Updated 10/22/2025

Part 3: Applications
L12: Decision trees (1/2)

Wolfgang Gatterbauer

cs7840 Foundations and Applications of Information Theory (fa25)

https://northeastern-datalab.github.io/cs7840/fa25/

10/20/2025

Pre-class conversations

- Last class recapitulation
- Piazza: is it cumbersome to find comments on your scribes? Any suggestion for a better system?
- Projects: I have time to talk today after class and also THU

- Today:
  - Decision trees
  - backed in: Occam, MDL, fun connections to topics you have seen before here or elsewhere

Decision trees

Formal setup

EXAMPLE: Classifying days based on weather conditions.

Index (not part of the "data")

day	(O)utlook	(T)emp.	(H)umidity	(W)ind	(C)lass
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

- What are the features (predictors)?
- What is the target concept (the response variable)?

?

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Formal setup

EXAMPLE: Classifying days based on weather conditions.

Class label y_i denotes weather a particular event happened (here whether tennis game happened)

Columns denote $m = 4$ features $\{X_j\}_{j=1}^{m}$.

Domain $\mathcal{X}_H$ of feature X_H is {high, normal}

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(C)lass**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

$\langle \mathbf{x}_4, y_4 \rangle$

Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$.

- Concrete Problem Setting in this Example:

  ?

- Input: ?

- Output: ?

Formal setup

EXAMPLE: Classifying days based on weather conditions.

Class label y_i denotes weather a particular event happened (here whether tennis game happened)

Columns denote $m = 4$ features $\{X_j\}_{j=1}^{m}$.

Domain $\mathcal{X}_H$ of feature X_H is {high, normal}

	Predictors				Response
day	(O)utlook	(T)emp.	(H)umidity	(W)ind	(C)lass
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

$\langle \mathbf{x}_4, y_4 \rangle$

Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$.

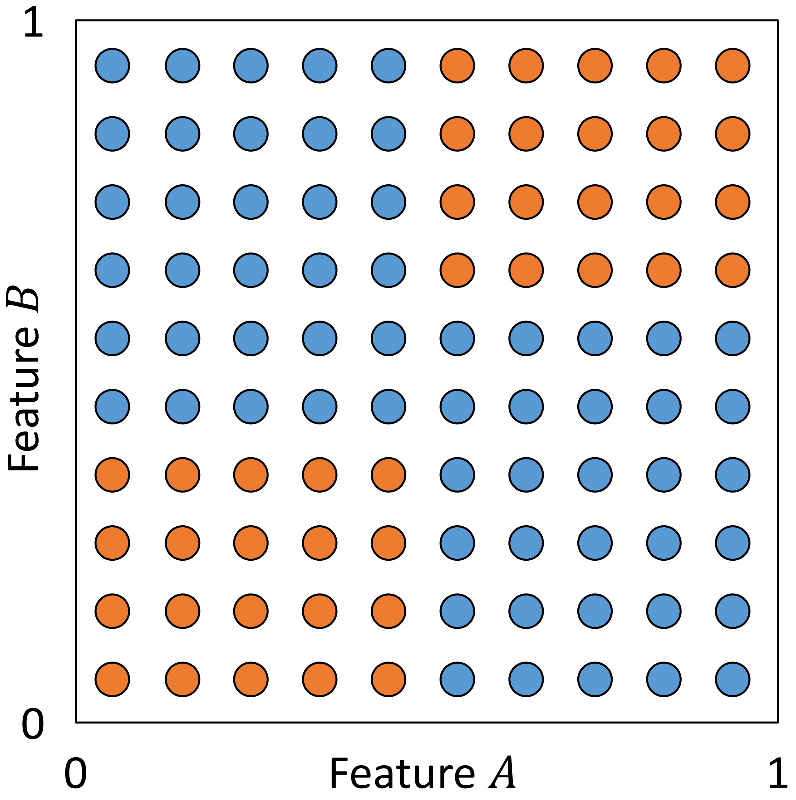
- Concrete Problem Setting in this Example:
  - Set of possible instances $\mathcal{X} = \mathcal{X}_O \times \mathcal{X}_T \times \mathcal{X}_H \times \mathcal{X}_W$
  - Set of possible labels $\mathcal{Y} = \{\text{yes, no}\}$ with size $k = |\mathcal{Y}| = 2$ (binary)
  - Unknown target function $f: \mathcal{X} \rightarrow \mathcal{Y}$
  - Set of function hypotheses $H = \{h \mid h: \mathcal{X} \rightarrow \mathcal{Y}\}$
- **Input:** n training examples of unknown target function f

$$\{\langle \mathbf{x}_i, y_i \rangle\}_{i=1}^{n} = \{\langle \mathbf{x}_1, y_1 \rangle, \dots, \langle \mathbf{x}_n, y_n \rangle\}$$

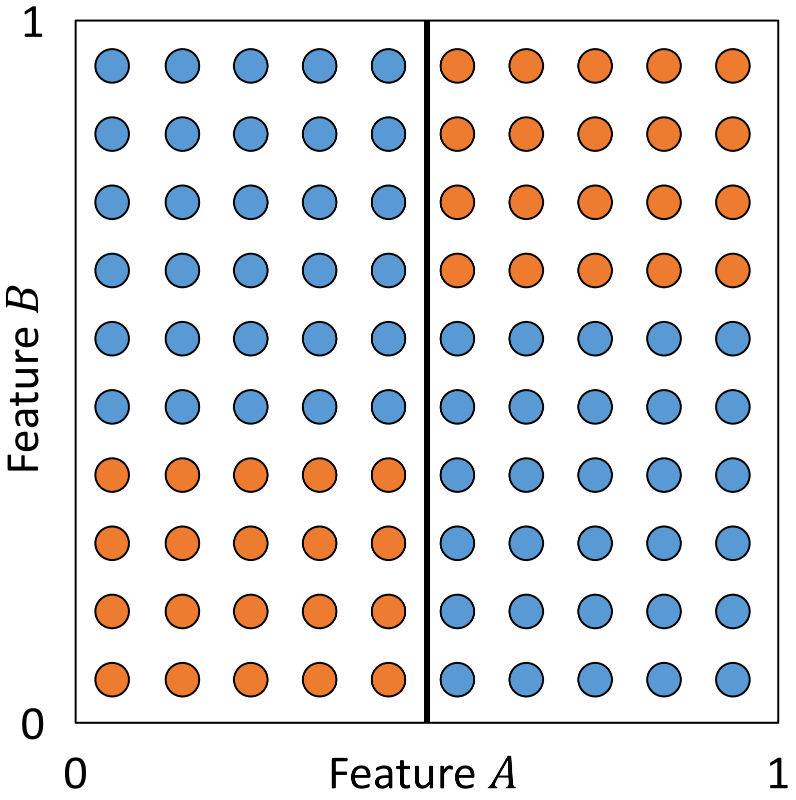
- **Output:** Hypothesis $h \in H$ that best approximates f

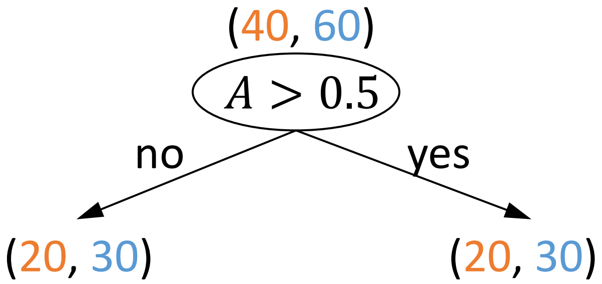
Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Decision Tree Classification



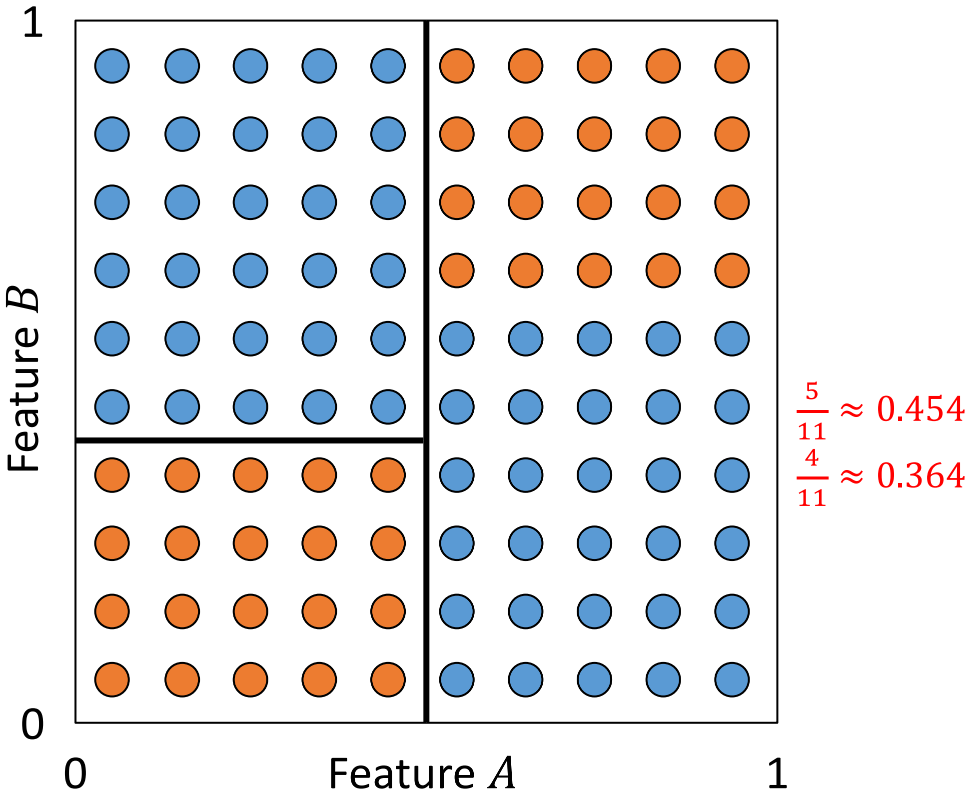
Decision Tree Classification

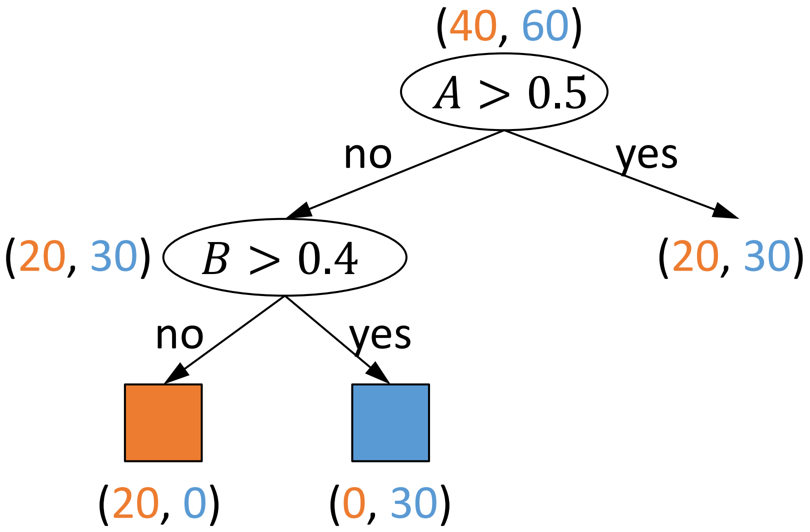




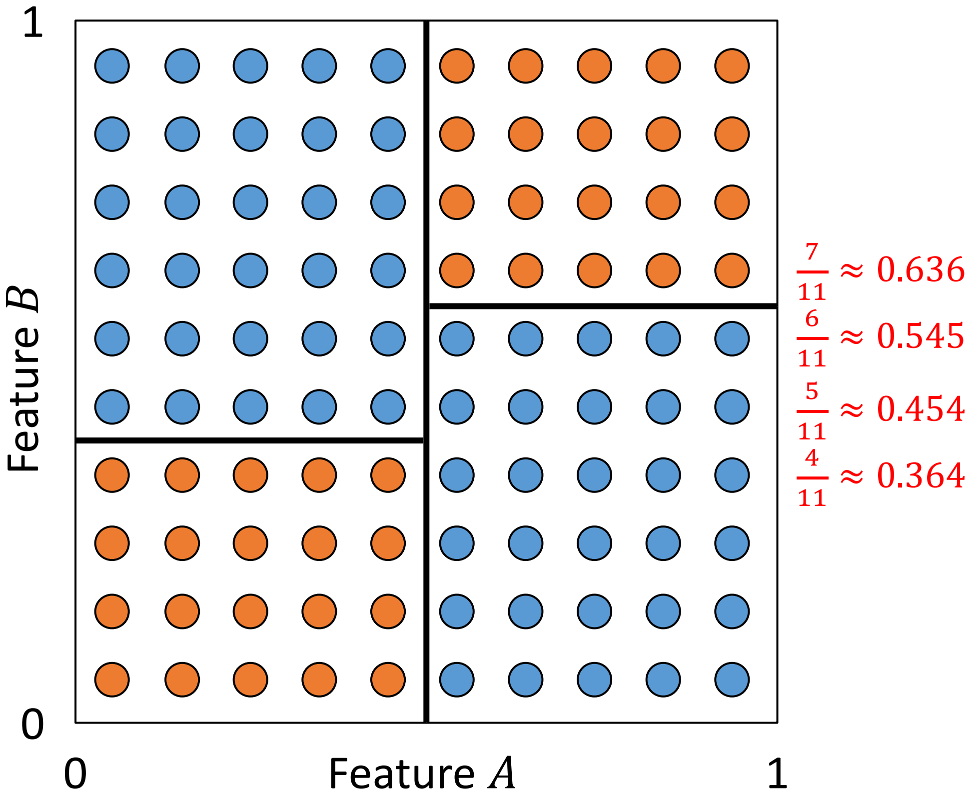
(This is not what a practical algorithm with do. We come back to that later.)

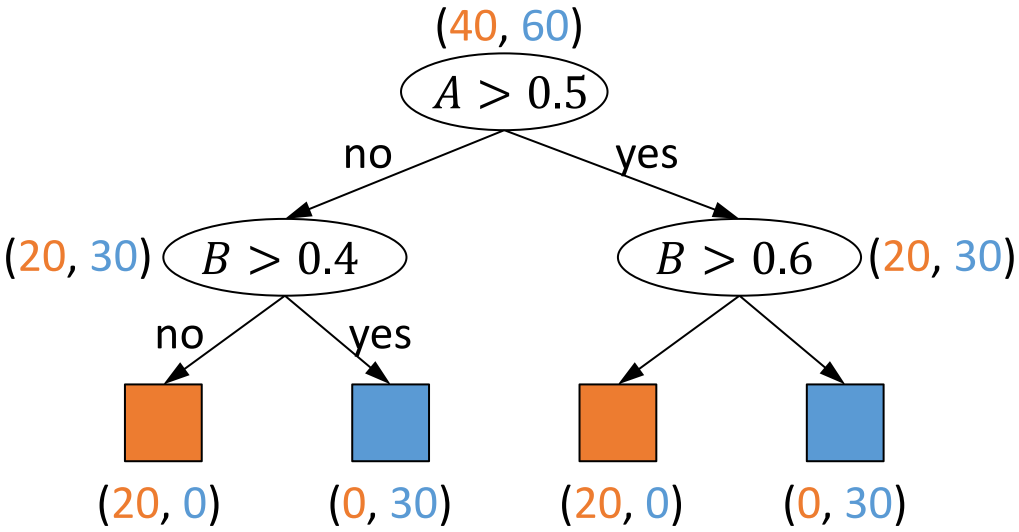
Decision Tree Classification



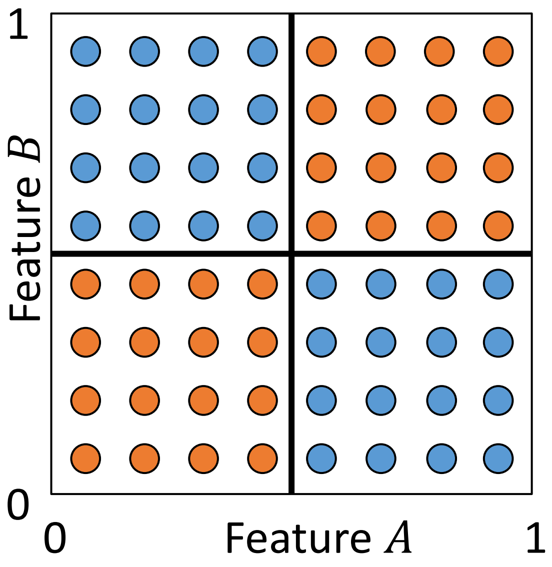


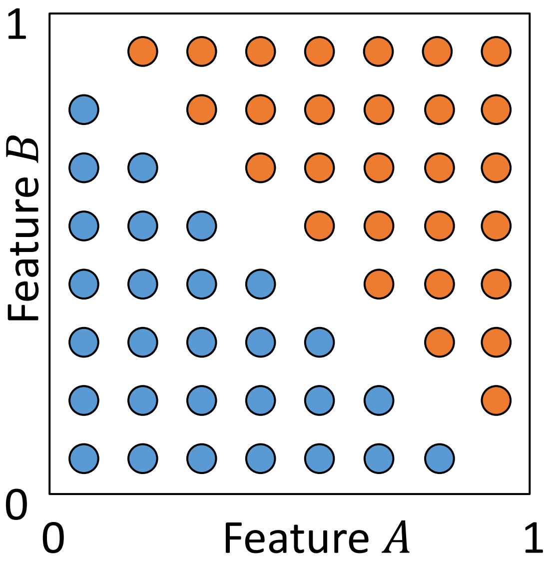
Decision Tree Classification

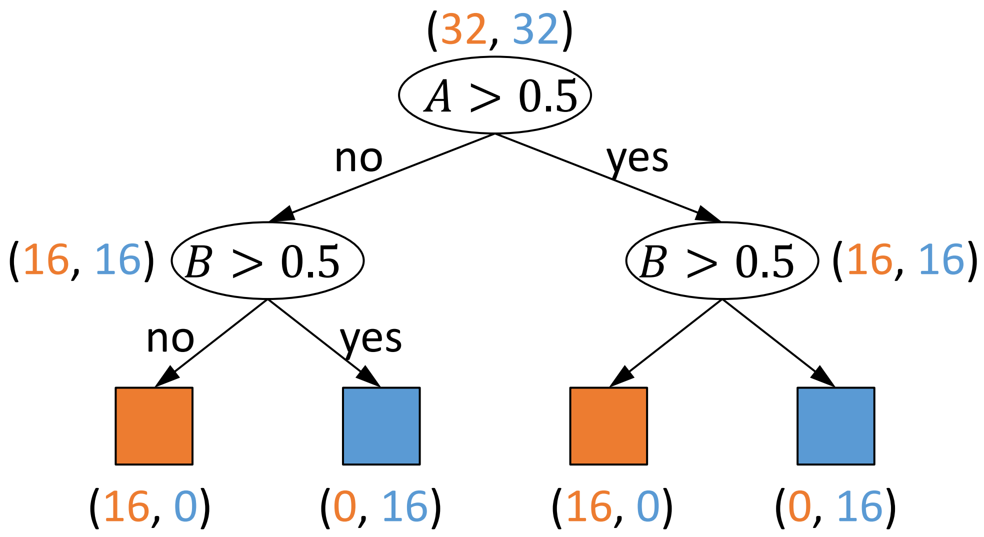




Rectilinear decision boundaries

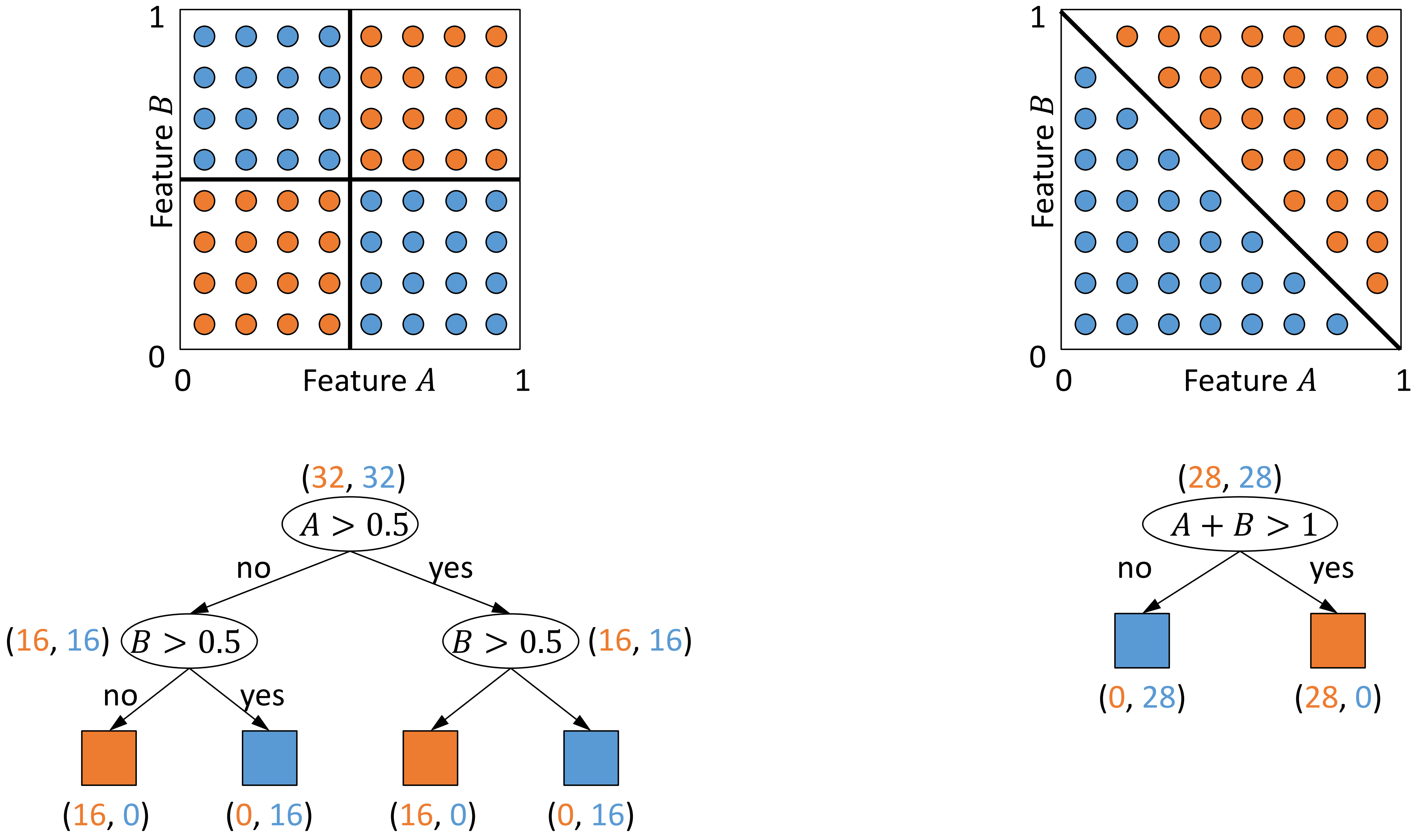






?

Rectilinear vs. oblique decision boundaries



Rectilinear vs. oblique decision boundaries

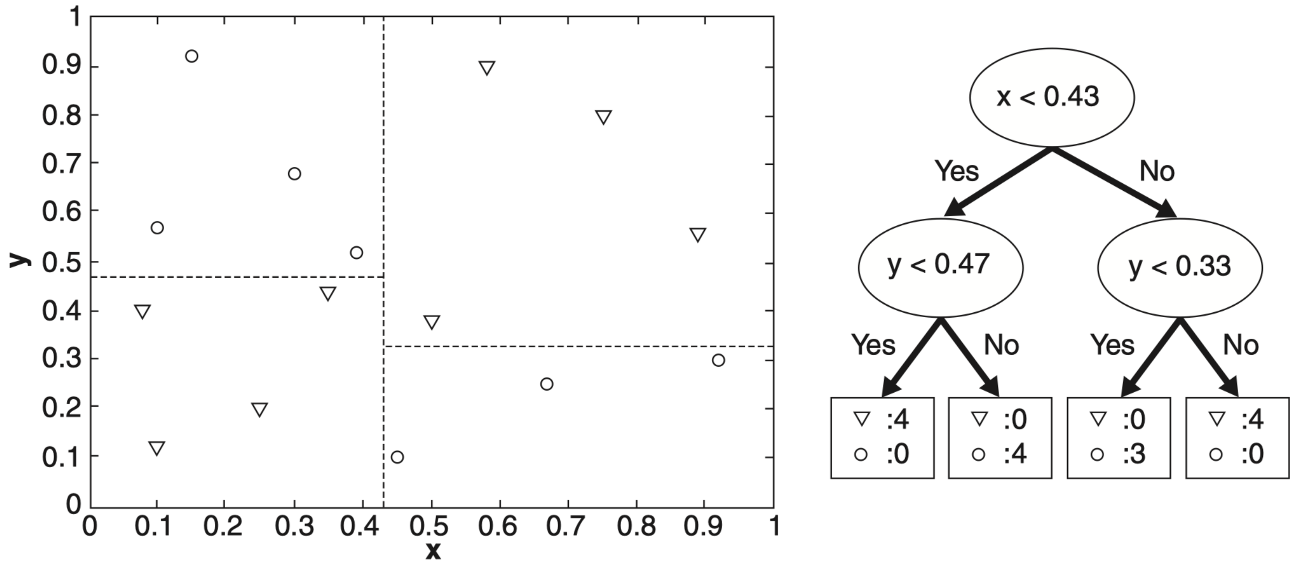


Figure 3.20. Example of a decision tree and its decision boundaries for a two-dimensional data set.

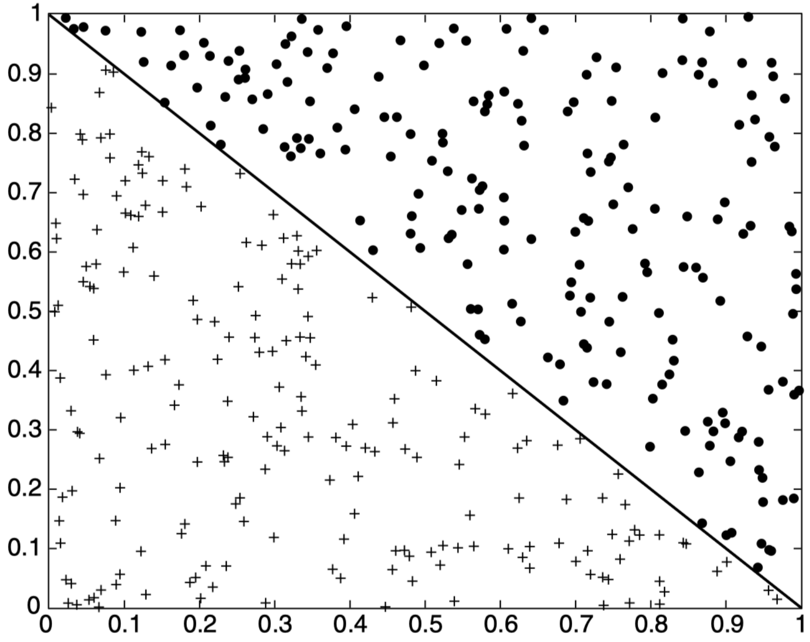


Figure 4.21. Example of data set that cannot be partitioned optimally using test conditions involving single attributes.

[Tan+'18] Tan, Steinbach, Kumar. Introduction to Data Mining, 2nd ed, 2018. https://www-users.cse.umn.edu/~kumar001/dmbook/index.php

Rectilinear vs. oblique decision boundaries

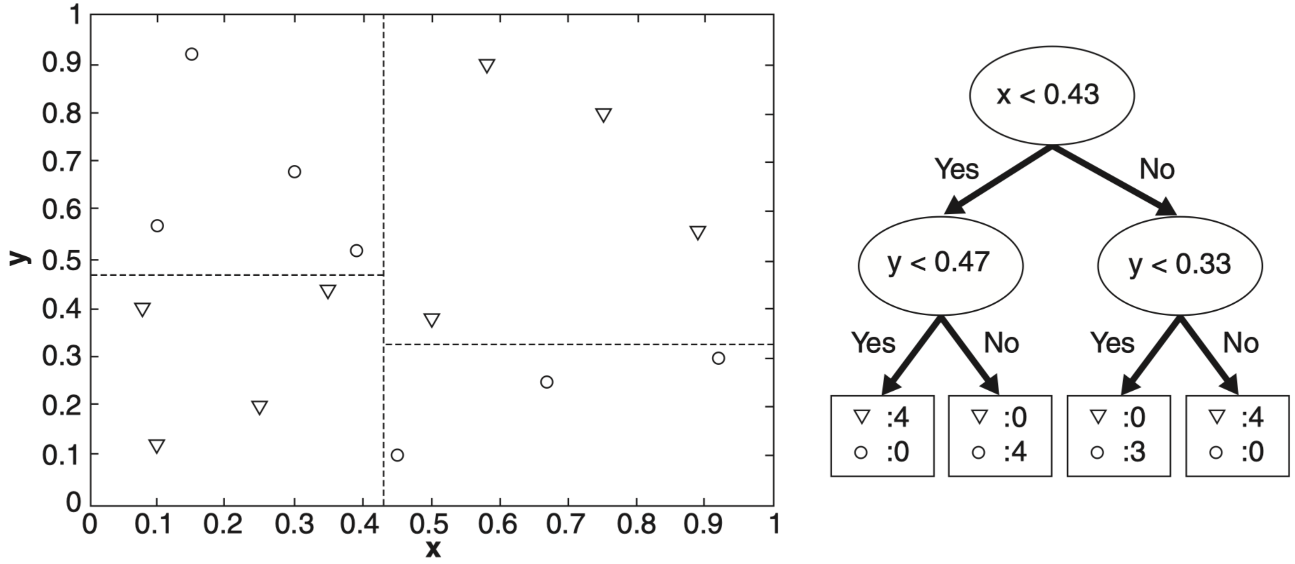


Figure 3.20. Example of a decision tree and its decision boundaries for a two-dimensional data set.

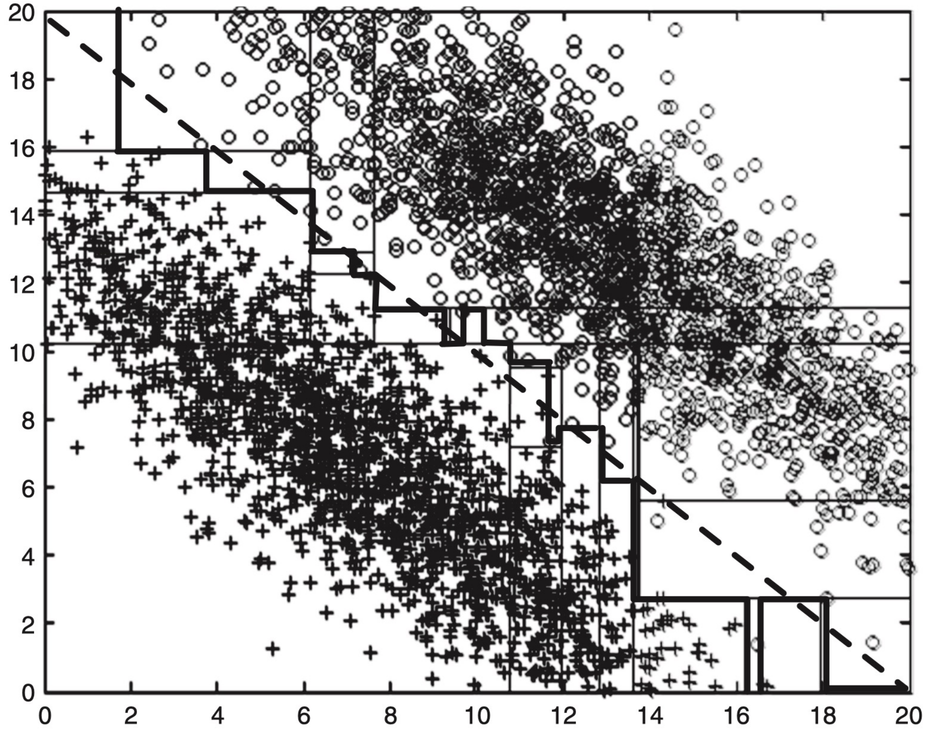


Figure 3.21. Example of data set that cannot be partitioned optimally using a decision tree with single attribute test conditions. The true decision boundary is shown by the dashed line.

[Tan+'18] Tan, Steinbach, Kumar. Introduction to Data Mining, 2nd ed, 2018. https://www-users.cse.umn.edu/~kumar001/dmbook/index.php
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Decision Trees

- Decision trees are a popular technique for both classification (categorical values) and regression (numerical values) tasks.
- During training (when developing the tree):
  - The training data is recursively partitioned into smaller yet increasingly homogenous regions
  - Each internal node tests an attribute to direct the example toward a leaf, which predicts the majority class (or mean value) of the examples contained in its region.
- During prediction (when classifying an instance):
  - The instance is routed from the root to a leaf according to the outcomes of the tests along the path. The leaf's label becomes the tree's prediction.

How to split?

Constructing Decision Trees (DTs)

- Given some training data, what is the "optimal" DT?
  - With optimal, we mean here the "smallest" that fits the data perfectly
- In general, finding an optimal DT is called intractable (NP-hard)
  - There are exponentially many DT's that could be constructed from a given set of attributes (exponential in number of attributes)
- In practice, we use greedy heuristics to construct a good DT (making a series of locally optimal decisions)

- **Hunt's algorithm**: a decision tree is grown in a recursive fashion by partitioning the training records into successively **"purer" subsets**
  - We can use different notions of "impurity"

Practical hardness of optimal decision trees?

CONSTRUCTING OPTIMAL BINARY DECISION TREES IS NP-COMPLETE*

Laurent HYAFIL
IRIA – Laboria, 78150 Rocquencourt, France

and

Ronald L. RIVEST
Dept. of Electrical Engineering and Computer Science, M.I.T., Cambridge, Massachusetts 02139, USA

Information Processing Letters'76

We demonstrate that constructing optimal binary decision trees is an NP-complete problem, where an optimal tree is one which minimizes the expected number of tests required to identify the unknown object.

Let $p(x_i)$ be the length of the path from the root of the tree to the terminal node naming x_i, that is, the number of tests required to identify x_i. Then the cost of this tree is merely the external path length, that is, $\Sigma_{x_i \in X} p(x_i)$. This model is identical to that studied by Garey [3].

The *decision tree* problem $\mathrm{DT}(\mathcal{T}, X, w)$ is to determine whether there exists a decision tree with cost less than or equal to w, given $\mathcal{T}$ and X.

To show that DT is NP-complete, we show that EC3 α DT, where EC3 is the problem of finding an exact cover for a set X, and where each of the subsets available for use contains exactly 3 elements. More

Effectiveness with modern ILP solvers.
Problem: we may need exponentially many statistics over the data (but exponential only in number of attribute not data size)

?

Cp. to Shannon-Fano top-down vs.
Huffman optimal bottom-up!

Hyafil, Rivest. Constructing optimal binary decision trees is NP-complete. Information Processing Letters, 1976. https://doi.org/10.1016/0020-0190(76)90095-8
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Practical hardness of optimal decision trees?

CONSTRUCTING OPTIMAL BINARY DECISION TREES IS NP-COMPLETE*

Laurent HYAFIL
IRIA – Laboria, 78150 Rocquencourt, France

and

Ronald L. RIVEST
Dept. of Electrical Engineering and Computer Science, M.I.T., Cambridge, Massachusetts 02139, USA

Information Processing Letters'76

Mach Learn (2017) 106:1039–1082
DOI 10.1007/s10994-017-5633-9

Optimal classification trees

Dimitris Bertsimas[1] · Jack Dunn[2]

7 Conclusions

In this paper, we have revisited the classical problem of decision tree creation under a modern MIO lens. We presented a novel MIO formulation for creating optimal decision trees that captures the discrete nature of this problem and motivates two new methods, OCT and OCT-H.

Experiments with synthetic data provide strong evidence that optimal decision trees can better recover the true generating decision tree in the data, contrary to the popular belief that such optimal methods will just overfit the training data at the expense of generalization ability.

Exploiting the astonishing progress of MIO solvers in the last 25 years, our comprehensive computational experiments showed that both the OCT and OCT-H problems are both practically solvable and deliver solutions that outperform the classical state-of-the-art methods, with average absolute improvements in out-of-sample accuracy over CART of 1–2% for OCT and 3–5% for OCT-H across all datasets, depending on the depth of the tree. We also provided guidance for each method that predict when each is most likely to outperform CART.

Hyafil, Rivest. Constructing optimal binary decision trees is NP-complete. Information Processing Letters, 1976. https://doi.org/10.1016/0020-0190(76)90095-8
Bertsimas, Dunn. Optimal classification trees. Machine Learning, 2017. https://doi.org/10.1007/s10994-017-5633-9
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Hunt's algorithm: Top-down induction of Decision trees

- Create a root node x; assign it all training examples: D_x
- Repeat {
  - If all records in D_x belong to the same class, then make x a leaf node and assign it the class label
  - Else:
    - Choose an attribute A that partitions the training records at node x into the **"purest" subsets**
    - For each value v of A, create a new child node $X_{A=v}$ and assign it the training examples $D_{A=v}$
  - Choose a non-leaf node x
- Until all nodes are leaves

There are different ways to measure "impurity" and we will discuss in a moment variants of how we could measure that

2.1 Divide and conquer

The skeleton of Hunt's method for constructing a decision tree from a set T of training cases is elegantly simple. Let the classes be denoted $\{C_1, C_2, \ldots, C_k\}$. There are three possibilities:

...

[Quinlan'93]

Likely origin of the name.
Hunt was Quinlan's PhD advisor

"Tan, Steinbach, Kumar. Introduction to Data Mining-Pearson" mentions "Hunt's algorithm" but does not leave a citation. Quinlan himself in his 1993 book on C4.5 refers to "Hunt, Marin, Stone. Experiments in induction. Academic press, 1966".

"Impurity" of subsets

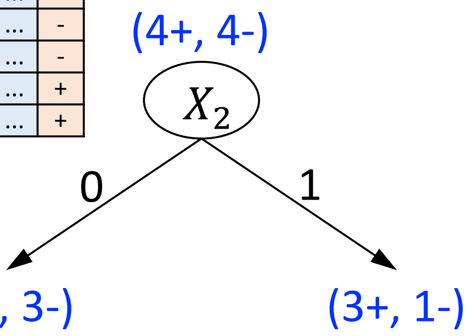
To determine how well a test condition performs, we compare the "impurity" of the parent node (before splitting) with the "impurity" of the child nodes (after splitting).

Call the impurity at node N: $I(N)$ currently a black box fct

Impurity before: $I(N)$

Impurity after: ?

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+



ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

"Impurity" of subsets

To determine how well a test condition performs, we compare the "impurity" of the parent node (before splitting) with the "impurity" of the child nodes (after splitting).

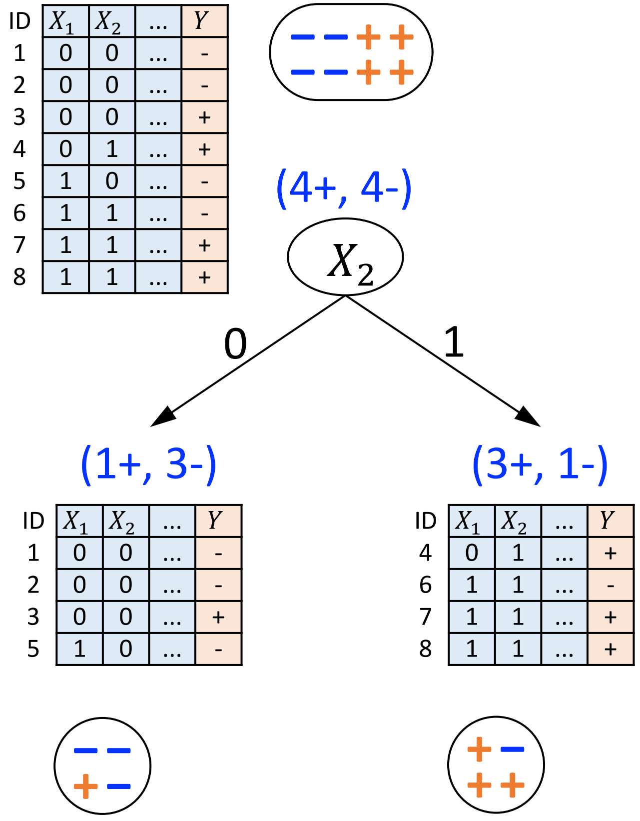
Call the impurity at node N: $I(N)$ currently a black box fct

Impurity before: $I(N)$

Impurity after: $\sum_{C\in\text{children}(N)} p_C \cdot I(C) = \mathbb{E}_{p(C)}[I(C)]$ Expected (weighted average) impurity

infor. gain (IG): $I(N) - \mathbb{E}_{p(C)}[I(C)]$

Have we seen that before ?



ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

"Impurity" of subsets

To determine how well a test condition performs, we compare the "impurity" of the parent node (before splitting) with the "impurity" of the child nodes (after splitting).

Call the impurity at node N: $I(N)$

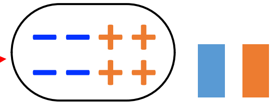
Impurity before: $I(N)$

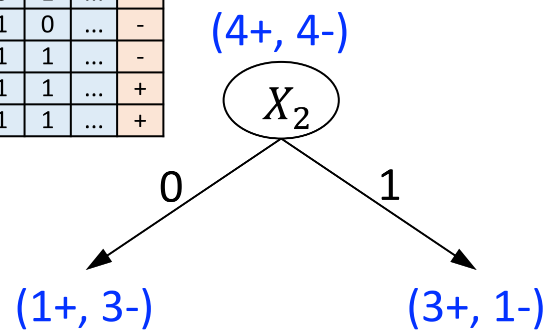
Impurity after: $\sum_{C \in \text{children}(N)} p_C \cdot I(C) = \mathbb{E}_{p(C)}[I(C)]$

infor. gain (IG): $I(N) - \mathbb{E}_{p(C)}[I(C)]$

What is the minimum encoding length for the labels (per label) before the split?

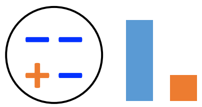
ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

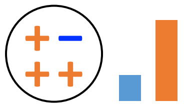




ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+





"Impurity" of subsets

To determine how well a test condition performs, we compare the "impurity" of the parent node (before splitting) with the "impurity" of the child nodes (after splitting).

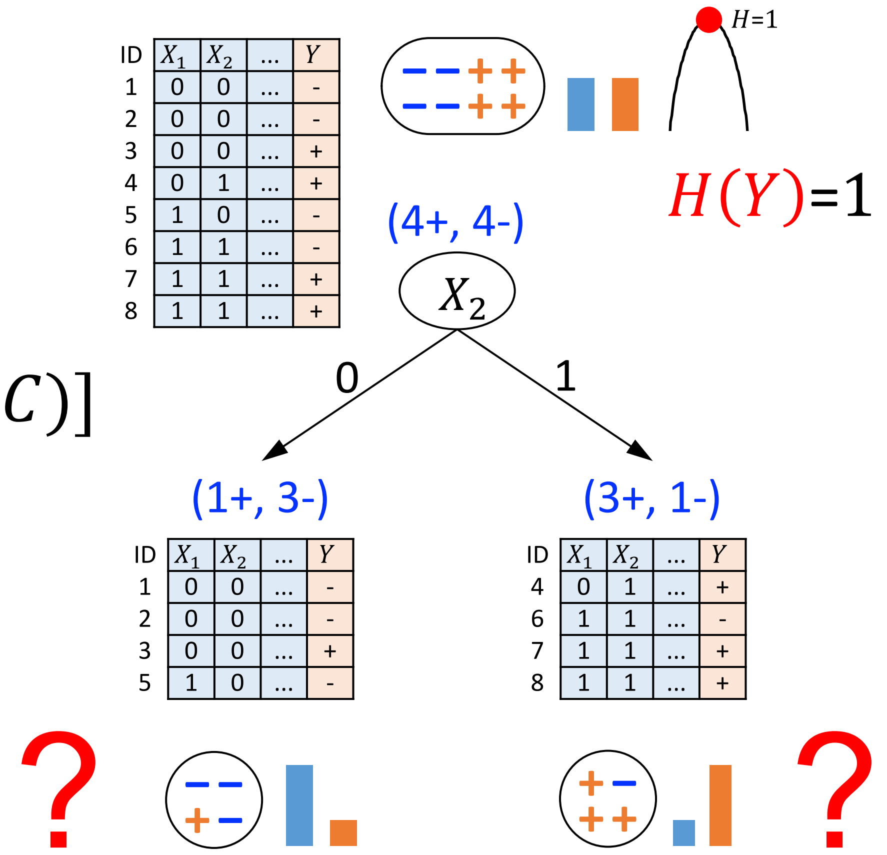
Call the impurity at node N: $I(N)$ $= H(Y)$

(per label)

Impurity before: $I(N)$

Impurity after: $\sum_{C \in \text{children}(N)} p_C \cdot I(C) = \mathbb{E}_{p(C)}[I(C)]$

infor. gain (IG): $I(N) - \mathbb{E}_{p(C)}[I(C)]$



ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

"Impurity" of subsets

To determine how well a test condition performs, we compare the "impurity" of the parent node (before splitting) with the "impurity" of the child nodes (after splitting).

Call the impurity at node N: $I(N)$ $= H(Y)$

(per label)

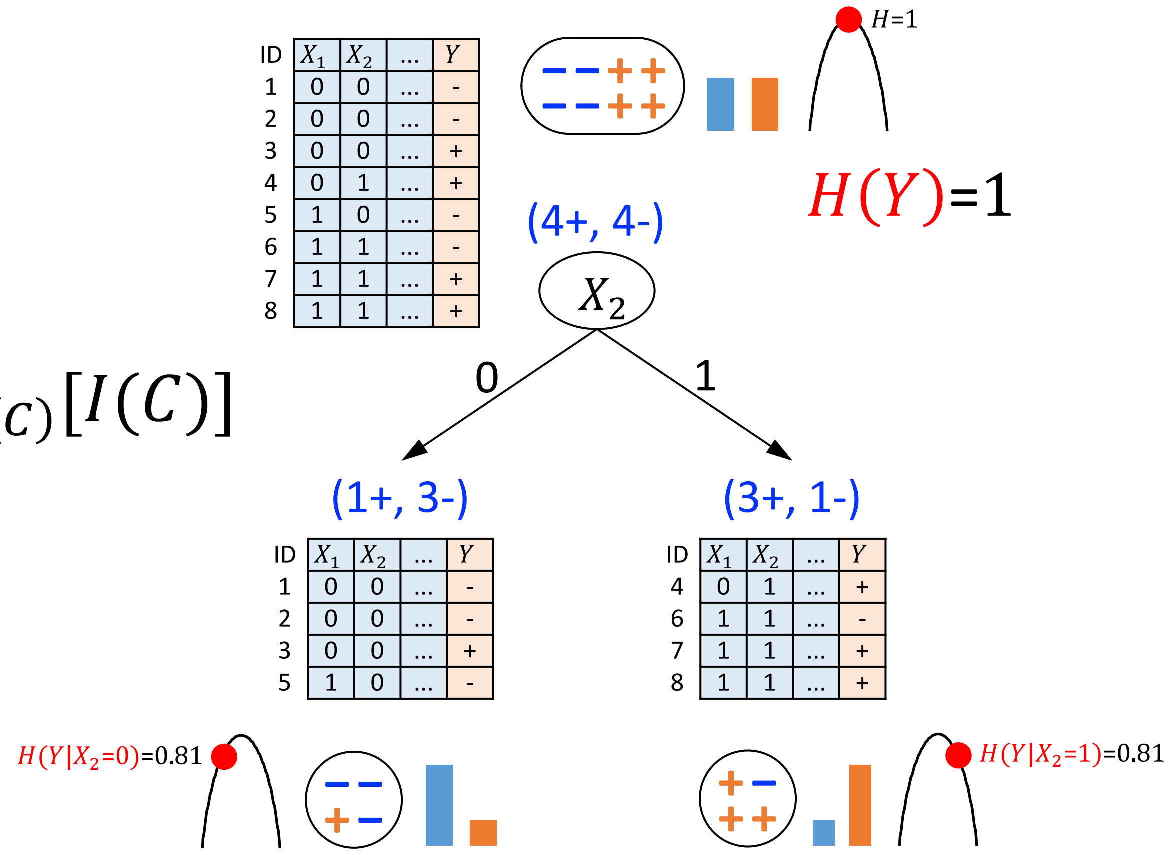
Impurity before: $I(N)$

Impurity after: $\sum_{C \in \text{children}(N)} p_C \cdot I(C) = \mathbb{E}_{p(C)}[I(C)]$

infor. gain (IG): $I(N) - \mathbb{E}_{p(C)}[I(C)]$

"information gain"

IG or Δ_{info}: $H(Y) -$?



ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

"Impurity" of subsets

To determine how well a test condition performs, we compare the "impurity" of the parent node (before splitting) with the "impurity" of the child nodes (after splitting).

Call the impurity at node N: $I(N)$ $= H(Y)$

(per label)

Impurity before: $I(N)$

Impurity after: $\sum_{C \in \text{children}(N)} p_C \cdot I(C) = \mathbb{E}_{p(C)}[I(C)]$

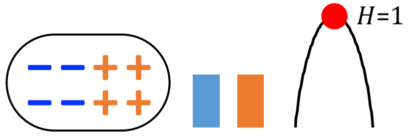
infor. gain (IG): $I(N) - \mathbb{E}_{p(C)}[I(C)]$

"information gain"

IG or Δ_{info}: $H(Y) - H(Y|X)$

Conditional entropy: the amount of information needed to describe the outcome of RV Y given that we know the value of another RV X.

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+



$H(Y)$=1

(4+, 4-)

X_2

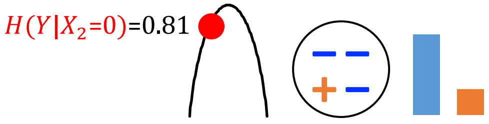
0 | 1

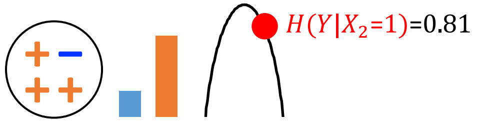
(1+, 3-)

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

(3+, 1-)

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+





$$H(Y|X) = \sum_x p(x) \cdot H(Y|X = x) = \mathbb{E}_{p(x)}[H(Y|X = x)]$$

"Impurity" of subsets

To determine how well a test condition performs, we compare the "impurity" of the parent node (before splitting) with the "impurity" of the child nodes (after splitting).

Call the impurity at node N: $I(N)$ $= H(Y)$

(per label)

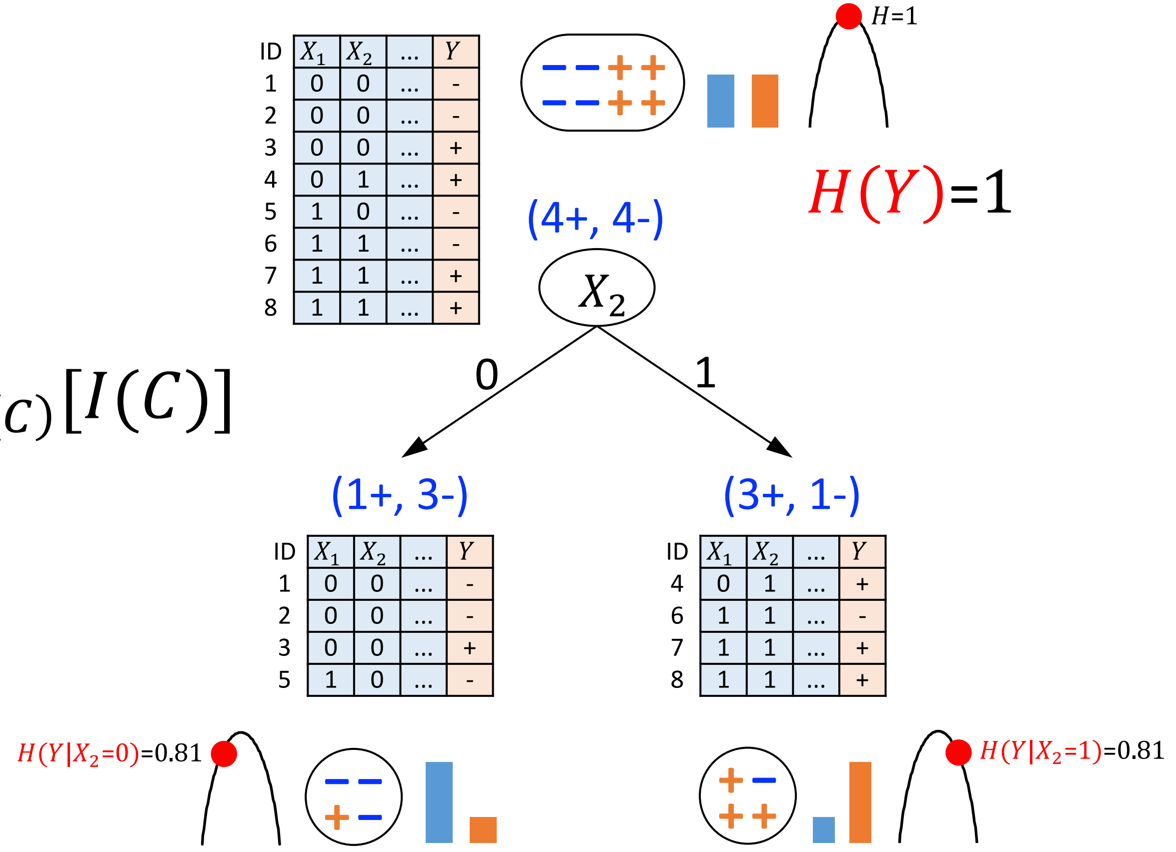
Impurity before: $I(N)$

Impurity after: $\sum_{C \in \text{children}(N)} p_C \cdot I(C) = \mathbb{E}_{p(C)}[I(C)]$

infor. gain (IG): $I(N) - \mathbb{E}_{p(C)}[I(C)]$

"information gain"

IG or Δ_{info}: $\boxed{H(Y) - H(Y|X) = I(X;Y)}$



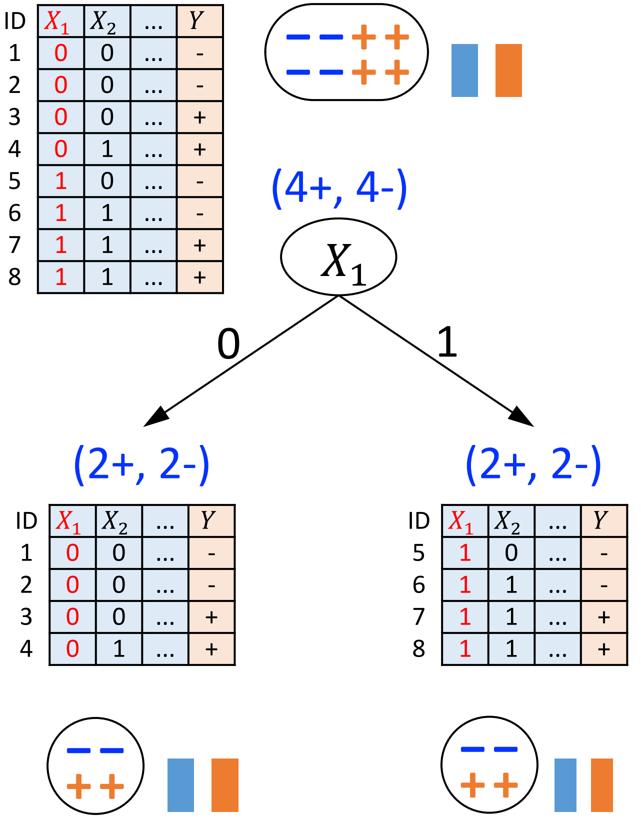
ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

Δ_{info} (information gain gain IG) is the reduction of class label entropy $H(Y)$ from the parent (i.e. the training data in a branch) to the average entropies of the children (i.e. the new partitions constructed from the values of variable X.

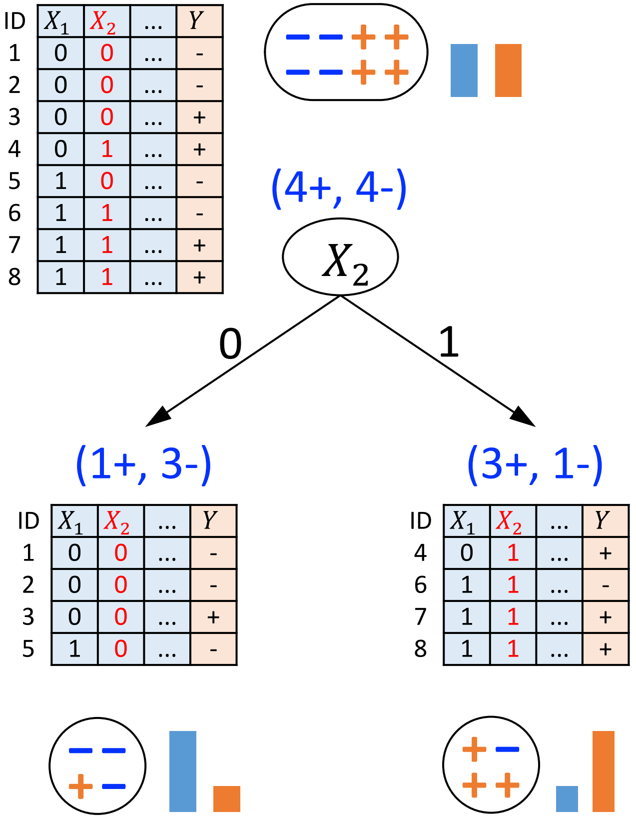
"Impurity" reduction



ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+

ID	X_1	X_2	...	Y
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+



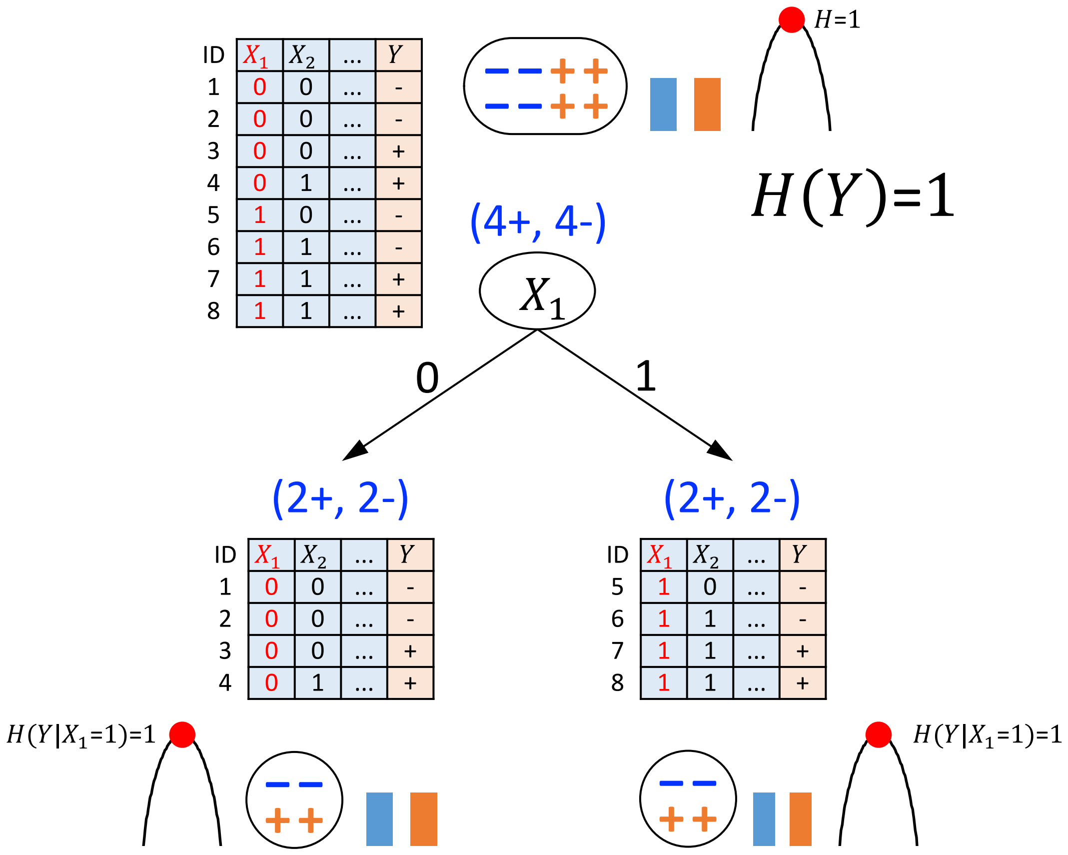
ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

Which variable would you choose ?

"Impurity" reduction, measured by entropy



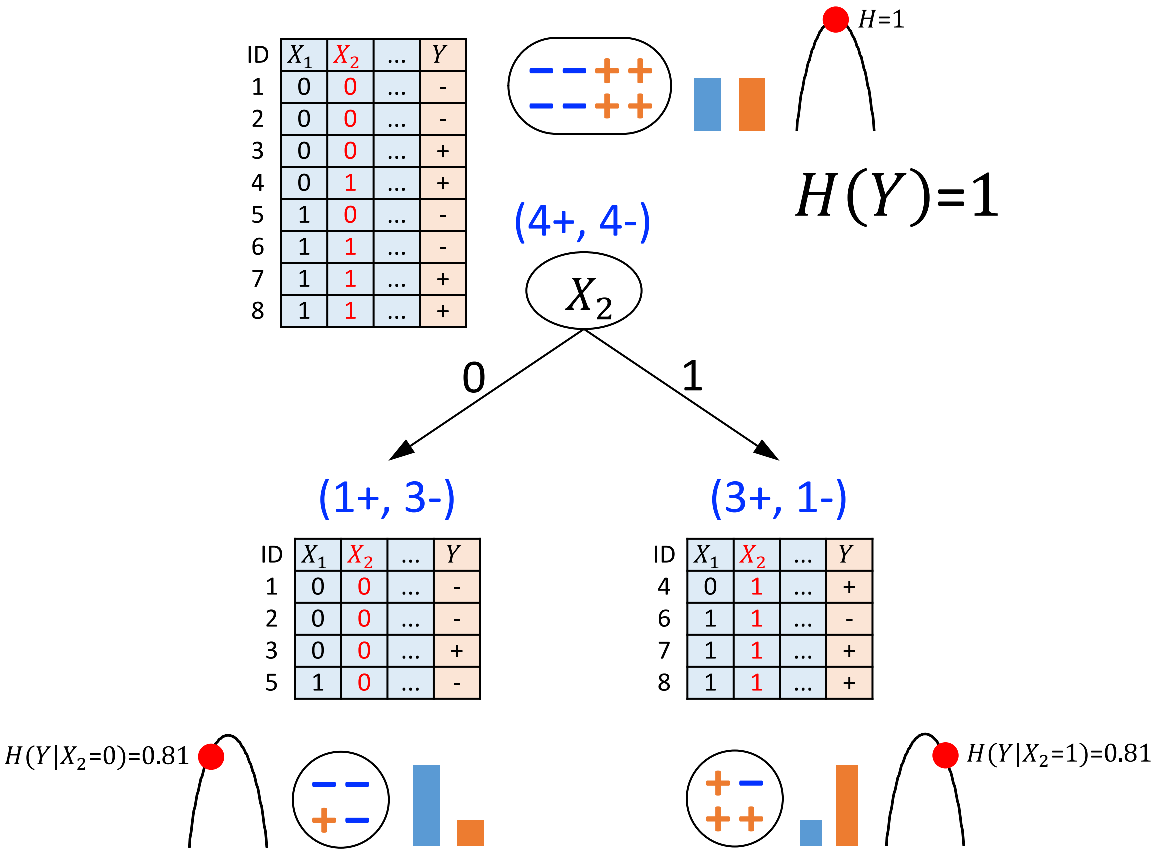
ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+

ID	X_1	X_2	...	Y
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

$H(Y|X_2)$=1

Δ_{info}= ?



ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

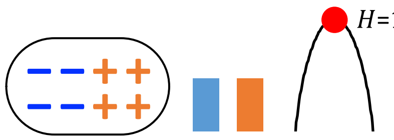
ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

$H(Y|X_2)$=0.81

Δ_{info}= ?

"Impurity" reduction, measured by entropy

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+



H=1

$H(Y)$=1

(4+, 4-)

X_1

0 1

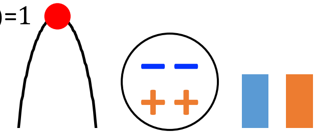
(2+, 2-)

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+

(2+, 2-)

ID	X_1	X_2	...	Y
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

$H(Y|X_1$=1)=1

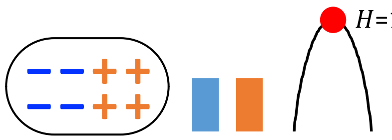


$H(Y|X_1$=1)=1

$H(Y|X_2)$=1

Δ_{info}=$I(Y;X_2)$=0

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+



H=1

$H(Y)$=1

(4+, 4-)

X_2

0 1

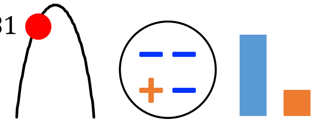
(1+, 3-)

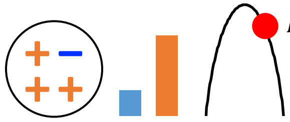
ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-

(3+, 1-)

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

$H(Y|X_2$=0)=0.81





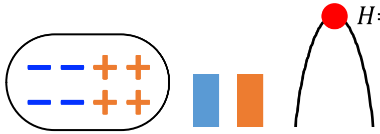
$H(Y|X_2$=1)=0.81

$H(Y|X_2)$=0.81

Δ_{info}=$I(Y;X_2)$=0.19

"Impurity" reduction, measured by entropy

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+



$H(Y)$=1

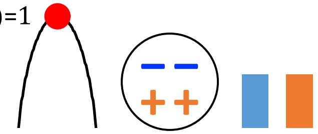
(4+, 4-)

X_1

0   1

(2+, 2-)

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+



(2+, 2-)

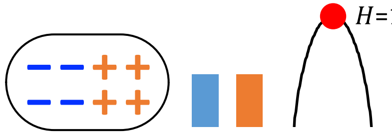
ID	X_1	X_2	...	Y
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

$H(Y|X_2)$=1

Δ_{info}=$I(Y;X_2)$=0

Uniform labels
(high impurity)
Not a good split ☹

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
4	0	1	...	+
5	1	0	...	-
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+



$H(Y)$=1

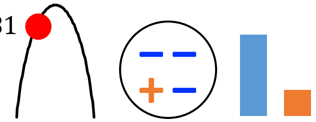
(4+, 4-)

X_2

0   1

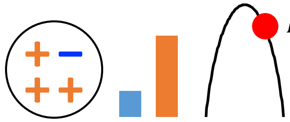
(1+, 3-)

ID	X_1	X_2	...	Y
1	0	0	...	-
2	0	0	...	-
3	0	0	...	+
5	1	0	...	-



(3+, 1-)

ID	X_1	X_2	...	Y
4	0	1	...	+
6	1	1	...	-
7	1	1	...	+
8	1	1	...	+

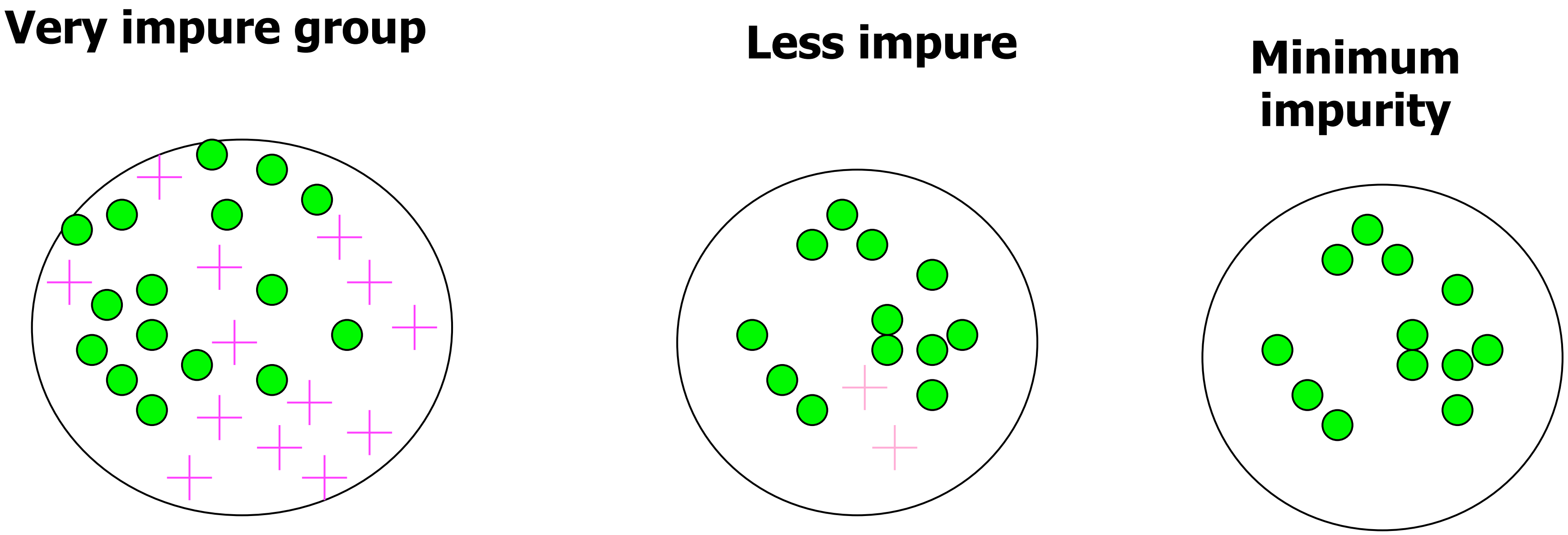


Non-uniform labels
(low impurity)
A better split ☺

$H(Y|X_2)$=0.81

Δ_{info}=$I(Y;X_2)$=0.19

Impurity

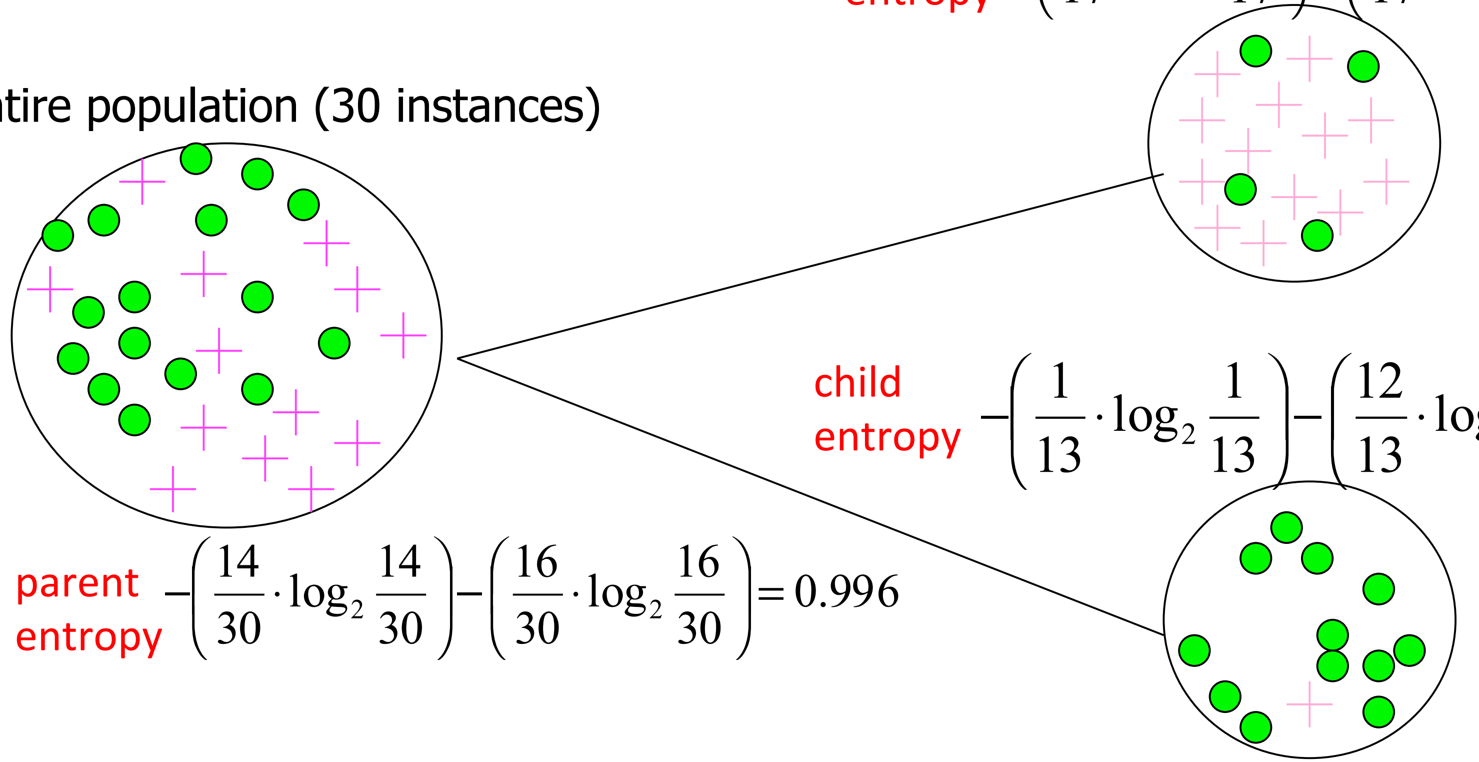


Calculating Information Gain

Information Gain = entropy(parent) – [average entropy(children)]

child entropy $-\left(\frac{13}{17}\cdot\log_2\frac{13}{17}\right)-\left(\frac{4}{17}\cdot\log_2\frac{4}{17}\right)=0.787$

Entire population (30 instances)



17 instances

child entropy $-\left(\frac{1}{13}\cdot\log_2\frac{1}{13}\right)-\left(\frac{12}{13}\cdot\log_2\frac{12}{13}\right)=0.391$

13 instances

parent entropy $-\left(\frac{14}{30}\cdot\log_2\frac{14}{30}\right)-\left(\frac{16}{30}\cdot\log_2\frac{16}{30}\right)=0.996$

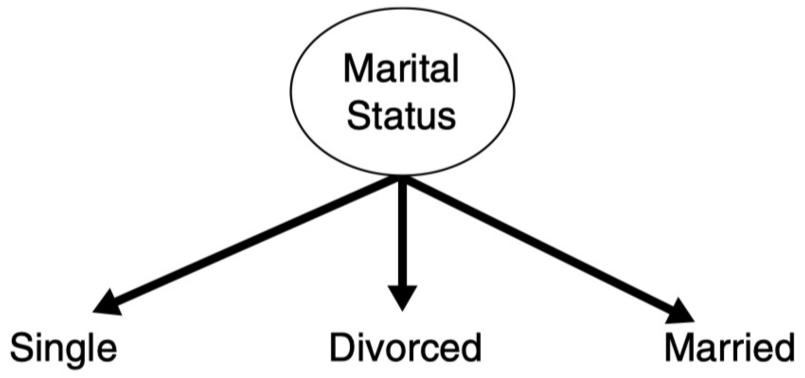
(Weighted) Average Entropy of Children = $\left(\frac{17}{30}\cdot 0.787\right)+\left(\frac{13}{30}\cdot 0.391\right)=0.615$

Information Gain= 0.996 - 0.615 = 0.38

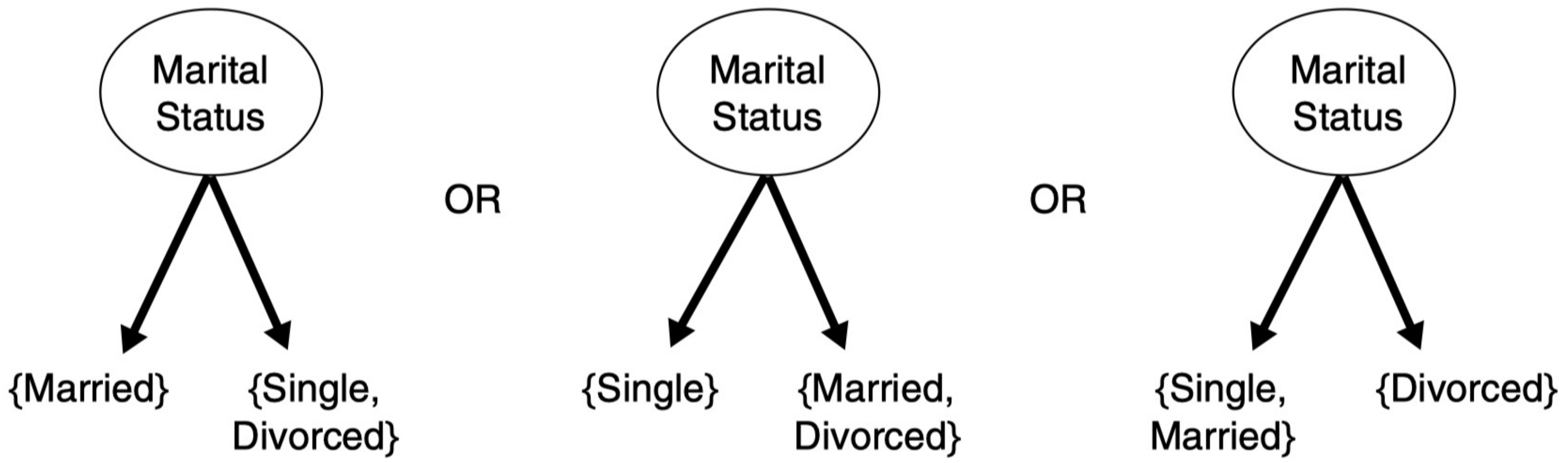
Slide by Eric Eaton (https://www.seas.upenn.edu/~eeaton/), based on a slide by Pedro Domingos (https://homes.cs.washington.edu/~pedrod/)
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Test conditions for nominal attributes

Multiway split



Binary split by grouping attributes



[Tan+'18] Tan, Steinbach, Kumar. Introduction to Data Mining, 2nd ed, 2018. Figure 3.8 https://www-users.cse.umn.edu/~kumar001/dmbook/index.php
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Impurity measures (in addition to entropy)

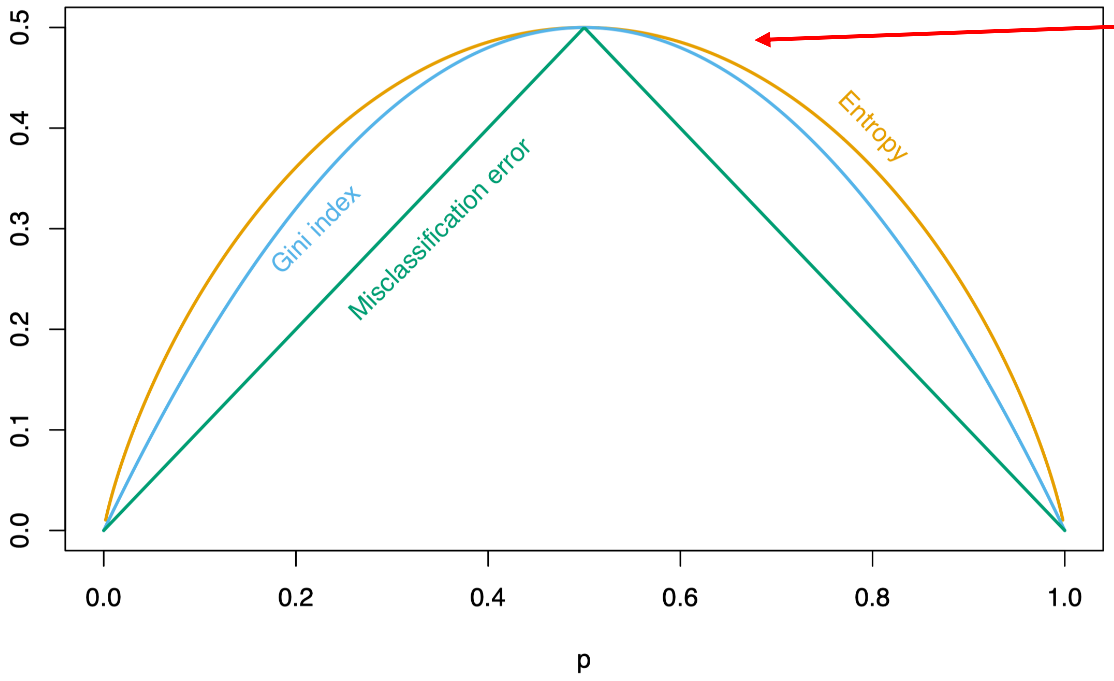
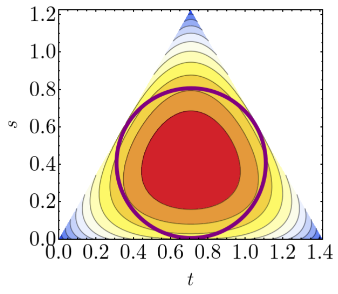


FIGURE 9.3. *Node impurity measures for two-class classification, as a function of the proportion p in class 2. Cross-entropy has been scaled to pass through* (0.5, 0.5).

Another way to think about the y-axis is $\dfrac{I(S)}{\max\limits_{S} I(S)}$

$$
\begin{aligned}
\text{Entropy} &= -\sum_{i=0}^{c-1} p_i(t)\log_2 p_i(t),\\
\text{Gini index} &= 1-\sum_{i=0}^{c-1} p_i(t)^2,\\
\text{Classification error} &= 1-\max_i[p_i(t)],
\end{aligned}
$$

Entropy	**Gini index**
measures the amount of uncertainty (or randomness) in a set / can be interpreted as the average amount of information needed to specify the class of an instance.	measures the probability of misclassifying a randomly chosen element in a set / can be interpreted as the expected error rate in a classifier.
The range of entropy is [0, lg(c)], where c is the number of classes.	The range of the Gini index is [0, 1-1/c] (often incorrectly stated as [0,1])
It has a bias toward selecting splits that result in a higher reduction of uncertainty (distinguishes more between highly impure and moderately impure splits, **better for imbalanced datasets**)	It has a bias toward selecting splits that result in a more balanced (equally sized) distribution of classes.
Entropy was proposed by Quinlan (ID3 and C4.5 algorithms). Earlier paper from 1979 is hard to find, usual citation: "*Quinlan. Induction of Decision Trees, 1986.*" https://doi.org/10.1007/BF00116251	Gini index is typically used in CART ("Classification and Regression Trees") "*Breiman, Friedman, Olshen, Stone: Classification and Regression Trees, 1984.*" https://doi.org/10.1201/9781315139470



Simple interesting scribe: create a notebook and figures that compare ternary Gini vs entropy function over the probability simplex. How are the derivatives close to "pure sets"? Two starter posts:
https://physics.stackexchange.com/questions/363545/what-is-the-relation-between-linear-purity-and-von-neumann-entropy-of-a-state
https://math.stackexchange.com/questions/3791203/what-do-the-level-sets-of-the-shannon-entropy-look-like

Left figure: [Hastie+'09] Hastie, Tibshirani, Friedman. The Elements of Statistical Learning, 2nd ed, 2009. https://doi.org/10.1007/978-0-387-84858-7

Gini index and "logical entropy"

When there are point probabilities $p = (p_1, ..., p_n)$ for p_j as the probability of the outcome $u_j \in U$ with $\sum_{j=1}^{n} p_j = 1$, then $\Pr(B_i) = \sum \{p_j : u_j \in B_i\}$ in the formula for logical entropy. This also gives the definition of logical entropy for any probability distribution $p = (p_1, ..., p_n)$,

$$h(p) = 1 - \sum_{j=1}^{n} p_j^2. \tag{2.3}$$

$$1 = 1^2 = (p_1 + ... + p_n)(p_1 + ... + p_n) = \sum_{j=1}^{n} p_i^2 + \sum_{j \neq k} p_j p_k \tag{2.4}$$

so that:

$$h(p) = 1 - \sum_{j=1}^{n} p_i^2 = \sum_{j=1}^{n} p_j (1 - p_j) = \sum_{j \neq k} p_j p_k = 2 \sum_{j<k} p_j p_k \tag{2.5}$$

Ellerman. Introduction to Logical Entropy and Its Relationship to Shannon Entropy, 2021. https://arxiv.org/abs/2112.01966
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Gini index and "logical entropy"

1.5 Brief History of the Logical Entropy Formula

The logical entropy formula $h(p) = \sum_i p_i (1 - p_i) = 1 - \sum_i p_i^2$ is the probability of getting distinct values $u_i \neq u_j$ in two independent samplings of the random variable u. The complementary measure $1 - h(p) = \sum_i p_i^2$ is the probability that the two drawings yield the same value from U. Thus $1 - \sum_i p_i^2$ is a measure of heterogeneity or diversity in keeping with our theme of information as distinctions, while the complementary measure $\sum_i p_i^2$ is a measure of homogeneity or concentration. Historically, the formula can be found in either form depending on the particular context. The p_i's might be relative shares such as the relative share of organisms of the ith species in some population of organisms, and then the interpretation of p_i as a probability arises by considering the random choice of an organism from the population.

According to I. J. Good, the formula has a certain naturalness: "If $p_1, \ldots, p_t$ are the probabilities of t mutually exclusive and exhaustive events, any statistician of this century who wanted a measure of homogeneity would have take about two seconds to suggest $\sum p_i^2$ which I shall call ρ." [13, p. 561] As noted by Bhargava and Uppuluri [4], the formula $1 - \sum p_i^2$ was used by Gini in 1912 [10] as a measure of "mutability" or diversity. But another development of the formula (in the complementary form) in the early twentieth century was in cryptography. The American cryptologist, William F. Friedman, devoted a 1922 book [9] to the "index of coincidence" (i.e., $\sum p_i^2$). Solomon Kullback (see the Kullback-Leibler divergence treated later) worked as an assistant to Friedman and wrote a book on cryptology which used the index [16].

Ellerman. New Foundations for Information Theory: Logical Entropy and Shannon Entropy, Springer, 2021. https://doi.org/10.1007/978-3-030-86552-8
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

"Logical entropy"

2.2 Logical Entropy, Not Shannon Entropy, Is a (Non-negative) Measure

…

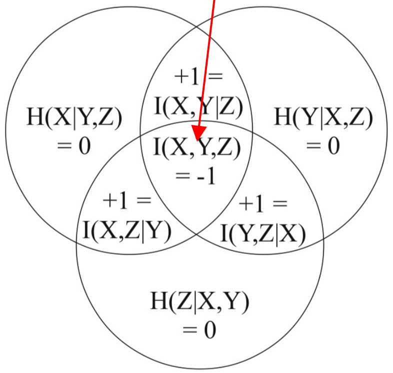
As we will see, for three or more random variables, the Shannon mutual information can have negative values—which has no known interpretation.

Recall that this concern can be easily avoided by more careful notation and *not* using the terminology of "mutual information" for what we called the "interaction information"

4.2 An Example of Negative Mutual Information for Shannon Entropy

Norman Abramson gives an example [1, pp. 130–131] where the Shannon mutual information of three variables is negative.[3] William Feller gives a similar concrete example that we will use [11, Exercise 26, p. 143]. Any probability theory textbook example to show that pair-wise independence does not imply mutual independence for three or more random variables would do as well.

Fig. 4.6 Negative 'area' $I(X, Y, Z)$ in Venn diagram



Ellerman. New Foundations for Information Theory: Logical Entropy and Shannon Entropy, Springer, 2021. https://doi.org/10.1007/978-3-030-86552-8
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

End-to-end "Tennis" Example by Tom Mitchell

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

$\langle \mathbf{x}_4, y_4 \rangle$

#no: 5
#yes: 9

$H(P) =$?

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

#no: 5
#yes: 9

What happens if we split by attribute W ?

$$H(P) = H\left(\frac{9}{14}, \frac{5}{14}\right) = 0.940$$

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
13	overcast	hot	normal	weak	yes
2	sunny	hot	high	strong	no
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
14	rain	mild	high	strong	no

What happens if we split by attribute W?

$$H(P|W) = \ ?$$

$$I(P;W) = \ ?$$

now partitioned by W

$$H(P) = H\left(\frac{9}{14}, \frac{5}{14}\right) = 0.940$$

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	(O)utlook	(T)emp.	(H)umidity	(W)ind	(P)lay
1	sunny	hot	high	weak	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
13	overcast	hot	normal	weak	yes
2	sunny	hot	high	strong	no
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
14	rain	mild	high	strong	no

now partitioned by W

$$H(P) = H\left(\frac{9}{14}, \frac{5}{14}\right) = 0.940$$

What happens if we split by attribute W?

$$\mathbb{E}_{p(v)}[H(P|W = v)]$$

$$H(P|W) = \sum_v p(v) \cdot H(P|W = v)$$

$$H(P|W = \text{weak}) = \ ?$$

$$H(P|W = \text{strong}) = \ ?$$

$$I(P;W) = \ ?$$

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	(O)utlook	(T)emp.	(H)umidity	(W)ind	(P)lay
1	sunny	hot	high	weak	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
13	overcast	hot	normal	weak	yes
2	sunny	hot	high	strong	no
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
14	rain	mild	high	strong	no

now partitioned by W

$$H(P) = H\left(\frac{9}{14}, \frac{5}{14}\right) = 0.940$$

What happens if we split by attribute W?

$$H(P|W) = \underbrace{\sum_v p(v) \cdot H(P|W=v)}_{\mathbb{E}_{p(v)}[H(P|W=v)]}$$

$$H(P|W = \text{weak}) = H\left(\frac{2}{8}, \frac{6}{8}\right) = 0.811$$

$$H(P|W = \text{strong}) = H\left(\frac{3}{6}, \frac{3}{6}\right) = 1$$

$$H(P|W) = \frac{8}{14} \cdot 0.811 + \frac{6}{14} \cdot 1 = 0.892$$

$$I(P;W) = \ ?$$

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
13	overcast	hot	normal	weak	yes
2	sunny	hot	high	strong	no
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
14	rain	mild	high	strong	no

now partitioned by W

$$H(P) = H\left(\frac{9}{14}, \frac{5}{14}\right) = 0.940$$

What happens if we split by attribute W?

$$\mathbb{E}_{p(v)}[H(P|W = v)]$$

$$H(P|W) = \sum_v p(v) \cdot H(P|W = v)$$

$$H(P|W = \text{weak}) = H\left(\frac{2}{8}, \frac{6}{8}\right) = 0.811$$

$$H(P|W = \text{strong}) = H\left(\frac{3}{6}, \frac{3}{6}\right) = 1$$

$$H(P|W) = \frac{8}{14} \cdot 0.811 + \frac{6}{14} \cdot 1 = 0.892$$

$$I(P;W) = H(P) - H(P|W) = 0.048$$

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

Now we calculate mutual information (aka information gain) between P and the other attributes

$$I(P;W) = H(P) - H(P|W) = 0.048$$

$$I(P;H) = 0.152$$

$$I(P;T) = 0.029$$

$$I(P;O) = 0.246$$

which attribute do we pick ?

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

Now we calculate mutual information (aka information gain) between P and the other attributes

$$I(P;W) = H(P) - H(P|W) = 0.048$$

$$I(P;H) = 0.152$$

$$I(P;T) = 0.029$$

$$I(P;O) = 0.246$$

We pick the attribute with the highest information gain

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

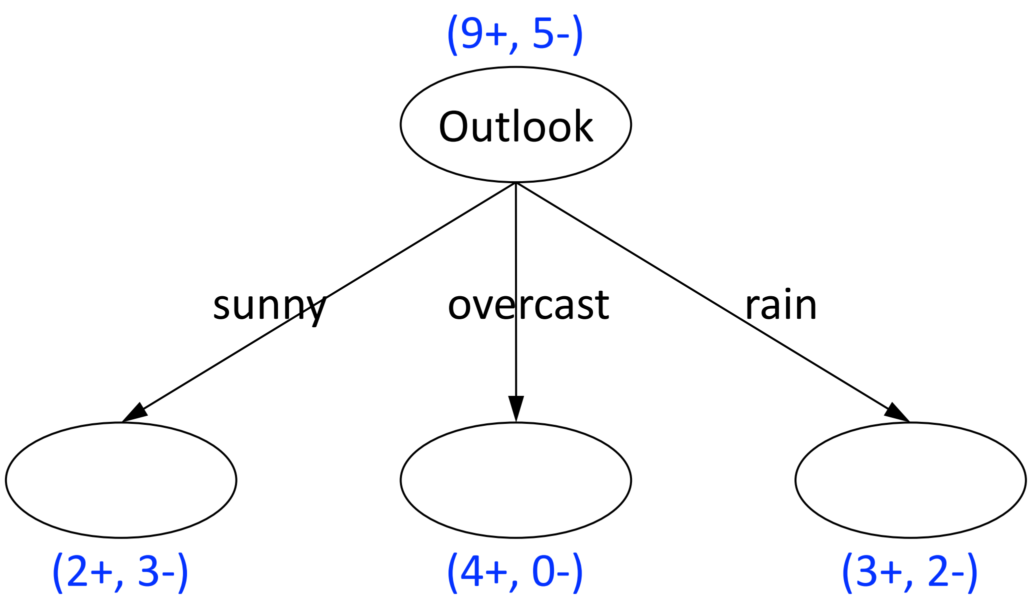
Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
10	rain	mild	normal	weak	yes
14	rain	mild	high	strong	no

now partitioned by O



$H(P) = 0.971$ $\qquad$ $H(P) =$?

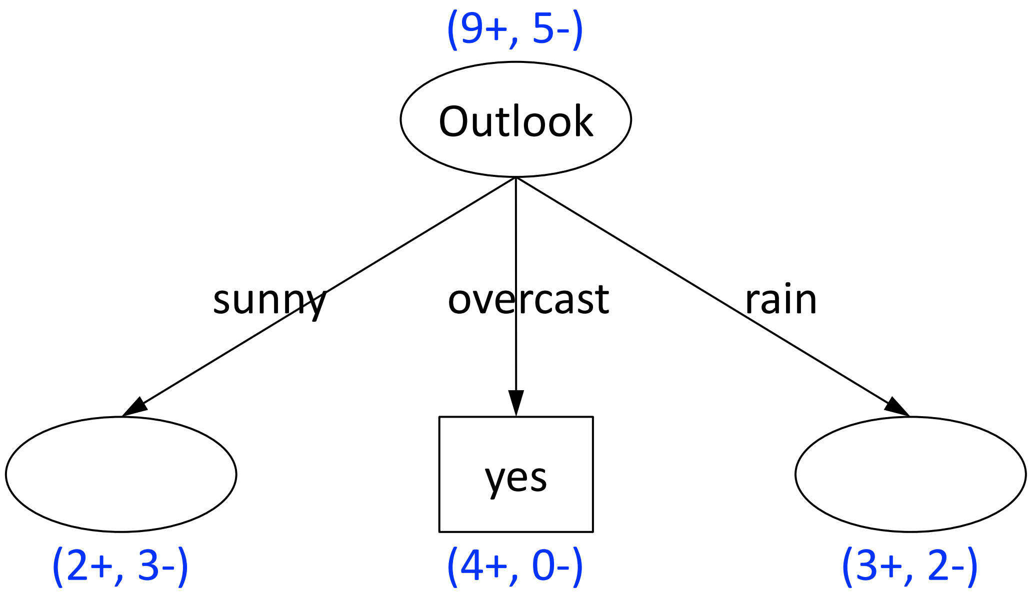
Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	(O)utlook	(T)emp.	(H)umidity	(W)ind	(P)lay
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
10	rain	mild	normal	weak	yes
14	rain	mild	high	strong	no

now partitioned by O



$H(P) = 0.971$ $H(P) = 0$

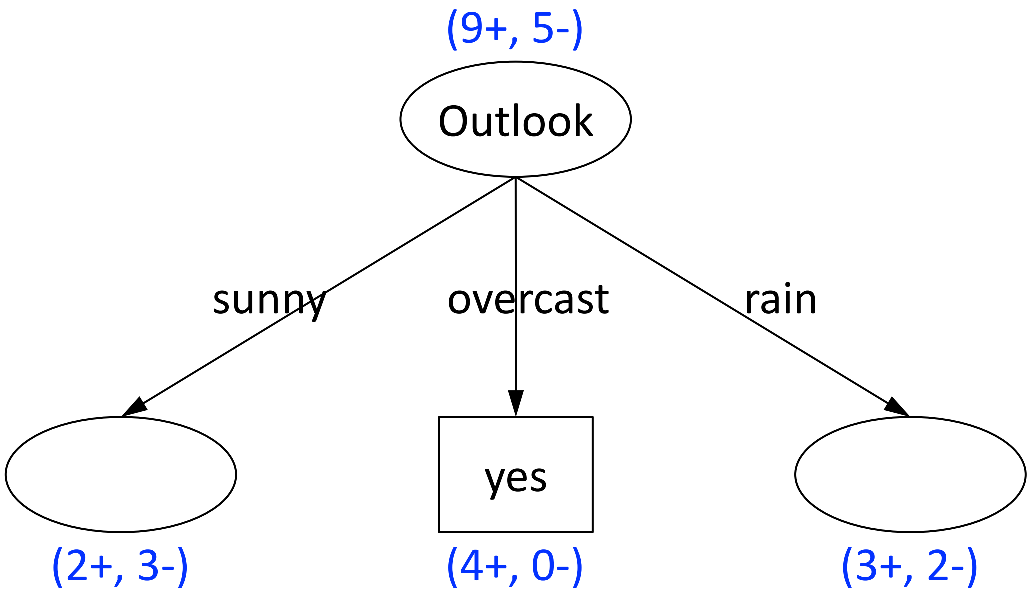
Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	(O)utlook	(T)emp.	(H)umidity	(W)ind	(P)lay
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
10	rain	mild	normal	weak	yes
14	rain	mild	high	strong	no

now partitioned by O



$H(P) = 0.971$

$I(P;T) = 0.571$

$I(P;H) = 0.971$

$I(P;W) = 0.020$

$H(P) = 0$

What next ?

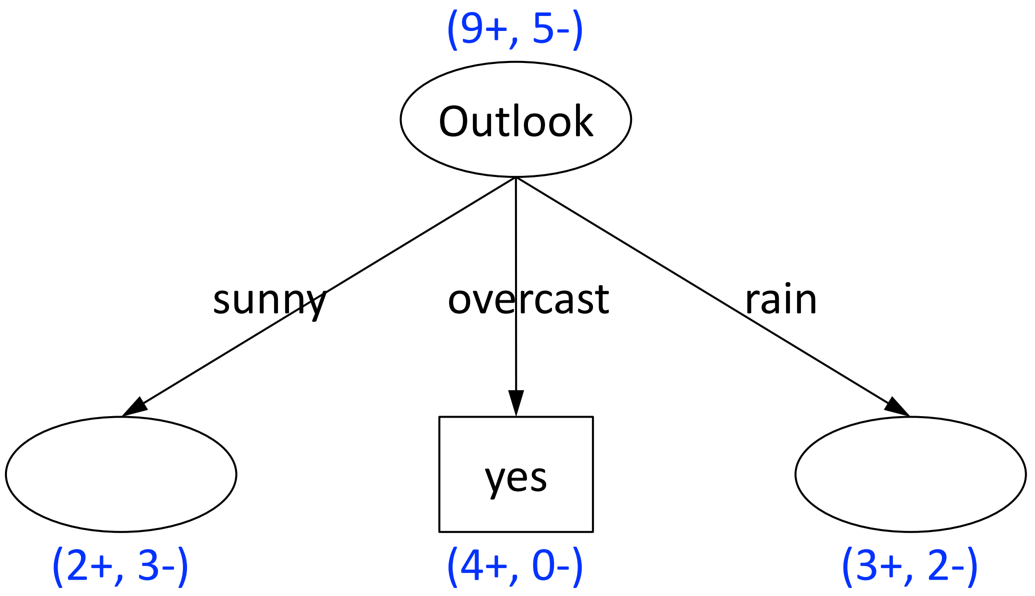
Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	(O)utlook	(T)emp.	(H)umidity	(W)ind	(P)lay
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
10	rain	mild	normal	weak	yes
14	rain	mild	high	strong	no

now partitioned by O



$H(P) = 0.971$
$I(P;T) = 0.571$
$I(P;H) = 0.971$
$I(P;W) = 0.020$

$H(P) = 0$

Perfect separation: We have no uncertainty left

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Tennis classification example

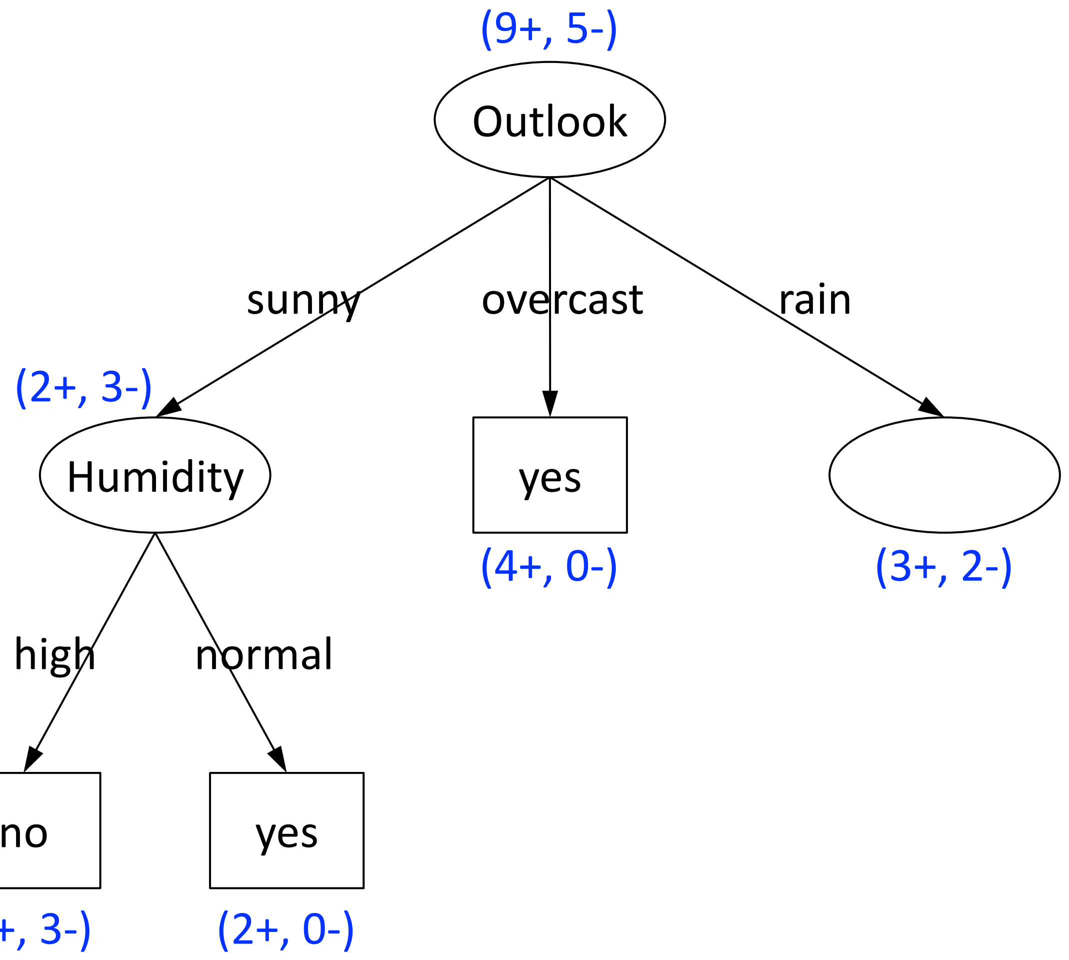
Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	(O)utlook	(T)emp.	(H)umidity	(W)ind	(P)lay
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
10	rain	mild	normal	weak	yes
14	rain	mild	high	strong	no

further partitioned by H

now partitioned by O

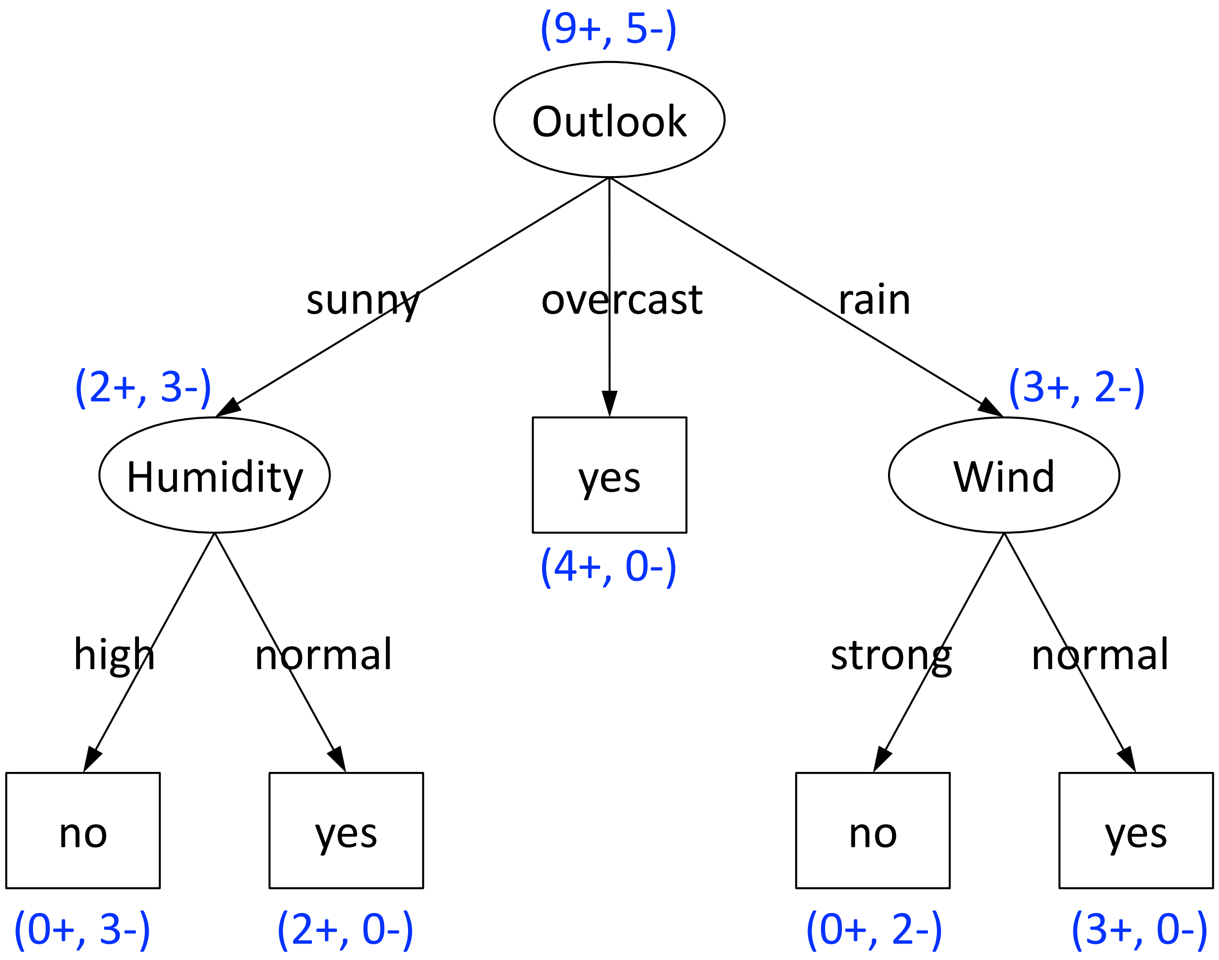


Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
11	sunny	mild	normal	strong	yes
3	overcast	hot	high	weak	yes
7	overcast	cool	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
6	rain	cool	normal	strong	no
14	rain	mild	high	strong	no
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
10	rain	mild	normal	weak	yes



Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Gain ratio

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

	Predictors				Response
day	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

I was missing a better predictor. Which one ?

Δ_{info}

$$I(P;O) = 0.246$$
$$I(P;H) = 0.152$$
$$I(P;W) = 0.048$$
$$I(P;T) = 0.029$$

$$H(P) = 0.940$$

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

Predictors					Response
(D)ay	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

The day has the highest mutual information

Δ_{info}

$$I(P;D) = 0.940$$

$$I(P;O) = 0.246$$

$$I(P;H) = 0.152$$

$$I(P;W) = 0.048$$

$$I(P;T) = 0.029$$

$$H(P) = 0.940$$

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Gain ratio

Disadvantage of information gain: It prefers attributes with large number of values that split the data into small, pure subsets

Quinlan's gain ratio (introduced with C4.5) uses normalization on the splitting criterion, i.e. it takes into account the number of outcomes produced by the attribute test condition and their distribution.

The **gain ratio** penalizes attributes such as Date by incorporating a term, called **split information**, that is sensitive to how broadly and uniformly the attribute splits the data

$$\text{Gain ratio} = \frac{\Delta_{\text{info}}}{\text{split info}}$$

The "split information" is just the entropy of the "split distribution", i.e. the distribution of the values of the attribute on which we split

If all are values balanced and we have k values, then $= \ln(k)$

Tennis classification example

Example: Deciding whether to play or not to play tennis on a Saturday (binary classification)

Columns denote 4 features X_i. Rows denote labeled instances $\langle \mathbf{x}_i, y_i \rangle$. Play denotes the classification.

Predictors					Response
(D)ay	**(O)utlook**	**(T)emp.**	**(H)umidity**	**(W)ind**	**(P)lay**
1	sunny	hot	high	weak	no
2	sunny	hot	high	strong	no
3	overcast	hot	high	weak	yes
4	rain	mild	high	weak	yes
5	rain	cool	normal	weak	yes
6	rain	cool	normal	strong	no
7	overcast	cool	normal	strong	yes
8	sunny	mild	high	weak	no
9	sunny	cool	normal	weak	yes
10	rain	mild	normal	weak	yes
11	sunny	mild	normal	strong	yes
12	overcast	mild	high	strong	yes
13	overcast	hot	normal	weak	yes
14	rain	mild	high	strong	no

The day has the highest mutual information

Δ_{info}	split info	gain ratio
$I(P;D) = 0.940$	$H(D) = 3.81$	0.247
$I(P;O) = 0.246$	$H(O) = 1.58$	0.156
$I(P;H) = 0.152$	$H(H) = 1$	0.152
$I(P;W) = 0.048$	$H(W) = 0.99$	0.048
$I(P;T) = 0.029$	$H(T) = 1.56$	0.019

$H(P) = 0.940$

$$\text{Gain ratio} = \frac{\Delta_{\text{info}}}{\text{split info}}$$

The normalization still does *not* help here. It would help if the data set was bigger as $H(D)$ grows with the size of the dataset, while $H(O)$ would stay the same.

Example from [Mitchell'97]. Introduction to Machine Learning, 1997. https://www.cs.cmu.edu/~tom/files/MachineLearningTomMitchell.pdf

Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Updated 10/23/2025

Part 3: Applications
L13: Decision trees (2/2) and MDL

Wolfgang Gatterbauer

cs7840 Foundations and Applications of Information Theory (fa25)

https://northeastern-datalab.github.io/cs7840/fa25/

10/23/2025

Pre-class conversations

- Last class recapitulation
- Projects: I have time to talk today after class

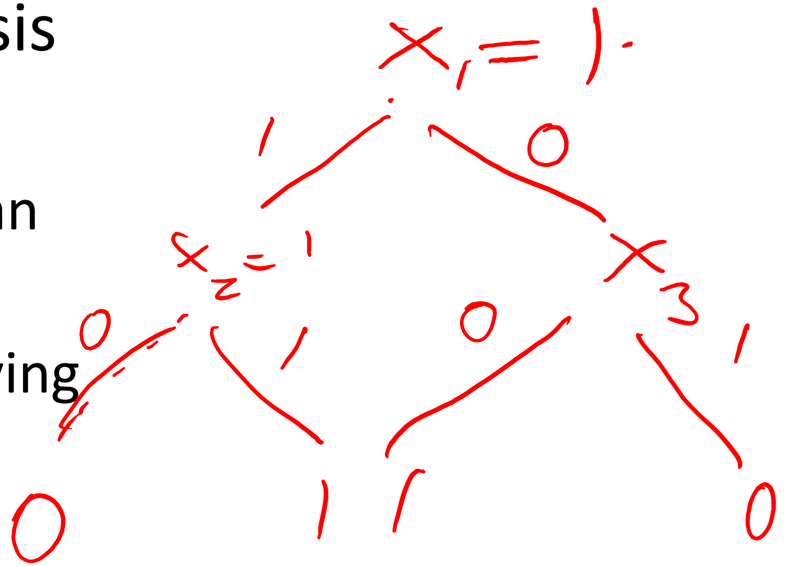
- Today:
  - Decision trees
  - Occam, MDL
  - Logistic regression

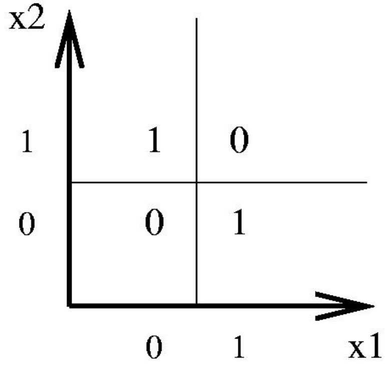
The Parity Function

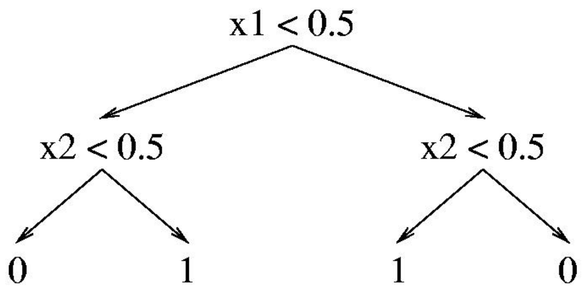
Expressiveness

Decision trees have a variable-sized hypothesis space

- As the #nodes (or depth) increases, the hypothesis space grows
  - Depth 1 ("decision stump"): can represent any boolean function of one feature
  - Depth 2: any boolean fn of two features; some involving three features (e.g., $(x_1 \wedge x_2) \vee (\neg x_1 \wedge \neg x_3)$)
  - etc.



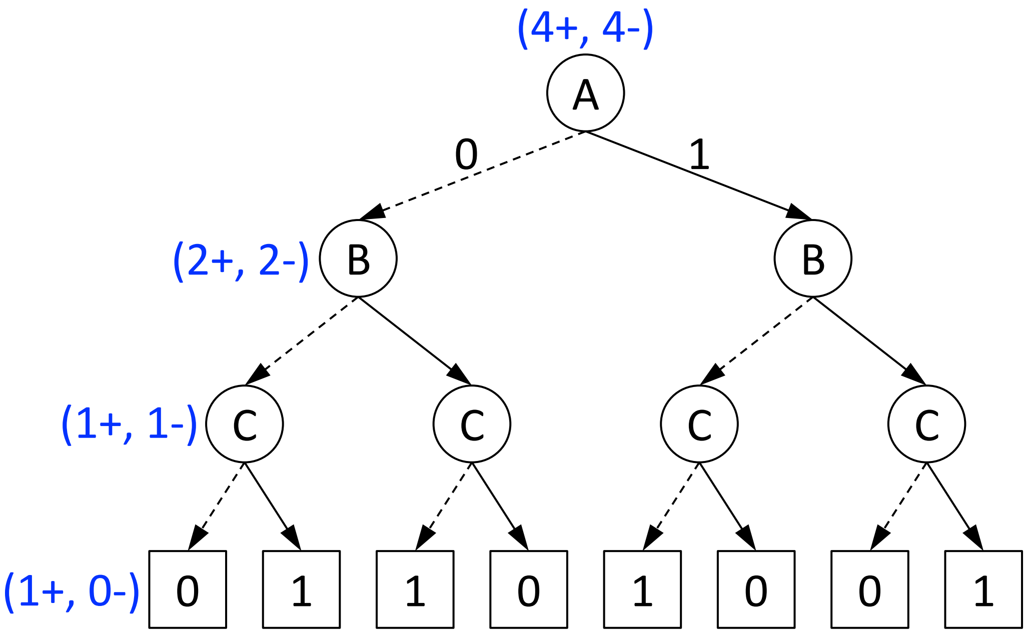




Parity function

Predictors			Resp.
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Decision Tree for parity function of 3 Boolean attributes: true iff an odd number of variables are true



$$H(Y) = \ ?$$

$$H(Y|A) = \ ?$$

$$H(Y|B) = \ ?$$

$$H(Y|A,B) = \ ?$$

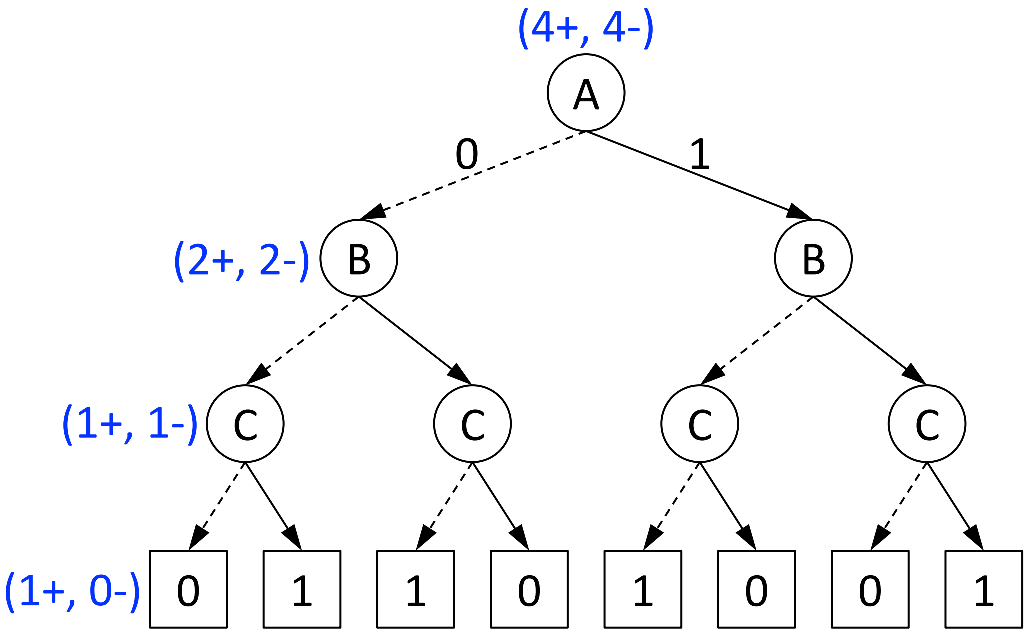
$$H(Y|A,B,C) = \ ?$$

Parity function

Predictors			Resp.
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Decision Tree for parity function of 3 Boolean attributes:
true iff an odd number of variables are true

Only combinations of attributes are informative!
Greedy can never learn that ☹



$$H(Y) = 1$$

$$H(Y|A) = 0.5 \cdot 1 + 0.5 \cdot 1 = 1$$

$$H(Y|B) = 0.5 \cdot 1 + 0.5 \cdot 1 = 1$$

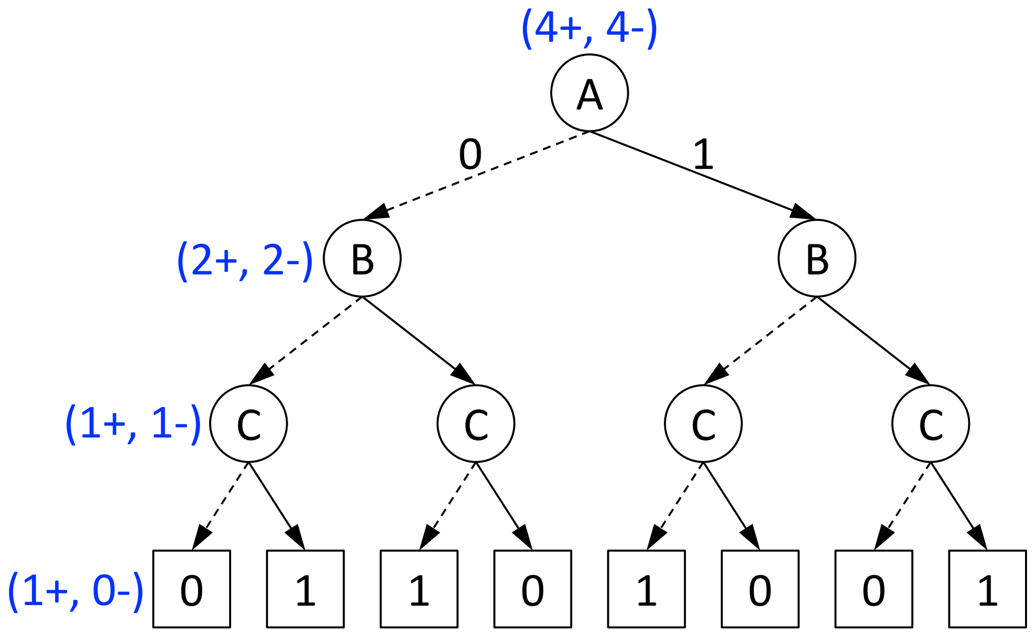
$$H(Y|A,B) = 0.5 \cdot 1 + 0.5 \cdot 1 = 1$$

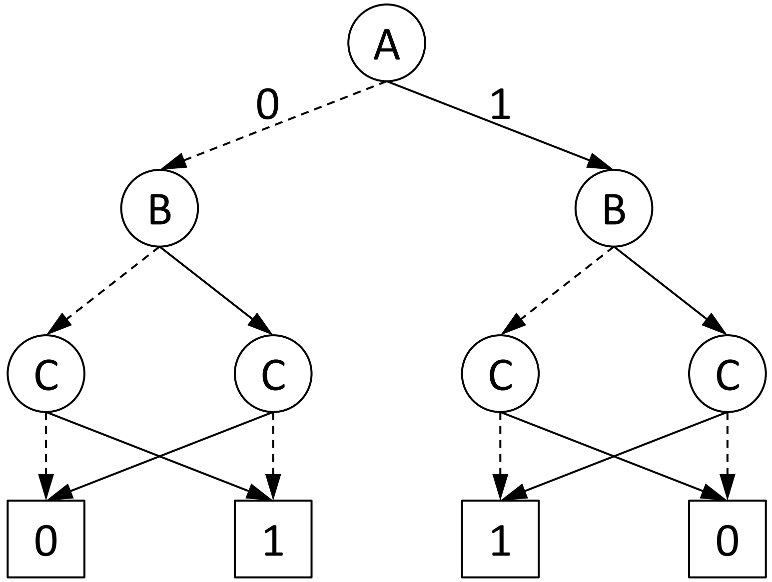
$$H(Y|A,B,C) = 0$$

Decision trees vs. circuits

Parity function

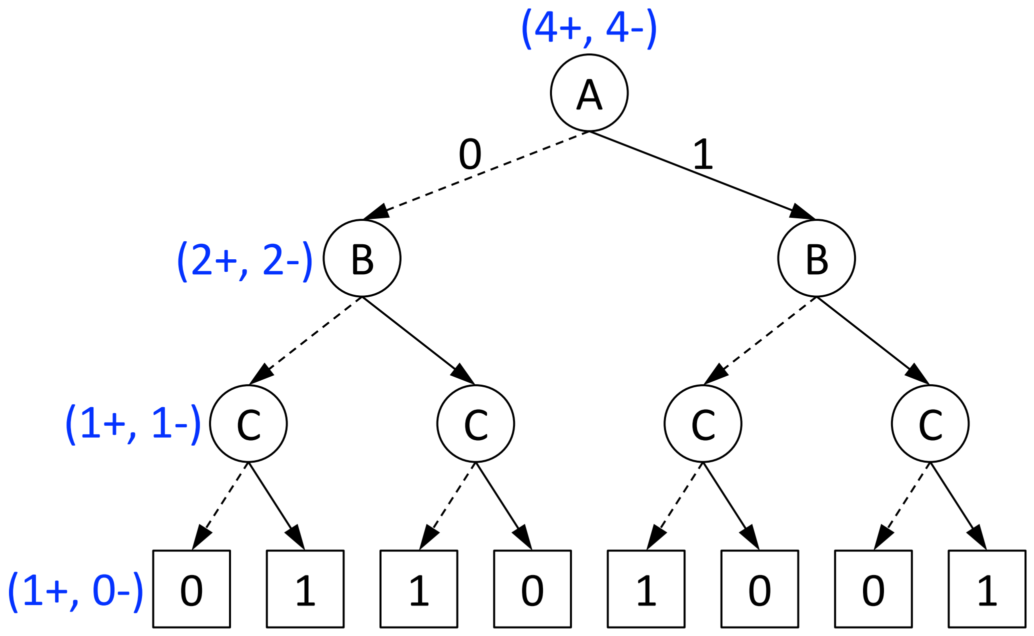
Predictors			Resp.
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

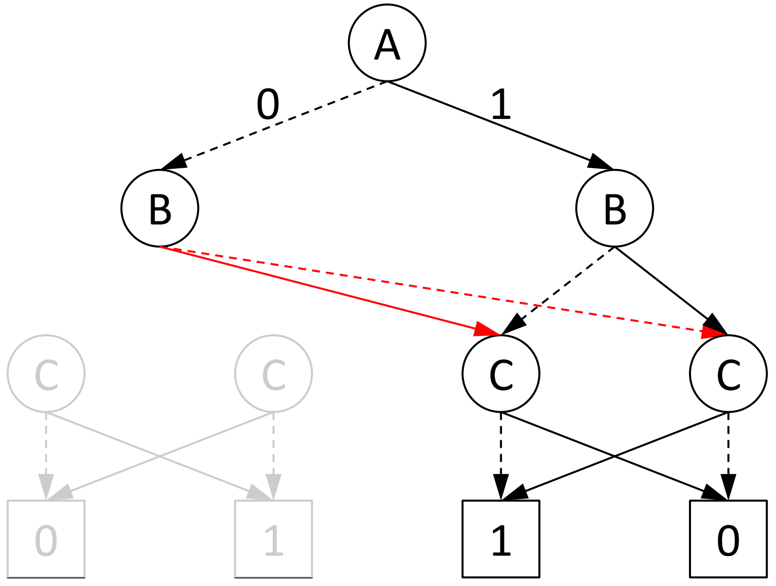




Parity function

Predictors			Resp.
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1



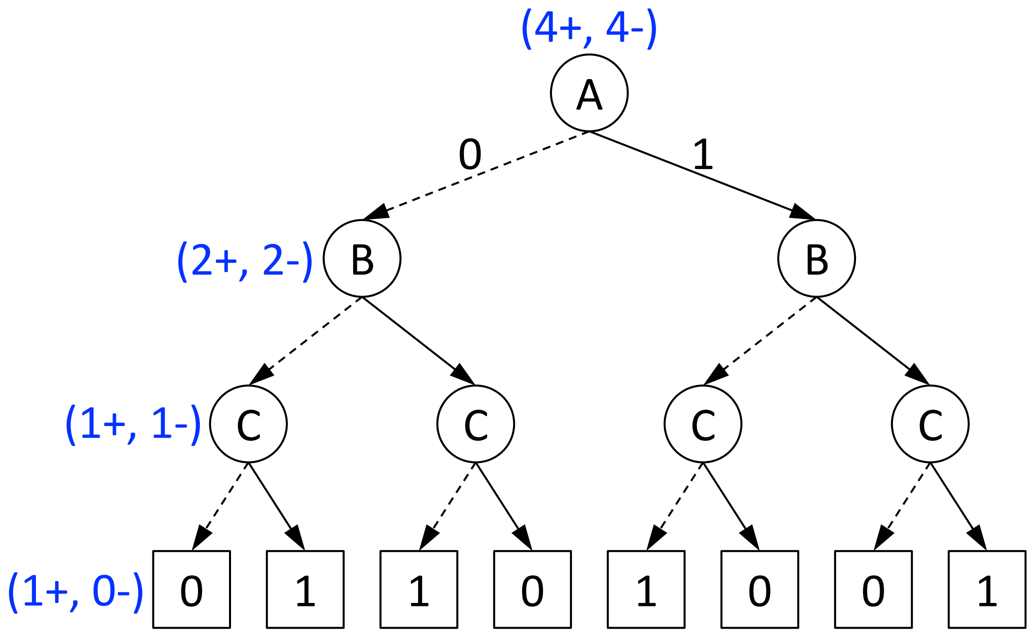


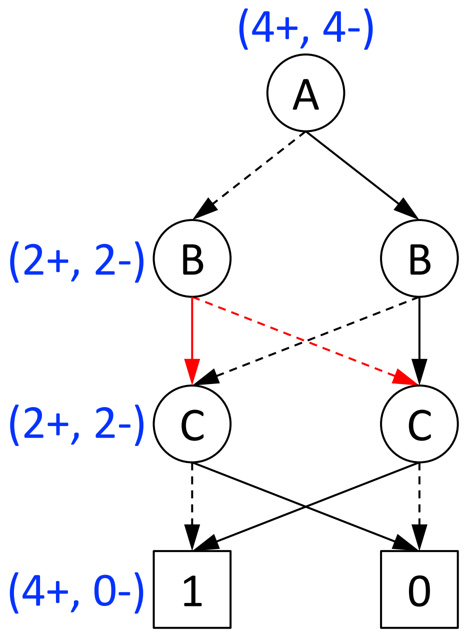
Parity function

Predictors			Resp.
A	**B**	**C**	**Y**
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

The DT grows exponentially with the number of attributes (linearly in the size of the truth table).

The OBDD (Ordered Binary Decision Diagrams) grows linearly in number of attributes (exponentially more succinct than truth table)





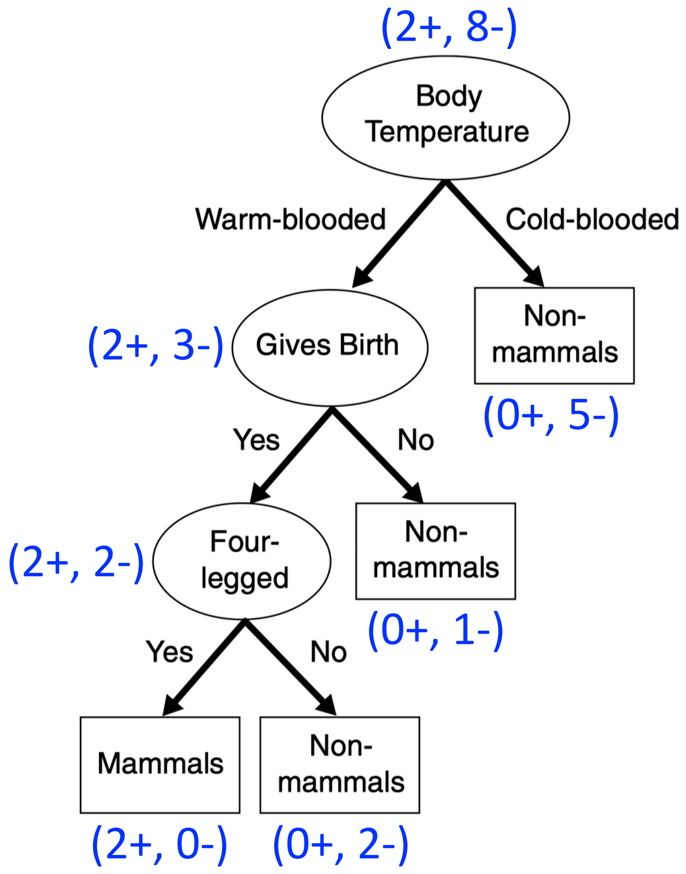
Overfitting

Overfitting due to presence of noise

Training set

name	Predictors			Label
	Body Temp.	**Gives Birth**	**4 legs**	**Mammal**
porcupine	warm	yes	yes	yes
cat	warm	yes	yes	yes
bat	warm	yes	no	no
whale	warm	yes	no	no
salamander	cold	no	yes	no
komodo dragon	cold	no	yes	no
python	cold	no	no	no
salmon	cold	no	no	no
eagle	warm	no	no	no
guppy	cold	yes	no	no

DT 1



0% training error

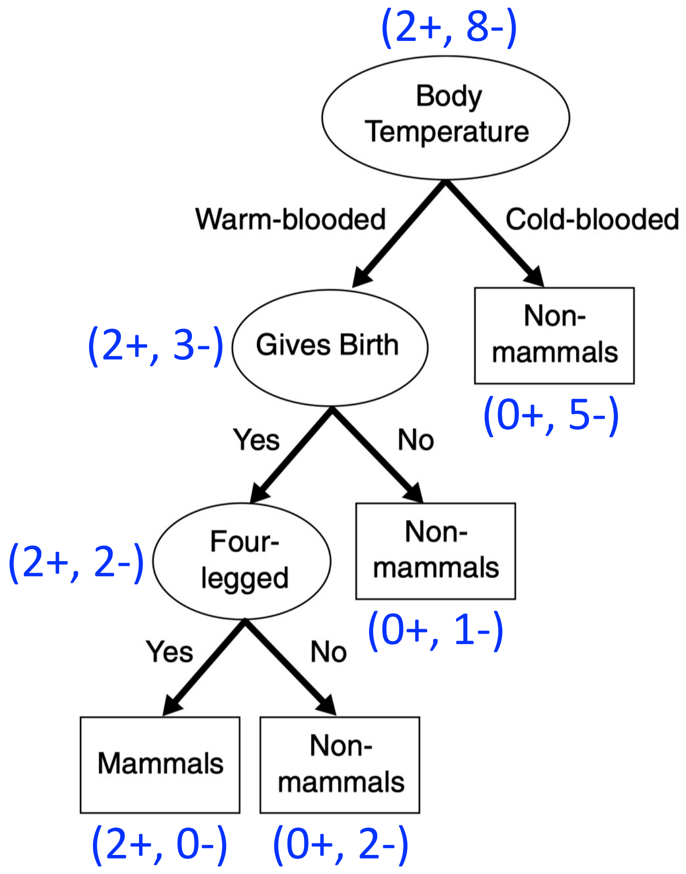
Overfitting due to presence of noise

Training set

	Predictors			Label
name	**Body Temp.**	**Gives Birth**	**4 legs**	**Mammal**
porcupine	warm	yes	yes	yes
cat	warm	yes	yes	yes
bat	warm	yes	no	no
whale	warm	yes	no	no
salamander	cold	no	yes	no
komodo dragon	cold	no	yes	no
python	cold	no	no	no
salmon	cold	no	no	no
eagle	warm	no	no	no
guppy	cold	yes	no	no

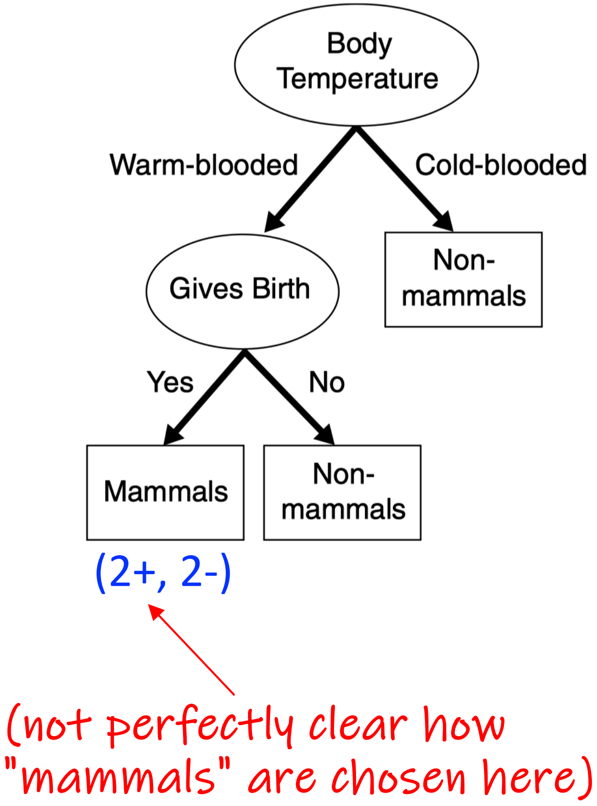
mislabeled

DT 1



0% training error

DT 2



20% training error

Overfitting due to presence of noise

Training set

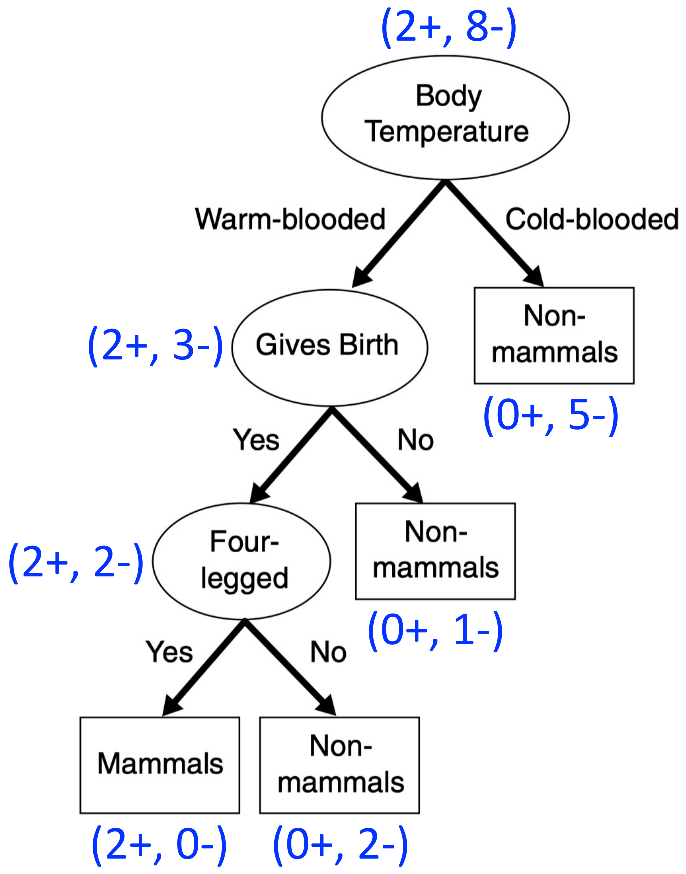
name	Body Temp.	Gives Birth	4 legs	Mammal
	Predictors			Label
porcupine	warm	yes	yes	yes
cat	warm	yes	yes	yes
bat	warm	yes	no	no
whale	warm	yes	no	no
salamander	cold	no	yes	no
komodo dragon	cold	no	yes	no
python	cold	no	no	no
salmon	cold	no	no	no
eagle	warm	no	no	no
guppy	cold	yes	no	no

mislabeled

Test set

human	warm	yes	no	yes
pigeon	warm	no	no	no
elephant	warm	yes	yes	yes
leopard shark	cold	yes	no	no
turtle	cold	no	yes	no
penguin	cold	no	no	no
eel	cold	no	no	no
dolphin	warm	yes	no	yes
spiny anteater	warm	no	yes	yes
gila monster	cold	no	yes	no

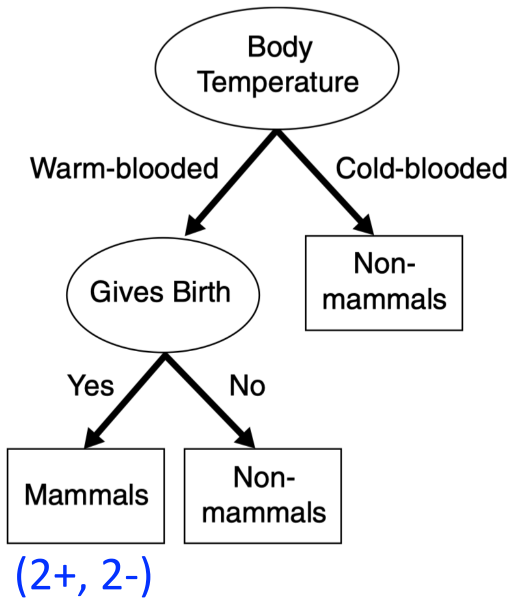
DT 1



0% training error

30% test error

DT 2



(not perfectly clear from the book why "mammals" are chosen here)

20% training error

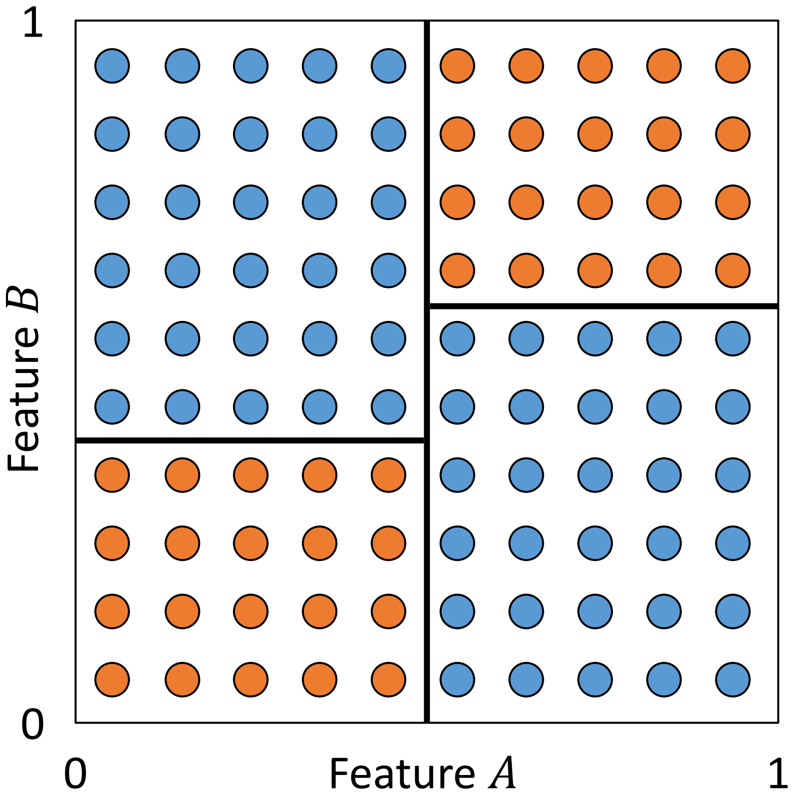
10% test error

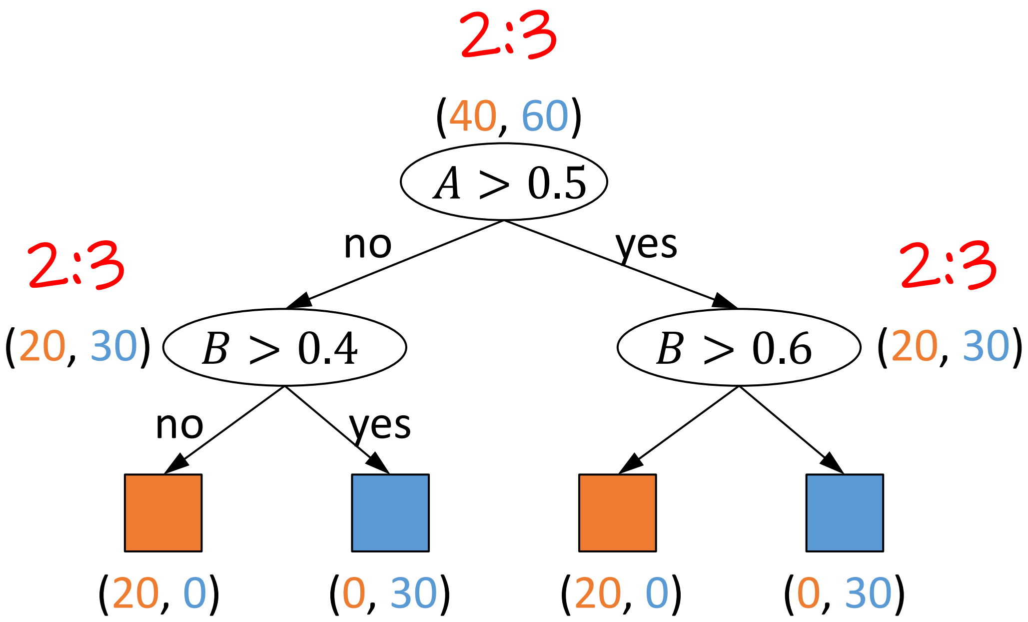
Example taken from Ch 4.4.1 of [Tan+'18] Tan, Steinbach, Karpatne, Kumar. Introduction to Data Mining, 2nd ed, 2018. https://www-users.cse.umn.edu/~kumar001/dmbook/index.php

Practical considerations

Decision Tree Classification

REPETITION

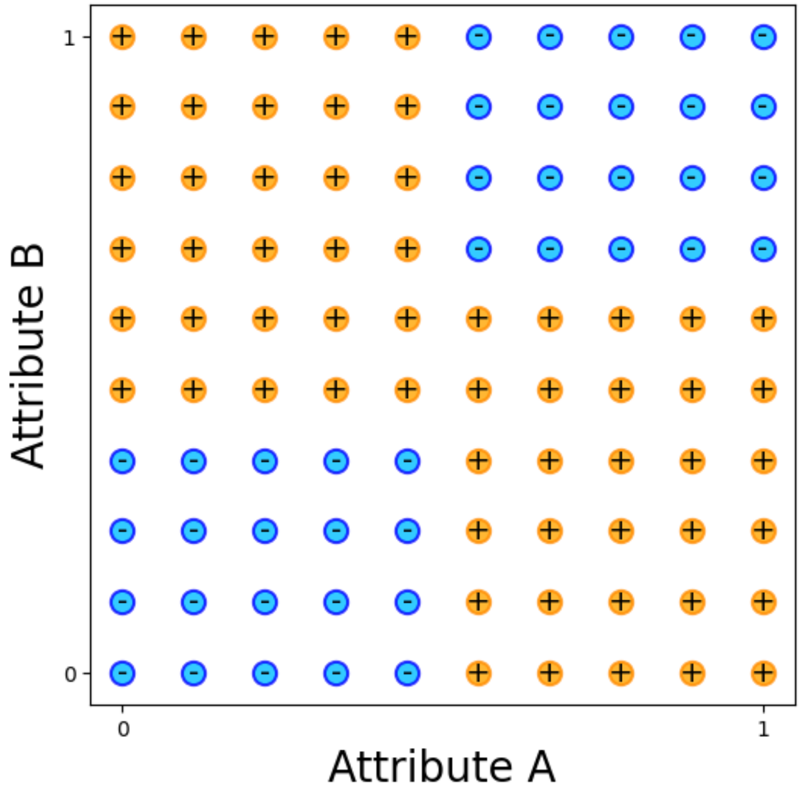




What will happen
if we apply information gain

?

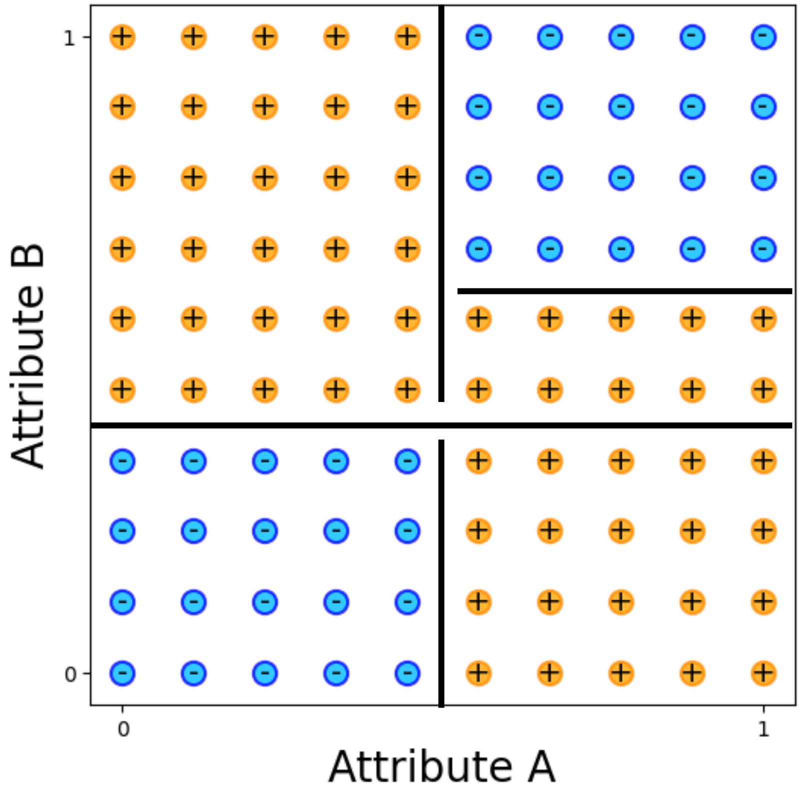
Decision Tree Classification

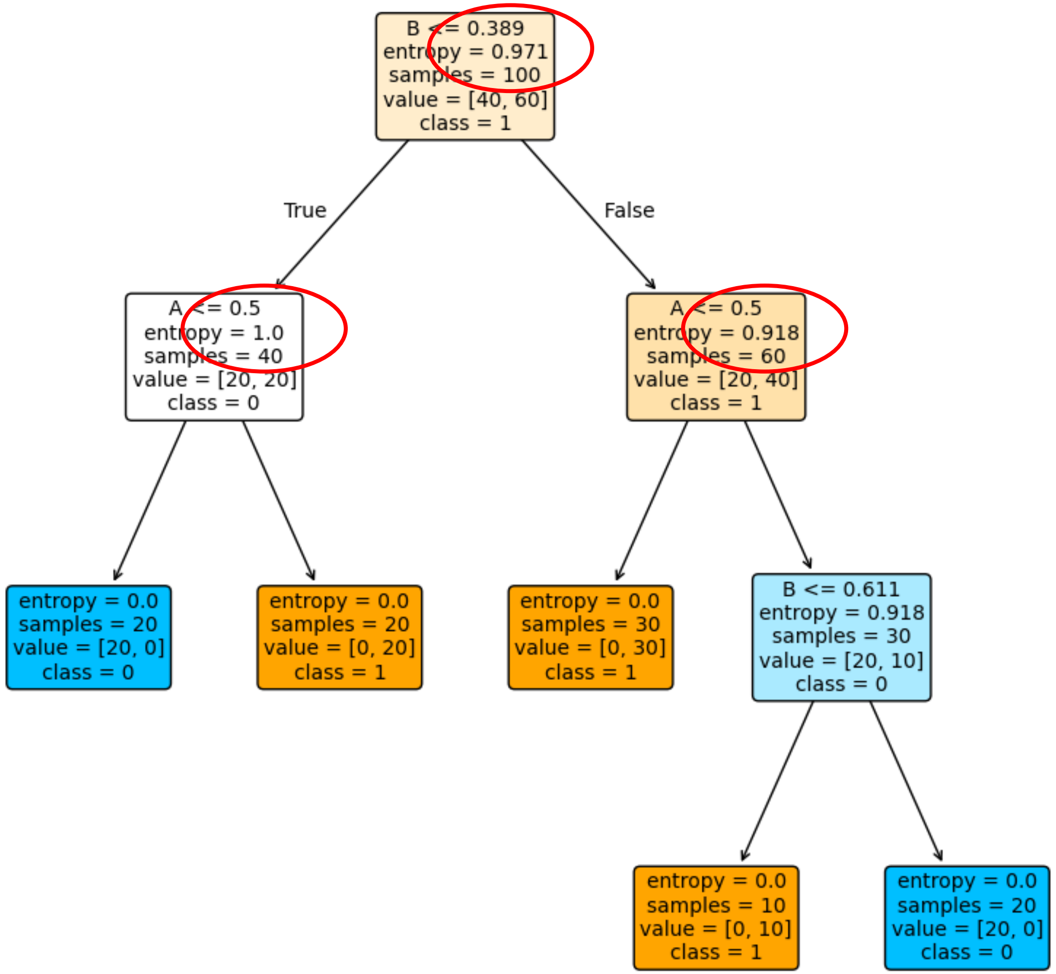


What will happen
if we apply information gain

?

Decision Tree Classification

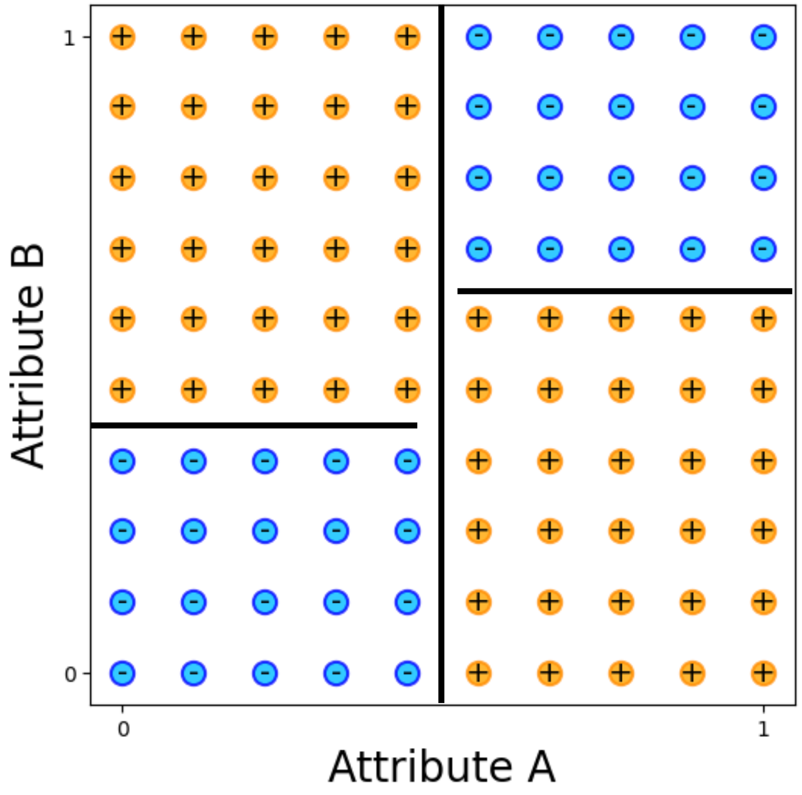




Python file 202: https://github.com/northeastern-datalab/cs7840-activities/tree/main/notebooks
Gatterbauer. Foundations and Applications of Information Theory: https://northeastern-datalab.github.io/cs7840/

Decision Tree Classification

as of October 2024



I could not figure out how to force scikit-learn to create the original smaller decision tree shown on the left (max_depth did not help either).

Can you?

?

AI Overview

While you can't directly force a decision tree in scikit-learn to use a particular attribute first, you can influence its behavior by:

1. Feature Engineering:

Create a new feature:

Combine the attribute you want to prioritize with other features or create a new feature based on its transformations. This can increase its importance in the decision-making process.

Scale the feature:

If the attribute has a different scale compared to other features, scaling it can make it more prominent in the tree's decision-making.

2. Hyperparameter Tuning:

max_depth:

Limiting the maximum depth of the tree can prevent it from exploring deeper levels where your desired attribute might be used.

min_samples_split:

This parameter sets the minimum number of samples required to split an internal node. Increasing this value can force the tree to consider attributes with higher information gain earlier.

min_samples_leaf:

This parameter sets the minimum number of samples required to be at a leaf node. Increasing this value can have a similar effect to increasing `min_samples_split`.

3. Custom Splitting Criteria:

- **Implement your own splitting criterion:** You can write a custom function to calculate the splitting criterion, giving more weight to the attribute you want to prioritize.

However, keep in mind that:

- **Decision trees are designed to find the best splits based on the data.** Forcing a specific attribute might lead to a suboptimal model.
- **The importance of an attribute depends on its relationship with the target variable.** If the attribute is not strongly correlated with the target, it might not be used even if you try to force it.

Here's an example of how to use feature engineering to influence the decision tree: