Lecture 06

Decision Trees

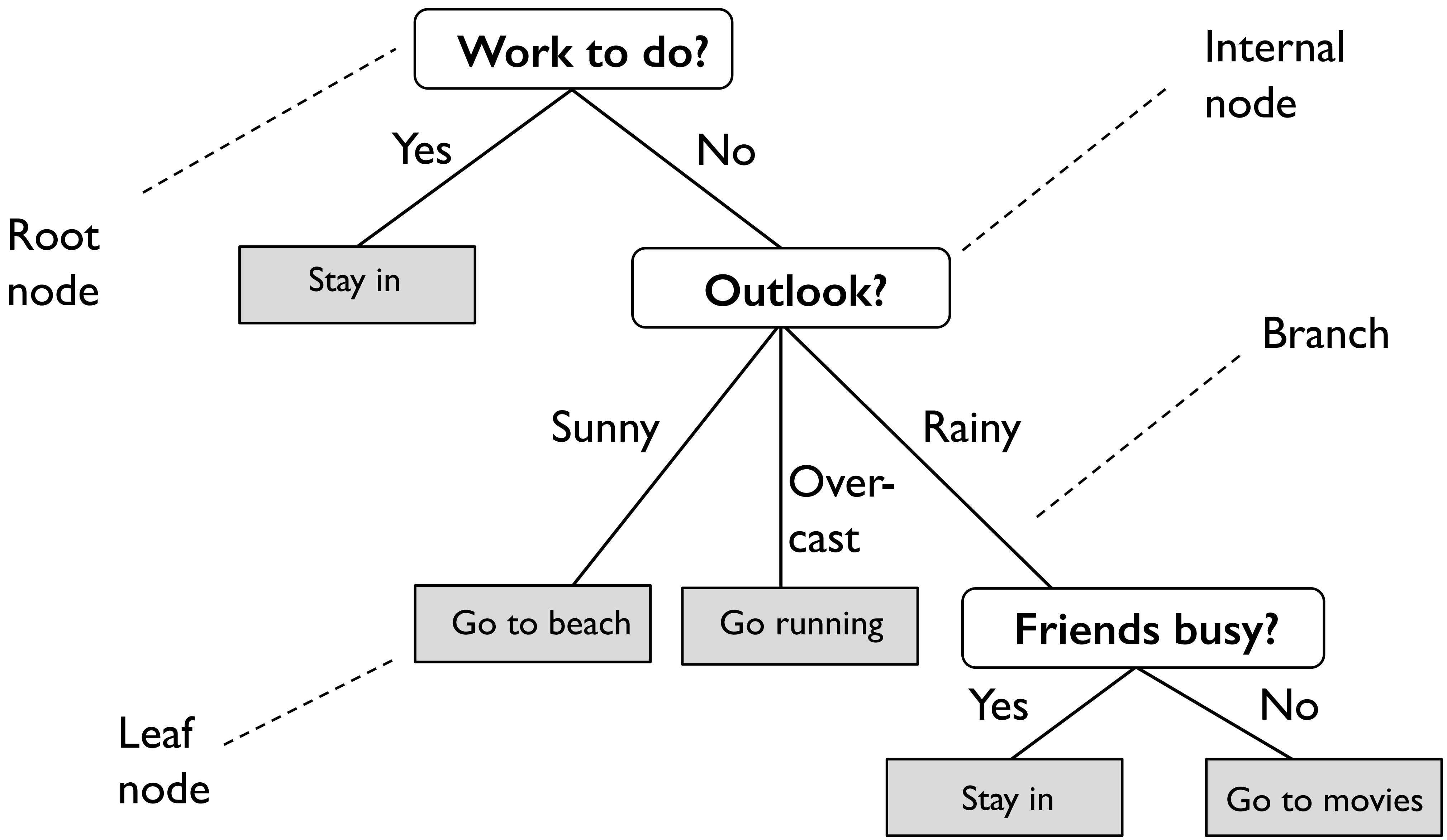
STAT 451: Intro to Machine Learning, Fall 2021

Sebastian Raschka

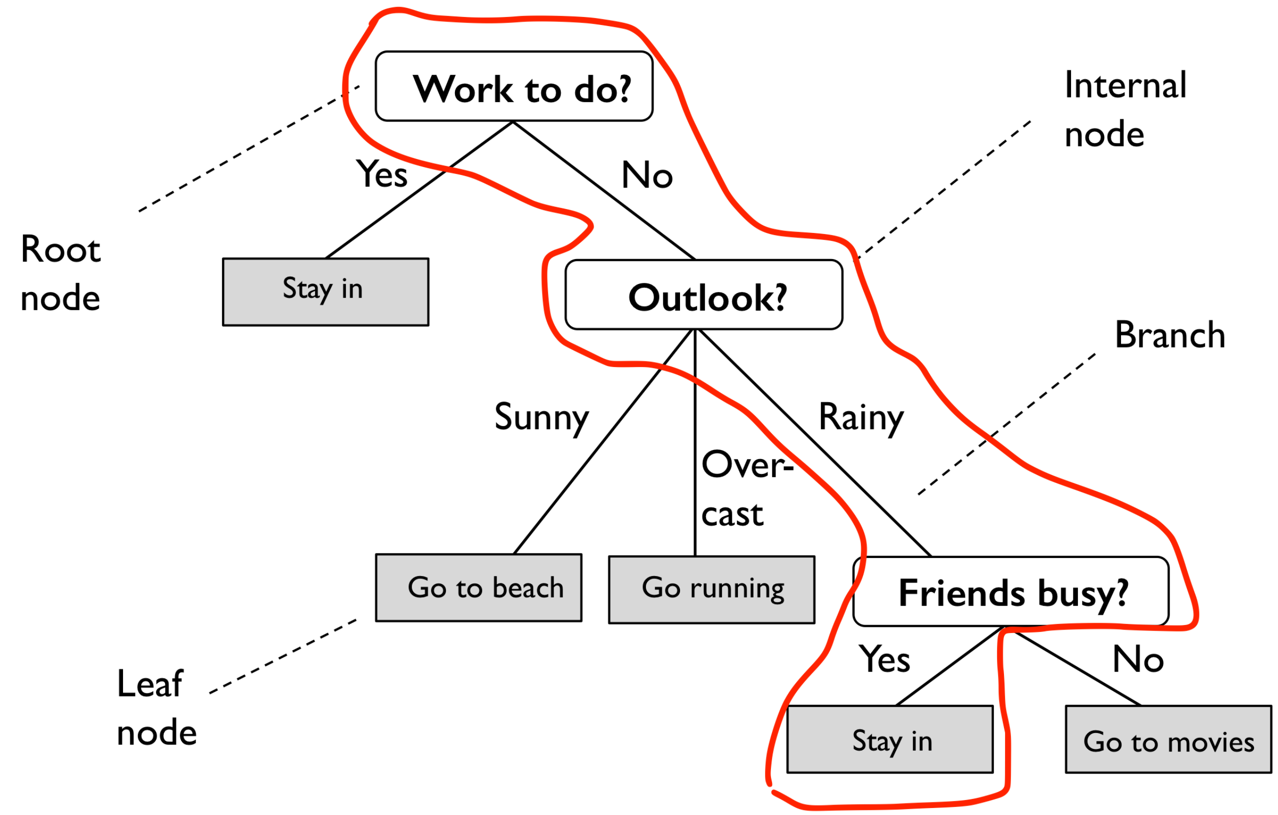
Topics

1. **Intro to decision trees**
2. Recursive algorithms & Big-O
3. Types of decision trees
4. Splitting criteria
5. Gini & Entropy vs misclassification error
6. Improvements & dealing with overfitting
7. Code example

Decision Tree Terminology

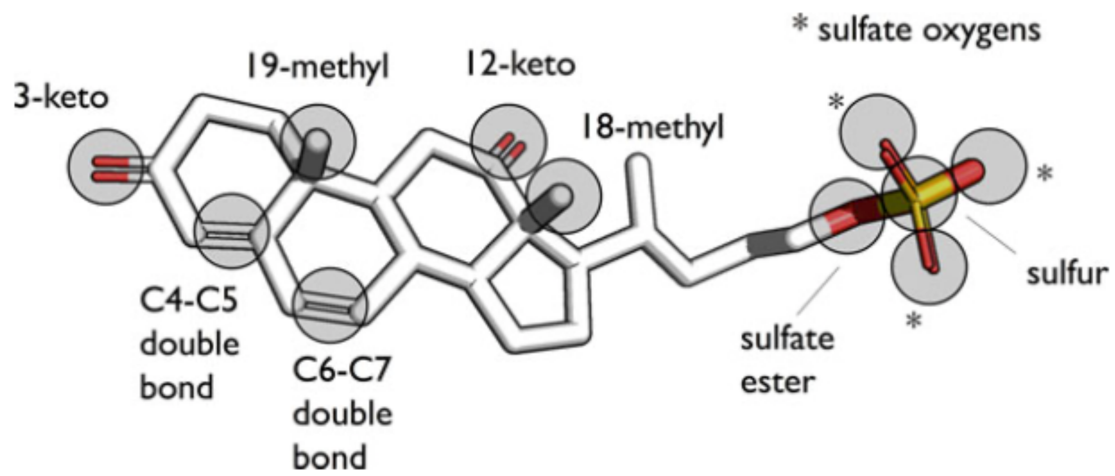


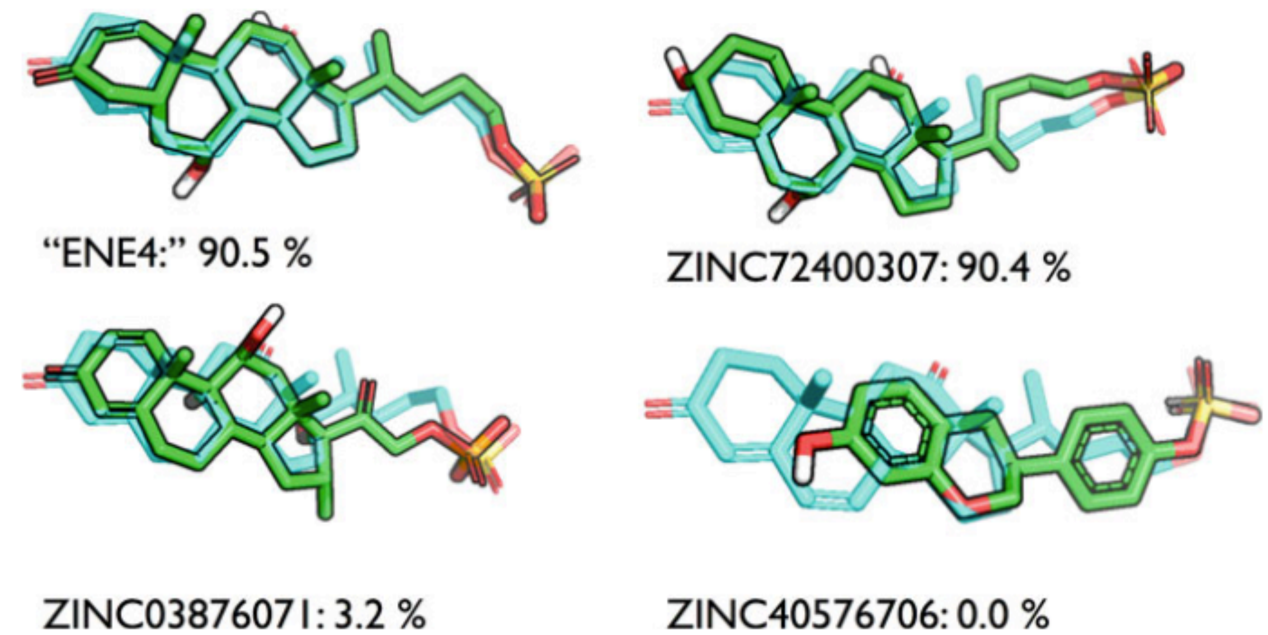
Decision Trees as Rulesets

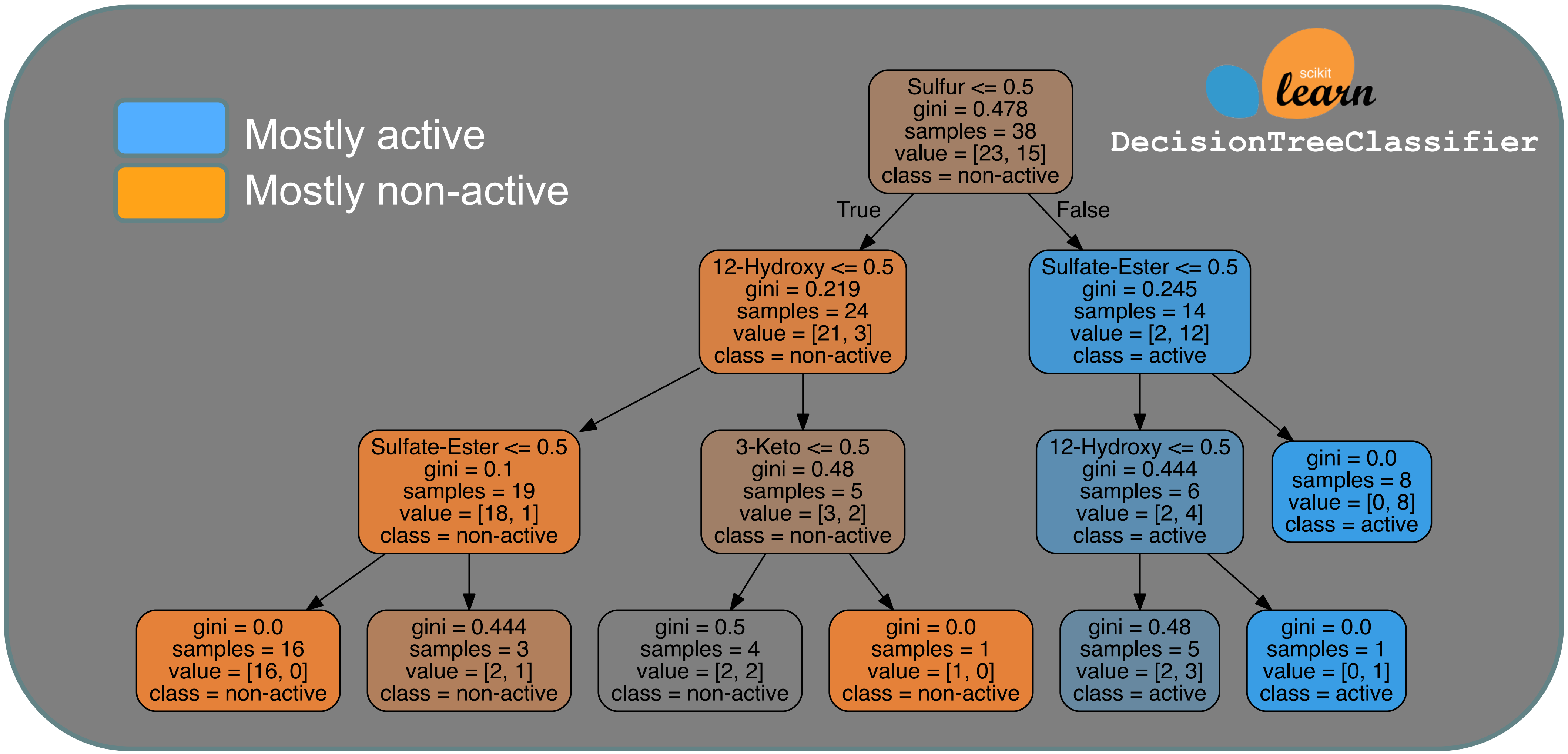


IF

THEN







Sebastian Raschka, Leslie A. Kuhn, Anne M. Scott, and Weiming Li (2018) *Computational Drug Discovery and Design: Automated Inference of Chemical Group Discriminants of Biological Activity from Virtual Screening Data.* Springer. ISBN: 978-1-4939-7755-0

ELSEVIER

Computer Methods and Programs in Biomedicine

Volume 192, August 2020, 105400

Decision tree-based diagnosis of coronary artery disease: CART model

Mohammad M. Ghiasi [a], Sohrab Zendehboudi [a], Ali Asghar Mohsenipour [b]

Show more

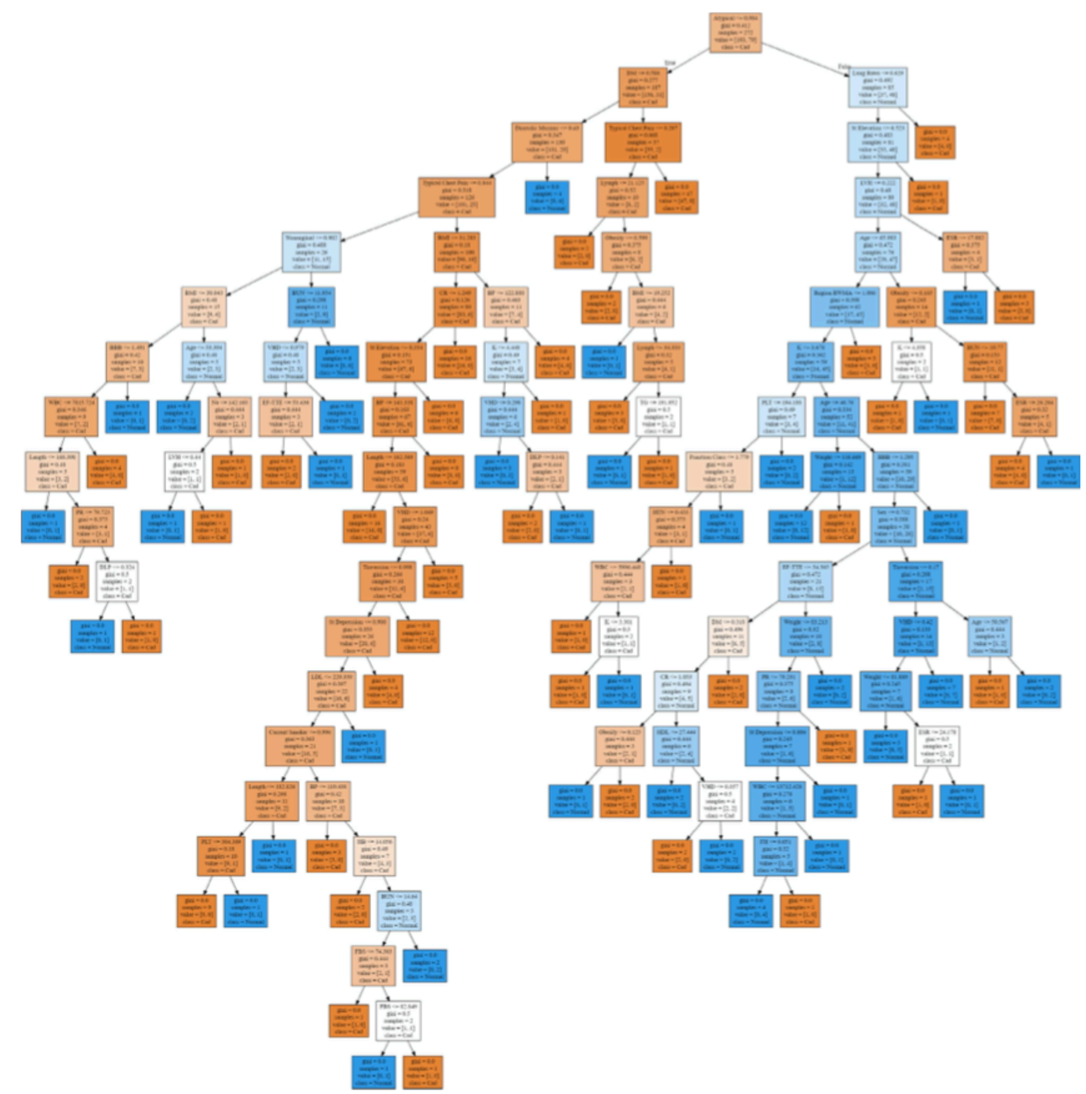


Fig. 2. Graphical representation of the CART model (using all features) introduced for CAD diagnosis.

https://www.sciencedirect.com/science/article/abs/pii/S0169260719308971

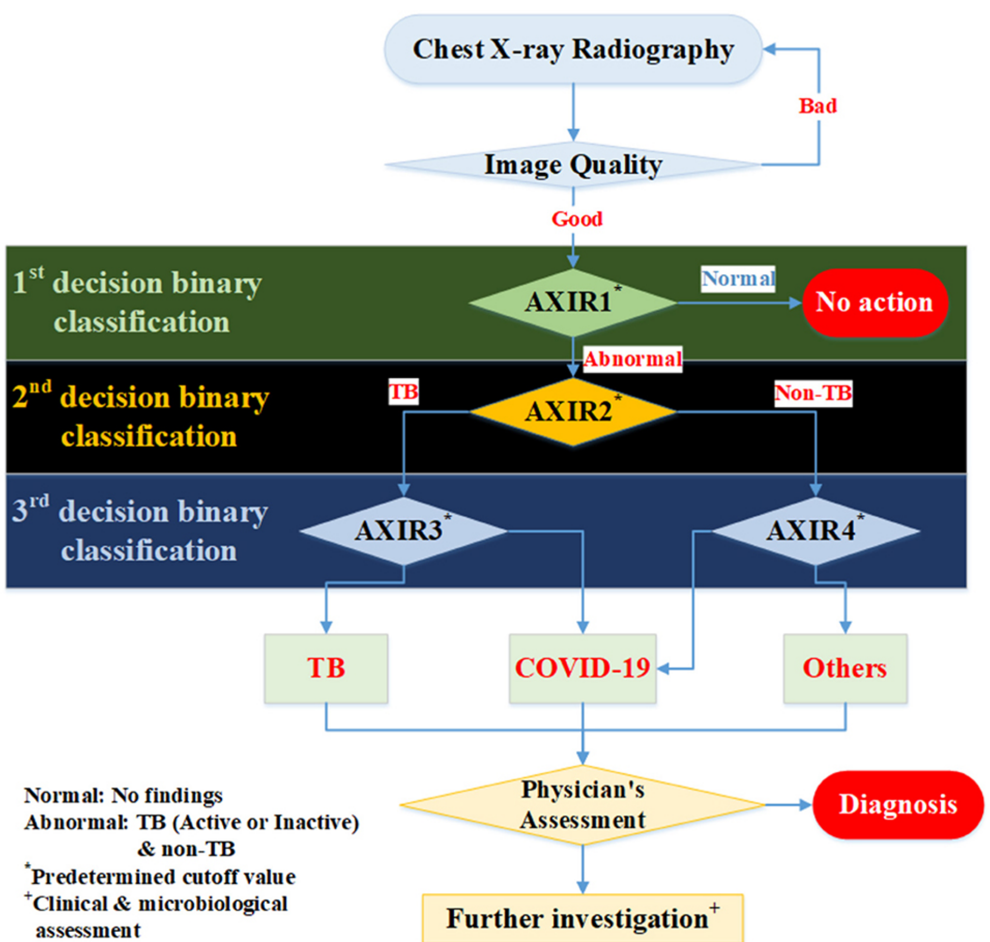
ORIGINAL RESEARCH article

Front. Med., 14 July 2020 | https://doi.org/10.3389/fmed.2020.00427

Deep Learning-Based Decision-Tree Classifier for COVID-19 Diagnosis From Chest X-ray Imaging

Seung Hoon Yoo[1], Hui Geng[1], Tin Lok Chiu[1], Siu Ki Yu[1], Dae Chul Cho[2], Jin Heo[2], Min Sung Choi[2], Il Hyun Choi[2], Cong Cung Van[3], Nguen Viet Nhung[3], Byung Jun Min[4*] and Ho Lee[5*]

https://www.frontiersin.org/articles/10.3389/fmed.2020.00427/full



Random forests, adaptive boosting, gradient boosting

Topics

1. Intro to decision trees
2. **Recursive algorithms & Big-O**
3. Types of decision trees
4. Splitting criteria
5. Gini & Entropy vs misclassification error
6. Improvements & dealing with overfitting
7. Code example

Recursion / Recursive Algorithms

```
def some_fun(x):
    if x == []:
        return 0
    else:
        return 1 + some_fun(x[1:])
```

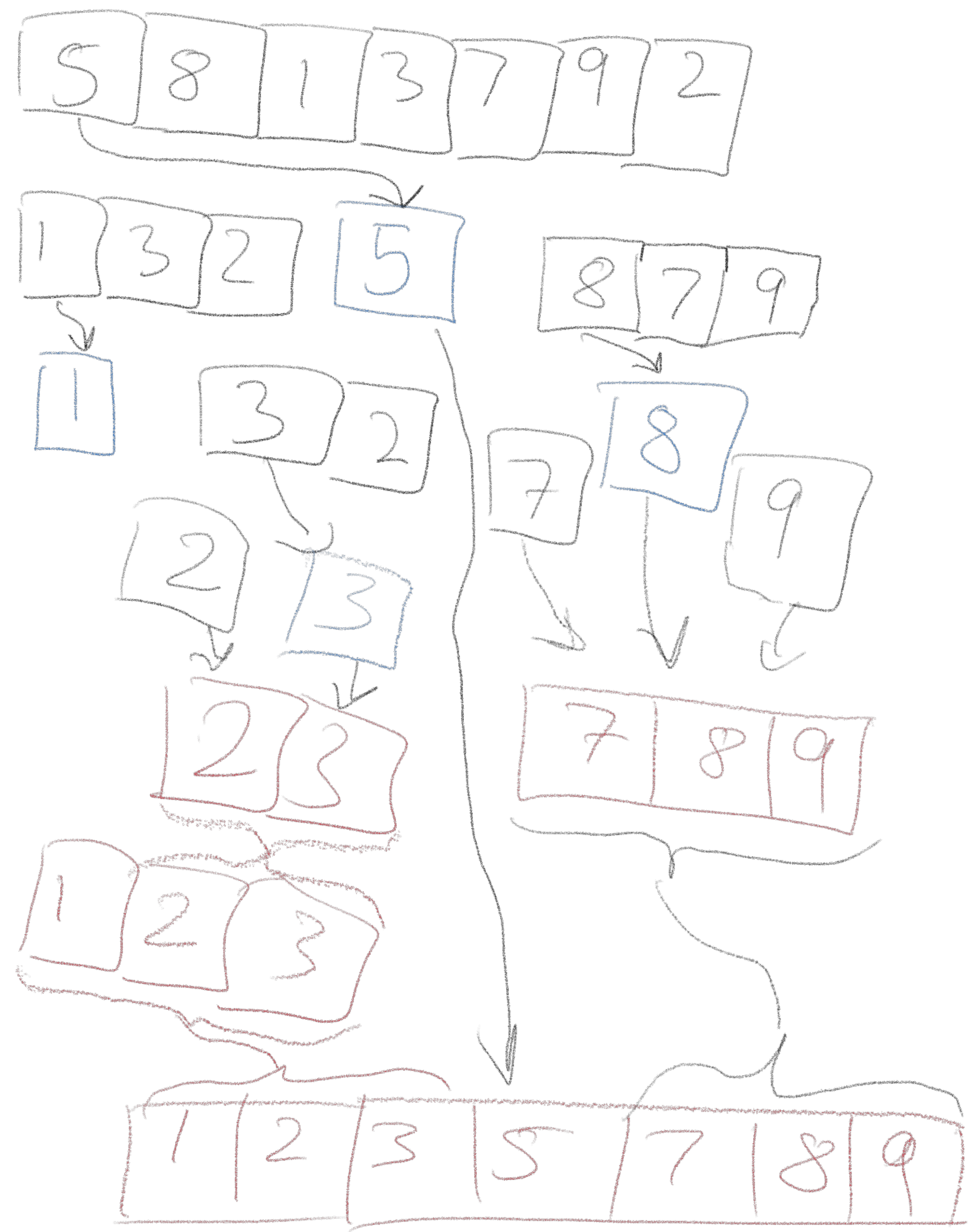
What does this function do?

Divide & Conquer Algorithms: Quicksort

```
def quicksort(array):
    if len(array) < 2:
        return array
    else:
        pivot = array[0]
        smaller, bigger = [], []
        for ele in array[1:]:
            if ele <= pivot:
                smaller.append(ele)
            else:
                bigger.append(ele)
        return quicksort(smaller) + [pivot] + quicksort(bigger)
```

Divide & Conquer Algorithms: Quicksort

```python
def quicksort(array):
    if len(array) < 2:
        return array
    else:
        pivot = array[0]
        smaller, bigger = [], []
        for ele in array[1:]:
            if ele <= pivot:
                smaller.append(ele)
            else:
                bigger.append(ele)
        return quicksort(smaller) + [pivot] + quicksort(bigger)
```



Time complexity of quicksort:

$\mathcal{O}(\_\_\_\_\_\_\_\_\_\_)$

```
def quicksort(array):
    if len(array) < 2:
        return array
    else:
        pivot = array[0]
        smaller, bigger = [], []
        for ele in array[1:]:
            if ele <= pivot:
                smaller.append(ele)
            else:
                bigger.append(ele)
        return quicksort(smaller) + [pivot] + quicksort(bigger)
```

Array Sorting Algorithms

Algorithm	Time Complexity			Space Complexity
	Best	Average	Worst *	Worst
Quicksort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n^2)$	$O(\log(n))$
Mergesort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n \log(n))$	$O(n)$
Timsort	$\Omega(n)$	$\Theta(n \log(n))$	$O(n \log(n))$	$O(n)$
Heapsort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n \log(n))$	$O(1)$
Bubble Sort	$\Omega(n)$	$\Theta(n^2)$	$O(n^2)$	$O(1)$
Insertion Sort	$\Omega(n)$	$\Theta(n^2)$	$O(n^2)$	$O(1)$
Selection Sort	$\Omega(n^2)$	$\Theta(n^2)$	$O(n^2)$	$O(1)$
Tree Sort	$\Omega(n \log(n))$	$\Theta(n \log(n))$	$O(n^2)$	$O(n)$
Shell Sort	$\Omega(n \log(n))$	$\Theta(n(\log(n))^2)$	$O(n(\log(n))^2)$	$O(1)$
Bucket Sort	$\Omega(n+k)$	$\Theta(n+k)$	$O(n^2)$	$O(n)$
Radix Sort	$\Omega(nk)$	$\Theta(nk)$	$O(nk)$	$O(n+k)$
Counting Sort	$\Omega(n+k)$	$\Theta(n+k)$	$O(n+k)$	$O(k)$
Cubesort	$\Omega(n)$	$\Theta(n \log(n))$	$O(n \log(n))$	$O(n)$

http://www.bigocheatsheet.com

* "worst" ~ inversely-sorted array

Decision Tree in Pseudocode

GenerateTree($\mathcal{D}$):

- if $y = 1 \; \forall \; \langle \mathbf{x}, \mathbf{y} \rangle \in \mathcal{D}$ or $y = 0 \; \forall \; \langle \mathbf{x}, y \rangle \in \mathcal{D}$:
  - return Tree
- else:
  - Pick best feature x_j:
    - $\mathcal{D}_0$ at $\text{Child}_0 : x_j = 0 \; \forall \; \langle \mathbf{x}, y \rangle \in \mathcal{D}$
    - $\mathcal{D}_1$ at $\text{Child}_1 : x_j = 1 \; \forall \; \langle \mathbf{x}, y \rangle \in \mathcal{D}$

  return Node(x_j, GenerateTree($\mathcal{D}_0$), GenerateTree($\mathcal{D}_1$))

Time Complexity ("Big-O")

Growing the tree: $\mathcal{O}(\ldots$

Tip: It can be shown that optimal split is on boundary between adjacent examples (similar feature value) with different class labels. $\longrightarrow$ Consider sorting

Fayyad, Usama Mohammad. "On the induction of decision trees for multiple concept learning." (1992).

Sorting a feature column : $\mathcal{O}(n \log n)$

Consider/sort m feature columns: $\mathcal{O}(m)$

Decision tree has up to n terminal leaf nodes

There are $2n - 1$ internal nodes in the tree

The number we split nodes is $2n - 1 - n = n - 1$

Complexity with re-sorting: $\mathcal{O}(mn^2 \log n)$

Complexity with sorting once & caching: $\mathcal{O}(mn \log n)$

Time Complexity ("Big-O")

Querying the tree: $\mathcal{O}(\ldots$

Topics

1. Intro to decision trees

2. Recursive algorithms & Big-O

3. Types of decision trees

4. Splitting criteria

5. Gini & Entropy vs misclassification error

6. Improvements & dealing with overfitting

7. Code example

Decision Tree in Pseudocode

GenerateTree($\mathcal{D}$):

- if $y = 1 \; \forall \; \langle \mathbf{x}, \mathbf{y} \rangle \in \mathcal{D}$ or $y = 0 \; \forall \; \langle \mathbf{x}, y \rangle \in \mathcal{D}$:
    - return Tree
- else:
    - Pick best feature x_j:
        - $\mathcal{D}_0$ at $\text{Child}_0 : x_j = 0 \; \forall \; \langle \mathbf{x}, y \rangle \in \mathcal{D}$
        - $\mathcal{D}_1$ at $\text{Child}_1 : x_j = 1 \; \forall \; \langle \mathbf{x}, y \rangle \in \mathcal{D}$

    return Node(x_j, GenerateTree($\mathcal{D}_0$), GenerateTree($\mathcal{D}_1$))

Generic Tree Growing Algorithm

1) Pick the feature that, when parent node is split, results in the largest information gain

2) Stop if child nodes are pure or information gain <= 0

3) Go back to step 1 for each of the two child nodes

Generic Tree Growing Algorithm

1) Pick the feature that, when parent node is split, results in the largest information gain

2) Stop if child nodes are pure or information gain <= 0

3) Go back to step 1 for each of the two child nodes

- How make predictions if features in dataset are not sufficient to make child nodes pure?

Design choices

- How to split
  - what measurement/criterion as measure of 'goodness'
  - binary vs multi-category split

- When to stop
  - if leaf nodes contain only examples of the same class
  - feature values are all the same for all examples
  - maximum number of splits
  - score threshold or statistical significance test

ID3 -- Iterative Dichotomizer 3

- one of the earlier decision tree algorithms
- Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81-106.
- cannot handle numeric features
- no pruning, prone to overfitting
- short and wide trees (compared to CART)
- maximizing information gain/minimizing entropy
- discrete features, binary and multi-category features

C4.5

- continuous and discrete features
- Ross Quinlan 1993, Quinlan, J. R. (1993). C4.5: Programming for machine learning. *Morgan Kauffmann*, *38*, 48.
- continuous is very expensive, because must consider all possible ranges
- handles missing attributes (ignores them in gain compute)
- post-pruning (bottom-up pruning)
- Gain Ratio

CART

- Breiman, L. (1984). *Classification and regression trees*. Belmont, Calif: Wadsworth International Group.
- continuous and discrete features
- strictly binary splits (taller trees than ID3, C4.5)
- binary splits can generate better trees than C4.5, but tend to be larger and harder to interpret; k-attributes has a ways to create a binary partitioning
- variance reduction in regression trees
- Gini impurity, twoing criteria in classification trees
- cost complexity pruning

Others

- CHAID (CHi-squared Automatic Interaction Detector); Kass, G. V. (1980). "An exploratory technique for investigating large quantities of categorical data". *Applied Statistics*. 29 (2): 119–127.
- MARS (Multivariate adaptive regression splines); Friedman, J. H. (1991). "Multivariate Adaptive Regression Splines". *The Annals of Statistics*. 19: 1
- C5.0 (patented)
- ...

Topics

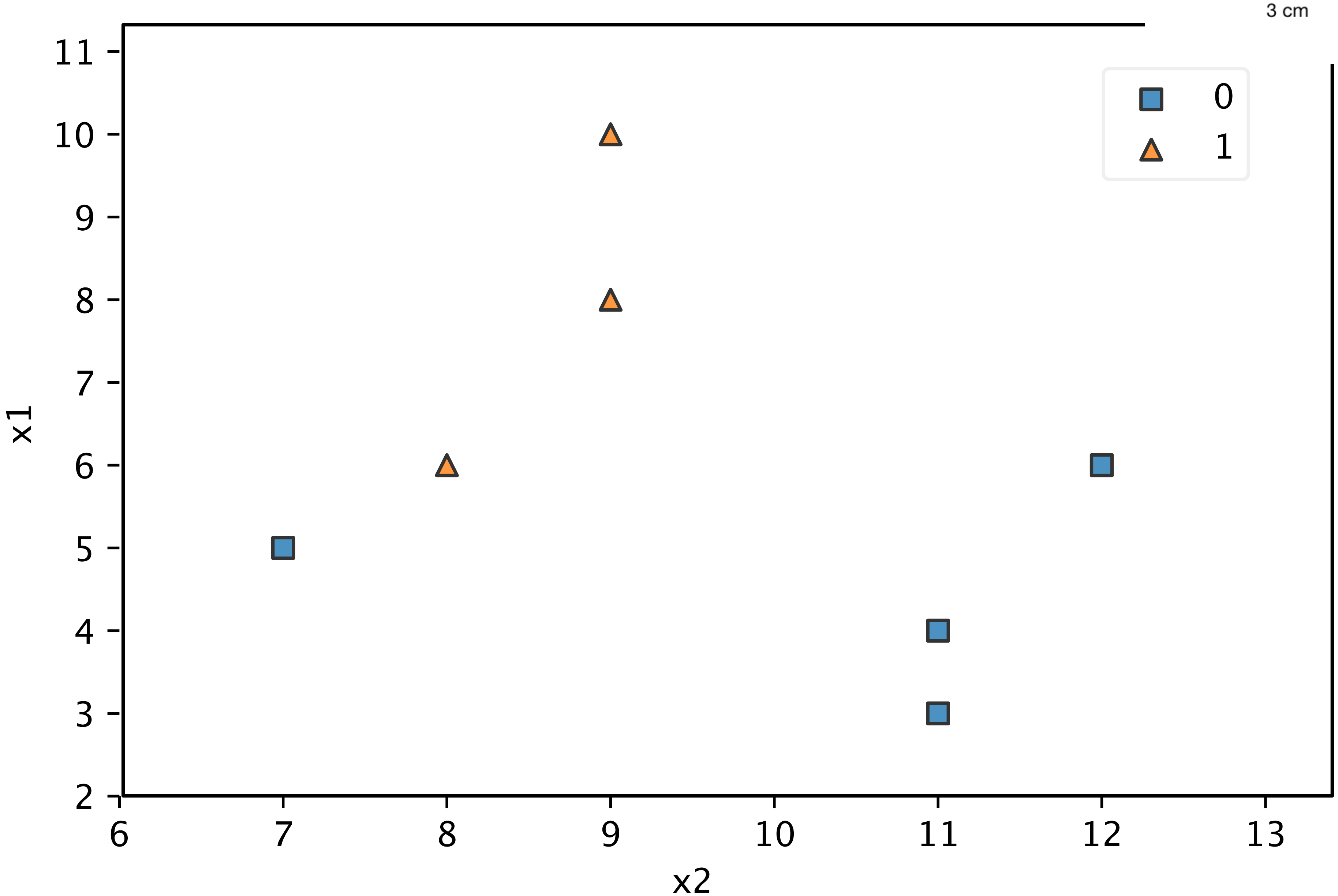
1. Intro to decision trees
2. Recursive algorithms & Big-O
3. Types of decision trees
4. **Splitting criteria**
5. Gini & Entropy vs misclassification error
6. Improvements & dealing with overfitting
7. Code example

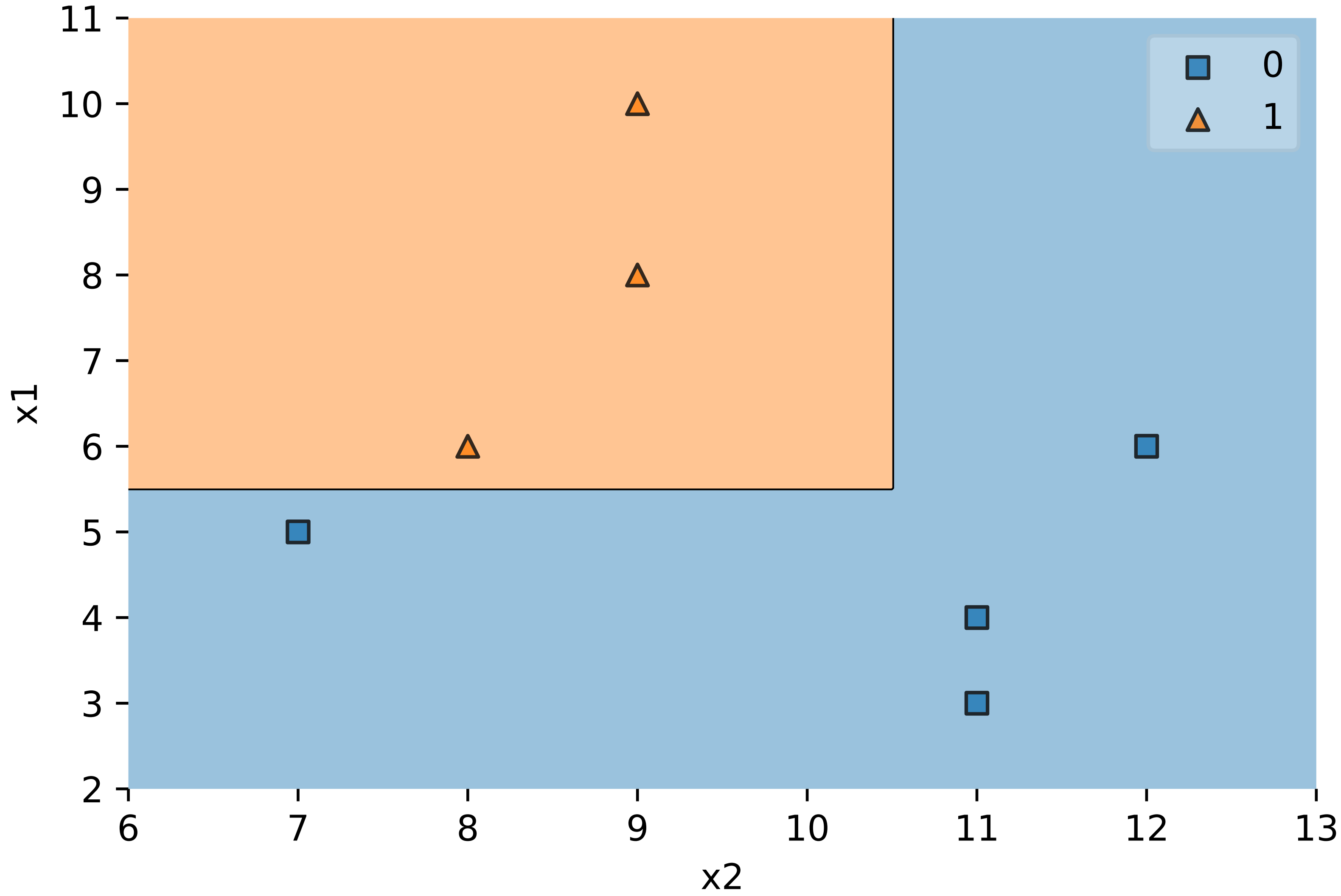
Finding a Decision Rule

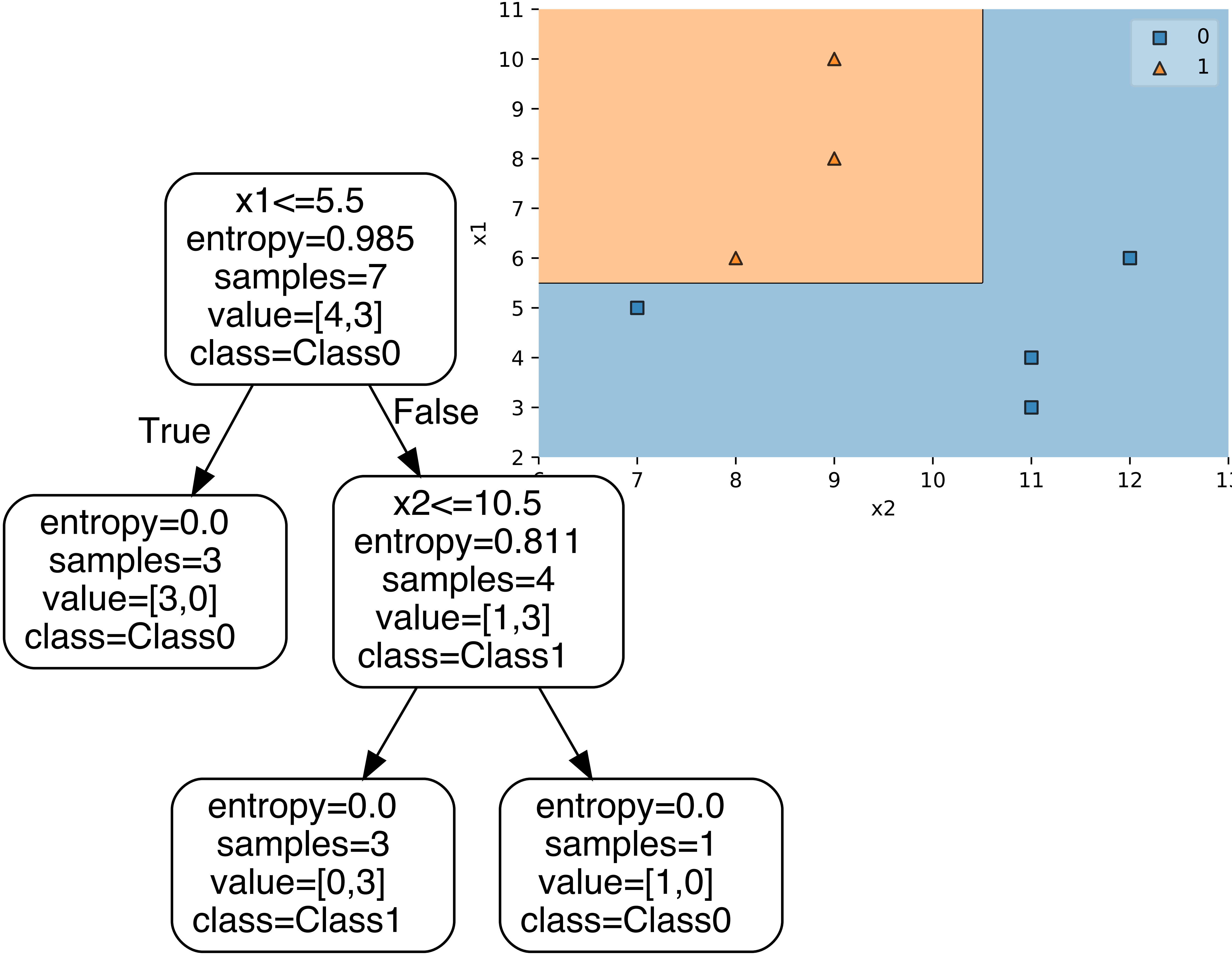
x_1	x_2	x_3	y
6 cm	8 cm	9 cm	1
4 cm	11 cm	2 cm	0
6 cm	12 cm	4 cm	0
10 cm	9 cm	3 cm	1
5 cm	7 cm	8 cm	0
8 cm	9 cm	3 cm	1
3 cm	11 cm	5 cm	0

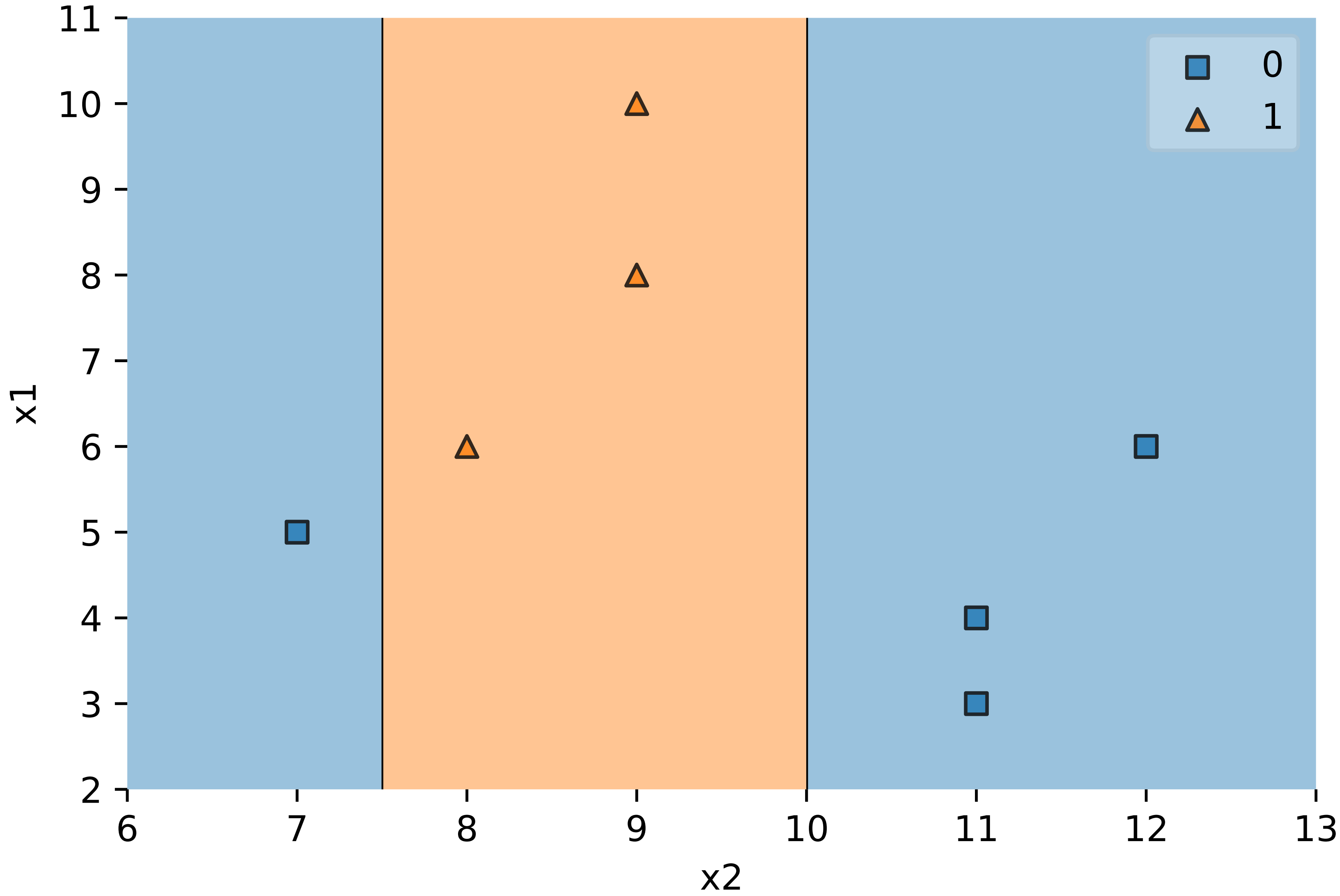
Drawing a Decision Boundary

x_1	x_2	x_3	y
6 cm	8 cm	9 cm	1
4 cm	11 cm	2 cm	0
6 cm	12 cm	4 cm	0
10 cm	9 cm	3 cm	1
5 cm	7 cm	8 cm	0
8 cm	9 cm	3 cm	1
3 cm	11 cm	5 cm	0

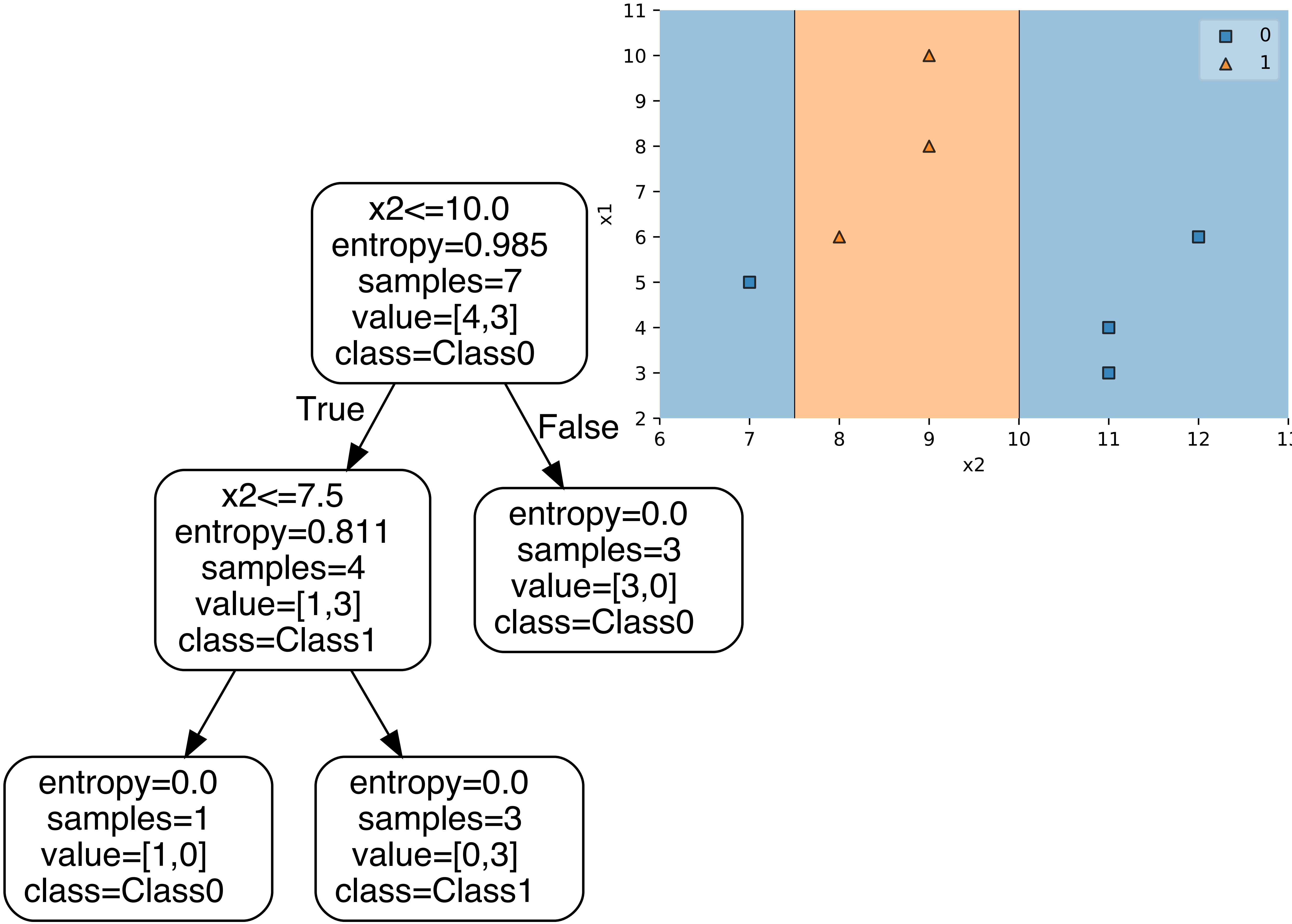



11
10
9
8
7
6
5
4
3
2
x1
6
7
8
9
10
11
12
13
x2
0
1


x1<=5.5
entropy=0.985
samples=7
value=[4,3]
class=Class0
True
False
entropy=0.0
samples=3
value=[3,0]
class=Class0
x2<=10.5
entropy=0.811
samples=4
value=[1,3]
class=Class1
entropy=0.0
samples=3
value=[0,3]
class=Class1
entropy=0.0
samples=1
value=[1,0]
class=Class0
0
1
x1
x2
2
3
4
5
6
7
8
9
10
11
7
8
9
10
11
12
13



11
10
9
8
7
6
5
4
3
2
x1
6
7
8
9
10
11
12
13
x2
0
1


x2<=10.0
entropy=0.985
samples=7
value=[4,3]
class=Class0
True
False
x2<=7.5
entropy=0.811
samples=4
value=[1,3]
class=Class1
entropy=0.0
samples=3
value=[3,0]
class=Class0
entropy=0.0
samples=1
value=[1,0]
class=Class0
entropy=0.0
samples=3
value=[0,3]
class=Class1
x1
x2
0
1

The Splitting Criterion

Information Gain

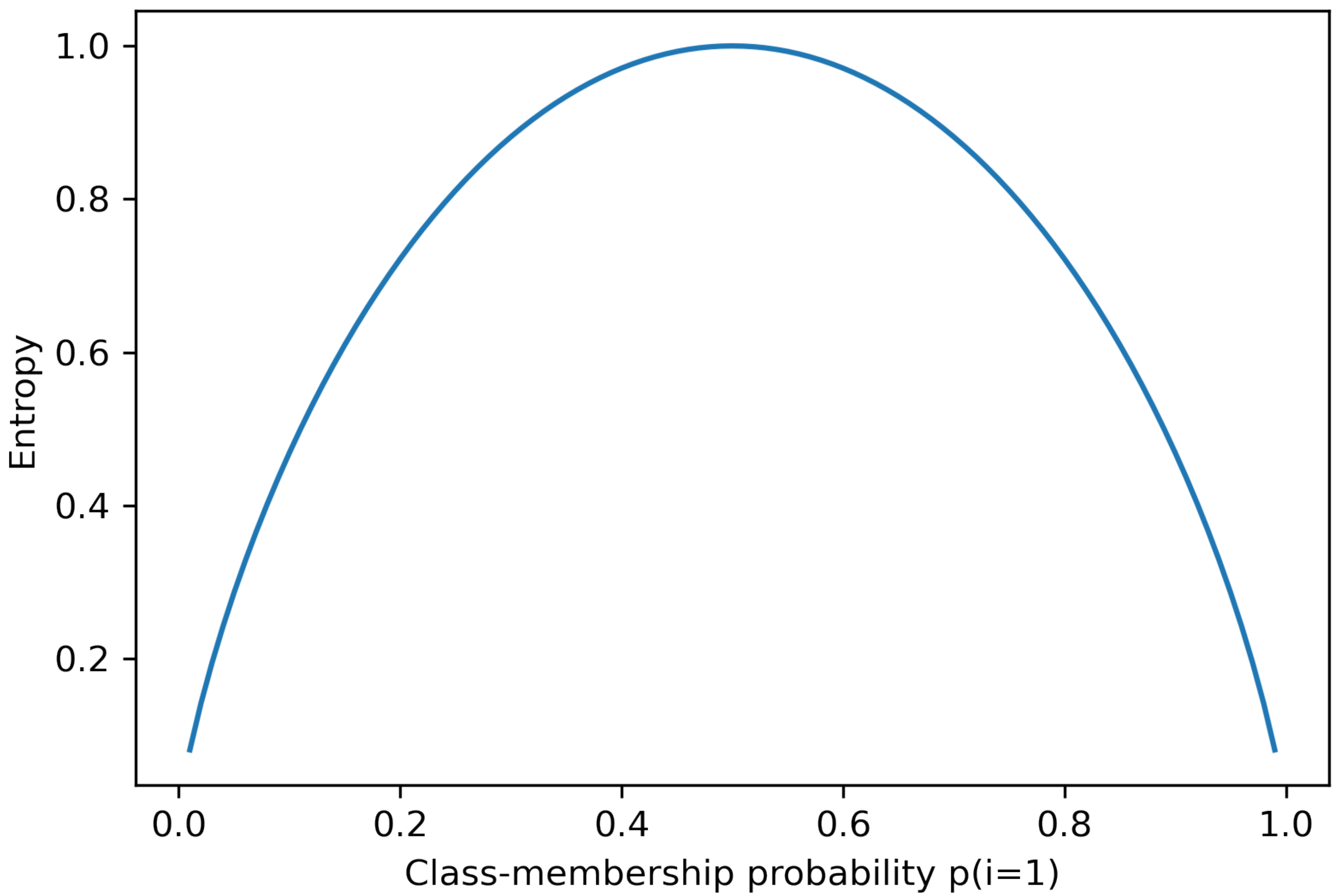
$$GAIN(\mathscr{D}, x_j) = H(\mathscr{D}) - \sum_{v \in Values(x_j)} \frac{|\mathscr{D}_v|}{|\mathscr{D}|} H(\mathscr{D}_v)$$

(Shannon) Entropy

$$H = -\sum_{i} p(i \mid x_j) \log_2 (p(i \mid x_j))$$

(Shannon) Entropy

$$H = -\sum_{i} p(i \mid x_j) \log_2(p(i \mid x_j))$$



Information Gain

$$GAIN(\mathscr{D}, x_j) = H(\mathscr{D}) - \sum_{v \in Values(x_j)} \frac{|\mathscr{D}_v|}{|\mathscr{D}|} H(\mathscr{D}_v)$$

Gini Impurity

$$Gini = 1 - \sum_i \left(p(i \mid x_j)^2\right)$$

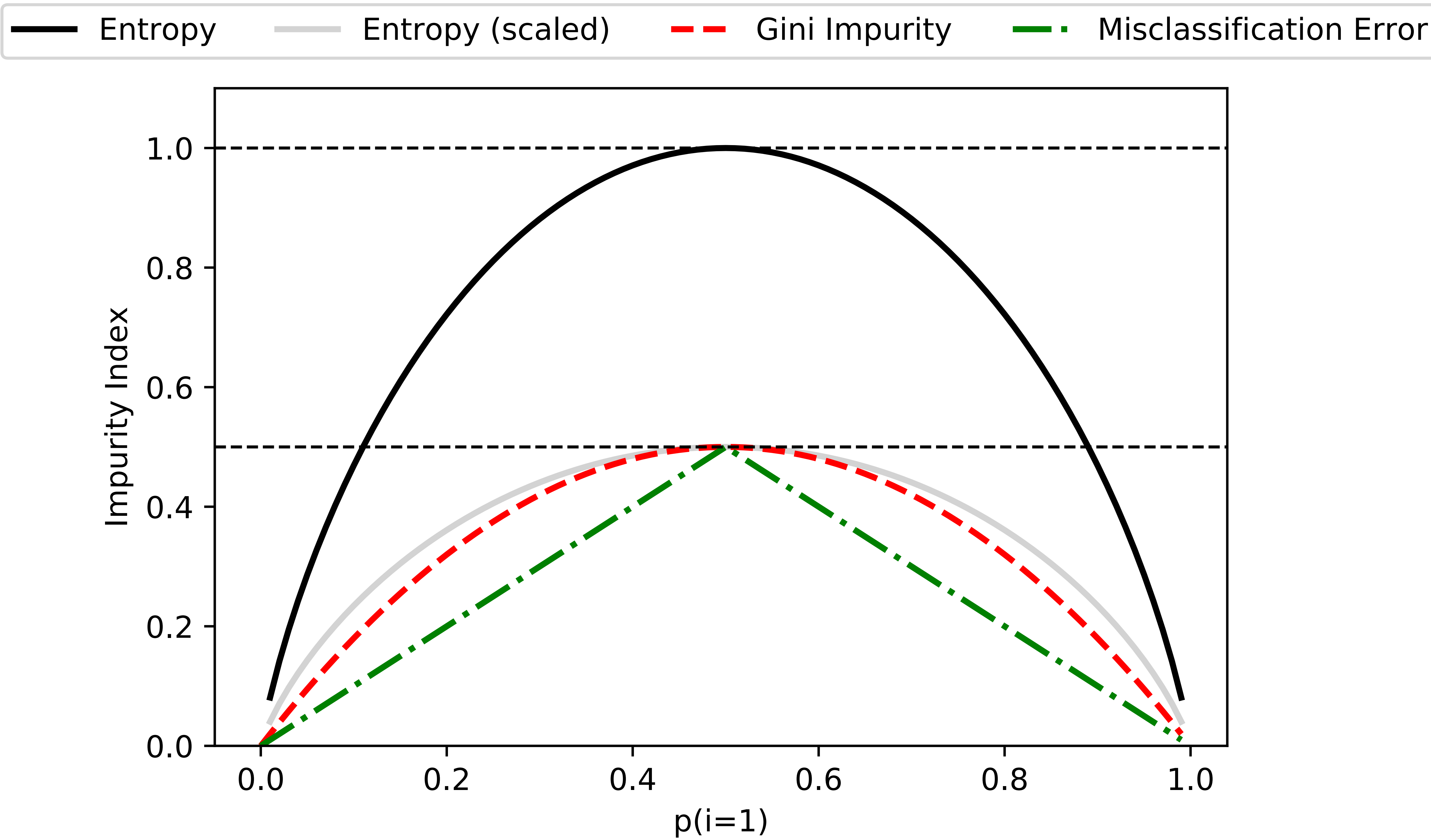
Misclassification Error

$$ERR = \frac{1}{n} \sum_{i=1}^{n} L(\hat{y}^{[i]}, y^{[i]}),$$

$$L(\hat{y}, y) = \begin{cases} 0 \textbf{ if } \hat{y} = y, \\ 1 \textbf{ otherwise.} \end{cases}$$

Misclassification Error

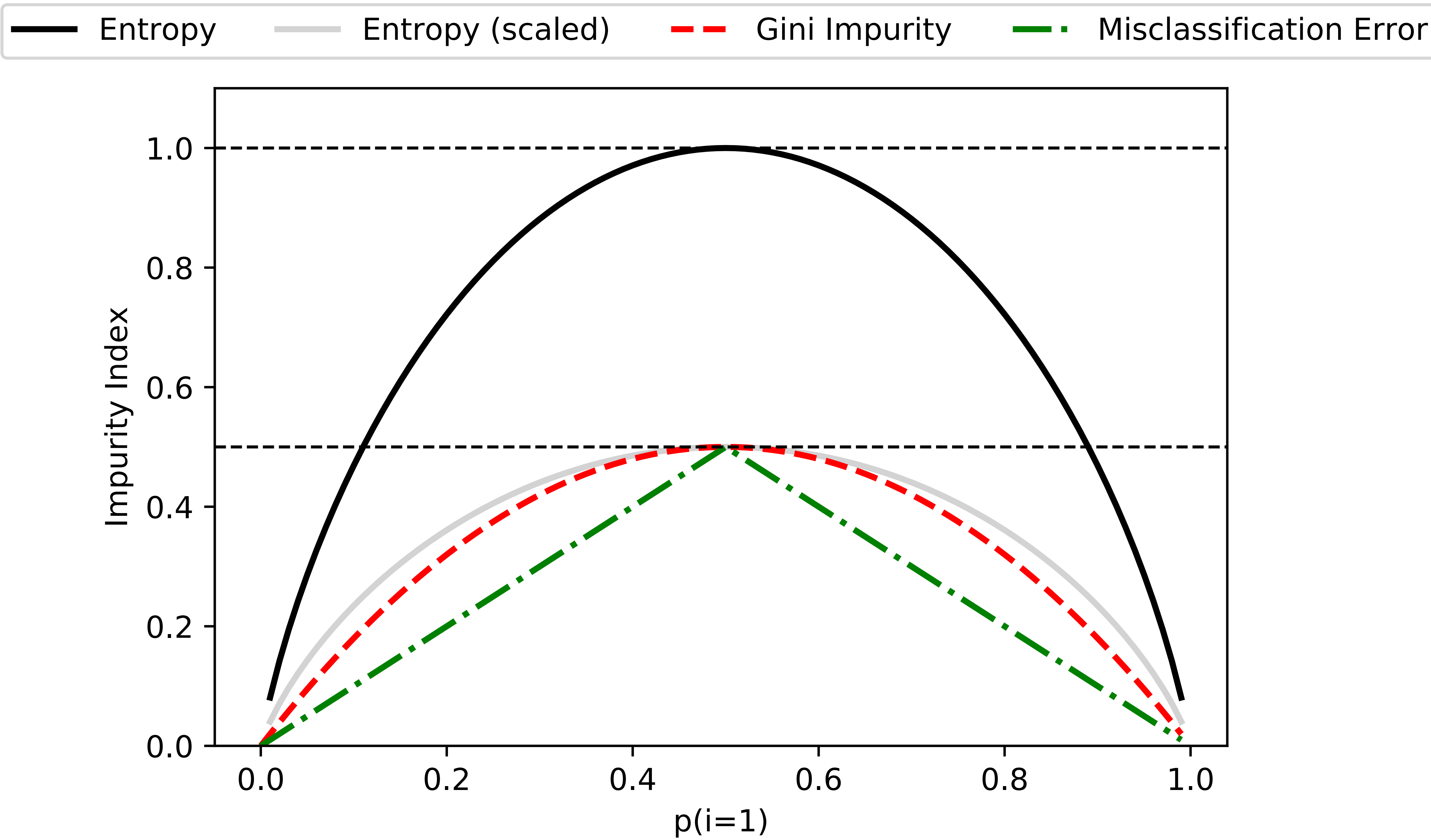
$$ERR = 1 - \max_{i}(p(i \mid x_j))$$



Entropy
Entropy (scaled)
Gini Impurity
Misclassification Error
Impurity Index
p(i=1)
0.0
0.2
0.4
0.6
0.8
1.0

Topics

1. Intro to decision trees
2. Recursive algorithms & Big-O
3. Types of decision trees
4. Splitting criteria
5. **Gini & Entropy vs misclassification error**
6. Improvements & dealing with overfitting
7. Code example



Entropy
Entropy (scaled)
Gini Impurity
Misclassification Error
1.0
0.8
0.6
0.4
0.2
0.0
Impurity Index
0.0
0.2
0.4
0.6
0.8
1.0
p(i=1)

Why Growing Decision Trees via Entropy instead of Misclassification Error?

Why Growing Decision Trees via Entropy instead of Misclassification Error?

$$GAIN(\mathscr{D}, x_j) = I(\mathscr{D}) - \sum_{v \in Values(x_j)} \frac{|\mathscr{D}_v|}{|\mathscr{D}|} I(\mathscr{D}_v)$$

Entropy

$$H = -\sum_{i} p(i \mid x_j)\log_2(p(i \mid x_j))$$

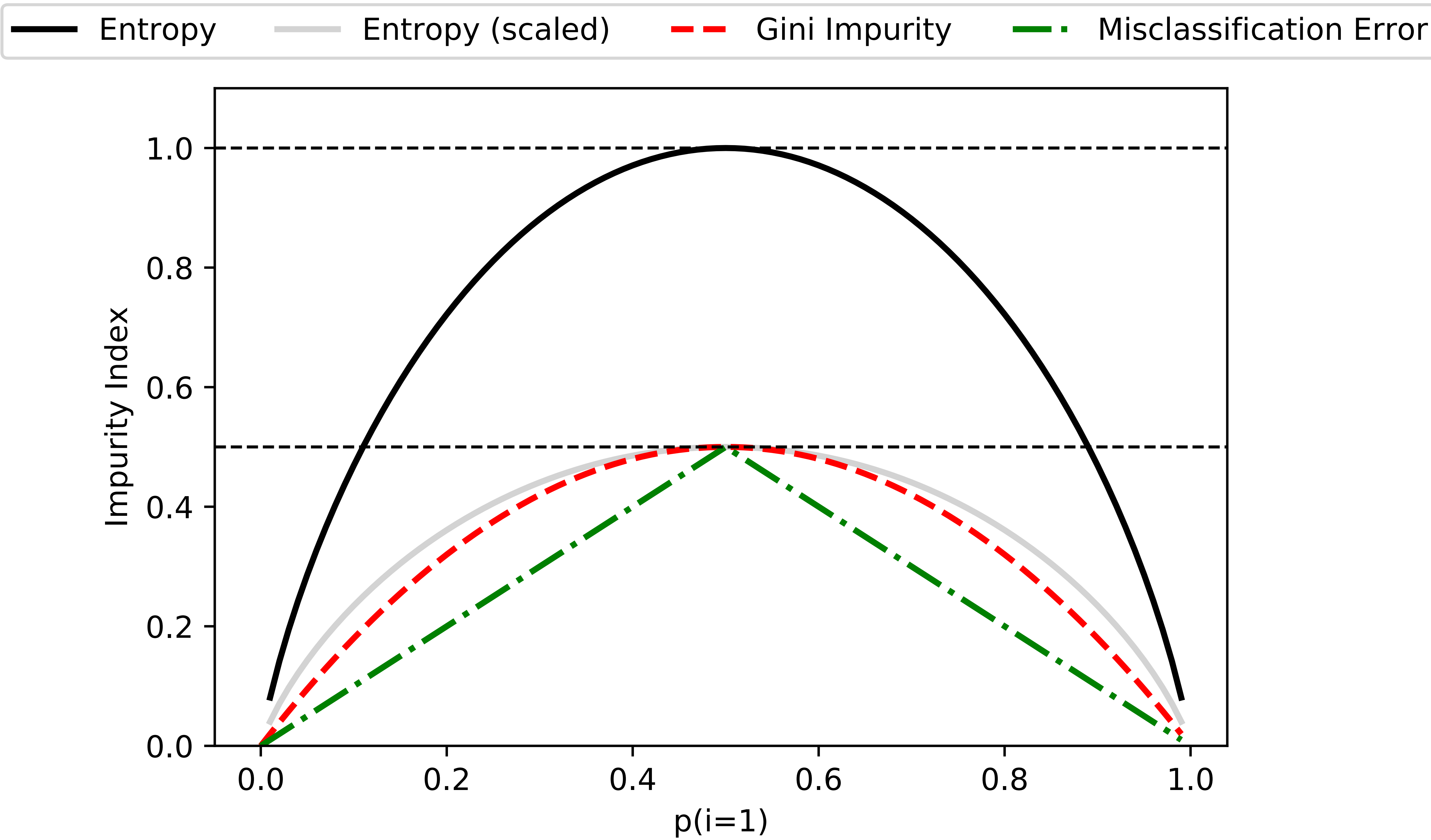
Gini Impurity

$$Gini = 1 - \sum_{i} \left(p(i \mid x_j)^2 \right)$$

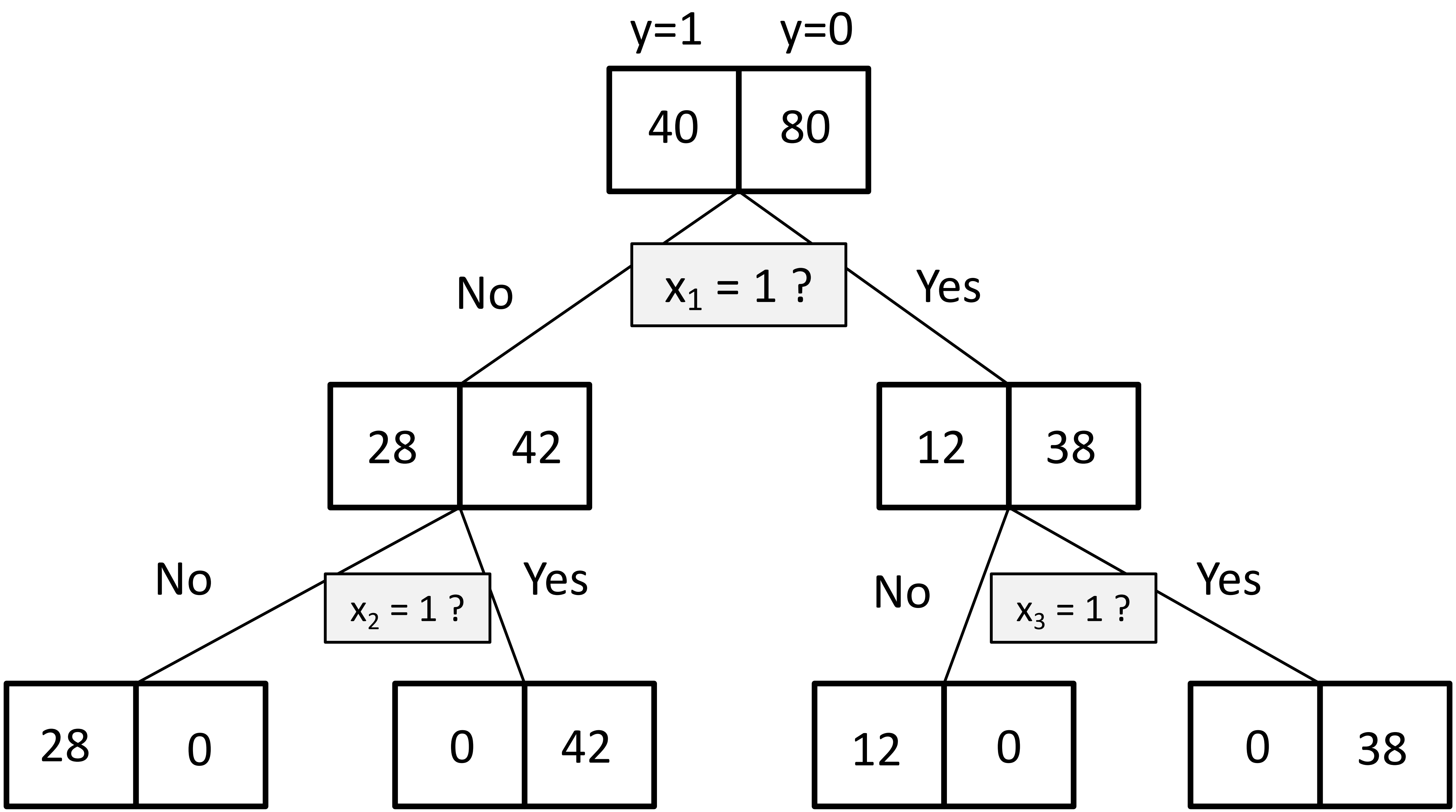
Misclassification Error

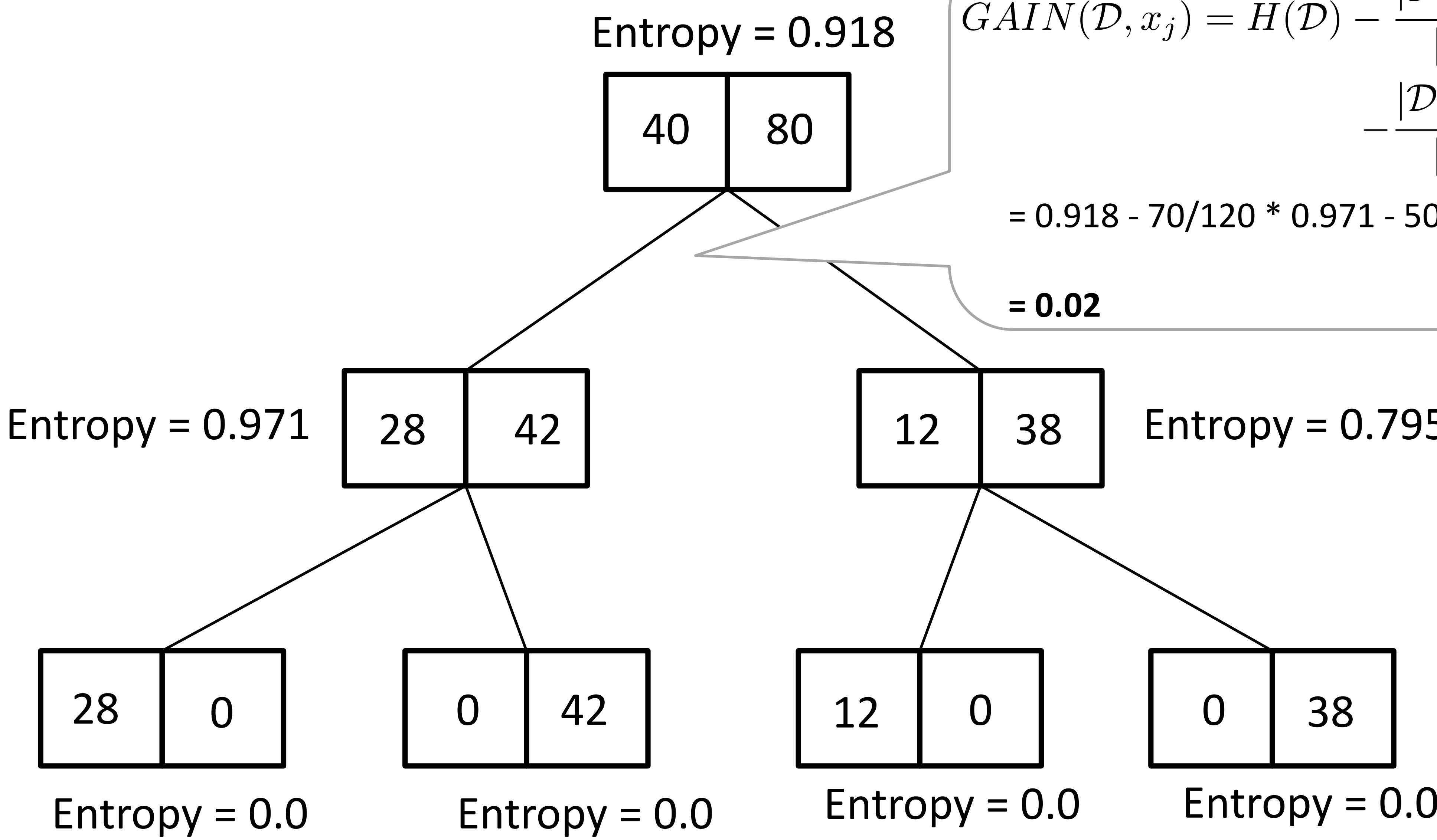
$$ERR = \frac{1}{n} \sum_{i=1}^{n} L(\hat{y}^{[i]}, y^{[i]}),$$

$$L(\hat{y}, y) = \begin{cases} 0 \textbf{ if } \hat{y} = y, \\ 1 \textbf{ otherwise.} \end{cases}$$



Entropy
Entropy (scaled)
Gini Impurity
Misclassification Error
Impurity Index
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0
p(i=1)

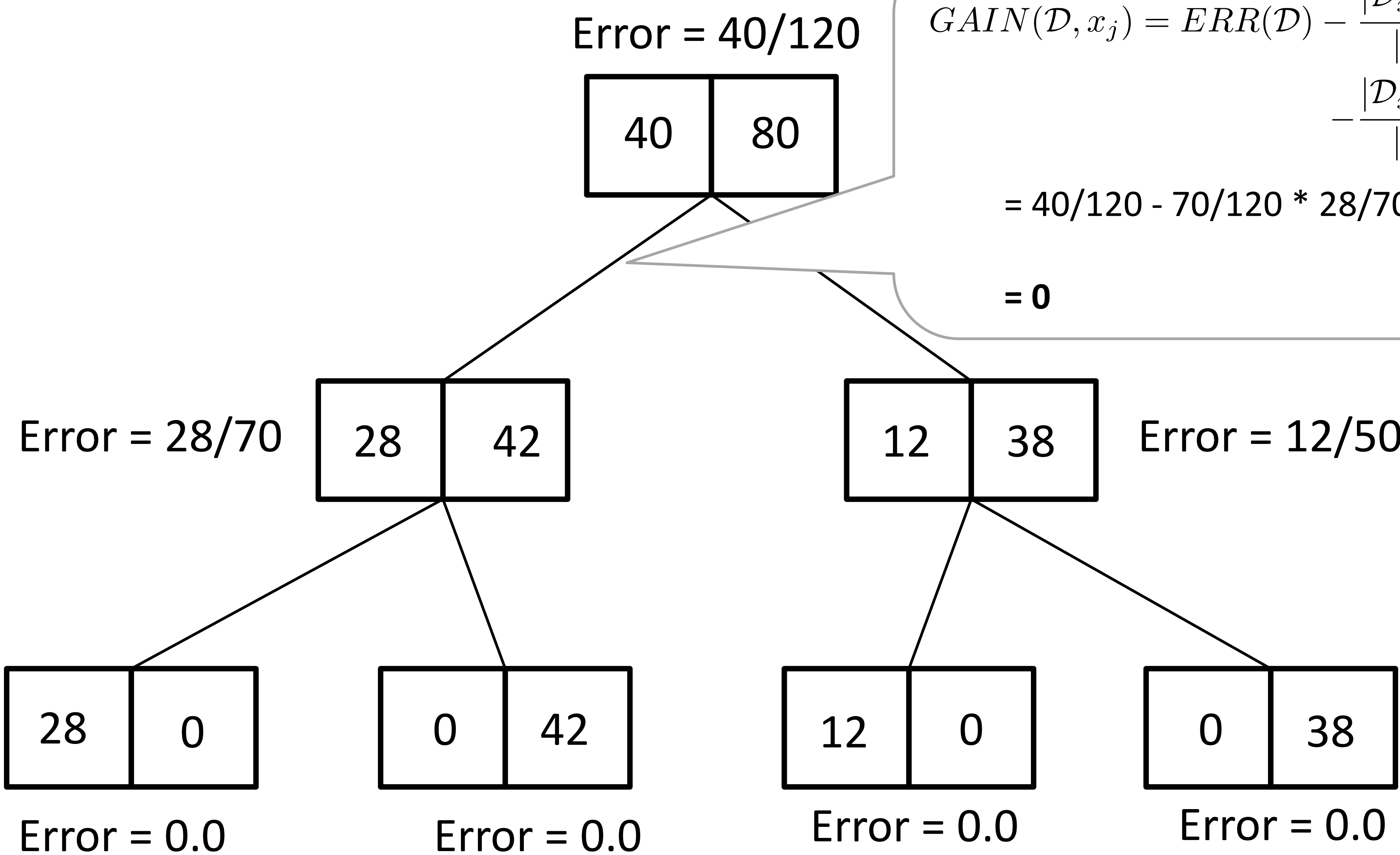

y=1
y=0
40
80
x_1 = 1 ?
No
Yes
28
42
12
38
x_2 = 1 ?
No
Yes
x_3 = 1 ?
No
Yes
28
0
0
42
12
0
0
38



$$GAIN(\mathcal{D}, x_j) = H(\mathcal{D}) - \frac{|\mathcal{D}_{x_j=1}|}{|\mathcal{D}|} H(\mathcal{D}_{x_j=1}) - \frac{|\mathcal{D}_{x_j=0}|}{|\mathcal{D}|} H(\mathcal{D}_{x_j=0})$$

= 0.918 - 70/120 * 0.971 - 50/120 * 0.795

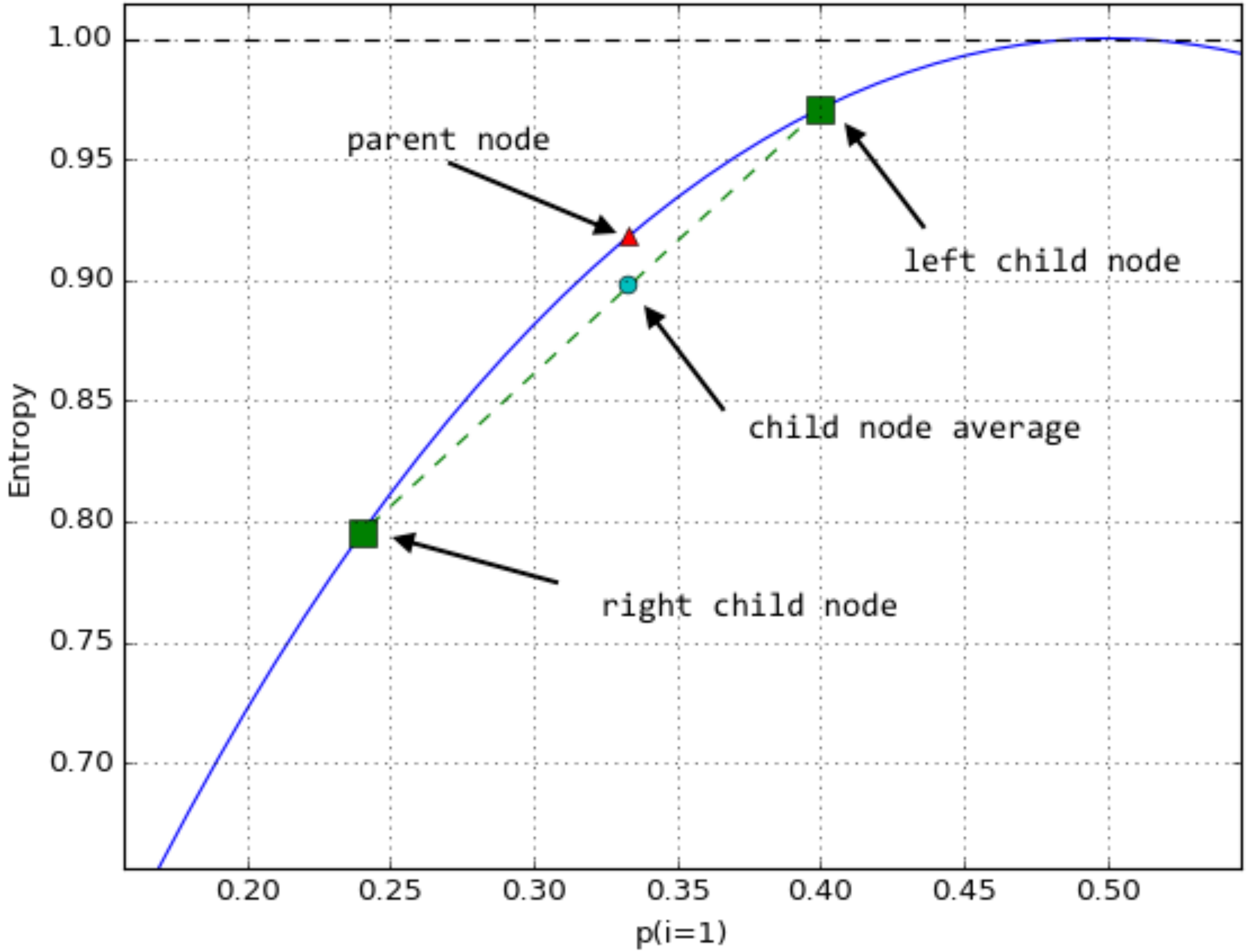
= 0.02



$$
\begin{aligned}
GAIN(\mathcal{D}, x_j) = ERR(\mathcal{D}) &- \frac{|\mathcal{D}_{x_j=1}|}{|\mathcal{D}|} ERR(\mathcal{D}_{x_j=1}) \\
&- \frac{|\mathcal{D}_{x_j=0}|}{|\mathcal{D}|} ERR(\mathcal{D}_{x_j=0})
\end{aligned}
$$

= 40/120 - 70/120 * 28/70 - 50/120 * 12/50

= 0



1.00
0.95
0.90
0.85
0.80
0.75
0.70
Entropy
0.20
0.25
0.30
0.35
0.40
0.45
0.50
p(i=1)
parent node
left child node
child node average
right child node

Lecture 6: Decision Trees

Topics

1. Intro to decision trees

2. Recursive algorithms & Big-O

3. Types of decision trees

4. Splitting criteria

5. Gini & Entropy vs misclassification error

6. Improvements & dealing with overfitting

7. Code example

Gain Ratio

$$GainRatio(\mathcal{D}, x_j) = \frac{Gain(\mathcal{D}, x_j)}{SplitInfo(\mathcal{D}, x_j)}$$

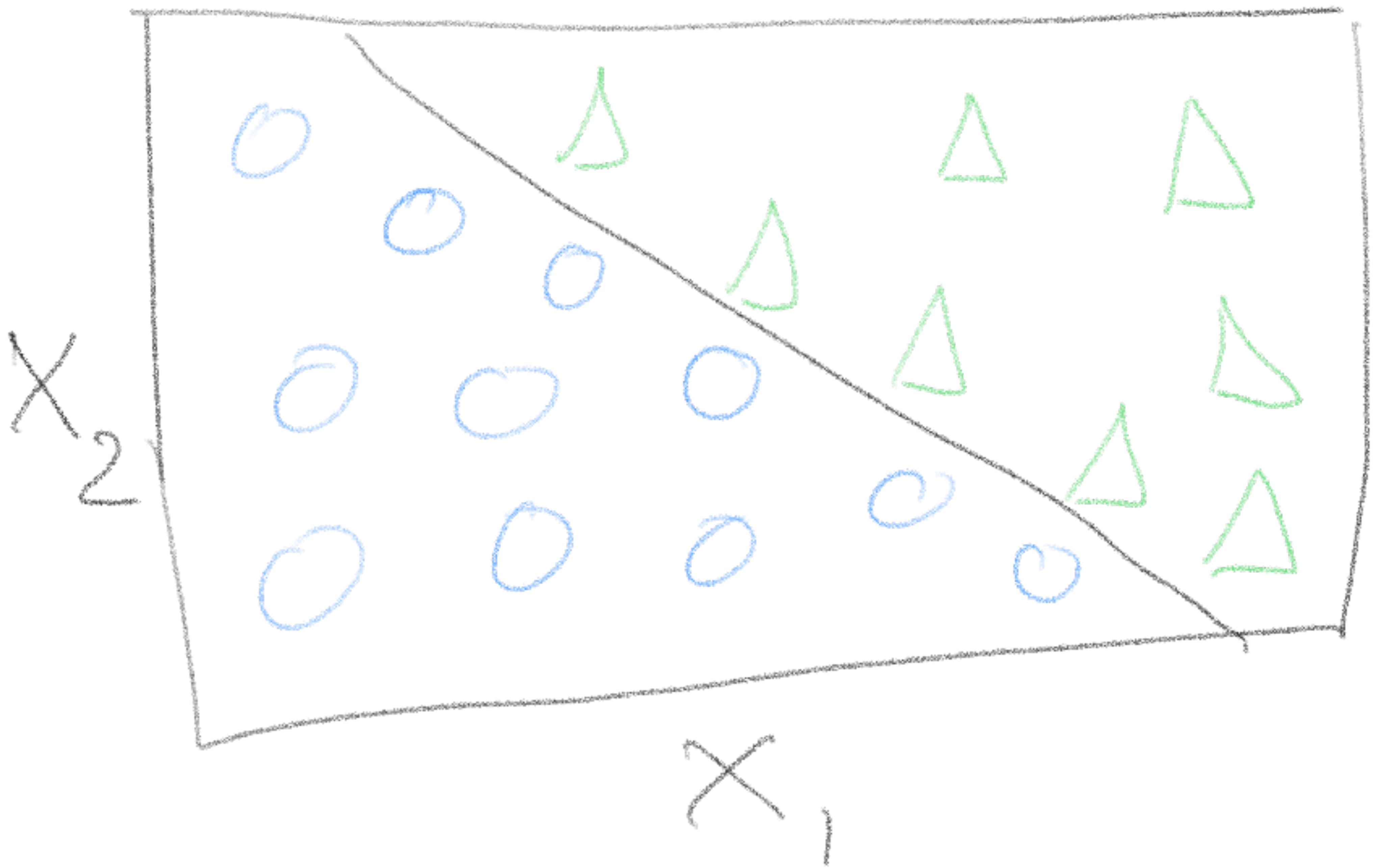
Quinlan 1986

where the split information measures the entropy of the feature:

$$SplitInfo(\mathcal{D}, x_j) = -\sum_{v \in x_j} \frac{|\mathcal{D}_v|}{|\mathcal{D}|} \log_2 \frac{|\mathcal{D}_v|}{|\mathcal{D}|}$$

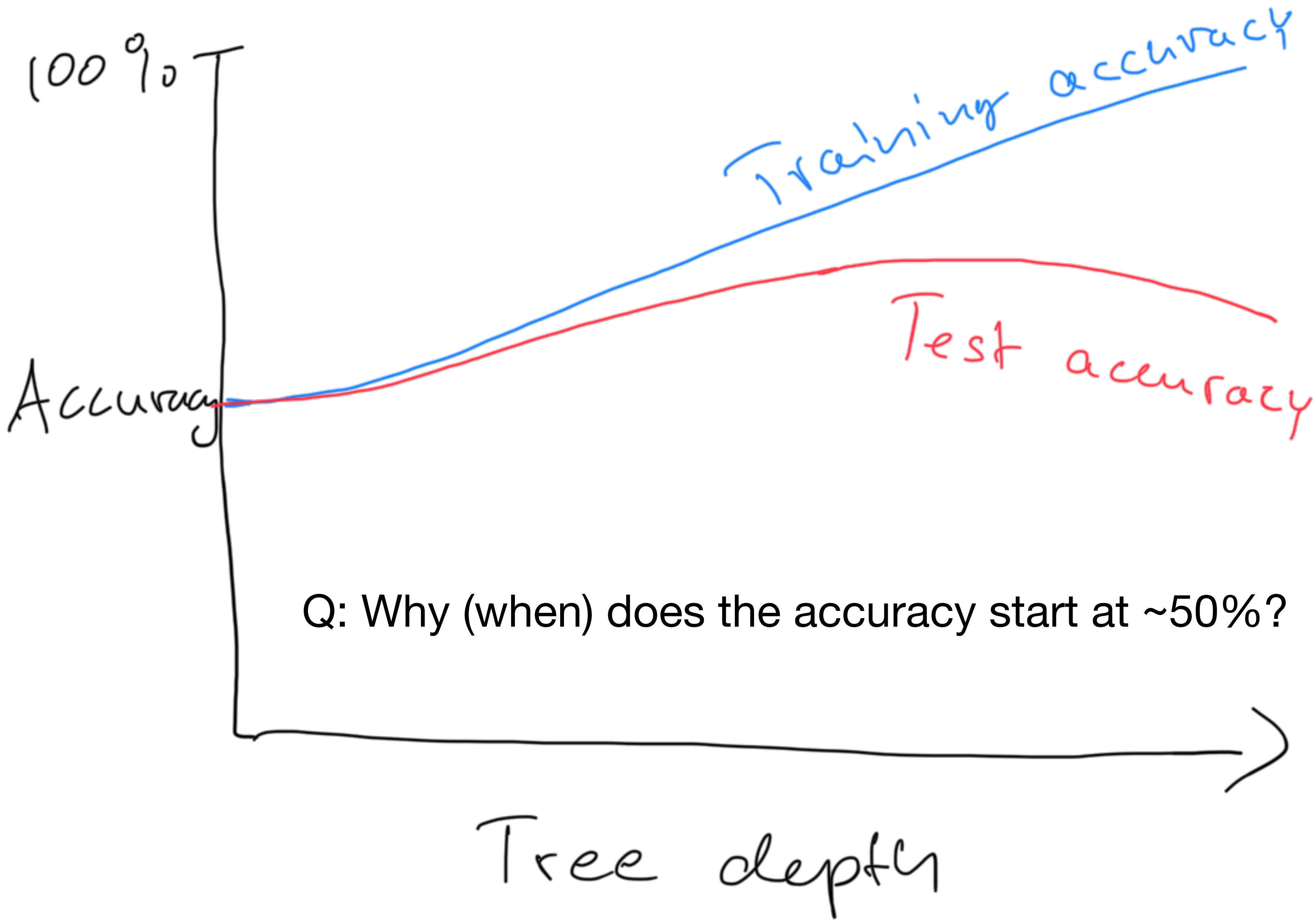
Penalizes splitting categorical attributes with many values (e.g., think date column, or really bad: row ID) via the split information

Shortcomings



How would the decision tree split look like?

Overfitting



Q: Why (when) does the accuracy start at ~50%?

Pre-Pruning

- Set a depth cut-off (maximum tree depth) *a priori*

- Cost-complexity pruning: , where is an impurity measure, is a tuning parameter, and is the total number of nodes.

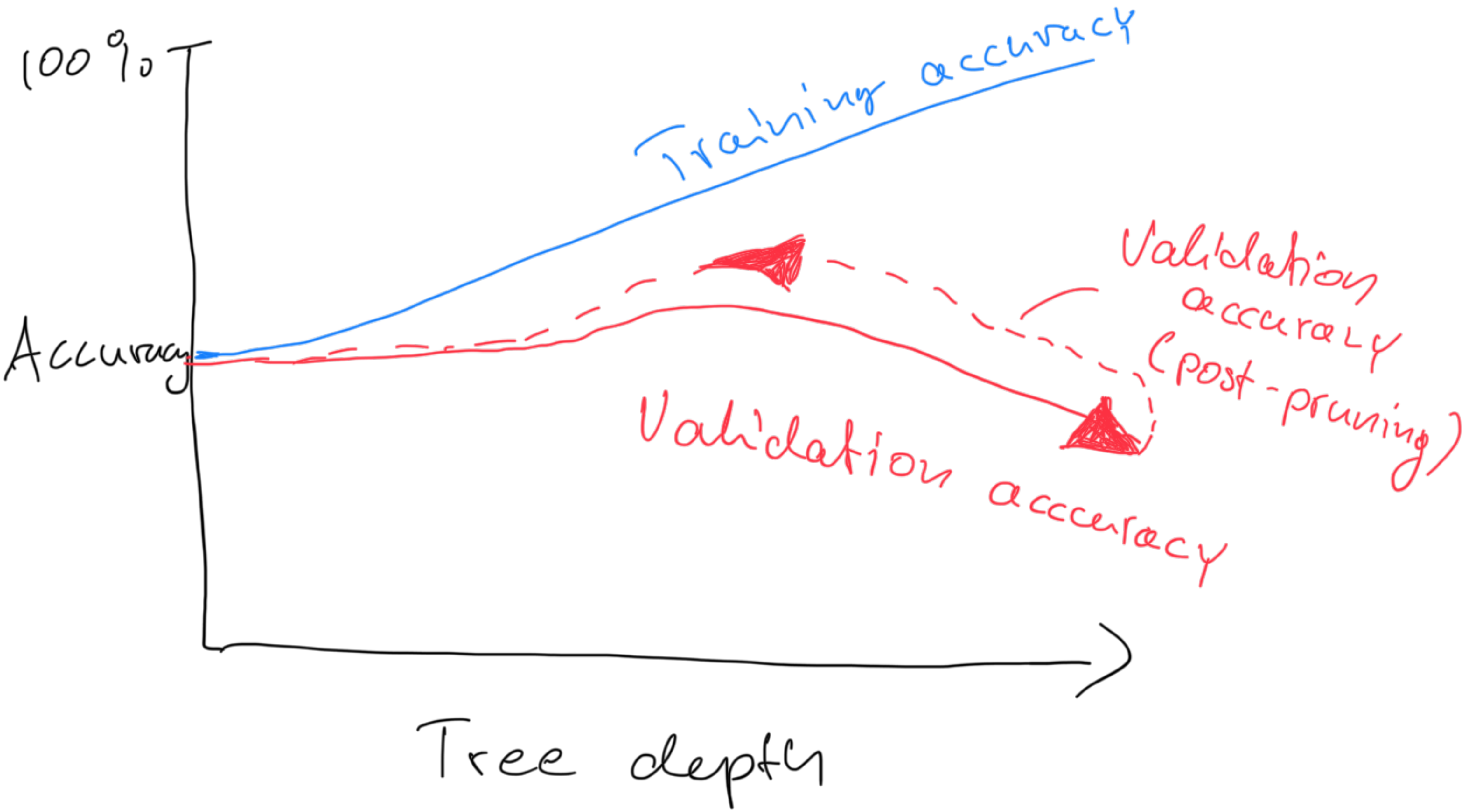
- Stop growing if split is not statistically significant (e.g., χ^2 test)

- Set a minimum number of data points for each node

Post-Pruning

- Grow full tree first, then remove nodes, in C4.5
- Reduced-error pruning, remove nodes via validation set eval. (problematic for limited data)
- Can also convert trees to rules first and then prune the rules

Post-Pruning



Regression Trees

Decision Tree Summary: Pros and Cons

- (+) Easy to interpret and communicate
- (+) Can represent "complete" hypothesis space
- (-) Easy to overfit
- (-) Elaborate pruning required
- (-) Expensive to just fit a "diagonal line"
- (-) Output range is bounded (dep. on training examples) in regression trees

Topics

1. Intro to decision trees

2. Recursive algorithms & Big-O

3. Types of decision trees

4. Splitting criteria

5. Gini & Entropy vs misclassification error

6. Improvements & dealing with overfitting

7. Code example

Demo