Decision Trees (DT)
Machine Learning

Dr. Amani RAAD

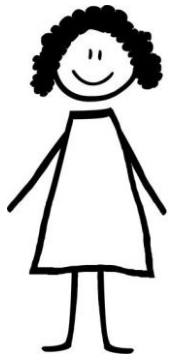
2022-2023

amani.raad@ul.edu.lb

Definition

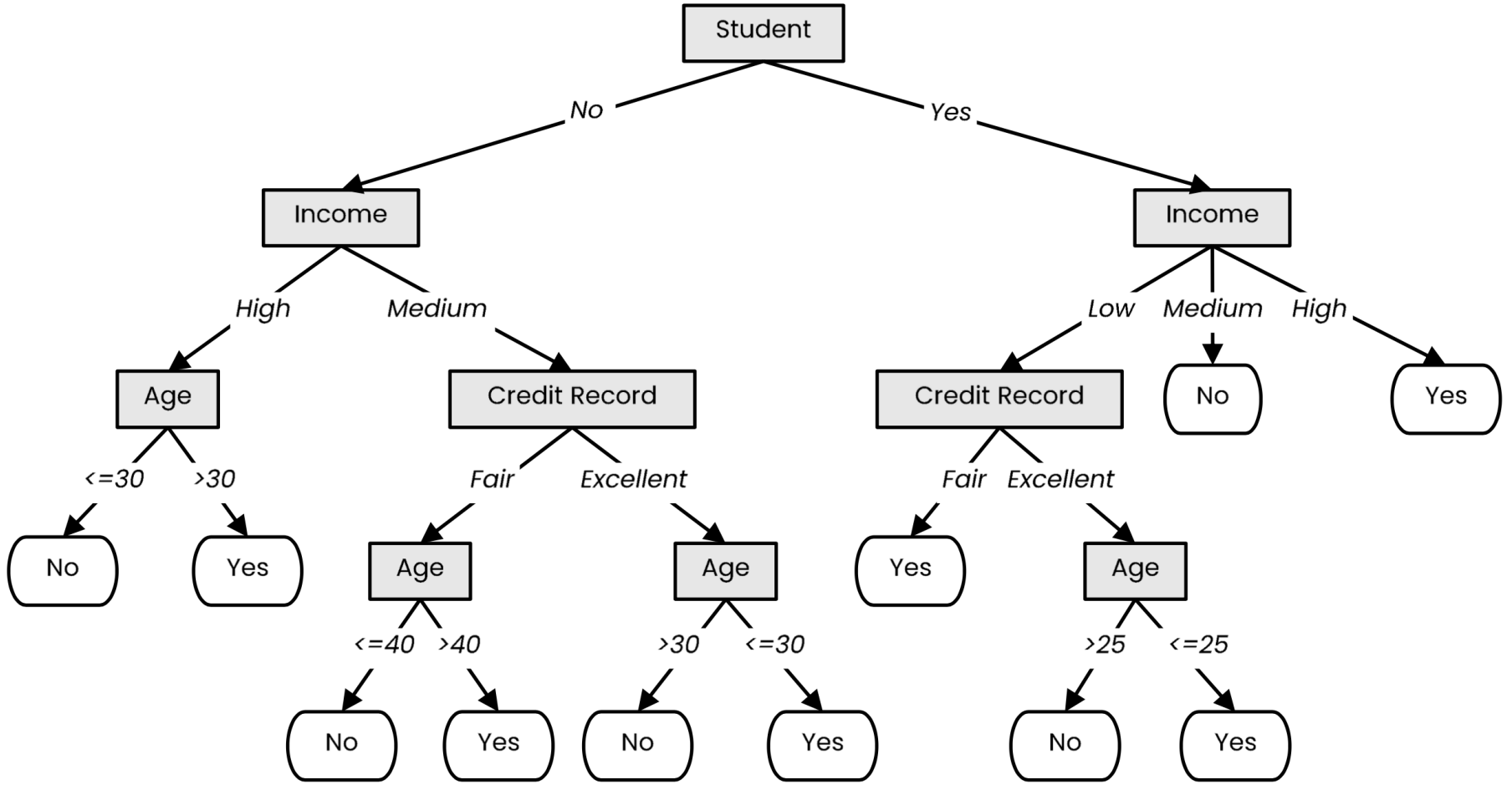
- A tree-like model that illustrates series of events leading to certain decisions
- Each node represents a test on an attribute and each branch is an outcome of that test

Who to loan?



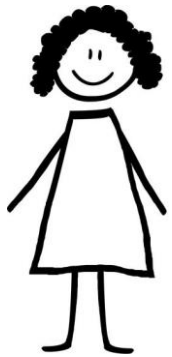
- Not a student
- 45 years old
- Medium income
- Fair credit record

- Student
- 27 years old
- Low income
- Excellent credit record



Definition

- A tree-like model that illustrates series of events leading to certain decisions
- Each node represents a test on an attribute and each branch is an outcome of that test

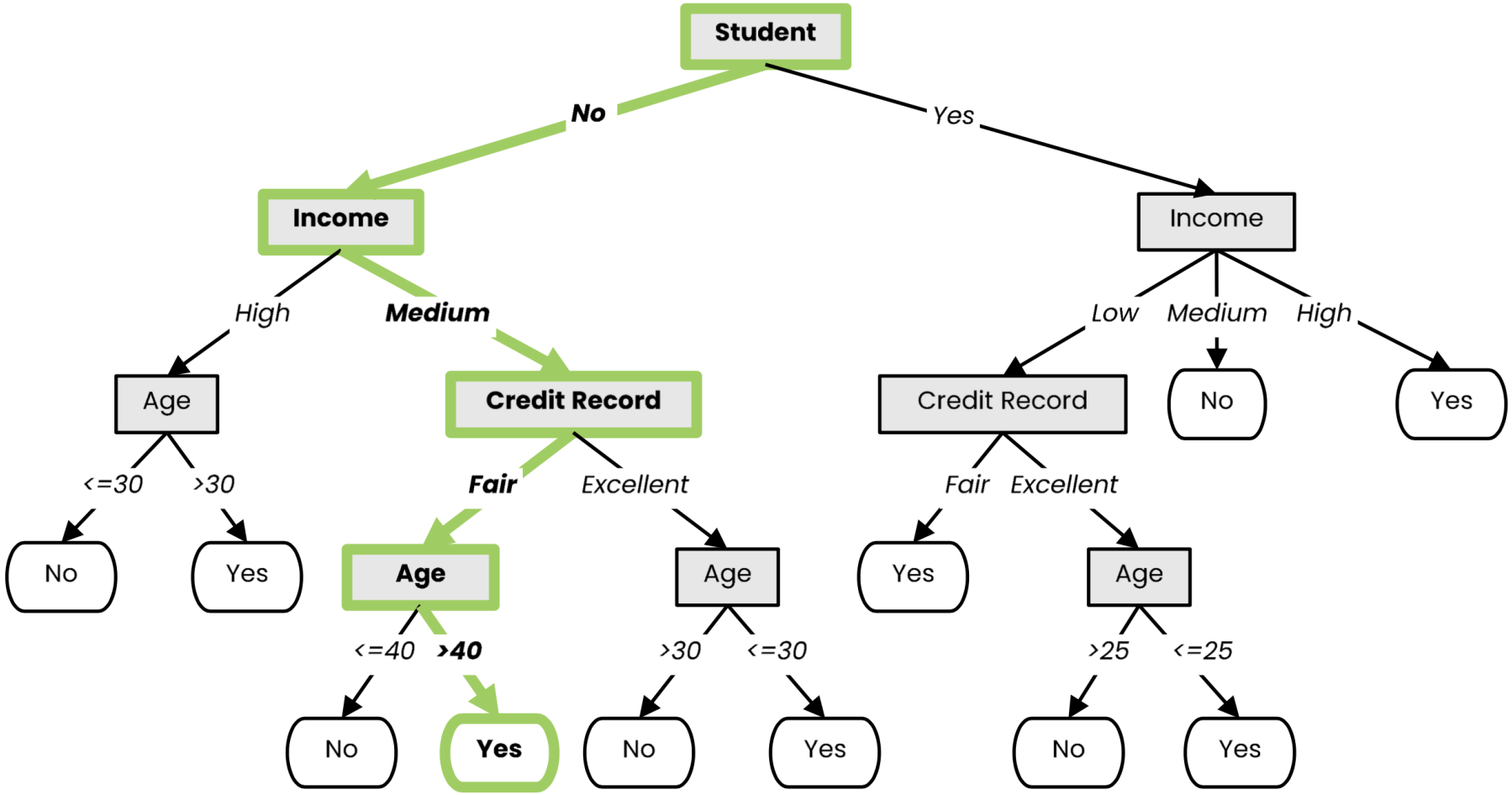


Who to loan?

- Not a student
- 45 years old
- Medium income
- Fair credit record

➢ Yes

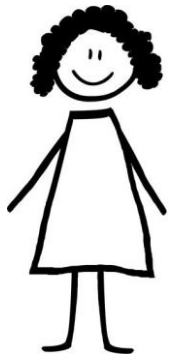
- Student
- 27 years old
- Low income
- Excellent credit record



Definition

- A tree-like model that illustrates series of events leading to certain decisions
- Each node represents a test on an attribute and each branch is an outcome of that test

Who to loan?

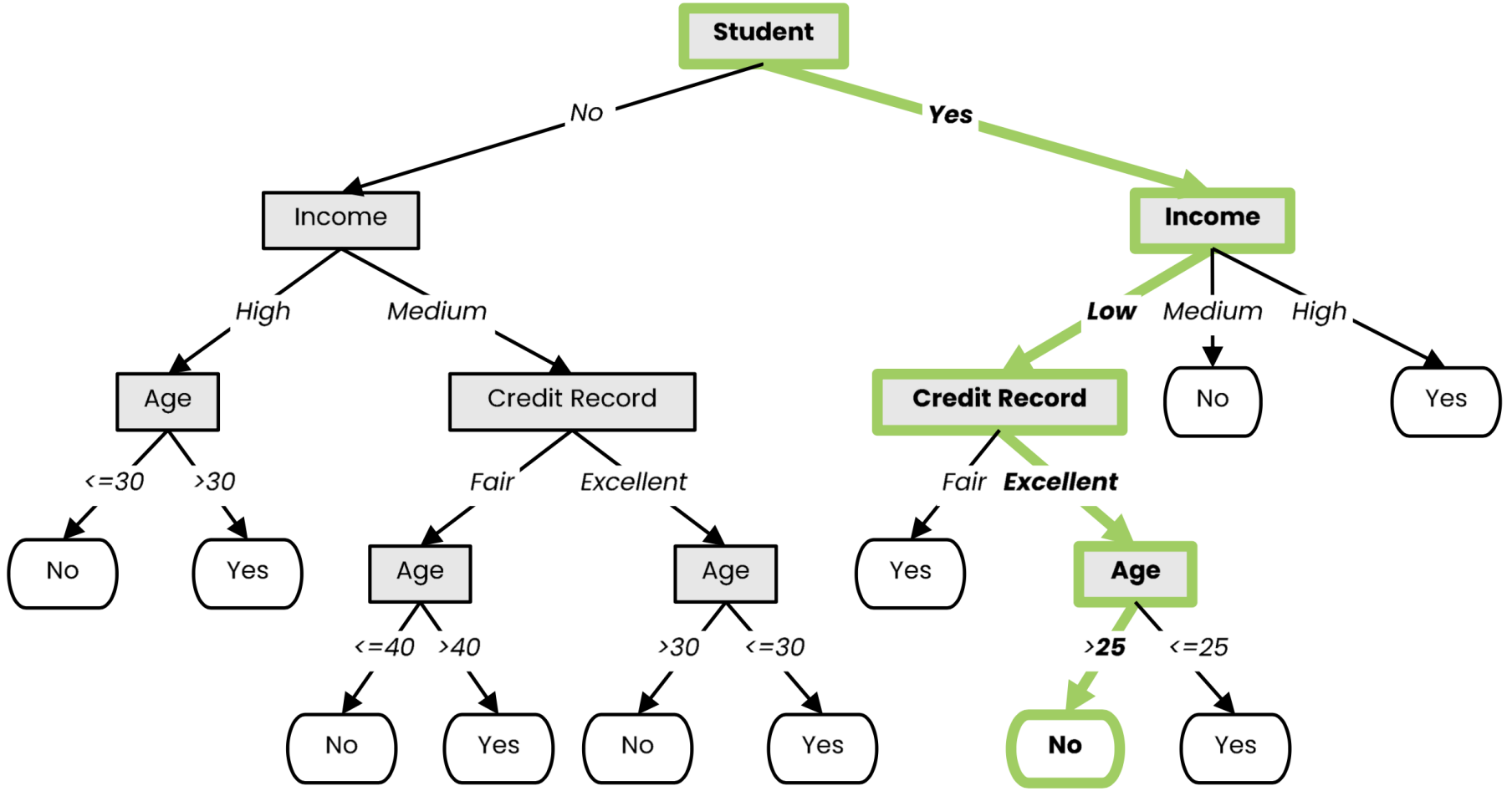


- Not a student
- 45 years old
- Medium income
- Fair credit record

➢ Yes

- Student
- 27 years old
- Low income
- Excellent credit record

➢ No



Why trees?

- Decision Trees (DTs) are a supervised learning technique that can be used in both a regression and a classification context. For this reason they are sometimes also referred to as **Classification And Regression Trees (CART)**.
- Interpretable/intuitive, popular in medical applications because they mimic the way a doctor thinks
- Model discrete outcomes nicely
- Can be very powerful, can be as complex as you need them
- C4.5 and CART - from “top 10” - decision trees are very popular
- BP’s GasOIL system for separating gas and oil on offshore platforms - decision trees replaced a hand-designed rules system with 2500 rules. C4.5-based system outperformed human experts and saved BP millions. (1986)
- can also learn to play tennis, analyze C-section risk, etc.

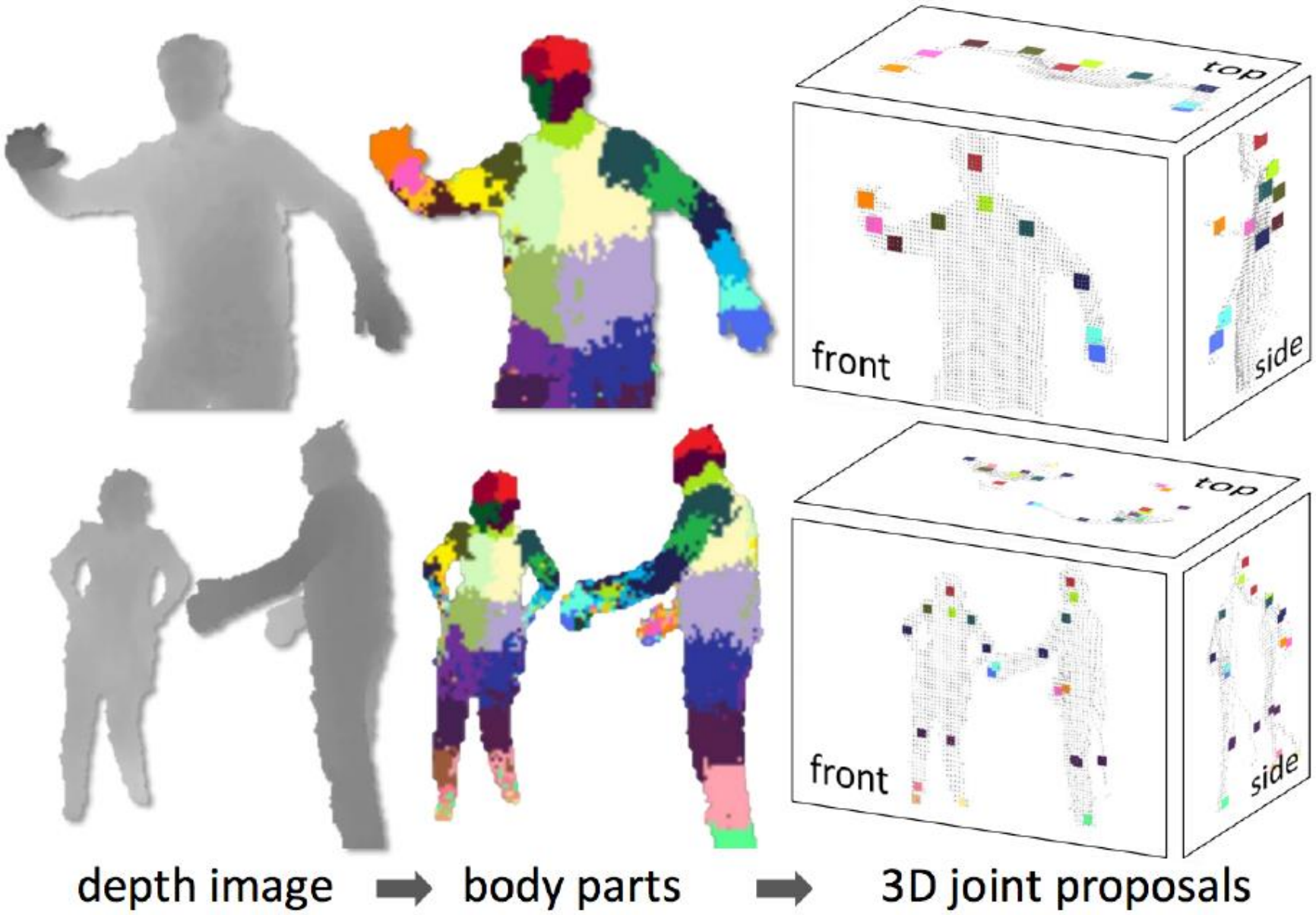
Decision Tree Types

- Classification tree analysis. Iterative Dichotomiser 3 (ID3), C4.5, (Quinlan, 1986)
- Classification And Regression Tree (CART) analysis is used to refer to both of procedures, first introduced by (Breiman et al., 1984)
- A Random Forest classifier uses a number of decision trees, in order to improve the classification rate.
- Boosting Trees can be used for regression-type and classification-type problems.

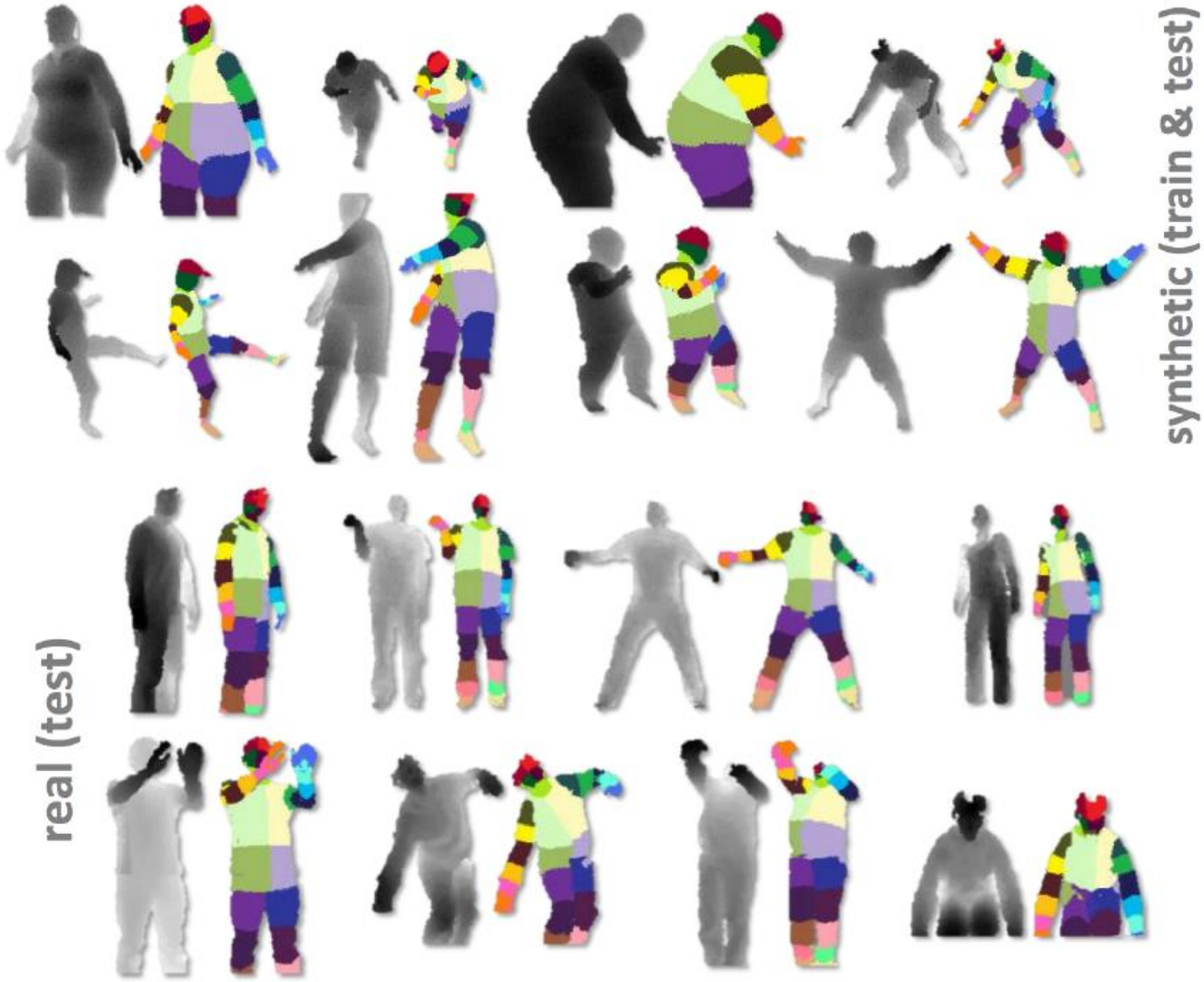
Applications of Decision Trees: XBox!

- Decision trees are in XBox

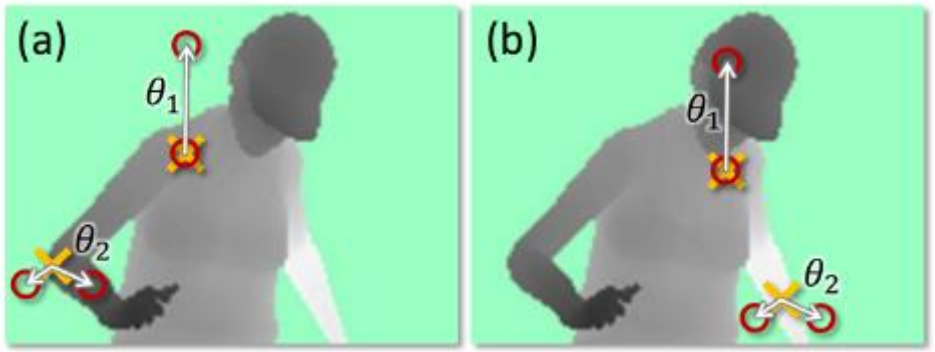
- Decision trees are in XBox: Classifying body parts

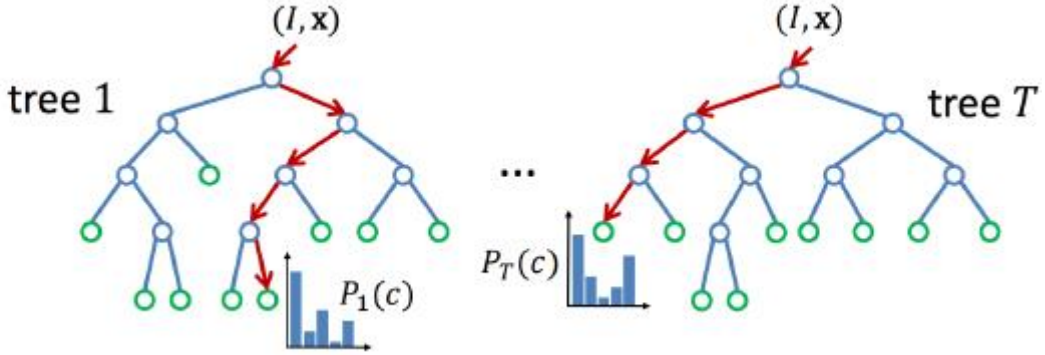


- Trained on million(s) of examples

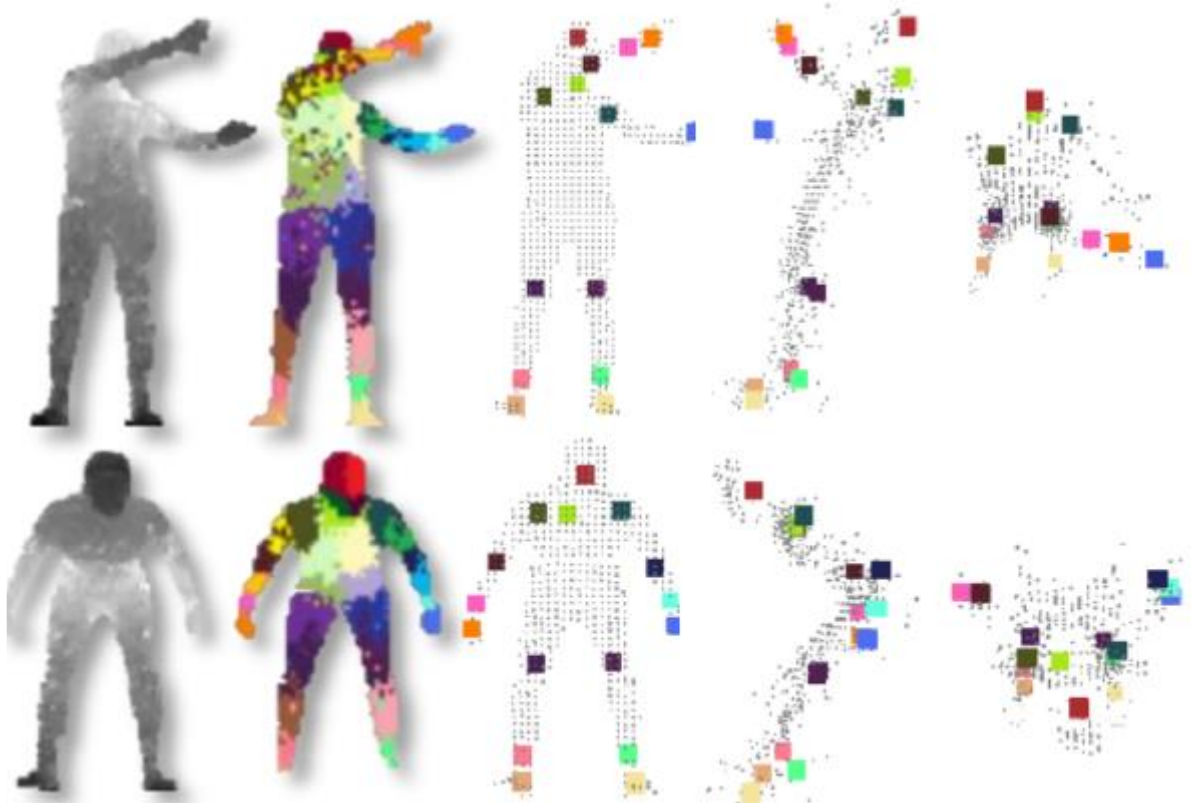


- Trained on million(s) of examples





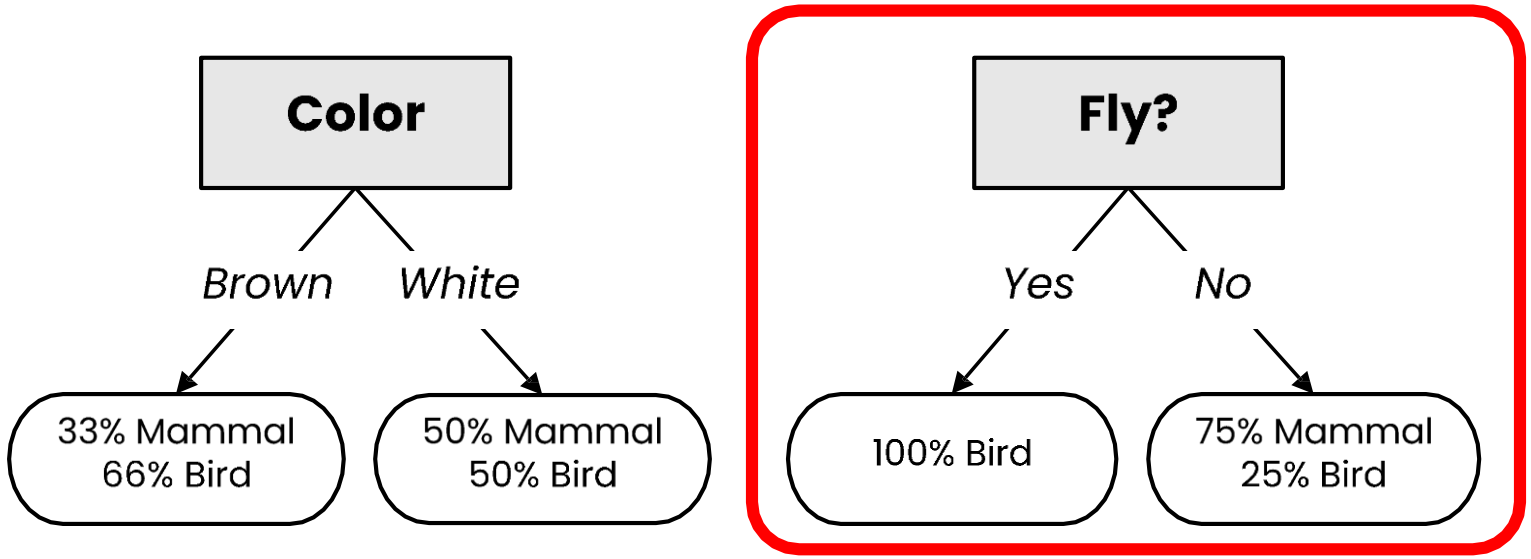
- Results:



How to build a decision tree?

What is a good attribute?

Does it fly?	Color	Class
No	Brown	**Mammal**
No	White	**Mammal**
Yes	Brown	**Bird**
Yes	White	**Bird**
No	White	**Mammal**
No	Brown	**Bird**
Yes	White	**Bird**



- Which attribute provides better splitting?
- Why?
  - Because the resulting subsets are more pure
  - Knowing the value of this attribute gives us more information about the label
    (the entropy of the subsets is lower)

Example: Will the customer wait for a table?

- Here are the attributes:

1.	Alternate: whether there is a suitable alternative restaurant nearby.
2.	Bar: whether the restaurant has a comfortable bar area to wait in.
3.	Fri/Sat: true on Fridays and Saturdays.
4.	Hungry: whether we are hungry.
5.	Patrons: how many people are in the restaurant (values are None, Some, and Full).
6.	Price: the restaurant's price range ($, $$, $$$).
7.	Raining: whether it is raining outside.
8.	Reservation: whether we made a reservation.
9.	Type: the kind of restaurant (French, Italian, Thai or Burger).
10.	WaitEstimate: the wait estimated by the host (0-10 minutes, 10-30, 30-60, >60).

Image by MIT OpenCourseWare, adapted from Russell and Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, 2009.

- Here are the examples:

Example	Attributes										Goal
	Alt	Bar	Fri	Hun	Pat	Price	Rain	Res	Type	Est	WillWait
X_1	Yes	No	No	Yes	Some	$$$	No	Yes	French	0-10	Yes
X_2	Yes	No	No	Yes	Full	$	No	No	Thai	30-60	No
X_3	No	Yes	No	No	Some	$	No	No	Burger	0-10	Yes
X_4	Yes	No	Yes	Yes	Full	$	No	No	Thai	10-30	Yes
X_5	Yes	No	Yes	No	Full	$$$	No	Yes	French	>60	No
X_6	No	Yes	No	Yes	Some	$$	Yes	Yes	Italian	0-10	Yes
X_7	No	Yes	No	No	None	$	Yes	No	Burger	0-10	No
X_8	No	No	No	Yes	Some	$$	Yes	Yes	Thai	0-10	Yes
X_9	No	Yes	Yes	No	Full	$	Yes	No	Burger	>60	No
X_{10}	Yes	Yes	Yes	Yes	Full	$$$	No	Yes	Italian	10-30	No
X_{11}	No	No	No	No	None	$	No	No	Thai	0-10	No
X_{12}	Yes	Yes	Yes	Yes	Full	$	No	No	Burger	30-60	Yes

Image by MIT OpenCourseWare, adapted from Russell and Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, 2009.

- Here are two options for the first feature to split at the top of the tree. Which one should we choose? Which one gives me the most information?

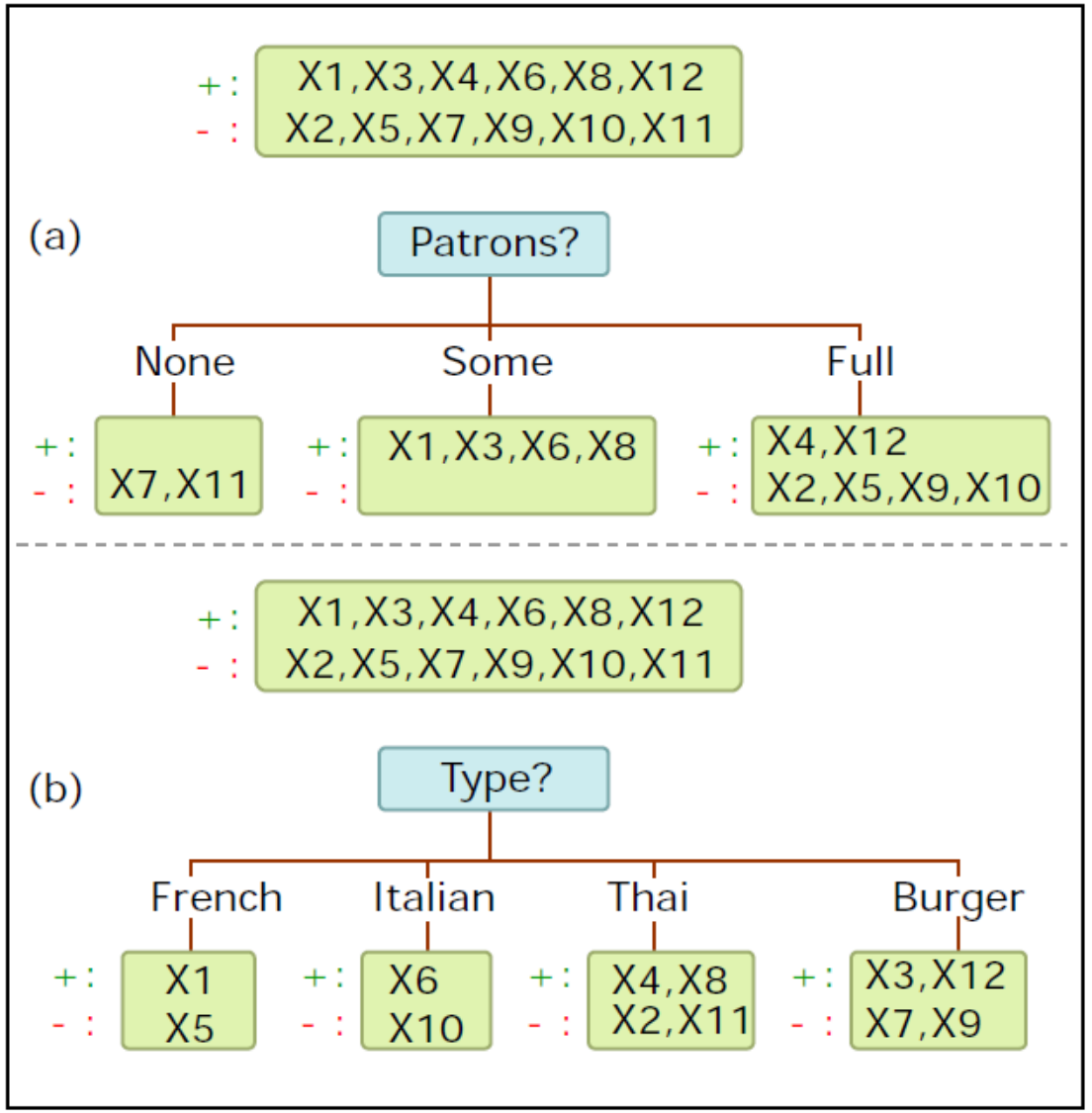


Image by MIT OpenCourseWare, adapted from Russell and Norvig, *Artificial Intelligence: A Modern Approach*, Prentice Hall, 2009.

Information Gain

Entropy

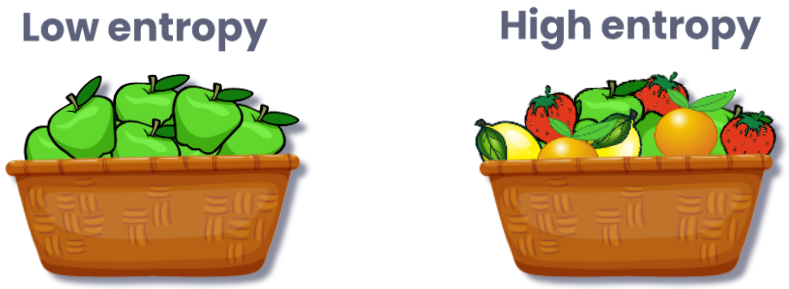
Shannon Entropy, Information Gain, and Picking Balls from Boxes

In his 1948 paper "A Mathematical Theory of Communication", Claude Shannon introduced the revolutionary notion of *Information Entropy*.

Claude Shannon

Entropy

- Entropy measures the degree of randomness in data

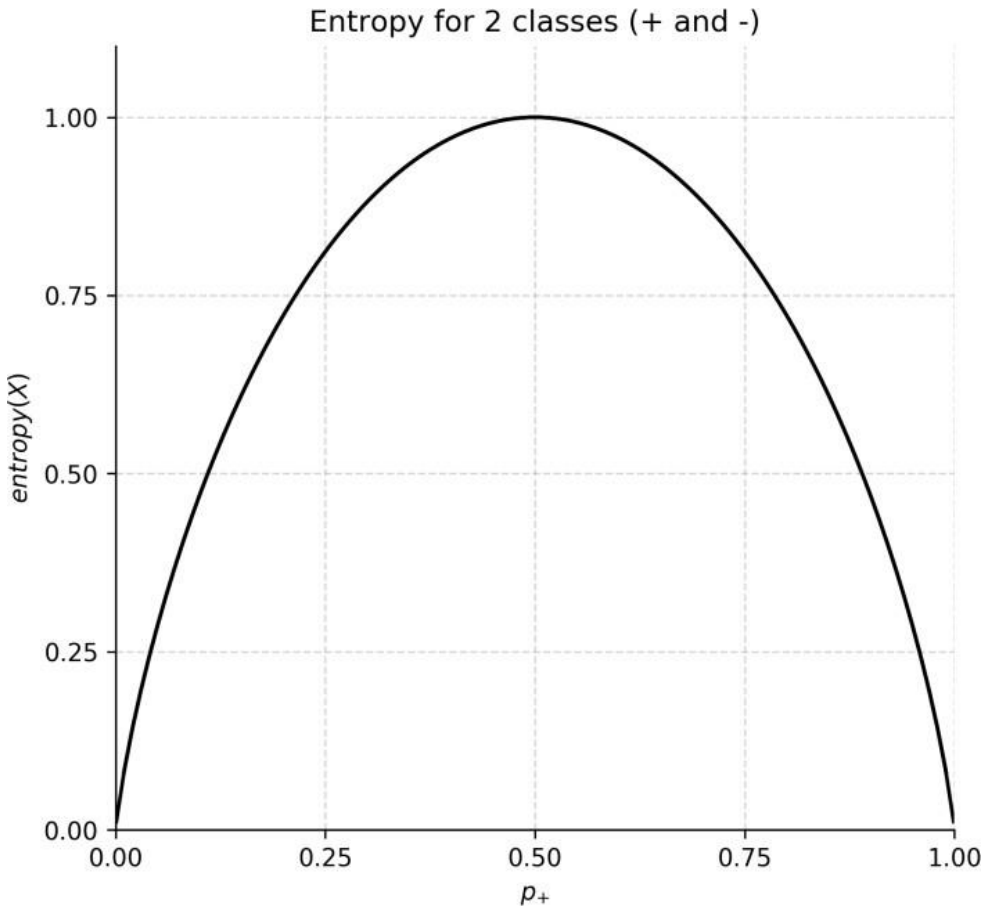


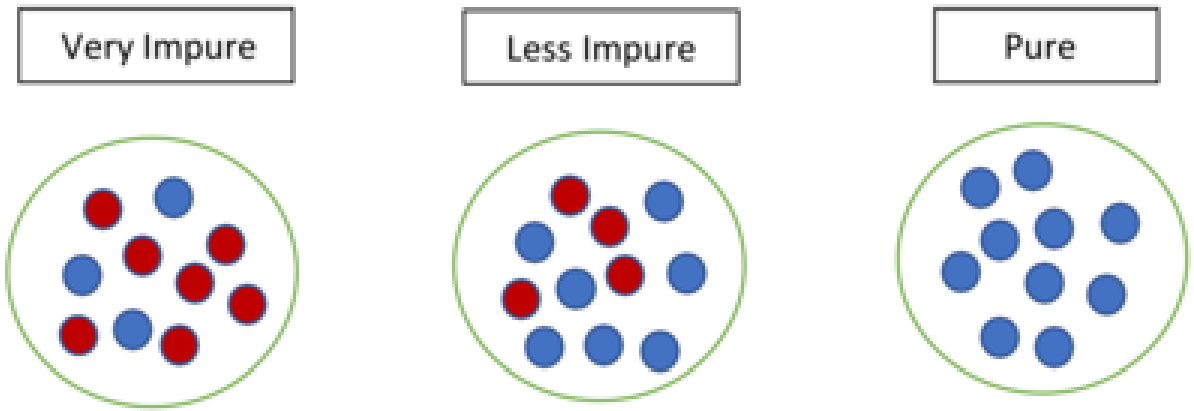
- For a set of samples X with k classes:

$$H = entropy(X) = -\sum_{i=1}^{k} p_i \log_2(p_i)$$

where p_i is the proportion of elements of class i

- Lower entropy implies greater predictability!





High entropy less entropy zero entropy

- The information gain of an attribute a is the expected reduction in entropy due to splitting on values of a
  - How much information about cloudiness do we get by discovering whether it is raining?

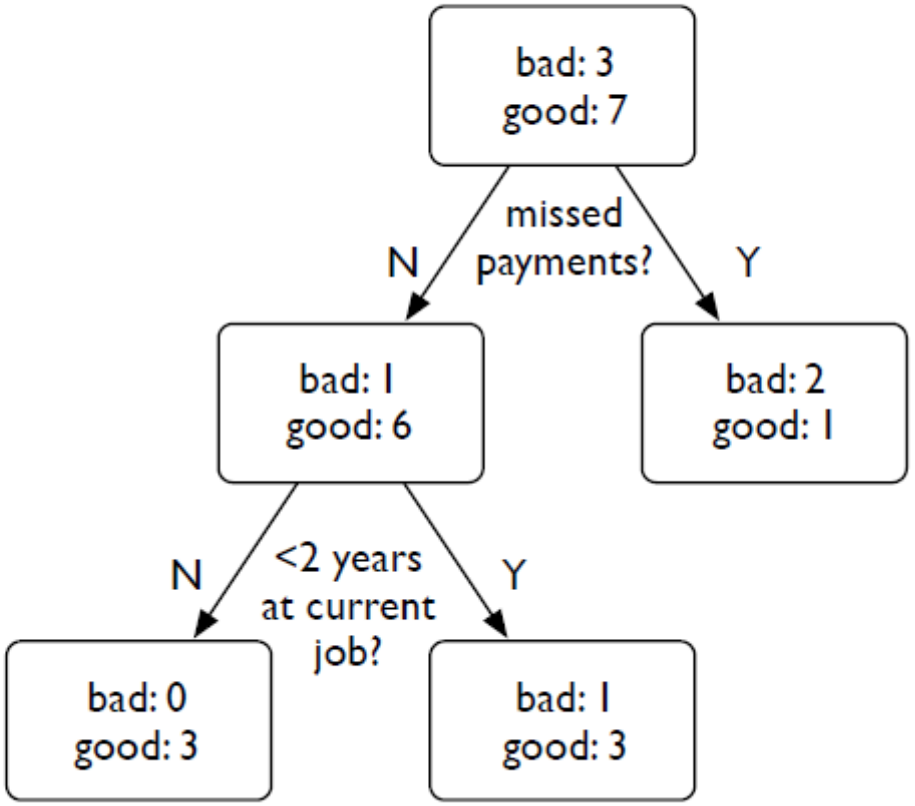
$$IG(Y|X) = H(Y) - H(Y|X) \approx 0.25\ bits$$

  - Also called **information gain** in Y due to X
  - If X is completely uninformative about Y : IG (Y | X) = 0
  - If X is completely informative about Y : IG (Y | X) = H(Y)

Credit Risk Example

Predicting credit risk

<2 years at current job?	missed payments?	defaulted?
N	N	N
Y	N	Y
N	N	N
N	N	N
N	Y	Y
Y	N	N
N	Y	N
N	Y	Y
Y	N	N
Y	N	N



- How many bits does it take to specify the attribute of 'defaulted?'
  - P(defaulted = Y) = 3/10
  - P(defaulted = N) = 7/10

$$
\begin{aligned}
H(Y) &= -\sum_{i=\mathrm{Y,N}} P(Y=y_i)\log_2 P(Y=y_i) \\
&= -0.3\log_2 0.3 - 0.7\log_2 0.7 \\
&= 0.8813
\end{aligned}
$$

- How much can we *reduce* the entropy (or uncertainty) of 'defaulted' by knowing the other attributes?
- Ideally, we could reduce it to zero, in which case we classify perfectly.

Predicting credit risk

<2 years at current job?	missed payments?	defaulted?
N	N	N
Y	N	Y
N	N	N
N	N	N
N	Y	Y
Y	N	N
N	Y	N
N	Y	Y
Y	N	N
Y	N	N

$$
\begin{aligned}
H(Y|X) &\equiv -\sum_x P(x) \sum_y P(y|x) \log_2 P(y|x) \\
&= -\sum_x P(x) \sum_y P(Y=y|X=x) \log_2 P(Y=y|X=x) \\
&= -\sum_x P(x) \sum_y H(Y|X=x)
\end{aligned}
$$

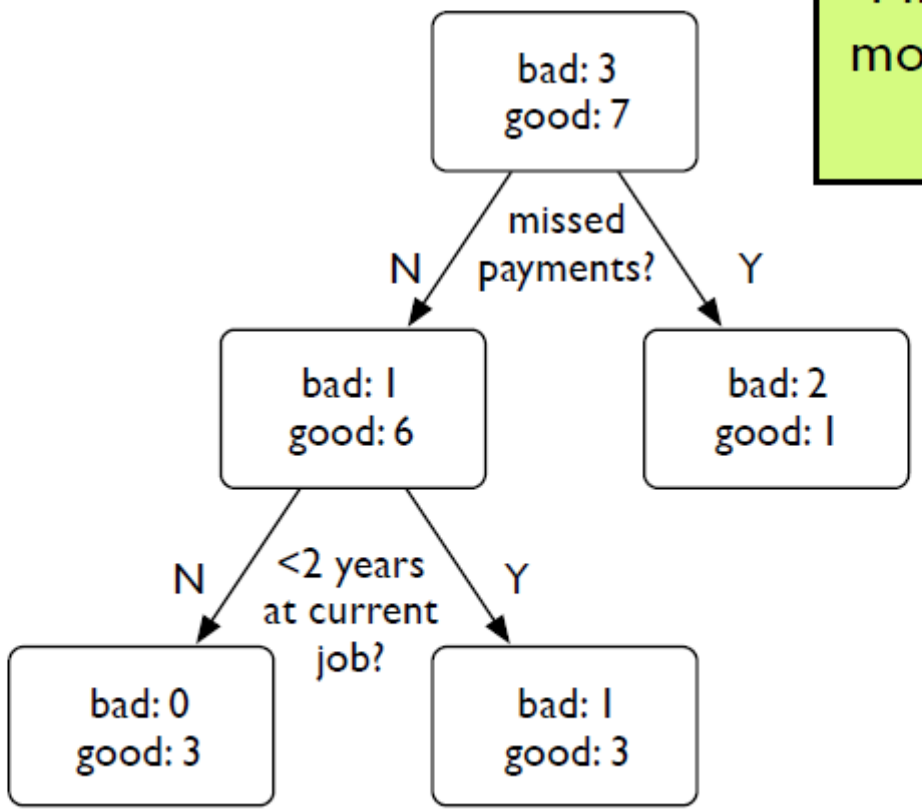
$$
\begin{aligned}
H(\text{defaulted}|<2\text{years}=\text{N}) &= -\frac{4}{4+2}\log_2\frac{4}{4+2} - \frac{2}{6}\log_2\frac{2}{6} = 0.9183 \\
H(\text{defaulted}|<2\text{years}=\text{Y}) &= -\frac{3}{4}\log_2\frac{3}{4} - \frac{1}{4}\log_2\frac{1}{4} = 0.8133 \\
H(\text{defaulted}|<2\text{yr}) &= \frac{6}{10}0.9183 + \frac{4}{10}0.8133 = 0.8763
\end{aligned}
$$

$$
\begin{aligned}
H(\text{defaulted}|\text{missed}=\text{N}) &= -\frac{6}{7}\log_2\frac{6}{7} - \frac{1}{7}\log_2\frac{1}{7} = 0.5917 \\
H(\text{defaulted}|\text{missed}=\text{Y}) &= -\frac{1}{3}\log_2\frac{1}{3} - \frac{2}{3}\log_2\frac{2}{3} = 0.9183 \\
H(\text{defaulted}|\text{missed}) &= \frac{7}{10}0.5917 + \frac{3}{10}0.9183 = 0.6897
\end{aligned}
$$

$$H(\text{defaulted}) - H(\text{defaulted}|< 2 \text{ years})$$
$$0.8813 - 0.8763 = 0.0050$$

$$H(\text{defaulted}) - H(\text{defaulted}|\text{missed})$$
$$0.8813 - 0.6897 = 0.1916$$

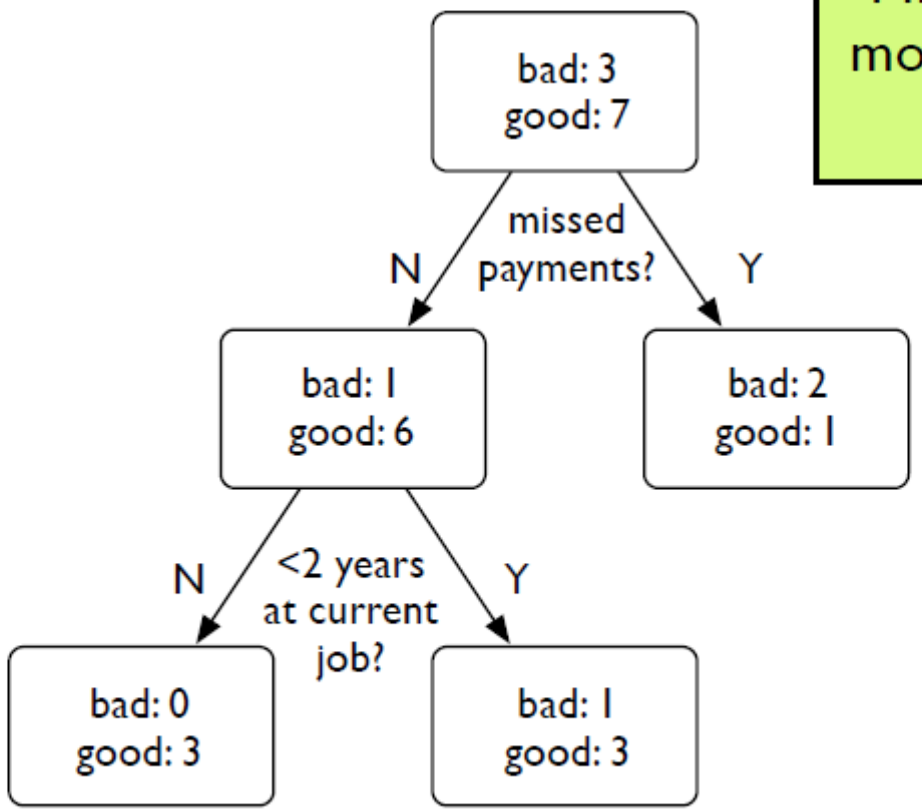
Missed payments are the most informative attribute about defaulting.



$$H(\text{defaulted}) - H(\text{defaulted}|< 2 \text{ years})$$
$$0.8813 - 0.8763 = 0.0050$$

$$H(\text{defaulted}) - H(\text{defaulted}|\text{missed})$$
$$0.8813 - 0.6897 = 0.1916$$

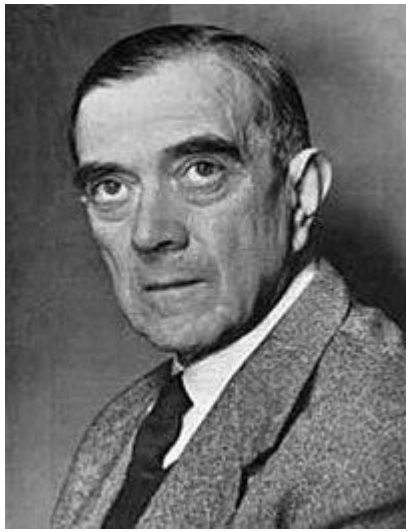
Missed payments are the most informative attribute about defaulting.



ID3 Algorithm (Python)

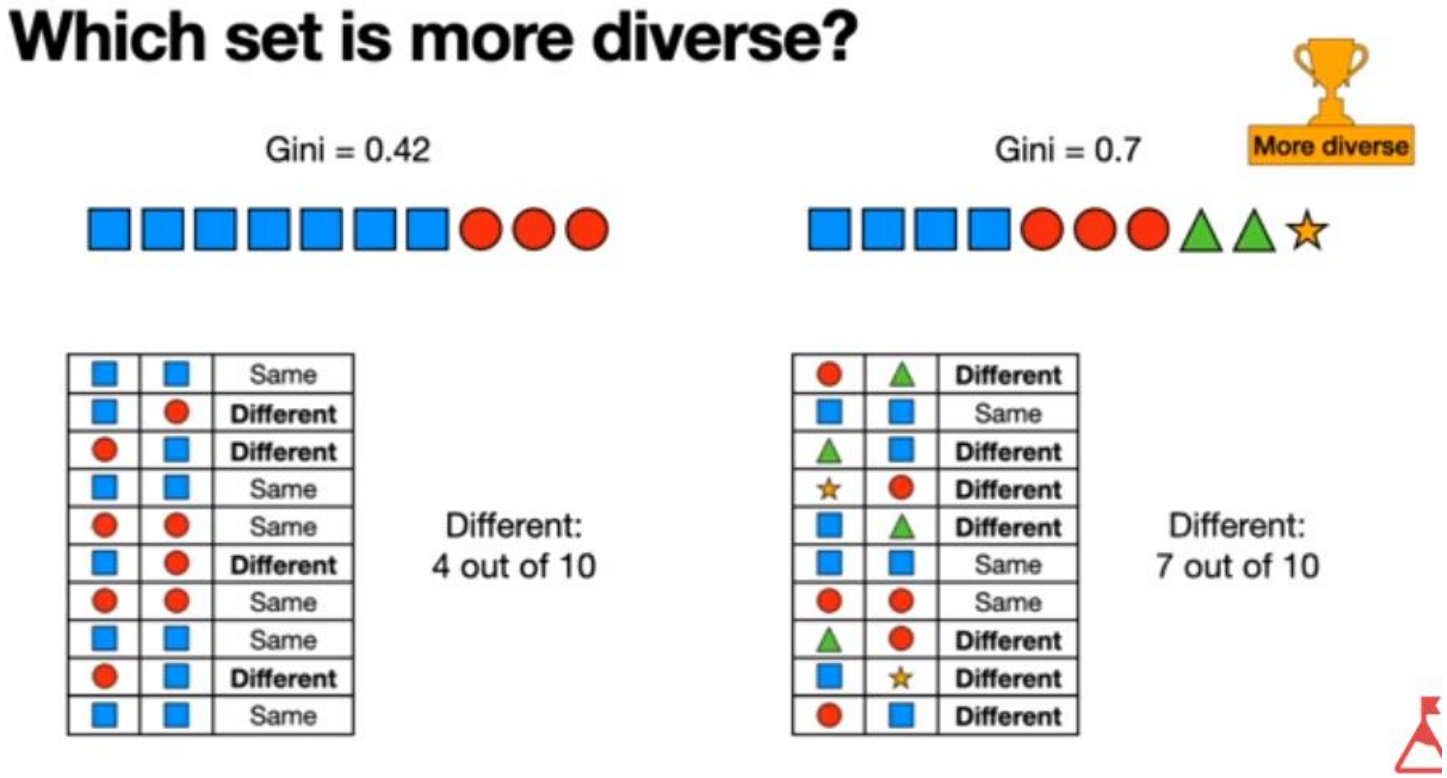
```
# ID = Iterative Dichotomiser
def ID3(X):
    node = TreeNode(X)
    if all_points_have_same_class(X):
        node.label = majority_label(X)
    else:
        a = select_attribute_with_highest_information_gain(X)
        if gain(X, a) == 0:
            node.label = majority_label(X)
        else:
            for v in values(a):
                X_v = {x ∈ X | x[a] == v}
                node.children.append(ID3(X_v))
    return node
```

Gini Impurity

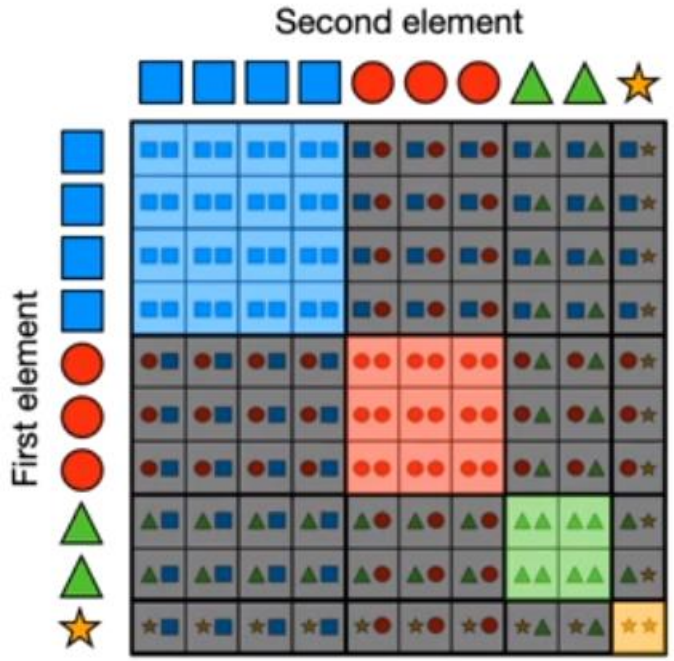


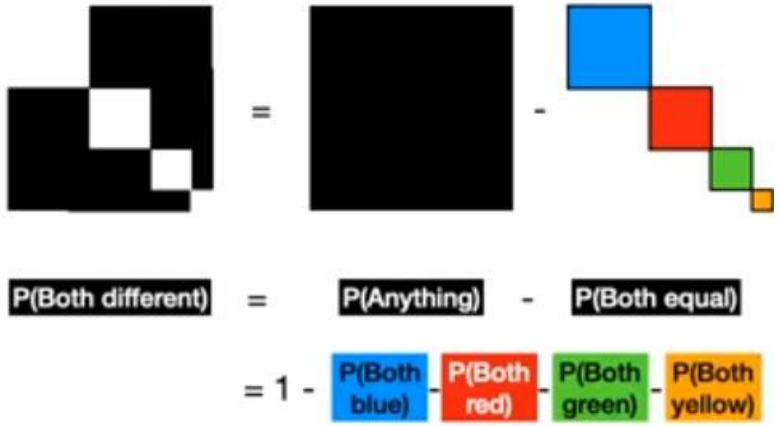
Corrado Gini

Gini impurity

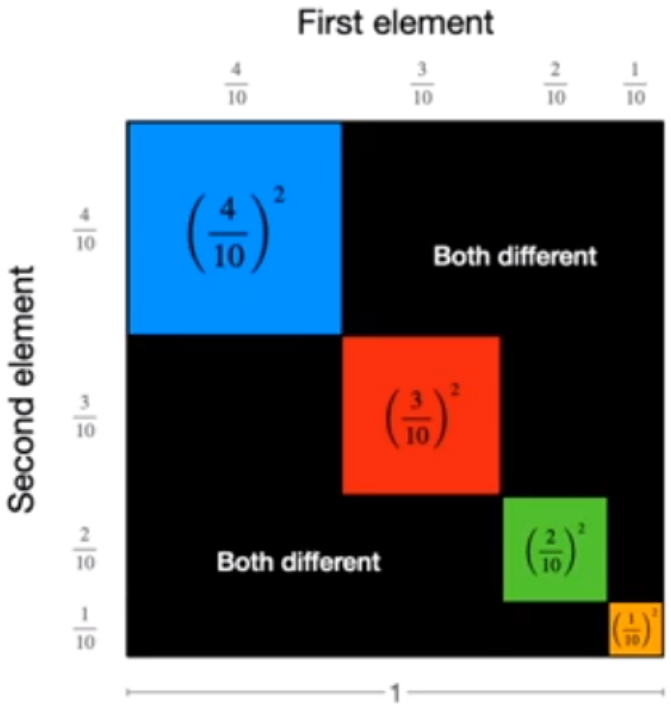


Gini Impurity





Gini Impurity



P(Both different) = P(Anything) - P(Both equal)

$= 1 -$ P(Both blue) - P(Both red) - P(Both green) - P(Both yellow)

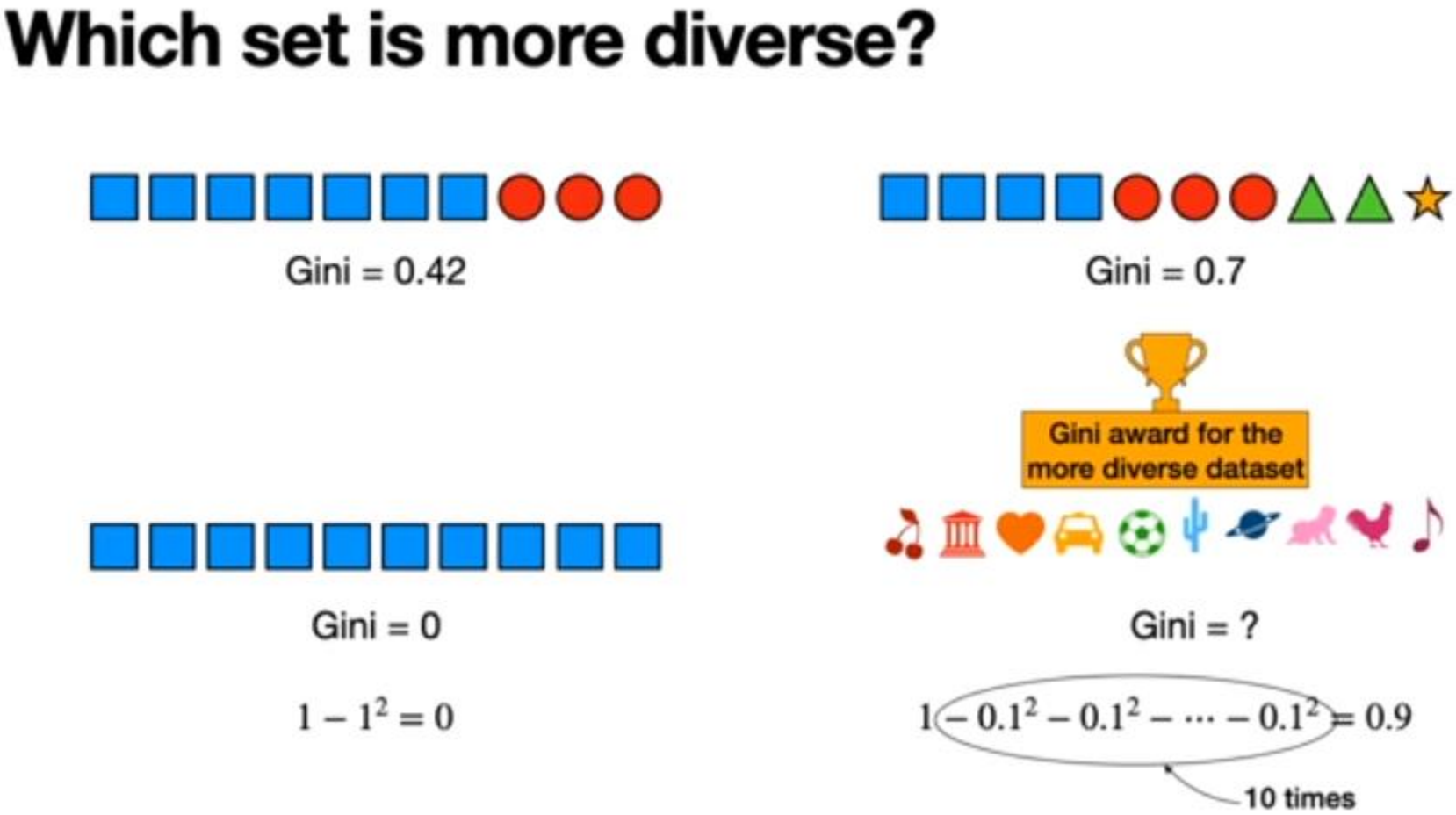
$= 1 - \left(\frac{4}{10}\right)^2 - \left(\frac{3}{10}\right)^2 - \left(\frac{2}{10}\right)^2 - \left(\frac{1}{10}\right)^2$

$= 1 - 0.4^2 - 0.3^2 - 0.2^2 - 0.1^2$

$= 1 - 0.16 - 0.09 - 0.04 - 0.01$

$= 0.70$

Gini Imp


Which set is more diverse?
Gini = 0.42
Gini = 0.7
Gini award for the more diverse dataset
Gini = 0
Gini = ?
$1 - 1^2 = 0$
$1 - 0.1^2 - 0.1^2 - \cdots - 0.1^2 = 0.9$
10 times

Gini impurity

General formula

n classes

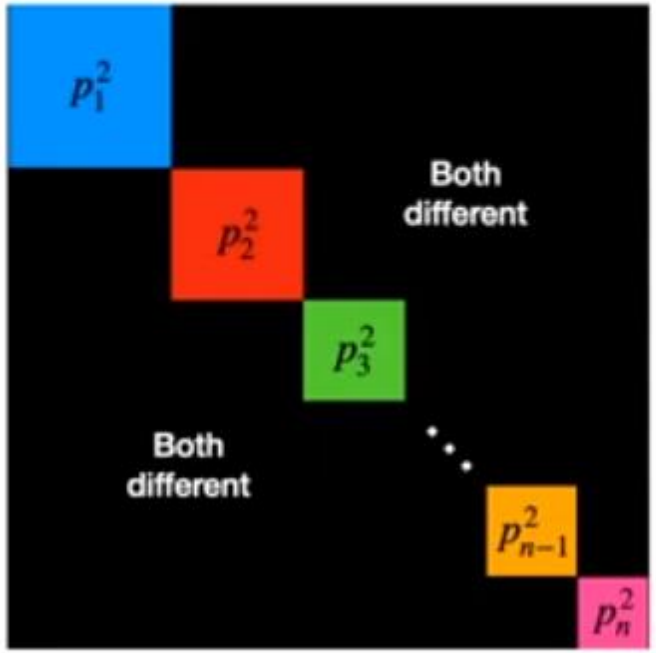
Proportions: $p_1, p_2, \ldots, p_n$

Gini impurity index: $1 - p_1^2 - p_2^2 - \cdots - p_n^2$

P(Both different)

P(Anything)

P(Both equal)



Gini Impurity

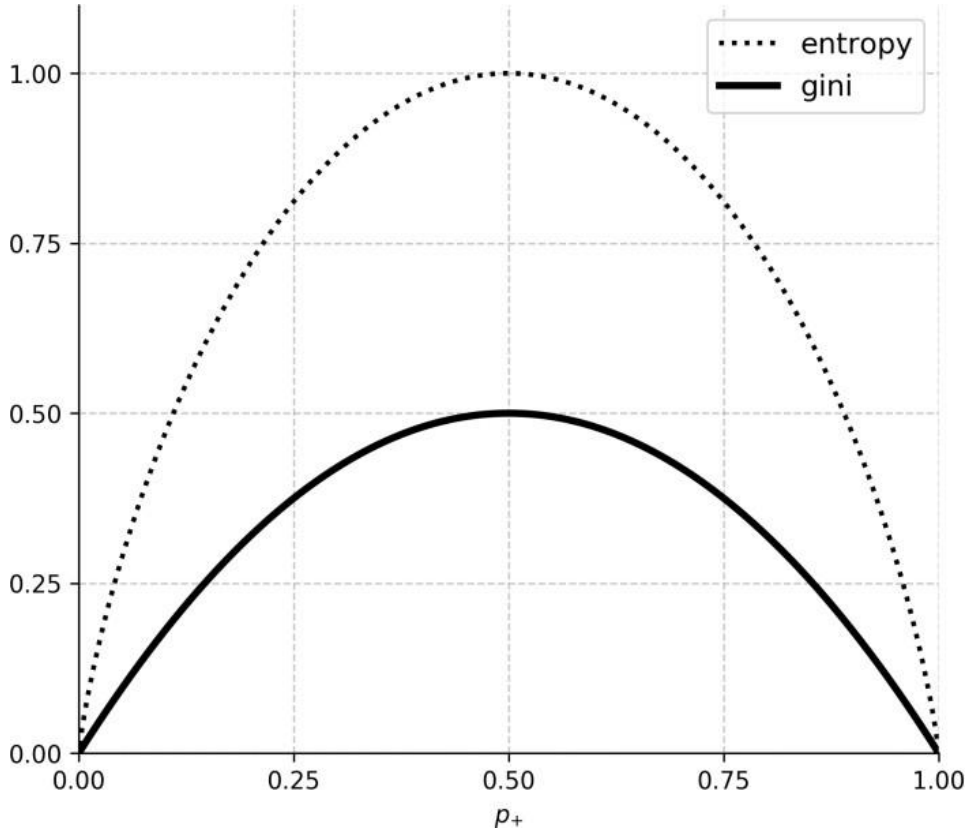
- Gini impurity measures how often a randomly chosen example would be incorrectly labeled if it was randomly labeled according to the label distribution

Error of classifying randomly picked fruit with randomly picked label

- For a set of samples X with k classes:

$$gini(X) = 1 - \sum_{i=1}^{k} p_i^2$$

where p_i is the proportion of elements of class i

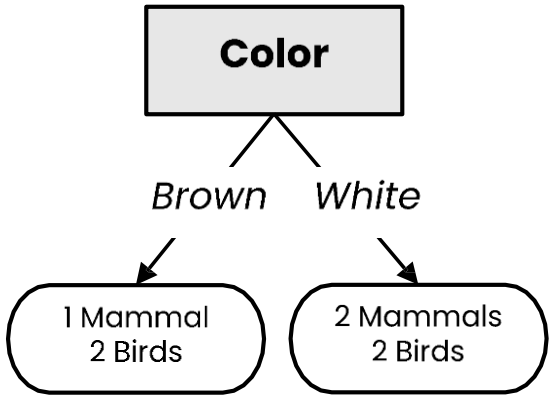


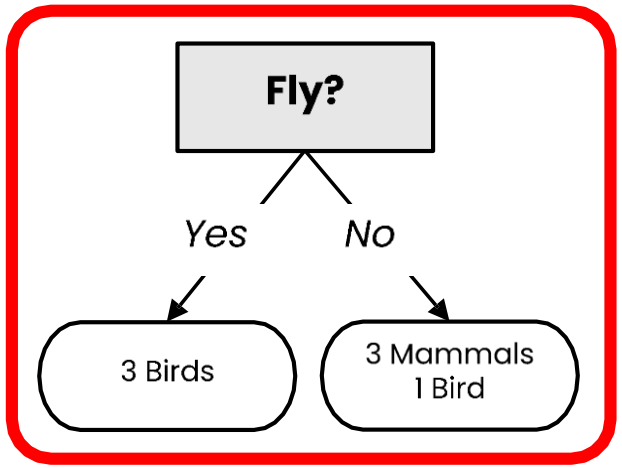
- Can be used as an alternative to entropy for selecting attributes!

Best attribute = highest impurity decrease

In practice, we compute $gini(X)$ only once!

Does it fly?	Color	Class
No	Brown	**Mammal**
No	White	**Mammal**
Yes	Brown	**Bird**
Yes	White	**Bird**
No	White	**Mammal**
No	Brown	**Bird**
Yes	White	**Bird**





$$gini\ (X) = 1 - \left(\frac{3}{7}\right)^2 - \left(\frac{4}{7}\right)^2 \approx 0.489$$

$$gini\ (X_{color=brown}) = 1 - \left(\frac{1}{3}\right)^2 - \left(\frac{2}{3}\right)^2 \approx 0.444$$

$$gini\ (X_{color=white}) = 0.5$$

$$\triangle\ \boldsymbol{gini\ (X, color) = 0.489 - \frac{3}{7} \cdot 0.444 - \frac{4}{7} \cdot 0.5 \approx 0.013}$$

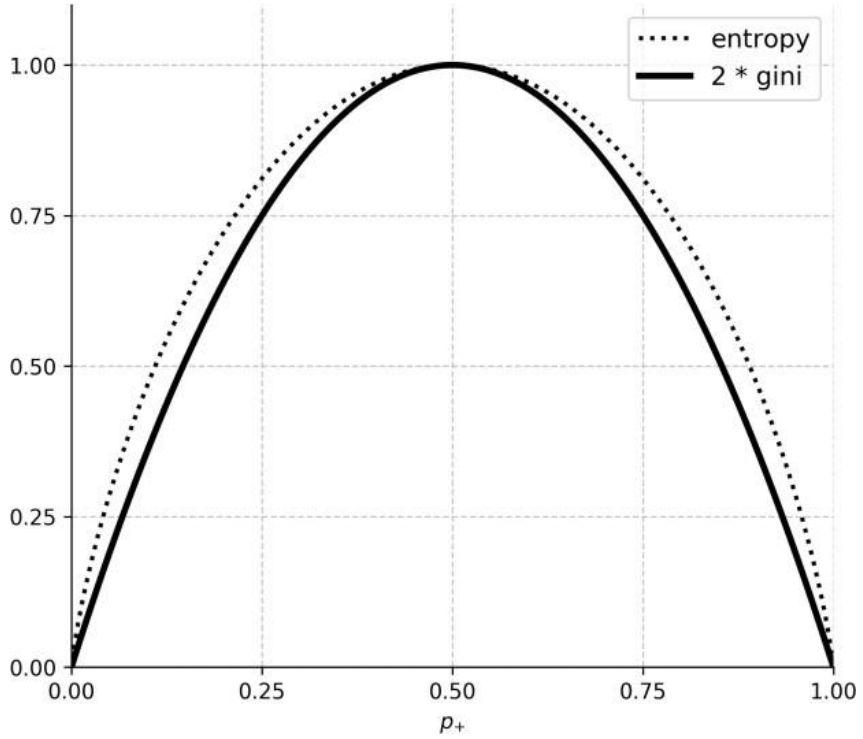
$$gini\ (X_{fly=yes}) = 0$$

$$gini\ (X_{fly=no}) = 1 - \left(\frac{3}{4}\right)^2 - \left(\frac{1}{4}\right)^2 \approx 0.375$$

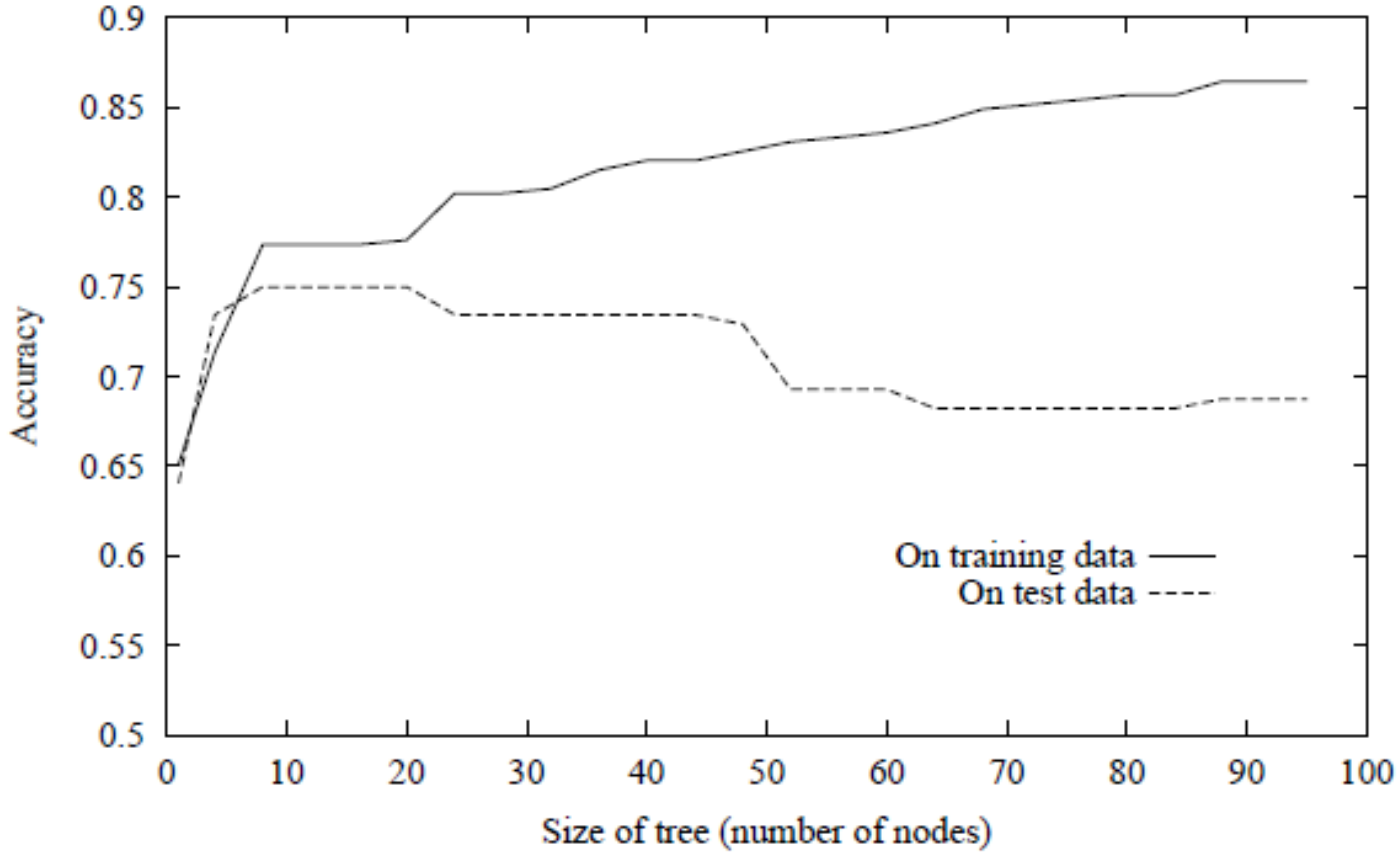
$$\triangle\ \boldsymbol{gini\ (X, fly) = 0.489 - \frac{3}{7} \cdot 0 - \frac{4}{7} \cdot 0.375 \approx 0.274}$$

Entropy versus Gini Impurity

- Entropy and Gini Impurity give similar results in practice
  - They only disagree in about 2% of cases
    "Theoretical Comparison between the Gini Index and Information Gain Criteria" [Răileanu & Stoffel, AMAI 2004]
  - Entropy might be slower to compute, because of the log



Problem of DT: Overfitting



Overfitting

Reasons for overfitting:

- noise in the data
- number of training examples is to small too produce a representative sample of the target function

How to avoid overfitting:

- **stop the tree grow earlier**, before it reaches the point where it perfectly classifies the training data
- allow overfitting and then **post-prune** the tree (more successful in practice!)

How to determine the perfect tree size:

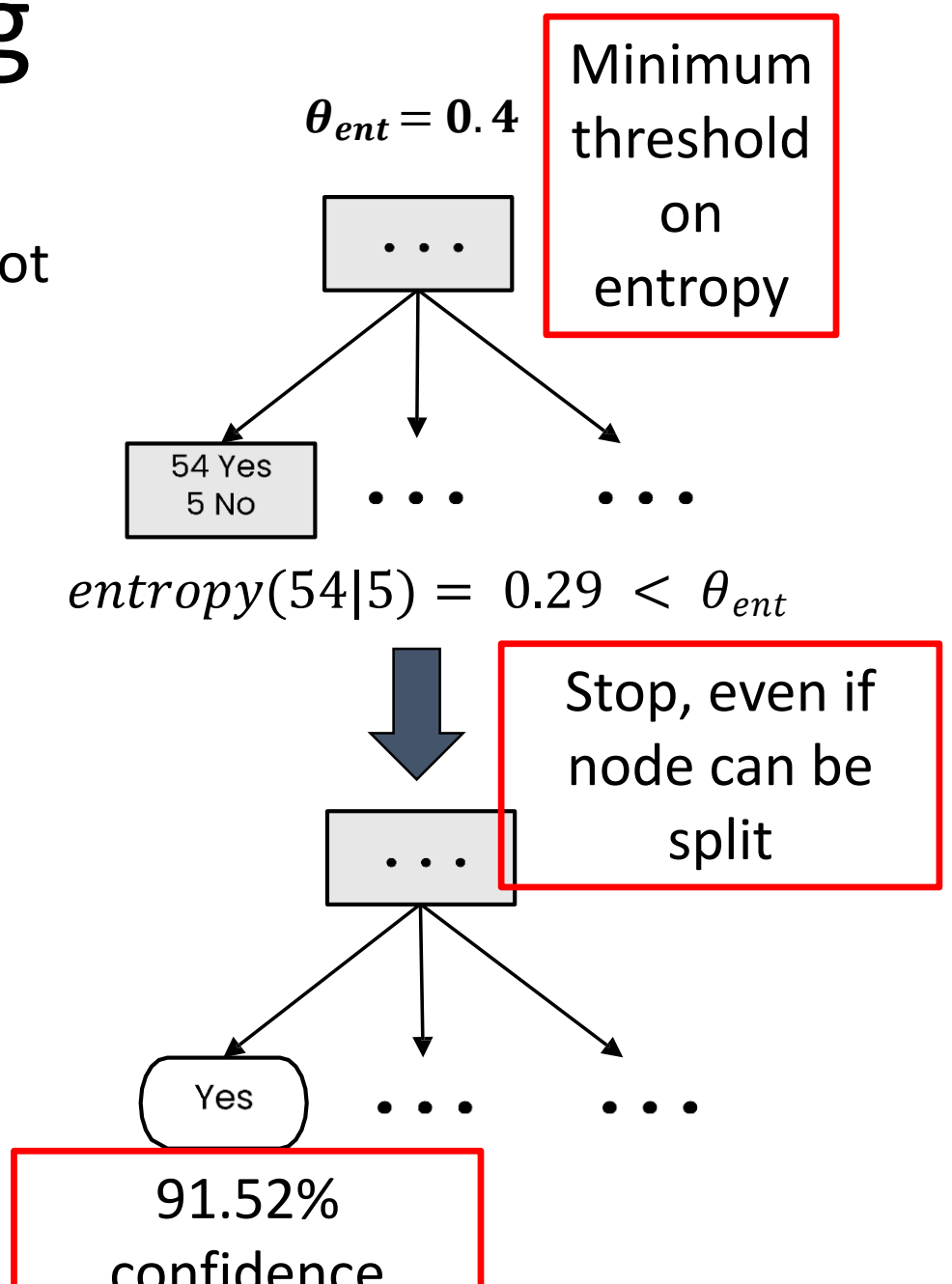
- separate validation set to evaluate utility of post-pruning
- apply statistical test to estimate whether expanding (or pruning) produces an improvement

Pre-pruning

- Pre-pruning implies early stopping:
  - If some condition is met, the current node will not be split, even if it is not 100% pure
  - It will become a leaf node with the label of the majority class in the current set

(the class distribution could be used as prediction confidence)

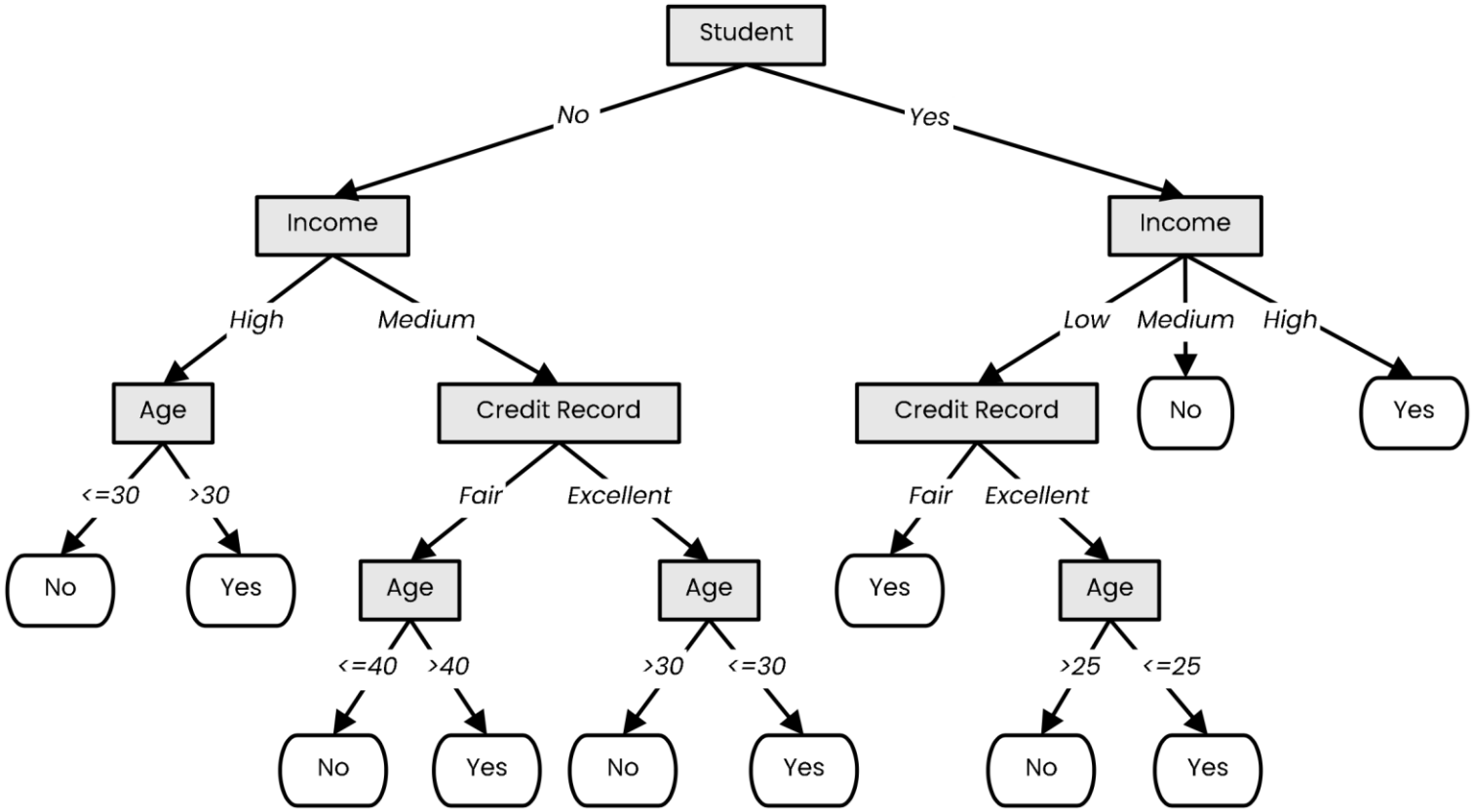
- Common stopping criteria include setting a threshold on:
  - Entropy (or Gini Impurity) of the current set
  - Number of samples in the current set
  - Gain of the best-splitting attribute
  - Depth of the tree

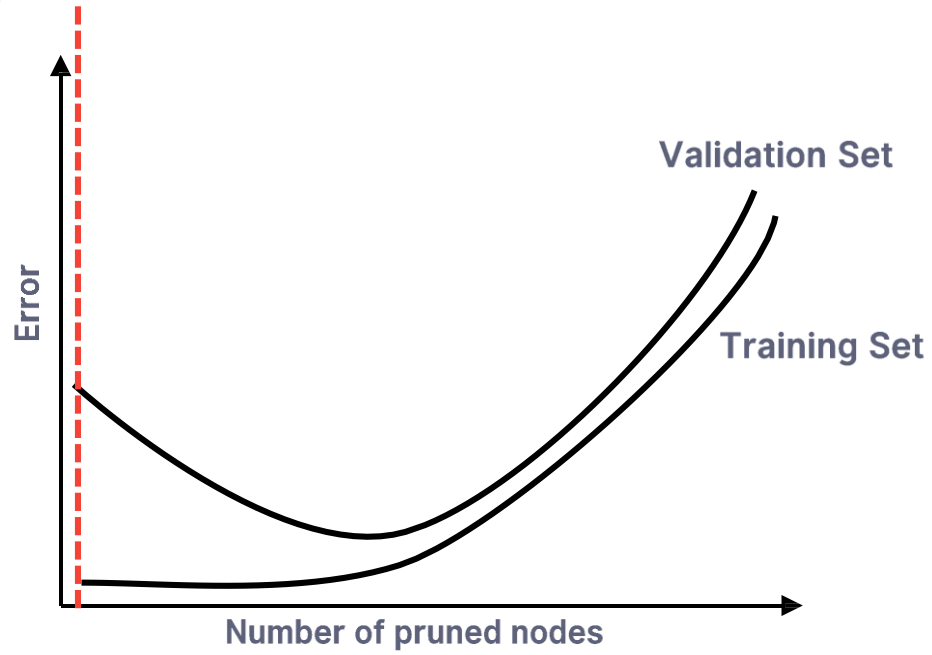


Post-pruning

- The aim of pruning is to discard parts of a classification model that describe random variation in the training sample rather than true features of the underlying domain.
- This makes the model more comprehensible to the user, and potentially more accurate on new data that has not been used for training the classifier.
- Statistical significance tests can be used to make pruning decisions in classification models. Reduced-error pruning (Quinlan, 1987), a standard algorithm for post-pruning decision trees, does not take statistical significance into account, but it is known to be one of the fastest pruning algorithms, producing trees that are both accurate and small.

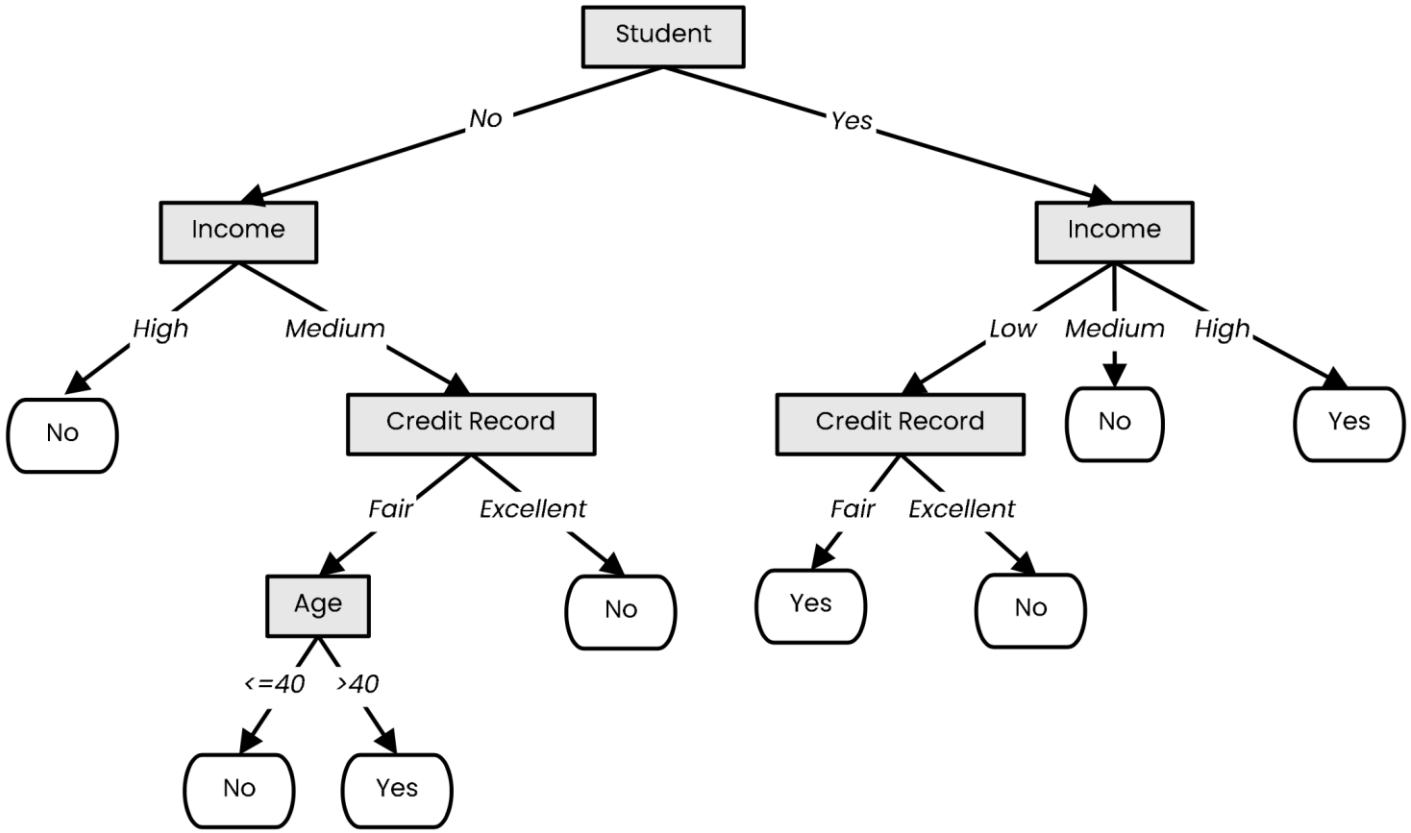
Post-pruning

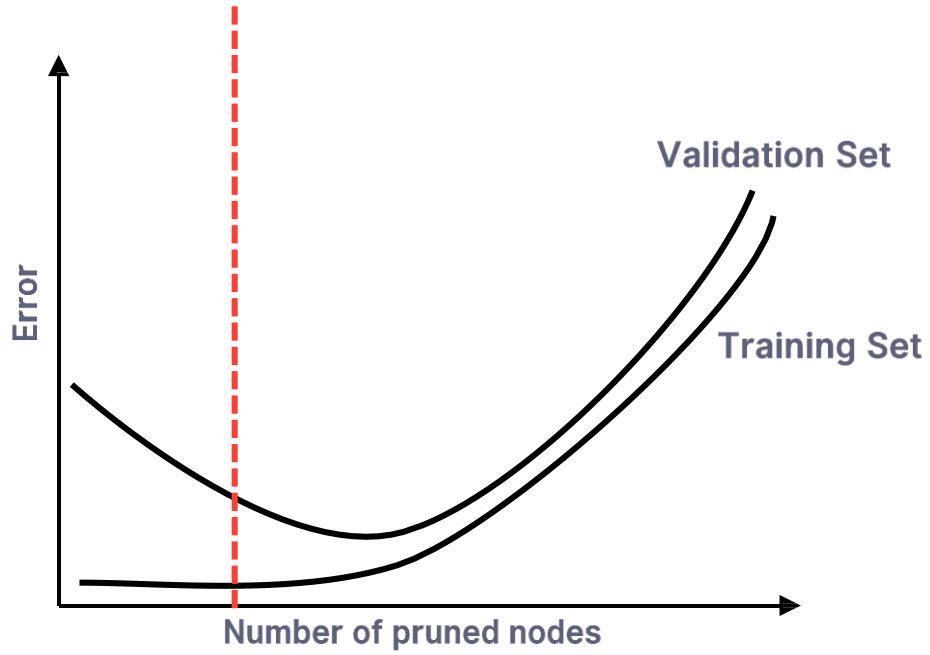




- Prune nodes in a bottom-up manner, if it decreases validation error

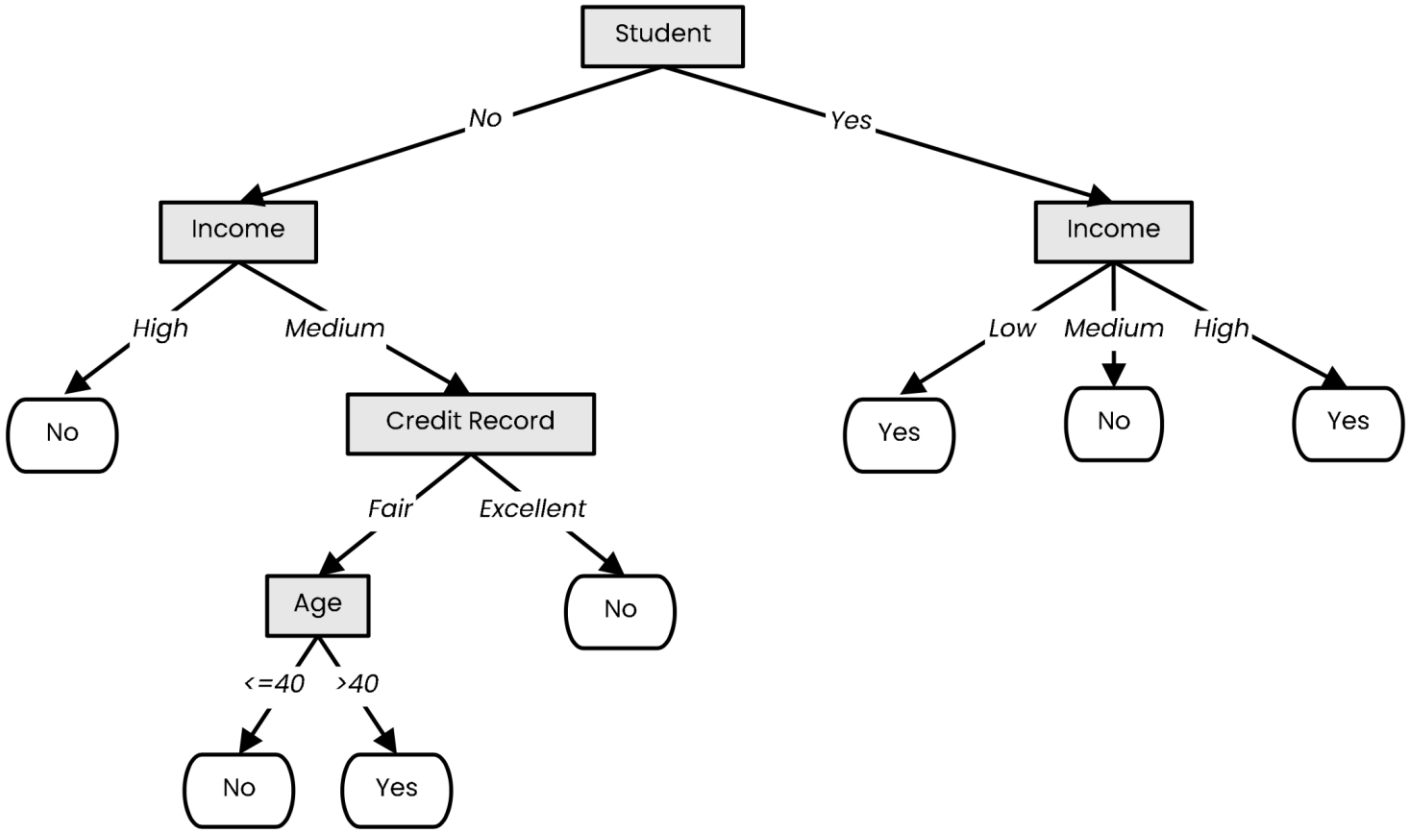
Post-pruning

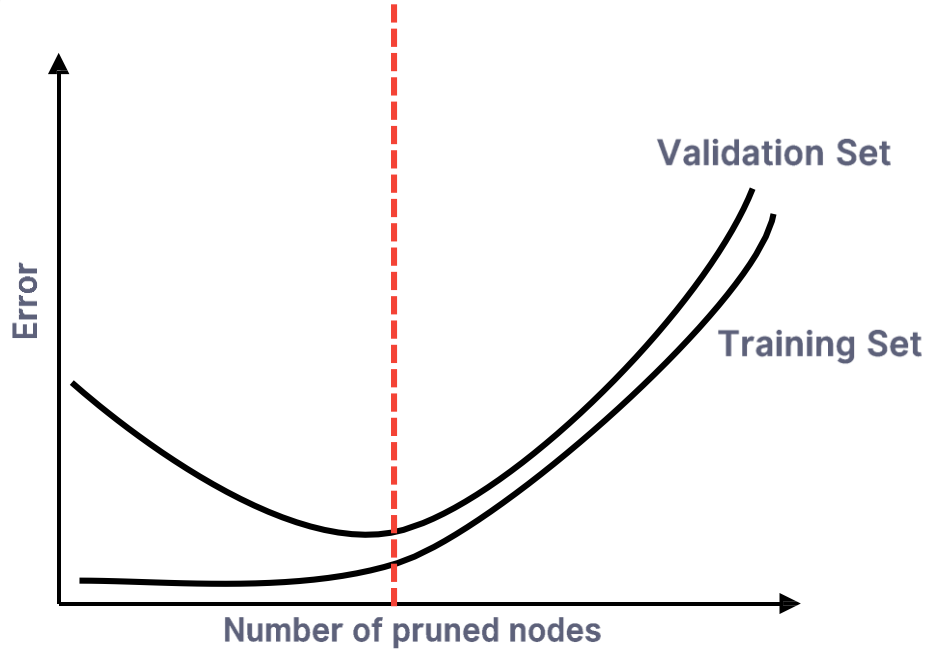




- Prune nodes in a bottom-up manner, if it decreases validation error

Post-pruning





- Prune nodes in a bottom-up manner, if it decreases validation error

Handling Numerical Attributes

Handling numerical attributes

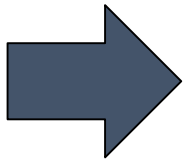
- Numerical attributes have to be treated differently
  - ➢ Find the best splitting value

Gain of numerical attribute a if we split at value t

$$gain(X, a, t) = entropy(X) - \frac{|X_{a \le t}|}{|X|} entropy(X_{a \le t}) - \frac{|X_{a > t}|}{|X|} entropy(X_{a > t})$$

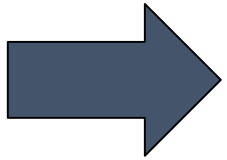
Humidity	Play Tennis?
90	No
87	No
93	Yes
89	Yes
79	Yes
59	No
77	Yes
91	No
68	Yes
80	Yes
72	Yes
96	Yes
74	Yes
97	No

Sort



Humidity	Play Tennis?
59	No
68	Yes
72	Yes
74	Yes
77	Yes
79	Yes
80	Yes
87	No
89	Yes
90	No
91	No
93	Yes
96	Yes
97	No

Mean of each consecutive pair



Candidate split values
63
70
73
75.5
78
79.5
83.5
88
89.5
90.5
92
94.5
96.5

Handling numerical attributes

- Numerical attributes have to be treated differently
  - ➢ Find the best splitting value

$$gain(X, a, t) = entropy(X) - \frac{|X_{a \le t}|}{|X|} entropy(X_{a \le t}) - \frac{|X_{a > t}|}{|X|} entropy(X_{a > t})$$

Humidity	Play Tennis?
90	No
87	No
93	Yes
89	Yes
79	Yes
59	No
77	Yes
91	No
68	Yes
80	Yes
72	Yes
96	Yes
74	Yes
97	No

Sort

Humidity	Play Tennis?
59	No
68	Yes
72	Yes
74	Yes
77	Yes
79	Yes
80	Yes
87	No
89	Yes
90	No
91	No
93	Yes
96	Yes
97	No

Mean of each consecutive pair

Candidate split values
63
70
73
75.5
78
79.5
83.5
88
89.5
90.5
92
94.5
96.5

$$gain\ (X, humidity, 83.5) =$$

Handling numerical attributes

- Numerical attributes have to be treated differently
  - ➢ Find the best splitting value

$$gain(X, a, t) = entropy(X) - \frac{|X_{a \le t}|}{|X|} entropy(X_{a \le t}) - \frac{|X_{a > t}|}{|X|} entropy(X_{a > t})$$

Humidity	Play Tennis?
90	No
87	No
93	Yes
89	Yes
79	Yes
59	No
77	Yes
91	No
68	Yes
80	Yes
72	Yes
96	Yes
74	Yes
97	No

Sort

Humidity	Play Tennis?
59	No
68	Yes
72	Yes
74	Yes
77	Yes
79	Yes
80	Yes
87	No
89	Yes
90	No
91	No
93	Yes
96	Yes
97	No

Mean of each consecutive pair

Candidate split values
63
70
73
75.5
78
79.5
83.5
88
89.5
90.5
92
94.5
96.5

$$gain\ (X, humidity, 83.5) = 0.94$$

Handling numerical attributes

- Numerical attributes have to be treated differently
  - ➢ Find the best splitting value

$$gain(X, a, t) = entropy(X) - \frac{|X_{a \le t}|}{|X|} entropy(X_{a \le t}) - \frac{|X_{a > t}|}{|X|} entropy(X_{a > t})$$

Humidity	Play Tennis?
90	No
87	No
93	Yes
89	Yes
79	Yes
59	No
77	Yes
91	No
68	Yes
80	Yes
72	Yes
96	Yes
74	Yes
97	No

Sort

Humidity	Play Tennis?
59	No
68	Yes
72	Yes
74	Yes
77	Yes
79	Yes
80	Yes
87	No
89	Yes
90	No
91	No
93	Yes
96	Yes
97	No

Mean of each consecutive pair

Candidate split values
63
70
73
75.5
78
79.5
83.5
88
89.5
90.5
92
94.5
96.5

$$gain\ (X, humidity, 83.5) = 0.94 - \frac{7}{14} \cdot 0.59$$

Handling numerical attributes

- Numerical attributes have to be treated differently
  - ➢ Find the best splitting value

$$gain(X, a, t) = entropy(X) - \frac{|X_{a \le t}|}{|X|} entropy(X_{a \le t}) - \frac{|X_{a > t}|}{|X|} entropy(X_{a > t})$$

Humidity	Play Tennis?
90	No
87	No
93	Yes
89	Yes
79	Yes
59	No
77	Yes
91	No
68	Yes
80	Yes
72	Yes
96	Yes
74	Yes
97	No

Sort

Humidity	Play Tennis?
59	No
68	Yes
72	Yes
74	Yes
77	Yes
79	Yes
80	Yes
87	No
89	Yes
90	No
91	No
93	Yes
96	Yes
97	No

Mean of each consecutive pair

Candidate split values
63
70
73
75.5
78
79.5
83.5
88
89.5
90.5
92
94.5
96.5

$$gain\ (X, humidity, 83.5) = 0.94 - \frac{7}{14} \cdot 0.59 - \frac{7}{14} \cdot 0.98$$

Handling numerical attributes

- Numerical attributes have to be treated differently
  - ➢ Find the best splitting value

$$gain(X, a, t) = entropy(X) - \frac{|X_{a \le t}|}{|X|} entropy(X_{a \le t}) - \frac{|X_{a > t}|}{|X|} entropy(X_{a > t})$$

Humidity	Play Tennis?
90	No
87	No
93	Yes
89	Yes
79	Yes
59	No
77	Yes
91	No
68	Yes
80	Yes
72	Yes
96	Yes
74	Yes
97	No

Sort

Humidity	Play Tennis?
59	No
68	Yes
72	Yes
74	Yes
77	Yes
79	Yes
80	Yes
87	No
89	Yes
90	No
91	No
93	Yes
96	Yes
97	No

Mean of each consecutive pair

Candidate split values
63
70
73
75.5
78
79.5
83.5
88
89.5
90.5
92
94.5
96.5

$$gain\ (X, humidity, 83.5) = 0.94 - \frac{7}{14} \cdot 0.59 - \frac{7}{14} \cdot 0.98 \approx \mathbf{0.152}$$

Handling numerical attributes

- Numerical attributes have to be treated differently
  - ➢ Find the best splitting value

Humidity	Play Tennis?
90	No
87	No
93	Yes
89	Yes
79	Yes
59	No
77	Yes
91	No
68	Yes
80	Yes
72	Yes
96	Yes
74	Yes
97	No

Sort

Humidity	Play Tennis?
59	No
68	Yes
72	Yes
74	Yes
77	Yes
79	Yes
80	Yes
87	No
89	Yes
90	No
91	No
93	Yes
96	Yes
97	No

Mean of each consecutive pair

Candidate split values
63
70
73
75.5
78
79.5
83.5
88
89.5
90.5
92
94.5
96.5

Gain for every candidate

Information gain
0.113
0.01
0.0004
0.015
0.045
0.09
0.152
0.048
0.102
0.025
0.0004
0.01
0.113

83.5 is the best splitting value with an information gain of 0.152

Handling numerical attributes

- Numerical attributes have to be treated differently
  - ➢ Find the best splitting value

Outlook	Temperature	Humidity	Wind	Play Tennis?
Sunny	Hot	> 83.5	Weak	**No**
Sunny	Hot	> 83.5	Strong	**No**
Overcast	Hot	> 83.5	Weak	**Yes**
Rainy	Mild	> 83.5	Weak	**Yes**
Rainy	Cool	≤ 83.5	Weak	**Yes**
Rainy	Cool	≤ 83.5	Strong	**No**
Overcast	Cool	≤ 83.5	Strong	**Yes**
Sunny	Mild	> 83.5	Weak	**No**
Sunny	Cool	≤ 83.5	Weak	**Yes**
Rainy	Mild	≤ 83.5	Weak	**Yes**
Sunny	Mild	≤ 83.5	Strong	**Yes**
Overcast	Mild	> 83.5	Strong	**Yes**
Overcast	Hot	≤ 83.5	Weak	**Yes**
Rainy	Mild	> 83.5	Strong	**No**

- 83.5 is the best splitting value for **Humidity** with an information gain of 0.152
- **Humidity** is now treated as a categorical attribute with two possible values
- A new optimal split is computed at every level of the tree
- A numerical attribute can be used several times in the tree, with different split values

Example: Riding Mowers

- Goal: Classify 24 households as owning or not owning riding mowers

- Predictors = Income, Lot Size

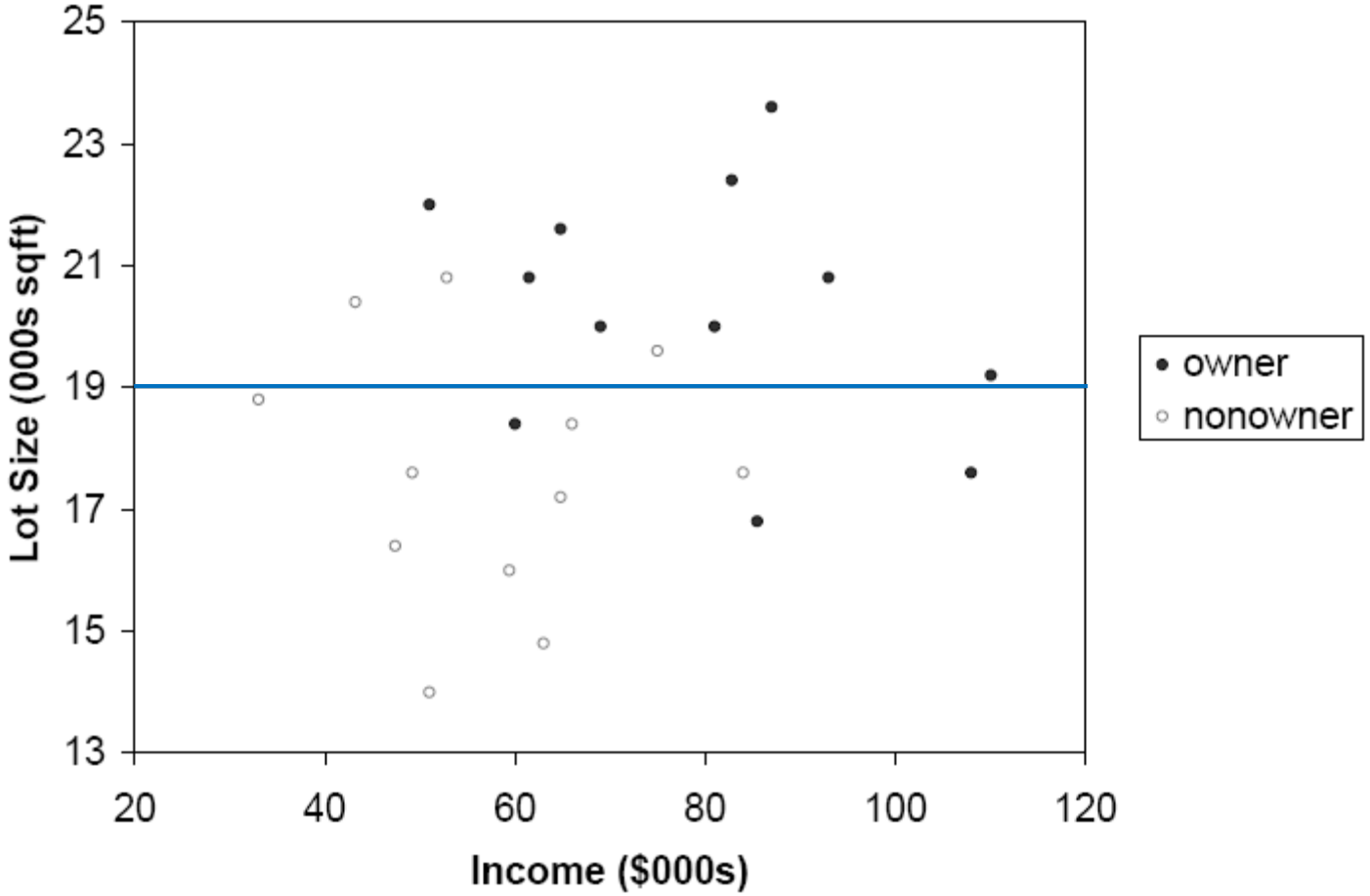
Income	Lot_Size	Ownership
60.0	18.4	owner
85.5	16.8	owner
64.8	21.6	owner
61.5	20.8	owner
87.0	23.6	owner
110.1	19.2	owner
108.0	17.6	owner
82.8	22.4	owner
69.0	20.0	owner
93.0	20.8	owner
51.0	22.0	owner
81.0	20.0	owner
75.0	19.6	non-owner
52.8	20.8	non-owner
64.8	17.2	non-owner
43.2	20.4	non-owner
84.0	17.6	non-owner
49.2	17.6	non-owner
59.4	16.0	non-owner
66.0	18.4	non-owner
47.4	16.4	non-owner
33.0	18.8	non-owner
51.0	14.0	non-owner
63.0	14.8	non-owner

How to split

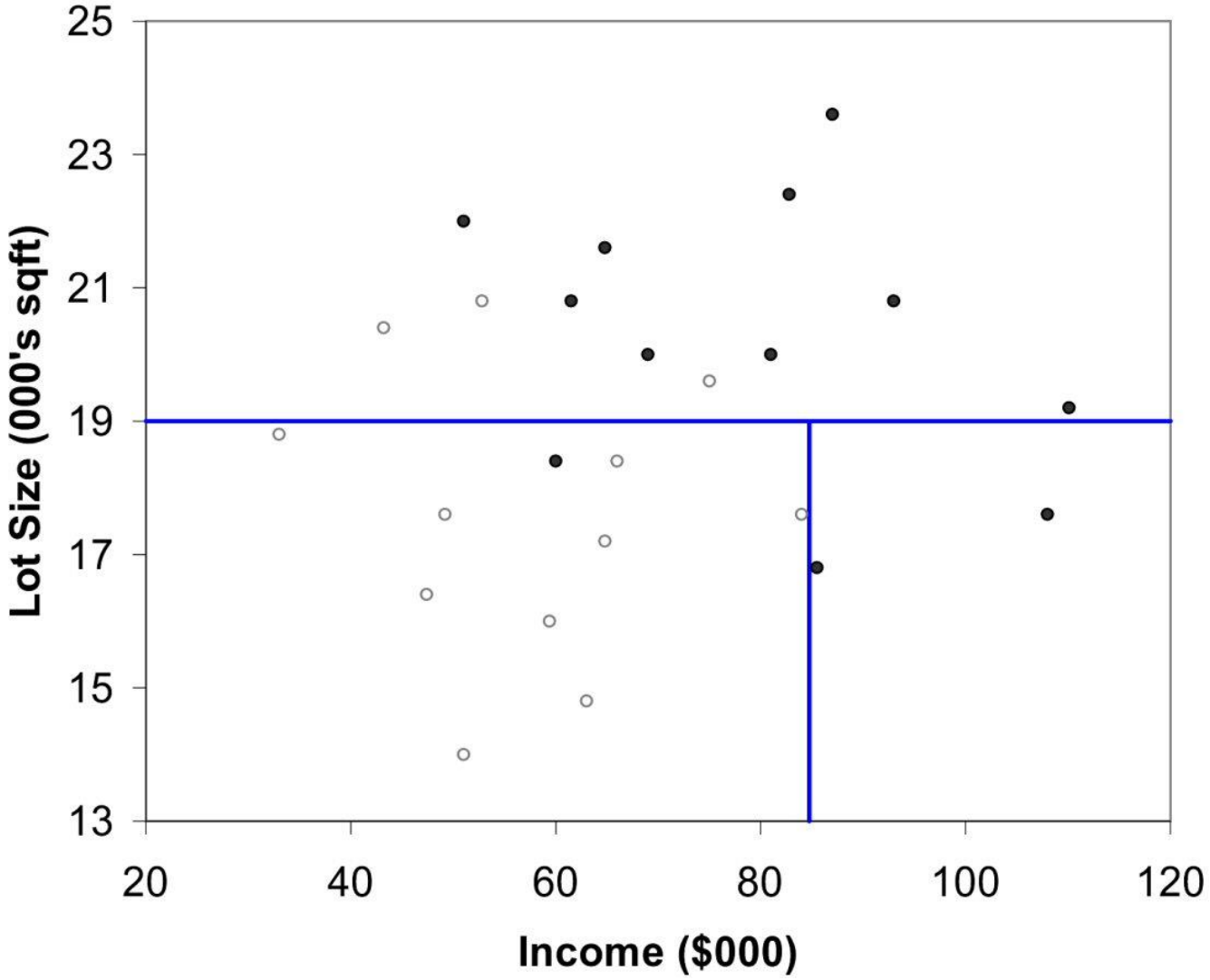
- Order records according to one variable, say lot size
- Find midpoints between successive values

  E.g. first midpoint is 14.4 (halfway between 14.0 and 14.8)
- Divide records into those with lotsize > 14.4 and those < 14.4
- After evaluating that split, try the next one, which is 15.4 (halfway between 14.8 and 16.0)

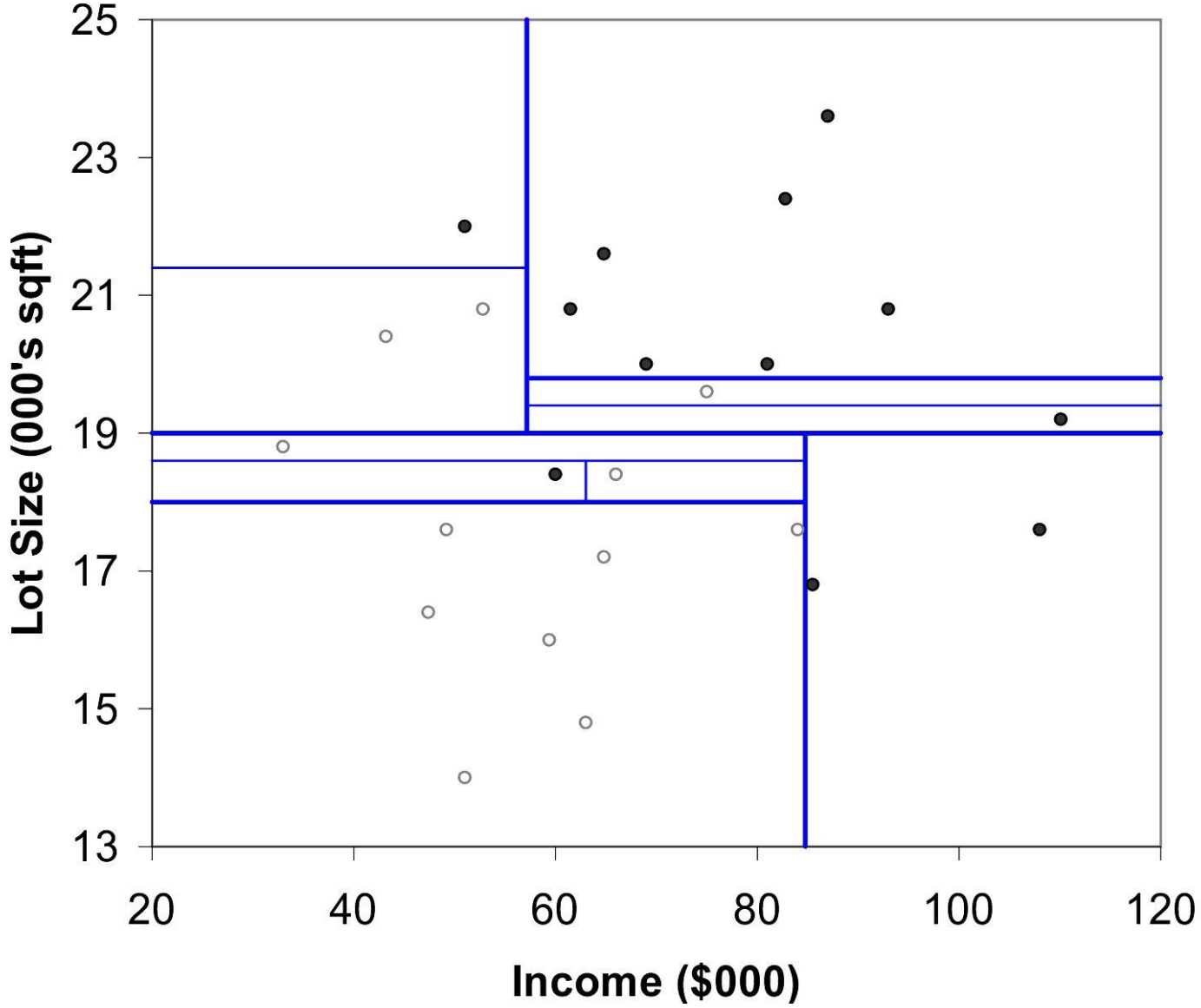
The first split: Lot Size = 19,000



Second Split: Income = $84,000



After All Splits



C4.5 Algorithm

- C4.5 algorithm is an extension of ID3 algorithm that brings several improvements:
  - Ability to handle both categorical (discrete) and numerical (continuous) attributes
    (continuous attributes are split by finding a best-splitting threshold)
  - Ability to handle missing values both at training and inference time
    (missing values at training are not used when information gain is computed; missing values at inference time are handled by exploring all corresponding branches)
  - Ability to handle attributes with different costs
  - Post-pruning in a bottom-up manner for removing branches that decrease validation error (i.e., that increase generalization capacity)

Disadvantages of DT

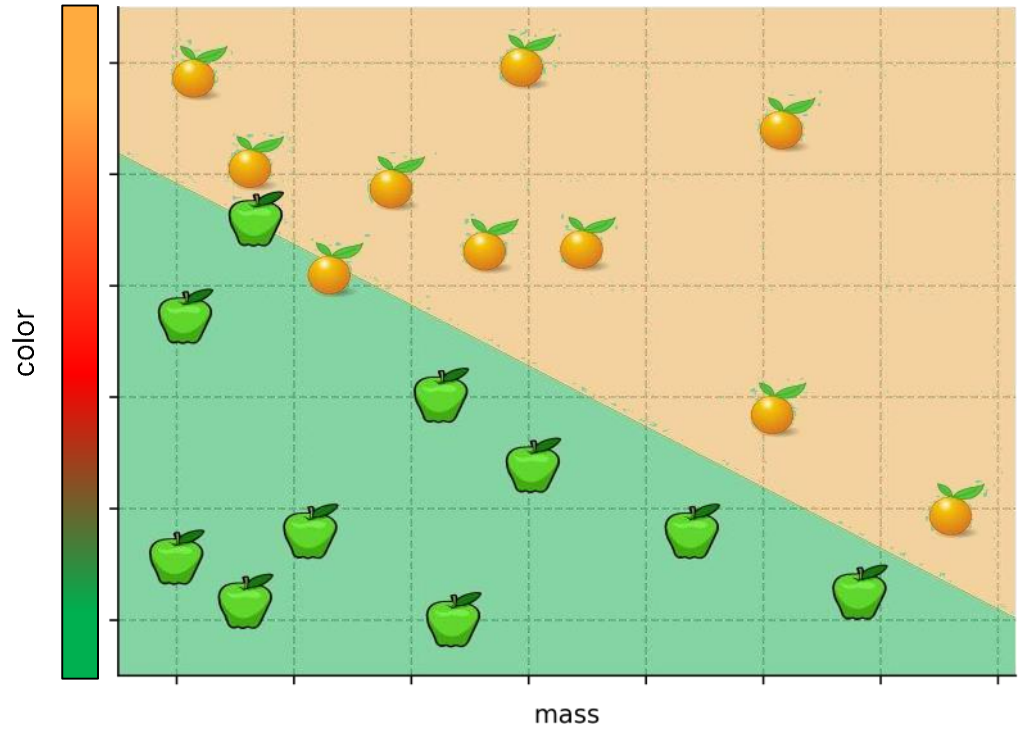
- May not perform well where there is structure in the data that is not well captured by horizontal or vertical splits

- Since the process deals with one variable at a time, no way to capture interactions between variables

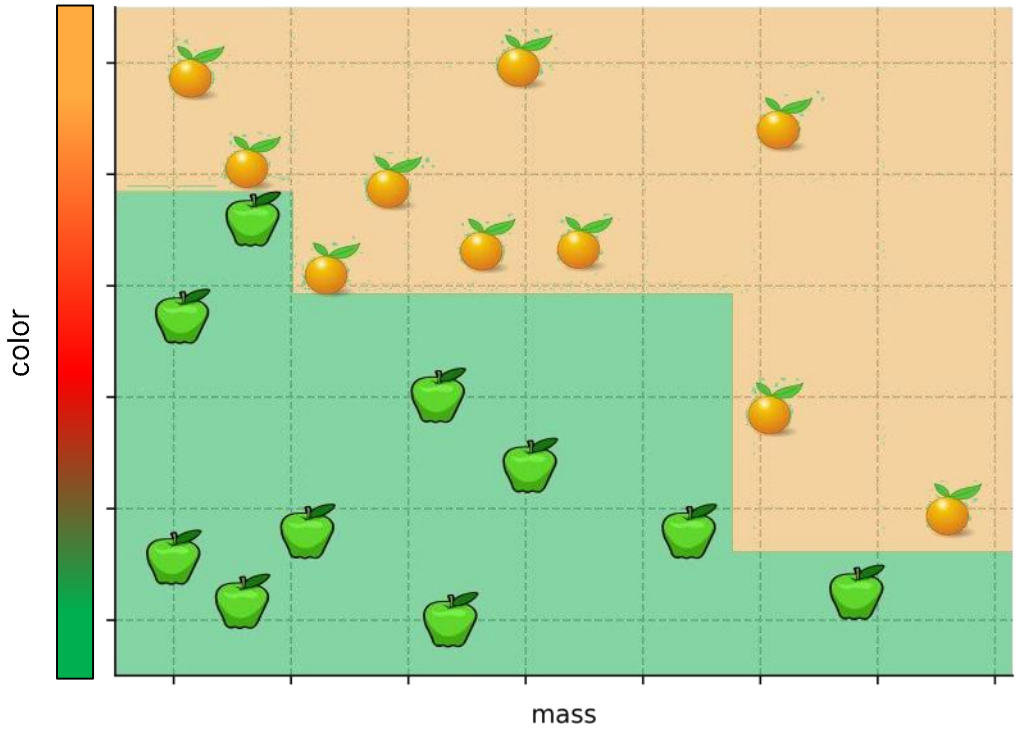
Decision Boundaries

- Decision trees produce non-linear decision boundaries

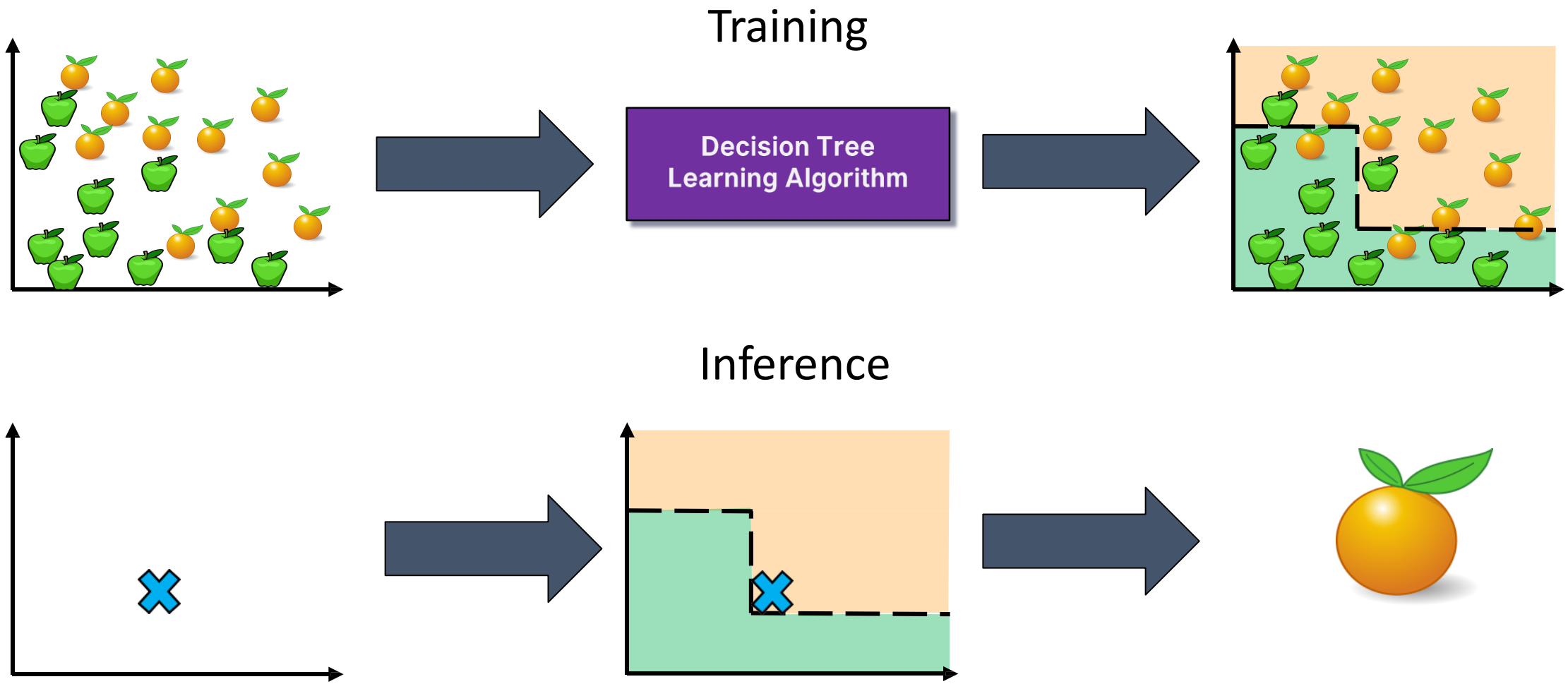
Support Vector Machines



Decision Tree



Decision Trees: Training and Inference



Random Forests

(Ensemble learning with decision trees)

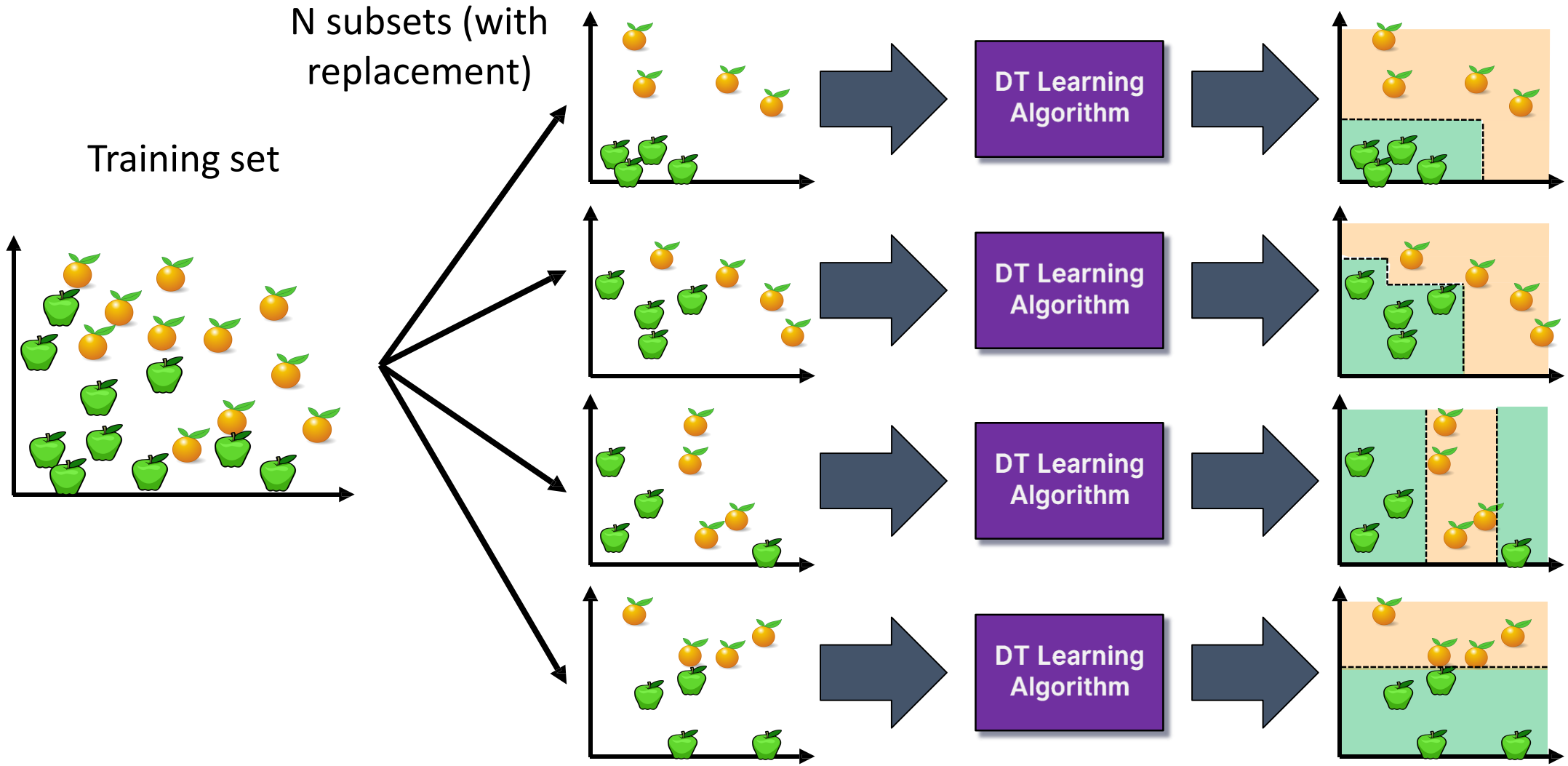
Random Forests

- Random Forests:
  - Instead of building a single decision tree and use it to make predictions, build many slightly different trees and combine their predictions
- We have a single data set, so how do we obtain slightly different trees?

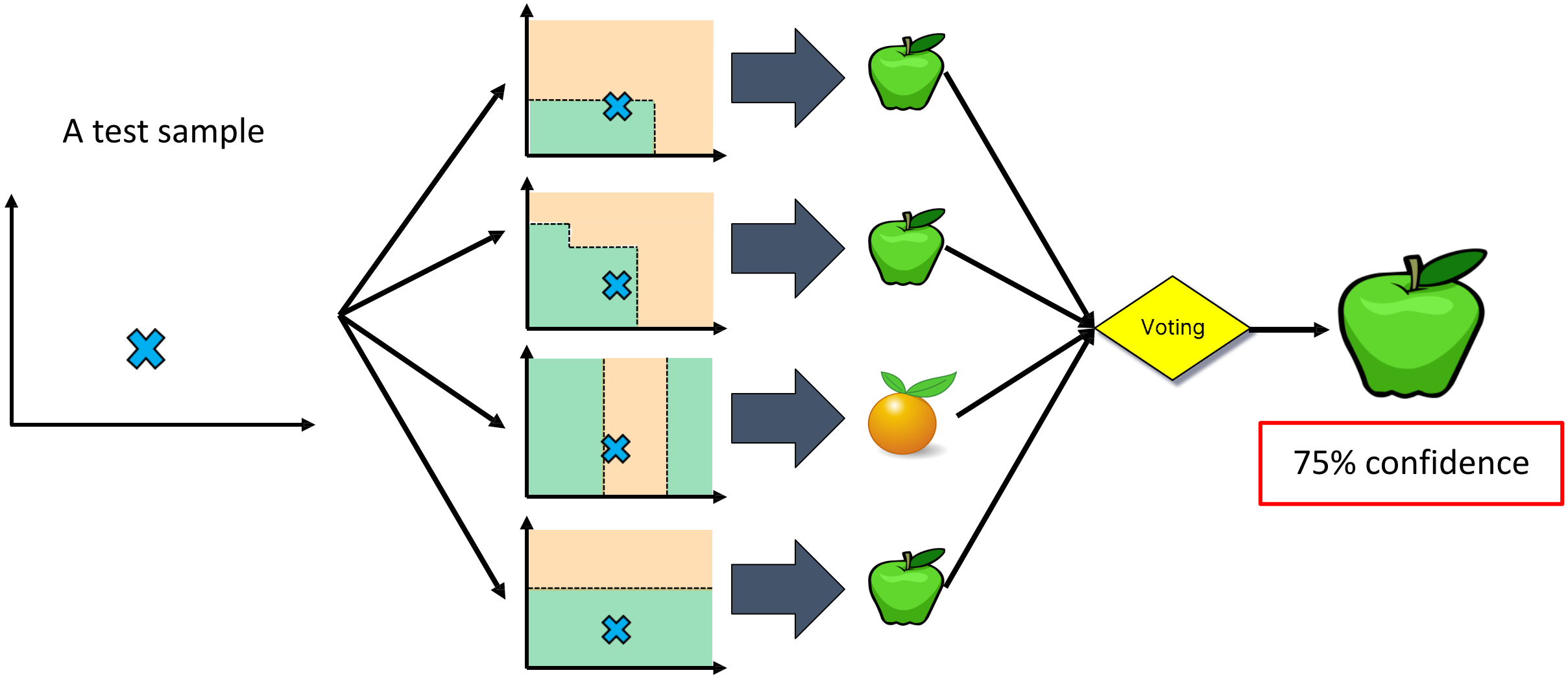
1. Bagging (**B**ootstrap **Agg**regat**ing**):
   - Take random subsets of data points from the training set to create N smaller data sets
   - Fit a decision tree on each subset

2. Random Subspace Method (also known as Feature Bagging):
   - Fit N different decision trees by constraining each one to operate on a random subset of features

Bagging at training time



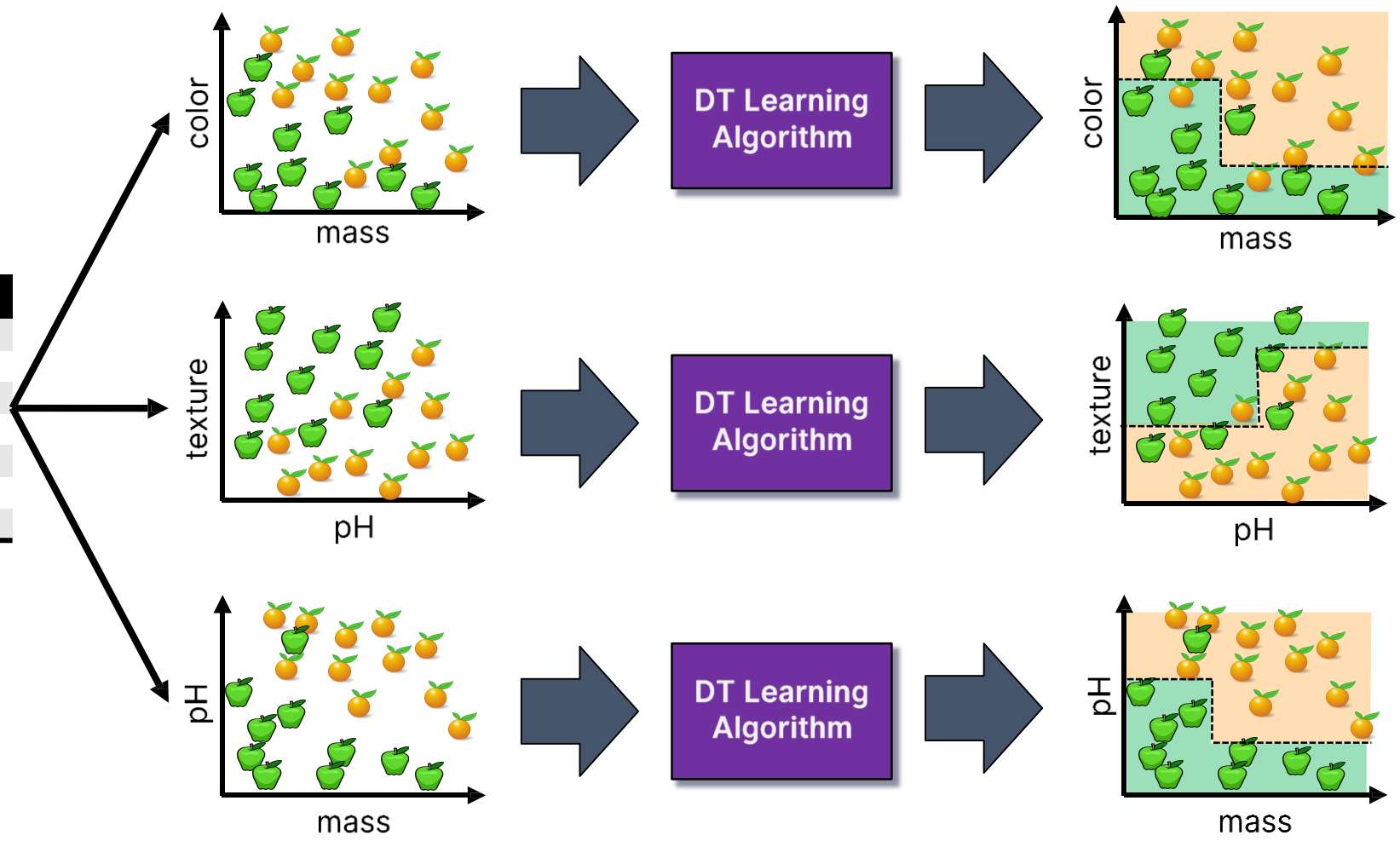
Bagging at inference time



Random Subspace Method at training time

Training data

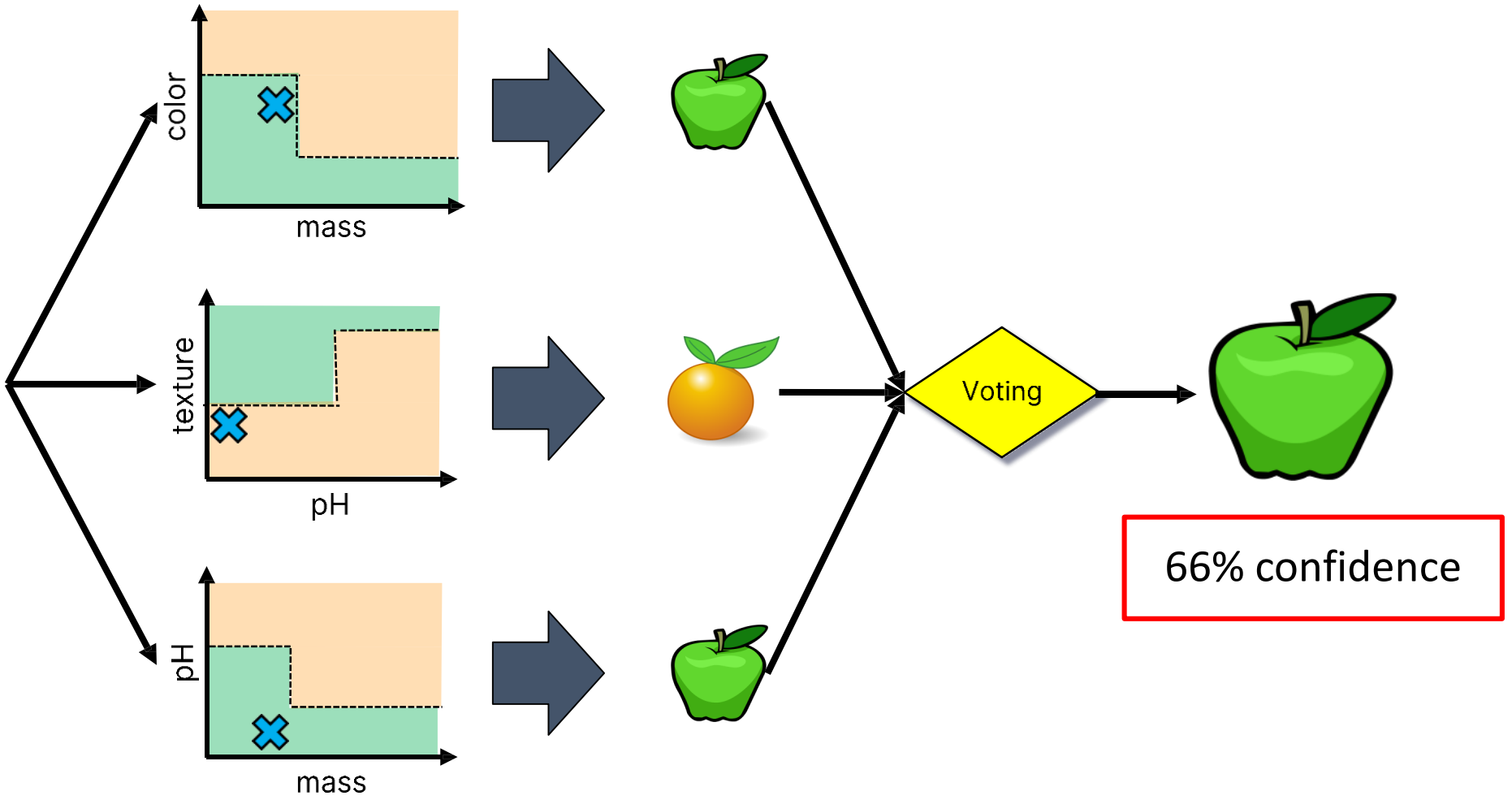
Mass (g)	Color	Texture	pH	Label
84	Green	Smooth	3.5	**Apple**
121	Orange	Rough	3.9	**Orange**
85	Red	Smooth	3.3	**Apple**
101	Orange	Smooth	3.7	**Orange**
111	Green	Rough	3.5	**Apple**
...				
117	Red	Rough	3.4	**Orange**



Random Subspace Method at inference time

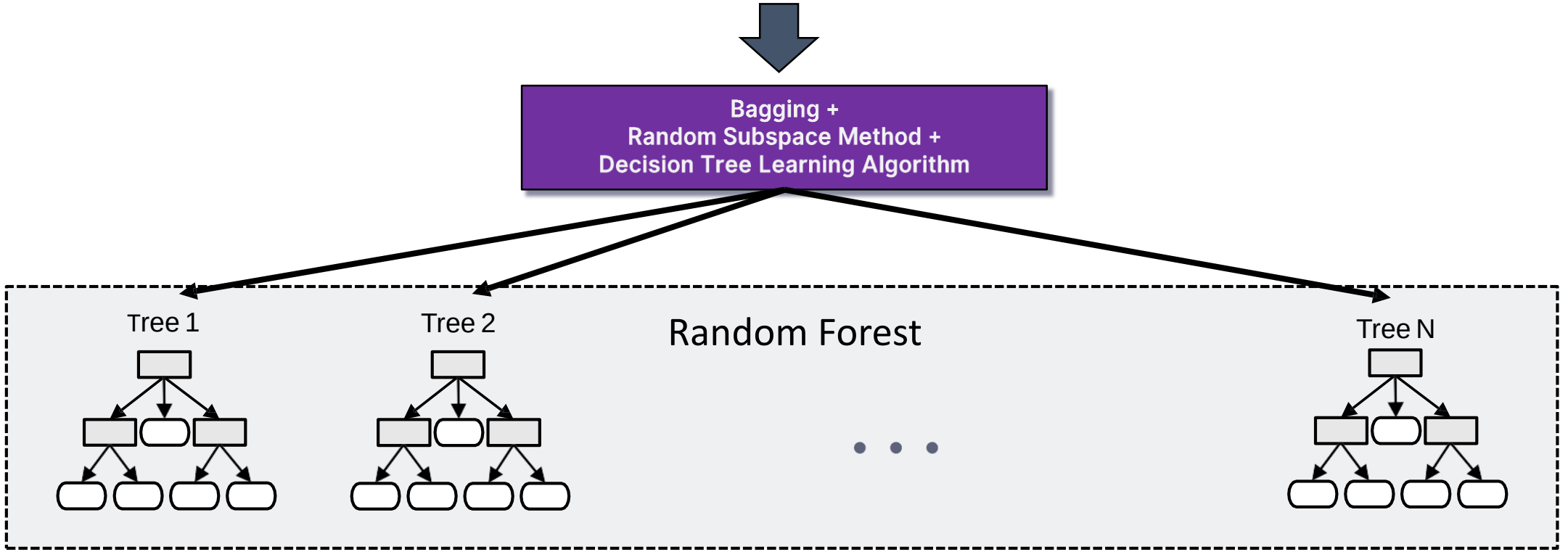
A test sample

87	Red	Smooth	3.1



Random Forests

Mass (g)	Color	Texture	pH	Label
84	Green	Smooth	3.5	**Apple**
121	Orange	Rough	3.9	**Orange**
85	Red	Smooth	3.3	**Apple**
101	Orange	Smooth	3.7	**Orange**
111	Green	Rough	3.5	**Apple**
...				
117	Red	Rough	3.4	**Orange**



History of Random Forests

- Introduction of the Random Subspace Method
  - "Random Decision Forests" [Ho, 1995] and "The Random Subspace Method for Constructing Decision Forests" [Ho, 1998]

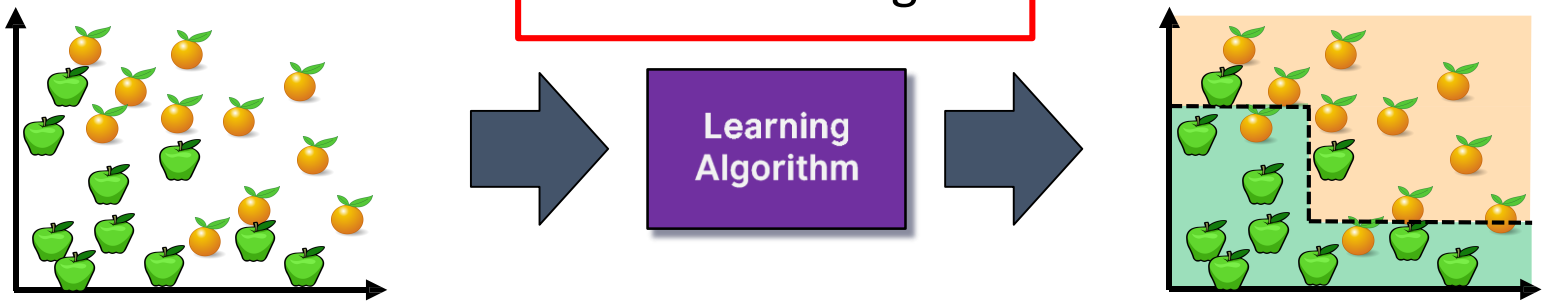
- Combined the Random Subspace Method with Bagging. Introduce the term Random Forest (a trademark of Leo Breiman and Adele Cutler)
  - "Random Forests" [Breiman, 2001]

Ensemble Learning

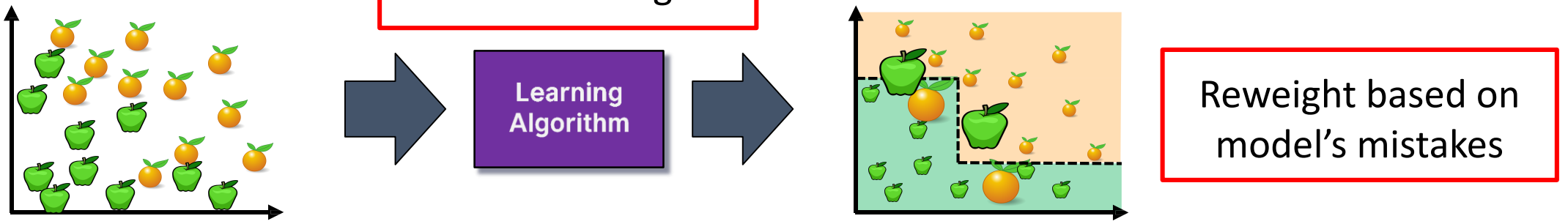
- Ensemble Learning:
  - Method that combines multiple learning algorithms to obtain performance improvements over its components
- **Random Forests** are one of the most common examples of ensemble learning
- Other commonly-used ensemble methods:
  - Bagging: multiple models on random subsets of data samples
  - Random Subspace Method: multiple models on random subsets of features
  - Boosting: train models iteratively, while making the current model focus on the mistakes of the previous ones by increasing the weight of misclassified samples

Boosting (XGBoost)

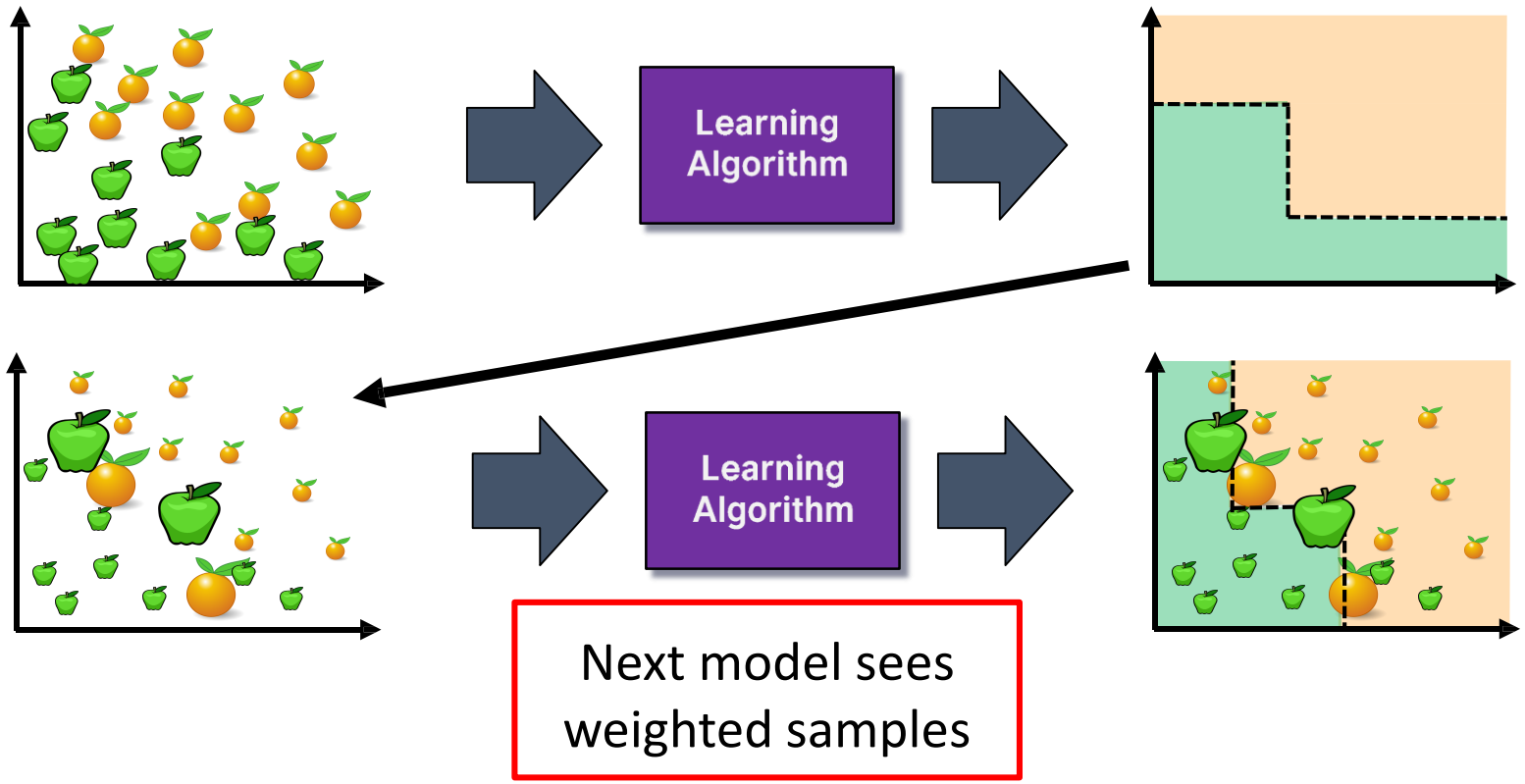
All samples have the same weight



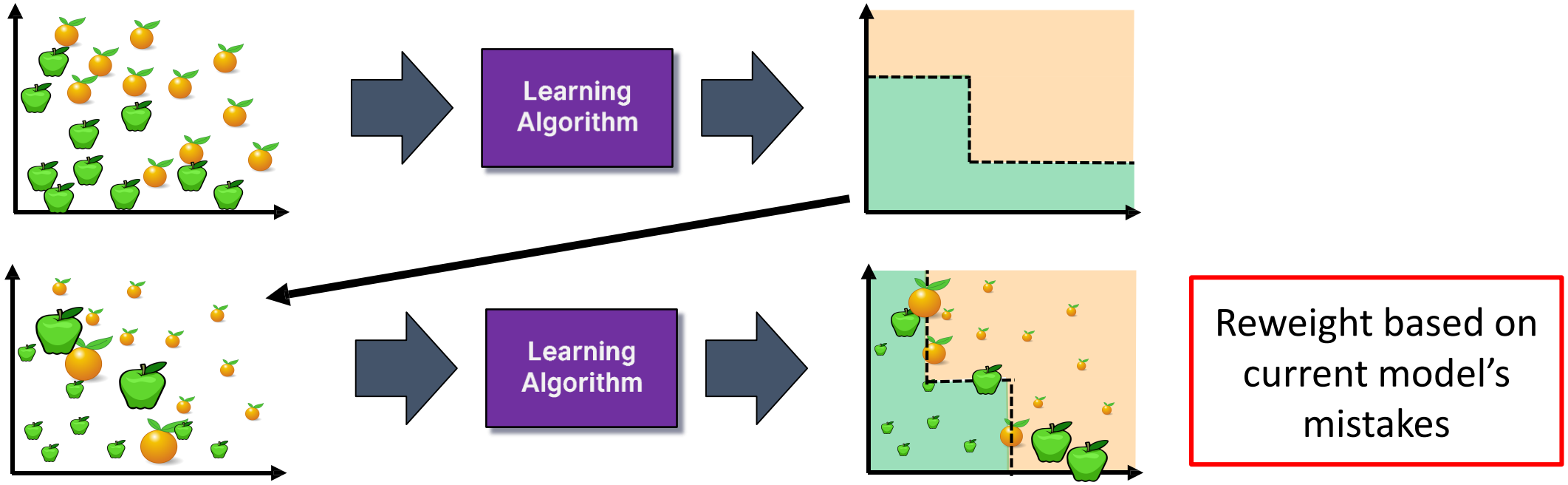
Boosting


All samples have the same weight
Learning Algorithm
Reweight based on model's mistakes

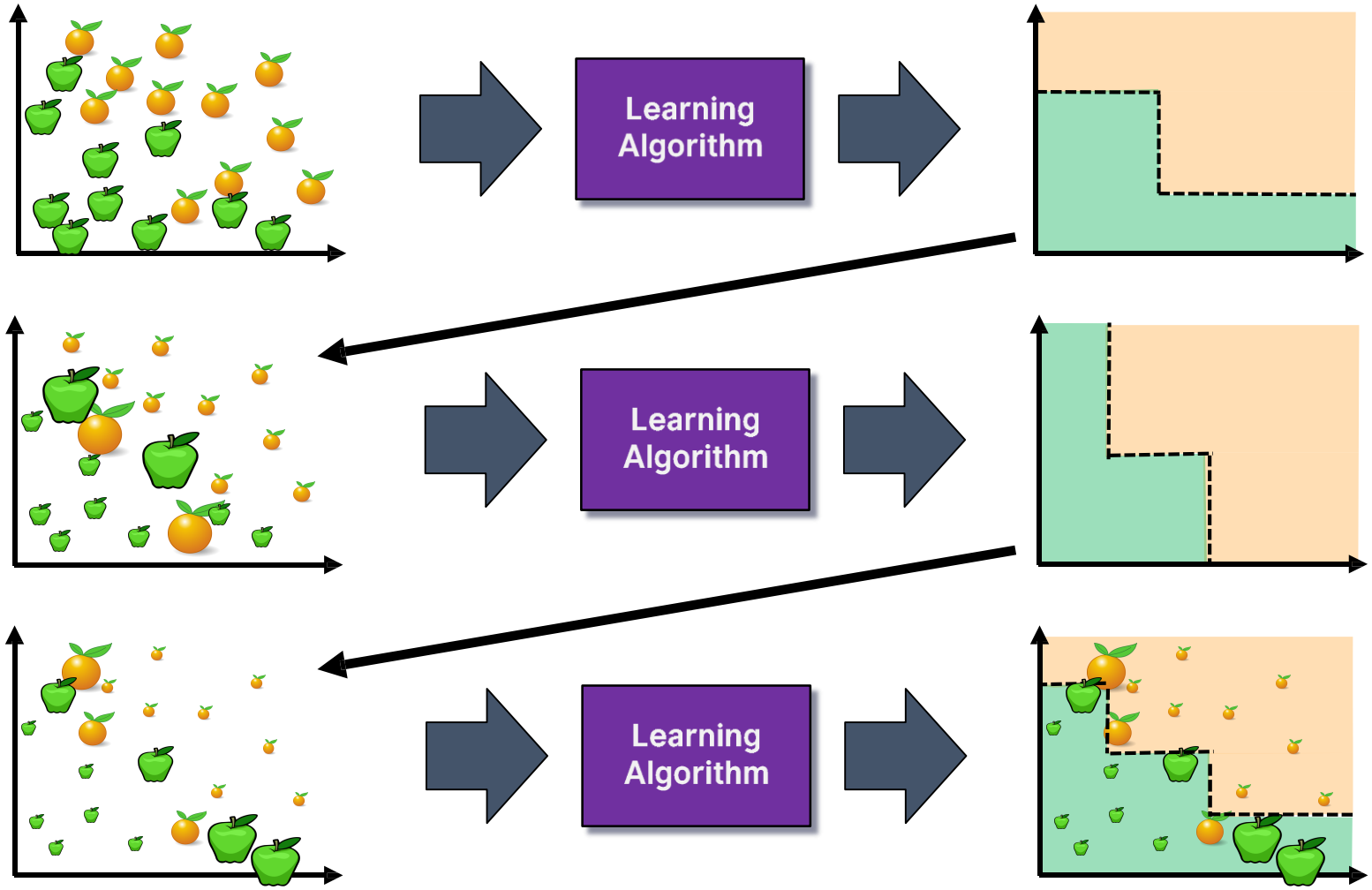
Boosting


Learning Algorithm
Learning Algorithm
Next model sees weighted samples

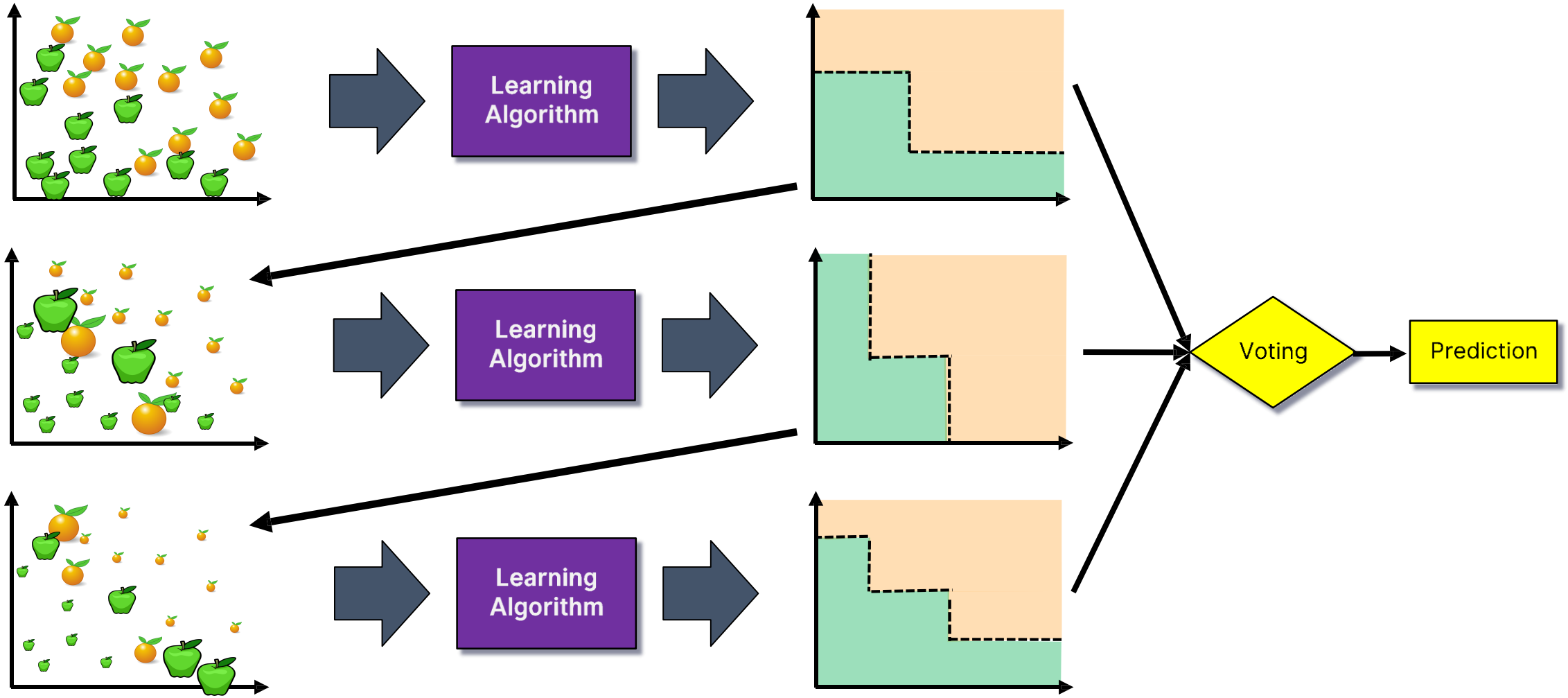
Boosting


Learning Algorithm
Learning Algorithm
Reweight based on current model's mistakes

Boosting


Learning Algorithm
Learning Algorithm
Learning Algorithm

Boosting


Learning Algorithm
Learning Algorithm
Learning Algorithm
Voting
Prediction

Decision Trees and Random Forest (Python)

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier

clf = DecisionTreeClassifier(criterion = "entropy",
min_samples_leaf = 3)
# Lots of parameters: criterion = "gini" / "entropy";
#                     max_depth;
#                     min_impurity_split;
clf.fit(X, y) # It can only handle numerical attributes!
# Categorical attributes need to be encoded, see LabelEncoder
and OneHotEncoder
clf.predict([x]) # Predict class for x
clf.feature_importances_ # Importance of each feature
clf.tree_ # The underlying tree object
clf = RandomForestClassifier(n_estimators = 20) # Random Forest
with 20 trees
```

Summary

- Ensemble Learning methods combine multiple learning algorithms to obtain performance improvements over its components
- Commonly-used ensemble methods:
  - Bagging (multiple models on random subsets of data samples)
  - Random Subspace Method (multiple models on random subsets of features)
  - Boosting (train models iteratively, while making the current model focus on the mistakes of the previous ones by increasing the weight of misclassified samples)
- **Random Forests** are an ensemble learning method that employ decision tree learning to build multiple trees through **bagging** and **random subspace method**.
  - They rectify the overfitting problem of decision trees!