Insper

Machine Learning

Decision Trees

Engenharia
Fábio Ayres <fabioja@insper.edu.br>

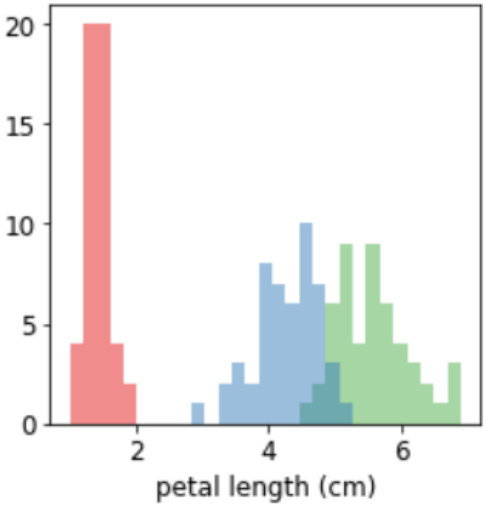
Iris

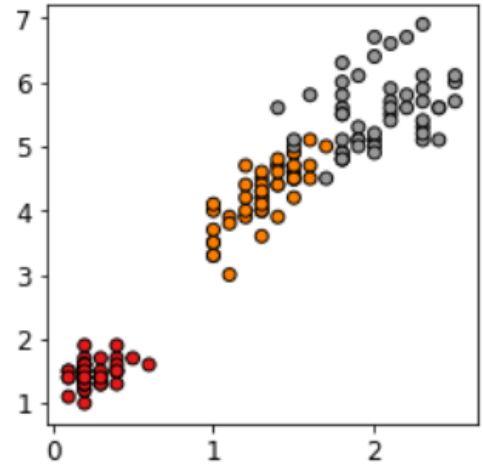
Iris Versicolor

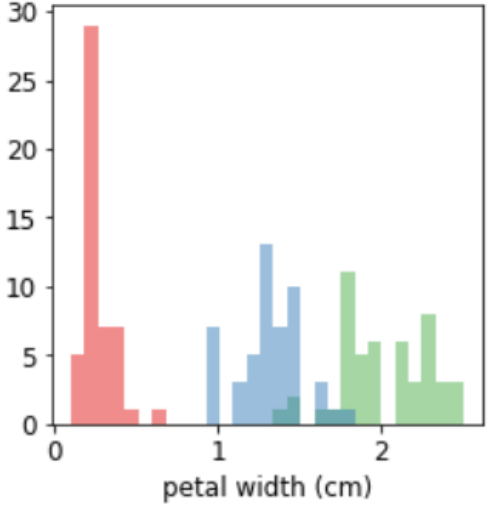
Iris Setosa

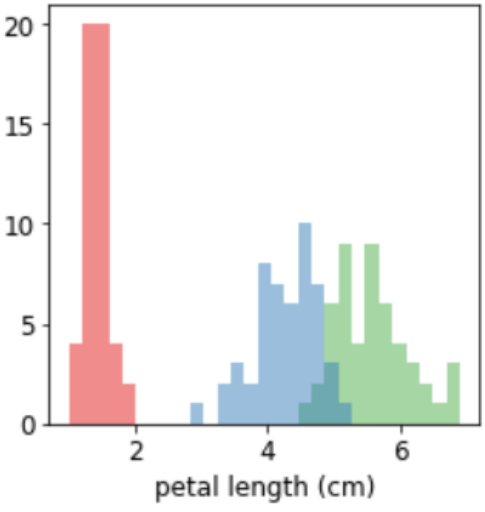
Iris Virginica

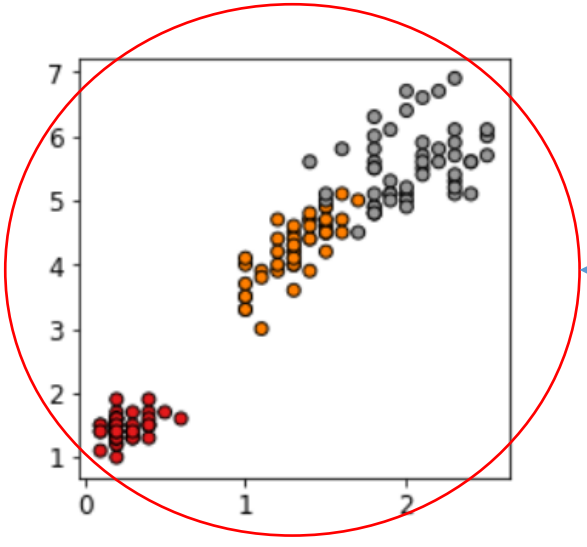
Fonte: https://www.datacamp.com/community/tutorials/machine-learning-in-r


20
15
10
5
0
2
4
6
petal length (cm)

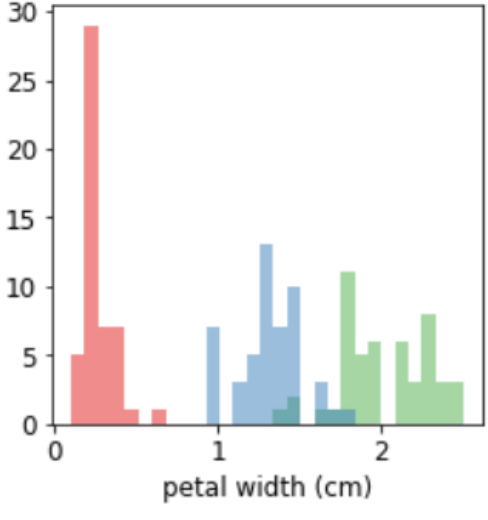

7
6
5
4
3
2
1
0
1
2

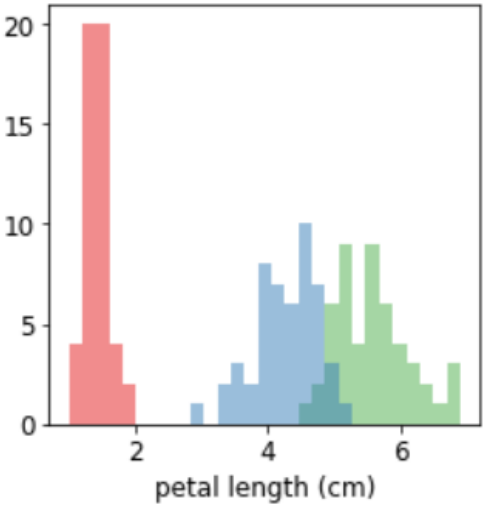

30
25
20
15
10
5
0
0
1
2
petal width (cm)

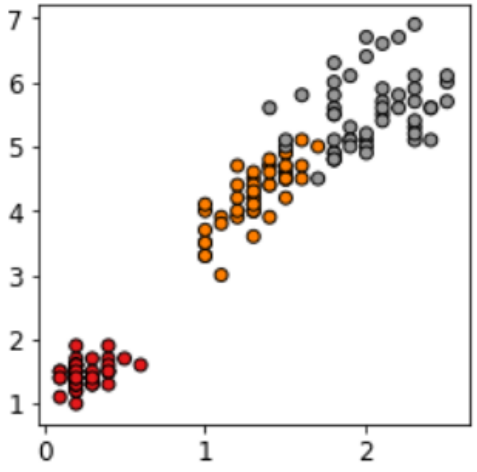




Tudo muito misturado!

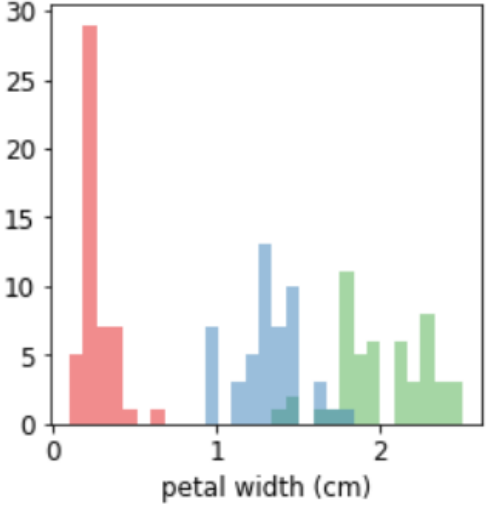






Ideia:

- escolhe uma feature
- escolhe um limiar
- separa em conjuntos mais homogêneos



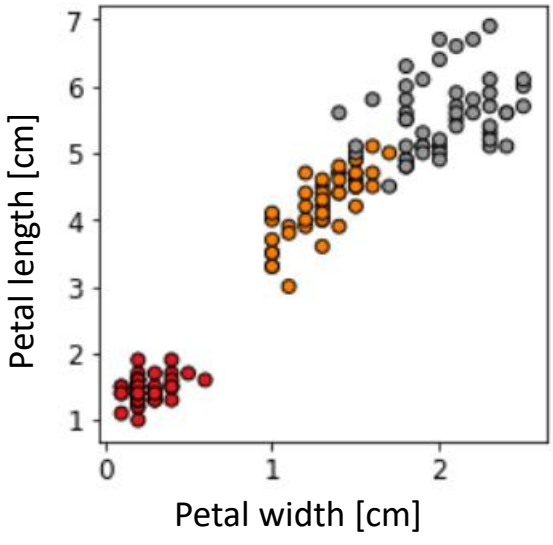
Medidas de “impureza”

- Queremos uma medida de impureza que seja
  - Zero para um conjunto completamente homogêneo
  - Se eu dobro (triplico, etc) o número de elementos em cada classe, a impureza é a mesma
    - Só a proporção de elementos por classe importa

Medidas de “impureza”

- Queremos uma medida de impureza que seja
  - Aumente para conjuntos mais misturados
    - Se o número de elementos por classe for o mesmo para todas as classes, a impureza é máxima para aquele número de classes
    - Quanto maior o número de classes existentes, maior a impureza máxima

Exemplo



Frequencias por classe:

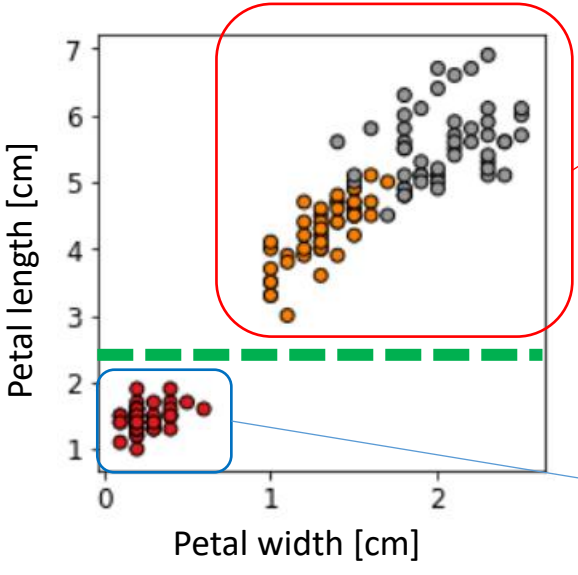
- 50 setosa
- 50 versicolor
- 50 virginica

Proporções por classe:

$$p = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]$$

Dividindo por limiar para uma feature escolhida

Mais puro!



Frequencias por classe:
- 0 setosa
- 50 versicolor
- 50 virginica

Proporções por classe:

$$p = \left[0, \frac{1}{2}, \frac{1}{2}\right]$$

Frequencias por classe:
- 50 setosa
- 0 versicolor
- 0 virginica

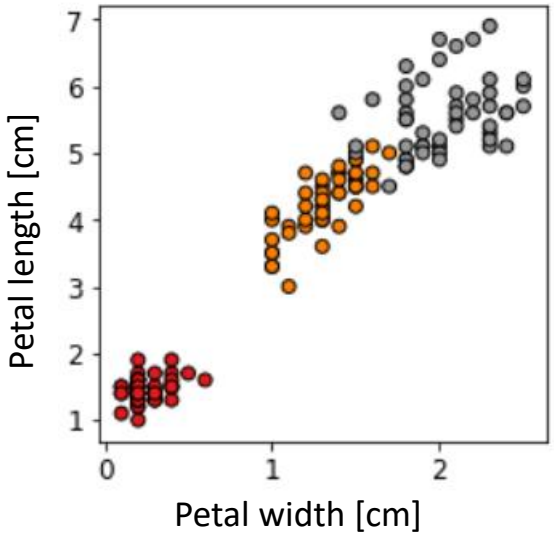
Proporções por classe:

$$p = [1,0,0]$$

Medidas de impureza comuns

Medida	Coeficiente Gini	Entropia
Definição	$G = 1 - \sum p_i^2$	$E = -\sum p_i \log_2 p_i$
Conjunto homogeneo $p_1 = 1$, no resto $p_i = 0$	$G = 1 - \left(1^2 + 0^2 + \cdots + 0^2\right) = 1 - 1 = 0$	$E = -\begin{pmatrix} 1 \times \log_2 1 \\ +0 \times \log_2 0 \\ + \cdots \\ +0 \times \log_2 0 \end{pmatrix} = 0$
Conjunto heterogeneo $p_i = 1/C$	$G = 1 - \left(\sum \left(\frac{1}{C}\right)^2\right) = 1 - C \times \frac{1}{C^2} = 1 - \frac{1}{C}$	$E = -\left(\sum \frac{1}{C} \times \log_2 \frac{1}{C}\right) = -C \times \frac{1}{C} \times \log_2 \frac{1}{C} = \log_2 C$

Exemplo



Frequencias por classe:

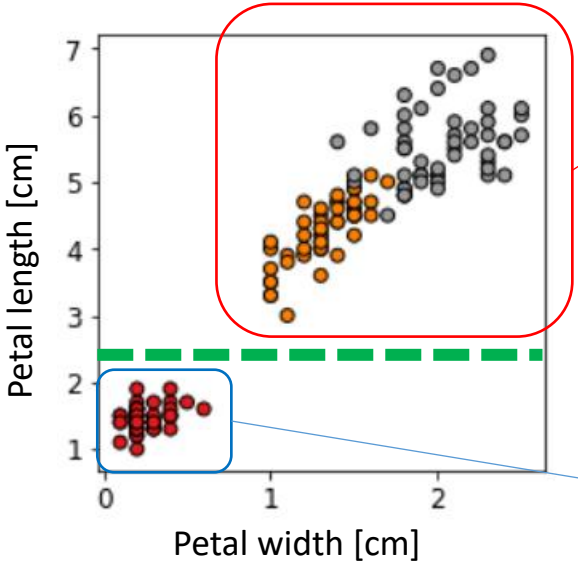
- 50 setosa
- 50 versicolor
- 50 virginica

Proporções por classe:

$$p = \left[\frac{1}{3}, \frac{1}{3}, \frac{1}{3}\right]$$

Dividindo por limiar para uma feature escolhida

Mais puro!



Frequencias por classe:
- 0 setosa
- 50 versicolor
- 50 virginica

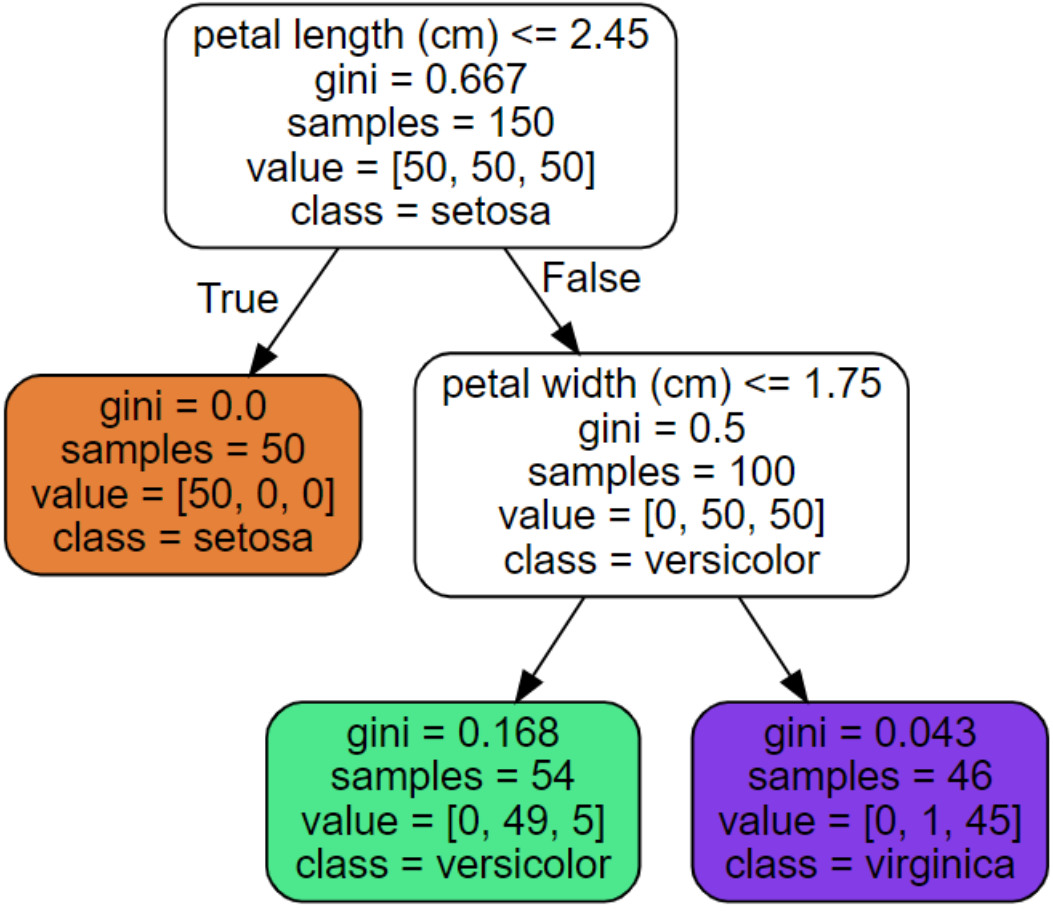
Proporções por classe:

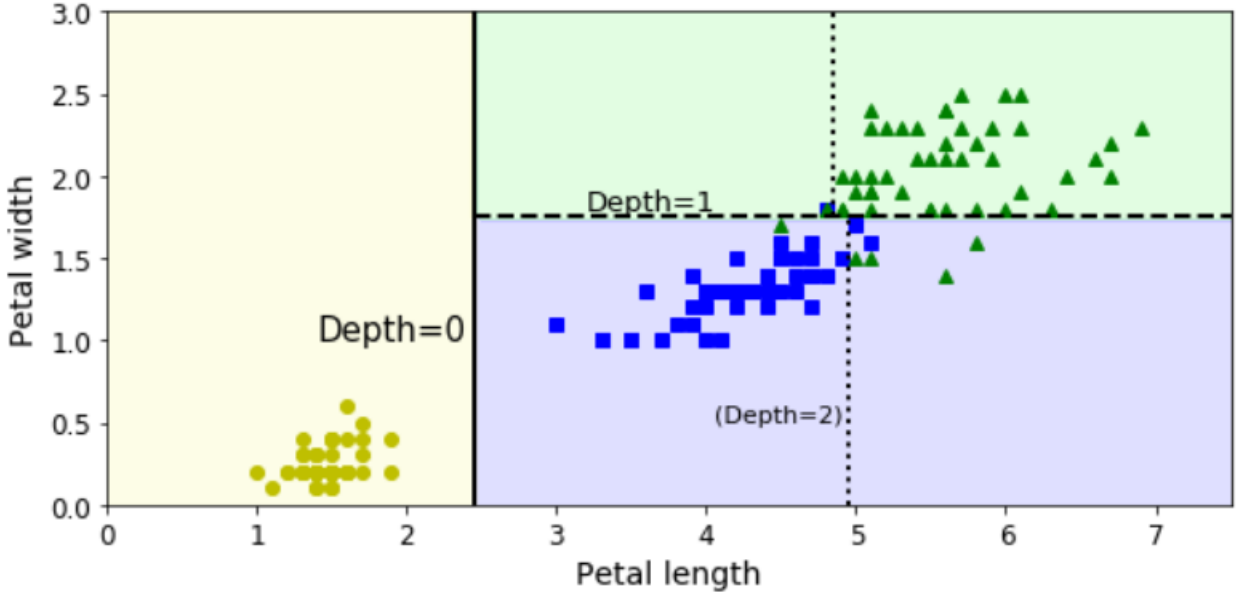
$$p = \left[0, \frac{1}{2}, \frac{1}{2}\right]$$

Frequencias por classe:
- 50 setosa
- 0 versicolor
- 0 virginica

Proporções por classe:

$$p = [1,0,0]$$

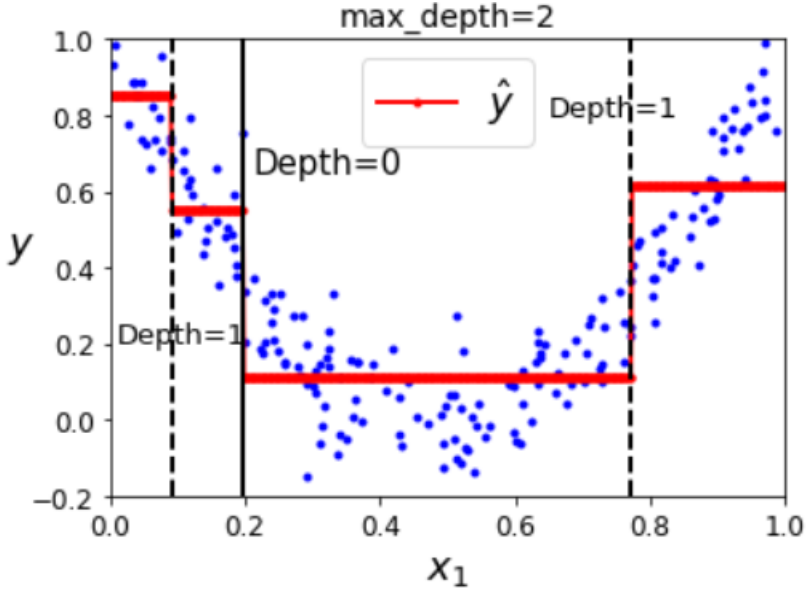

petal length (cm) <= 2.45
gini = 0.667
samples = 150
value = [50, 50, 50]
class = setosa
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
class = setosa
petal width (cm) <= 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
class = versicolor
gini = 0.168
samples = 54
value = [0, 49, 5]
class = versicolor
gini = 0.043
samples = 46
value = [0, 1, 45]
class = virginica


Petal width
Petal length
Depth=0
Depth=1
(Depth=2)
3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
1
2
3
4
5
6
7

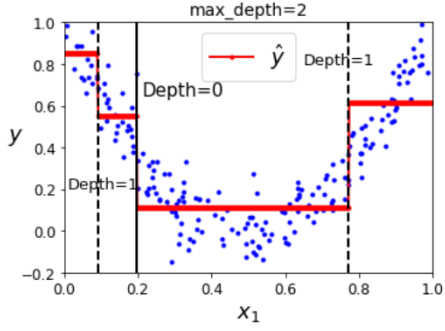
Algoritmo CART

CART: Classification and Regression Trees

- Sim, regressão também! Basta trocar a medida de impureza!



CART para regressão



Algoritmo CART: treinamento

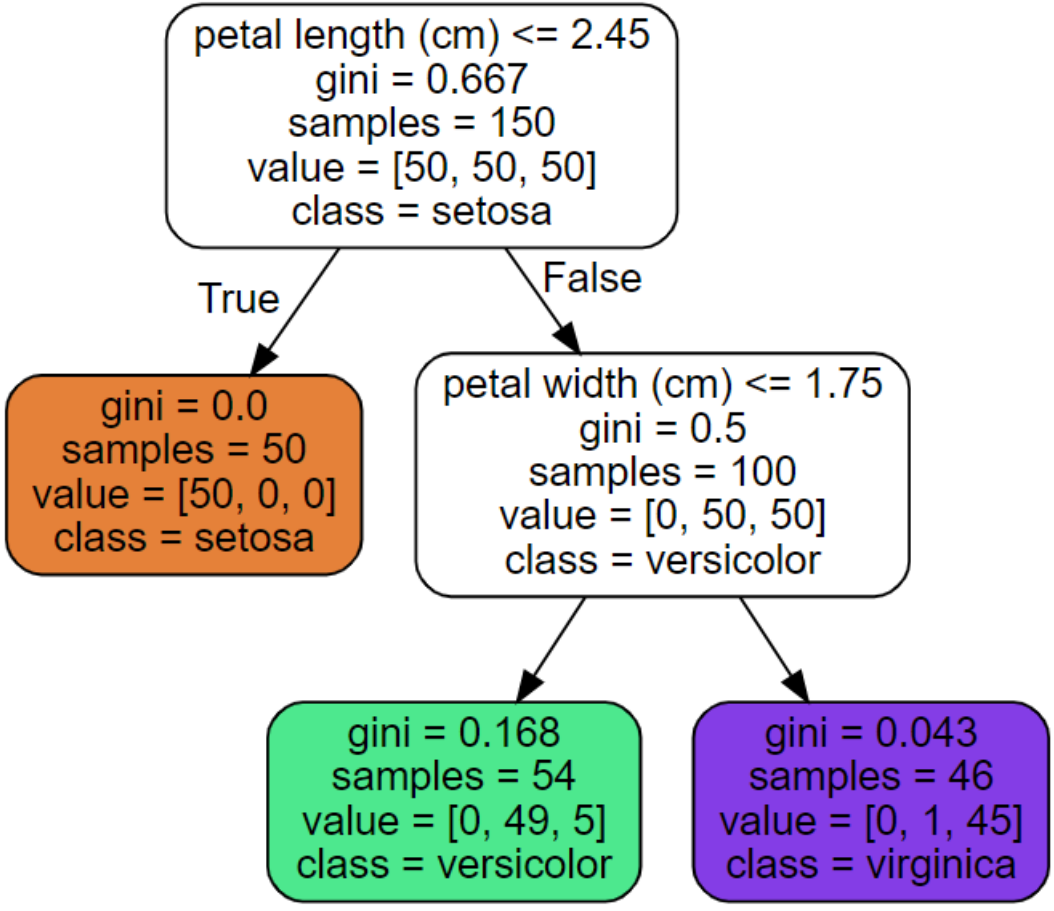
- Testa todas as features e todos os thresholds
  - Basta testar os thresholds correspondentes aos valores das amostras
- Para cada combinação feature e threshold, avaliar a função de custo do CART:

$$J = \frac{m_{left}}{m} G_{left} + \frac{m_{right}}{m} G_{right}$$

O que acontece com J se o conjunto já for puro?

Algoritmo CART: treinamento

- Se a melhor combinação (feature, threshold) efetivamente melhora a função de custo, dividir o conjunto de pontos de treinamento.
- Repetir recursivamente o algoritmo para cada partição


petal length (cm) <= 2.45
gini = 0.667
samples = 150
value = [50, 50, 50]
class = setosa
True
False
gini = 0.0
samples = 50
value = [50, 0, 0]
class = setosa
petal width (cm) <= 1.75
gini = 0.5
samples = 100
value = [0, 50, 50]
class = versicolor
gini = 0.168
samples = 54
value = [0, 49, 5]
class = versicolor
gini = 0.043
samples = 46
value = [0, 1, 45]
class = virginica

Algoritmo CART: predição

Para uma nova amostra:

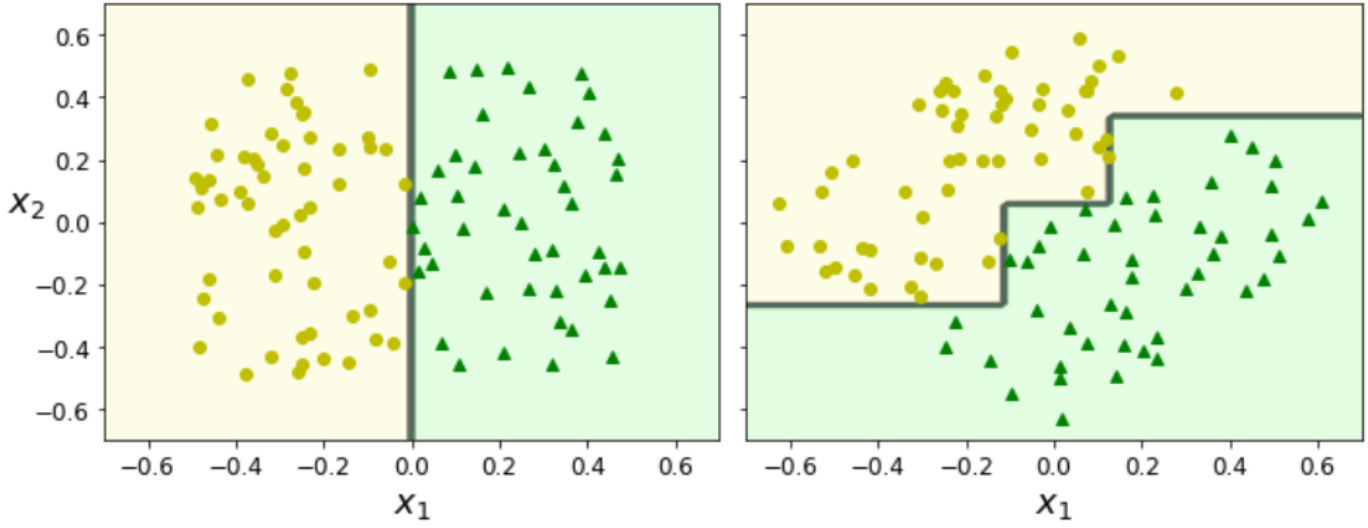
- Percorre a árvore até chegar na folha
- Retorna o valor da decisão na folha
  - Classificação: retorna a classe mais proeminente
  - Regressão: retorna o valor médio das amostras da folha

Vantagens da árvore de decisão

- Não precisa de *scaling* como a SVM
- Fácil de implementar
- Paralelizável
- **INTERPRETÁVEL**
  - Features mais importantes aparecem mais cedo na árvore!
  - Podemos saber a incerteza da predição olhando a impureza do nó de decisão

Desvantagens

- Preferência por fronteiras de decisão ortogonais e alinhadas com os eixos cartesianos
- Não é invariante à rotação



Insper