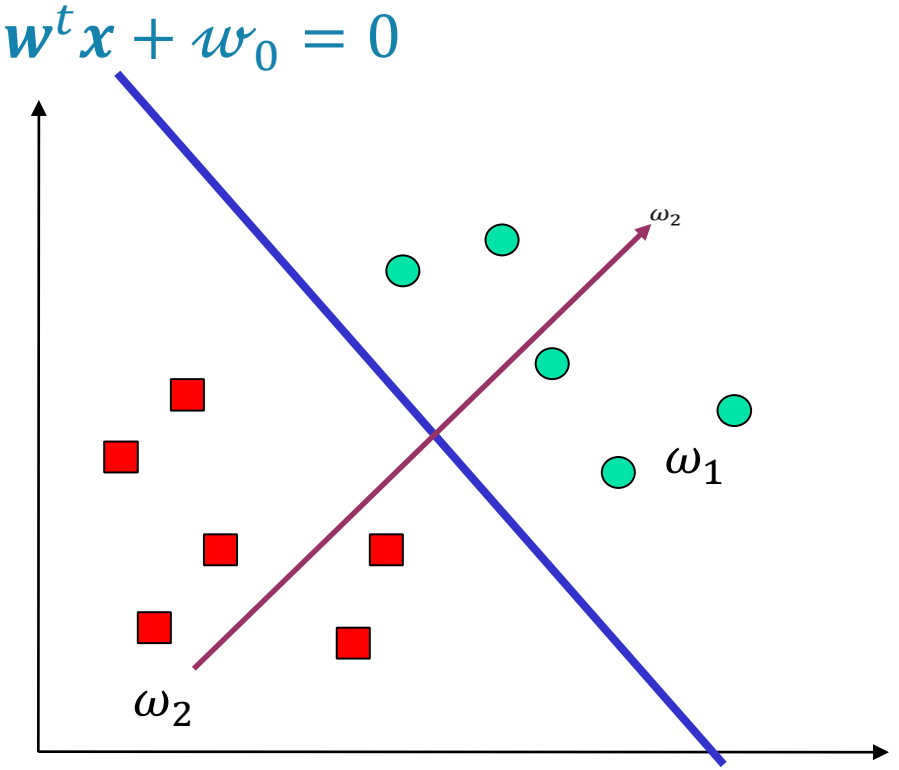


MÁQUINA DE VETOR DE SUPORTE

A fim de generalizar a solução, $g(\boldsymbol{x})$ é um hiperplano de separação no espaço de características de *d* dimensões

O hiperplano é caracterizado por sua direção (determinado por $\boldsymbol{w}$) e sua exata posição no espaço (determinado por w_0)

O vetor $\boldsymbol{w}$ é ortogonal ao hiperplano de separação



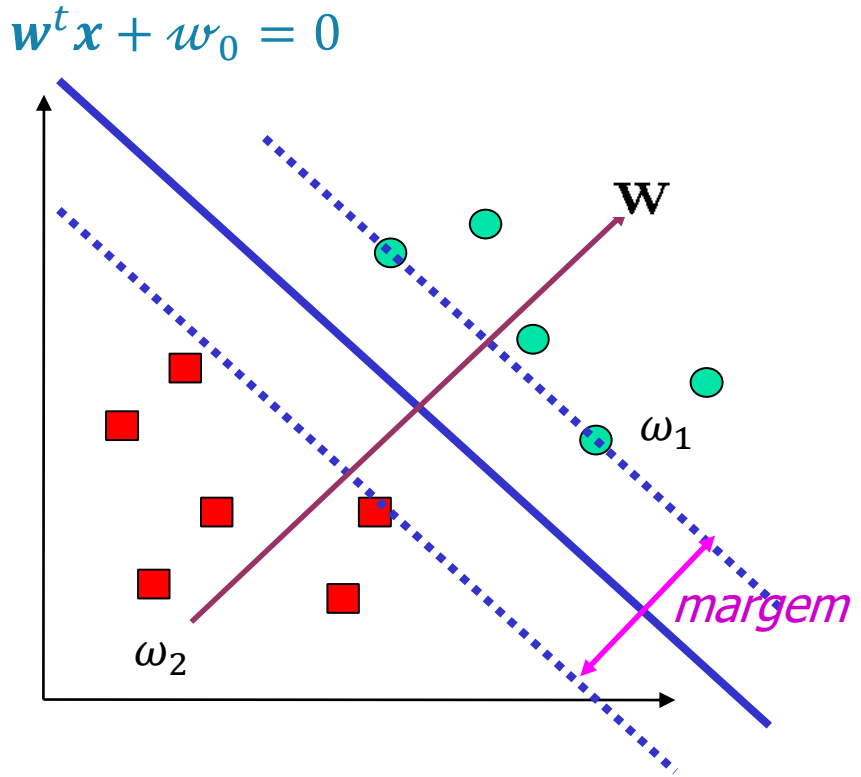
MÁQUINA DE VETOR DE SUPORTE

Como existem diversos muitos hiperplanos possíveis, é razoável selecionar um hiperplano de separação que tem a maior distância ao ponto mais próximo da classe ω_1 e o mais próximo da classe ω_2

- A soma das distâncias do hiperplano aos pontos mais próximos das classes ω_1 e ω_2 é chamada de **margem**

O objetivo então é achar a direção que produz a margem máxima possível, e por isso este tipo de classificador é chamado de classificador de margem máxima

- Quando maior a margem, melhor é a generalização!



MÁQUINA DE VETOR DE SUPORTE

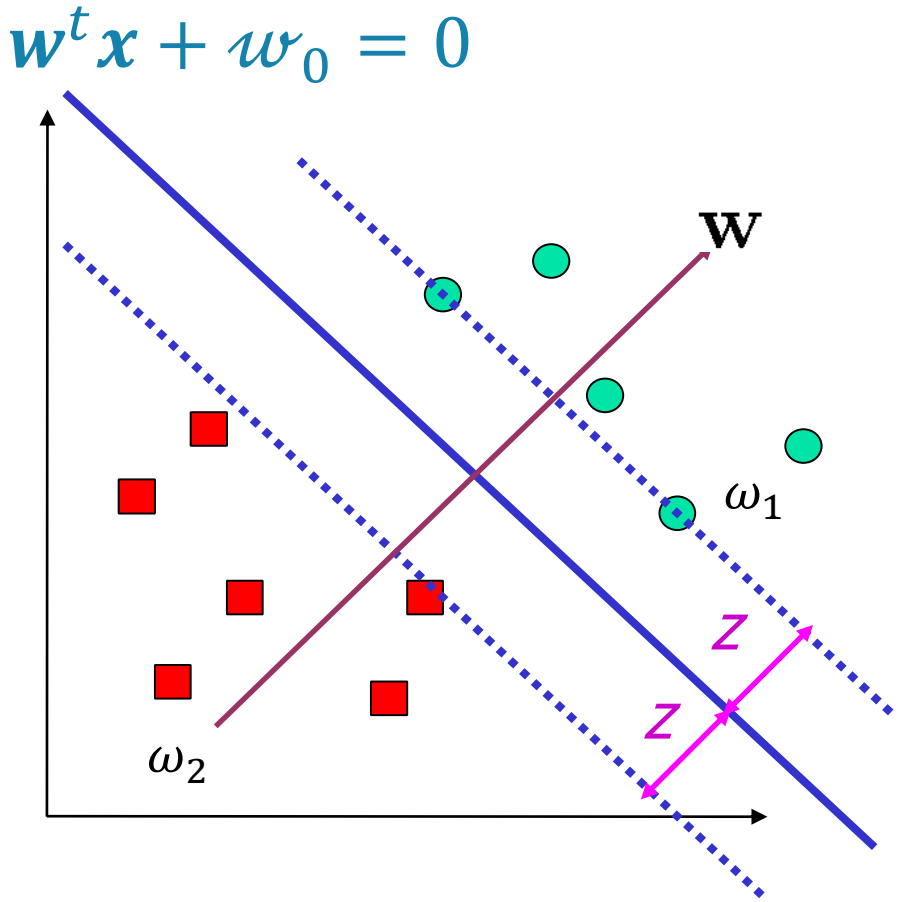
Como não queremos beneficiar nenhuma classe, é razoável selecionar um hiperplano que tem a mesma distância (z) dos respectivos pontos das classes ω_1 e ω_2

A distância de um ponto ao hiperplano

$$z = \frac{|g(\boldsymbol{x})|}{\|\boldsymbol{w}\|}$$

onde $|g(\boldsymbol{x})|$ é a distância euclidiana entre o ponto e o hiperplano

Agora o vetor solução deve ser achado de tal maneira que o valor de $g(\boldsymbol{x})$, nos vetores mais próximos em ω_1 e ω_2, é igual a 1 para ω_1 e -1 para ω_2



MÁQUINA DE VETOR DE SUPORTE

Assim a margem do hiperplano $g(\boldsymbol{x})$ é dada por

$$\frac{1}{||\boldsymbol{w}||} + \frac{1}{||\boldsymbol{w}||} = \frac{2}{||\boldsymbol{w}||}$$

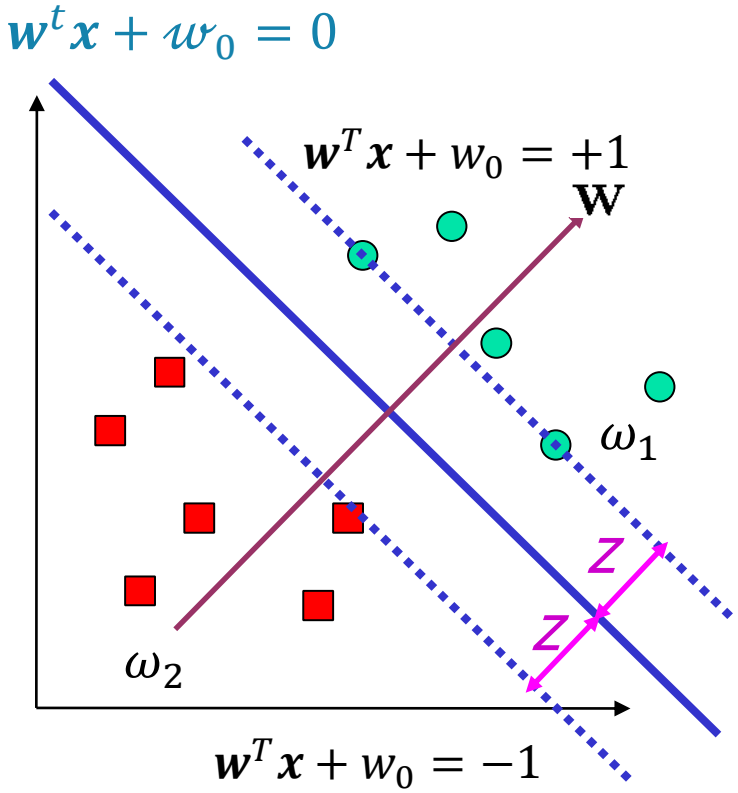
desde que

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 \geq +1, \forall \boldsymbol{x} \in \omega_1$$
$$\boldsymbol{w}^T\boldsymbol{x} + w_0 \leq -1, \forall \boldsymbol{x} \in \omega_2$$

Os vetores mais próximos do hiperplano de separação, chamados de vetores de suporte, em uma combinação linear definem os hiperplanos canônicos

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = +1$$

$$\boldsymbol{w}^T\boldsymbol{x} + w_0 = -1$$

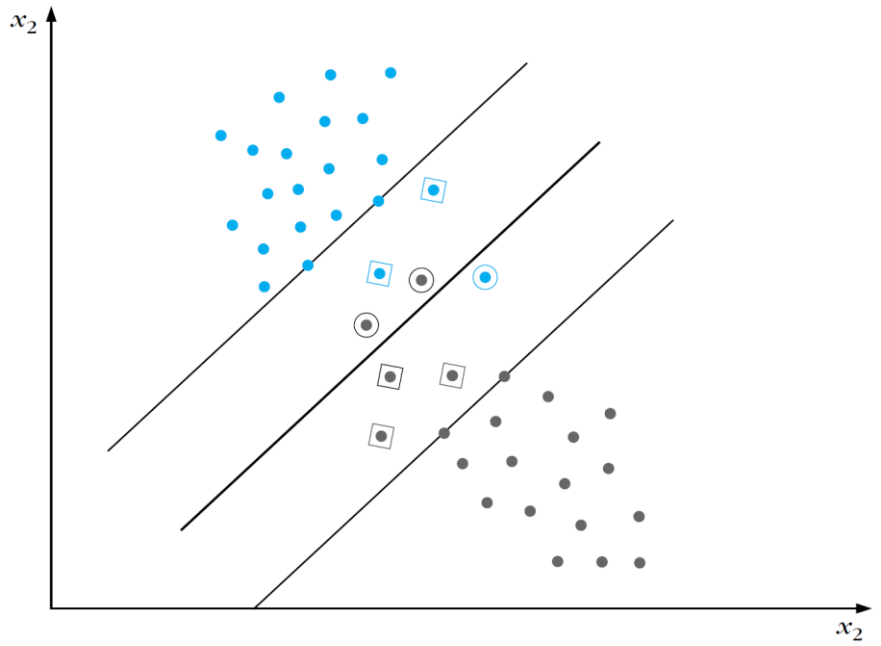


MÁQUINA DE VETOR DE SUPORTE

Os vetores de suporte constituem os elementos críticos do conjunto de treinamento

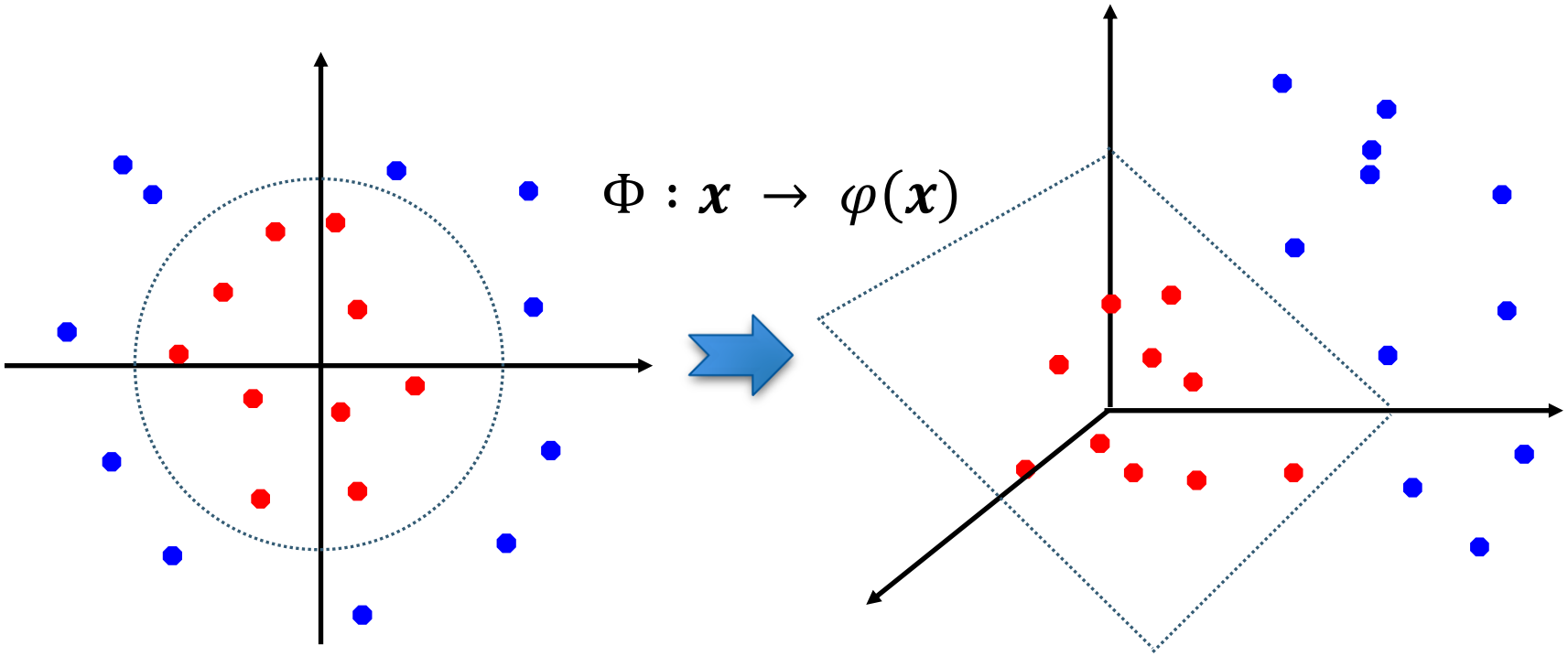
O classificador com hiperplano ótimo de uma máquina de vetor de suporte é único

Entretanto, quando as classes não são separáveis, a ideia do hiperplano ótimo não é mais válido



MÁQUINA DE VETOR DE SUPORTE – NÃO LINEAR

A ideia é mapear o espaço de características original em um espaço de maior dimensão onde os vetores das classes são separáveis



MÁQUINA DE VETOR DE SUPORTE – NÃO LINEAR

O classificador linear é baseado em um produto escalar entre os vetores
$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i{}^t \, \boldsymbol{x}_j$

Se cada vetor for mapeado em um espaço com mais dimensões via alguma transformação, $\Phi : \boldsymbol{x} \rightarrow \varphi(\boldsymbol{x})$, o produto escalar

$$K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \varphi(\boldsymbol{x}_i)^t \varphi(\boldsymbol{x}_j)$$

Uma função kernel é alguma função que corresponde a um produto interno em algum espaço de características expandido

MÁQUINA DE VETOR DE SUPORTE – NÃO LINEAR

Exemplos de funções kernel

- **Linear**

$$K(x_i, x_j) = {x_i}^t \, x_j$$

- **Polinomial de grau p**

$$K(x_i, x_j) = (1 + {x_i}^t \, x_j)^p$$

- **Radial Basis (Gaussiana)**

$$K(x_i, x_j) = e^{-\frac{\|x_i - x_j\|^2}{2\sigma^2}}$$

- Sigmoidal

$$K(x_i, x_j) = \tanh \beta_0 {x_i}^t \, x_j + \beta_1$$

MÁQUINA DE VETOR DE SUPORTE

Propriedades

- Flexibilidade em escolher a função de similaridade
- Consegue lidar com grandes conjuntos de dados
  - Somente vetores de suporte são utilizados para especificar o hiperplano de separação
- Habilidade de lidar com espaço de características de grandes dimensões
  - Complexidade não depende da dimensionalidade do espaço de características
- Modificações na otimização pode controlar o overfitting

Fraquezas

- É sensível ao ruído
- Considera somente 2 classes