Classificação

ACH5504 – Mineração de Dados

Notas de aulas baseadas no livro

"*Introduction to Data Mining*"

Tan, Steinbach, Karpatne, Kumar

Resumo

- Definição de classificação
- Técnicas de classificação
- Métodos baseados na arvores de decisão
- Vizinhos mias próximos (k-Nearest-Neighbor)

Definição de classificação

- Para uma coleção de registros (conjunto de treinamento)
  - Cada registro é caracterizado por uma tupla $(\boldsymbol{x}, y)$, onde $\boldsymbol{x}$ é o conjunto de atributos e y é o rótulo da classe
    - $\boldsymbol{x}$: atributo, preditor, variável independente, entrada
    - y: classe, resposta, variável dependente, saída
- Tarefa:
  - Aprender um modelo que mapeia cada atributo definido como $\boldsymbol{x}$ em um dos rótulos de classe predefinidos y

Exemplos de tarefa de classificação

Tarefa	Conjunto de atributos, $\boldsymbol{x}$	Rótulo da classe, y
Categorizando mensagens de e-mail	Atributos extraídos do cabeçalho e do conteúdo da mensagem de e-mail	spam ou não spam
Identificando células tumorais	Características extraídas das varreduras de MRI	células malignas ou benignas
Catalogação de galáxias	Características extraídas das imagens do telescópio	Galáxias elípticas, espirais ou irregulares

Tarefa de classificação

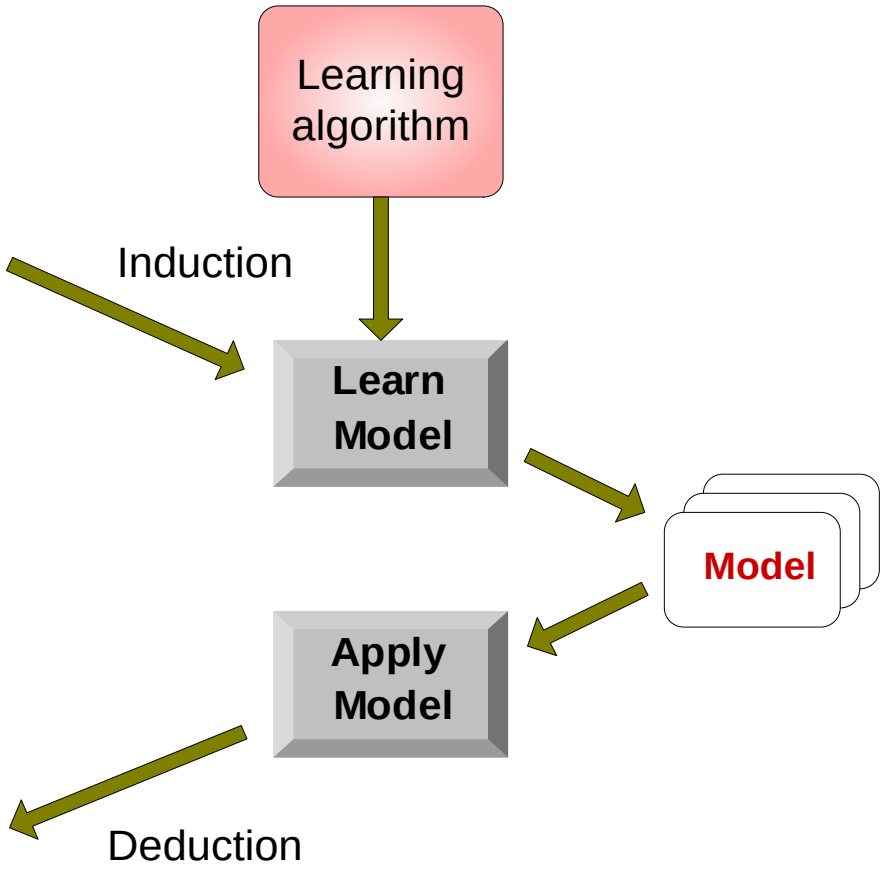
- Construção de modelo
  - Com base no conjunto do treino, um modelo (regras, árvore de decisão, fórmula matemática) é construído
  - Aprendizado supervisionado (atributo classe)

- Uso do modelo
  - O modelo é utilizado para classificar instâncias (não vistas) do conjunto de teste, estimando a acurácia
  - Acurácia é a percentagem de instâncias corretamente classificadas

Abordagem geral para construir um modelo

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set



Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

Técnicas de classificação

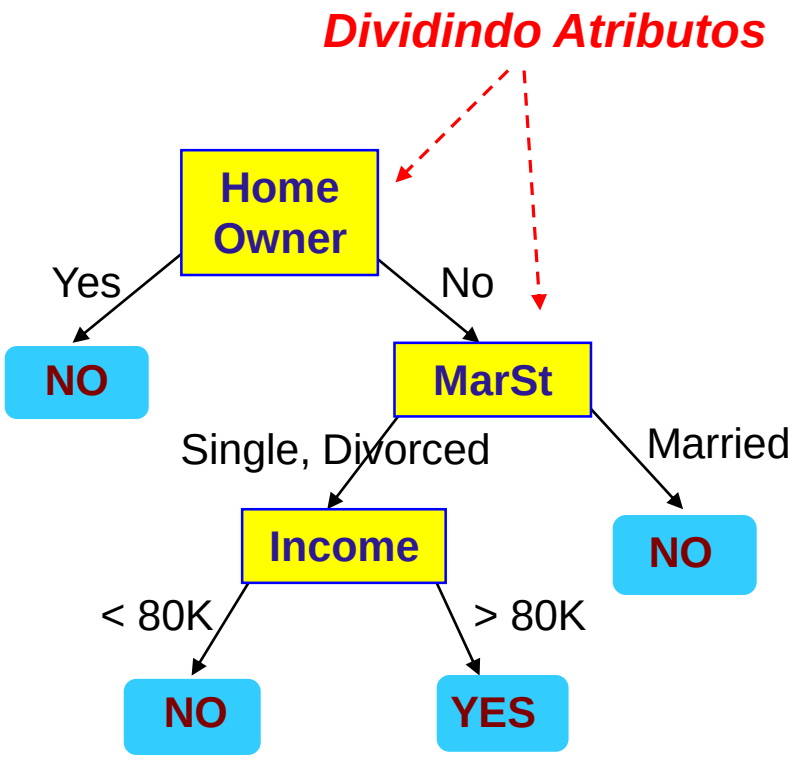
- Classificadores de base
  - Árvore de Decisão
  - Métodos baseados nas regras
  - Nearest-neighbor (Vizinho-mais-próximo)
  - Naïve Bayes
  - Neural Networks
  - Support Vector Machines (SVM)
  - Deep Learning

- Classificadores de Assembleias
  - Boosting, Bagging, Random Forests

Exemplo de uma árvore de decisão

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
	categórico	categórico	contínuo	classe
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

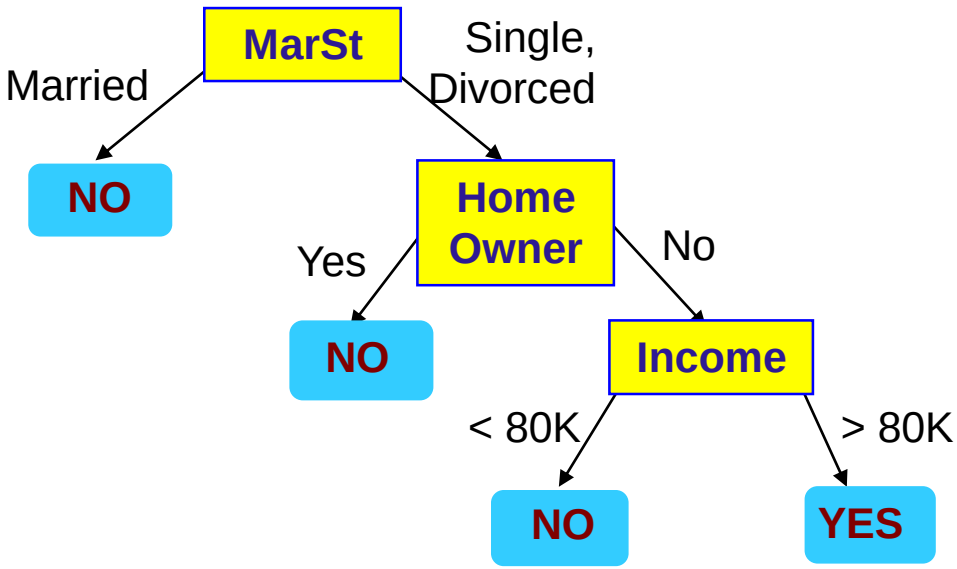
Dados de treino



Modelo: Árvore de decisão

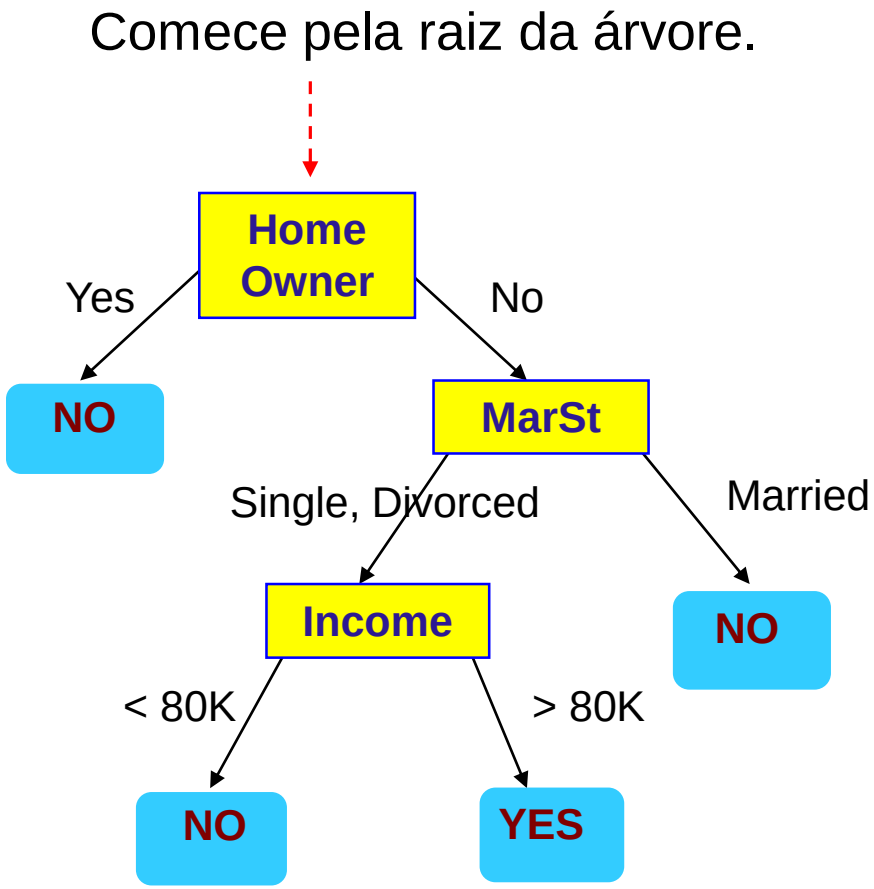
Outro exemplo de árvore de decisão

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
	categórico	categórico	contínuo	classe
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Pode haver mais de uma árvore que se encaixa nos mesmos dados!

Aplicando o modelo para dados de teste



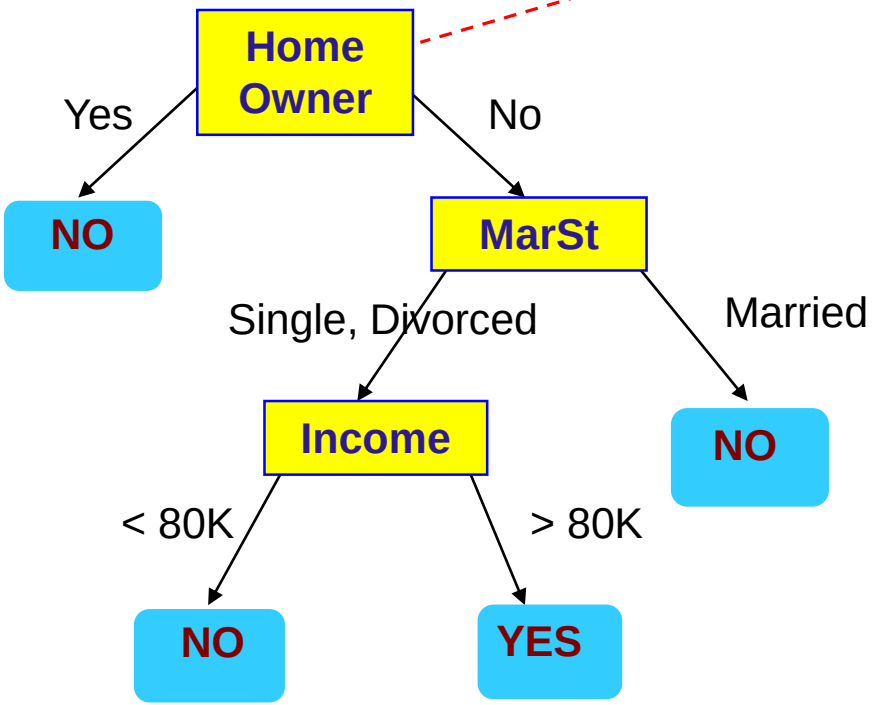
Dados de teste

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?

Aplicando o modelo para dados de teste

Dados de teste

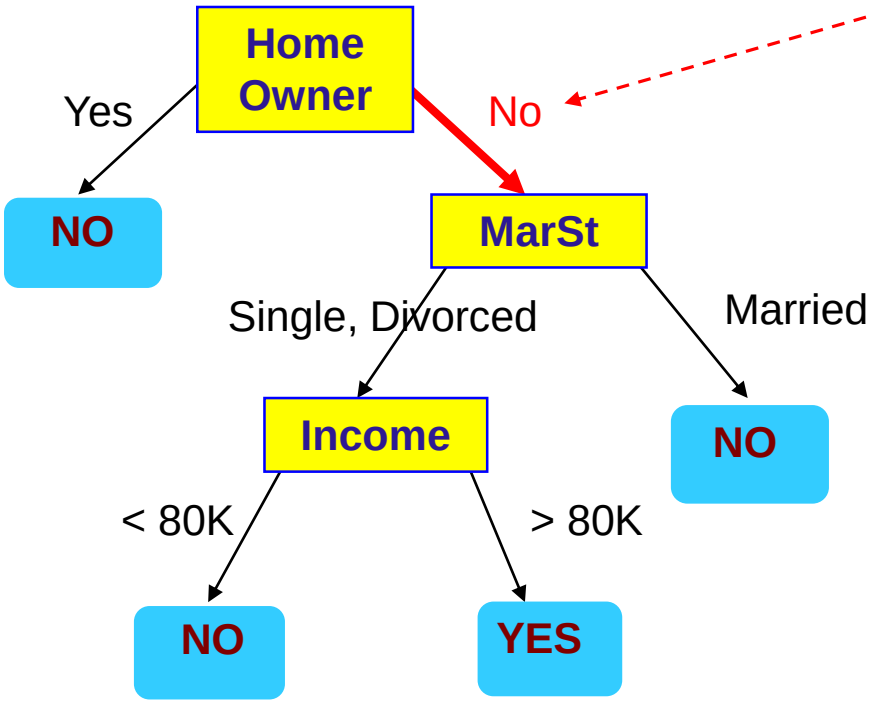
Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



Aplicando o modelo para dados de teste

Dados de teste

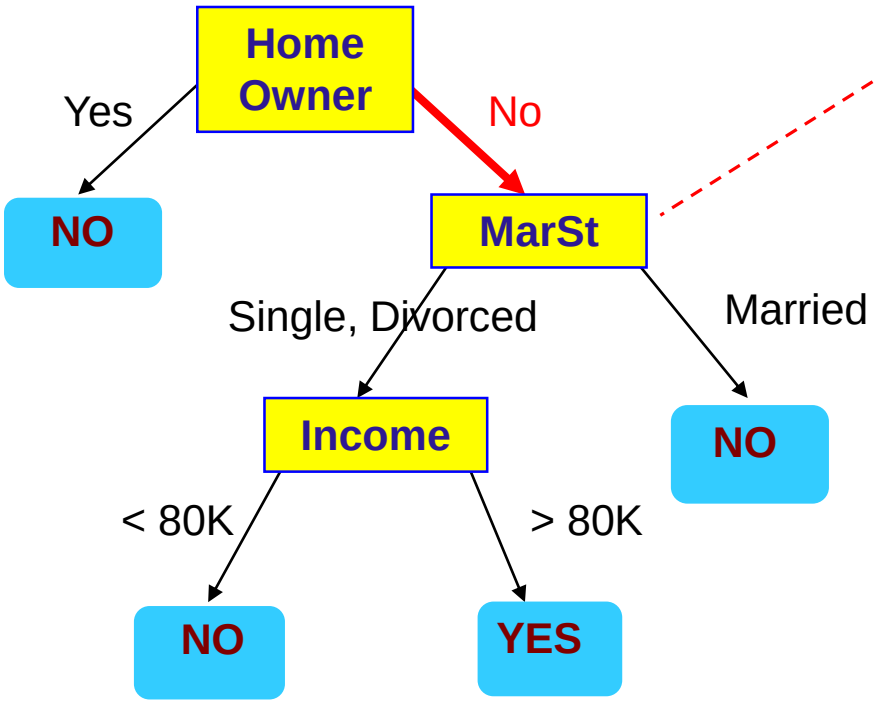
Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



Aplicando o modelo para dados de teste

Dados de teste

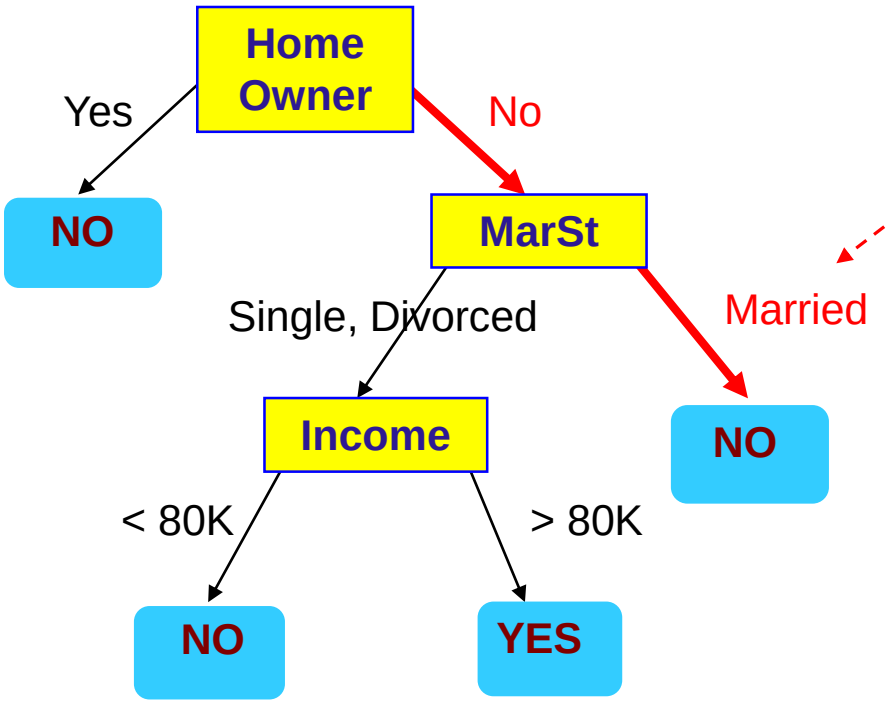
Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



Aplicando o modelo para dados de teste

Dados de teste

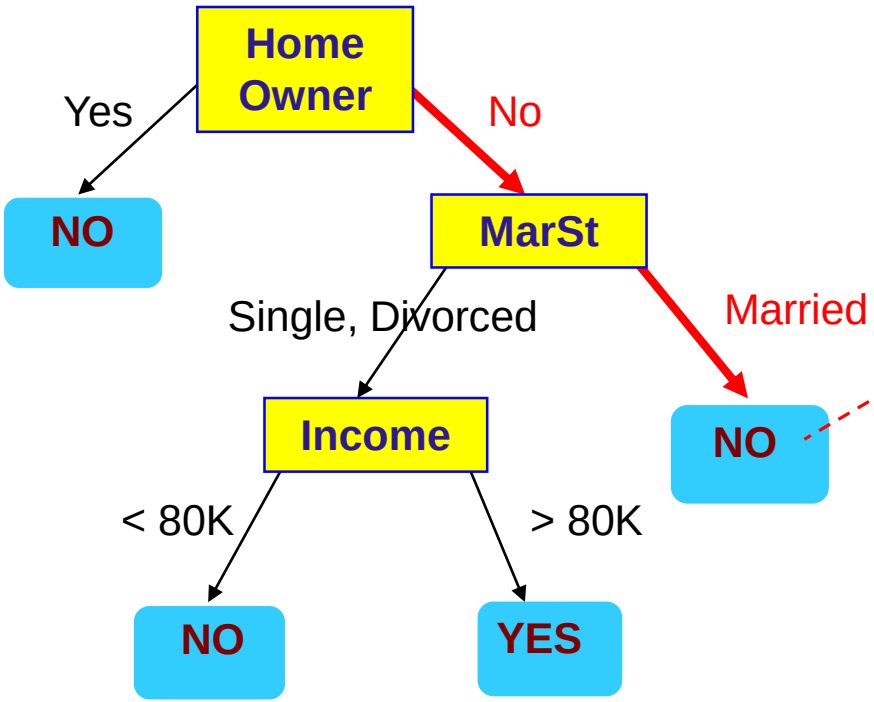
Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



Aplicando o modelo para dados de teste

Dados de teste

Home Owner	Marital Status	Annual Income	Defaulted Borrower
No	Married	80K	?



Atribui a classe “No”

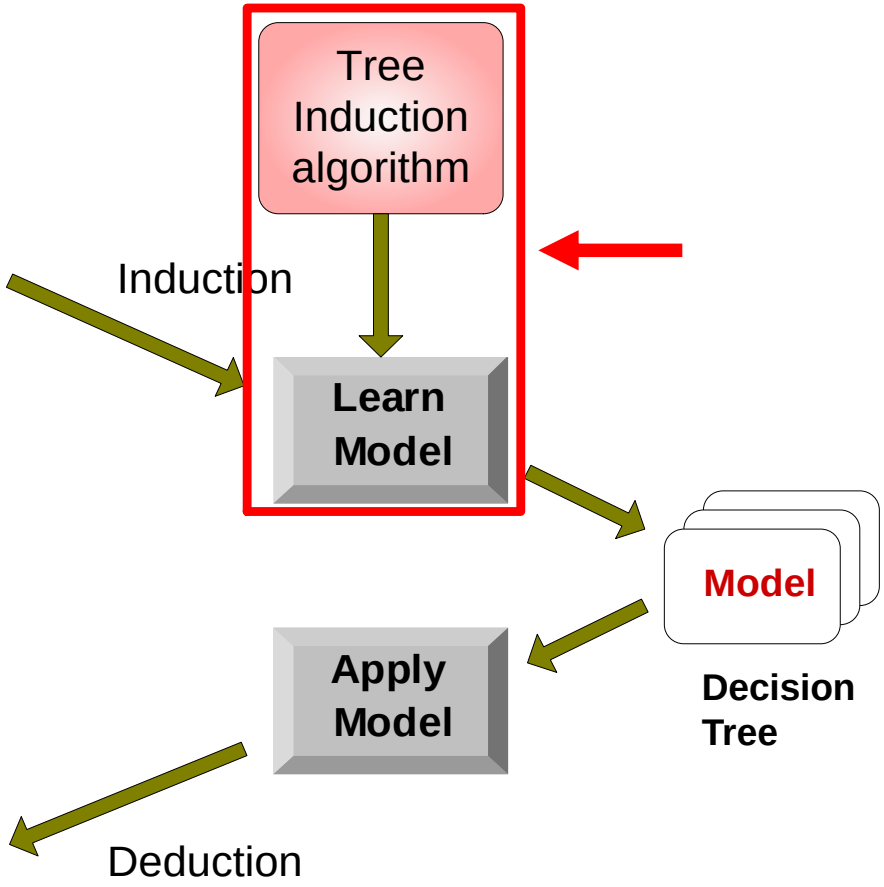
Tarefa de Classificação com Árvore de Decisão

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set

Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set



Indução de Árvore de Decisão

- Árvore em que nós internos (não-folha) são testes em atributos e cada ramo é um resultado do teste e cada nó terminal (folha) é uma classe
- Testes podem ser binários ou multi-valorados
- Dada uma instância de teste, seus atributos são testados a partir da raíz até encontrar um nó folha
- Pode ser convertida para regras de classificação
- Não requer conhecimento do domínio ou determinação de parâmetros
- Pode lidar com dados multi-dimensionais
- São fáceis de interpretar
- Aprendizado e classificação são rápidos e simples

Indução de Árvore de Decisão

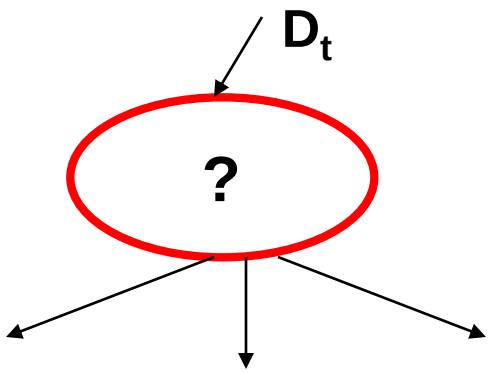
- Algoritmos mais comuns:
  - Algoritmo de Hunt (um dos primeiros)
  - CART
  - ID3, C4.5
  - SLIQ, SPRINT

Estrutura general de algoritmo de Hunt

- Deixe D_t ser o conjunto de registros de treino que atingem um nó t

- Procedimento geral:
  - Se D_t contém registros que pertencem a mesma classe y_t, então t é um nó de folha rotulado como y_t
  - Se D_t contiver registros que pertencem a mais de uma classe, use um teste de atributo para dividir os dados em subconjuntos menores. Aplique recursivamente o procedimento a cada subconjunto.

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



Algoritmo de Hunt

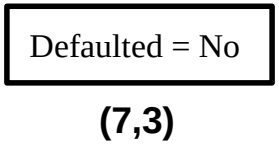
Defaulted = No

(7,3)

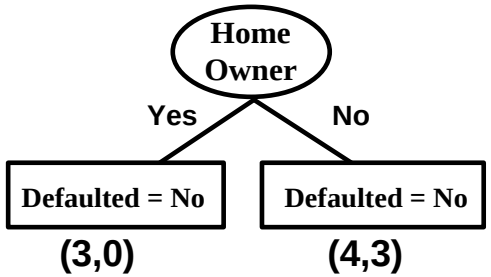
(a)

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Algoritmo de Hunt



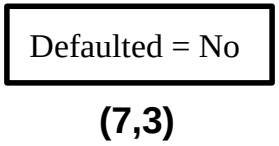
(a)



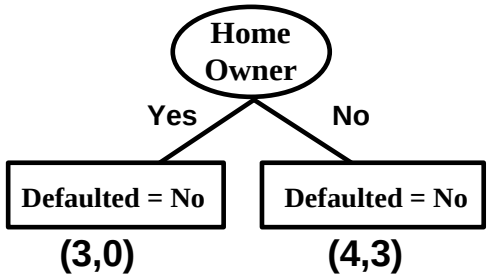
(b)

ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Algoritmo de Hunt

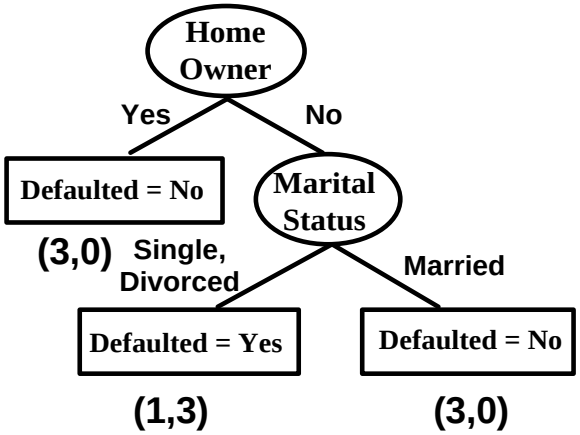


(a)



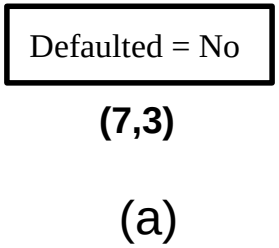
(b)

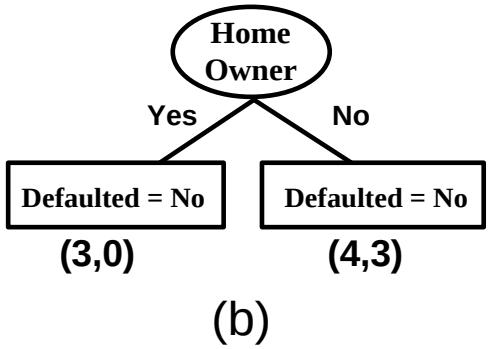
ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

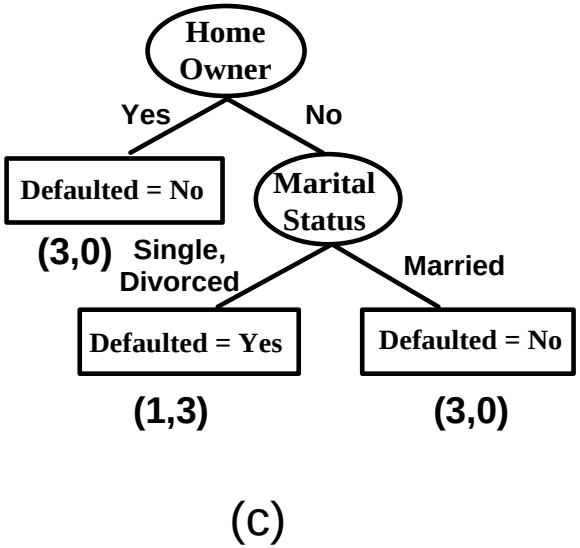


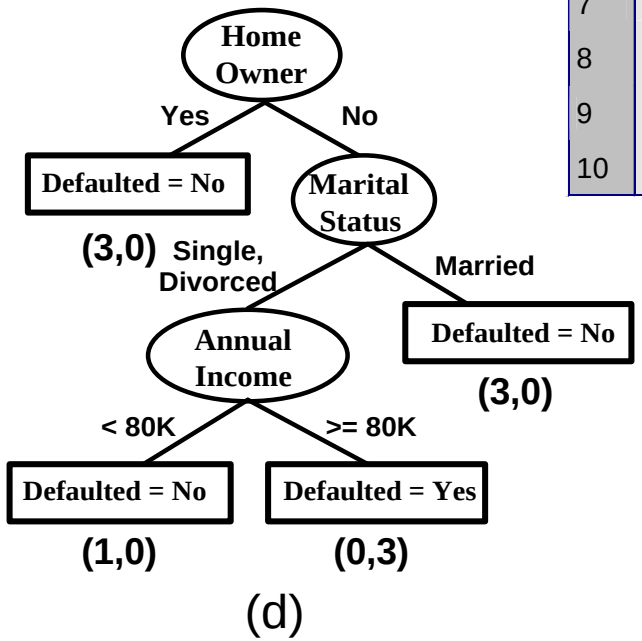
(c)

Algoritmo de Hunt









ID	Home Owner	Marital Status	Annual Income	Defaulted Borrower
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Problemas de Indução de Árvore de Decisão

- Como os registros de treinamento devem ser divididos?
  - Método para especificar a condição de teste
    - dependendo do tipo de atributo
  - Medida para avaliar se uma condição de teste é boa suficiente

- Como deve parar o procedimento de divisão?
  - Interromper a divisão se todos os registros pertencerem à mesma classe ou tiverem valores de atributo idênticos
  - Rescisão antecipada

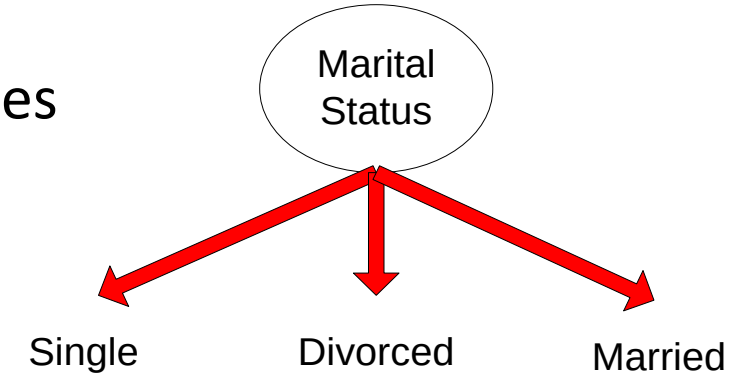
Método para expressar condições de teste

- Depende de tipo de atributo
  - Binário
  - Nominal
  - Ordinal
  - Contínuo

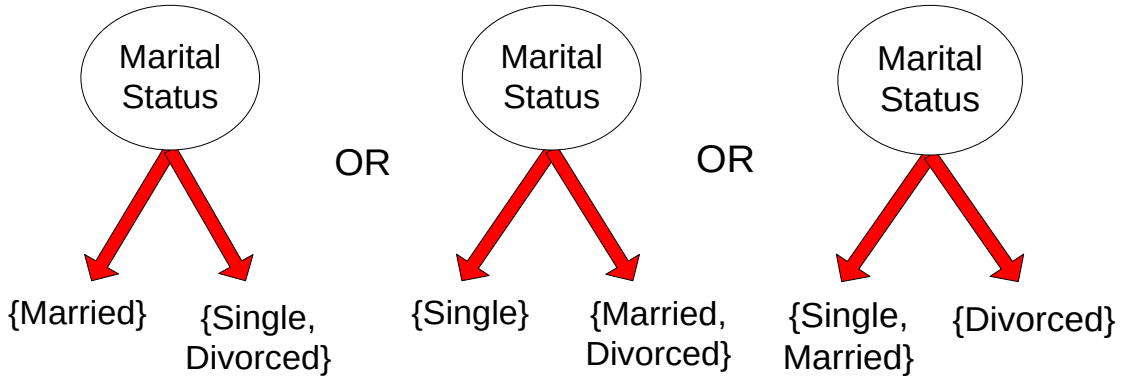
- Depende do número de maneiras de dividir
  - Divisão em 2
  - Multi-divisão

Condição de teste para atributos nominais

- Multi-divisão:
  - Use tantas partições como valores distintos.

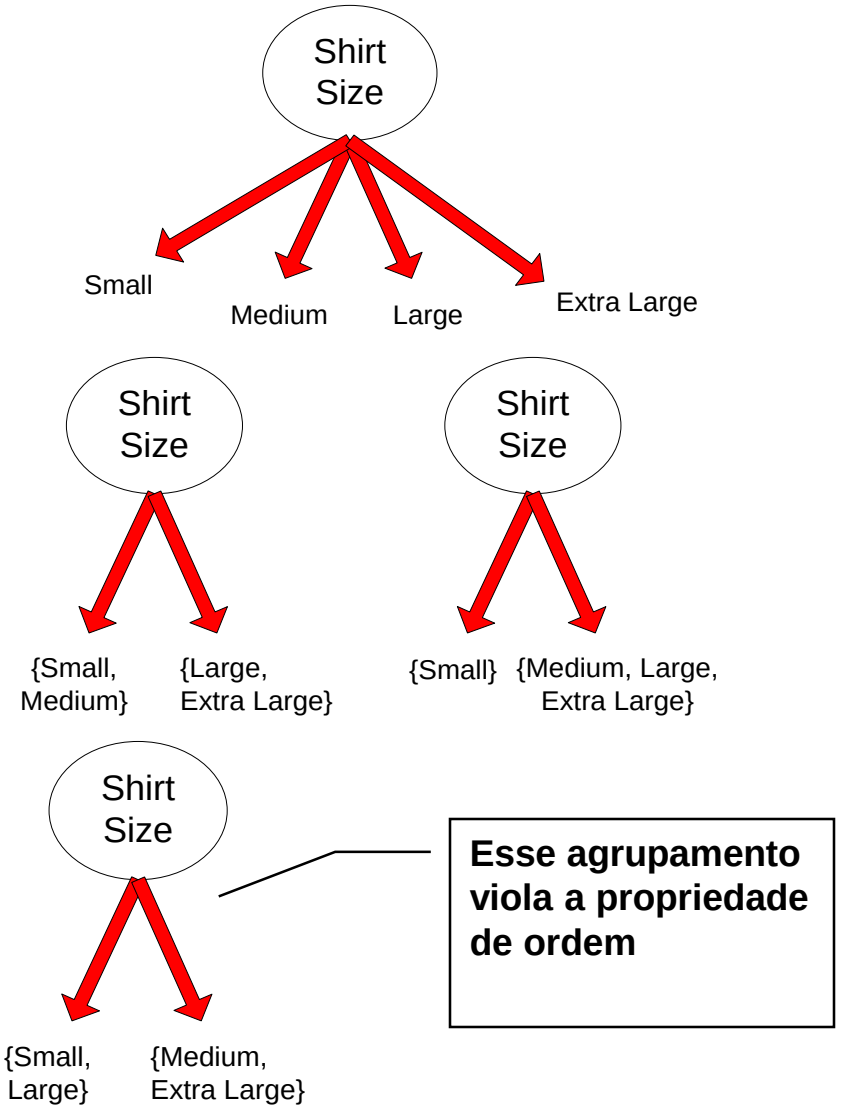


- Divisão binária:
  - Divide valores em dois subconjuntos

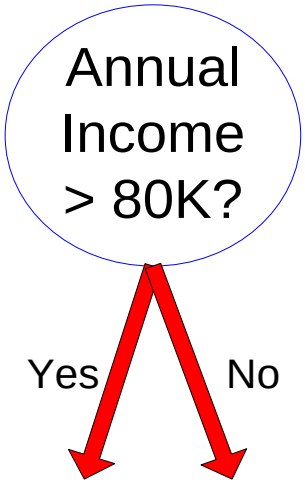


Condição de teste para atributos ordinais

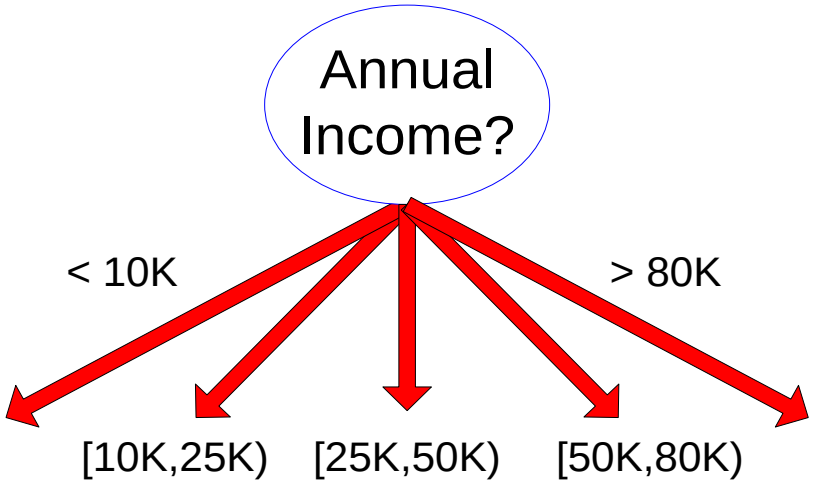
- Multi-divisão:
  - Use tantas partições como valores distintos
- Divisão binária:
  - Divide valores em dois subconjuntos
  - Preserva propriedade de ordem entre valores de atributo



Condição de teste para atributos contínuos



(i) Binary split



(ii) Multi-way split

Divisão de atributos contínuos

- Abordagens diferentes
  - Discretização para formar um atributo categórico ordinal

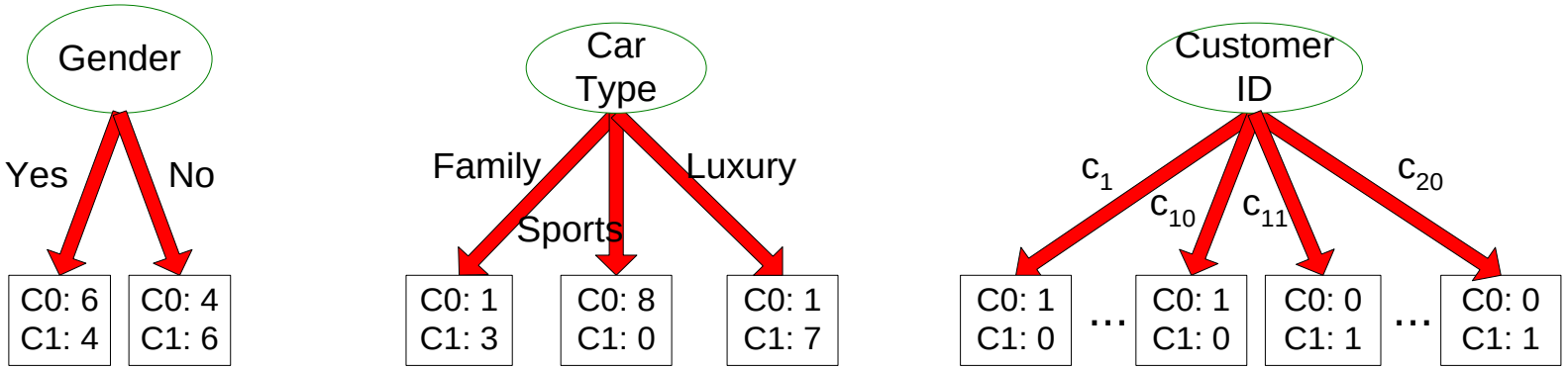
    As faixas podem ser encontradas pela divisão de intervalos iguais, frequências iguais (percentis), ou pelo agrupamento.
    - Estático – discretizar uma vez no início
    - Dinâmico – repita em cada nó

  - Decisão binária: $(A < v)$ ou $(A \geq v)$
    - considere todas as divisões possíveis e encontra o melhor corte
    - pode ser mais caro computacionalmente

Como determinar a melhor divisão?

Antes de dividir: 10 registros de classe 0, 10 registros de classe 1

Customer Id	Gender	Car Type	Shirt Size	Class
1	M	Family	Small	C0
2	M	Sports	Medium	C0
3	M	Sports	Medium	C0
4	M	Sports	Large	C0
5	M	Sports	Extra Large	C0
6	M	Sports	Extra Large	C0
7	F	Sports	Small	C0
8	F	Sports	Small	C0
9	F	Sports	Medium	C0
10	F	Luxury	Large	C0
11	M	Family	Large	C1
12	M	Family	Extra Large	C1
13	M	Family	Medium	C1
14	M	Luxury	Extra Large	C1
15	F	Luxury	Small	C1
16	F	Luxury	Small	C1
17	F	Luxury	Medium	C1
18	F	Luxury	Medium	C1
19	F	Luxury	Medium	C1
20	F	Luxury	Large	C1



Qual condição de teste é a melhor?

Como determinar a melhor divisão?

- Abordagem gulosa:
  - Os nós com distribuição de classe mais pura são preferidos

- Precisa uma medida da impureza do nó:

C0: 5 C1: 5	C0: 9 C1: 1
Alto grau de impureza	**Baixo grau de impureza**

Medidas de impureza de nó

- Gini Index

$$GINI(t) = 1 - \sum_j [p(j|t)]^2$$

- Entropia

$$E(t) = -\sum_j p(j|t) \log p\,(j|t)$$

- Erro de classificação incorreta

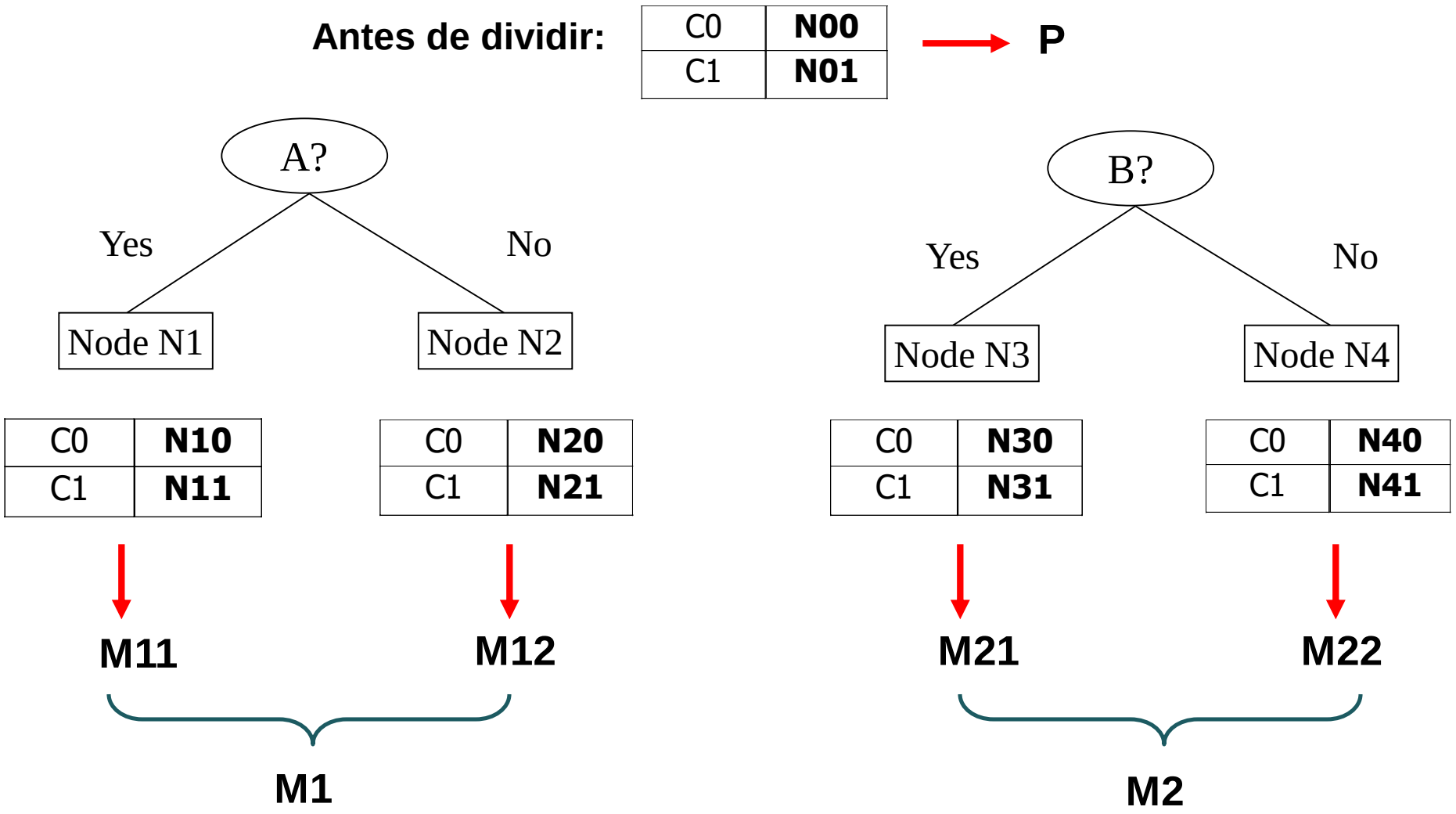
$$Erro(t) = 1 - \max_i [\boldsymbol{p}(i|t)]$$

- $p(i|t)$ – fração de registros com classe i para o nó t

Procurando a melhor divisão

1. Calcular a medida de impureza (P) antes de dividir
2. Calcular a medida de impureza (M) depois de dividir
   - Calcular a medida de impureza de cada nó filho
   - M é a impureza ponderada de filhos
3. Escolha a condição de teste de atributo que produz o maior ganho ou equivalentemente, menor medida de impureza após a separação (M)
4. **Ganho = P - M**

Procurando a melhor divisão



Medida de Impureza: GINI

- Índice de Gini para um determinado nó t:

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

(ATT: $p(j|t)$ é a frequência relativa da classe j no nó t).

  - Máximo (1 - 1/n_c) quando os registros são distribuídos igualmente entre todas as classes, implicando informações menos interessantes
  - Mínimo (0.0) quando todos os registros pertencem a uma classe, implicando a informação mais interessante

Medida da impureza: GINI

- Índice de Gini para um determinado nó t:

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

(ATT: $p(j|t)$ é a frequência relativa da classe j no nó t).

  - Para um problema de 2 classes (p, 1 – p):
    - GINI = $1 - p^2 - (1 - p)^2 = 2p\,(1\text{-}p)$

C1	**0**
C2	**6**
Gini=0.000	

C1	**1**
C2	**5**
Gini=0.278	

C1	**2**
C2	**4**
Gini=0.444	

C1	**3**
C2	**3**
Gini=0.500	

Calculando o índice Gini de um único nó

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Gini = 1 – P(C1)² – P(C2)² = 1 – 0 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Gini = 1 – (1/6)² – (5/6)² = 0.278

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Gini = 1 – (2/6)² – (4/6)² = 0.444

Calculando o índice Gini para uma coleção de nós

- Quando o nó p é dividido em k partições (filhos)

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

onde, n_i = o número de registros para filho i,

n = o número de registros do nó parente p.

- Escolha o atributo que minimiza o índice de Gini médio ponderado das crianças

- Gini index é usado nos algoritmos de árvores de decisão CART, SLIQ, SPRINT

Medida da impureza: Entropia

- Entropia para um nó t:

$$Entropy(t) = -\sum_j p(j \mid t) \log p(j \mid t)$$

(ATT: $p(j|t)$ é a frequência relativa da classe j no nó t).

  - Máximo (log n_c) quando os registros são distribuídos igualmente entre todas as classes, implicando menor informação
  - Mínimo (0.0) quando todos os registros pertencem a uma classe, implicando maior informação

– As computações baseadas em entropia são bastante semelhantes às computações do índice GINI

Calculando a entropia de um único nó

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

Ganho de informação computacional após a divisão

- Ganho de informação:

$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

Nó pai, p é dividido em k partições;

n_i é o número de registros em partição i

  - Escolha a divisão que atinge a maior parte da redução (maximiza o ganho)
  - Usado em algoritmos ID3 e C4.5 de árvores de decisão

Medida de impureza: Erro de classificação

- Erro de classificação para nó t :

$$Error(t) = 1 - \max_i P(i \mid t)$$

  - Máximo (1 - 1/n_c) quando os registros são distribuídos igualmente entre todas as classes, implicando informações menos interessantes
  - Mínimo (0.0) quando todos os registros pertencem a uma classe, implicando a informação mais interessante

Calculando Erro de um único nó

$$Error(t) = 1 - \max_i P(i \mid t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0 P(C2) = 6/6 = 1

Error = 1 – max (0, 1) = 1 – 1 = 0

C1	**1**
C2	**5**

P(C1) = 1/6 P(C2) = 5/6

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

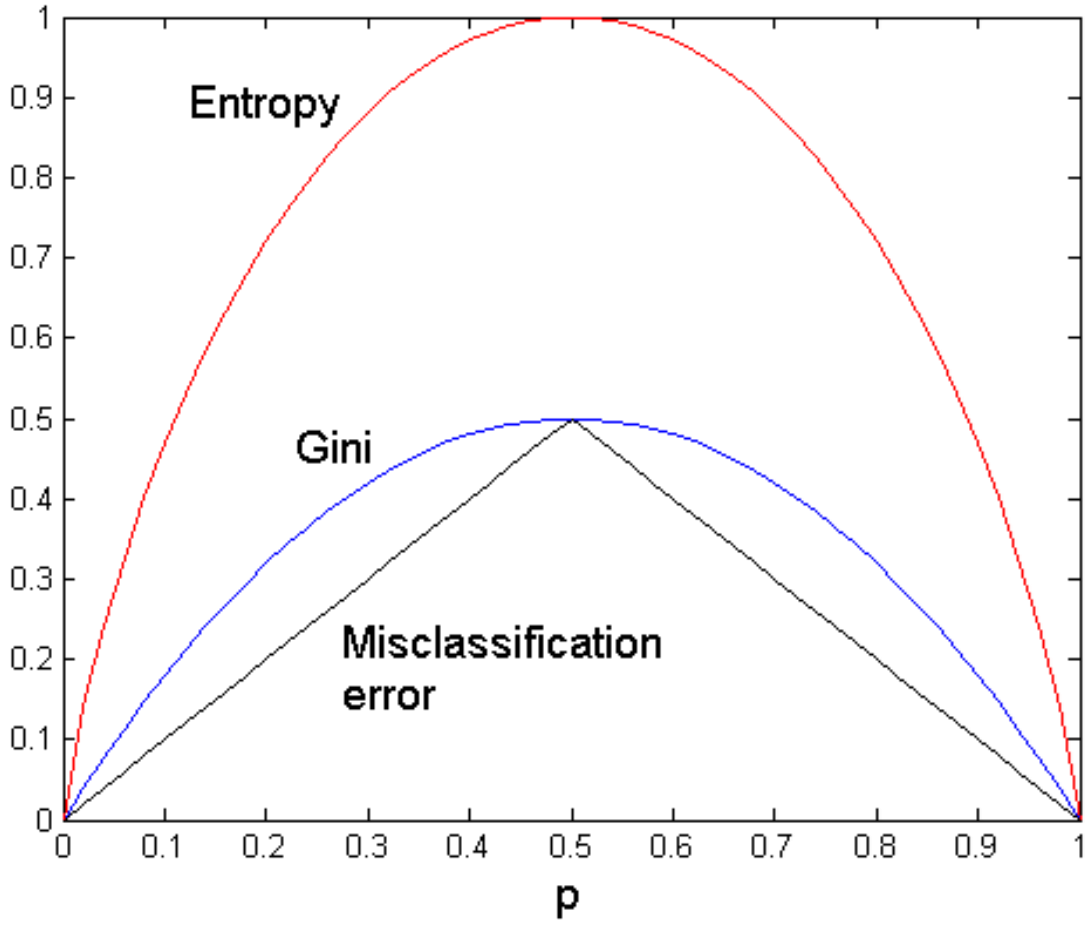
C1	**2**
C2	**4**

P(C1) = 2/6 P(C2) = 4/6

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

Comparação de Medidas de Impureza

Para um problema de 2 classes:



Classificação baseada em árvore de decisão

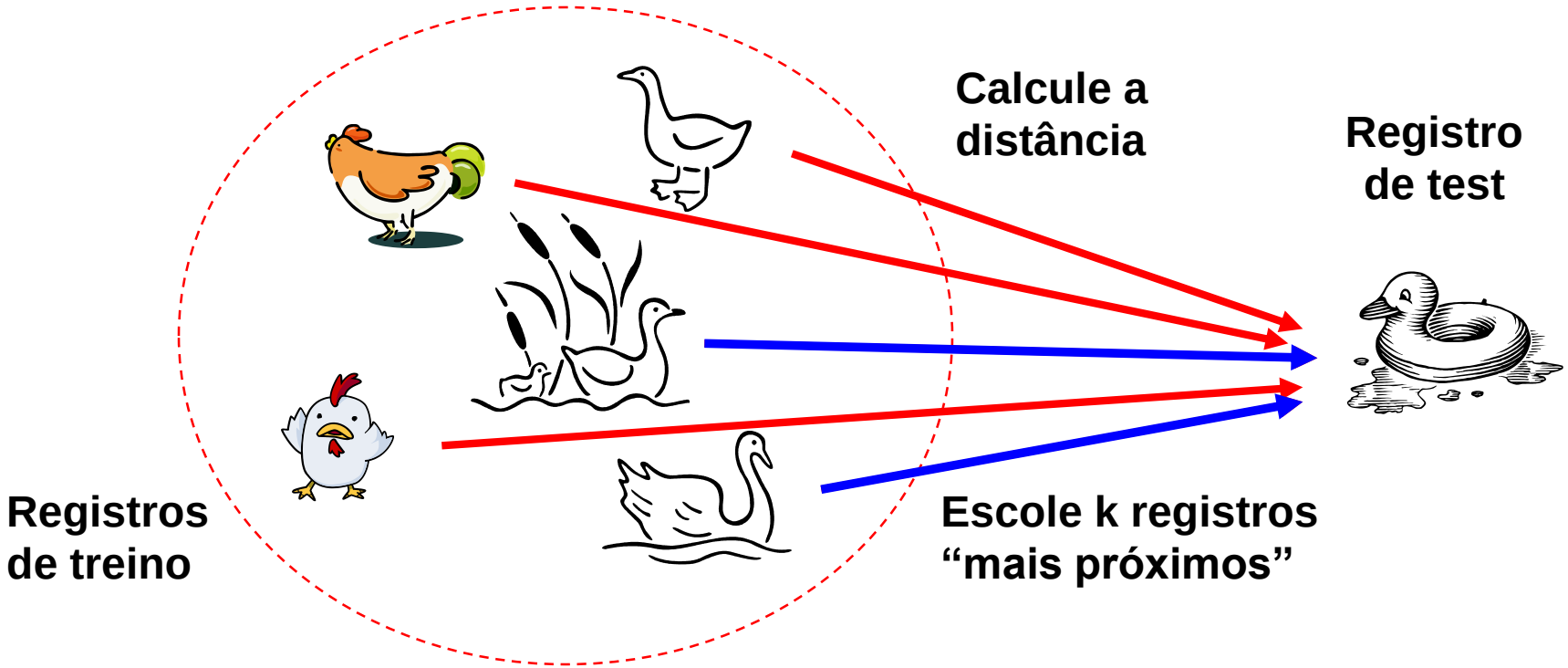
- Vantagens:
  - Barata para construir
  - Extremamente rápida na classificação de registros desconhecidos
  - Fácil de interpretar para árvores pequenas
  - Robusto ao ruído (especial quando os métodos para evitar o overfitting são empregados)
  - Pode lidar facilmente com atributos redundantes ou irrelevantes (a menos que os atributos estejam interagindo)
- Desvantagens:
  - O espaço de árvores de decisão possíveis é exponencialmente grande. Abordagens gulosas são muitas vezes incapazes de encontrar a melhor árvore.
  - Cada limite de decisão envolve apenas um único atributo

Classificadores baseados em instâncias

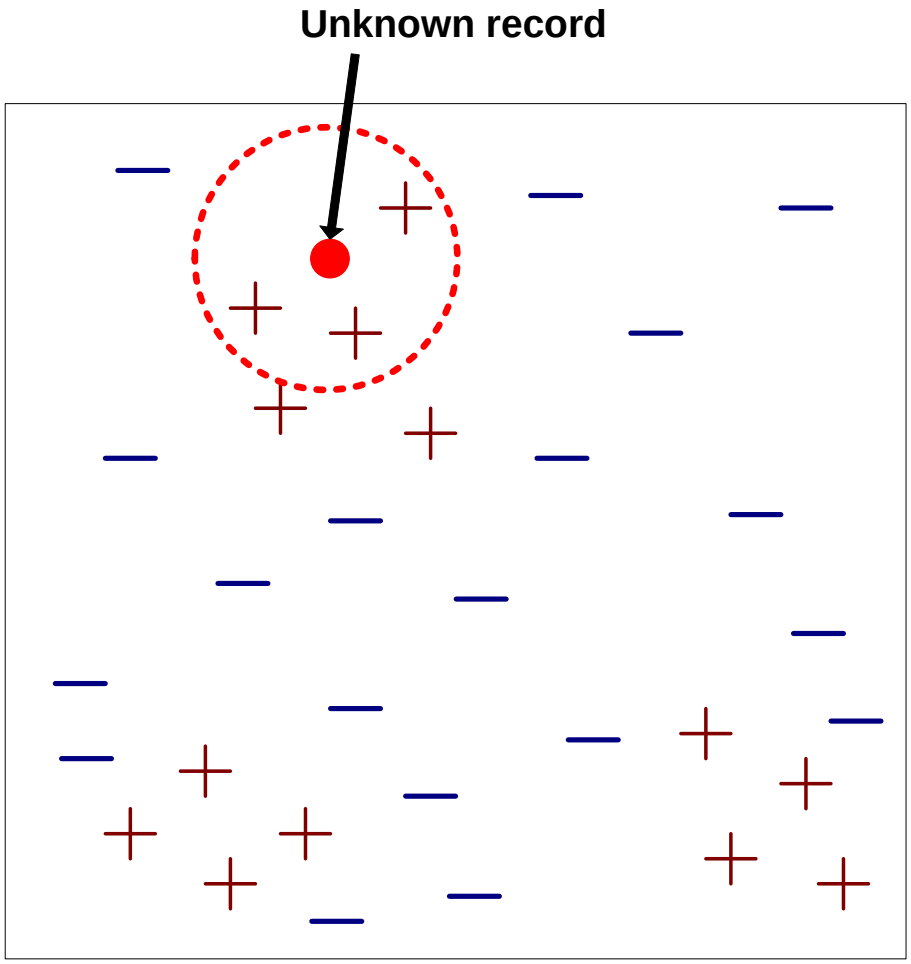
- Exemplos:
  - Rote-learner
    - Memoriza dados de treinamento inteiros e executa a classificação somente se os atributos do registro correspondem a exatamente um dos exemplos de treinamento
  - Nearest neighbor
    - Usa pontos “k-mais próximos” (vizinhos mais próximos) para a realização de classificação

Classificador Nearest Neighbor

- Ideia básica:
  - Se ele anda como um pato, emite som como um pato, então provavelmente é um pato



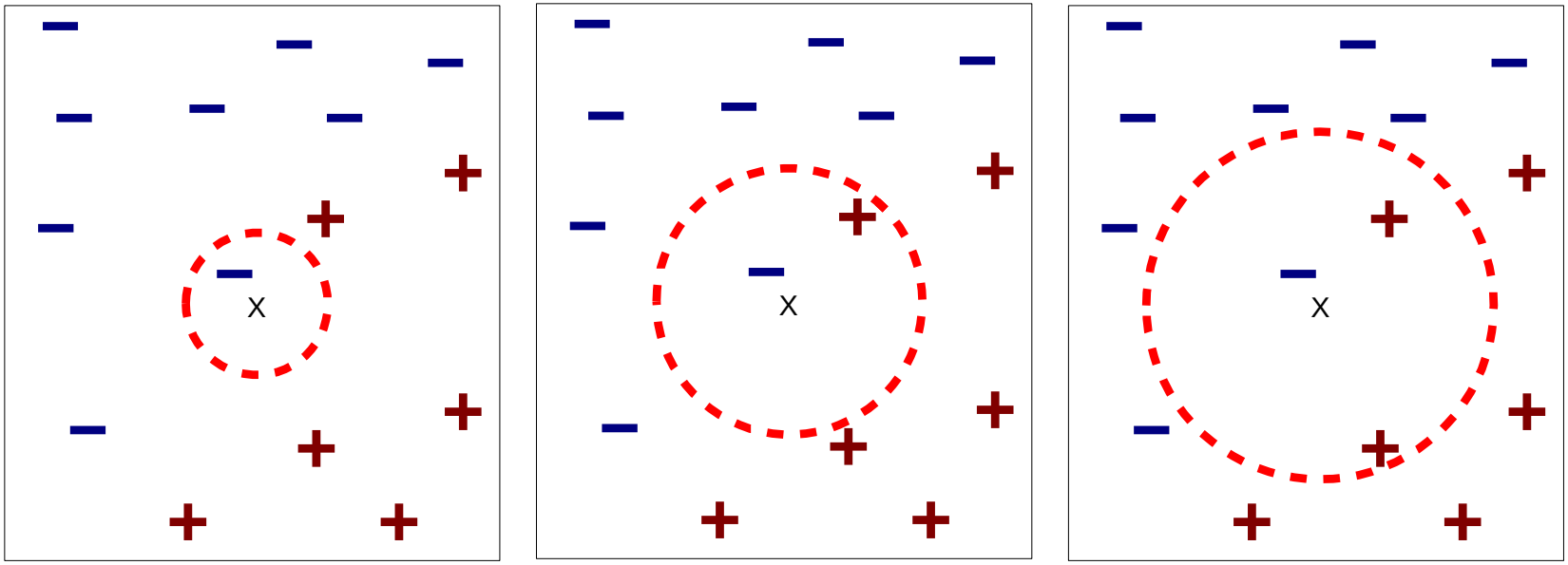
Classificador Nearest-Neighbor



- Requer três coisas
  - O conjunto de registros rotulados
  - Métrica para calcular a distância entre registros
  - O valor de *k*, o número de vizinhos mais próximos para recuperar

- Para classificar um registro desconhecido:
  - Calcule a distância para outros registros de treino
  - Identifique *k* vizinhos mais próximos
  - Usar rótulos de classe de vizinhos mais próximos para determinar o rótulo de classe de registro desconhecido (por exemplo, tomando votação por maioria)

Definição de Nearest Neighbor

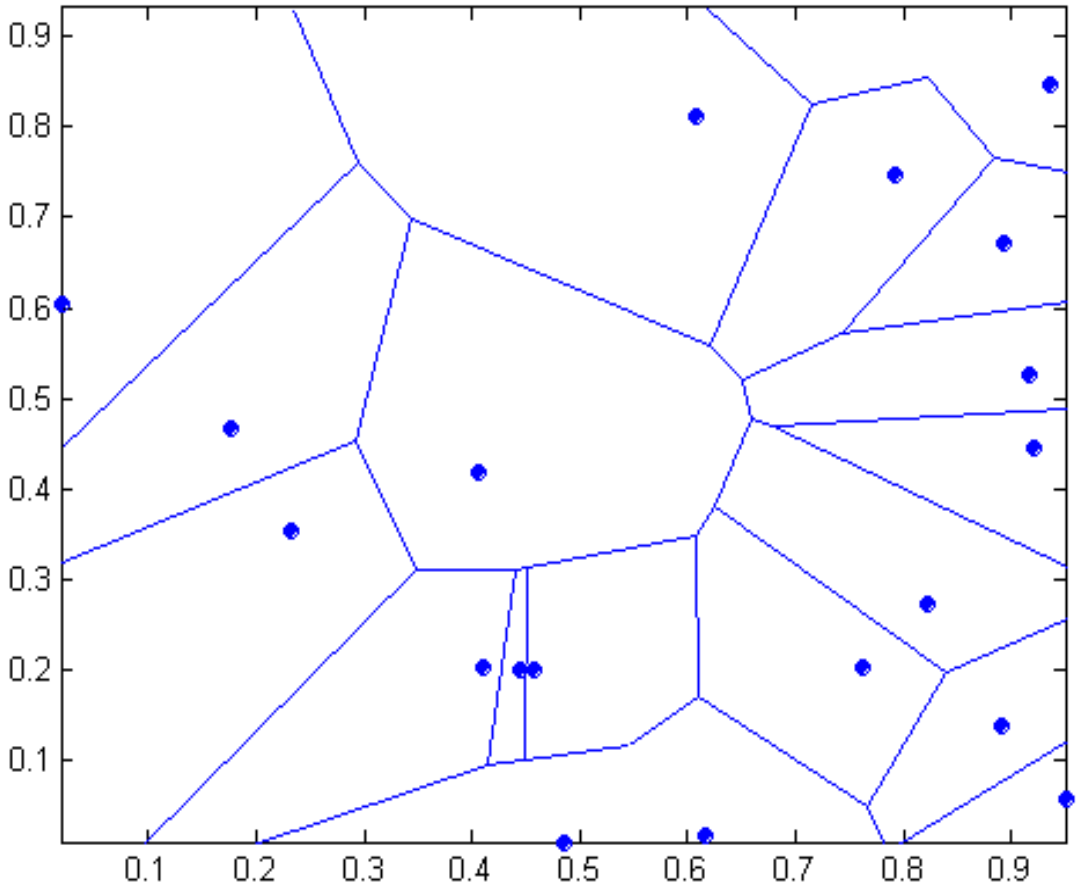


(a) 1-nearest neighbor (b) 2-nearest neighbor (c) 3-nearest neighbor

K-vizinhos mais próximos de um registro x são pontos de dados que têm as menores distâncias de k para x

1 vizinho mais próximo

Diagrama de Voronoi



Classificação Nearest Neighbor

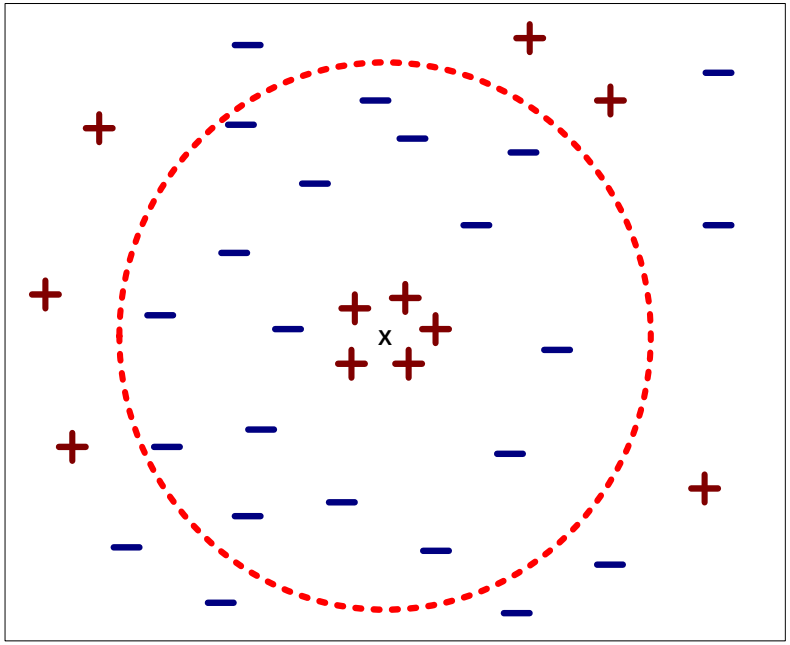
- Calcule a distância entre dois pontos:
  - Distância euclidiana

$$d(p,q) = \sqrt{\sum_i (p_i - q_i)^2}$$

- Determine a classe usando a lista de vizinhos mais próximos
  - Faça o voto da maioria dos rótulos de classe entre os vizinhos k-mais próximos
  - Pesa o voto de acordo com a distância
    - fator de peso, $w = 1/d^2$

Classificação Nearest Neighbor…

- Escolhendo o valor de k:
  - Se k for muito pequeno, fica sensível a pontos de ruído
  - Se k for muito grande, o bairro pode incluir pontos de outras classes



Classificação Nearest Neighbor…

- Problemas de normalização
  - Atributos talvez precisam ser normalizados para evitar que um atributo domina a medida de distância
  - Exemplos:
    - a altura de uma pessoa pode variar de 1.5 m a 1.8 m
    - peso de uma pessoa pode variar de 50kg a 150kg
    - renda de uma pessoa pode variar de $10K para $1M

Classificação Nearest Neighbor…

- A seleção da certa medida de similaridade é crítica:

1 1 1 1 1 1 1 1 1 1 1 0		0 0 0 0 0 0 0 0 0 0 0 1
	vs	
0 1 1 1 1 1 1 1 1 1 1 1		1 0 0 0 0 0 0 0 0 0 0 0

Distância euclidiana = 1,4142 para ambos pares

Classificação Nearest Neighbor…

- Classificadores k-NN são aprendizes preguiçosos, pois não constroem modelos explicitamente
- Classificação de registros desconhecidos é relativamente cara
- Pode produzir fronteiras arbitrarias de classes
- Decisões são baseadas nas informações locais
- A seleção da medida de proximidade correta é essencial
- Atributos supérfluos ou redundantes podem criar problemas
- Atributos ausentes são difíceis para classificar

Melhorando a eficiência de k-NN

- Evite o calculo de distâncias para todos os objetos no conjunto de treino
  - Métodos de acesso multi-dimensional (árvores k-d)
  - Fast approximate similarity search
  - Locality Sensitive Hashing (LSH)
- Condensação
  - Determine um conjunto menor de objetos que da o mesmo desempenho
- Edição
  - Remove objetos para melhorar a eficiência