Decision Trees

Chapter 6: pp 175 – 187

- Simple & easy to understand but very effective
- Example:
  - Classify a flower if the petal length < 2.45 cm
  - Classify a flower if the petal length > 2.45 and the petal width < 1.75cm

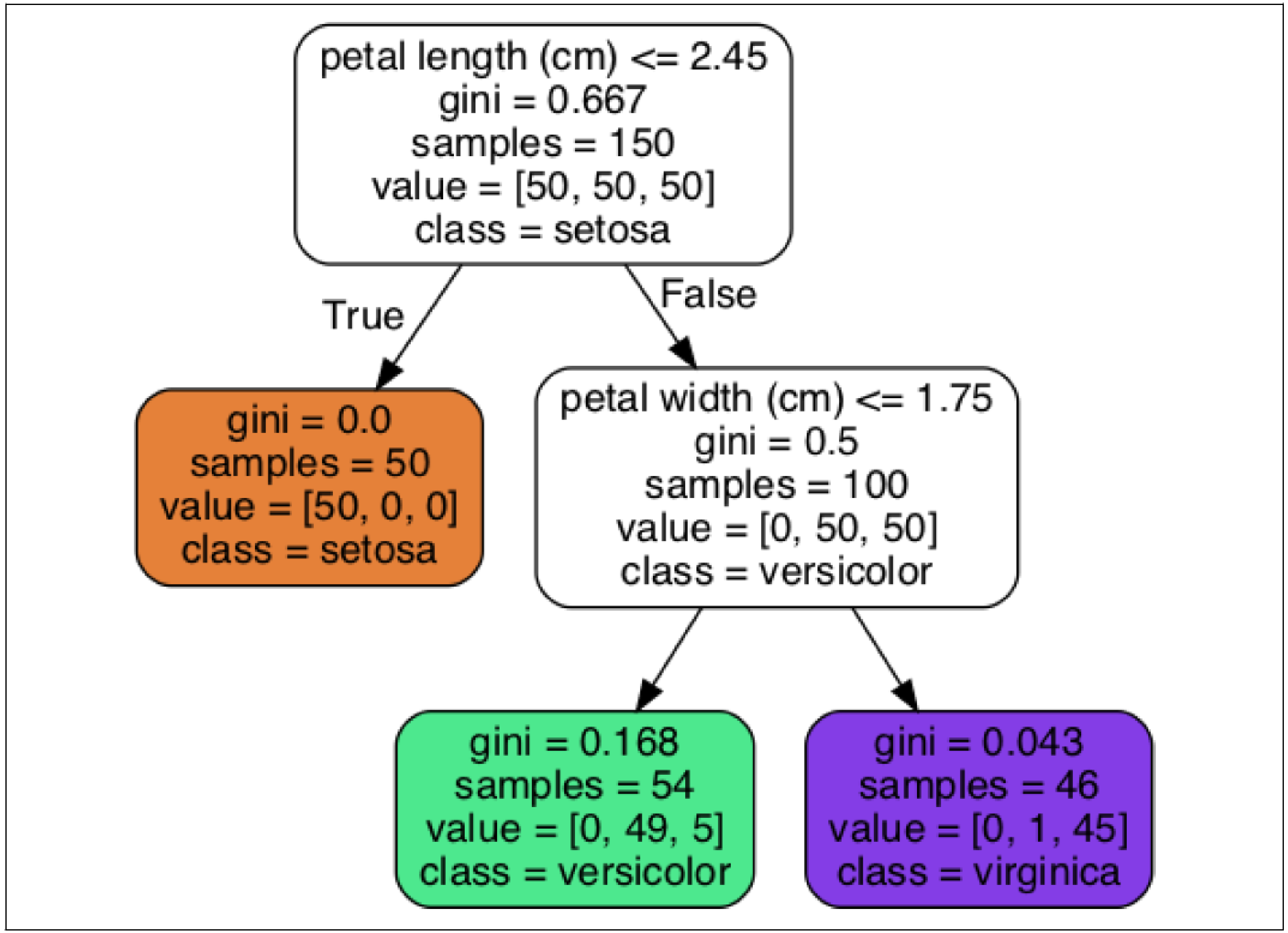


Figure 6-1. Iris Decision Tree

Gini Impurity Score

Equation 6-1. Gini impurity

$$G_i = 1 - \sum_{k=1}^{n} {p_{i,k}}^2$$

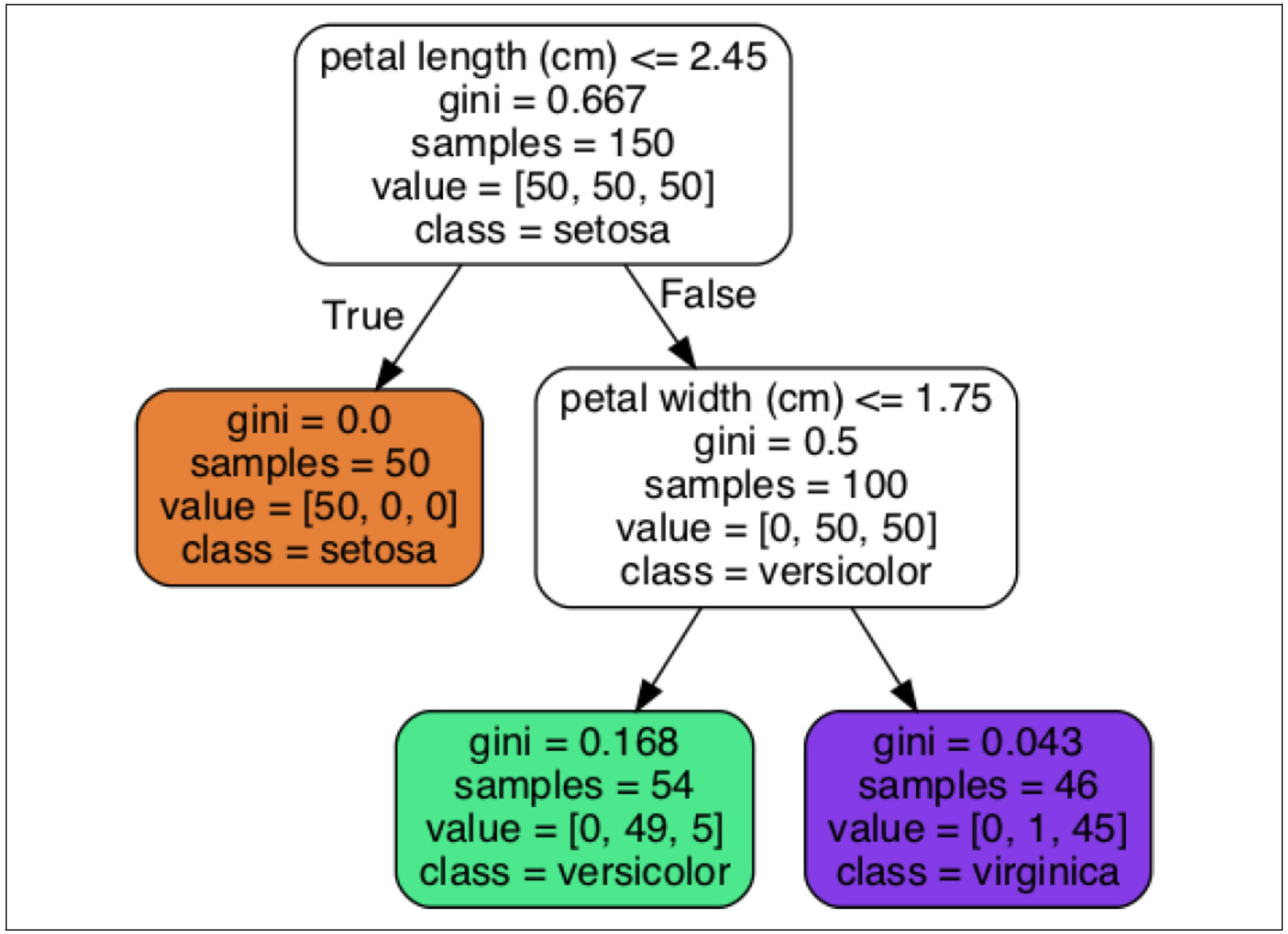
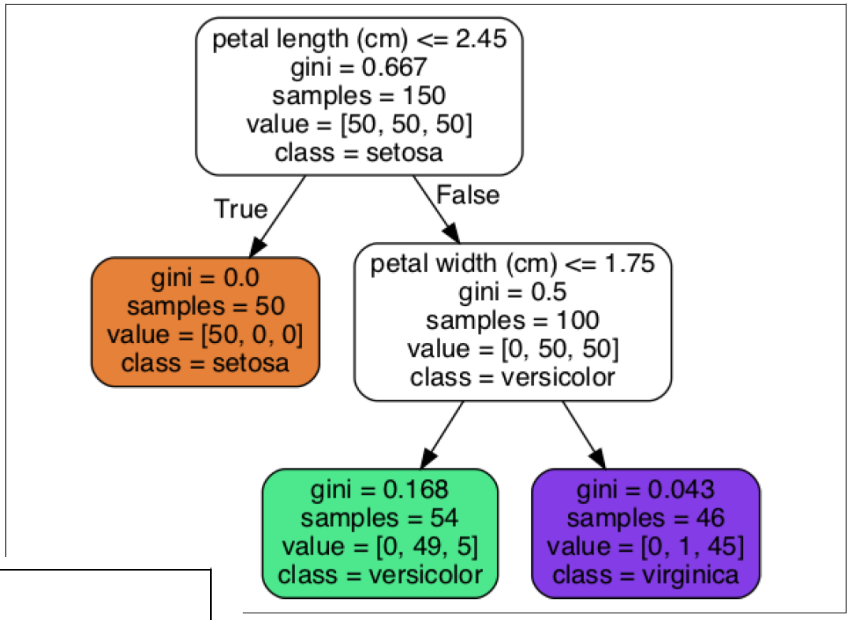


Figure 6-1. Iris Decision Tree

$p_{i,k}$ is the ratio of class k instances among the training instances in the i^{th} node.

Decision Boundaries

- Decision Trees divide the feature space into axis parallel (hyper)-rectangles
- Each rectangular region is labeled with one label



Tree

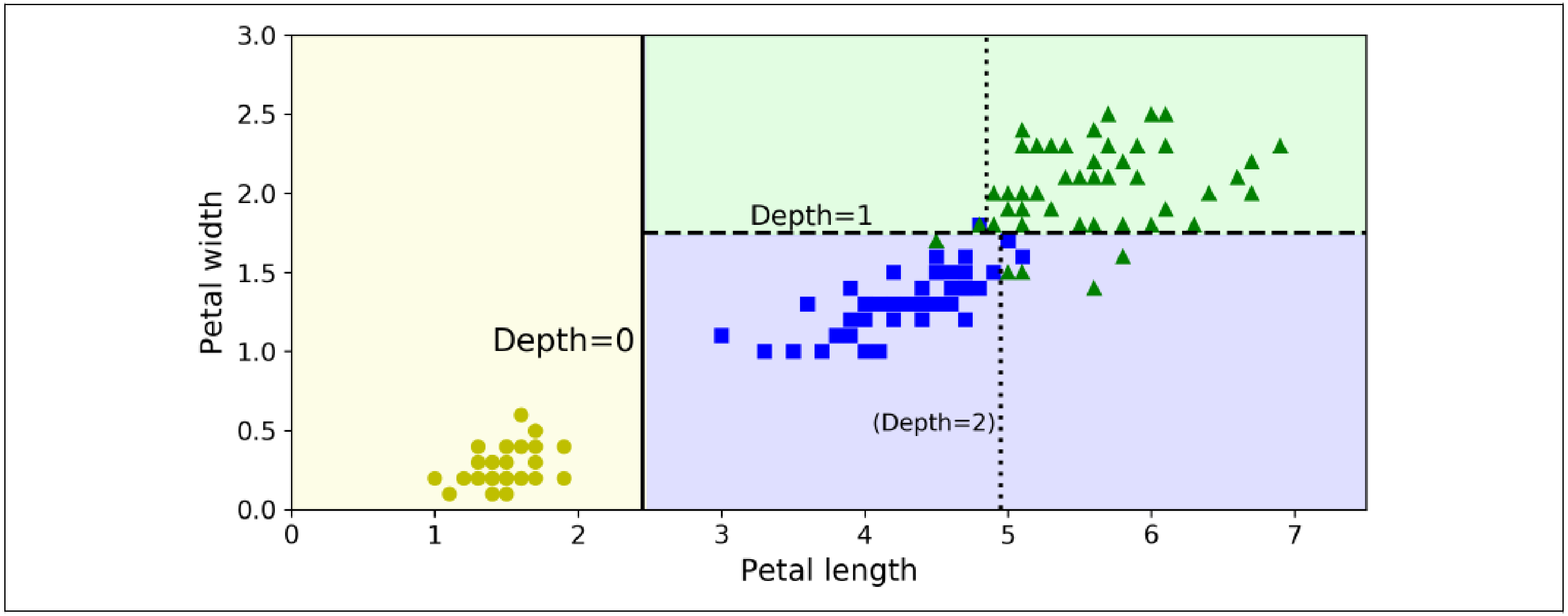
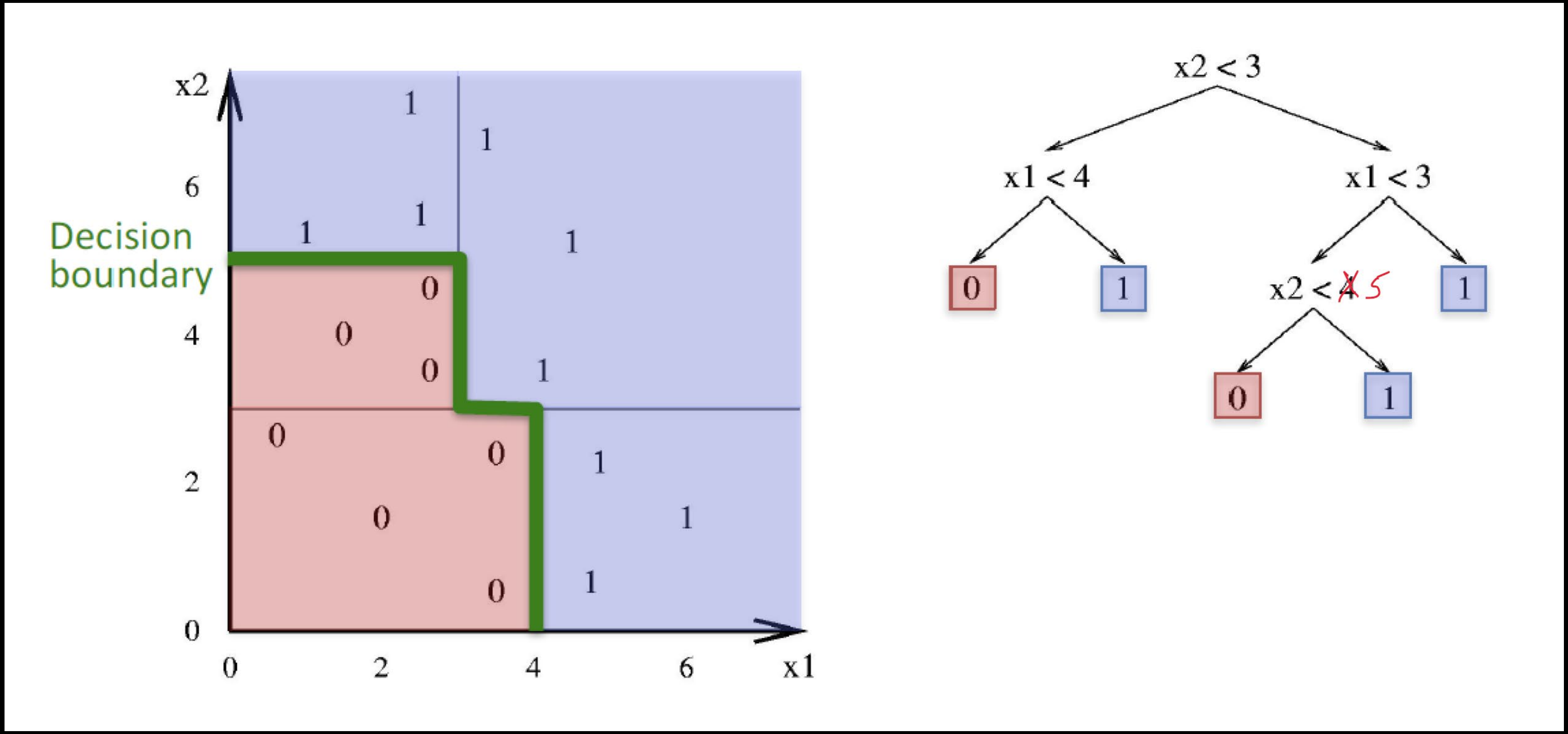


Figure 6-2. Decision Tree decision boundaries

Another Example (depth=2)



Estimating Class Probabilities

- Class probabilities for a flower whose petal length and width are 4 cm and 0.5 cm, respectively
  - Setosa = 0/54 = 0%
  - Versicolor = 49/54 = 90.7%
  - Virginica = 5/54 = 9.3%

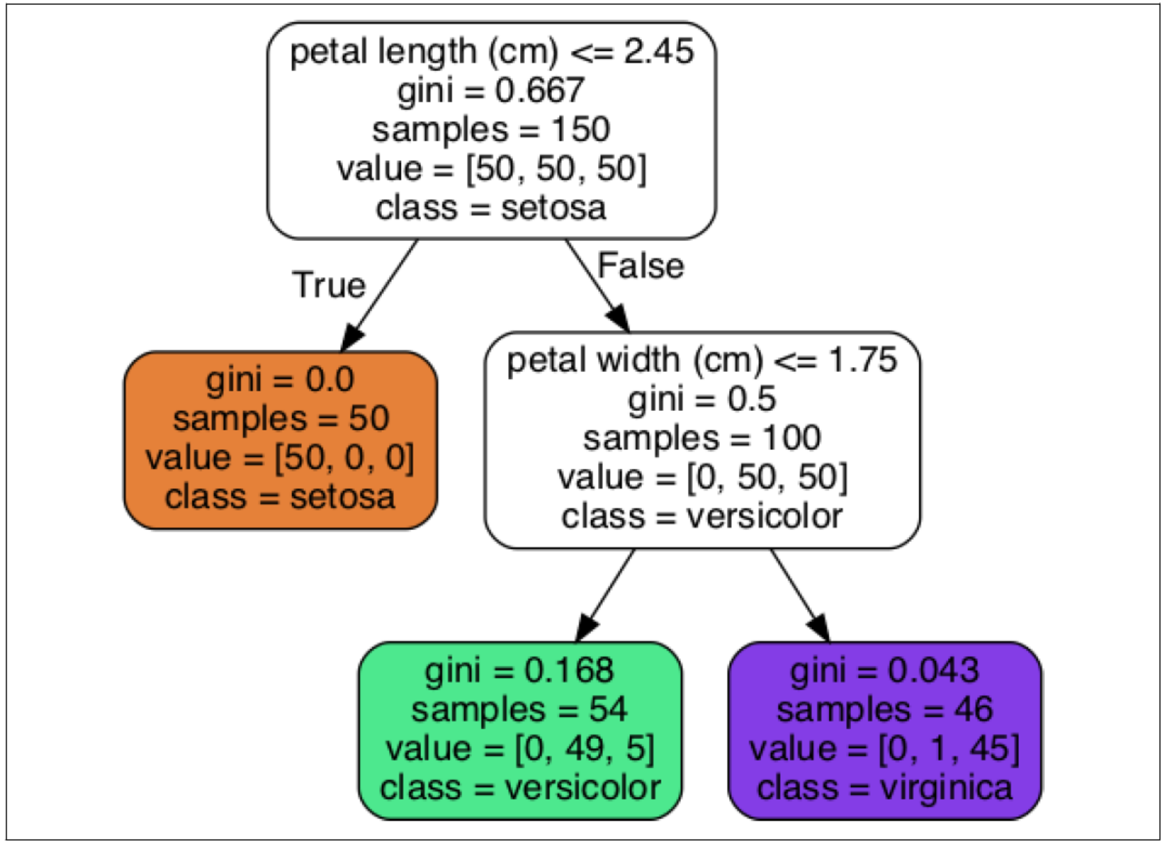


Figure 6-1. Iris Decision Tree

- Petal length = 4 cm, petal width = 0.5 cm → 90.7% Versicolor
- Petal length = 6 cm, petal width = 1.5 cm → still 90.7% Versicolor !

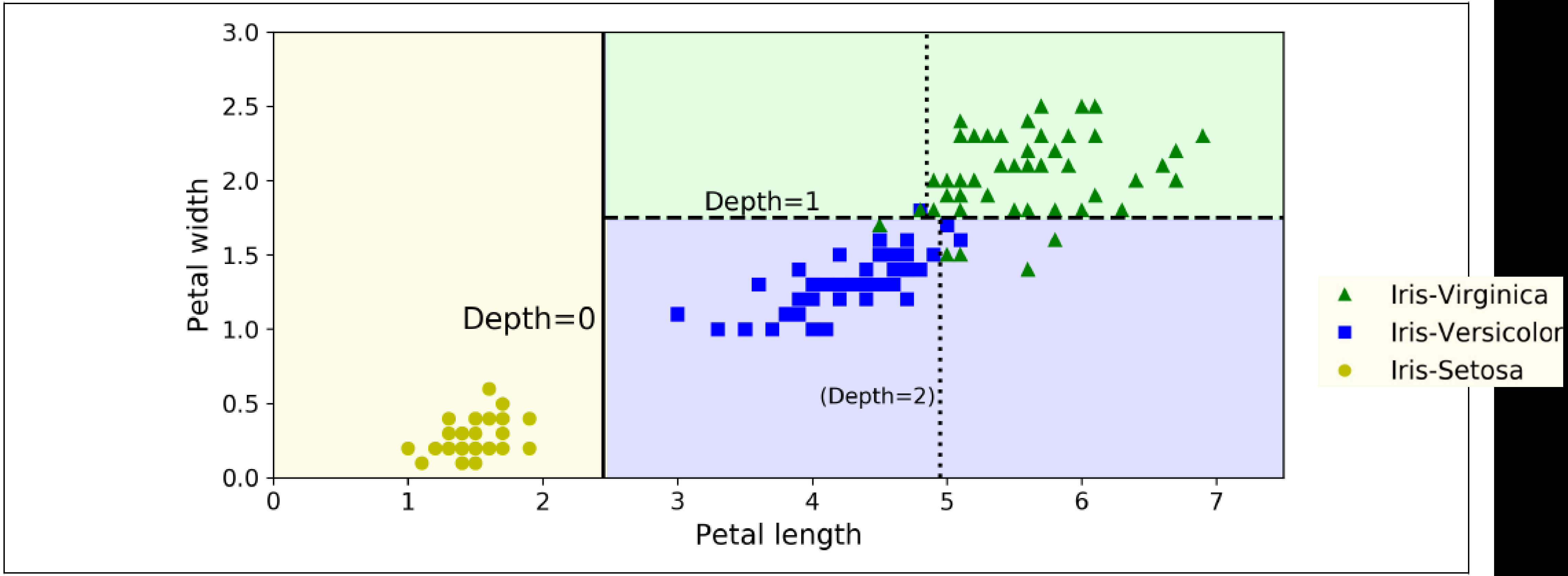


Figure 6-2. Decision Tree decision boundaries

How To Train Decision Trees

- CART (Classification And Regression Tree) algorithm
- Recursively splits the training data into 2 subsets based on feature k and threshold t_k
  - Chosen to produce the purest subsets (minimize Eq 6-2)

Equation 6-2. CART cost function for classification

$$J(k, t_k) = \frac{m_{\text{left}}}{m} G_{\text{left}} + \frac{m_{\text{right}}}{m} G_{\text{right}}$$

$$\text{where} \begin{cases} G_{\text{left/right}} \text{ measures the impurity of the left/right subset,} \\ m_{\text{left/right}} \text{ is the number of instances in the left/right subset.} \end{cases}$$

  - Stopping conditions: maximum depth, maximum leaf nodes, purest nodes etc

Computational Complexity

- Prediction
  - Only requires checking one feature → quite fast
  - Complexity: $O(log_2(m)\ )$
- Training
  - Compares all features on all samples at each node
  - Complexity: $O(n \times m\ log_2(m)\ )$

Entropy

- Another impurity measure is the *entropy*
  - A set has an entropy of 0 when it contains instances of only one class

Equation 6-3. Entropy

$$H_i = -\sum_{\substack{k=1 \\ p_{i,k} \neq 0}}^{n} p_{i,k} \log_2 \left(p_{i,k}\right)$$

- $p_{i,k}$ is the ratio of class k instances among the training instances in the i^{th} node.

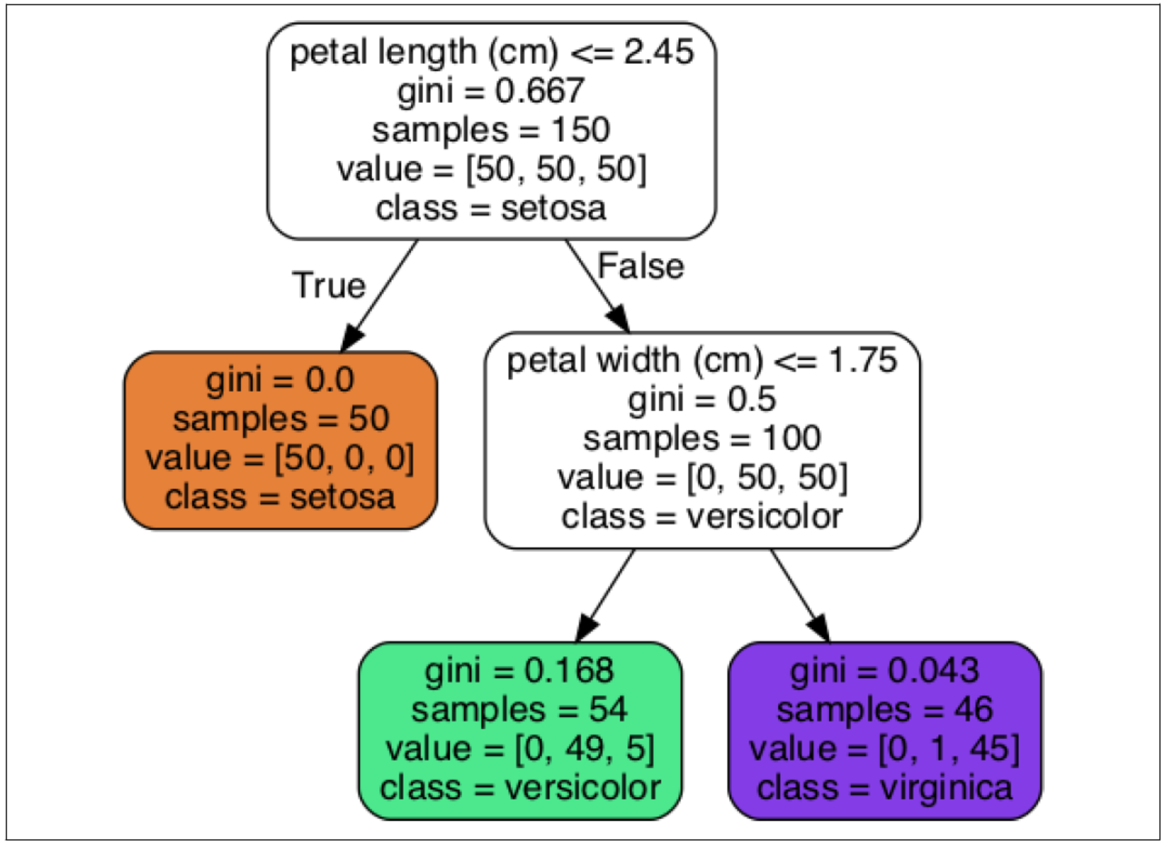
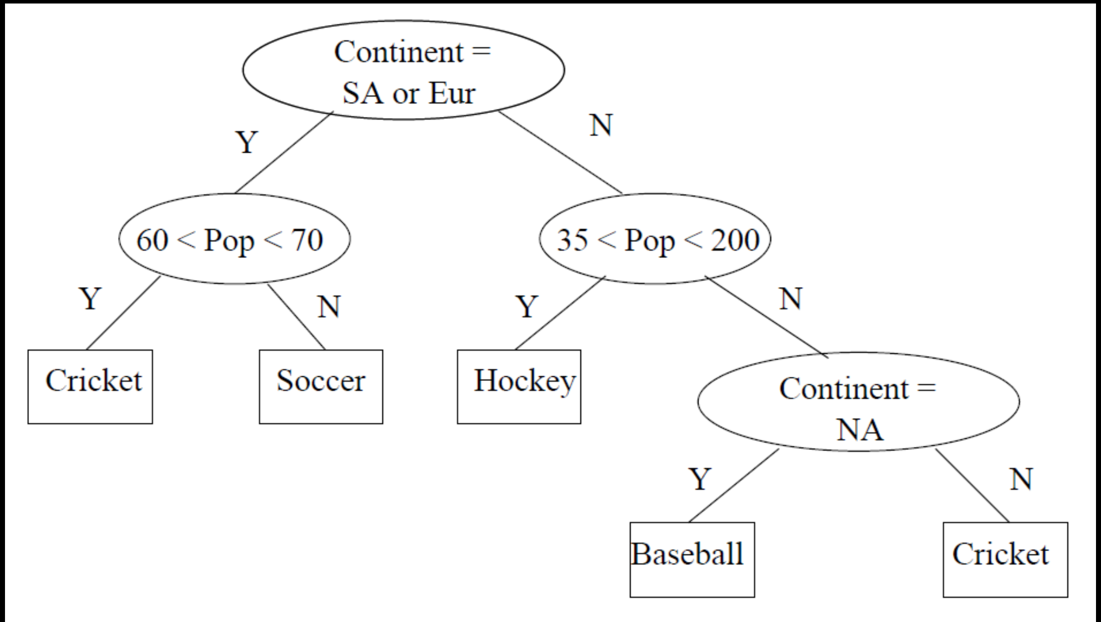


Figure 6-1. Iris Decision Tree

Example: country favorite sports

Country	Continent	Population	Sport
Argentina	SA	44	Soccer
Australia	Aus	34	Cricket
Brazil	SA	211	Soccer
Canada	NA	36	Hockey
Cuba	NA	11	Baseball
Germany	Eur	80	Soccer
India	Asia	1342	Cricket
Italy	Eur	59	Soccer
Russia	Asia	143	Hockey
Spain	Eur	46	Soccer
United Kingdom	Eur	65	Cricket
United States	NA	326	Baseball



- Left child:
  - GINI index = 0.278
  - Entropy = 0.650

- Mining Massive Datasets by J. Leskovec, A. Rajaraman, J. Ullman

Regularization Hyper-parameters

- Decision Trees model is non-parametric (Logistic Regression is parametric)
- Regularization involve restricting the shape of the tree
  - Maximum tree depth
  - Maximum number of leaf nodes
  - Maximum number of features evaluated for splitting at a node
  - Minimum number of samples a node must have before it can split
  - Minimum number of samples of a leaf node

- Moons dataset

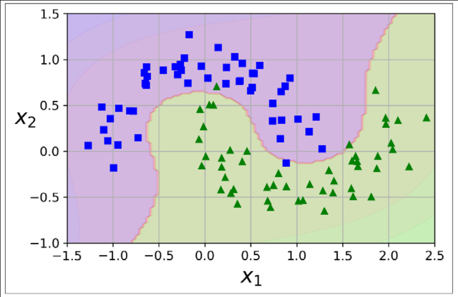


Figure 5-6. Linear SVM classifier using polynomial features

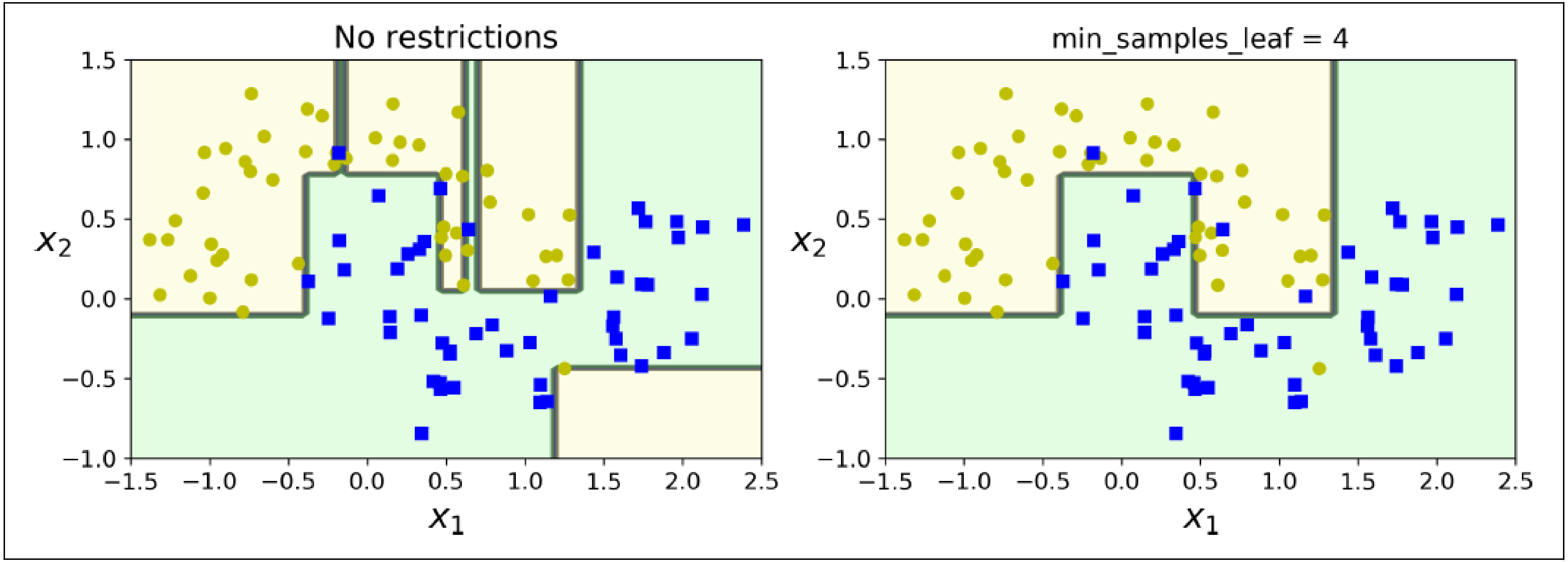


Figure 6-3. Regularization using min_samples_leaf

Regression

- Example:
- Predict the value of y for $x_1 = 0.6 \rightarrow 0.111$

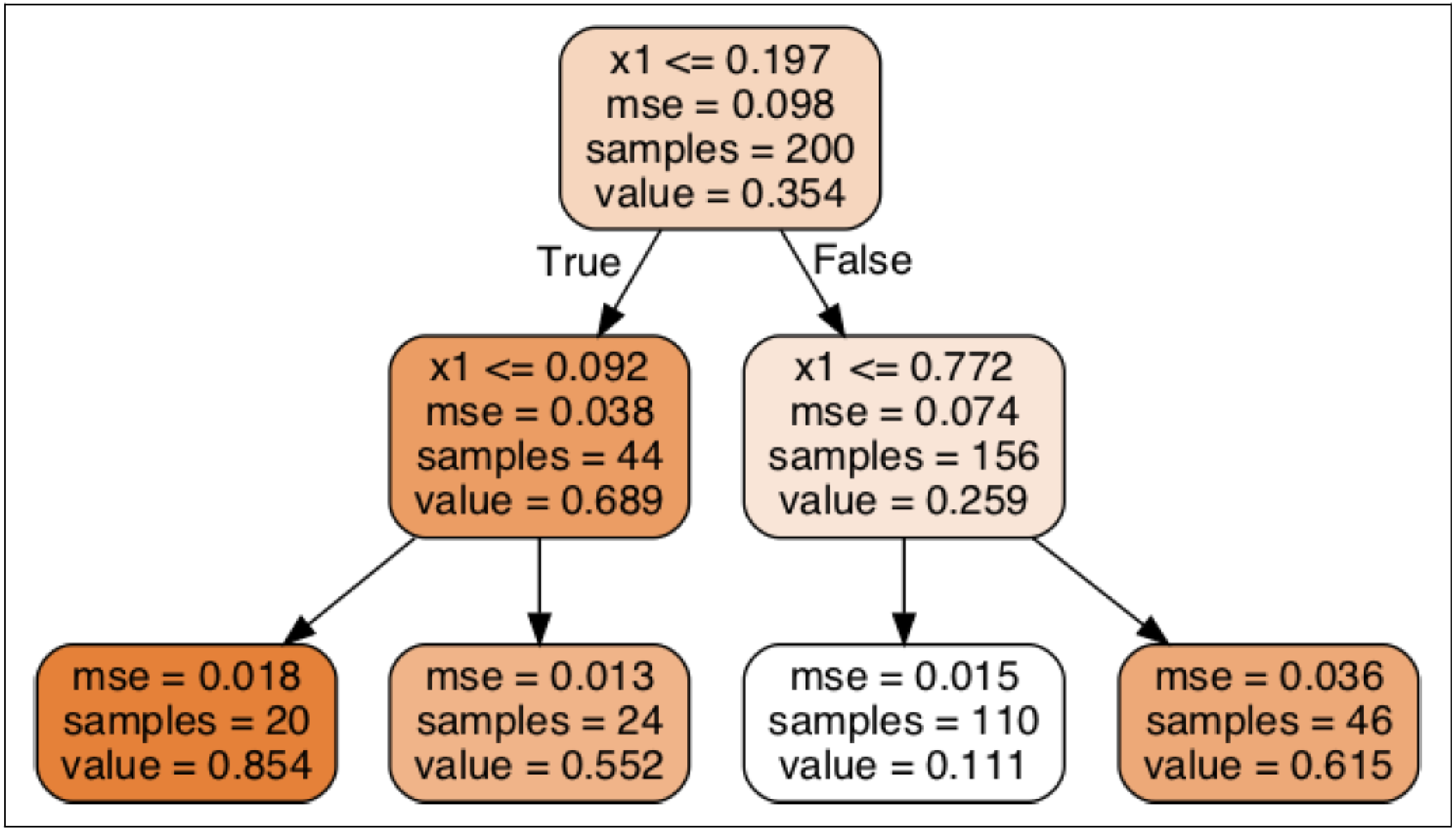


Figure 6-4. A Decision Tree for regression

- Predict the value of y for $x_1 = 0.04$ if max_depth = 2 → 0.85
- Predict the value of y for $x_1 = 0.04$ if max_depth = 3 → 0.95

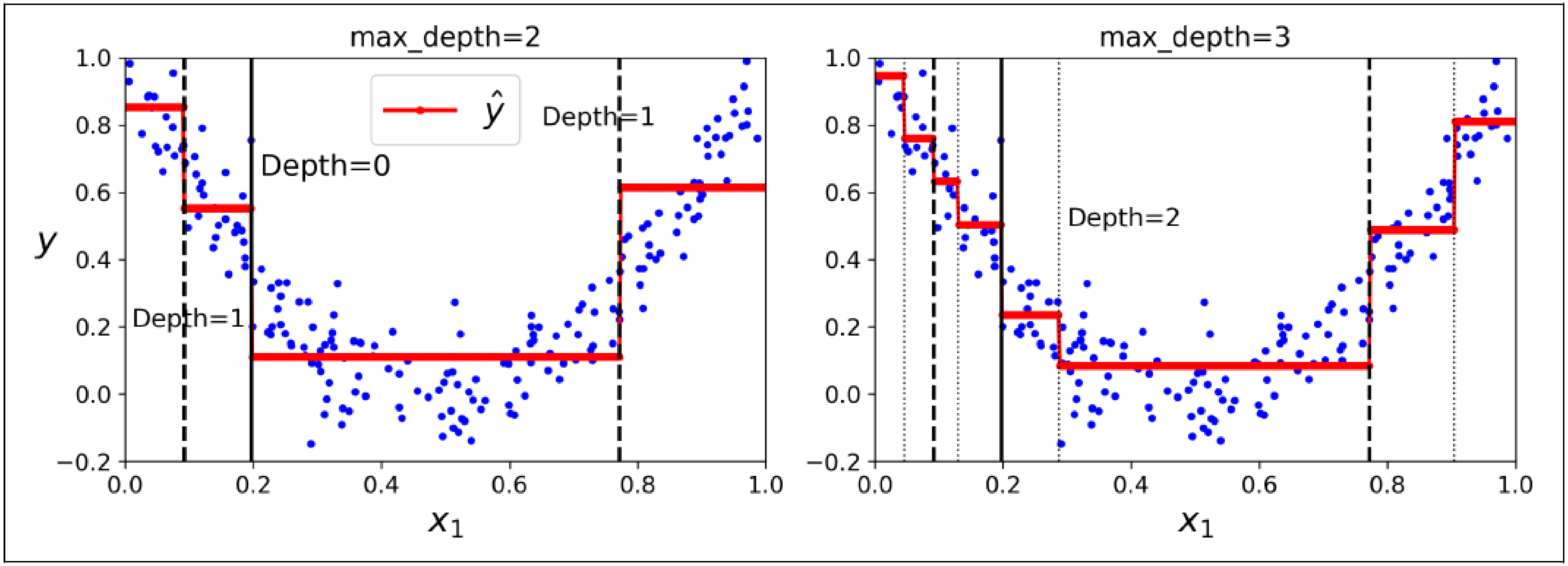


Figure 6-5. Predictions of two Decision Tree regression models

- Training: split the training set to minimize the MSE

Equation 6-4. CART cost function for regression

$$J(k, t_k) = \frac{m_{\text{left}}}{m}\text{MSE}_{\text{left}} + \frac{m_{\text{right}}}{m}\text{MSE}_{\text{right}} \quad \text{where} \begin{cases} \text{MSE}_{\text{node}} = \sum_{i \in \text{node}} \left(\hat{y}_{\text{node}} - y^{(i)}\right)^2 \\ \hat{y}_{\text{node}} = \frac{1}{m_{\text{node}}} \sum_{i \in \text{node}} y^{(i)} \end{cases}$$

- Apply regularization to minimize overfitting

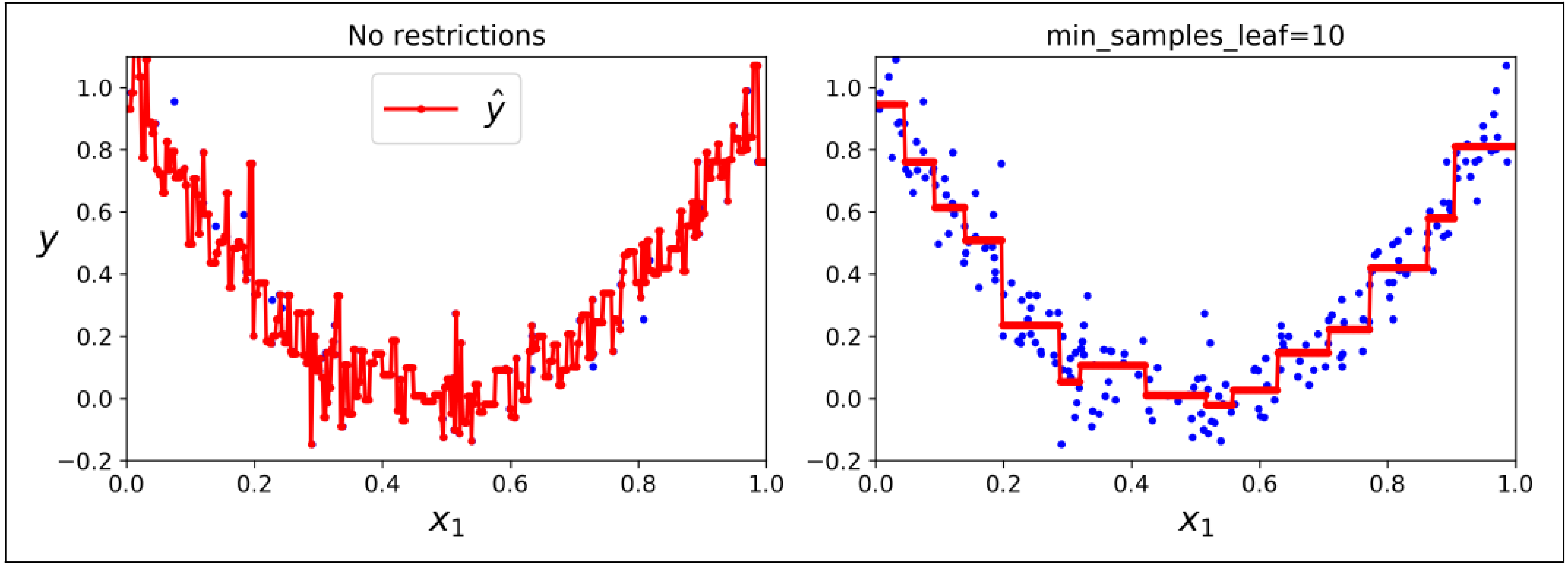


Figure 6-6. Regularizing a Decision Tree regressor

Instability

- Pros: simple to understand, easy to use, versatile, powerful
- Cons: very sensitive to data orthogonality (rotation)

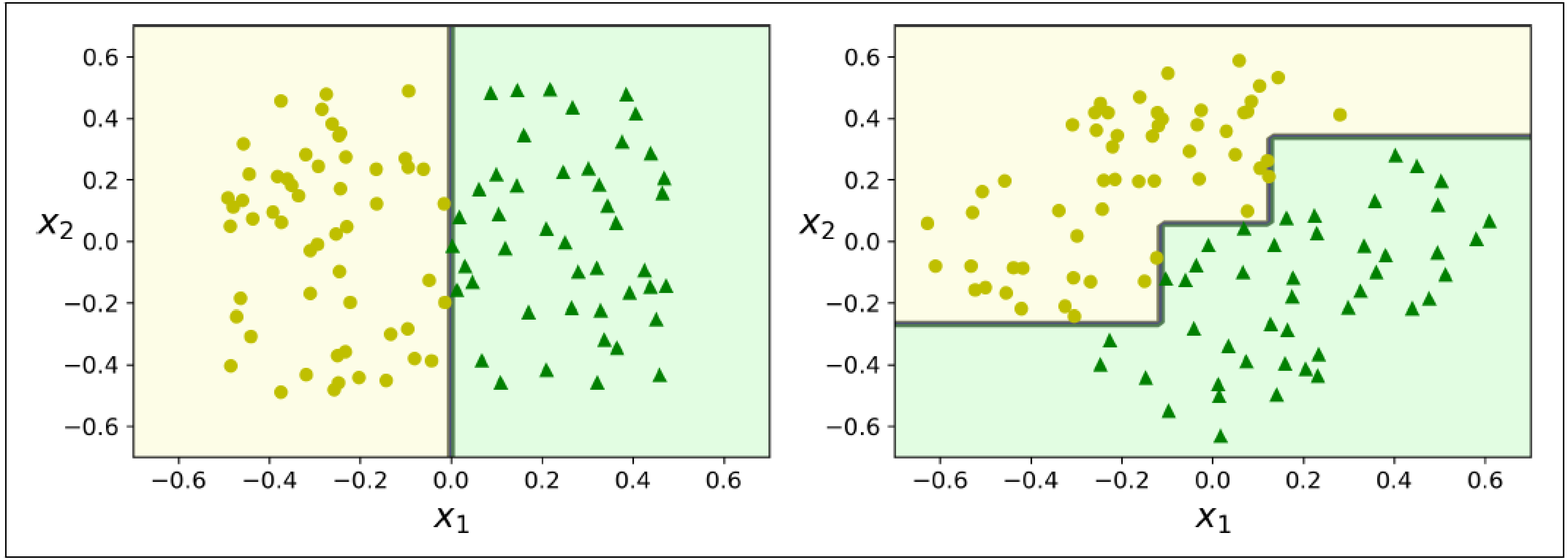


Figure 6-7. Sensitivity to training set rotation

Instability

- Cons: very sensitive to small variations in the training data
  - Example: remove the widest Versicolor example (petals 4.8 cm long, 1.8 cm wide)
  - → Decision Tree (Fig. 6-8) looks very different than Fig. 6-2

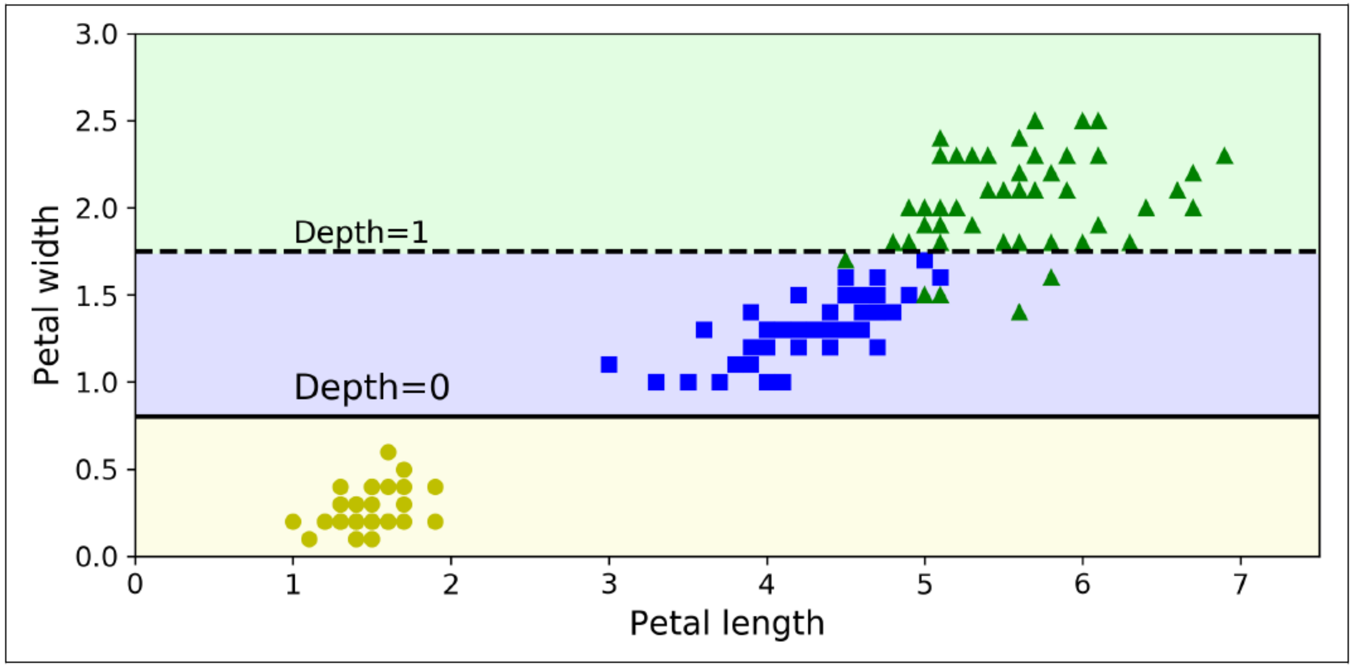


Figure 6-8. Sensitivity to training set details