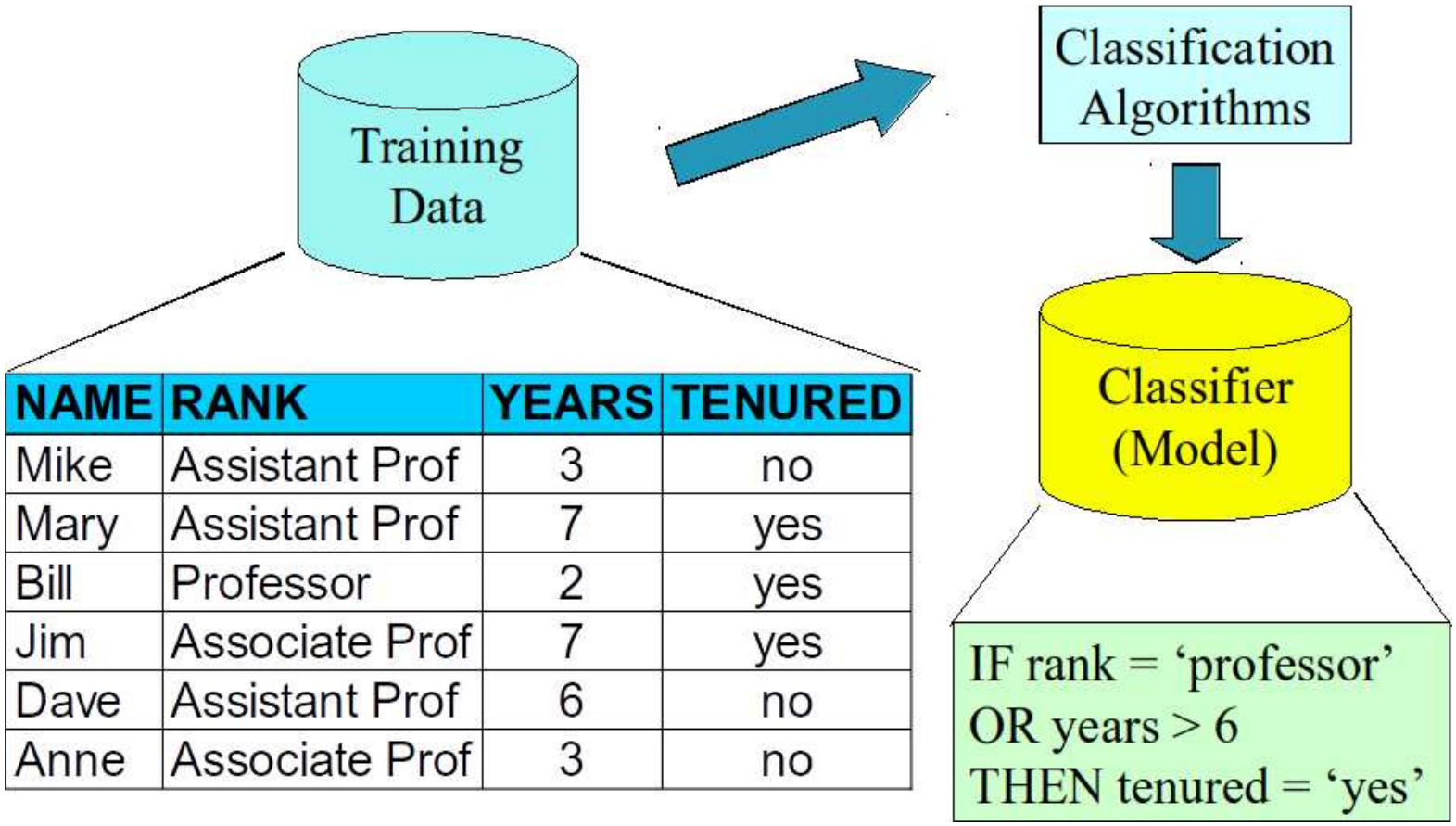
Mineração de Dados

Classificação

Sumário

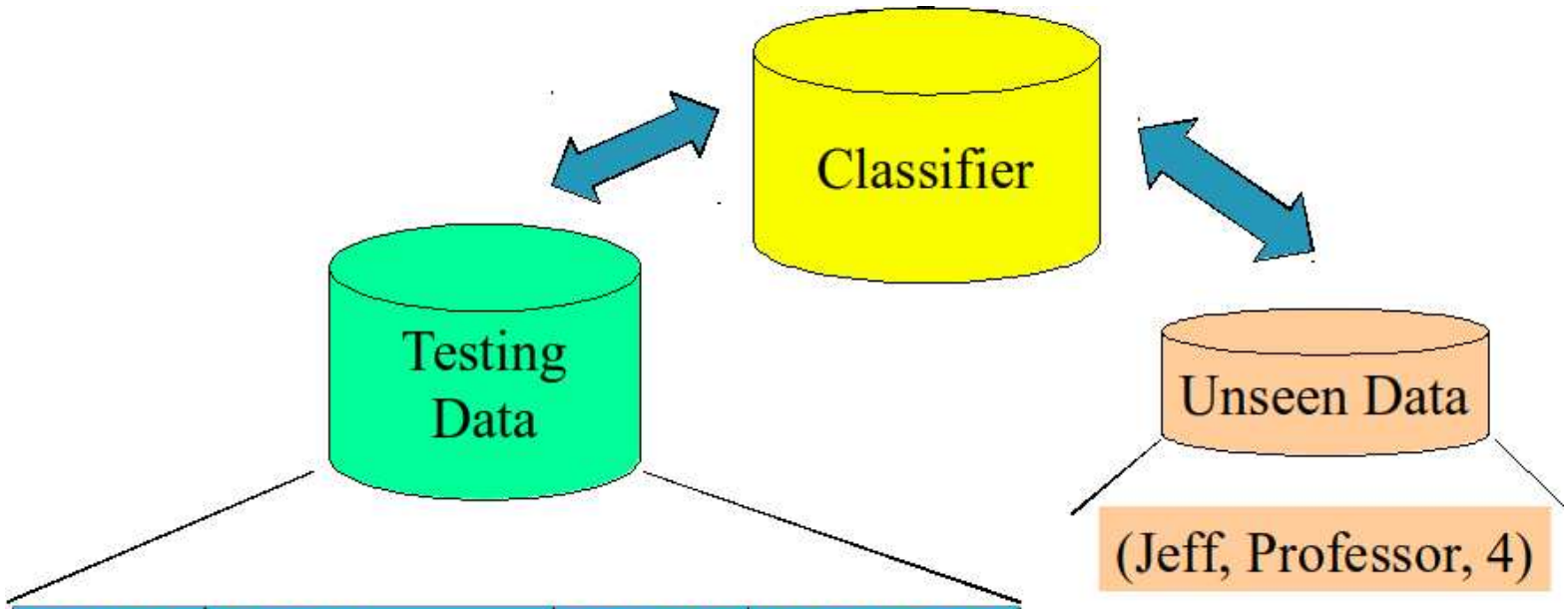
Processo de Classificação

Construção do Modelo



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

Utilização do Modelo



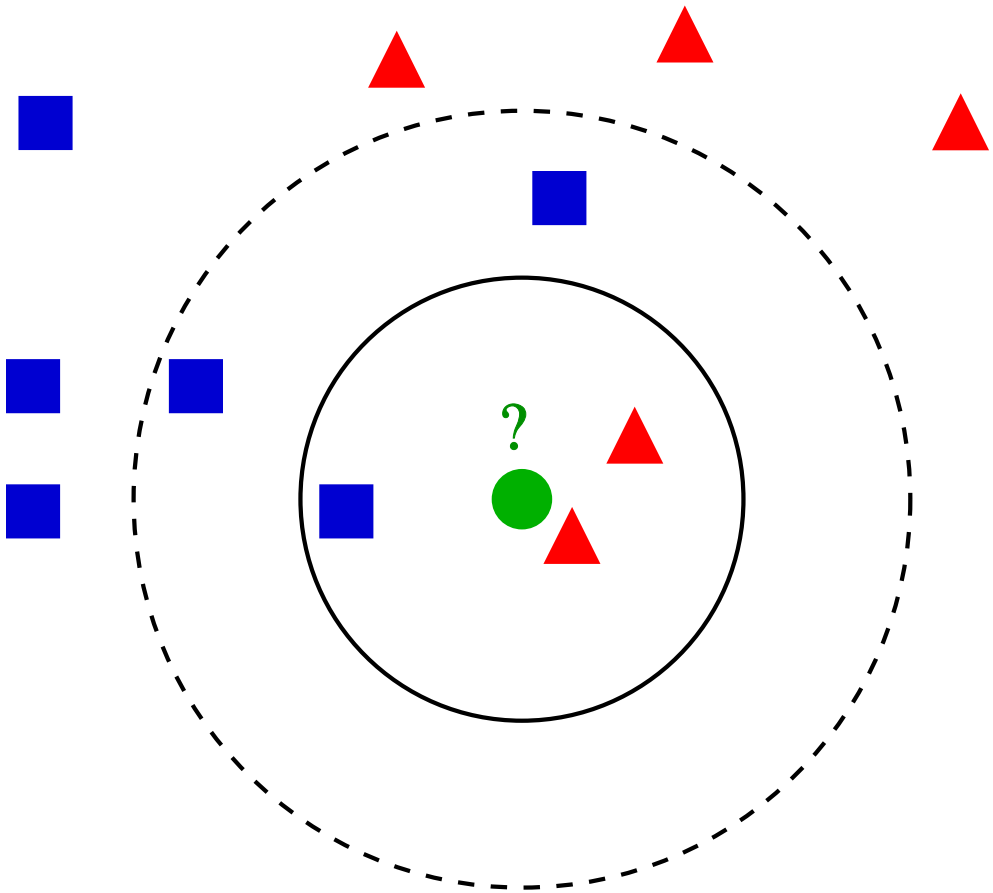
NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Vizinhos mais Próximos

kNN: Classificação

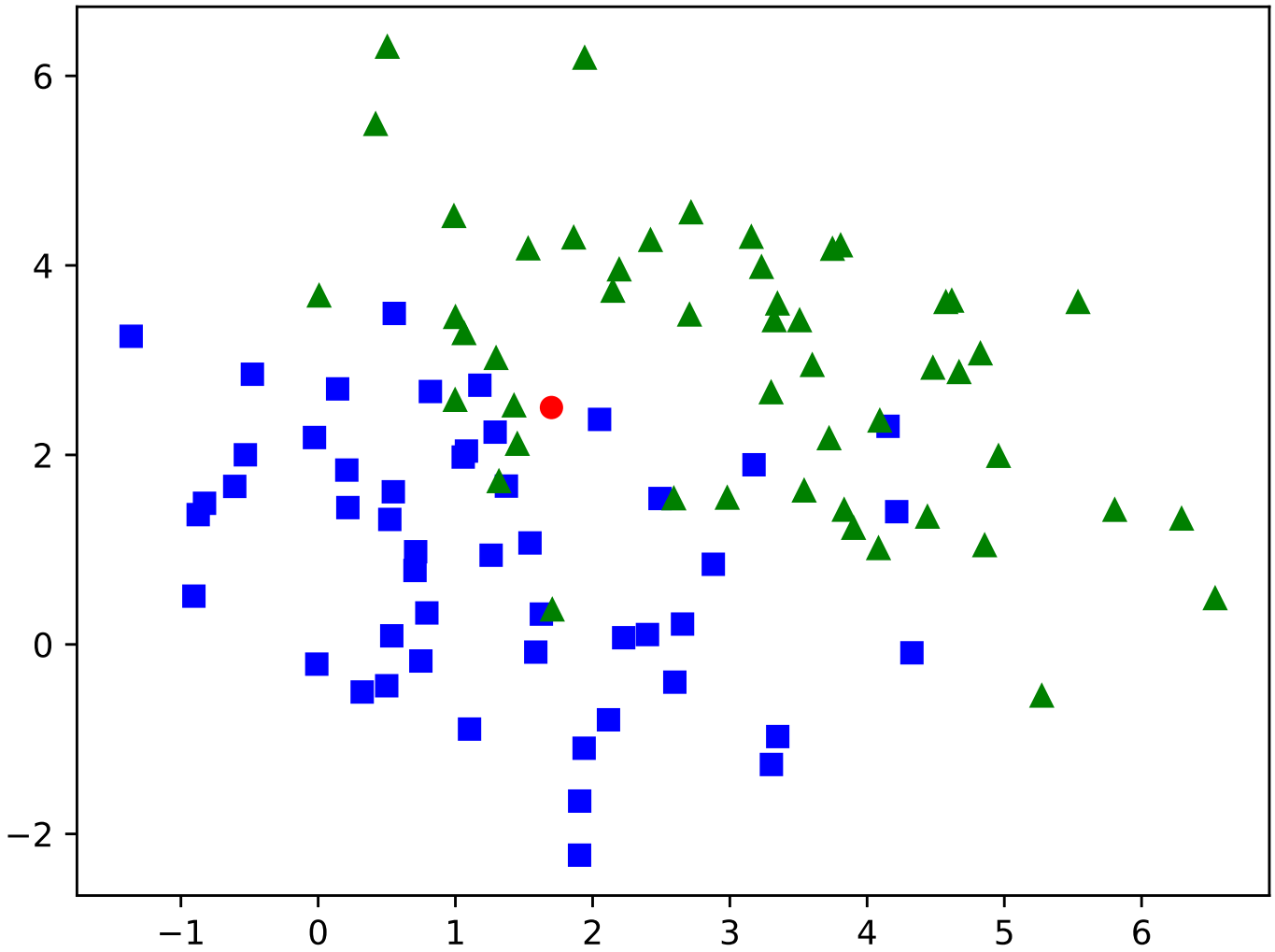
- Similar ao kNN para regressão
- Neste caso, a classe é inferida por meio de uma votação entre os vizinhos mais próximos
- Pode-se usar ponderação
  - Inverso da distância, por exemplo

kNN: Classificação

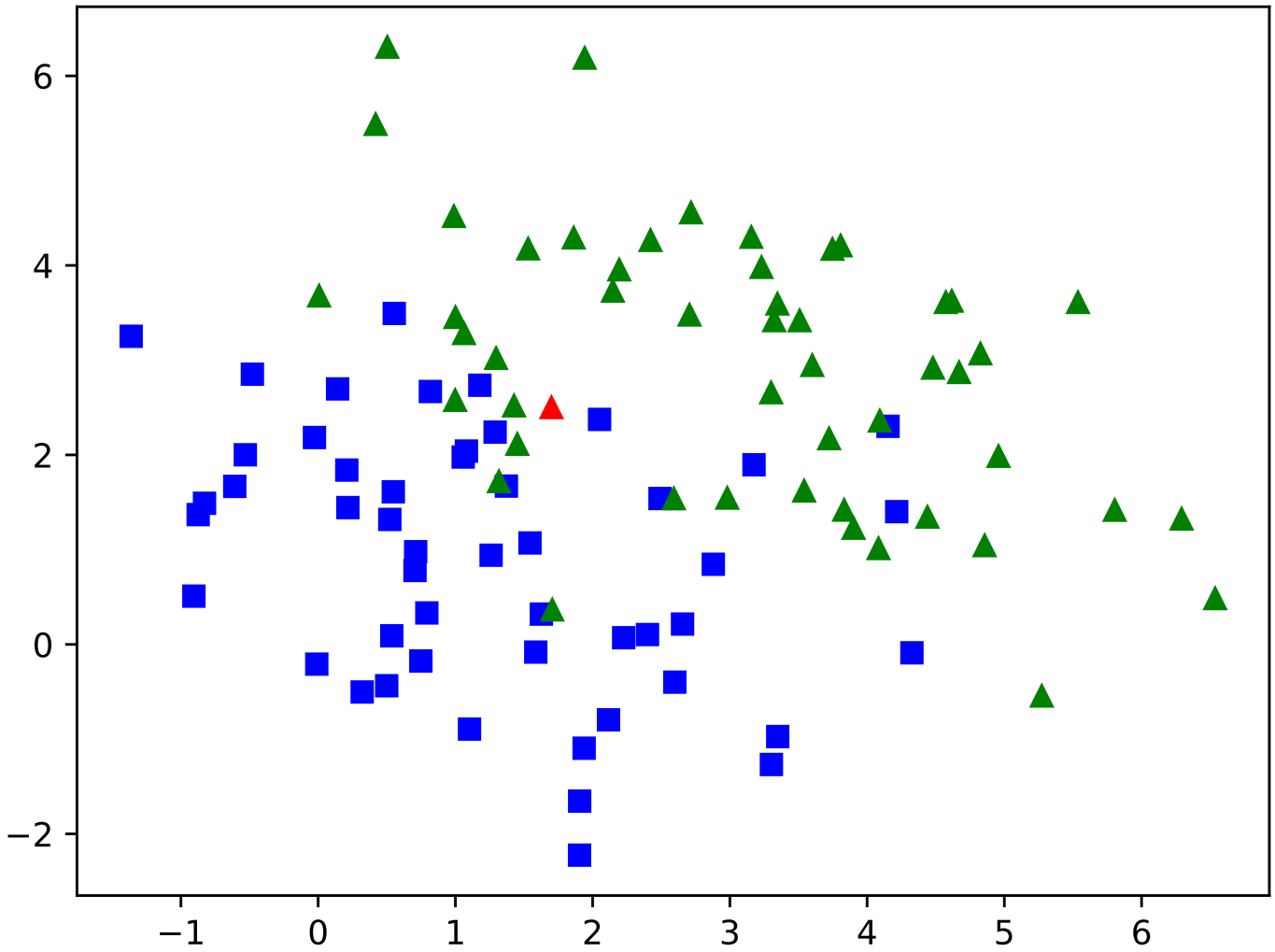


https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

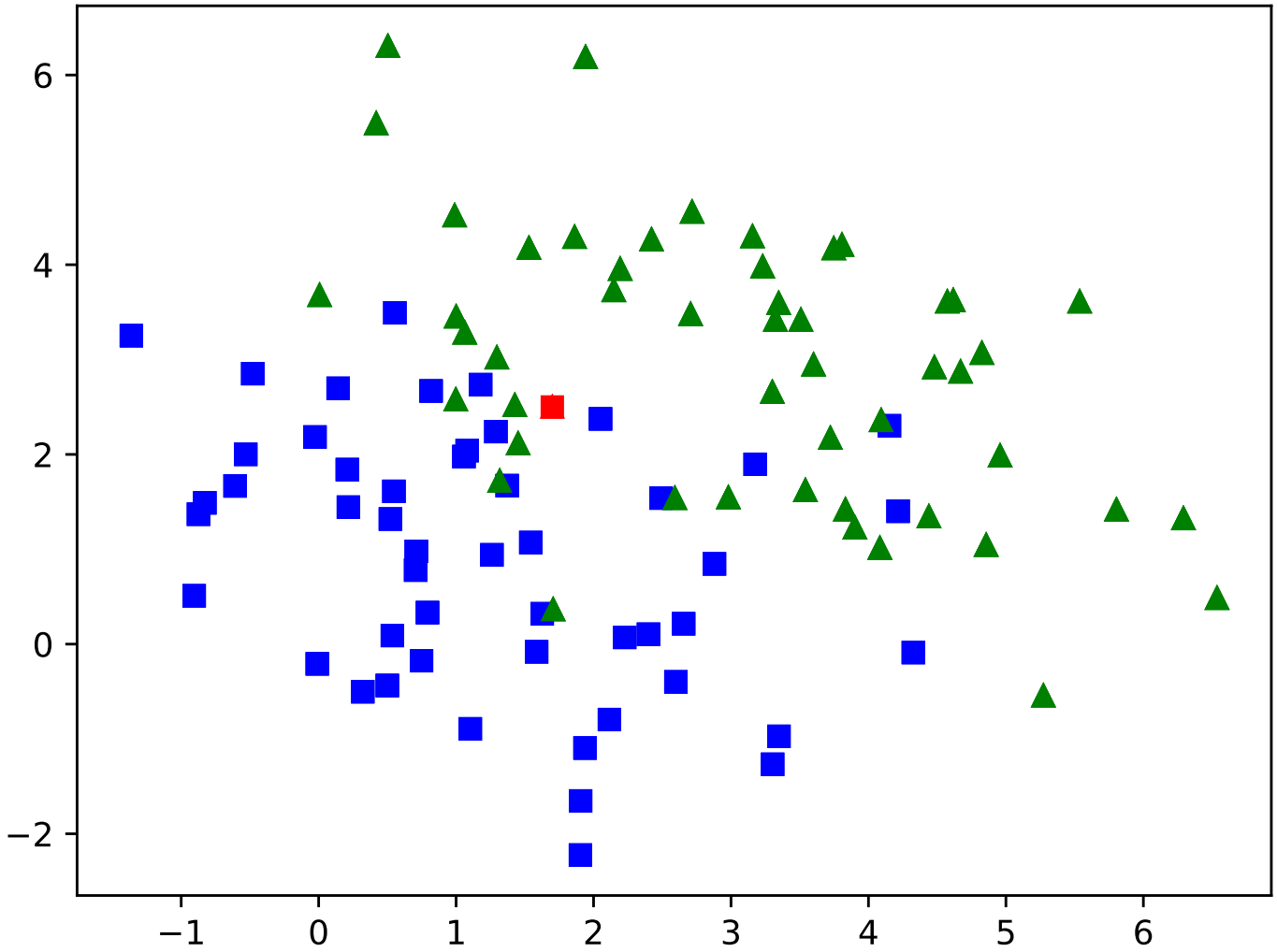
kNN: Classificação



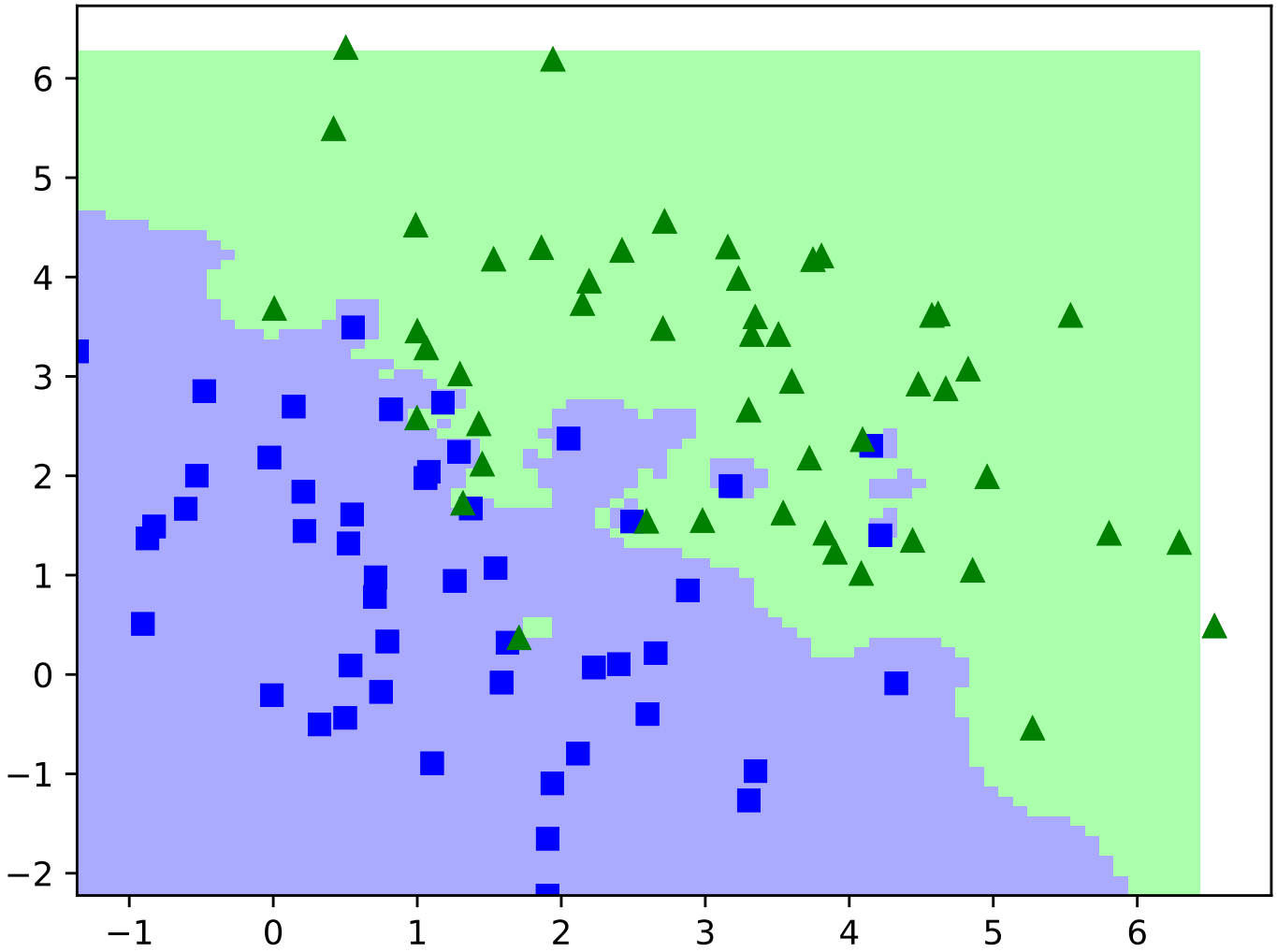
kNN: Classificação



kNN: Classificação com $k = 5$



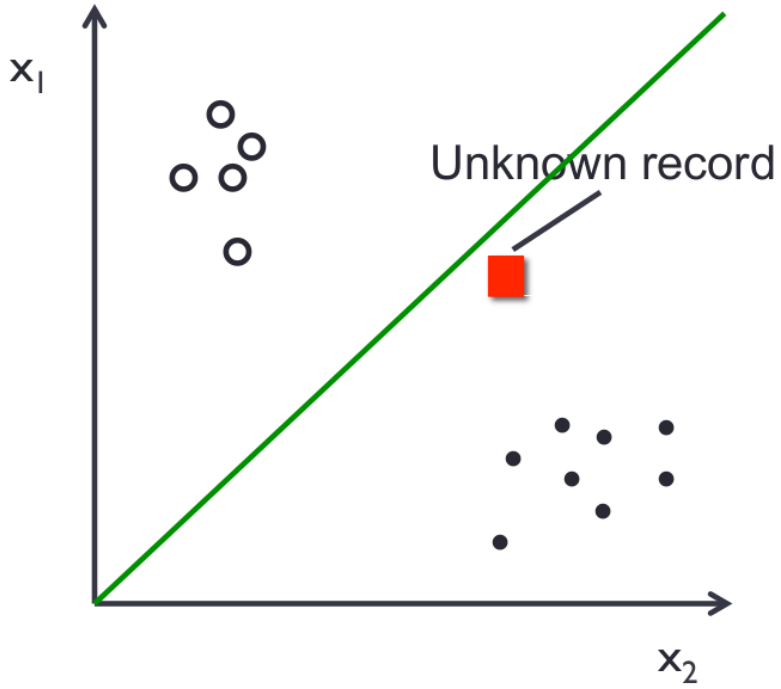
kNN: Classificação



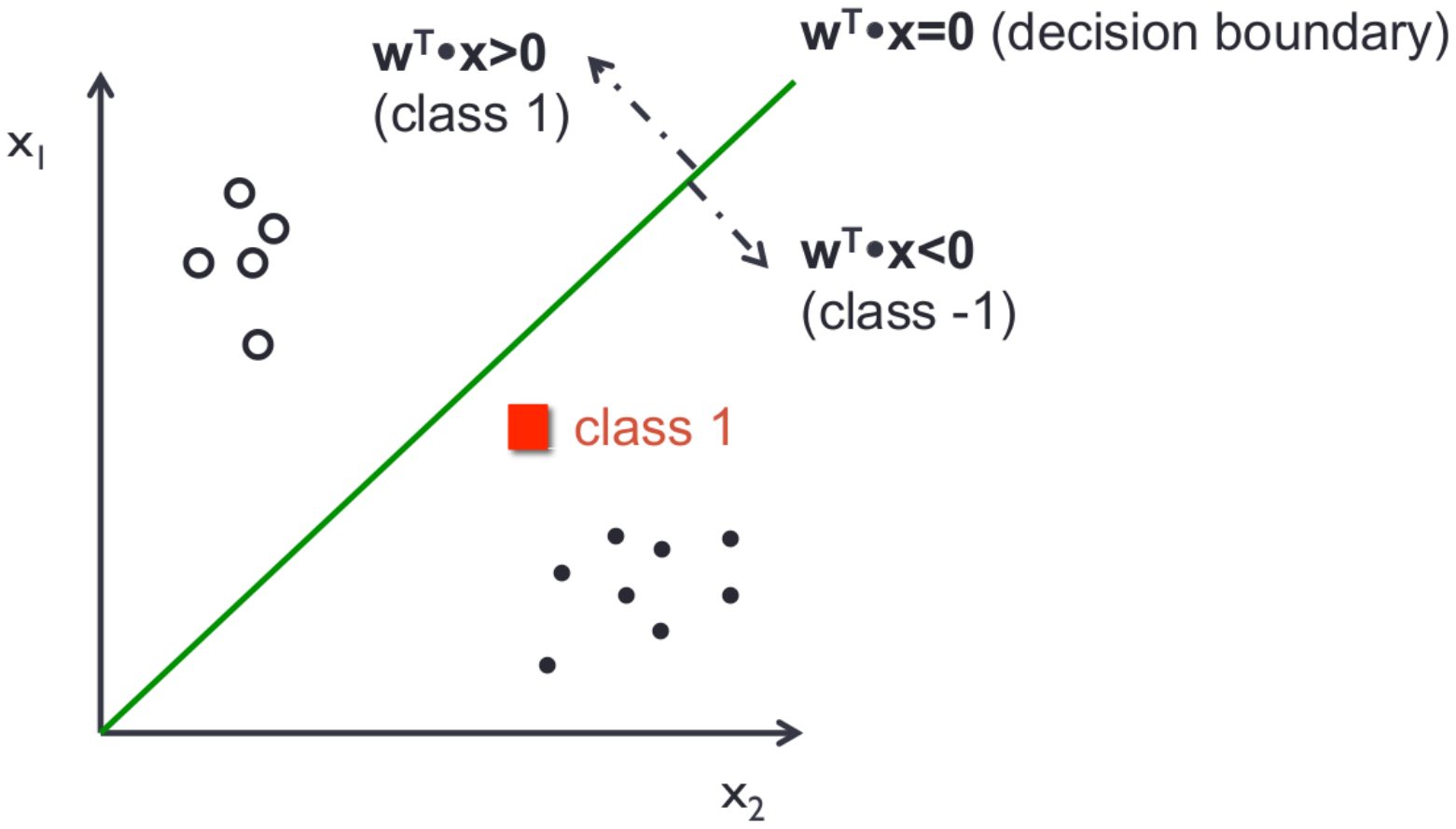
Método dos Mínimos Quadrados

Classificação via Mínimos Quadrados

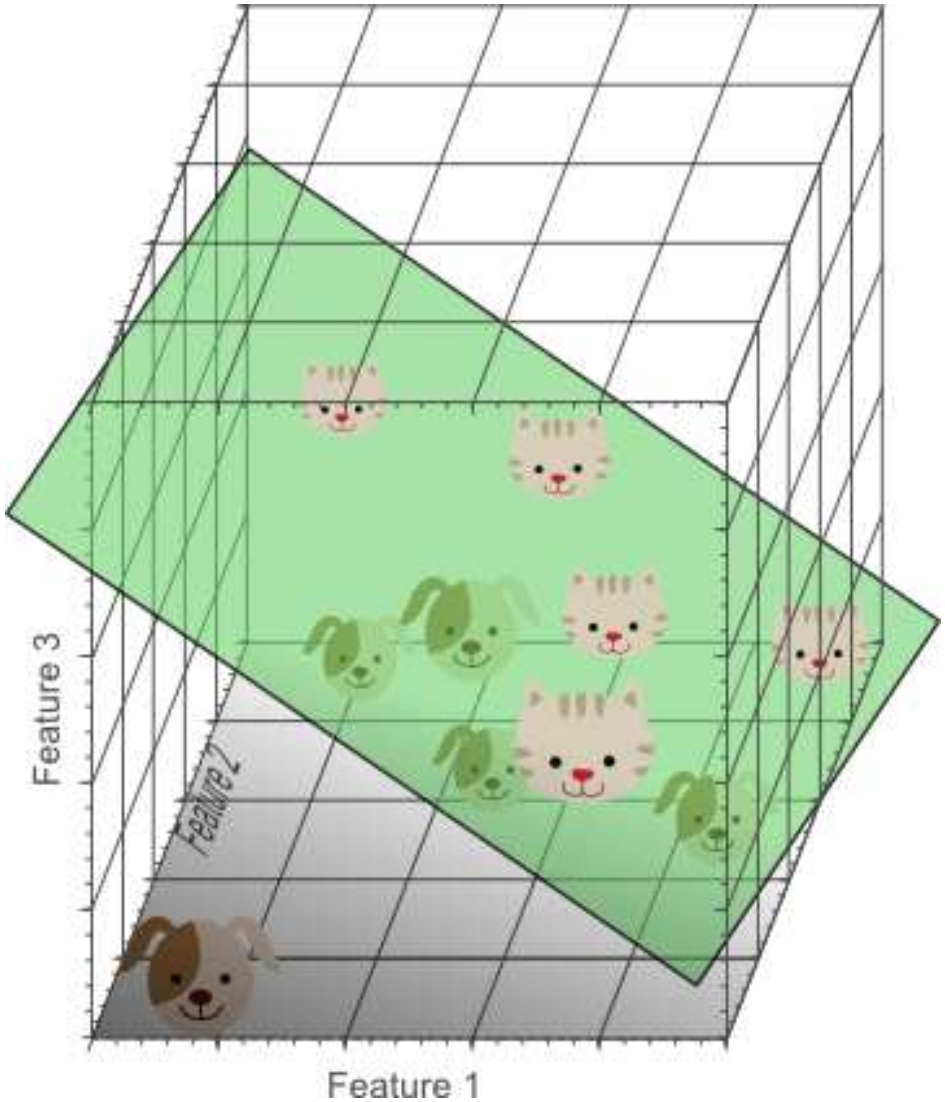
- O mesmo método pode ser usado para classificação
- O modelo gera uma superfície de separação



Classificação via Mínimos Quadrados

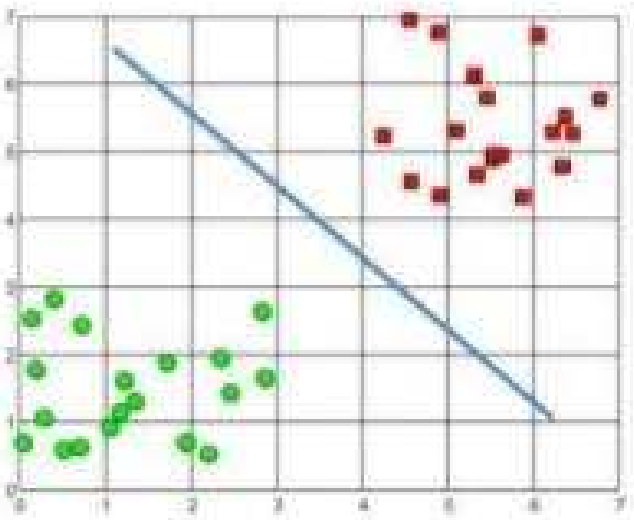


Classificação via Mínimos Quadrados

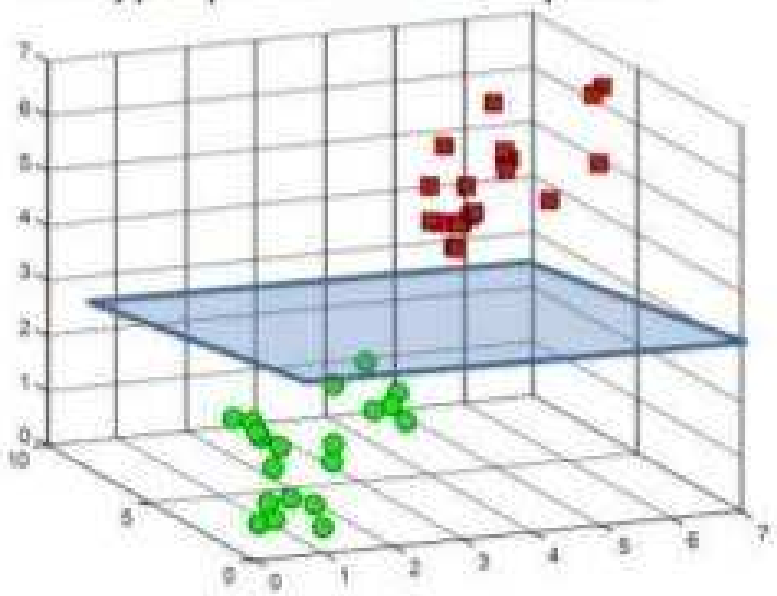


Classificação via Mínimos Quadrados

A hyperplane in $\mathbb{R}^2$ is a line



A hyperplane in $\mathbb{R}^3$ is a plane



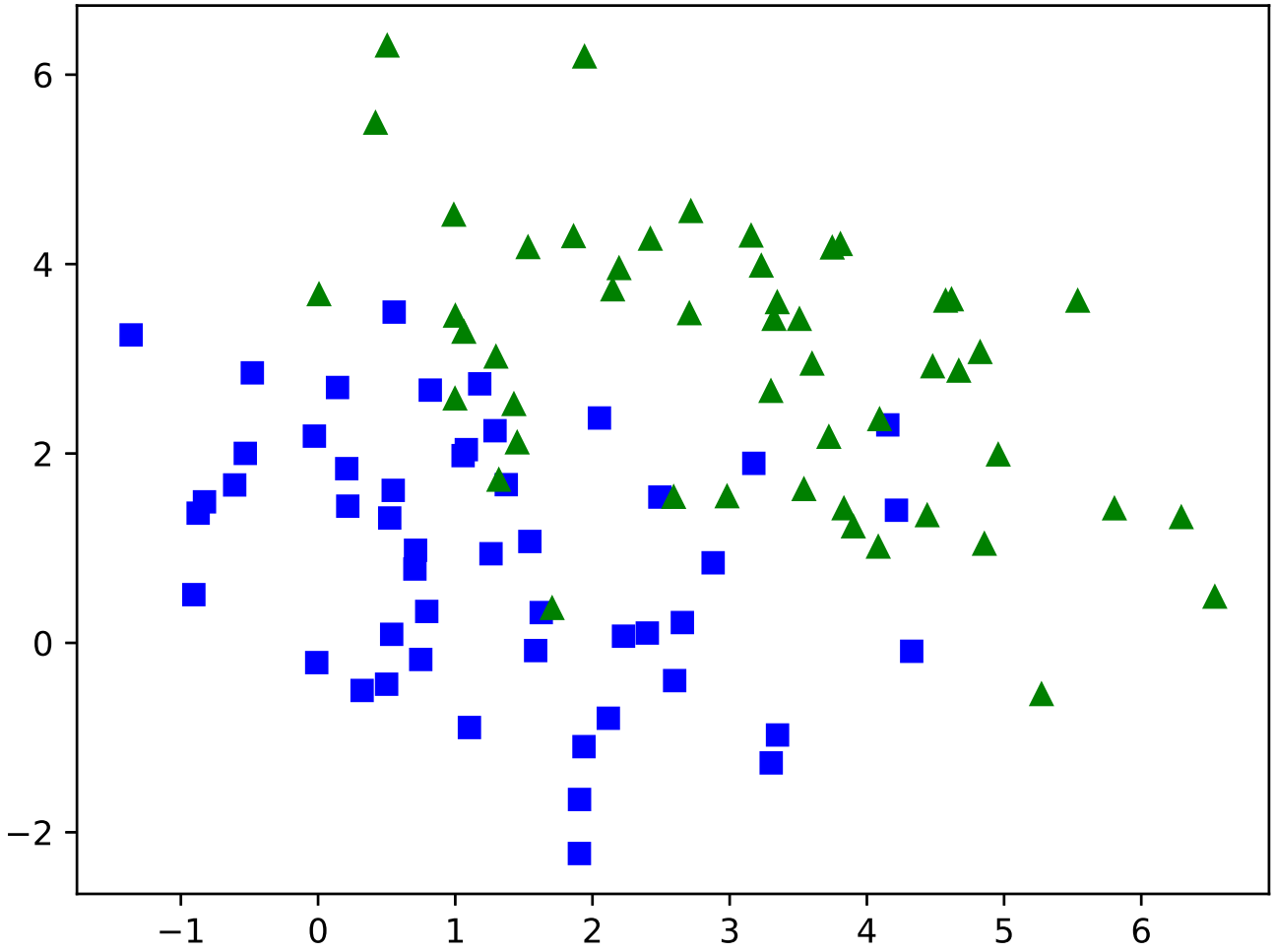
A hyperplane in $\mathbb{R}^n$ is an n-1 dimensional subspace

Método dos Mínimos Quadrados: Python

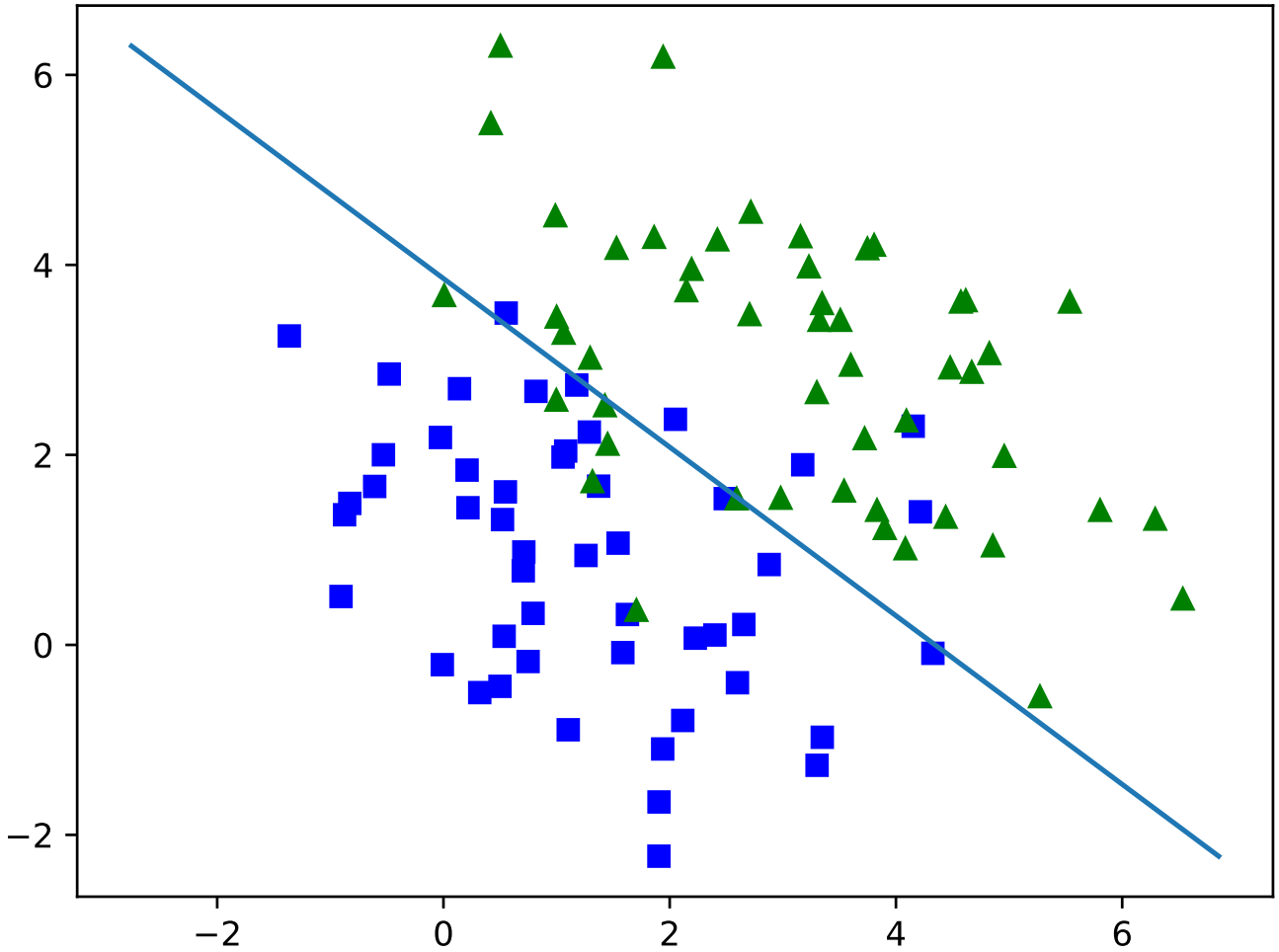
- Arquivo dos dados de entrada: `class_1_tr_X.dat`
- Arquivo dos valores esperados: `class_1_tr_Y.dat`

```
x1,x2                                   y
4.1525267236847 7,2.30191850430311      -1
1.37166852444448,1.67089522310923       -1
1.62651299618563,0.32002267025167 5     -1
1.90556026445907,-1.65342465419423      -1
2.11557633795811,-0.795657656851138     -1
...                                     ...
1.42725086615869,2.51795284362126       1
5.80426651901857,1.41610146547005       1
5.2727032012369,-0.544831984744078      1
3.30064702289753,2.65651491151454       1
```

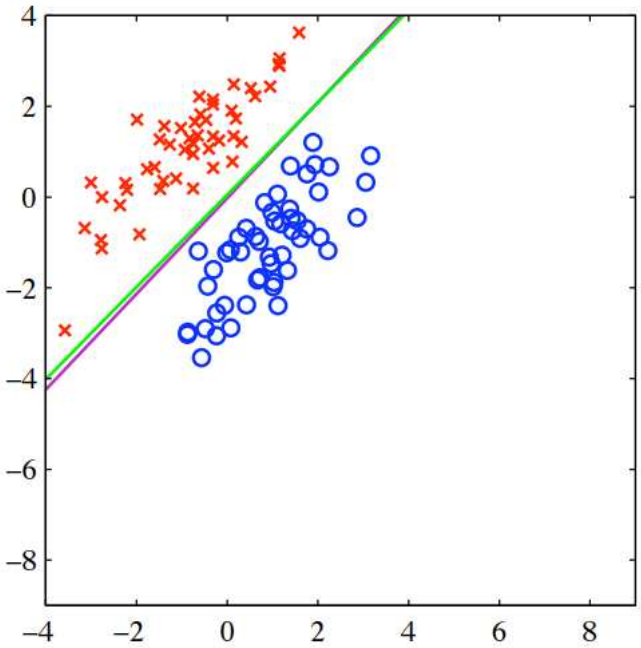
Classificação via Mínimos Quadrados

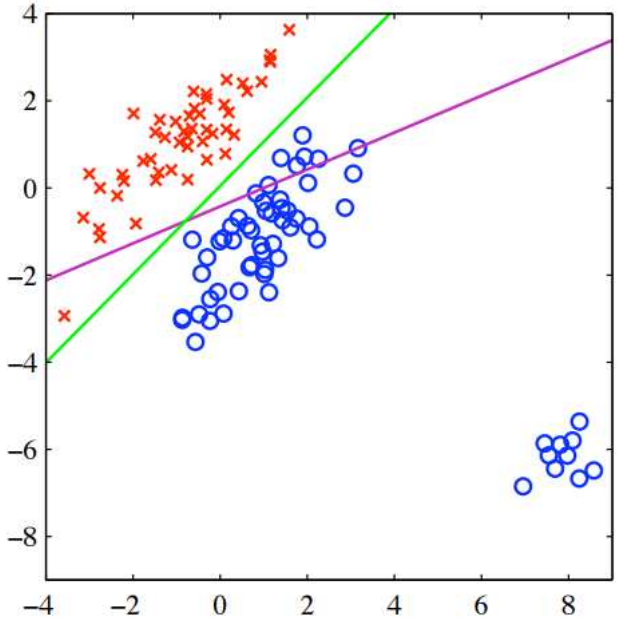


Classificação via Mínimos Quadrados



Classificação via Mínimos Quadrados



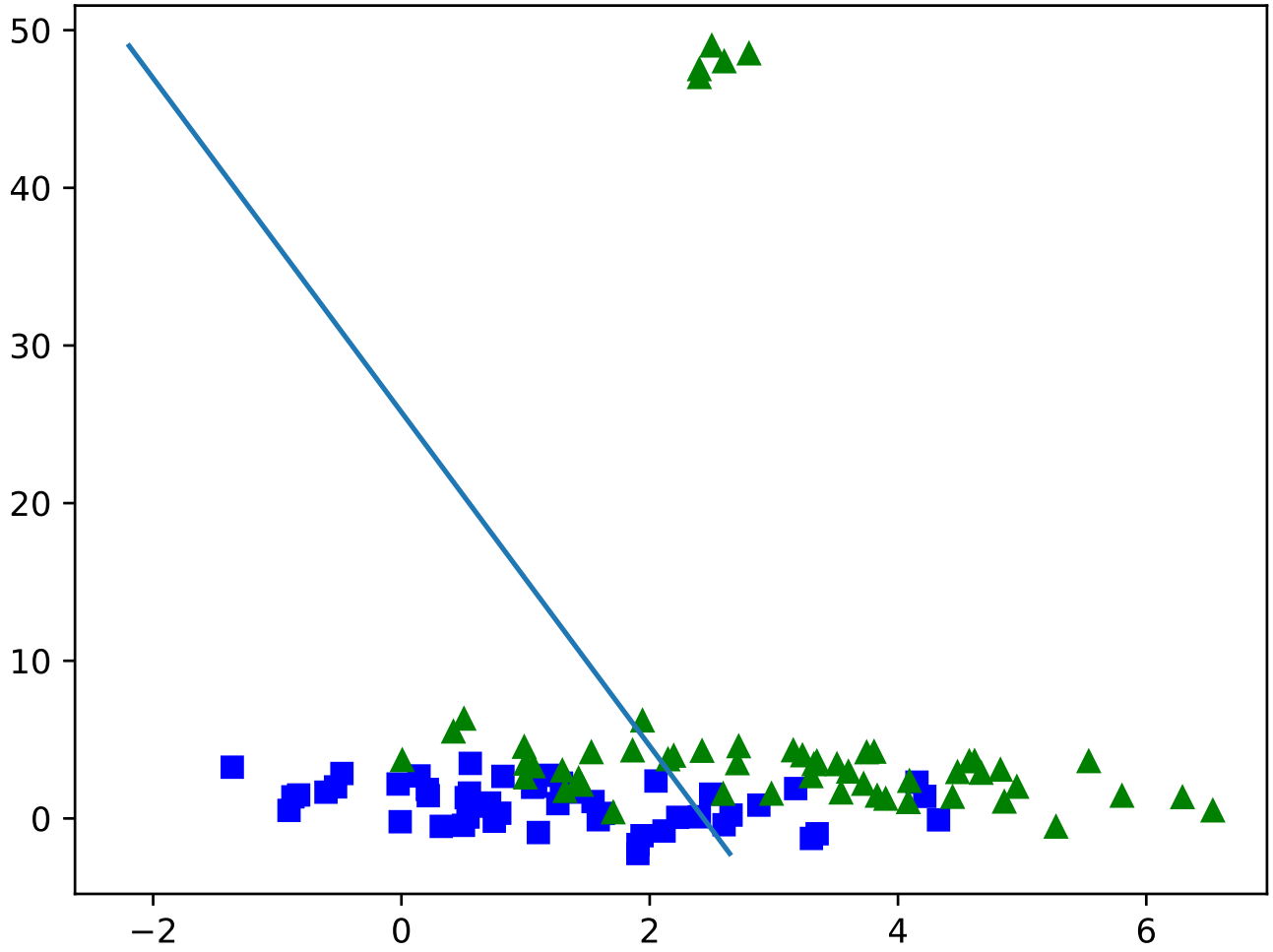


Método dos Mínimos Quadrados: Python

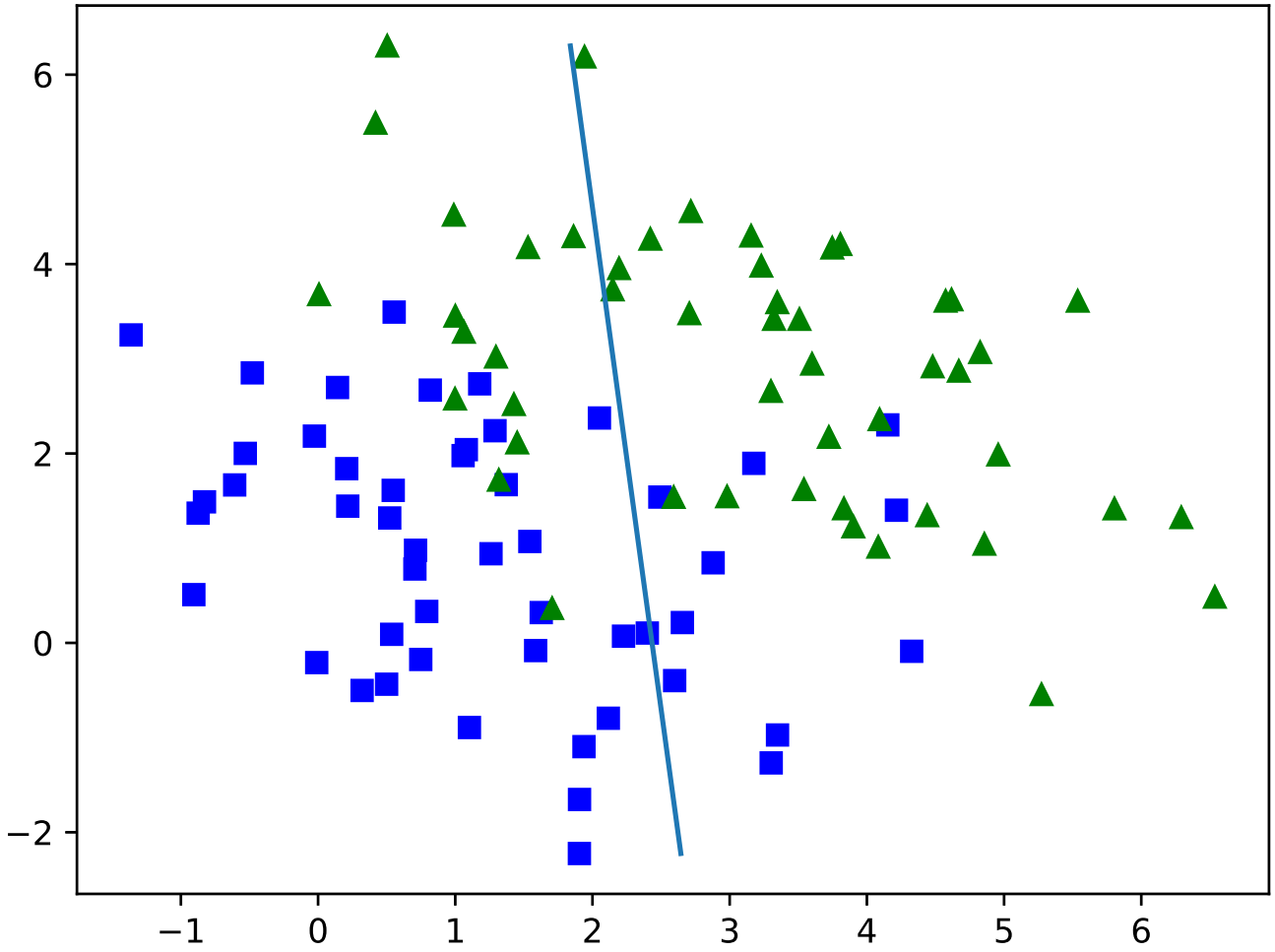
- Acrescentando os pontos que seguem nos arquivos de dados

x1,x2	y
2.5,49	1
2.6,48	1
2.4,47	1
2.8,48.5	1
2.4,47.5	1

Classificação via Mínimos Quadrados



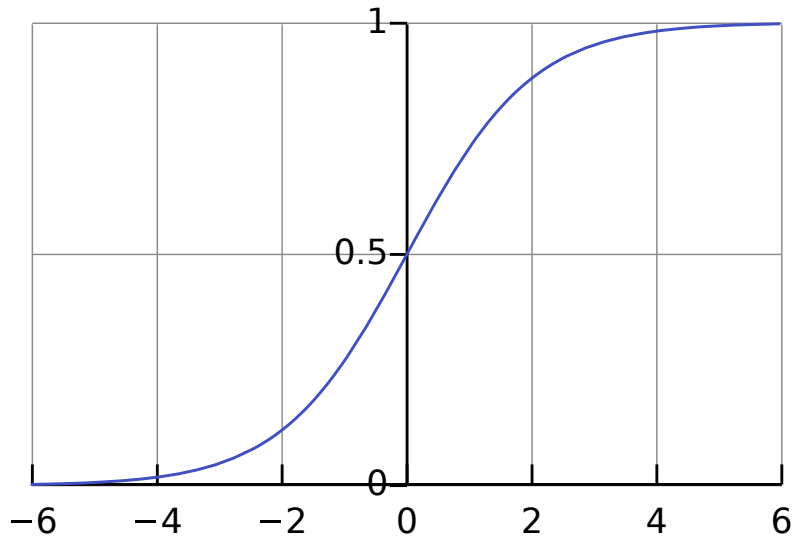
Classificação via Mínimos Quadrados



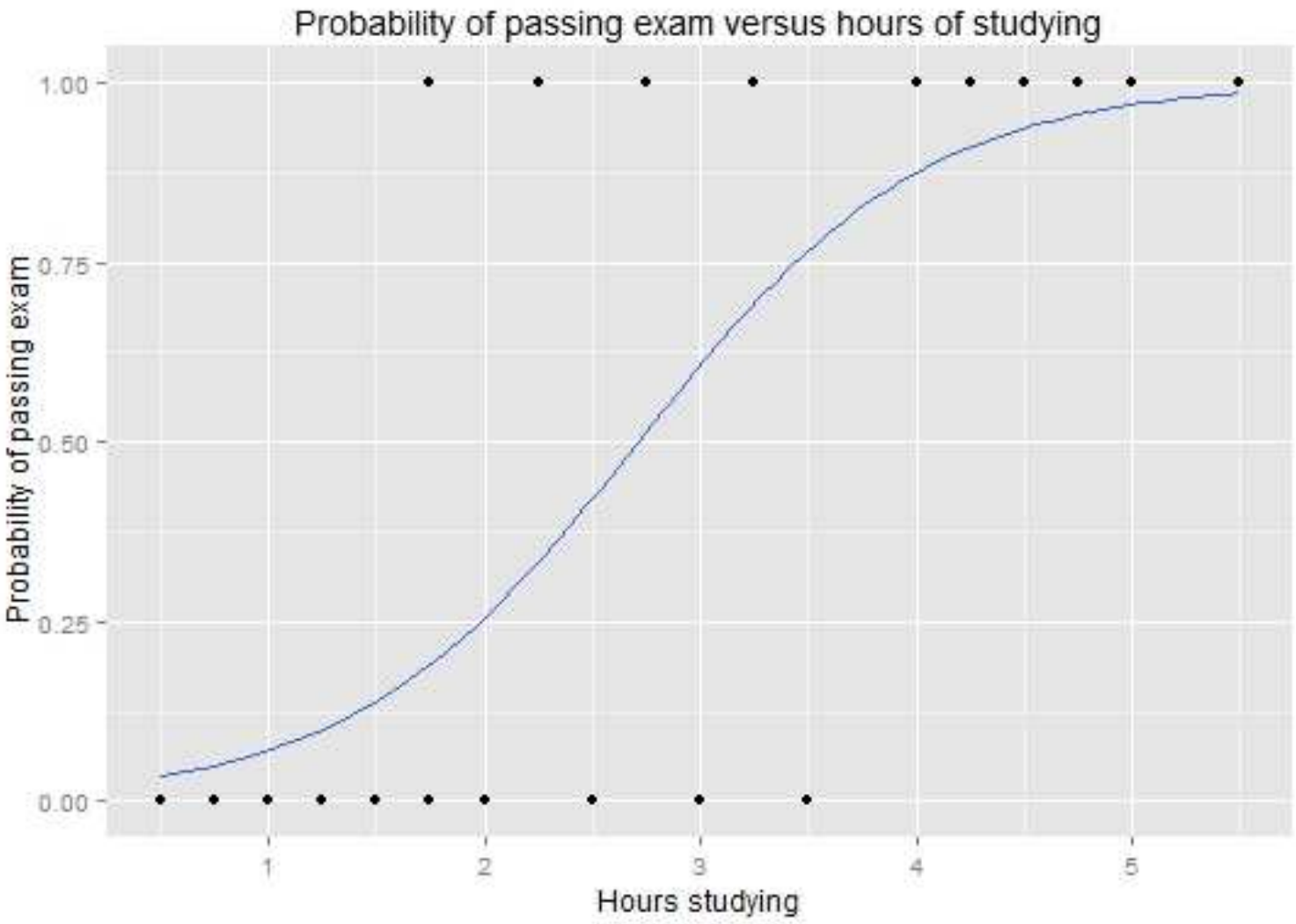
Regressão Logística

Regressão Logística

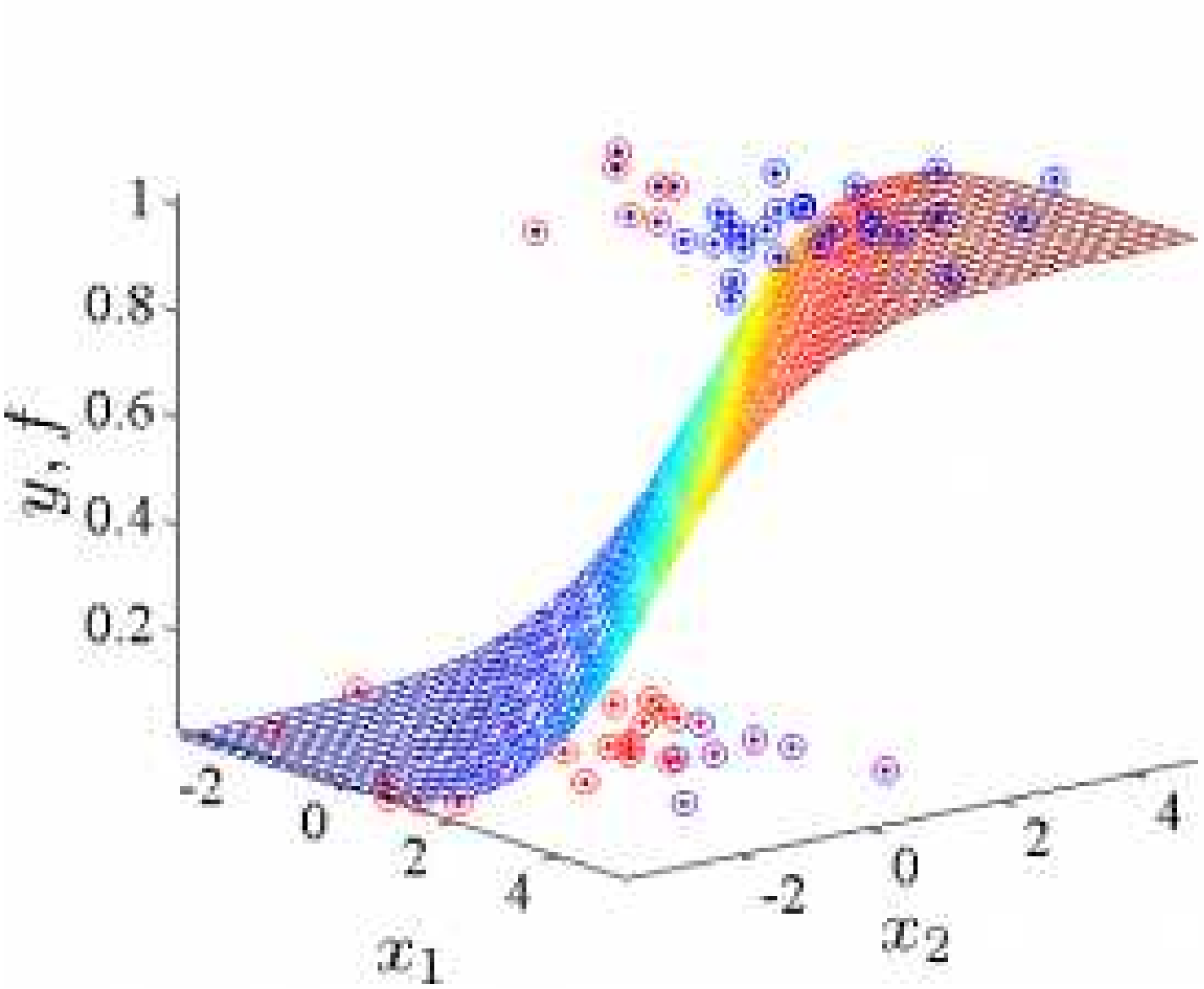
- $g(\mathbf{x}) = \dfrac{1}{1 + e^{-\mathbf{c}^\mathsf{T}\mathbf{x}}}$
- $g(\mathbf{x})$ pode ser interpretada como a probabilidade de x estar associado a um valor esperado 1 ($P(Y = 1|X)$)
- Função logística inversa: $\ln\left(\dfrac{g(\mathbf{x})}{1 - g(\mathbf{x})}\right) = \mathbf{c}^\mathsf{T}\mathbf{x}$



Regressão Logística



Regressão Logística



Regressão Logística

- Para ajustar o modelo logístico, usa-se a verossimilhança dada por

$$L(\mathbf{c}) = \prod_{i=1}^{n} g(\mathbf{x}_i)^{y_i}(1 - g(\mathbf{x}_i))^{1-y_i}; \quad y_i \in \{0, 1\}$$

- Para facilitar o cálculo de derivadas parciais, normalmente adota-se o $\ln$ desta função

$$l(\mathbf{c}) = \sum_{i=1}^{n} y_i \ln(g(\mathbf{x}_i)) + (1 - y_i) \ln(1 - g(\mathbf{x}_i))$$

- A função custo (a ser minimizada) passa a ser $J(\mathbf{c}) = -l(\mathbf{c})$
- O problema de otimização envolve montar um sistema de equações
- O sistema é formado pelas derivadas parciais de $l(\mathbf{c})$
- O cálculo dos coeficientes do modelo pode ser feito via Gradiente Descendente (Estocástico)

Regressão Logística

- Método dos Gradientes

$$\mathbf{c} = \mathbf{c} + \eta \Delta J$$

- Onde

$$a_j = \mathbf{c}^\mathsf{T} \mathbf{x}_j$$

$$J(\mathbf{c}) = -l(\mathbf{c})$$

$$\frac{\partial J}{\partial c_i} = \sum_{j=0}^{N} \frac{\partial J}{\partial g(\mathbf{x}_j)} \frac{\partial g(\mathbf{x}_j)}{\partial a_j} \frac{\partial a_j}{\partial c_i}$$

$$\frac{\partial J}{\partial g(\mathbf{x}_j)} = -y_j \frac{1}{g(\mathbf{x}_j)} - (1 - y_j) \frac{1}{1 - g(\mathbf{x}_j)}$$

$$\frac{\partial g(\mathbf{x}_j)}{\partial a_j} = g(\mathbf{x}_j)(1 - g(\mathbf{x}_j))$$

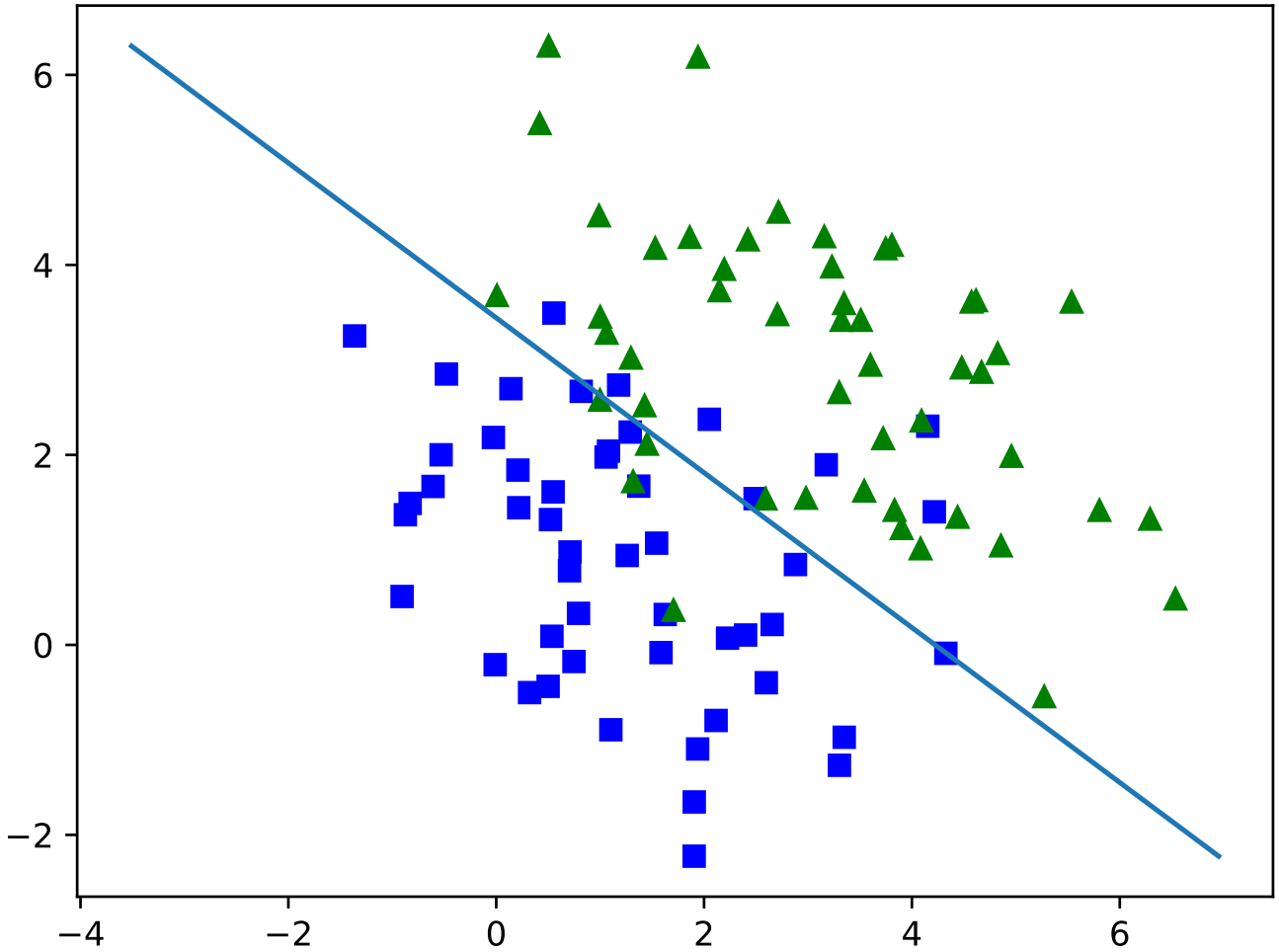
$$\frac{\partial a_j}{\partial c_i} = x_{ji}$$

Regressão Logística: Python

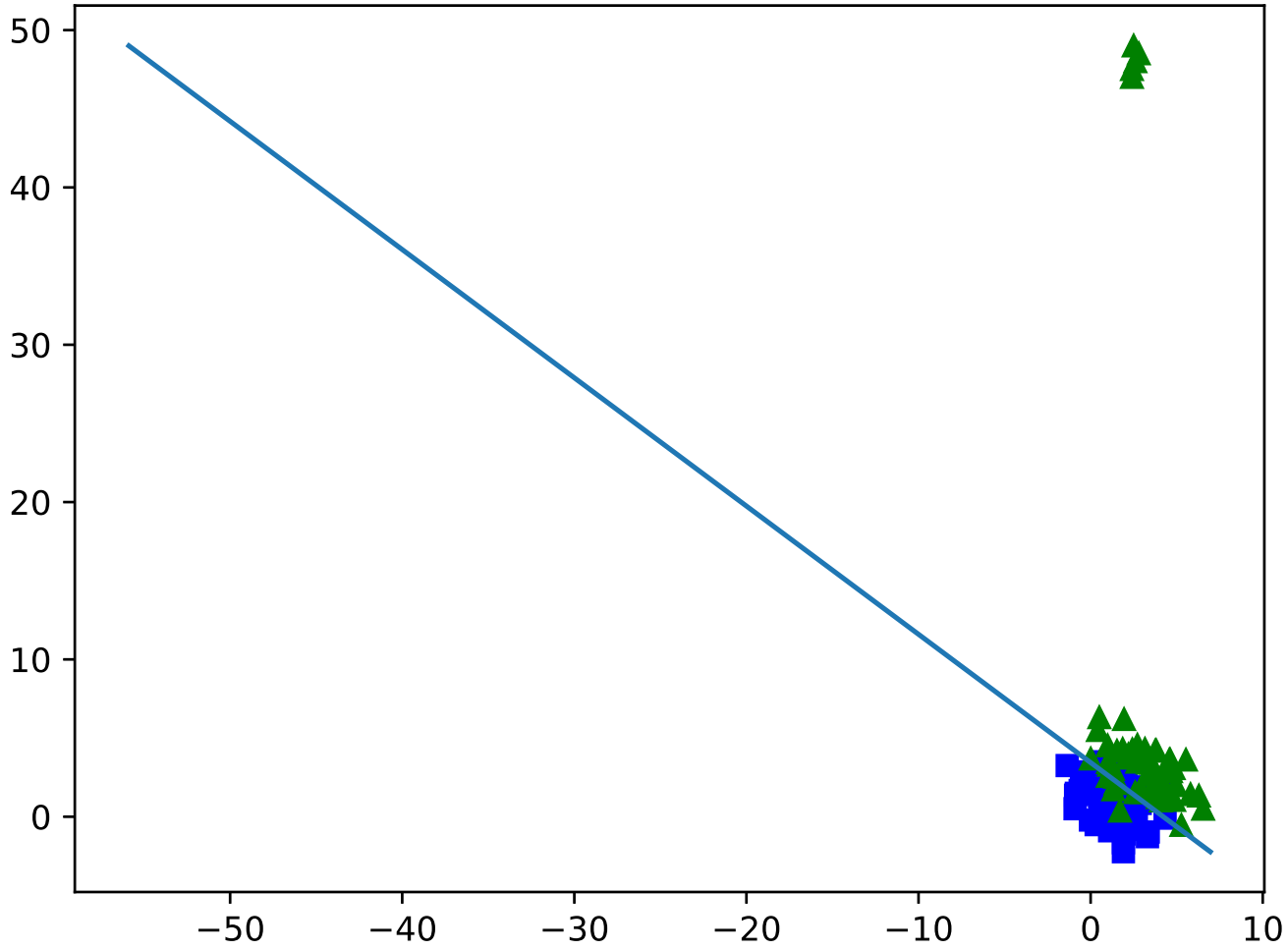
- Arquivo dos dados de entrada: `class_1_tr_X.dat`
- Arquivo dos valores esperados: `class_1_tr_Y.dat`

```
x1,x2                                   y
4.1525267236847 7,2.30191850430311      0
1.37166852444448,1.67089522310923       0
1.62651299618563,0.32002267025167 5     0
1.90556026445907,-1.65342465419423      0
2.11557633795811,-0.795657656851138     0
...                                     ...
1.42725086615869,2.51795284362126       1
5.80426651901857,1.41610146547005       1
5.2727032012369,-0.544831984744078      1
3.30064702289753,2.65651491151454       1
```

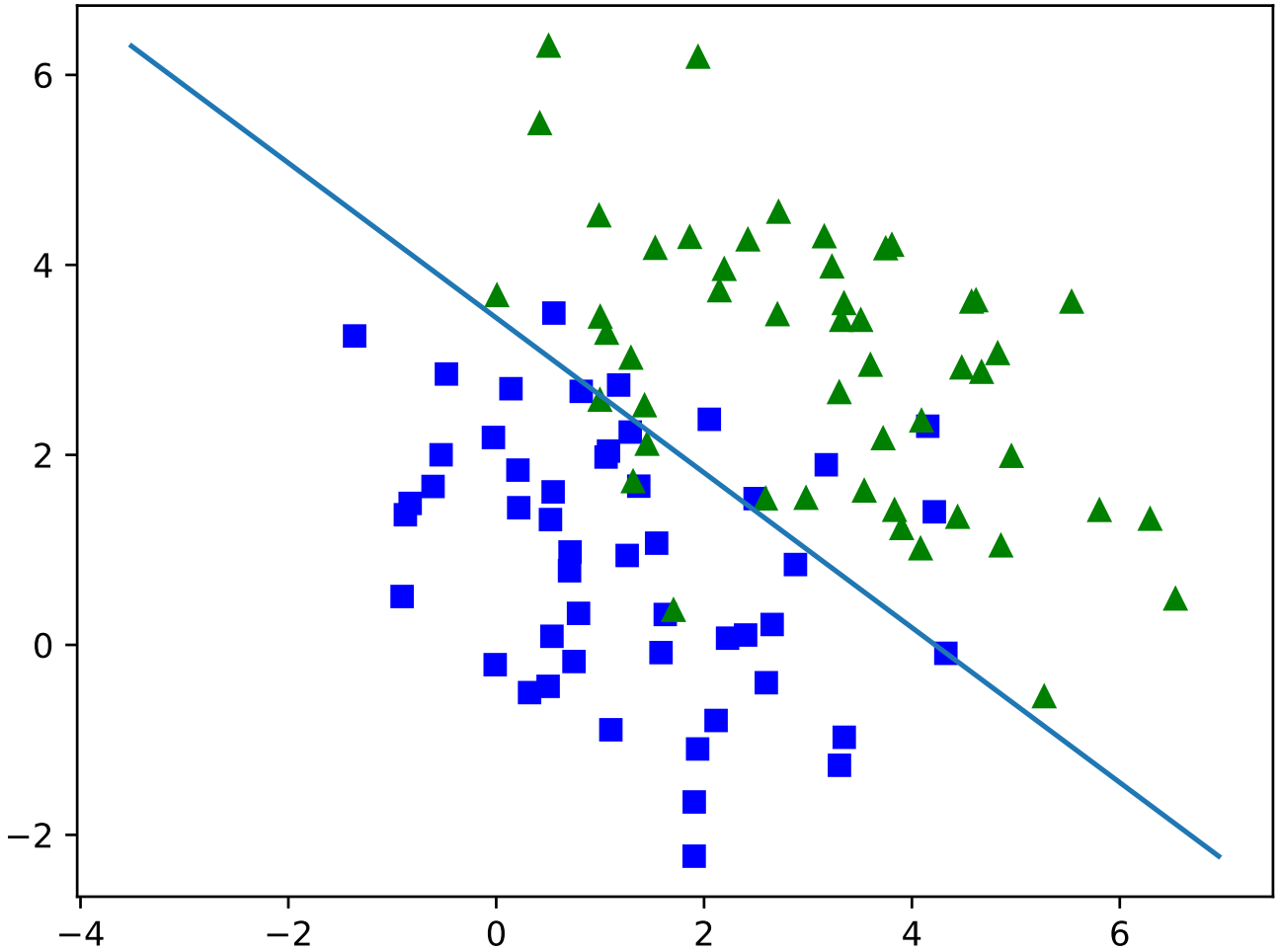
Regressão Logística: Python



Regressão Logística: com dados discrepantes



Regressão Logística: com dados discrepantes



Regressão Logística: Interpretação do Modelo

- Considerando o caso unidimensional, a razão de chance (OR) ao tomar dois valores distintos x_i e x_{i+1} é

$$OR = \frac{g(x_{i+1})}{g(x_i)} = \frac{e^{c_0+c_1 x_{i+1}}}{e^{c_0+c_1 x_i}} = e^{c_1(x_{i+1}-x_i)}$$

- Fazendo $x_{i+1} - x_i = 1$, então $OR = e^{c_1}$

- Pode-se interpretar que

$$OR > 1 \Rightarrow g(x_{i+1}) > g(x_i)$$

e que

$$OR < 1 \Rightarrow g(x_{i+1}) < g(x_i)$$

- Quanto maior o valor do coeficiente (em módulo), maior a importância do atributo para o modelo

Regressão Logística: Interpretação do Modelo

- Adaptado do exemplo apresentado em `http://www.estatisticacomr.uff.br/?p=598`
- $n = 30$ instâncias
- Variável dependente: autoavaliação de saúde (0=não boa, 1=boa)
- idade variando de 20 a 95 anos
- renda familiar per capita (1=Mais de 3 s.m, 0= Até 3 s.m=base)

Naïve Bayes

Naïve Bayes

- Classificador estatístico
  - realiza predições probabilísticas
- Fundamento
  - Baseado no Teorema de Bayes
- Desempenho
  - Um classificador Naïve Bayes tem desempenho comparável com árvores de decisão e redes neuronais
- Incremental
  - Cada exemplo de treinamento pode afetar incrementalmente a probabilidade da hipótese ser correta

Naïve Bayes: Teorema de Bayes

- Seja D os dados de treinamento, dada uma tupla X, e a probabilidade *à posteriori* $P(H|X)$ da hipótese H, então o teorema de Bayes é dado por

$$P(H|X) = \frac{P(X|H)P(H)}{P(X)}$$

- Observa-se que

$$P(H|X)P(X) = P(H \cap X) = P(X \cap H) = P(X|H)P(H)$$

- Informalmente, pode-se dizer que

$$\textit{à posteriori} = \frac{\text{verossimilhança} \times \textit{à priori}}{\text{evidência}}$$

- Dificuldade prática: requer o conhecimento inicial de muitas probabilidades, o que pode resultar em um alto custo computacional

Naïve Bayes: Teorema de Bayes

- Teorema de Bayes: $P(H|X) = \frac{P(X|H)P(H)}{P(X)}$
  - X é uma amostra (evidência): classe desconhecida
  - H é a hipótese de que X pertence à classe C
  - A classificação envolve a determinação de $P(H|X)$: a probabilidade de que a hipótese é atendida dada a amostra X
  - $P(H)$ é a probabilidade *à priori*
  - $P(H)$ indica, por exemplo, a probabilidade de X comprar um computador independente da idade, renda, etc
  - $P(X)$ é a probabilidade da observação da amostra
  - $P(X|H)$ é a verossimilhança: probabilidade de observar X dado que a hipótese ocorra
  - Por exemplo, dado que X irá comprar um computador, $P(X|H)$ indica a probabilidade de X ter idade entre 31 e 40, possuir uma renda média, etc
- Indica-se que X pertence à classe C_i se e somente se a probabilidade $P(C_i|X)$ é a mais alta entre todas as probabilidades $P(C_k|X)$, para todas as k classes

Naïve Bayes

- Seja D o conjunto de instâncias de treinamento contendo os valores esperados de suas classes
- Cada instância é representada por um vetor X de dimensão n
- Supondo que existam m classes, ou seja, $C_1, \ldots, C_m$
- A classificação é feita determinando a máximo probabilidade *à posteriori*, ou seja, identificando C_i que gera o máximo valor de $P(C_i|X)$
- Isso pode ser derivado do Teorema de Bayes
- Observa-se que $P(X)$ é constante para todas as classes, logo apenas $P(X|C_i)P(C_i)$ precisa ser maximizado

Classificador Naïve Bayes

- Assume-se que os dados são condicionalmente independentes

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i)$$

- Precisa-se apenas "contar" as ocorrências das classes
- Se A_k é categórico, $P(x_k|C_i)$ é o número de tuplas em C_i que tem o valor x_k para o atributo A_k dividido por $|C_{i,D}|$
  - $|C_{i,D}|$ é o número de tuplas de C_i em D
- Se A_k é contínuo, $P(x_k|C_i)$ é normalmente computado baseado numa distribuição Gaussiana com média μ e desvio padrão σ

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

Exemplo

Class:
C1:buys_computer = ‘yes’
C2:buys_computer = ‘no’

Data to be classified:
X = (age <=30,
Income = medium,
Student = yes
Credit_rating = Fair)

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

Exemplo

age	income	student	credit_rating	com
<=30	high	no	fair	no
<=30	high	no	excellent	no
31..40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31..40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31..40	medium	no	excellent	yes
31..40	high	yes	fair	yes
>40	medium	no	excellent	no

- $P(C_i)$: P(buys_computer = "yes") = 9/14 = 0.643
  P(buys_computer = "no") = 5/14= 0.357
- Compute $P(X|C_i)$ for each class
  P(age = "<=30" | buys_computer = "yes") = 2/9 = 0.222
  P(age = "<= 30" | buys_computer = "no") = 3/5 = 0.6
  P(income = "medium" | buys_computer = "yes") = 4/9 = 0.444
  P(income = "medium" | buys_computer = "no") = 2/5 = 0.4
  P(student = "yes" | buys_computer = "yes) = 6/9 = 0.667
  P(student = "yes" | buys_computer = "no") = 1/5 = 0.2
  P(credit_rating = "fair" | buys_computer = "yes") = 6/9 = 0.667
  P(credit_rating = "fair" | buys_computer = "no") = 2/5 = 0.4
- **X = (age <= 30 , income = medium, student = yes, credit_rating = fair)**

$P(X|C_i)$: P(X|buys_computer = "yes") = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
P(X|buys_computer = "no") = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

$P(X|C_i)*P(C_i)$: P(X|buys_computer = "yes") * P(buys_computer = "yes") = 0.028
P(X|buys_computer = "no") * P(buys_computer = "no") = 0.007

Therefore, X belongs to class ("buys_computer = yes")

Problema

- Naïve Bayes requer que todas as probabilidades condicionais sejam diferentes de zero
  - Caso contrário, $P(X|C_i) = 0$
- Por exemplo, haveria um problema se num banco de dados com 1000 tuplas a informação de "income" tiver
  - 0 low
  - 990 medium
  - 10 high
- Para contornar essa situação, pode-se adotar a correção de Laplace
  - Soma-se 1 em cada caso

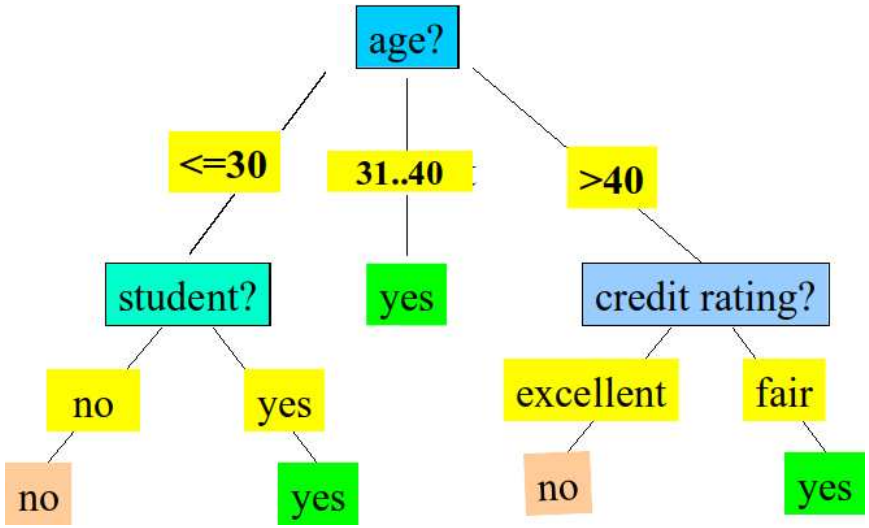
Árvores de Decisão

Árvores de Decisão

- Estrutura em forma de árvore
- Cada nó interno corresponde a um teste envolvendo atributos
  - Normalmente, um atributo é considerado por nó
- Os nós folhas representam classes ou distribuições de classes
- Os caminhos da raiz até as folhas correspondem às regras de classificação
- Vantagens
  - Concisa
  - Fácil de visualizar
  - Interpretável
  - Trata tipos diferentes e multidimensionais de dados
  - Fornece modelos com boa acurácia

Exemplo: compram computador

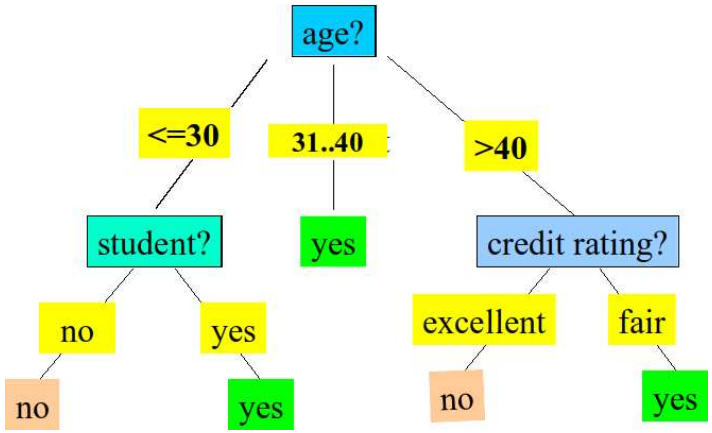
age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no



Árvores de Decisão

- Uma vez construída a árvore, ela pode ser usada como classificador
  - Testar os valores dos atributos da nova instância nos nós da árvore até atingir um nó folha

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

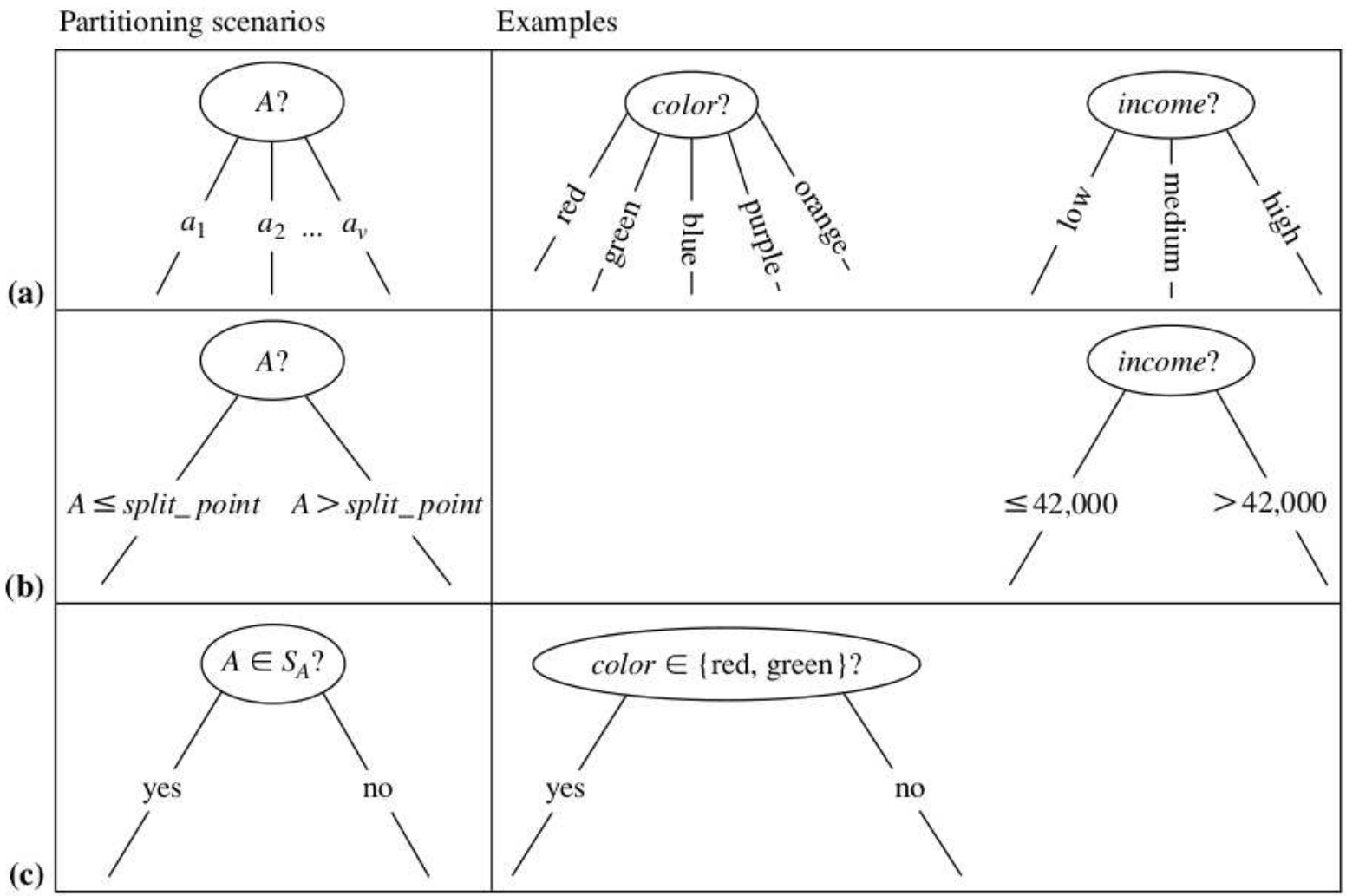


Algoritmo Básico

- Algoritmo básico para a indução de árvores de decisão
  - Algoritmo guloso
  - A árvore é criada recursivamente de cima para baixo (top-down) por divisão e conquista
  - Inicialmente todos os dados de treinamento estão na raiz
  - Neste algoritmo básico, todos os dados são considerados categóricos
    - valores contínuos devem ser discretizados
  - As instâncias são particionadas recursivamente com base num dos atributos
  - A seleção do atributo é feita por uma heurística ou medida estatística
    - por exemplo, ganho de informação

- Critérios de Parada
  - Todos os dados num certo nó pertencem a uma única classe
  - Não há mais atributos para particionar os dados
  - Não há mais instâncias num ramo da árvore

Partição dos Dados



Seleção de Atributo

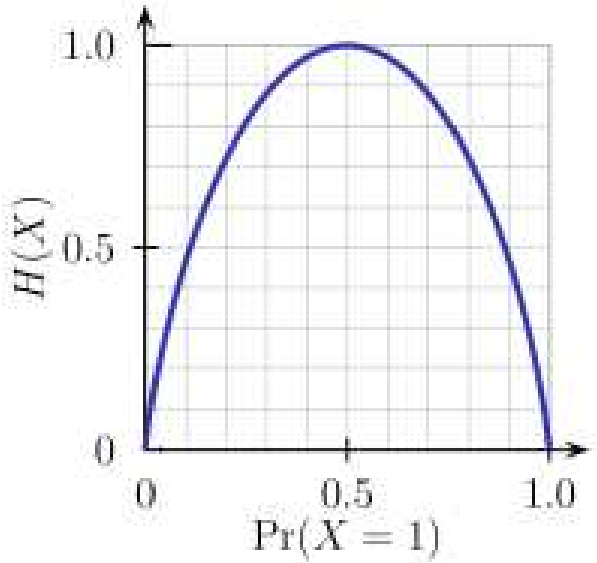
- É necessário definir uma medida para ser usada no critério de separação dos dados
- Idealmente, deseja-se particionar os dados de modo que os subconjuntos sejam puros
  - todas as instâncias pertencem à mesma classe
- Em alguns casos, pode ser necessário definir também o ponto de separação
  - valor do atributo contínuo
  - estrutura da árvore fica limitada ao caso binário
- Três alternativas populares
  - Ganho de informação: ID3
  - Razão de ganho: C4.5
  - Índice Gini: CART

Entropia: Breve Introdução

- Teoria da Informação
- Mede a incerteza relacionada a uma variável aleatória
- Para uma variável aleatória discreta Y e m valores $\{y_1, \ldots, y_m\}$

$$H(Y) = -\sum_{i=1}^{m} p_i \log_2(p_i), \text{ onde } p_i = P(Y = y_i)$$

- Interpretação
  - Valores grandes: alta incerteza
  - Valores pequenos: baixa incerteza
- Para $m = 2$:



Ganho de Informação

- Diversas medidas podem ser adotadas para selecionar atributos
- Muitas vezes os atributos com maior ganho de informação são selecionados
  - Medida adotada no ID3 e C4.5

Ganho de Informação

- Seja p_i a probabilidade de uma tupla arbitrária de D pertencer à classe C_i, ou seja, $p_i = \frac{|C_{i,D}|}{|D|}$
- Estimativa da quantidade de informação necessária para classificar uma tupla em D

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

- Quanto maior a impuridade dos dados, mais informação é necessária para a classificação
- $Info(D)$ também é chamado de **entropia**

Ganho de Informação

- Deseja-se agora separar as instâncias em D levando-se em consideração algum atributo A que possui v valores distintos
- Neste ponto, os tipos dos dados são considerados
  - definições para as partições
- As partições devem ser as mais puras possíveis
  - idealmente puras
- Quanta informação ainda é necessária para ter uma classificação exata (após a separação dos dados em relação a A)?

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} Info(D_j)$$

- O termo $\frac{|D_j|}{|D|}$ opera como uma ponderação
- Menor quantidade de informação ainda necessária para particionar os dados $\Rightarrow$ maior a pureza do subconjunto

Ganho de Informação

- Sendo

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} Info(D_j)$$

- O ganho de informação ao ramificar os dados utilizando o atributo A é então definido como

$$Gain(A) = Info(D) - Info_A(D)$$

- $Gain(A)$ indica o quanto se ganha ao particionar os dados usando o atributo A

- O atributo que maximiza o ganho de informação é escolhido para gerar o nó e seus valores representam as arestas para os nós do próximo nível

Ganho de Informação

- Class P: buys_computer = "yes"
- Class N: buys_computer = "no"

$$Info(D) = I(9,5) = -\frac{9}{14}\log_2(\frac{9}{14}) - \frac{5}{14}\log_2(\frac{5}{14}) = 0.940$$

age	p_i	n_i	$I(p_i, n_i)$
<=30	2	3	0.971
31…40	4	0	0
>40	3	2	0.971

age	income	student	credit_rating	buys_computer
<=30	high	no	fair	no
<=30	high	no	excellent	no
31…40	high	no	fair	yes
>40	medium	no	fair	yes
>40	low	yes	fair	yes
>40	low	yes	excellent	no
31…40	low	yes	excellent	yes
<=30	medium	no	fair	no
<=30	low	yes	fair	yes
>40	medium	yes	fair	yes
<=30	medium	yes	excellent	yes
31…40	medium	no	excellent	yes
31…40	high	yes	fair	yes
>40	medium	no	excellent	no

$$Info_{age}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694$$

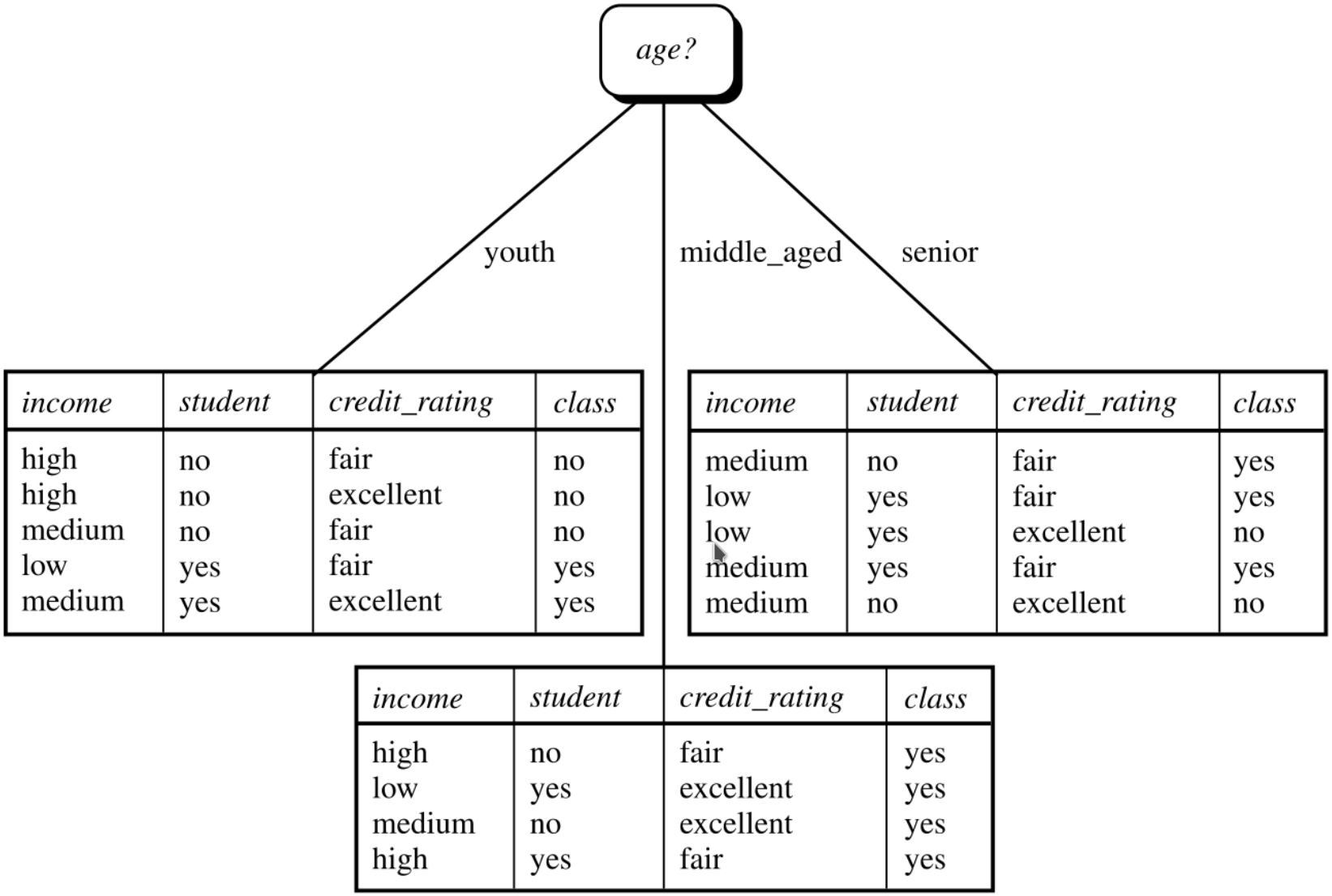
$\frac{5}{14}I(2,3)$ means "age <=30" has 5 out of 14 samples, with 2 yes's and 3 no's. Hence

$$Gain(age) = Info(D) - Info_{age}(D) = 0.246$$

Similarly,

$$Gain(income) = 0.029$$
$$Gain(student) = 0.151$$
$$Gain(credit\_rating) = 0.048$$

Ganho de Informação



income	*student*	*credit_rating*	*class*
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	*student*	*credit_rating*	*class*
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

income	*student*	*credit_rating*	*class*
high	no	fair	yes
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

C4.5: Ganho de Informação em Atributos Contínuos

- Seja A um atributo com valores contínuos
- Deve-se definir pontos de separação dos dados considerando A
  - Ordena-se os dados em relação a A
  - Identifica-se as instâncias adjacentes com mesma classificação
  - O ponto de separação entre os grupos pode ser o ponto médio entre os valores que limitam os grupos

Temperature	40	48	60	72	80	90
PlayTennis	No	No	Yes	Yes	Yes	No

- Separações
  - $D1$ é o conjunto de tuplas de D em que $A \leq$ ponto de separação, e $D2$ são aquelas em que $A >$ ponto de separação
  - No exemplo, há 2 atributos lógicos
    - Temperature$_{>54}$ e Temperature$_{>85}$
    - o melhor será aquele com maior ganho de informação

C4.5: Razão de Ganho

- O ganho de informação (adotado diretamente no IDE3) gera um viés para atributos com uma grande quantidade de valores
  - Por exemplo, ID geraria uma quebra em MUITOS conjuntos puros ($Info_{ID}(D) = 0$)

- O C4.5 utiliza uma normalização do ganho de informação

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo_A(D)}$$

onde

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \log_2 \left(\frac{|D_j|}{|D|} \right)$$

- Esse valor representa o potencial da informação gerada ao separar os dados de treinamento D em v partições
  - Possíveis valores do atributo A

C4.5: Razão de Ganho

- O $SplitInfo_A(D)$ representa a entropia do atributo A sobre o conjunto D
- Pode levar à escolha de um atributo apenas por $SplitInfo_A(D)$ ser pequeno
- A razão torna-se instável à medida que $SplitInfo_A(D)$ se aproxima de zero
- Solução: o método é aplicado aos casos em que $Gain(A)$ assume valores acima da média
- O atributo com maior $GainRatio(A)$ é selecionado

C4.5: Razão de Ganho

- Exemplo:

$$SplitInfo_{income}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557$$

resulta em $GainRatio(\text{income}) = 0.029/1.557 = 0.019$

RID	age	income	student	credit_rating	Class: buys_computer
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no

Gini Index

- Dado um conjunto de dados D que contém exemplos de m classes, então o Índice Gini pode ser definido como

$$gini(D) = 1 - \sum_{j=1}^{m} p_j^2$$

onde p_j é a frequência relativa da classe j em D

- O Índice Gini mede a impuridade de D
- Ao dividir o conjunto D usando A em dois conjuntos D_1 e D_2, então

$$gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$$

- A redução em impuridade:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

- O atributo e conjuntos com menor valor de $gini_A(D)$ (maior redução de impuridade) é escolhido como ponto de separação

Computando o Índice Gini

- Por exemplo, D tem 9 instâncias com `buys-computer = ‘‘yes’’` e 5 com `‘‘no’’`

$$gini(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right)^2 = 0.459$$

- Supondo que o atributo `income` particione D em 10 instâncias em D_1 e 4 em D_2 da seguinte forma

$$gini_{income \in \{low, medium\}}(D) = \frac{10}{14} gini(D_1) + \frac{4}{14} gini(D_2) = 0.443$$

- Além disso,

$$gini_{income \in \{low, high\}}(D) = 0.458$$

$$gini_{income \in \{medium, high\}}(D) = 0.450$$

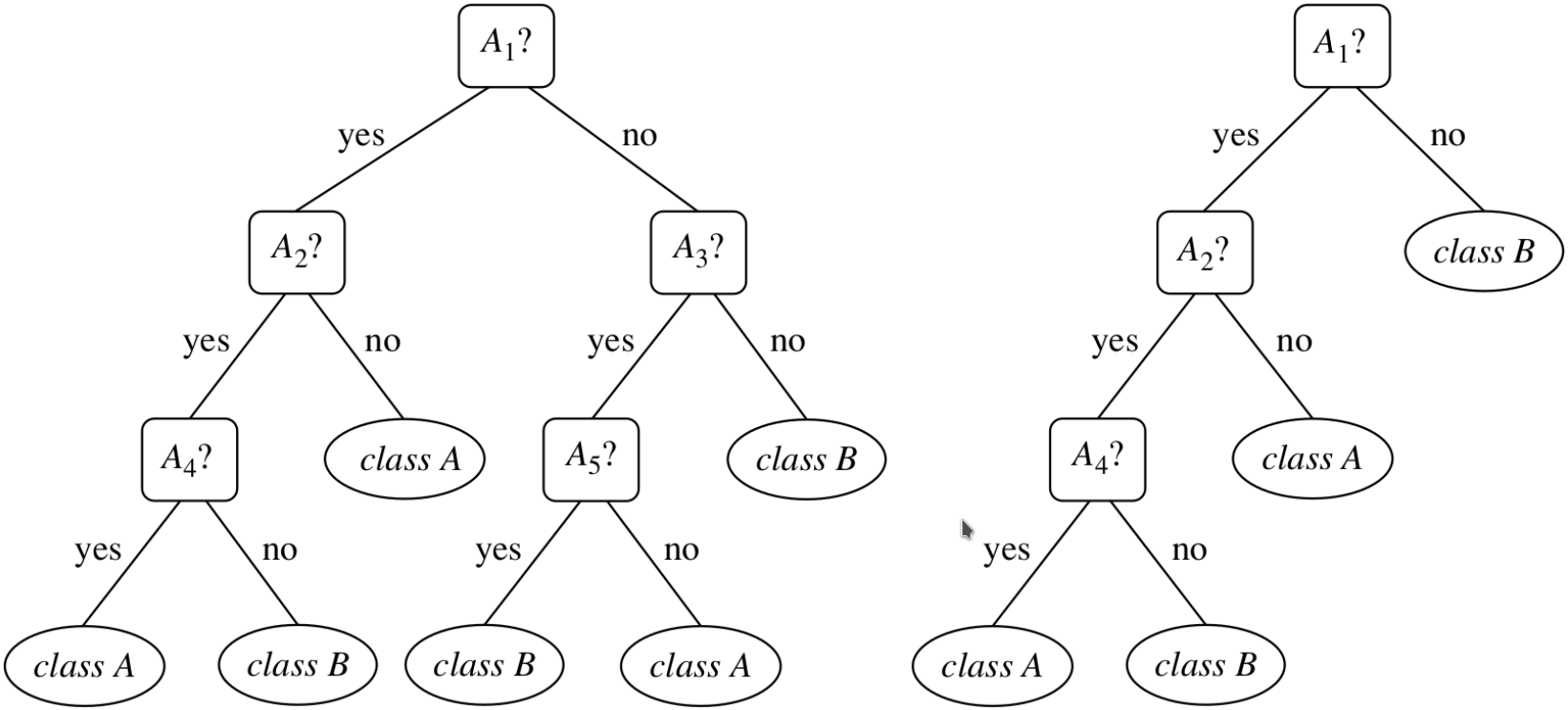
- Assim, os dados devem ser divididos em $\{low, medium\}$ e $\{high\}$

Comparação

- Em geral, as três medidas obtém bons resultados
  - Ganho de informação
    - viés na direção de atributos com muitos valores
  - Razão de ganho
    - tende a preferir separações desbalanceadas, na qual uma partição é muito menor do que outras
  - Índice Gini
    - viés na direção de atributos com muitos valores
    - dificuldade em situações onde há muitas classes
    - tende a equilibrar o tamanho das partições

Poda da Árvore

- Uma árvore induzida pode superajustar os dados
  - Muitos ramos podem refletir anomalias nos ruídos e *outliers*
  - Acurácia ruim em novos dados



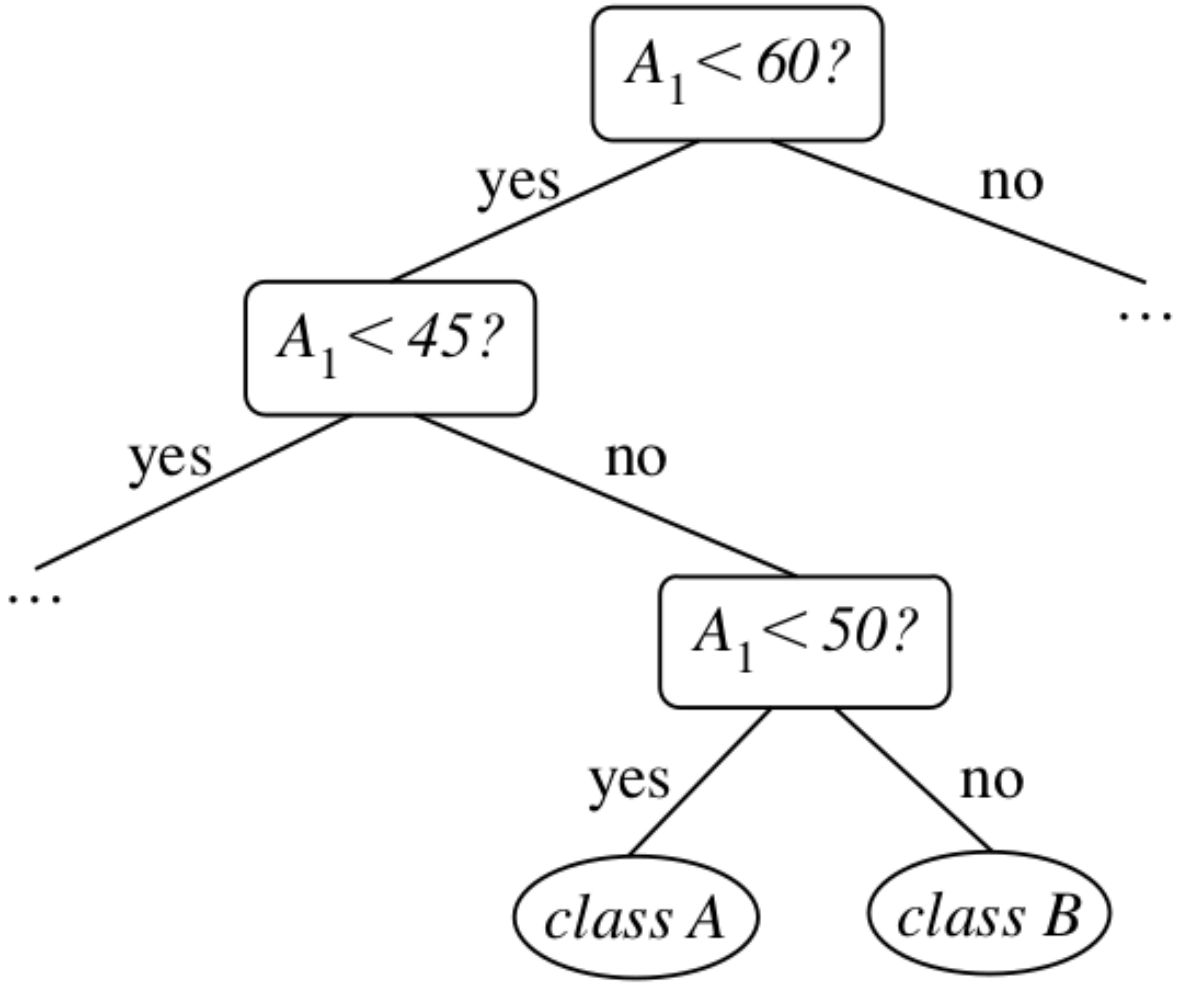
Poda da Árvore

- Duas alternativas para evitar superajuste dos dados
- Pré-poda
  - Limita a construção da árvore antes: não divide um nó com base num limiar de qualidade
  - Difícil de escolher um parâmetro adequado
- Pós-poda
  - Remove ramificações de árvores completamente geradas
  - Pode ser feito usando os dados de treinamento (relação entre o número de folhas e erro da árvore)
  - Pode-se usar um conjunto de dados diferente dos dados de treinamento para indicar a qualidade da poda

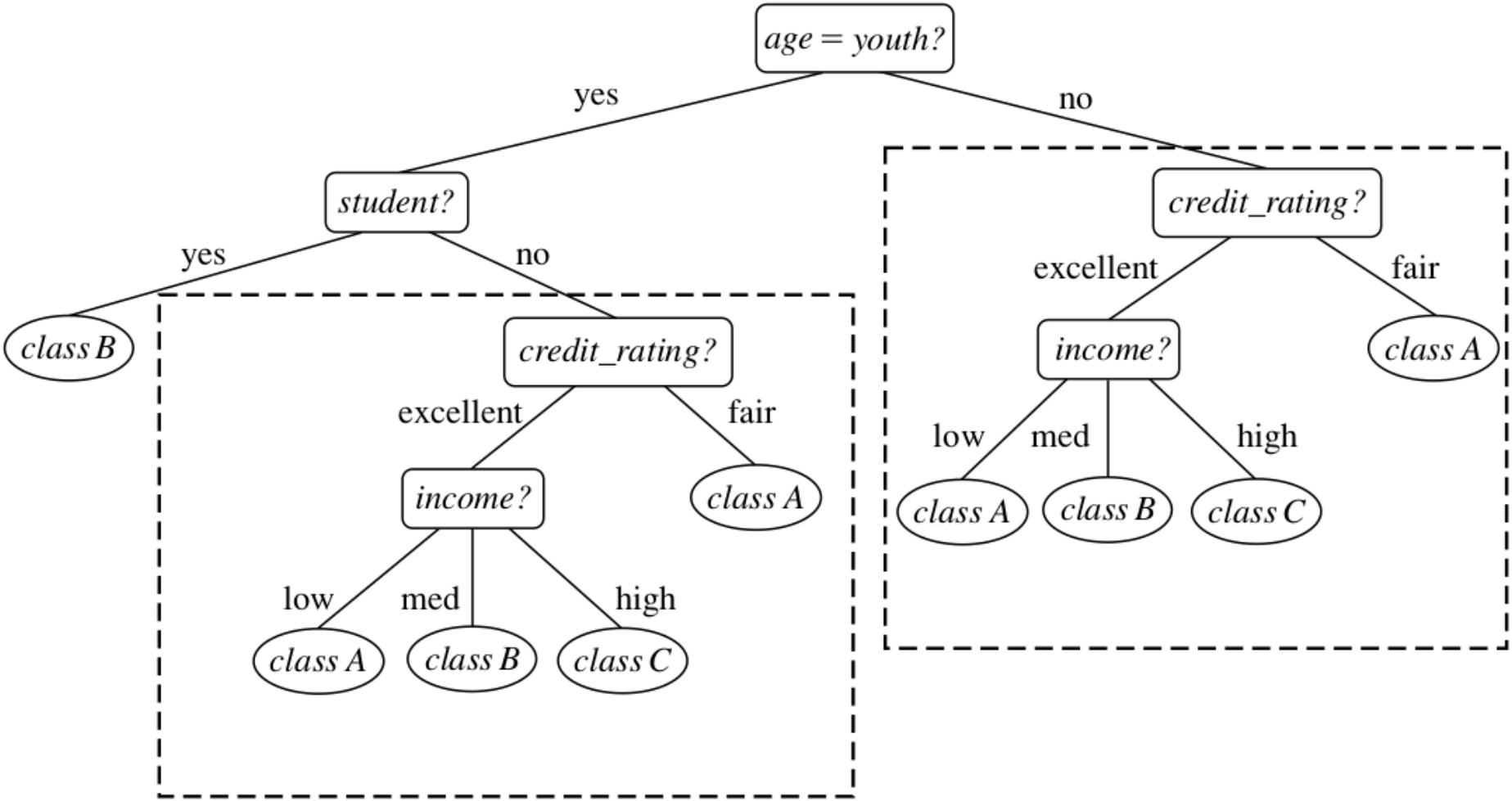
Árvores de Decisão: Características

- Permite atributos contínuos
  - Define novas discretizações dinamicamente com base nos valores dos atributos contínuos e nas classes
- Tratamento de dados faltantes
  - Novo valor
  - Atribuir o valor mais comum
  - Atribuir uma probabilidade para os possíveis valores

Árvores de Decisão – Repetição



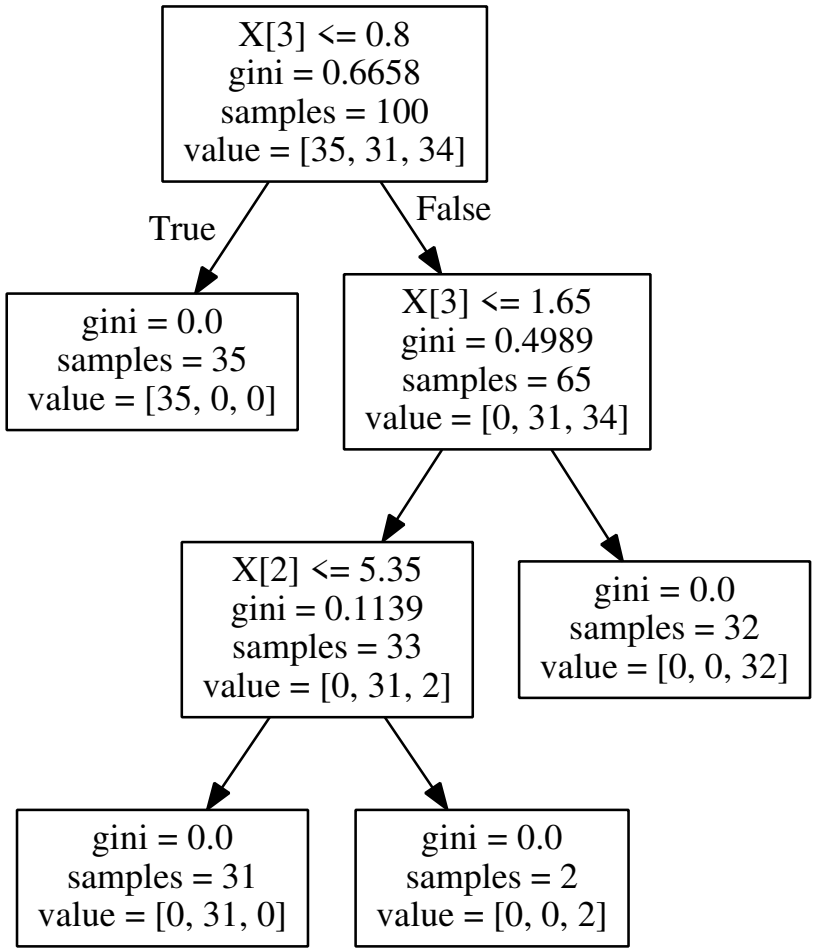
Árvores de Decisão – Replicação



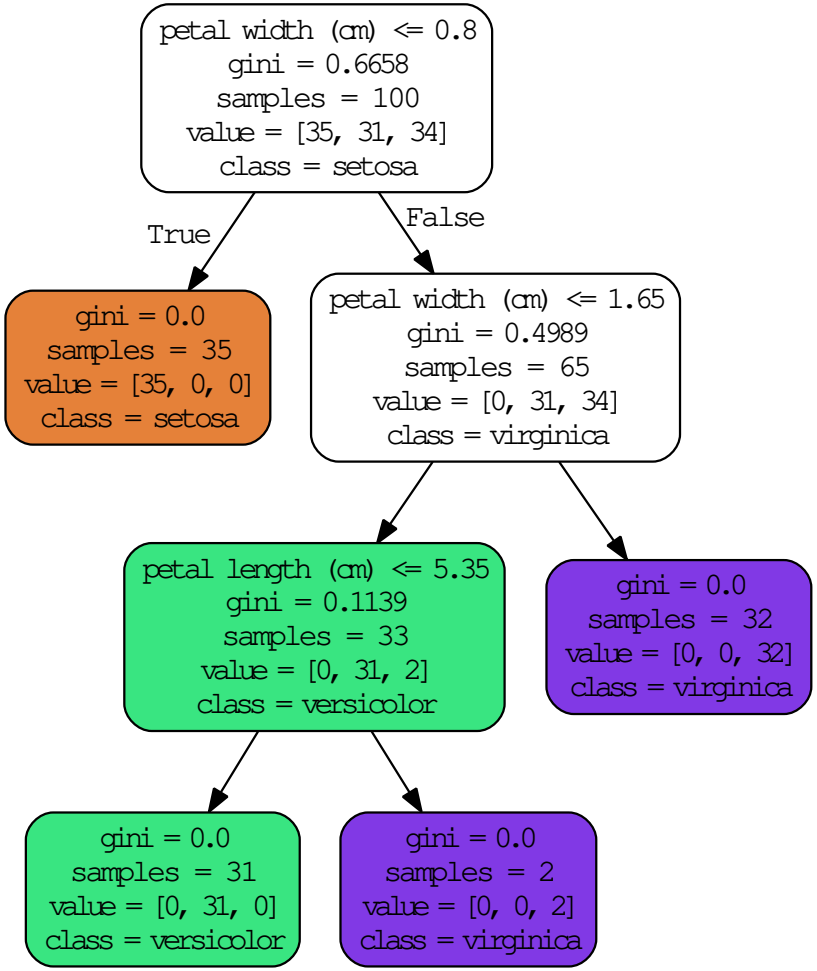
Árvores de Decisão

- A classe `DecisionTreeClassifier` implementa algoritmos para geração de Árvores de Decisão
- A escolha dos atributos pode ser feita via índice Gini (padrão) ou entropia
- Existem diversos parâmetros para gerar a árvore com restrições de poda: profundidade máxima, menor quantidade de instância para gerar separação de dados, etc
- Há gerador de gráfico dos modelos

Árvores de Decisão



Árvores de Decisão



Árvores de Decisão