# Machine Learning

## Chapter 3: Decision Trees

**Fall 2021**

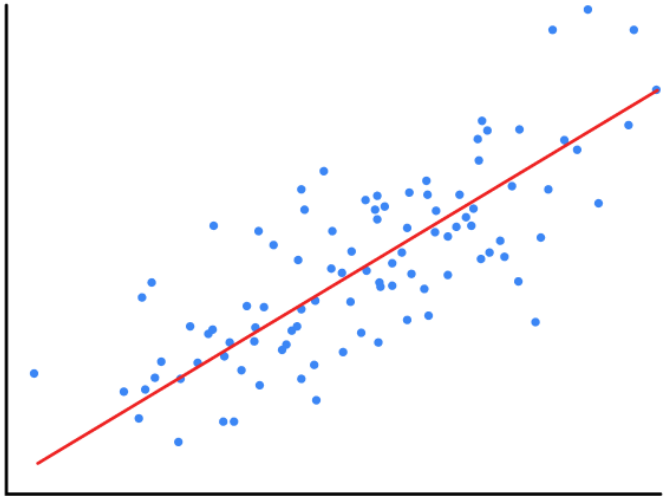
**Instructor: Xiaodong Gu**

# What we have learned so far?

Linear model for regression

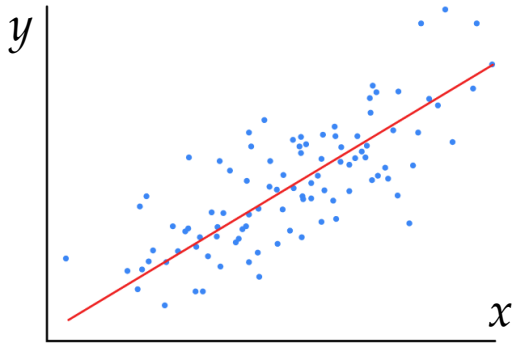
$$y = f_\theta(x) = \theta_0 + \sum_{j=1}^{d} \theta_j x_j$$

Linear model for regression is a (d+1)-dimensional hyperplane



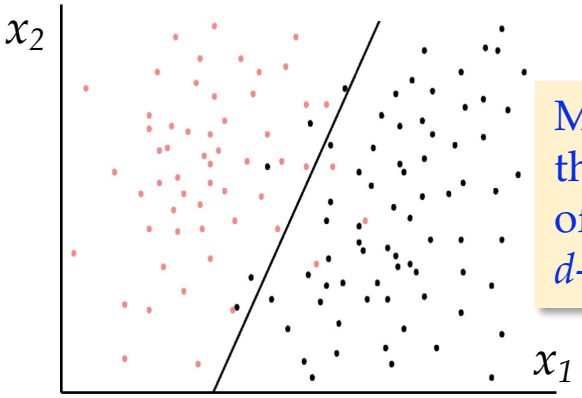
# Regression vs. Classification

- Machine Learning
  - **Supervised Learning**
    - Regression (✓)
    - Classification
    - …
  - Unsupervised Learning
  - Reinforcement Learning




Model the data points spread in (d+1)-dim space.

**Regression:** (predicts real-valued labels)



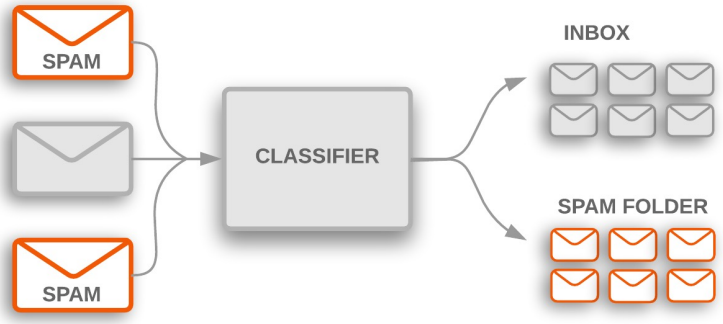

Model the boundaries that separate the data of different labels in *d*-dim space

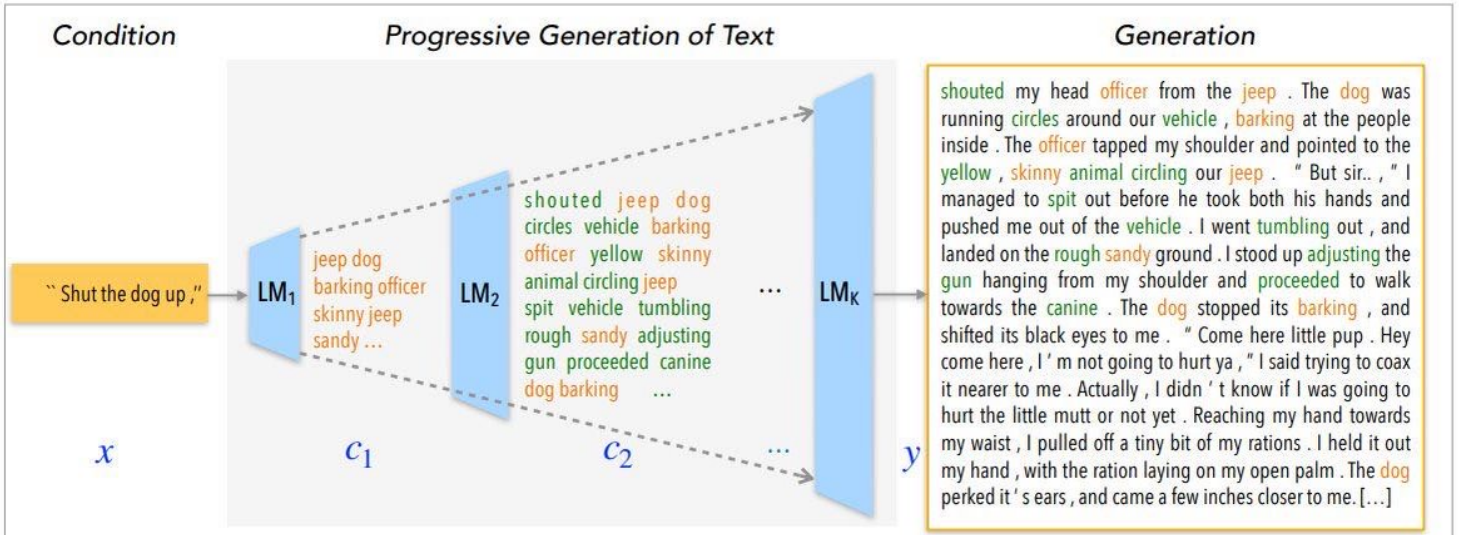
**Classification:** (Predicts categorical labels)

# Classification in Machine Learning

"Salmon"

"sea bass"

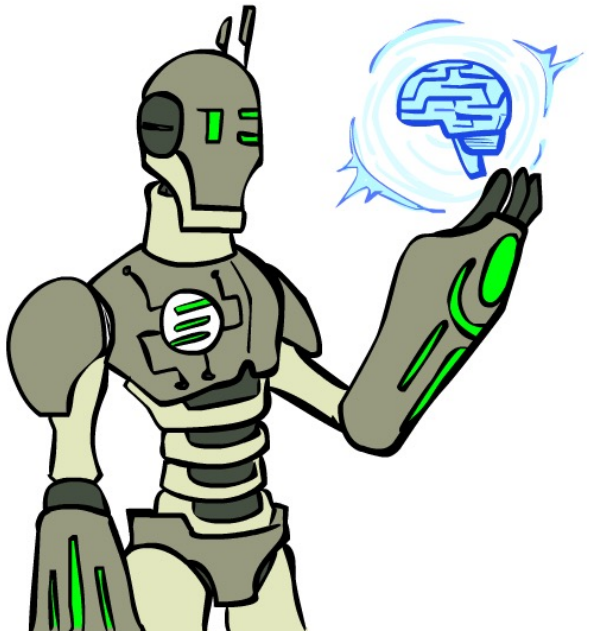


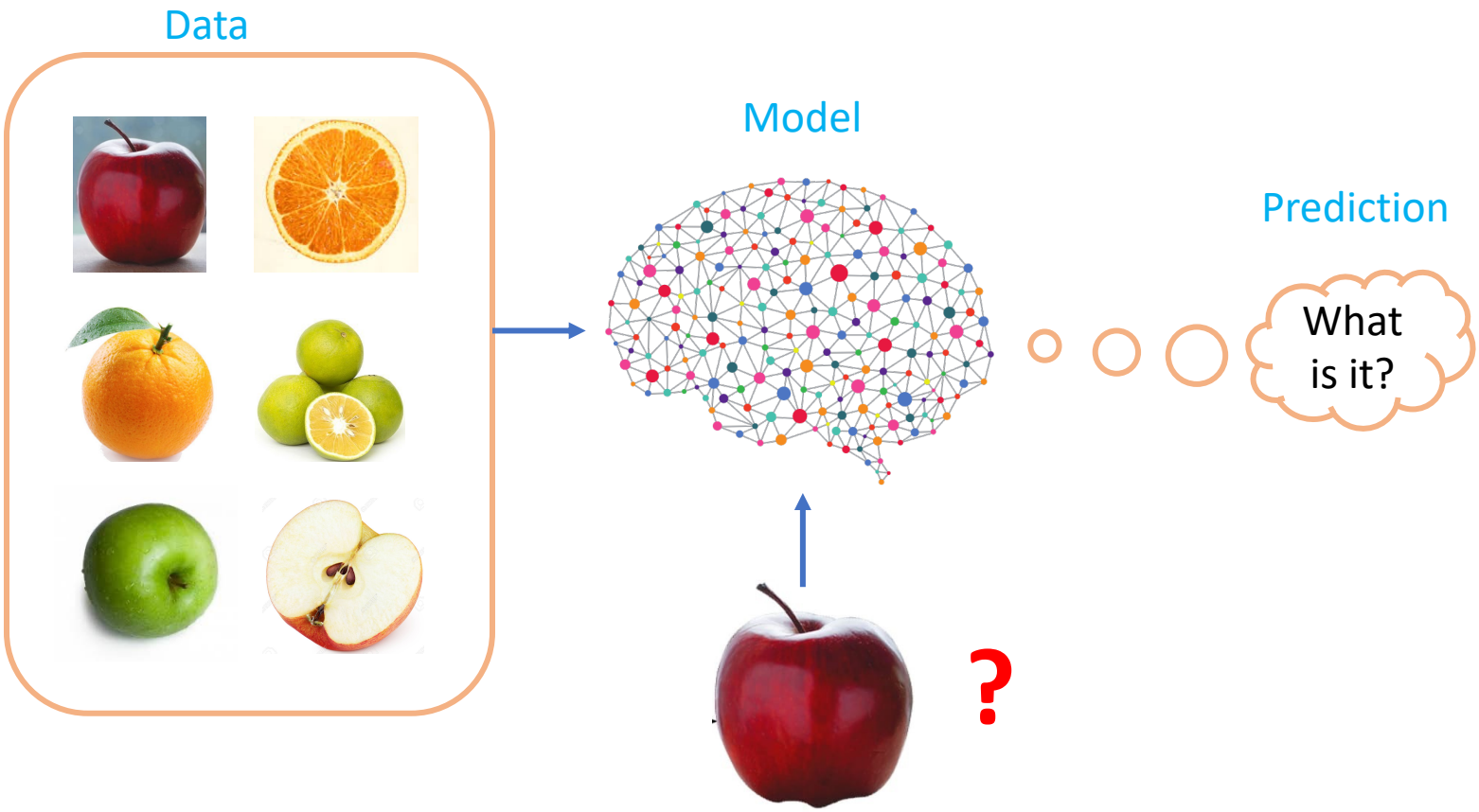
# Today

Let's start from a simple family of classification approaches

- Decision Trees



# Recap: Automatic Fruit Classification

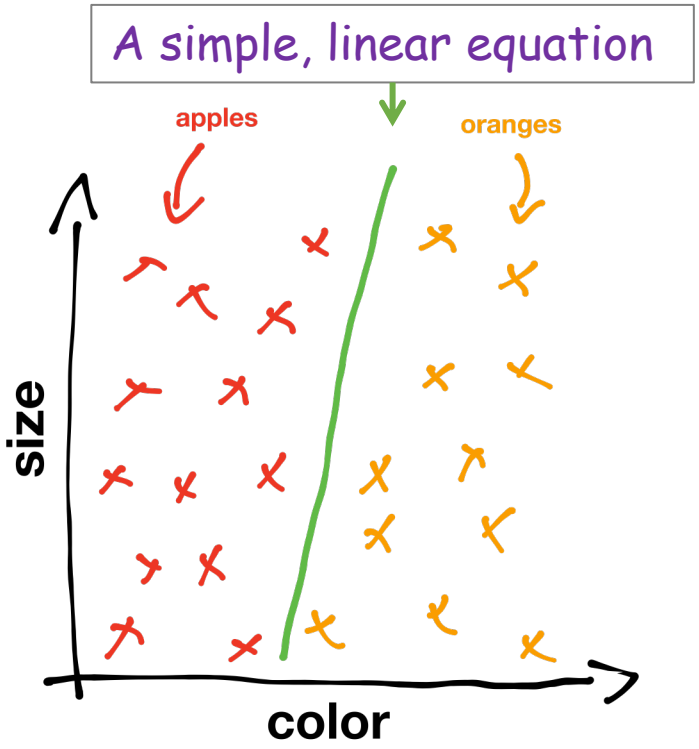


# Recap: Automatic Fruit Classification

## Learning (Training)

Partition the feature space into 2 regions, one for each type of fruit.

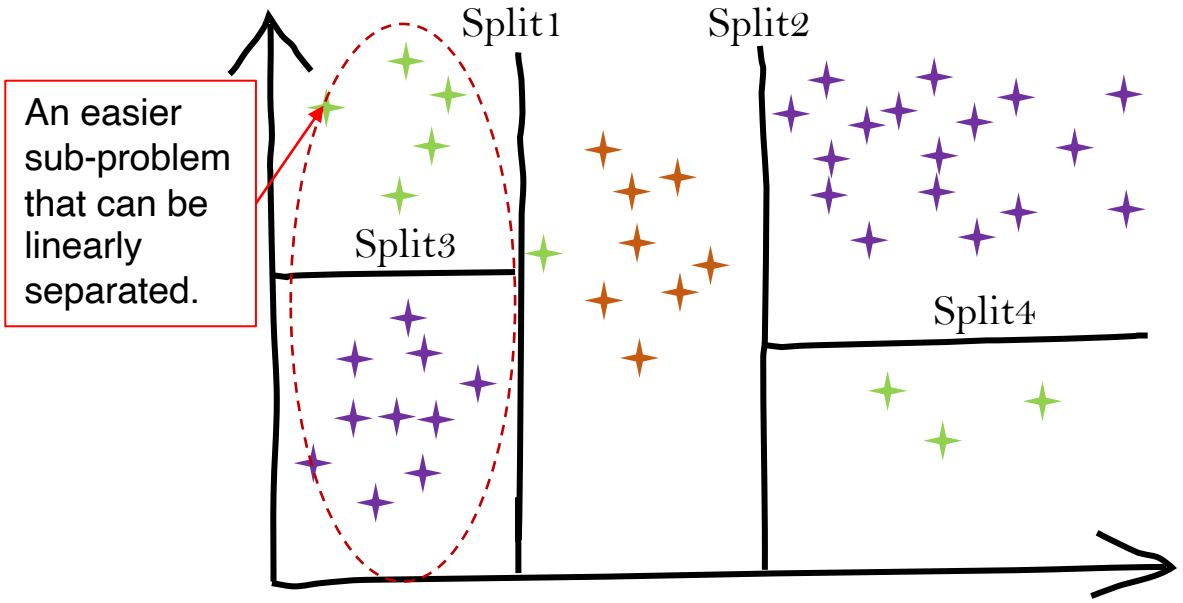
- decision boundary




How to represent the decision boundary?

# From Linear to Piecewise Linear

## What if the data is distributed like this?



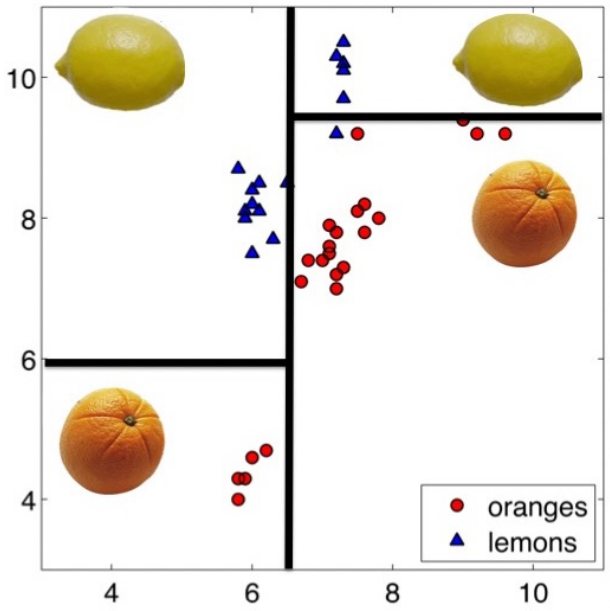

Combines a set of different linear models, each covering a region of the input space disjoint from the others

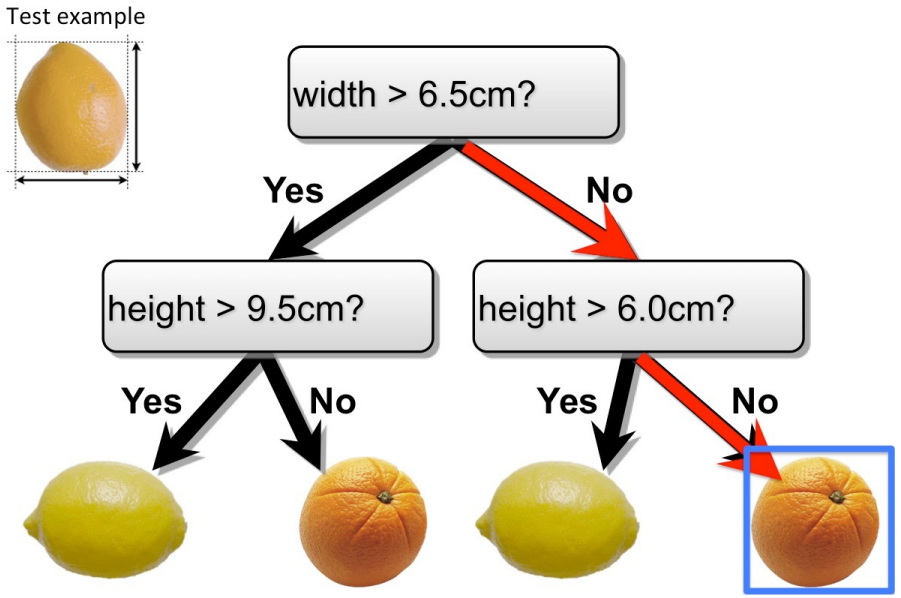
- Nonlinear (can not be linearly separated)
- But have local-piecewise linearity along axes
- Can be nominal (e.g., “male”, “female”)

# Decision Tree: The Key Idea

classify a data sample through a sequence of *if-then* questions.

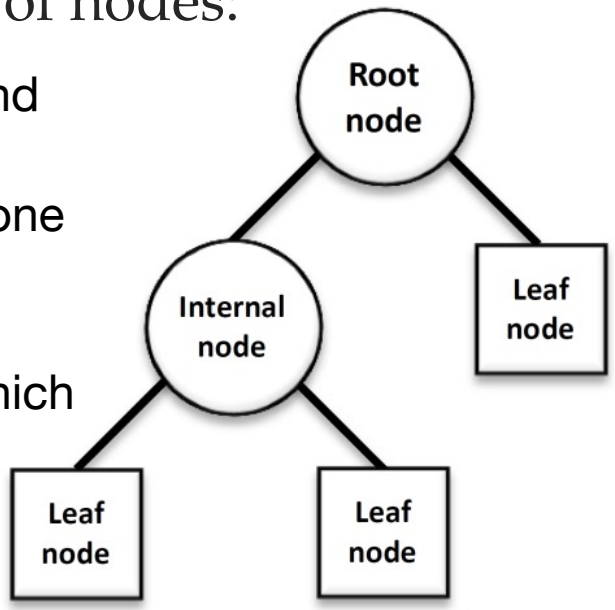
- Rule-Based
- Splitting Data Attributes.



# Model Structure

- A decision tree consists of three types of nodes:
  1. A root node that has no incoming edges and zero or mode outgoing edges
  2. Internal nodes, each of which has exactly one incoming edge and two or more outgoing edges
  3. Leaf nodes (or terminal nodes), each of which has exactly one incoming edge and no outgoing edges




- Each leaf node is assigned a class label
- Non-terminal nodes contain attribute test conditions to separate records that have different characteristics

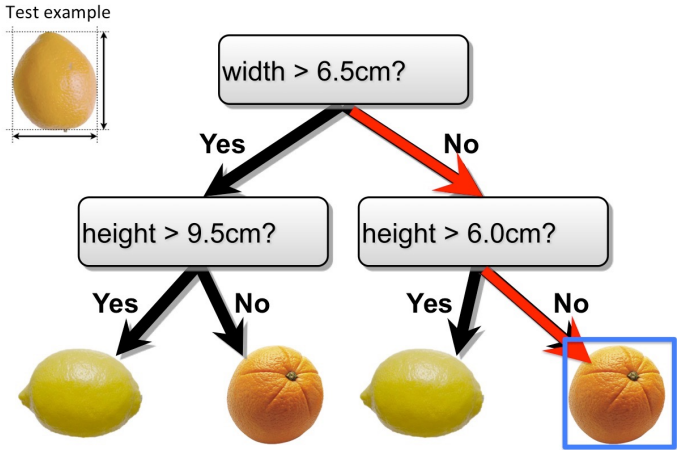
How to build (train) a decision tree?

# Training – (Build a Decision Tree)

## top-down divide-and-conquer learning procedure

1.Construct a root node which contains the whole data set.
2.Selecting an attribute that benefit the task most according to some criterion.
3.Split the examples of the current node into subsets based on values of the selected attributes.
4.Create a child node for each subset and passes the examples in the subset to the node.
5.Recursively repeat step 2~4 until some stopping criterion is met.

| ID | width | height | Type |
|---|---|---|---|
| 1 | 5.2 | 8.0 | pear |
| 2 | 6.7 | 9.8 | pear |
| 3 | 7.2 | 7.5 | orang |
| 4 | 6.1 | 5.3 | orang |
| 5 | 4.1 | 6.5 | pear |



# Key Problems

There could be more than one tree that fits the same data!

Exponentially many decision trees can be built.

**Q: How to efficiently construct the decision tree from data?**

Goal: minimize impurity of class Y as much as possible when selecting each attribute X.

Any decision tree will successively split the data into smaller and smaller subsets. It would be ideal if all the samples associated with a leaf node were from the same class. Such a subset, or node, is considered *pure* in this case.

**Q: How to select attribute to decrease impurity?**

A: Select X that has the maximum Information Gain, Chi-Square, Gini …

Each metric corresponds to a training algorithm.

# Training Algorithms


ID3 (Information Gain)

C4.5 (Gain Ratio)

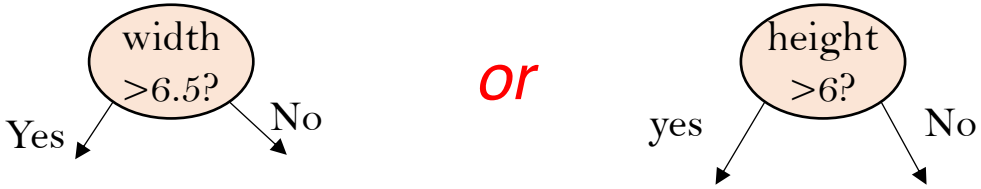
CART (Gini Index)

…

# ID3

Split attributes that have the maximum Information Gain.

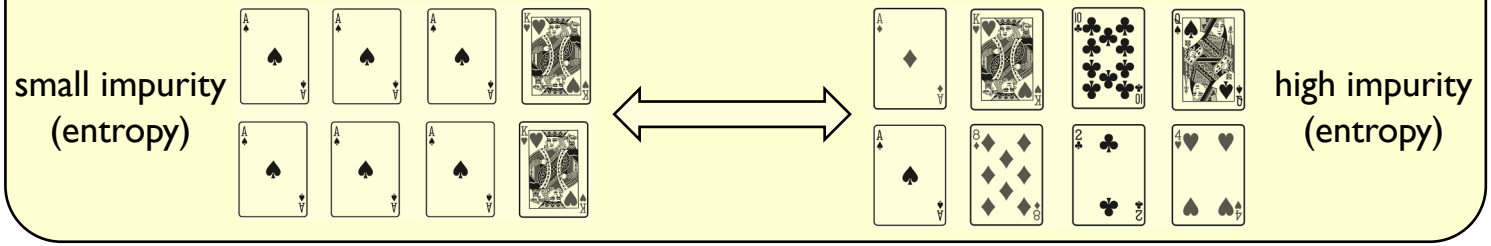
- Try to split on each possible attribute in a dataset. See which split works "best".




- Measure the reduction in the overall entropy of a set of instances

Recap: Entropy – measures the impurity of the elements of a set.

$$H(D) = -\sum_{k=1}^{K} p_k \log p_k$$



# ID3

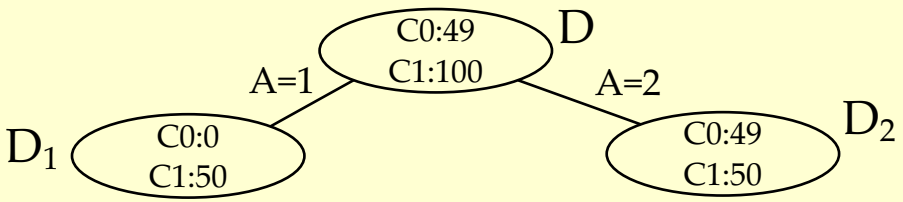
$$H(D) = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log \frac{|C_k|}{|D|}$$ ($|C_k|$: # samples of class $C_k$ in dataset D)

$$H(D|A) = \sum_{i=1}^{n} \frac{|D_i|}{|D|} H(D_i) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \left(\sum_{k=1}^{K} \frac{|D_{ik}|}{|D_i|} \log \frac{|D_{ik}|}{|D_i|}\right)$$

($|D_i|$: # samples whose attribute A is set to the i-th value in D; $|D_{ik}|$: #samples of class $C_k$ in $D_i$)

$$\text{Gain}(D,A) = H(D) - H(D|A)$$

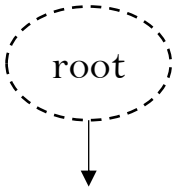
What is the information gain of this split?




- Root entropy: $H(D) = -49/149 \log(49/149) - 100/149 \log(100/149) \approx 0.91$
- Leaves entropy: $H(D|A=1) = 0$, $H(D|A=2) \approx 1$
- IG(D|A) $\approx 0.91 - \left(\frac{1}{3} \cdot 0 + \frac{2}{3} \cdot 1\right) \approx 0.24 > 0$

# ID3 – Example

| ID | width | height | Type |
|---|---|---|---|
| 1 | 5.2 | 8.0 | pear |
| 2 | 6.7 | 9.8 | pear |
| 3 | 7.2 | 7.5 | orang |
| 4 | 6.1 | 5.3 | orang |
| 5 | 4.1 | 6.5 | pear |




$\{1,2,3,4,5\}$

1. create a root node

2. calculate the entropy of the whole (sub) dataset.

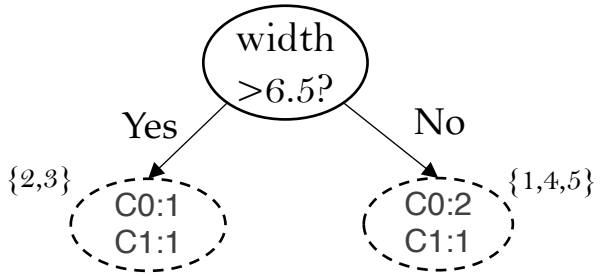
| pear | orange |
|---|---|
| 3 | 2 |

$H(D) = - 3/5 \log 3/5 – 2/5 \log 2/5 = 0.97$

# ID3 – Example

3. calculate the IG of each single attribute and pick the attribute with the max IG.

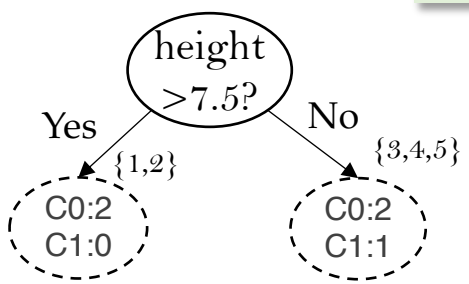
| ID | width | Type |
|---|---|---|
| 1 | 5.2 | pear |
| 2 | 6.7 | pear |
| 3 | 7.2 | orang |
| 4 | 6.1 | orang |
| 5 | 4.1 | pear |




H=-1/2log1/2-1/2log1/2 = 1

H=-2/3log2/3-1/3log1/3 =0.92

| ID | height | Type |
|---|---|---|
| 1 | 8.0 | pear |
| 2 | 9.8 | pear |
| 3 | 7.5 | orang |
| 4 | 5.3 | orang |
| 5 | 6.5 | pear |




H=-1log1-0log0 = 0

H=-2/3log2/3-1/3log1/3 =0.92

Which attribute shall we split on?

H(D|width) = 2/5 H(D|width>6.5) + 3/5H(D|width<6.5)
= 2/5 x 1 + 3/5 x 0.92 = 0.95
Gain(D|width) = H(D) – H(D|width) = 0.97-0.95=0.02

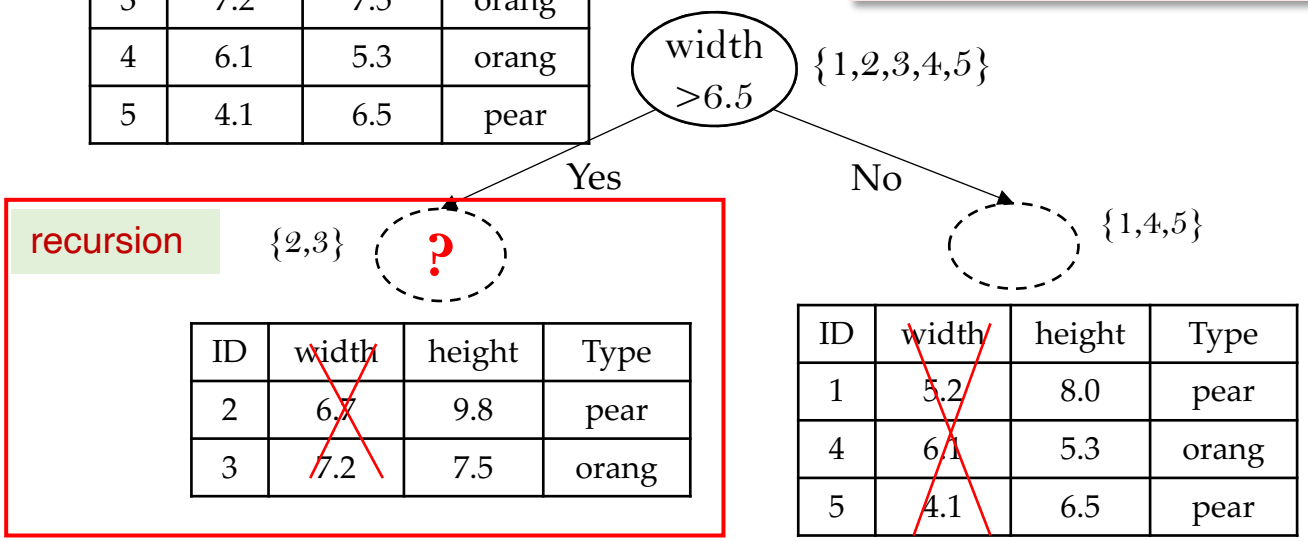
H(D|height) = 2/5 H(D|height>7.5) + 3/5H(D|height<7.5)
= 2/5 x 0 + 3/5 x 0.92 = 0.55
Gain(D|height) = H(D) – H(D|height) = 0.97-0.55=0.42

# ID3 – Example

4. Assign the (root) node the attribute with max IG. Grow for each attribute value an outgoing branch and add unlabeled nodes at the end.

| ID | width | height | Type |
|---|---|---|---|
| 1 | 5.2 | 8.0 | pear |
| 2 | 6.7 | 9.8 | pear |
| 3 | 7.2 | 7.5 | orang |
| 4 | 6.1 | 5.3 | orang |
| 5 | 4.1 | 6.5 | pear |

5. Split the dataset along the values of the maxIG feature and remove this feature from the dataset.




| ID | ~~width~~ | height | Type |
|---|---|---|---|
| 2 | ~~6.7~~ | 9.8 | pear |
| 3 | ~~7.2~~ | 7.5 | orang |

| ID | ~~width~~ | height | Type |
|---|---|---|---|
| 1 | ~~5.2~~ | 8.0 | pear |
| 4 | ~~6.1~~ | 5.3 | orang |
| 5 | ~~4.1~~ | 6.5 | pear |

6. For each of the sub-datasets, repeat steps 2-5 until a stopping criteria is satisfied → here the recursion kicks in.

# CART

find the best split using Gini Index.

$$\text{Gini (D)} = \sum_{k=1}^{K} \frac{|C_k|}{|D|}\left(1 - \frac{|C_k|}{|D|}\right) = 1 - \sum_{k=1}^{K}\left(\frac{|C_k|}{|D|}\right)^2$$

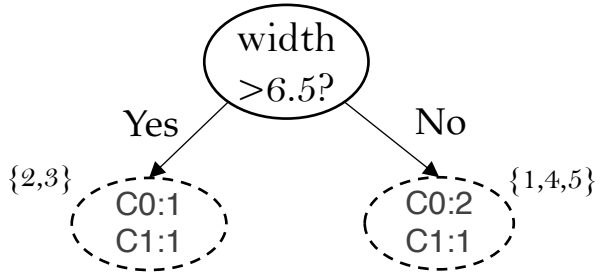
$$\text{Gini (D | A)} = \sum_{i=1}^{n} \frac{|D_i|}{|D|} \text{Gini}(D_i)$$

- Gini represents the probability that two randomly selected samples come from different classes.

Gini is cheaper in computation than Entropy which needs to compute *log* functions.

# CART – Example

| ID | width | Type |
|---|---|---|
| 1 | 5.2 | pear |
| 2 | 6.7 | pear |
| 3 | 7.2 | orang |
| 4 | 6.1 | orang |
| 5 | 4.1 | pear |

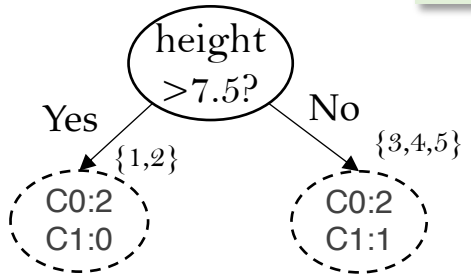



Gini = 1-(1/2)$^2$–(1/2)$^2$ = 0.5

Gini = 1-(2/3)$^2$-(1/3)$^2$ =0.44

Gini(Dlwidth) = 2/5Gini(Dlwidth>6.5) + 3/5Gini(Dlwidth<6.5)
= 2/5 x 0.5 + 3/5 x 0.44 =
Gain (Dlwidth) = Gini(D) – Gini(Dlwidth) =

| ID | height | Type |
|---|---|---|
| 1 | 8.0 | pear |
| 2 | 9.8 | pear |
| 3 | 7.5 | orang |
| 4 | 5.3 | orang |
| 5 | 6.5 | pear |




Gini=1-(2/2)$^2$-(0/2)$^2$ =0

Gini=1-(2/3)$^2$-(1/3)$^2$ =0.44

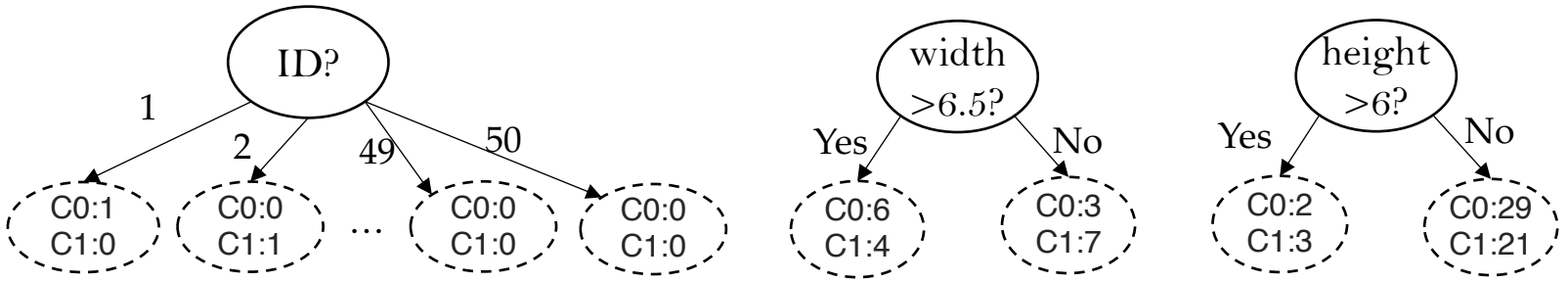
Gini(Dlheight) = 2/5Gini(Dlheight>7.5)+3/5Gini(Dlheight<7.5)
= 2/5 x 0 + 3/5 x 0.44 =
Gain (Dlheight) = Gini(D) – Gini(Dlheight) =

Which attribute shall we split on?

# Shortcoming of ID3 and CART

**Entropy and Gini impurity measures favor attributes with large number of distinct values.**

Possible nodes to split on:




- ID will result in perfectly pure children.
- Will have the greatest information gain.
- Should have been removed as a predictor variable.

# C4.5

To avoid bias of favoring attributes with large number of distinct values, C4.5 uses gain ratio instead of information gain.

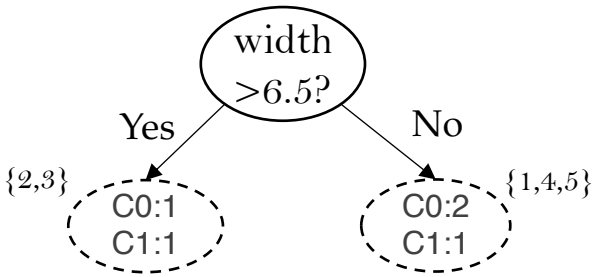
- takes into account the number of outcomes produced by attribute split condition.
- Adjusts information gain by the entropy of the partitioning.

$$\textbf{Gain Ratio}\ (D, A) = \frac{\text{Gain}(D,A)}{H_A(D)}$$

$$H_A(D) = -\sum_{i=1}^{n} \frac{|D_i|}{|D|} \log \frac{|D_i|}{|D|}$$

# C4.5 – Example

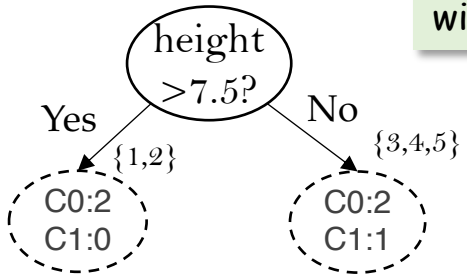
| ID | width | Type |
|---|---|---|
| 1 | 5.2 | pear |
| 2 | 6.7 | pear |
| 3 | 7.2 | orang |
| 4 | 6.1 | orang |
| 5 | 4.1 | pear |




H=-1/2log1/2-1/2log1/2 = 1

H=-2/3log2/3-1/3log1/3 =0.92

| ID | height | Type |
|---|---|---|
| 1 | 8.0 | pear |
| 2 | 9.8 | pear |
| 3 | 7.5 | orang |
| 4 | 5.3 | orang |
| 5 | 6.5 | pear |




H=-1log1-0log0 = 0

H=-2/3log2/3-1/3log1/3 =0.92

Which attribute shall we split on? width

H(D|width) = 2/5 H(D|width>6.5) + 3/5H(D|width<6.5)
= 2/5 x 1 + 3/5 x 0.92 = 0.95
Gain (D|width) = H(D) – H(D|width) = 0.97-0.95=0.02
$H_{width}(D)$ = - 2/5 log 2/5 – 3/5 log 3/5 = 0.97
GainRatio (D|width) = 0.02 /0.97= 0.02

H(D|height) = 2/5 H(D|height>7.5) + 3/5H(D|height<7.5)
= 2/5 x 0 + 3/5 x 0.92 = 0.55
Gain (D|height) = H(D) –H(D|height) = 0.97-0.55 = 0.42
$H_{height}(D)$ = -2/5 log 2/5 – 3/5 log 3/5 = 0.97
GainRatio (D|height) = 0.43 /0.97= 0.5

# Stop Criteria


- Purity

  The leaves contain the training examples from the same class

- Minimum number of points

  The number of training examples contained in the leaves are less than a threshold

- No more attribute to used for split (ID3)

# Pruning

- Why pruning?
  - To reduce the chance of overfitting

- Pruning strategy
  - Prepruning

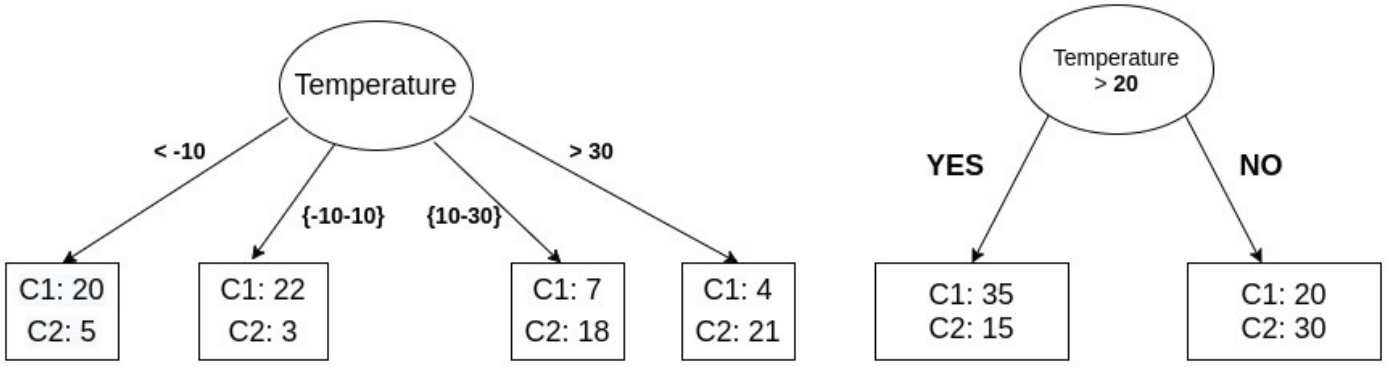
    Stop growing tree early if the goodness measure is less than a threshold
  - Postpruning

    Remove branches after a tree has been fully grown.

postpruning usually performs better than prepruning, but its computational cost is heavier.

# Splitting Continuous-valued Attributes

- Convert a continuous-valued attribute into a categorical attribute by splitting it into *n* range buckets with equal width.
  - The number of categories (buckets) is a hyper-parameter.
  - More sophisticated methods involve the use of unsupervised clustering algorithms to define the optimal categories.
- Binary split using a comparison operator (≥, ≤)
  - Need to determine the threshold which is computationally expensive.
  - Can sort the values of the continuous attribute and take the midpoint.



# Evaluation

Test set: $\{(x_1, y_1), \ldots, (x_n, y_n)\}$

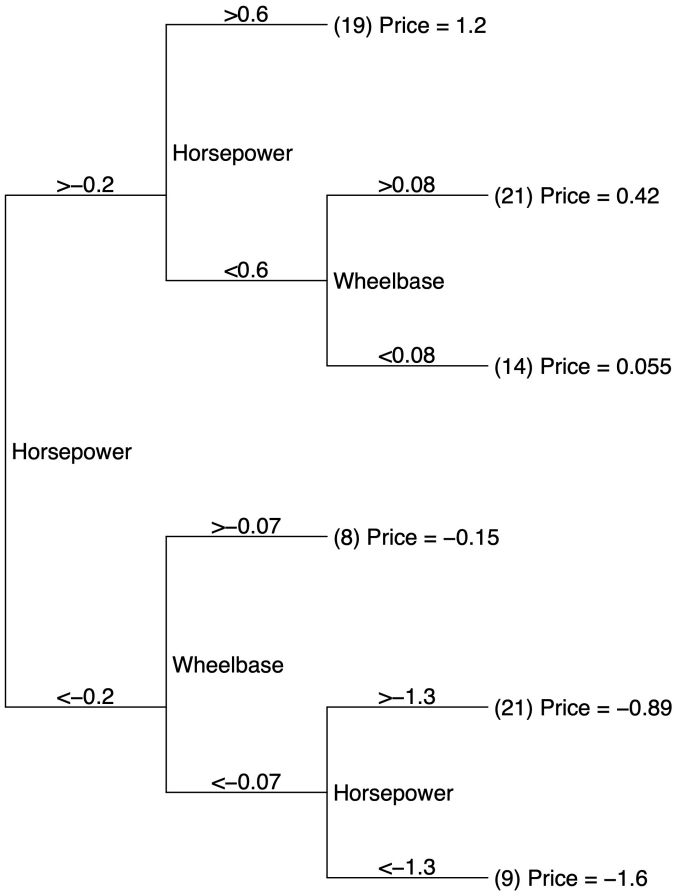
Error Rate: proportion of mistakes that are made by applying the tree to the testing samples:

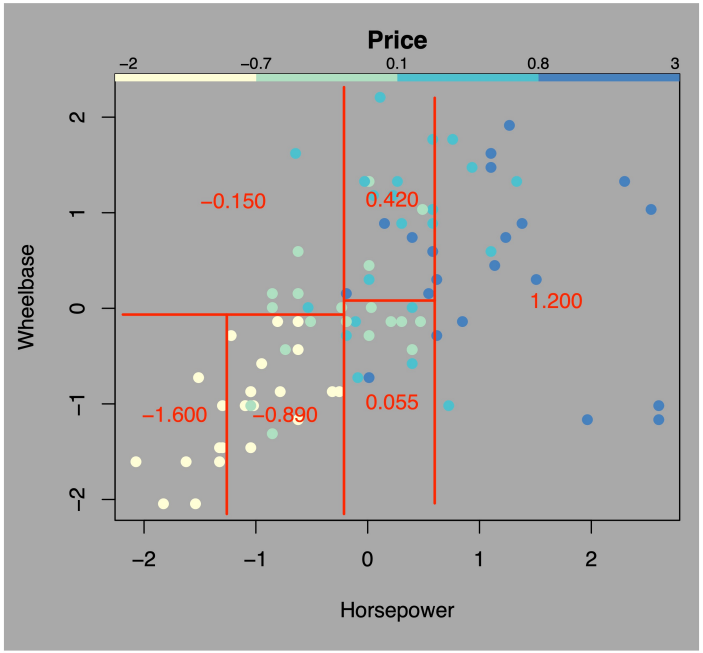
$$\text{Error Rate} = \frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

- $\hat{y}_i$ is the predicted class for the i-th record.
- I() is an indicator variable: I = 1 if $y_i \neq \hat{y}_i$ and 0 otherwise.

# Regression Trees*

Regression: $y = \theta_0 + \theta_1 x_1 + \ldots + \theta_d x_d$



# Decision Tree vs. Regression Tree*

## Classification Trees

- Gain of a split: "reduction of impurity.
- Larger gain => better split

$$\text{gain} = \text{H(parent)} - \Sigma_i \, p_i \, \text{H}(\text{ch}_i)$$

## Regression Trees

- Impurity (variance) at a node:

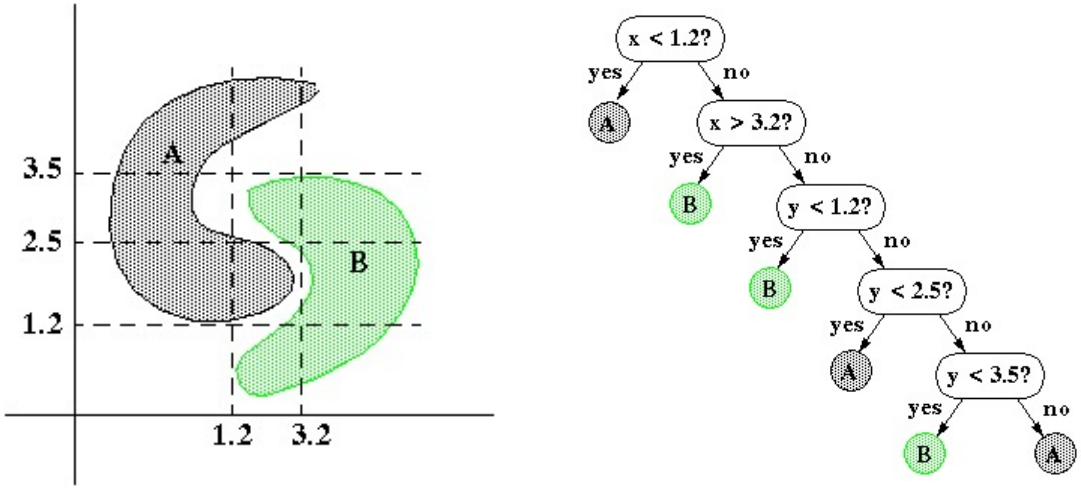
$$\text{var}(t, D) = \frac{\sum_{i=1}^{n} (t_i - \bar{t})^2}{n - 1}$$

- Select feature to split on that minimizes the weighted variance across all resulting partitions

# Pros and Cons

## Advantages:

- Simple to understand, explain, and visualization,
- Little effort for data preprocessing,
- can produce nonlinear decision surfaces,
- data-driven and can give arbitrarily high levels of precision on the training data.



# Pros and Cons

**Disadvantages:**

- overfitting