Decision Trees

CSCI-P556 Applied Machine Learning
Lecture 20

D.S. Williamson

Agenda and Learning Outcomes

Today's Topics

- **Topics**:
  - Decision Trees
- **Announcements**
  - Pairings for Homework

Decisions

A Restaurant Example

- Suppose you are deciding whether you will (or won't) wait for a table at a restaurant.
- **Different features/attributes contribute to this decision**:
    - Alternative restaurant options available? Is it a bar? Is it Friday?
    - Are you hungry? How busy is the restaurant? Price?
    - ….

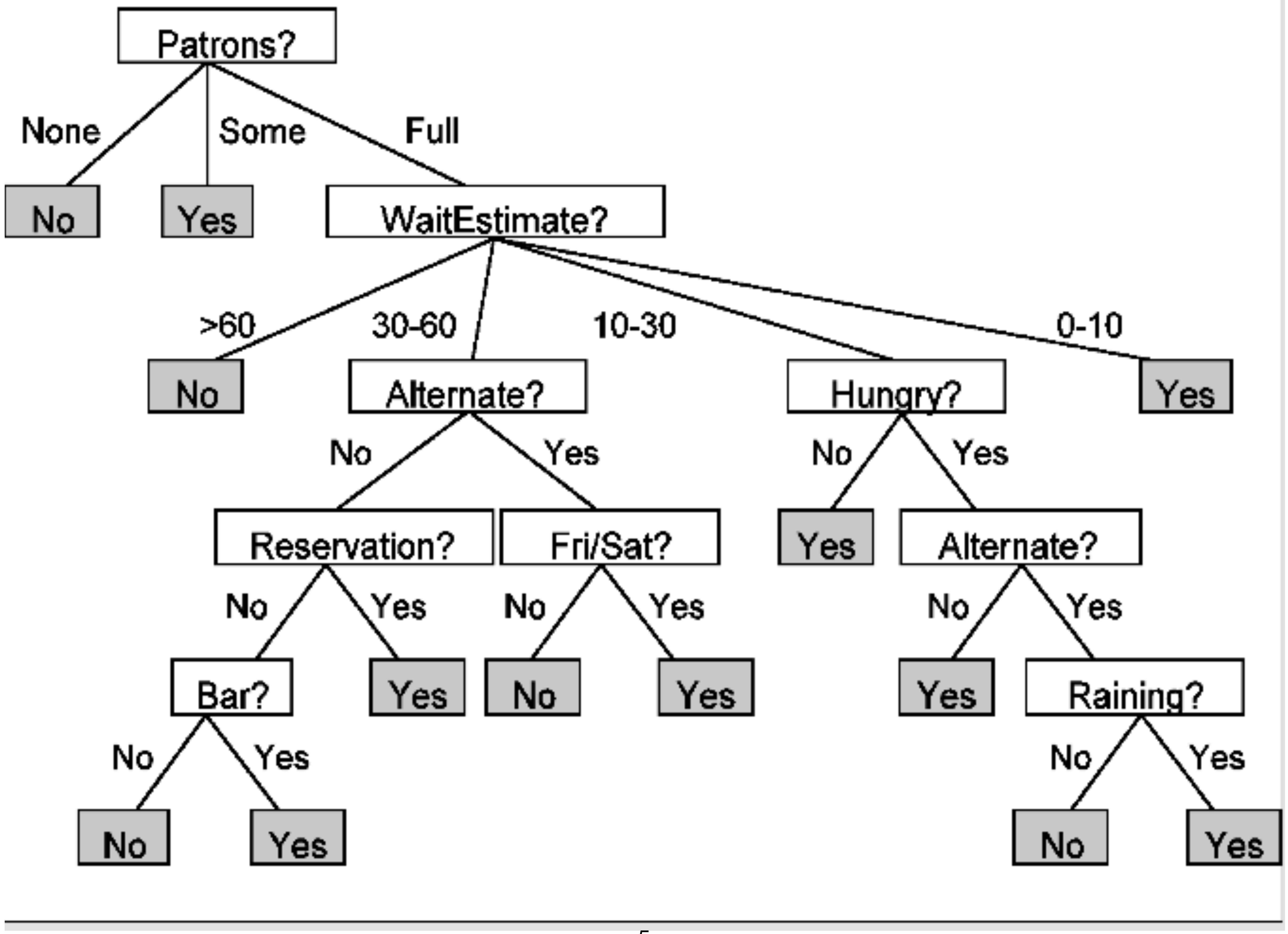
Decisions

A Restaurant Example

Ex.	Features/Attributes										Target
	Alt.	Bar	Friday	Hungry	Patrons	Price	Rain	Reservati	Type	Wait	Will Wait
X1	T	F	F	T	Some	$$$	F	T	French	0-10	T
X2	T	F	F	T	Full	$	F	F	Thai	30-60	F
X3	F	T	F	F	Some	$	F	F	Burger	0-10	T
X4	T	F	T	T	Full	$	F	F	Thai	10-30	T
X5	T	F	T	F	Full	$$$	F	T	French	>60	F
X6	F	T	F	T	Some	$$	T	T	Italian	0-10	T
X7	F	T	F	F	None	$	T	F	Burger	0-10	F
X8	F	F	F	T	Some	$$	T	T	Thai	0-10	T
X9	F	T	T	F	Full	$	T	F	Burger	>60	F
X10	T	T	T	T	Full	$$$	F	T	Italian	10-30	F
X11	F	F	F	F	None	$	F	F	Thai	0-10	F
X12	T	T	T	T	Full	$	F	F	Burger	30-60	T

Decision Tree

A Restaurant Example



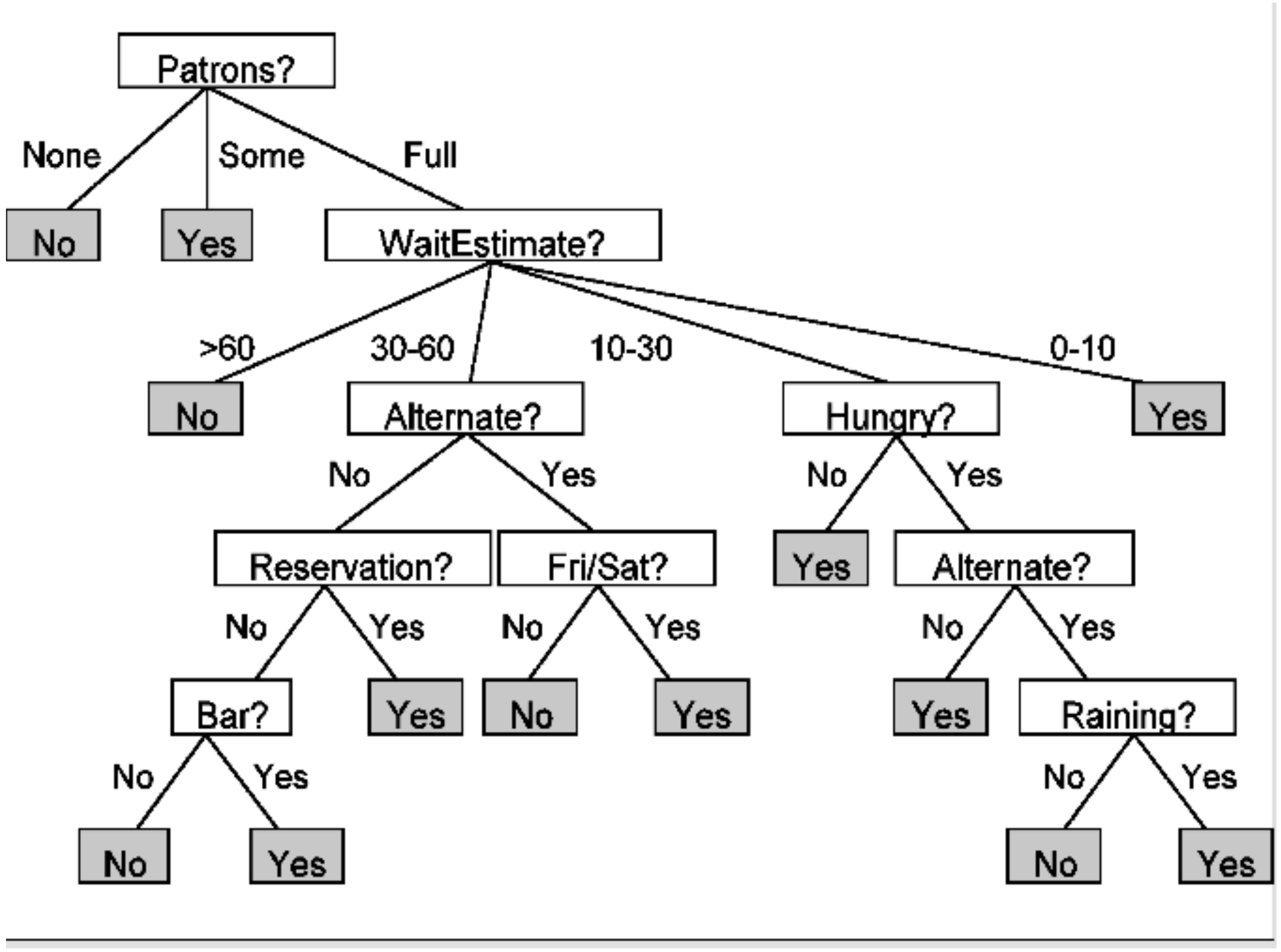
Decision Trees

- **Input examples:** feature vectors
- **Output:**
  - Interior nodes: yes/no decision (e.g. conditionals)
  - Leaf nodes: Action or classification

- A method that can be applied directly to the data without the need for preprocessing or tuning of learning algorithm
- Learns by progressively subdividing data into clusters with homogeneous properties
- Good at determining which features are good discriminators

Decision Trees

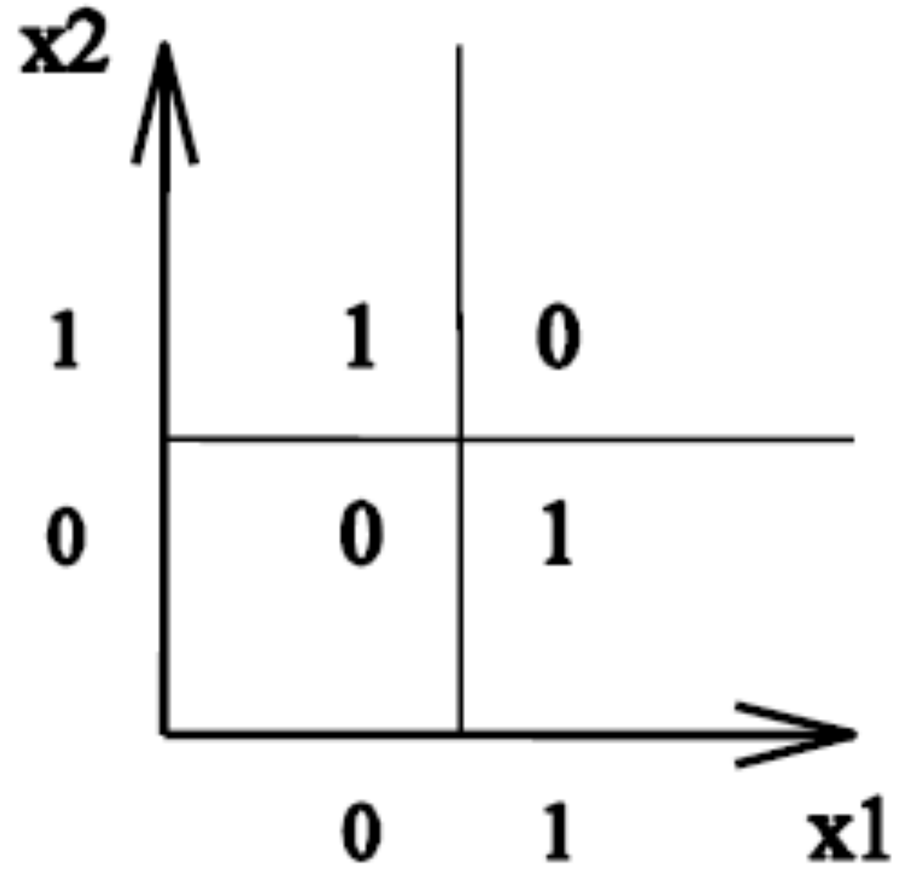
- **Decision tree representation:**
  - Each *internal node* tests an *attribute/ feature*
  - Each *branch* corresponds to *attribute/feature value*
  - Each *leaf* node assigns a *classification*

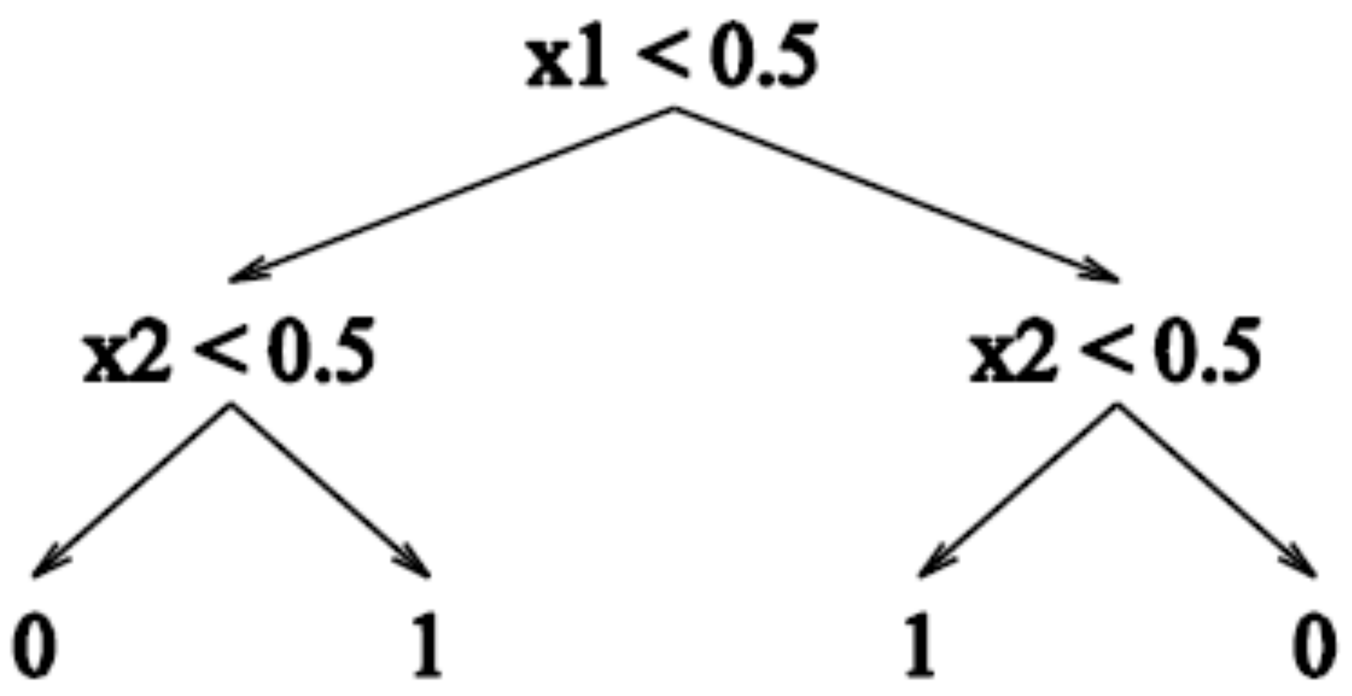


Decision Tree Decision Boundaries

- Decision Trees divide the feature space into axis-parallel rectangles and label each rectangle with one of the K classes

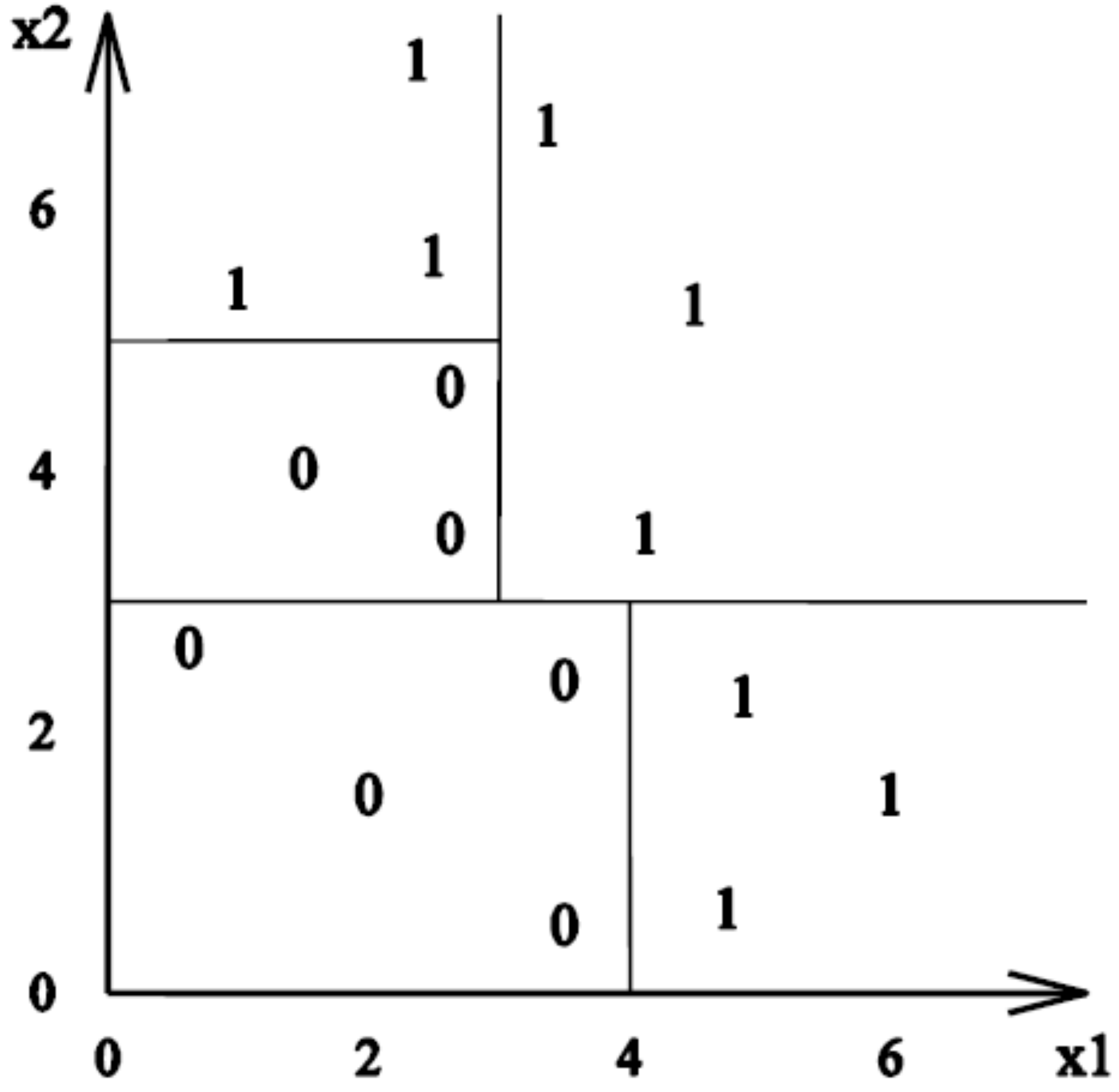
XOR





Decision Tree Decision Boundaries

- What is the Decision Tree for this problem? Write as nested if.



When do you use Decision trees?

- Instances describable by attribute-value pairs
- Target function is discrete valued (though can be used for regression)
- Disjunctive hypothesis (e.g. nested conditions) may be required
- Possibly noisy training data
- Need for interpretable model

Examples:

- Equipment or medical diagnosis
- Credit risk analysis
- Modeling calendar scheduling preferences

Building a decision tree

Classification and Regression Tree (CART) Algorithm

1) **Find a feature *k* and threshold for the feature, t_k,** that splits the data into two subsets in the purest sense (e.g. subsets come from the same class)
2) **Divide the data** into two subsets based on (k, t_k)
3) **Recursively return to step 1** and split the two subsets
4) **Stop when**:
    1) the maximum depth is reached (specified parameter) or
    2) the subsets cannot be split into two subsets that reduce impurity or child nodes contain data of one class

Choosing the Best Partition

Classification and Regression Tree (CART) algorithm

- CART is a ***greedy*** algorithm: searches for optimum split at root of tree and repeats this process at each level. ***Solution is not guaranteed to be optimum***.

- Greedy algorithm's choose the split that optimizes ***performance*** at each node
    - Generate all possible splits
    - For each split, calculate performance
    - Choose split with highest gain or purest subsets

- Performance is measured in terms of ***Gini (measure of purity)*** or ***Entropy (measure of uncertainty)***

CART Impurity Performance Measures

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

n : number of classes

k : index for class

i : index for node in tree

$p_{i,k}$: percentage of class k instances

Find feature and threshold that minimizes the following cost function

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

Gini of right subset

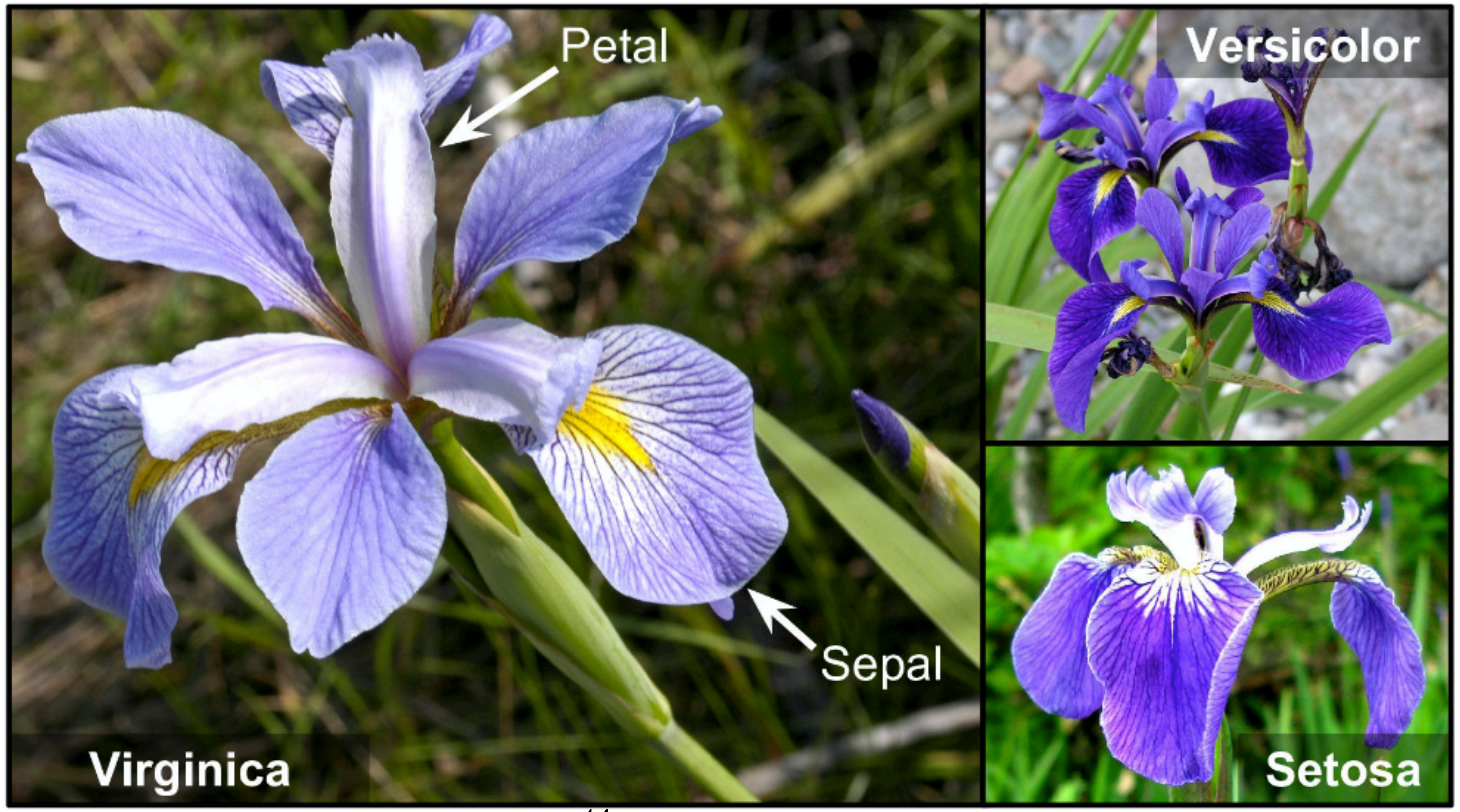
Gini of left subset

m_{left} : number of instances in the left subset

m_{right} : number of instances in the right subset

Iris Dataset Example

- **Goal**: Classify flower based on length and width of petal
- **Flower classes**: Iris-Setosa, Iris-Versicolor, Iris-Virginia



Iris Dataset Example

Train a Decision Tree Classifier: Gini Loss

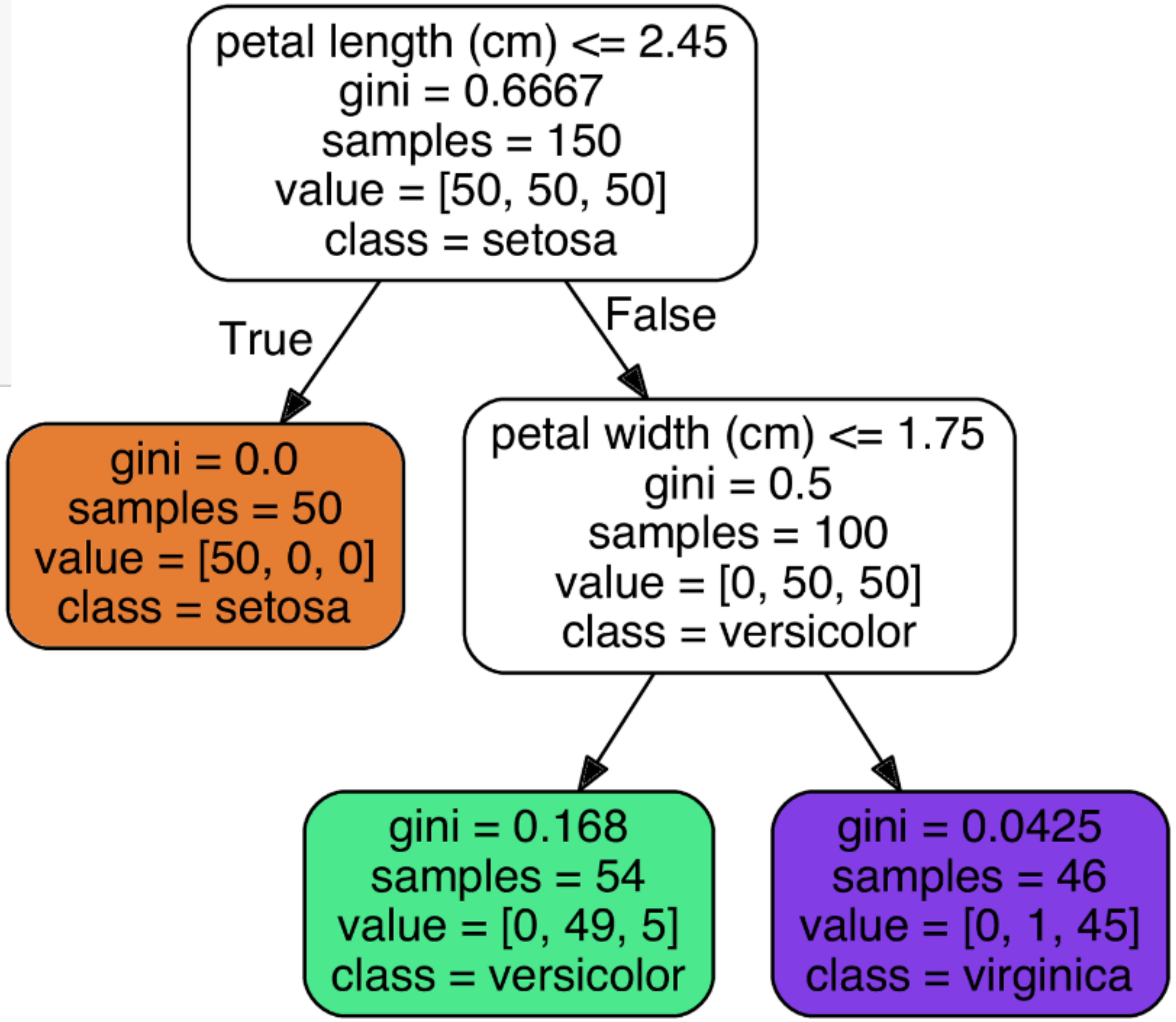
```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier

iris = load_iris()
X = iris.data[:, 2:] # petal length and width
y = iris.target

tree_clf = DecisionTreeClassifier(max_depth=2, random_state=42)
tree_clf.fit(X, y)
```

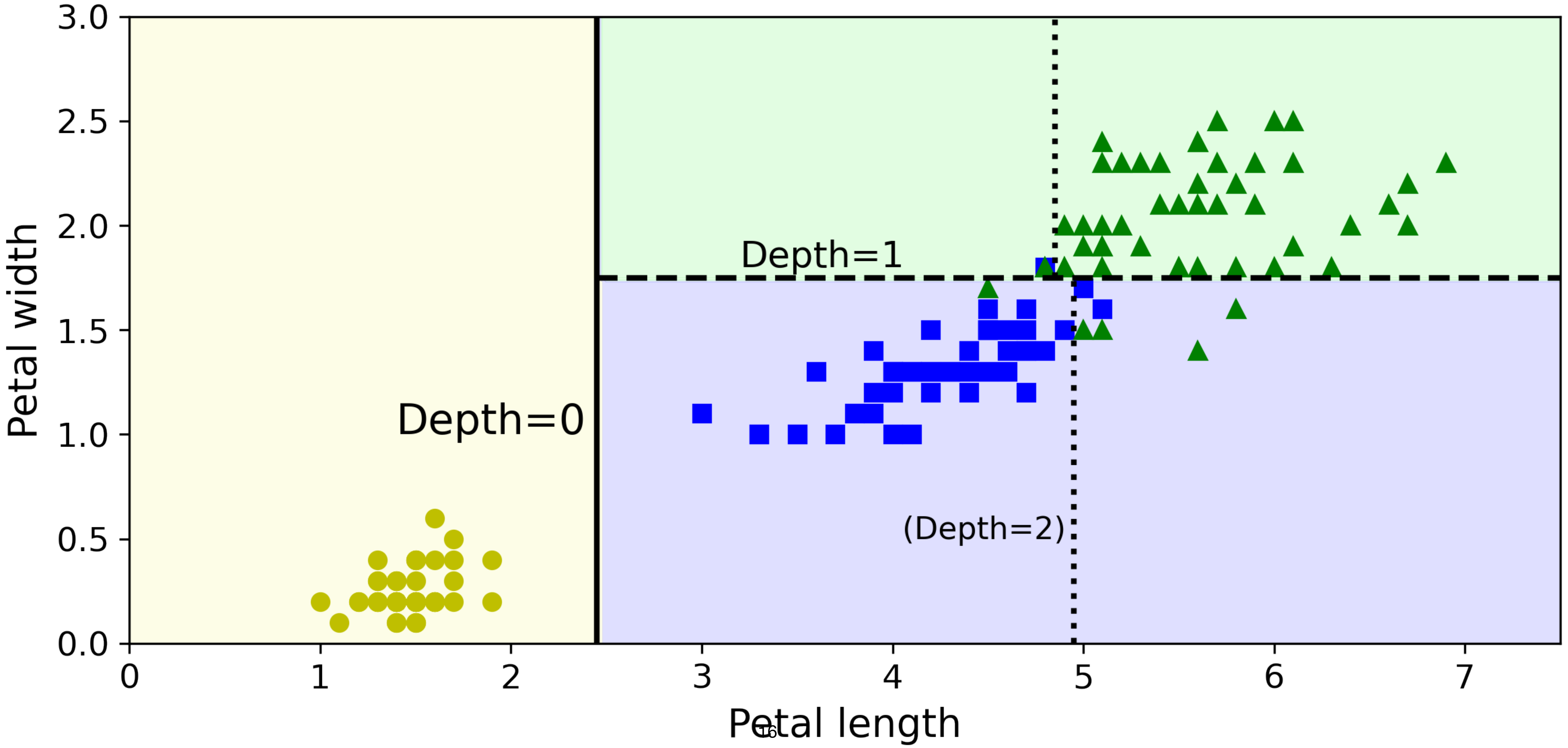
- Gini at root:

$$1 - \left(\frac{50}{150}\right)^2 - \left(\frac{50}{150}\right)^2 - \left(\frac{50}{150}\right)^2$$



Iris Dataset Example

Train a Decision Tree Classifier: Gini Loss



CART Entropy Performance Measures

A measure of uncertainty

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

n : number of classes

k : index for class

i : index for node in tree

$p_{i,k}$: percentage of class k instances

$H_i = 1$ for uniformly distributed data

- **Which measure to use?** Both lead to similar trees, so it does not really matter all the time!
  - Gini is slightly faster
  - Entropy tends to produce slightly more balanced trees

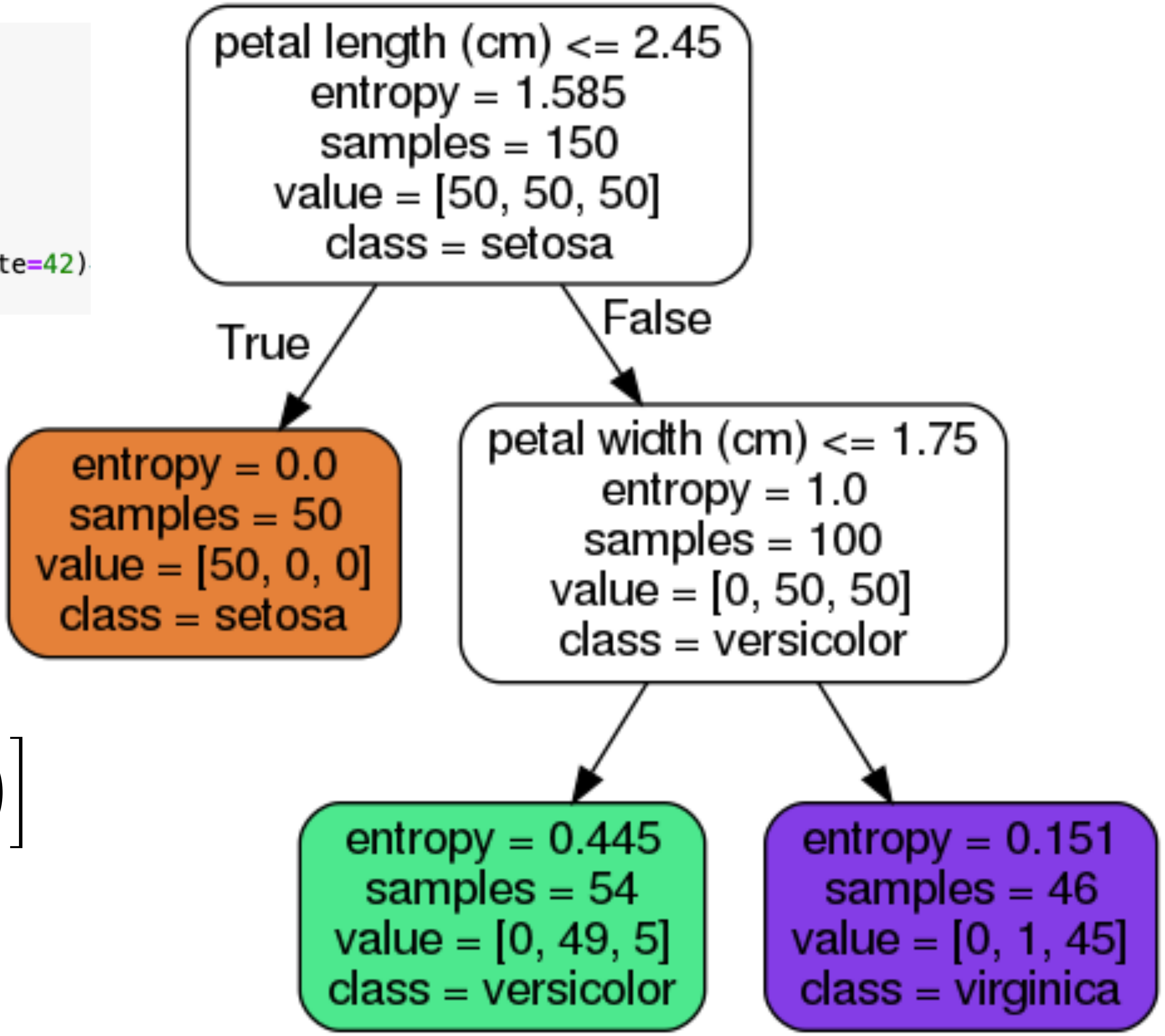
Iris Dataset Example

Train a Decision Tree Classifier: Entropy Loss

```
from sklearn.datasets import load_iris
from sklearn.tree import DecisionTreeClassifier

iris = load_iris()
X = iris.data[:, 2:] # petal length and width
y = iris.target

tree_clf = DecisionTreeClassifier(criterion="entropy",max_depth=2, random_state=42)
tree_clf.fit(X, y)
```



- Resulting Decision Tree (same as before using Gini)
- Entropy at root:

$$H_{root} = -\left[\left(\frac{50}{150}\right)\log_2\left(\frac{50}{150}\right) + \left(\frac{50}{150}\right)\log_2\left(\frac{50}{150}\right) + \left(\frac{50}{150}\right)\log_2\left(\frac{50}{150}\right)\right]$$

Feature and Threshold Selection with Entropy

- When determining when and how to split the data for the nodes, choose the feature k and threshold t_k that has the highest ***mutual information (e.g. information gain)***

- Gain is calculated by computing the difference between the entropy of the parent and the weighted sum of the entropy for the child nodes

Entropy: $$H_i = -\sum_{k=1}^{n} p_{i,k} \log_2(p_{i,k})$$

$H_i = 1$ for uniformly distributed data

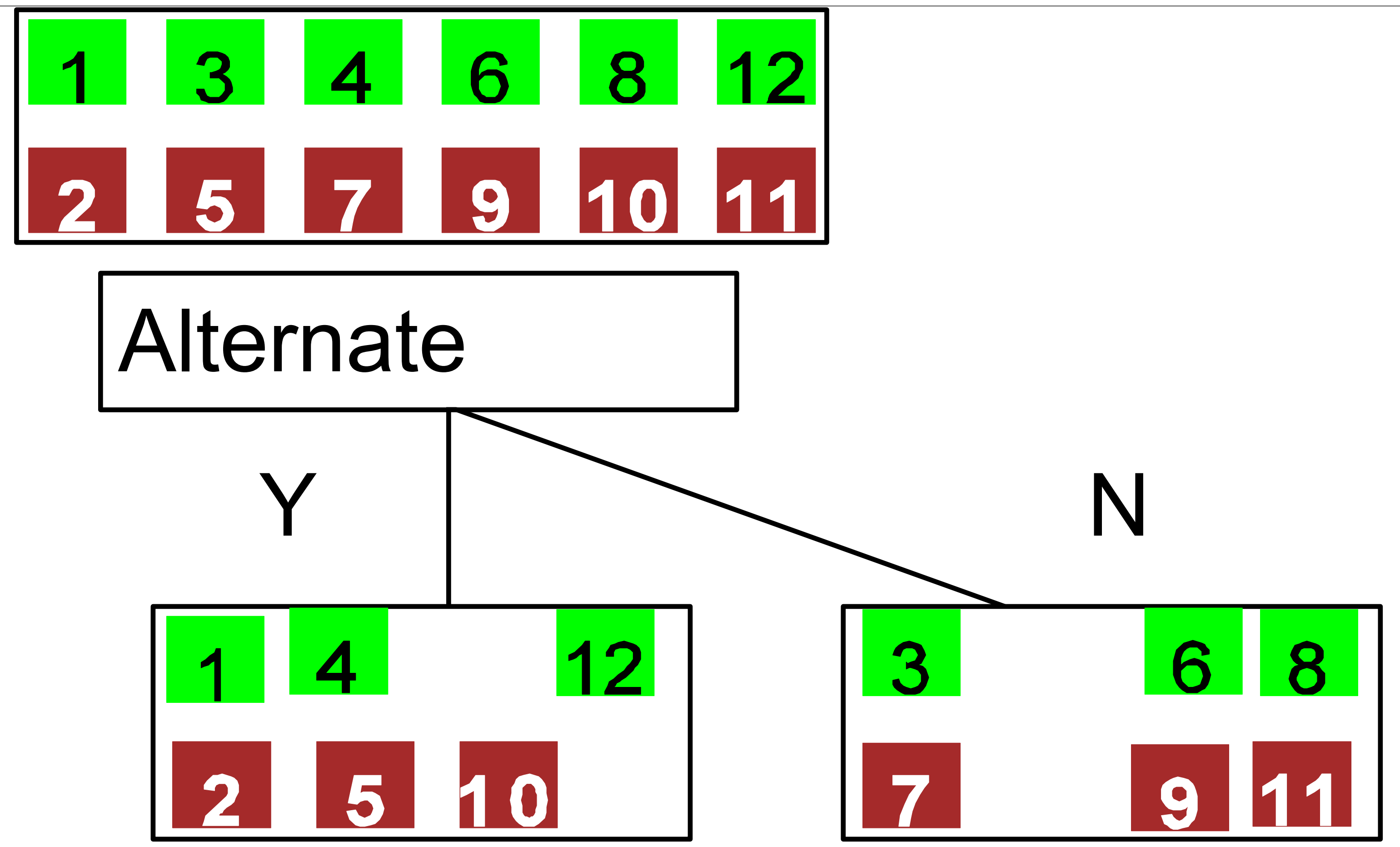
Recall: Decisions

A Restaurant Example

Ex.	Features/Attributes										Target
	Alt.	Bar	Friday	Hungry	Patrons	Price	Rain	Reservati	Type	Wait	Will Wait
X1	T	F	F	T	Some	$$$	F	T	French	0-10	T
X2	T	F	F	T	Full	$	F	F	Thai	30-60	F
X3	F	T	F	F	Some	$	F	F	Burger	0-10	T
X4	T	F	T	T	Full	$	F	F	Thai	10-30	T
X5	T	F	T	F	Full	$$$	F	T	French	>60	F
X6	F	T	F	T	Some	$$	T	T	Italian	0-10	T
X7	F	T	F	F	None	$	T	F	Burger	0-10	F
X8	F	F	F	T	Some	$$	T	T	Thai	0-10	T
X9	F	T	T	F	Full	$	T	F	Burger	>60	F
X10	T	T	T	T	Full	$$$	F	T	Italian	10-30	F
X11	F	F	F	F	None	$	F	F	Thai	0-10	F
X12	T	T	T	T	Full	$	F	F	Burger	30-60	T

Determine what Attribute should be used to split

Consider 'Alternate' restaurant feature



•6 records where Alt=Y and 6 where Alt=N. Hence, $H_{alt} = 1$

Gain = 1 – [6/12*H_{left} + 6/12*H_{right}]

= 1 – [.5*1 + .5*1] = 0

Green: Wait=Y
Brown: Wait=N

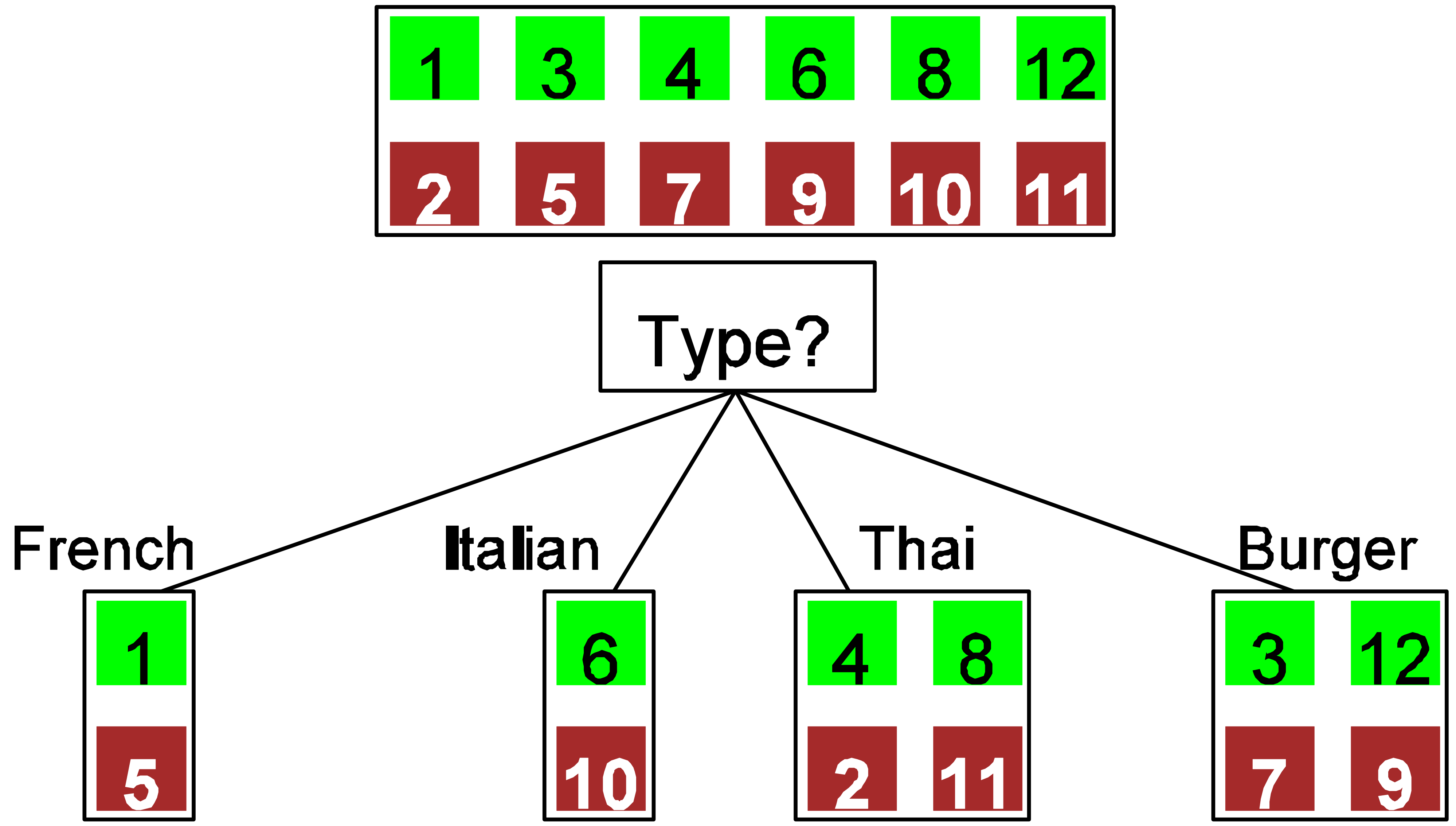
k = 6, 3 have WAIT=t, 3 have WAIT=f

$H_{left} = H_{right}$ = -3/6$\log_2$3/6 – 3/6$\log_2$3/6

Determine what Attribute should be used to split

Consider ‘Type’ restaurant feature

Green: Wait=Y
Brown: Wait=N



$$\text{Gain} = 1 - [2/12 * H_{french} + 2/12 * H_{italian} + 4/12 * H_{Thai} + 4/12 * H_{Burger}]$$

$$= 1 - [1/6 + 1/6 + 2/6 + 2/6] = 0$$

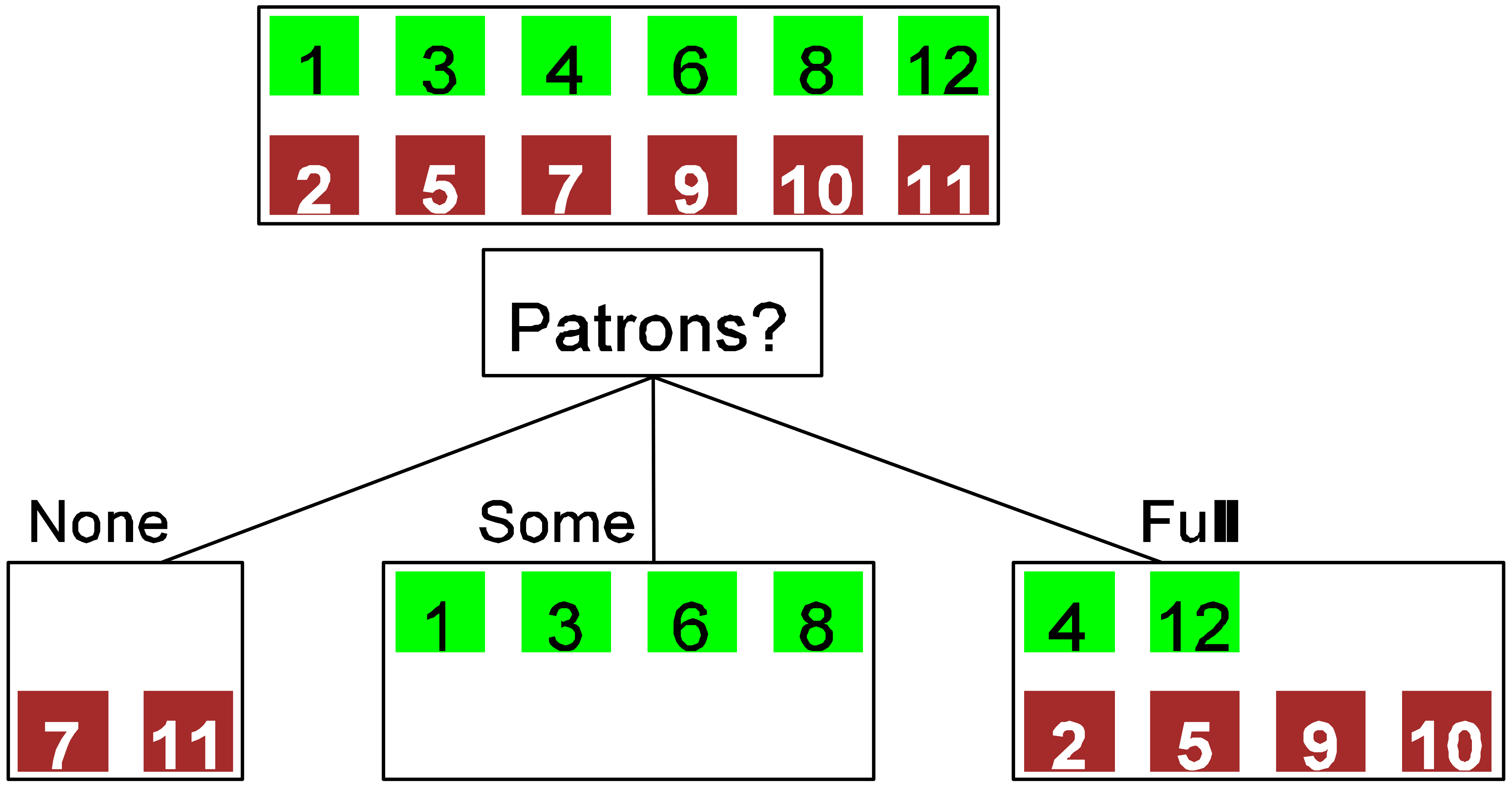
k = 4, 2 have WAIT=t, 2 have WAIT=f

$$H_{Thai} = H_{Burger} = -2/4 \log_2 2/4 - 2/4 \log_2 2/4$$

Determine what Attribute should be used to split

Consider ‘Patrons’ restaurant feature

Green: Wait=Y
Brown: Wait=N



- Gain = 1 – [2/12*H_{None}+4/12*H_{Some}+6/12*H_{Full}]
- = 1 – [2/12*0 + 4/12*0 + 6/12*.918]
- = 1 - .459 = .541

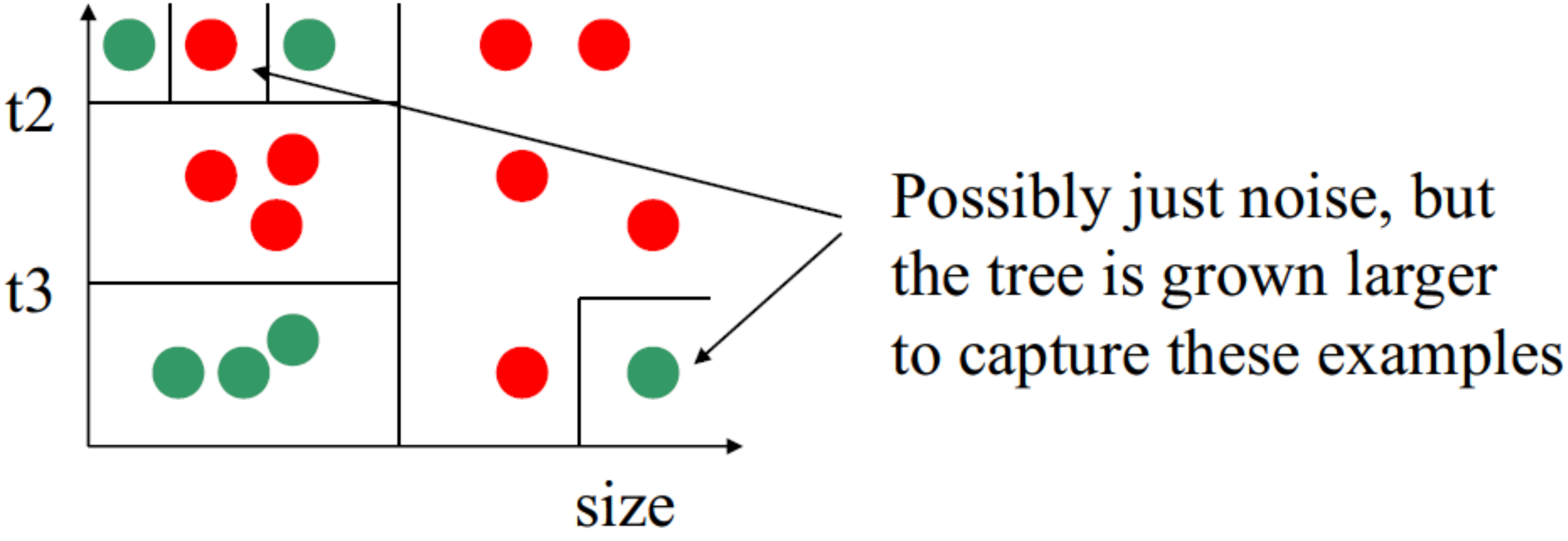
- So this is a better split than ‘Alt’ and ‘Type’

What do we do with the decision tree?

- Present it with new data (test data)
- See how well it classifies
- Report classification success on test data
- Deploy it in the field

Overfitting

- Decision Trees have a very flexible hypothesis space
- As the nodes increase, we can represent arbitrarily complex decision boundaries — training set error is always zero
- This can lead to ***overfitting***



Avoiding Overfitting

Regularization for a Decision Tree

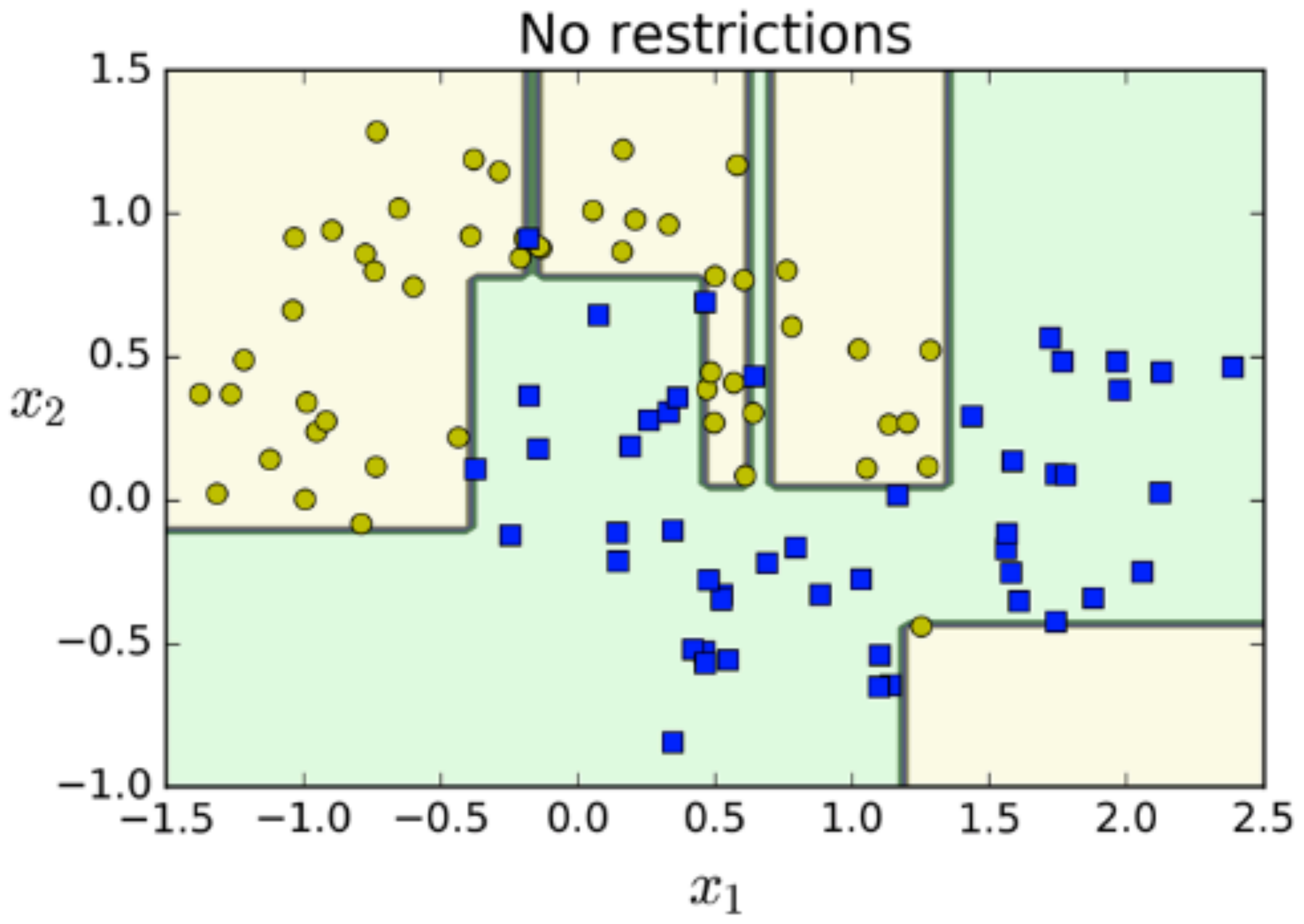
- Typical stopping criterion
  - No error (if all instances belong to same class)
  - If all the attribute values are same

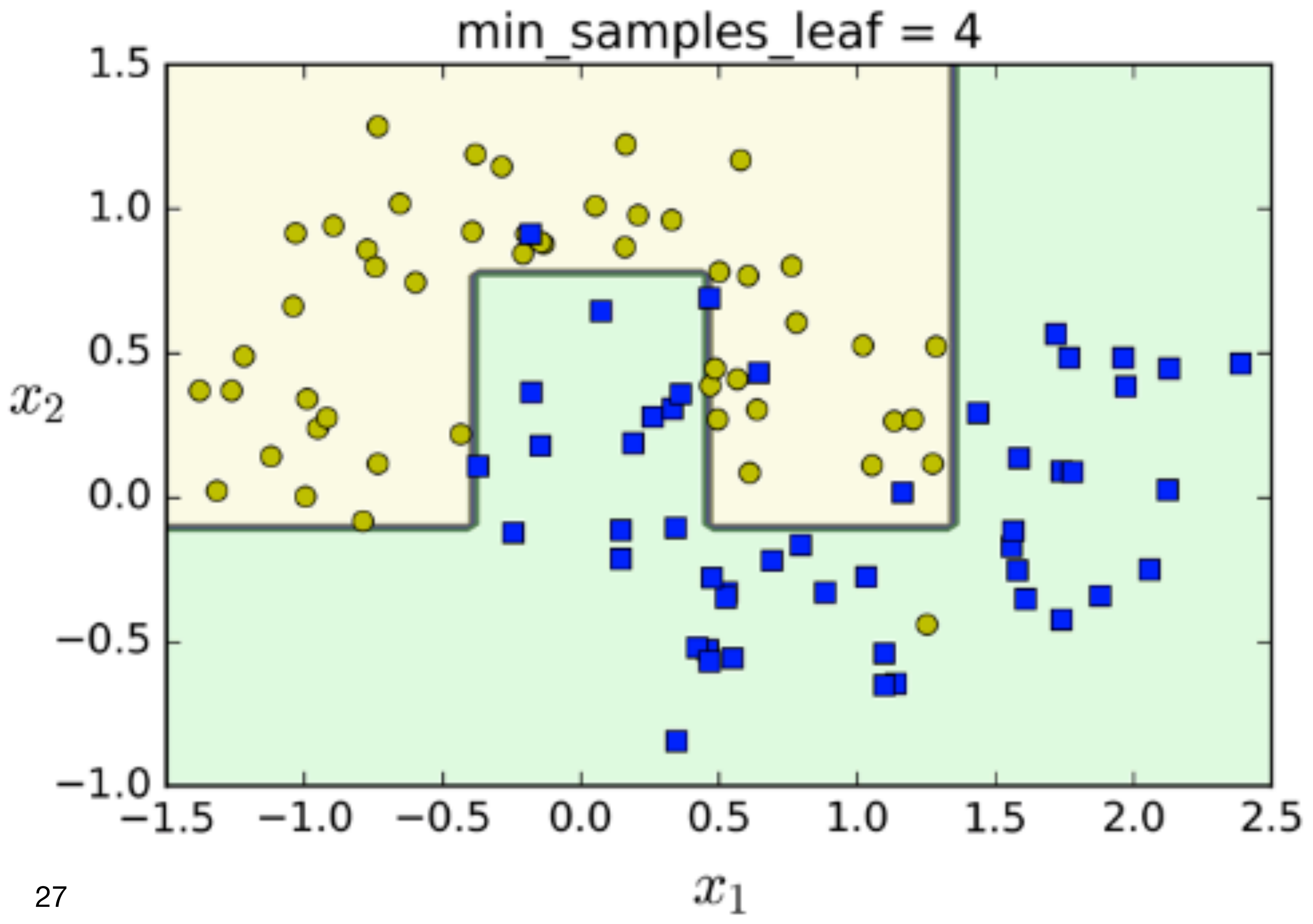
- More restrictive conditions
  - Increase the minimum number of samples a node must have
  - Increase the minimum number of samples a leaf must have
  - Reduce the maximum number of leaf nodes
  - Reduce the maximum allowable depth
  - Reduce the maximum number of features that are evaluated for splitting

Avoiding Overfitting

Setting minimum number for leaf samples

- Decision tree on the left is clearly overfitting
- Regularization avoids overfitting

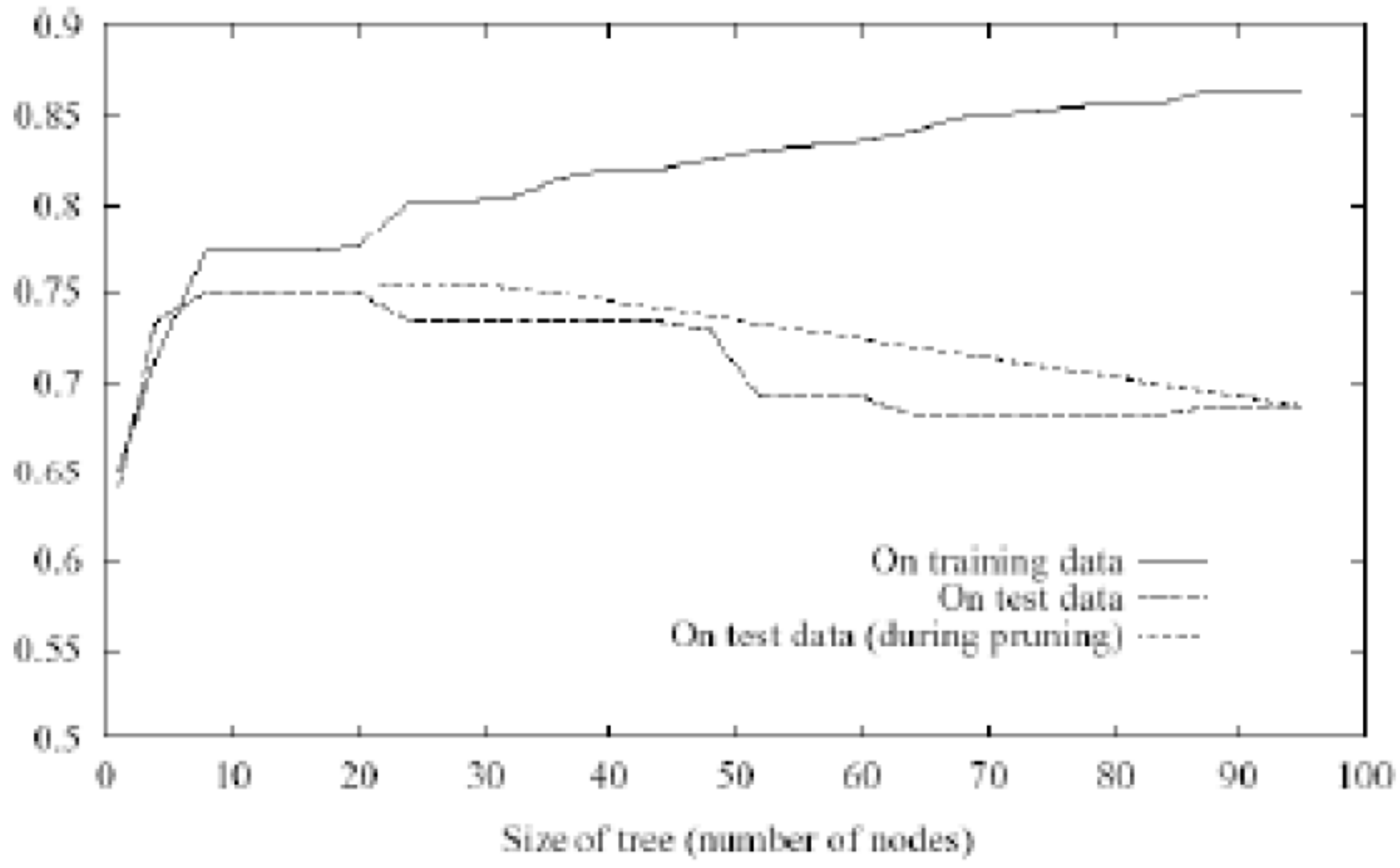




Avoiding Overfitting

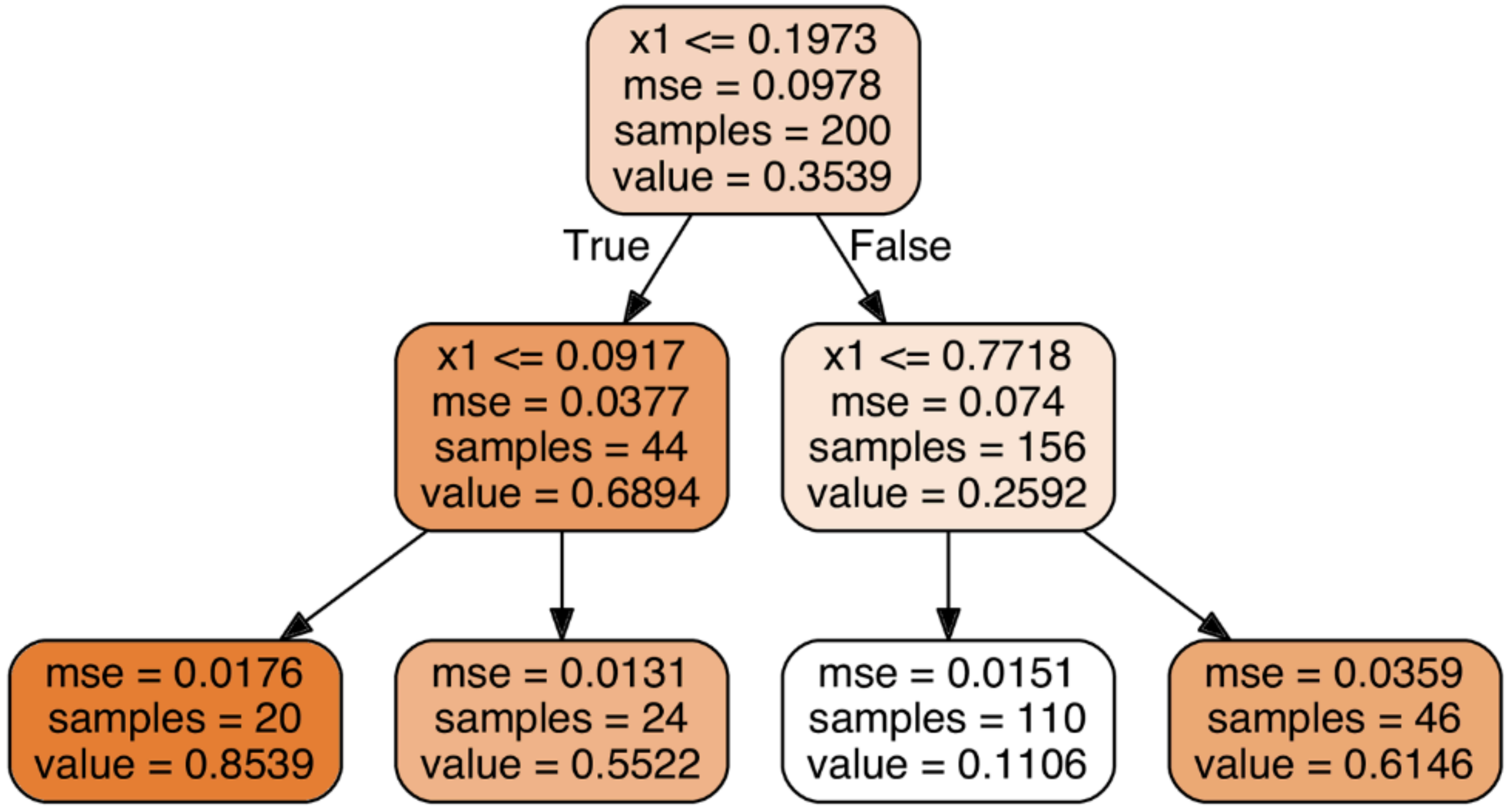
Pruning

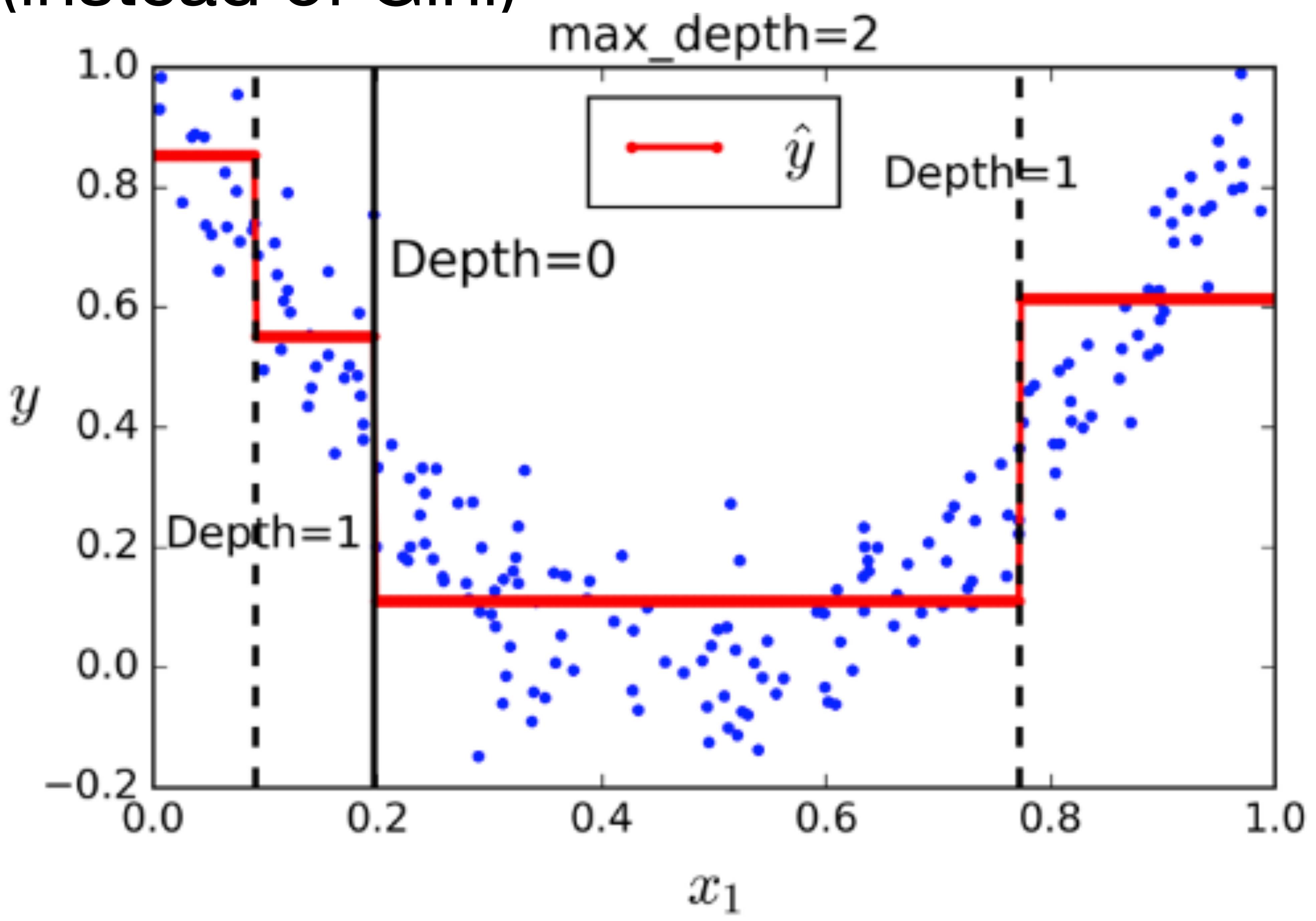
- **Idea**: Grow a full tree, then post-prune
  - Separate training data into training set and validation set
  - Evaluate impact on validation set when a node is “pruned”
  - Greedily remove the one that improves the performance the most
  - Produces the smallest version of most accurate subtree
- May start pruning at root or leaf nodes. Simpliest start at leaf.



Decision Trees for Regression

- In Scikit-Learn, use *DecisionTreeRegressor*
- Instead of predicting a class, it predicts a value
- Traverse tree as is done for classification to generate estimated values
- CART algorithm splits data by minimizing MSE (instead of Gini)





Pros/Cons of Decision Trees

- **Advantages:**
  - Do not require data pre-processing
  - Easy to interpret and understand, unlike other algorithms (e.g. DNNs)
  - Decision trees perform feature selection. Easily can see key features and values for them
- **Disadvantages**:
  - Decision trees can overfit
  - They can be unstable to small variations in the data.
  - Not guaranteed to find globally optimal solution
  - Data set may need to be balanced amongst classes

Next Class

Ensemble Learning and Random Forests