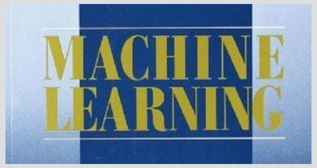
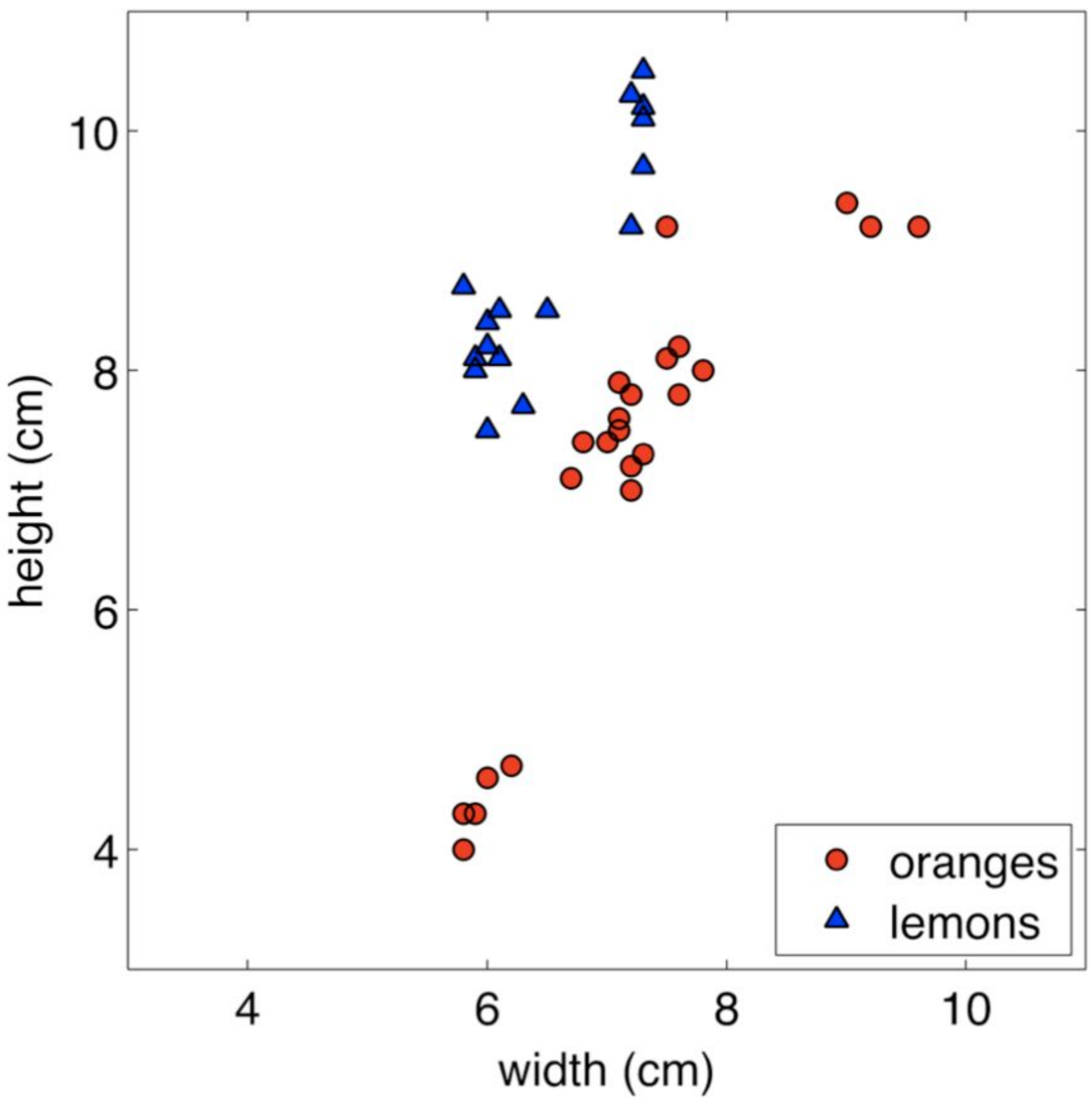
Decision Trees



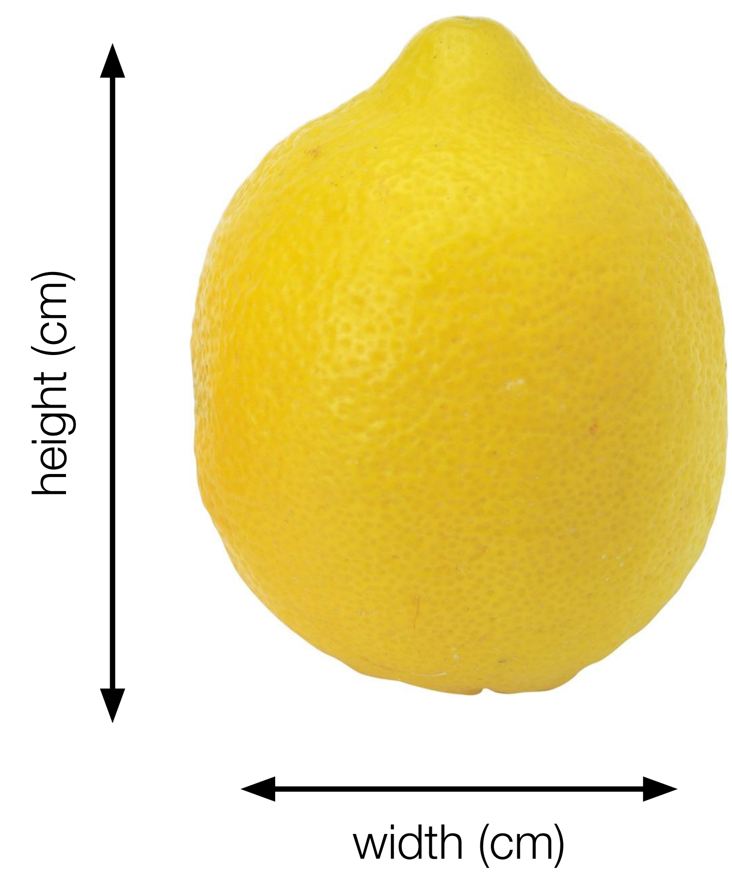
Pages
52-73

Decision Trees - Intuition

- Let's take another look at our toy classification task:

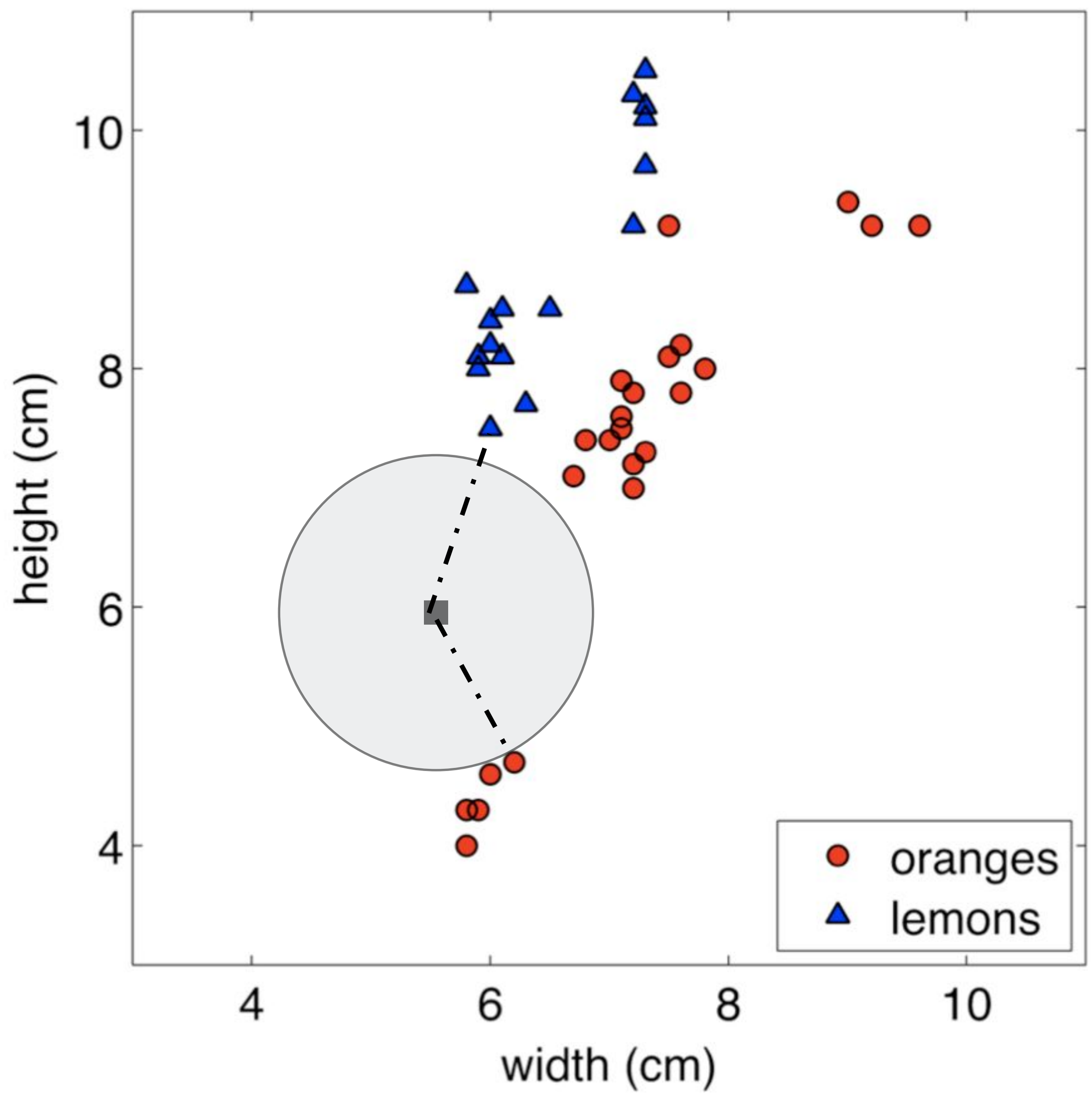


Binary classifier based on two simple features:

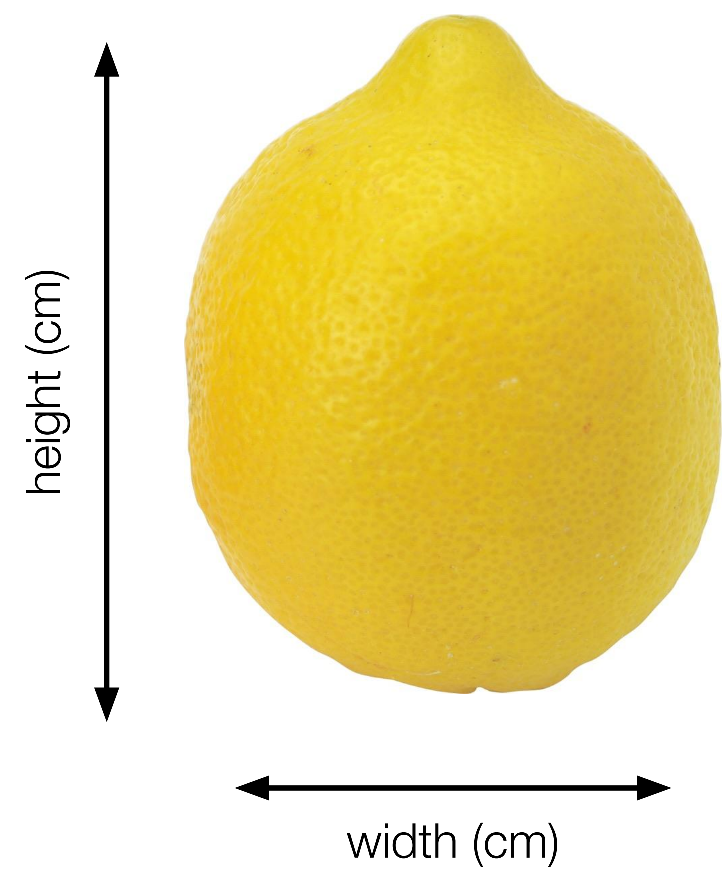


Decision Trees - Intuition

- k-NN: “oranges” vs “lemons”

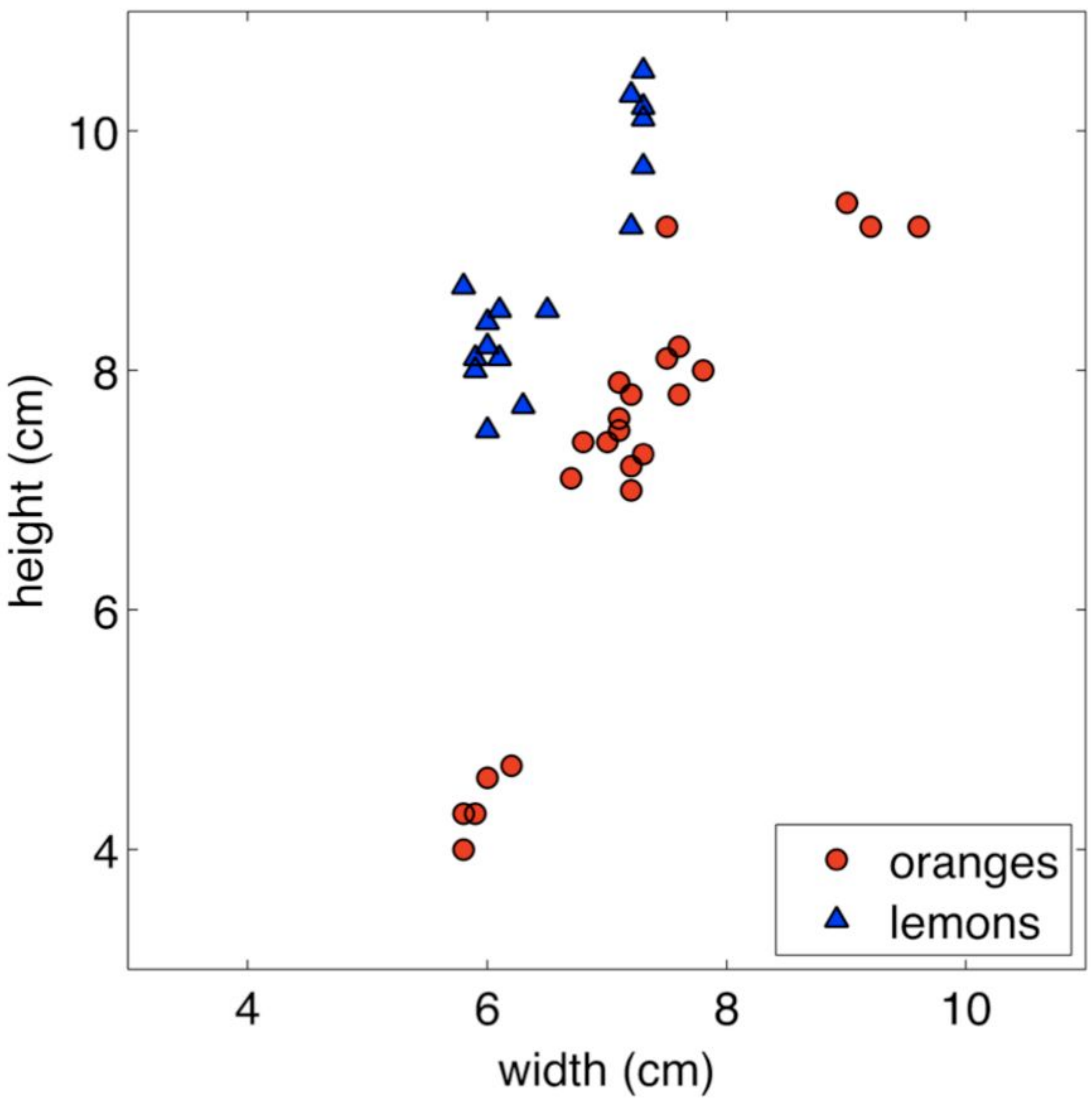


Binary classifier based on two simple features:

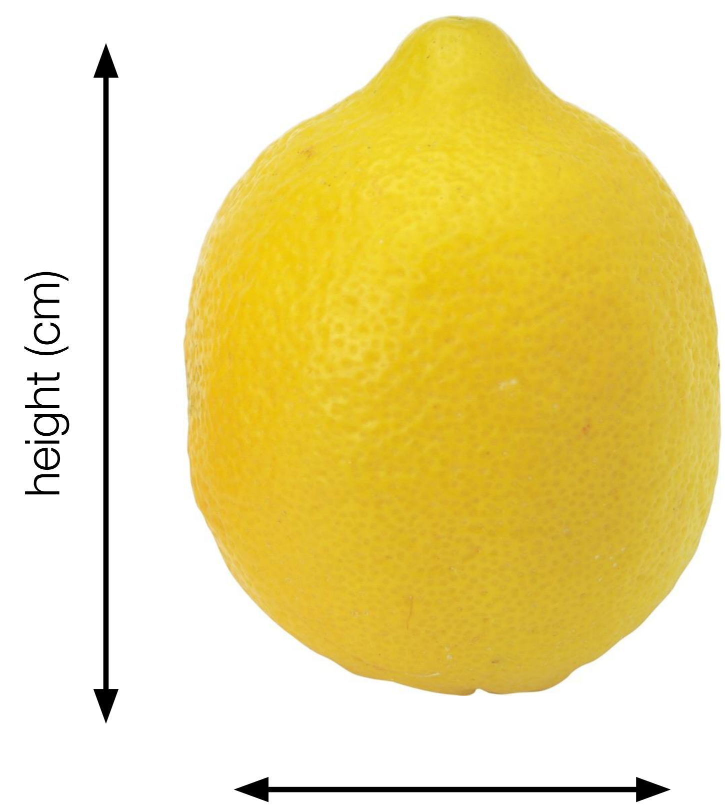


Decision Trees - Intuition

- Let’s take another look at our toy classification task:



Binary classifier based on two simple features:

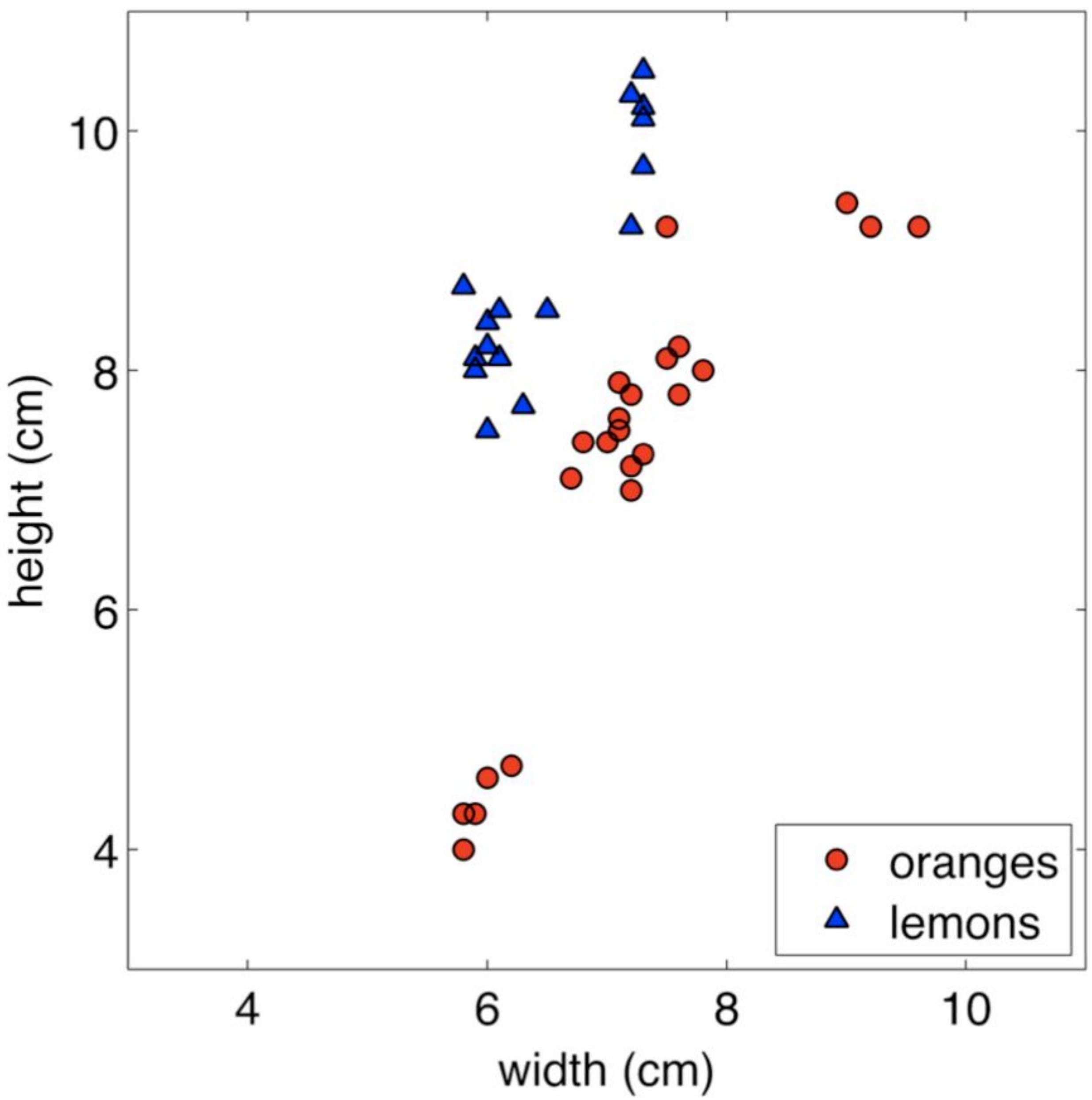


Decision Trees

- A *decision tree* is a structure in which:
  - Internal nodes represent attributes (features)
  - Leaf nodes represent class labels or target values
- Decision trees are widely used in operations research to support decision analysis
- They are also a popular algorithm in AI/ML
  - A good property of decision trees is that they provide <u>interpretable</u> results
  - A decision tree represents the hypothesis function $h(\mathbf{x})$ we want to model through machine learning

Decision Trees - Intuition

- Decision tree classifier: “oranges” vs “lemons”

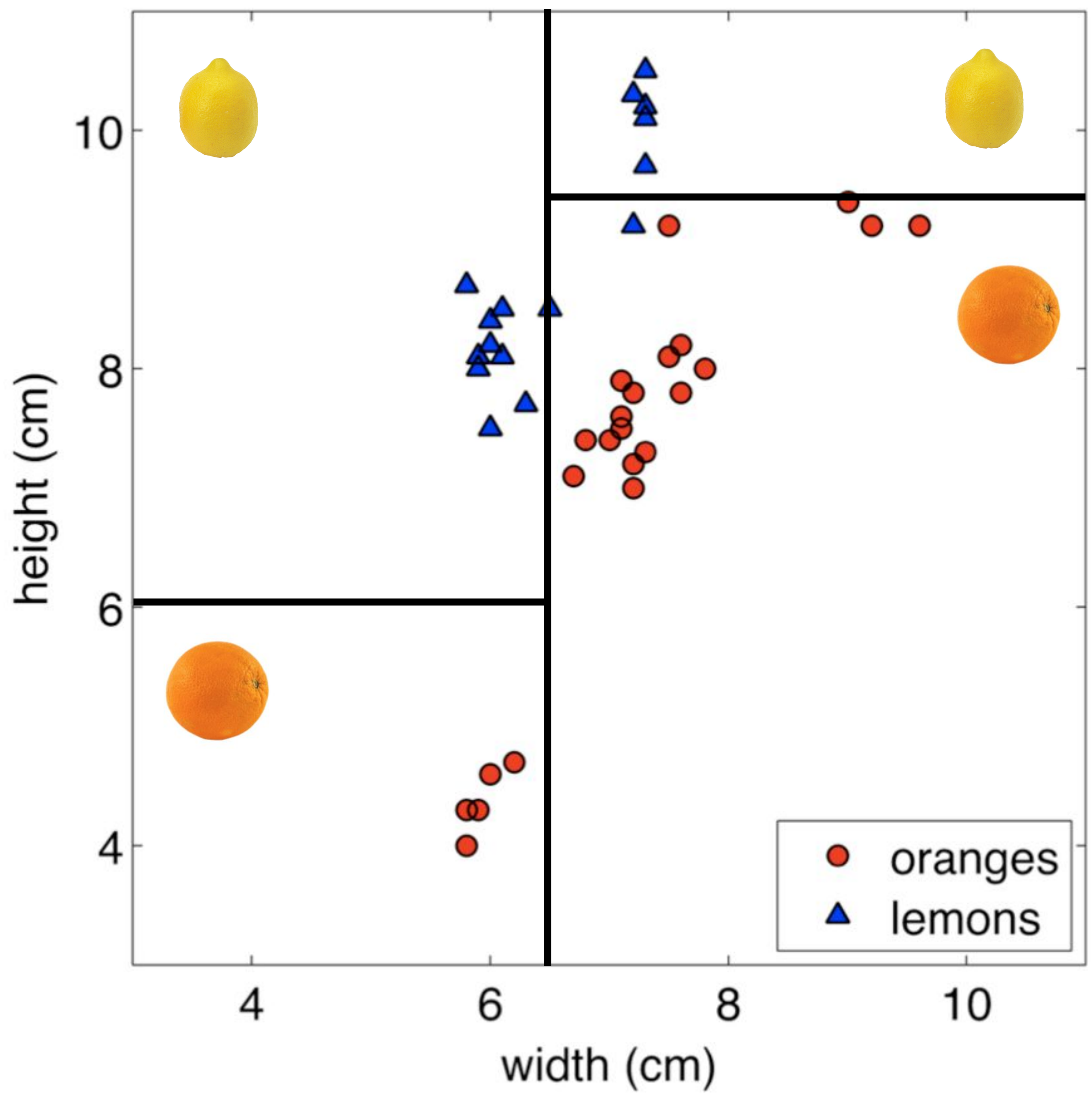


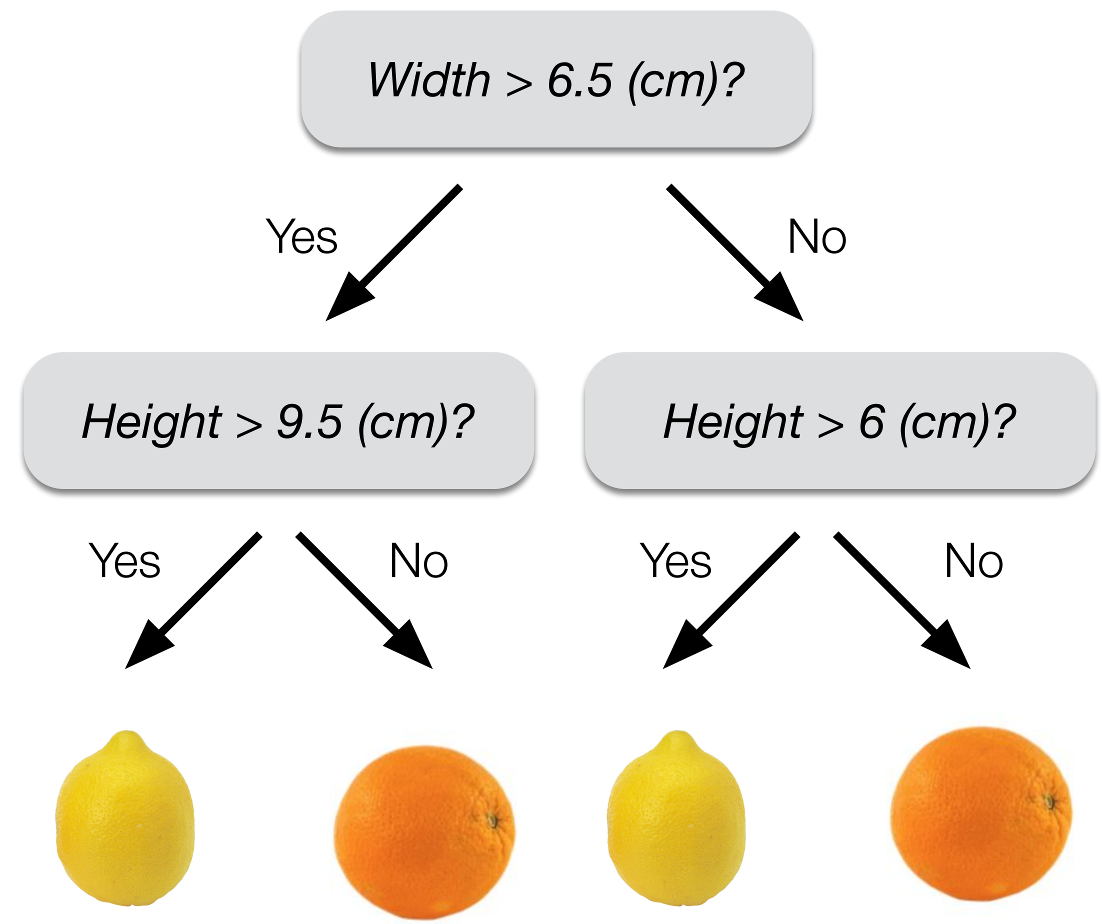
General idea:

- Pick an attribute and do a simple test
- Conditioned on a choice, pick another attribute
- In the leaves, assign a class with majority vote
- Do other branches as well

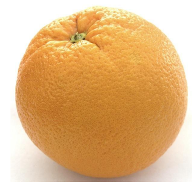
Decision Trees - Intuition

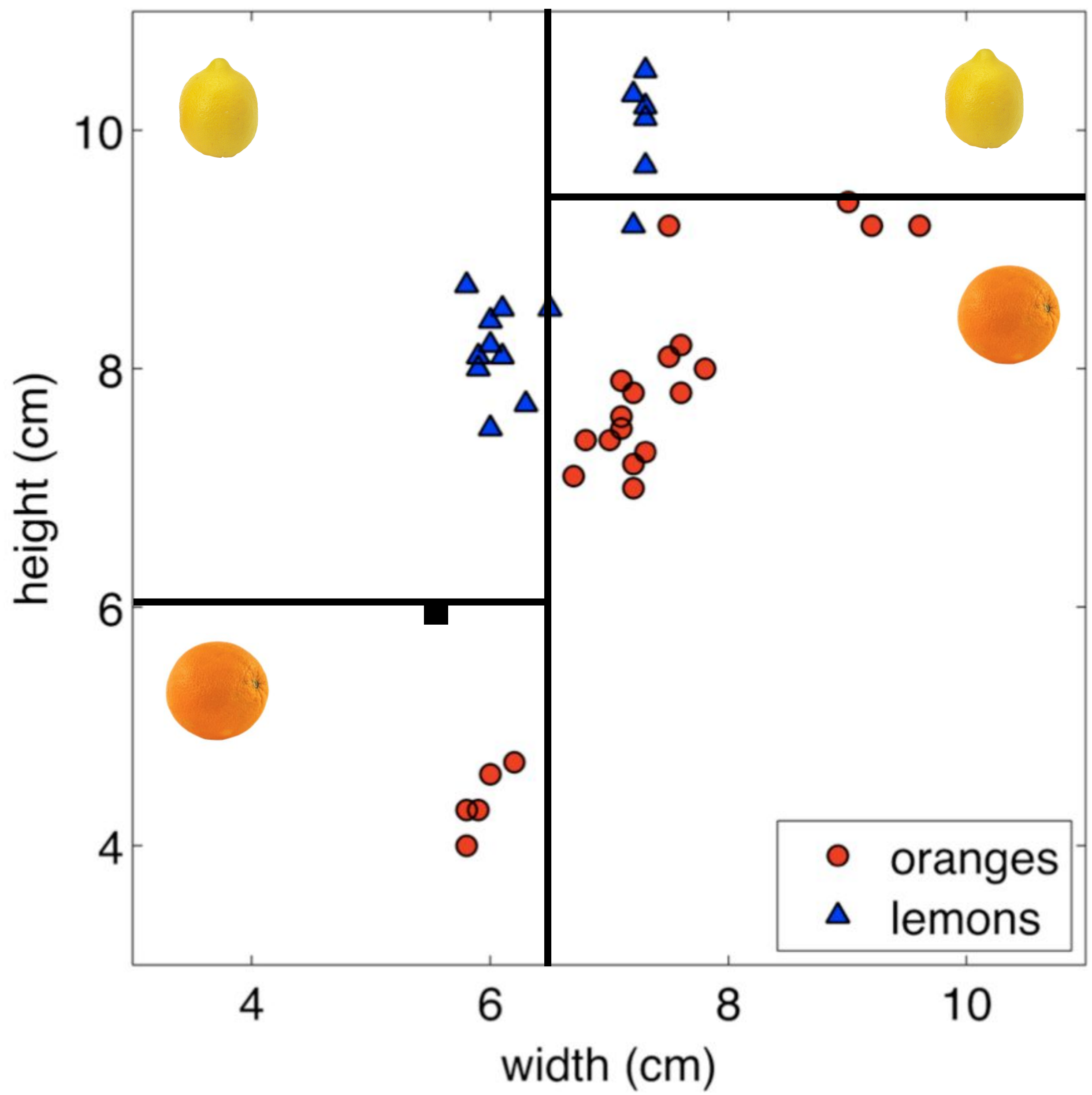
- Decision tree classifier: "oranges" vs "lemons"

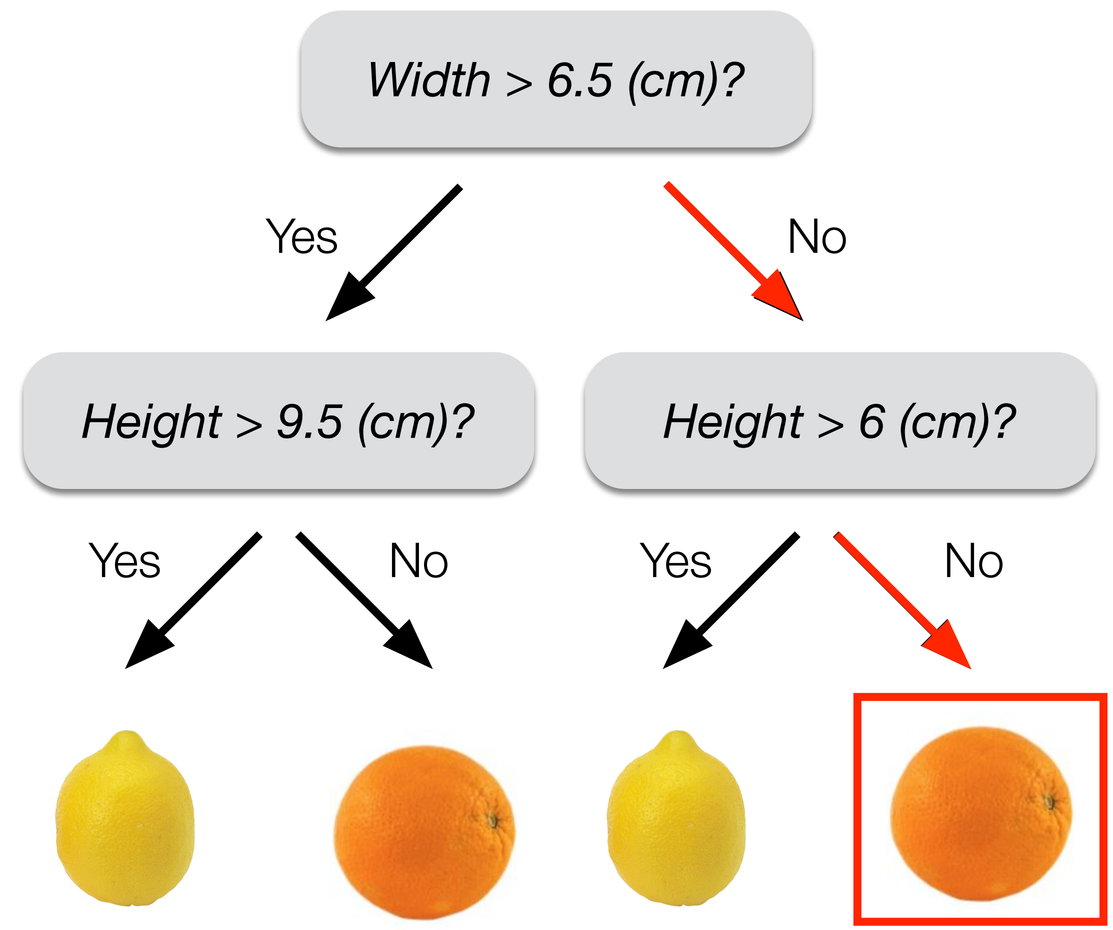




Decision Trees - Intuition

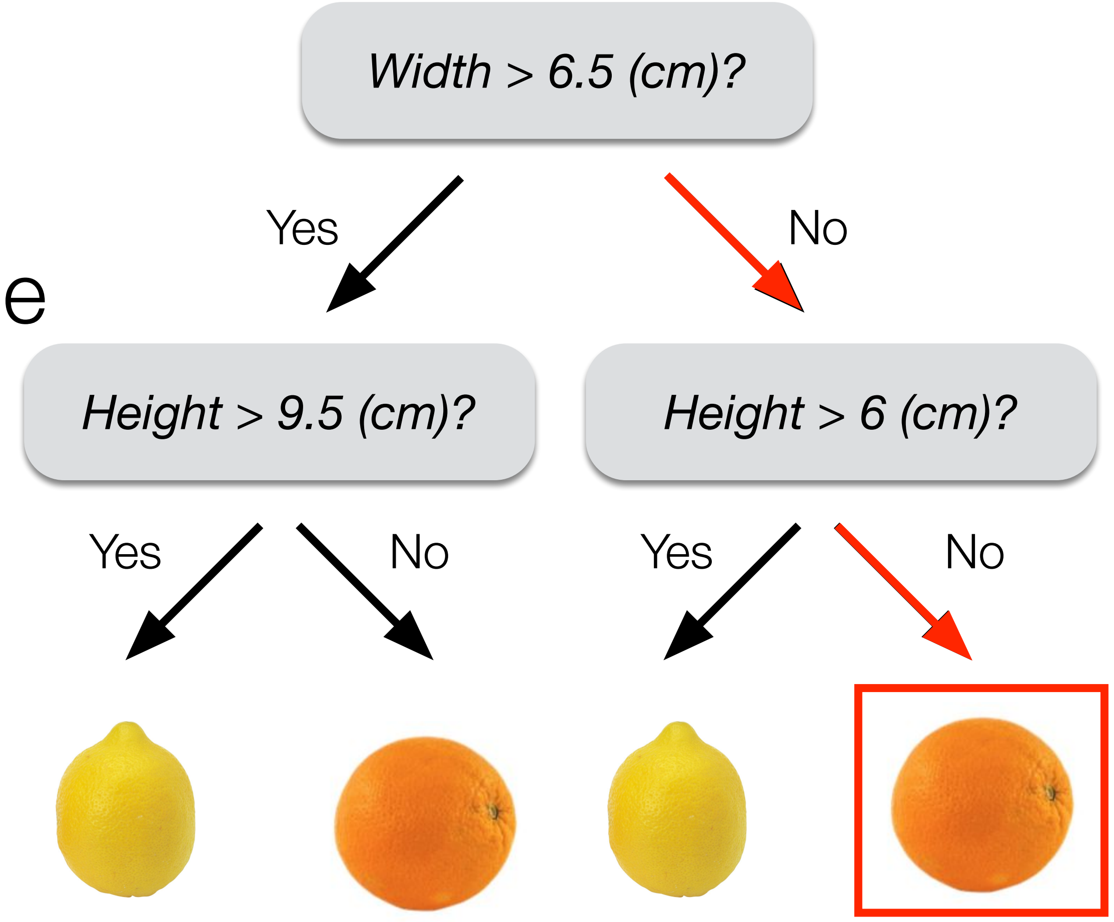
- At test time: ■ =  → "orange"





Decision Trees: ingredients

- Internal nodes
  - Related to test attributes
- Branching
  - It’s determined by attribute value
- Leaf nodes
  - Outputs / Class assignments



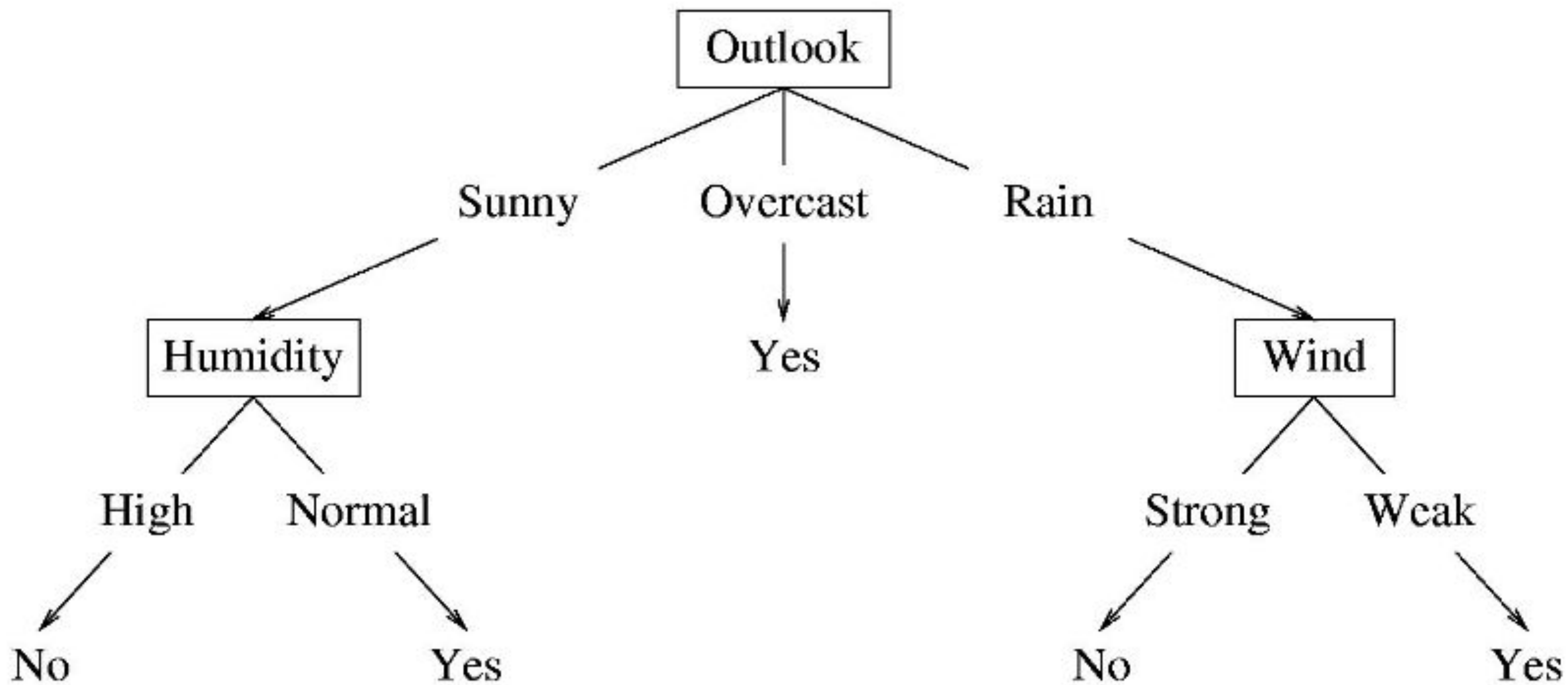
Decision Tree

$$\langle \boldsymbol{x}_i, y_i \rangle$$

Predictors				Response
Outlook	**Temperature**	**Humidity**	**Wind**	**Class**
Sunny	Hot	High	Weak	No
Sunny	Hot	High	Strong	No
Overcast	Hot	High	Weak	Yes
Rain	Mild	High	Weak	Yes
Rain	Cool	Normal	Weak	Yes
Rain	Cool	Normal	Strong	No
Overcast	Cool	Normal	Strong	Yes
Sunny	Mild	High	Weak	No
Sunny	Cool	Normal	Weak	Yes
Rain	Mild	Normal	Weak	Yes
Sunny	Mild	Normal	Strong	Yes
Overcast	Mild	High	Strong	Yes
Overcast	Hot	Normal	Weak	Yes
Rain	Mild	High	Strong	No

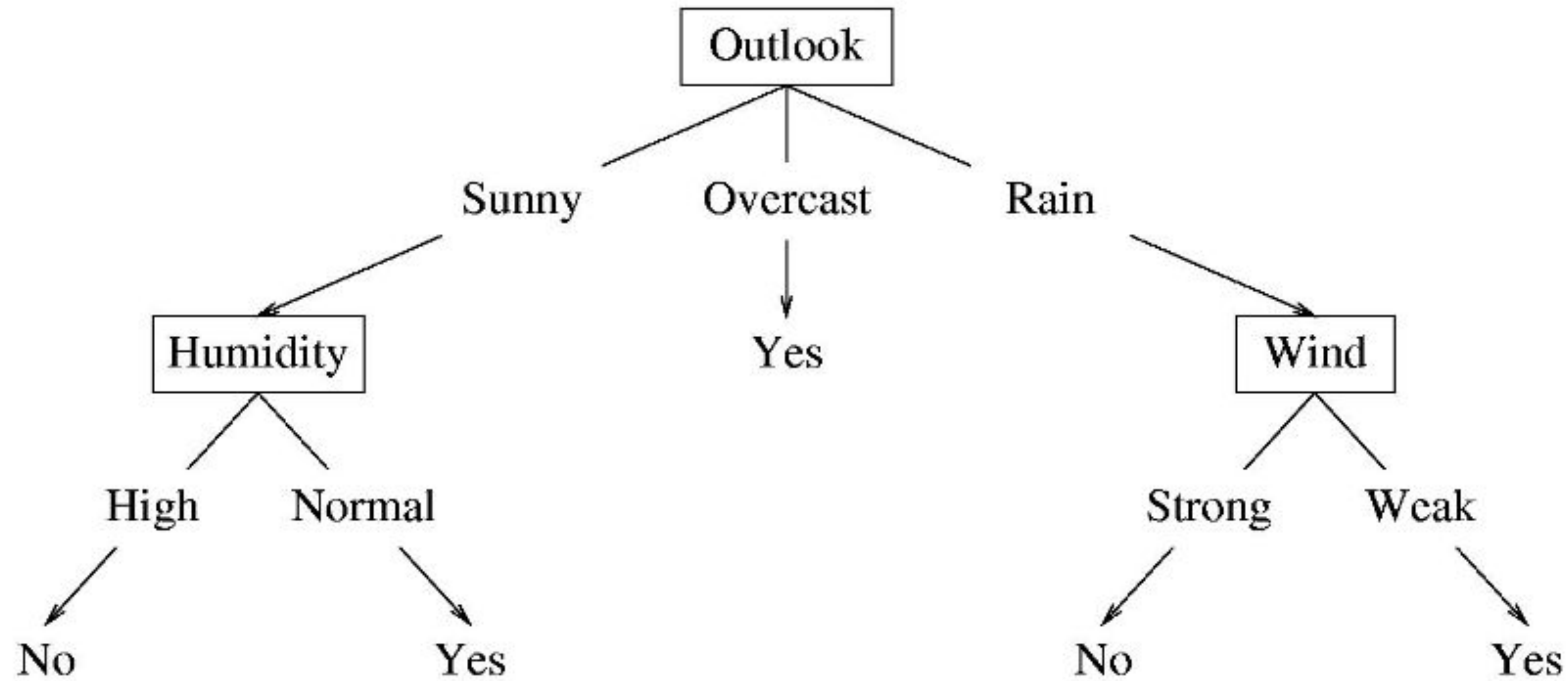
Decision Tree

- A possible decision tree can be represented as



Decision Tree

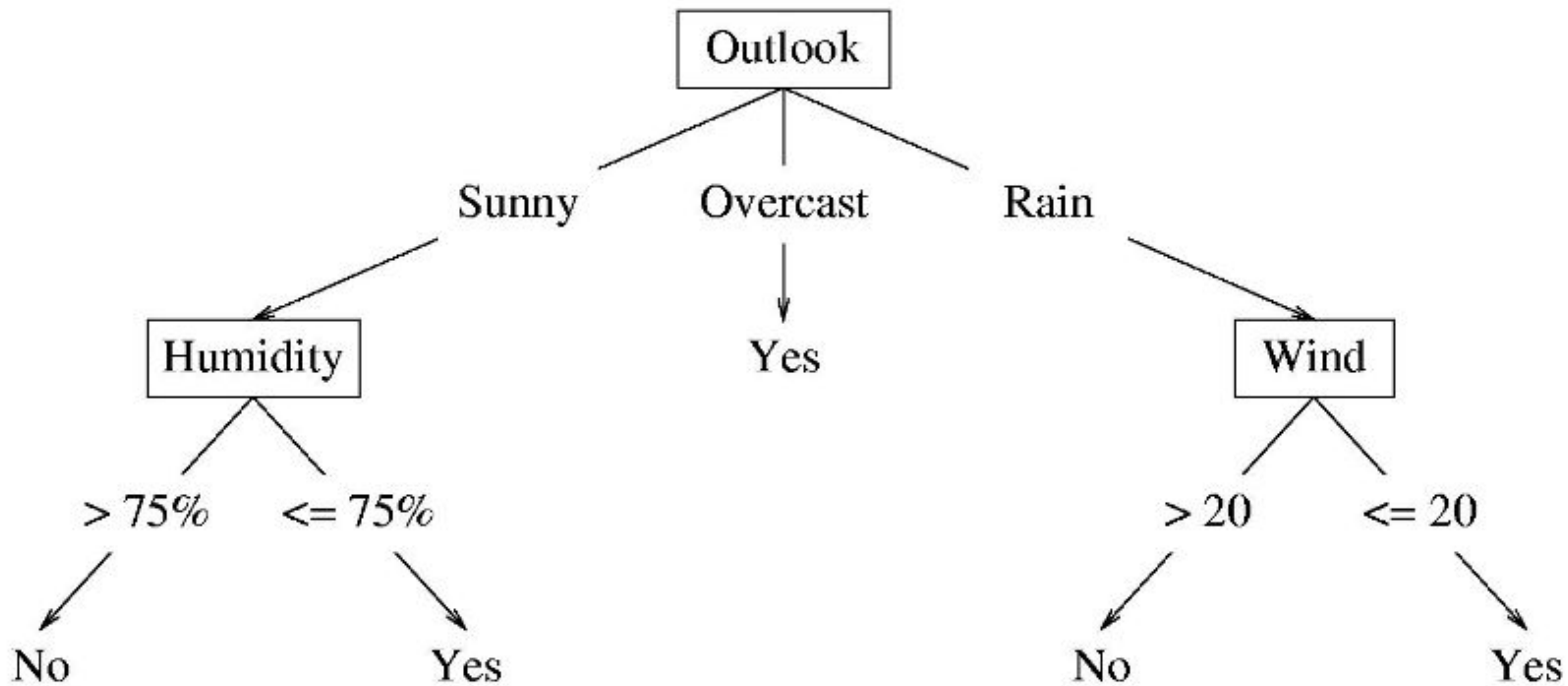
- A possible decision tree can be represented as



- What prediction would we make for
  - <outlook=sunny, temperature=hot, humidity=high, wind=weak> ?

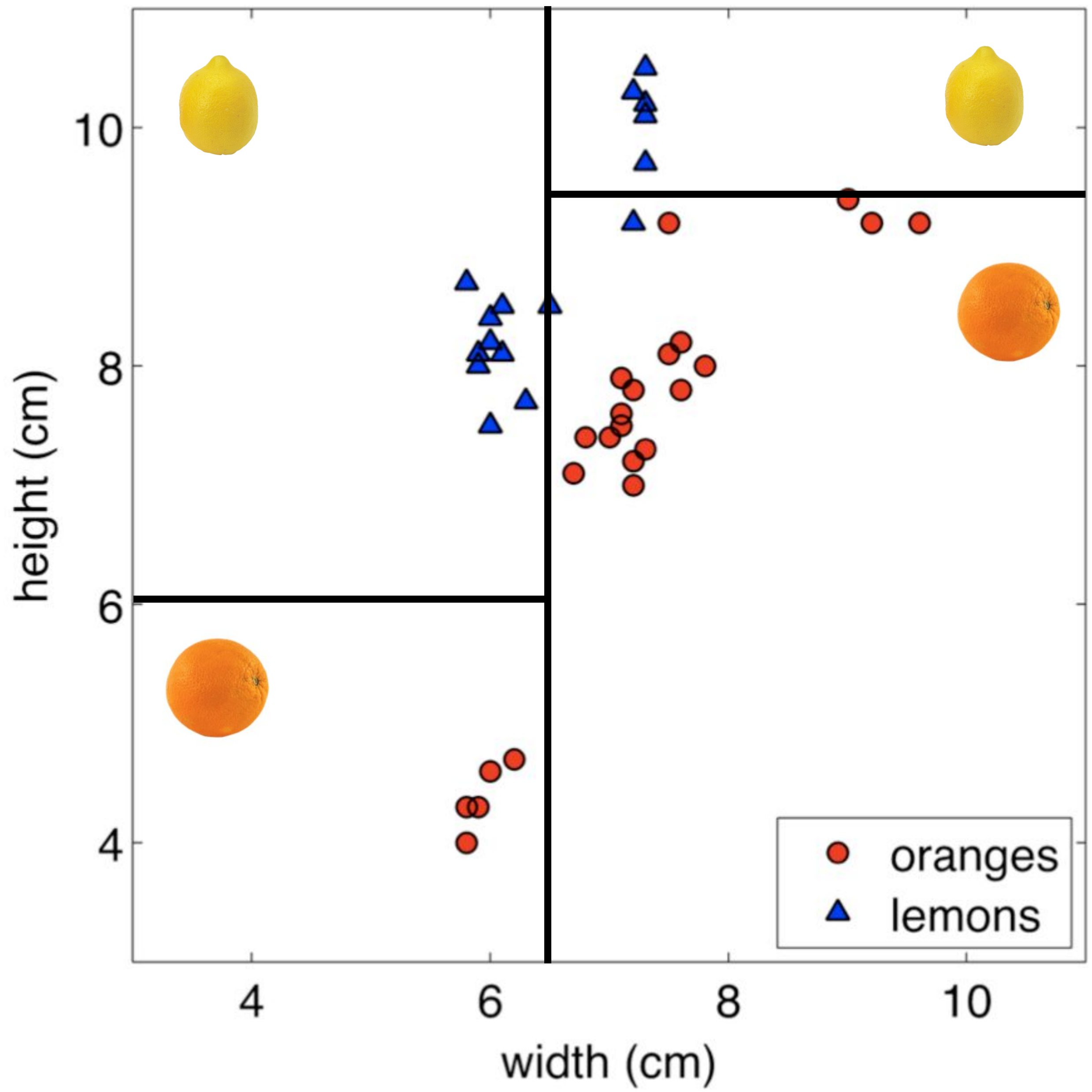
Decision Tree

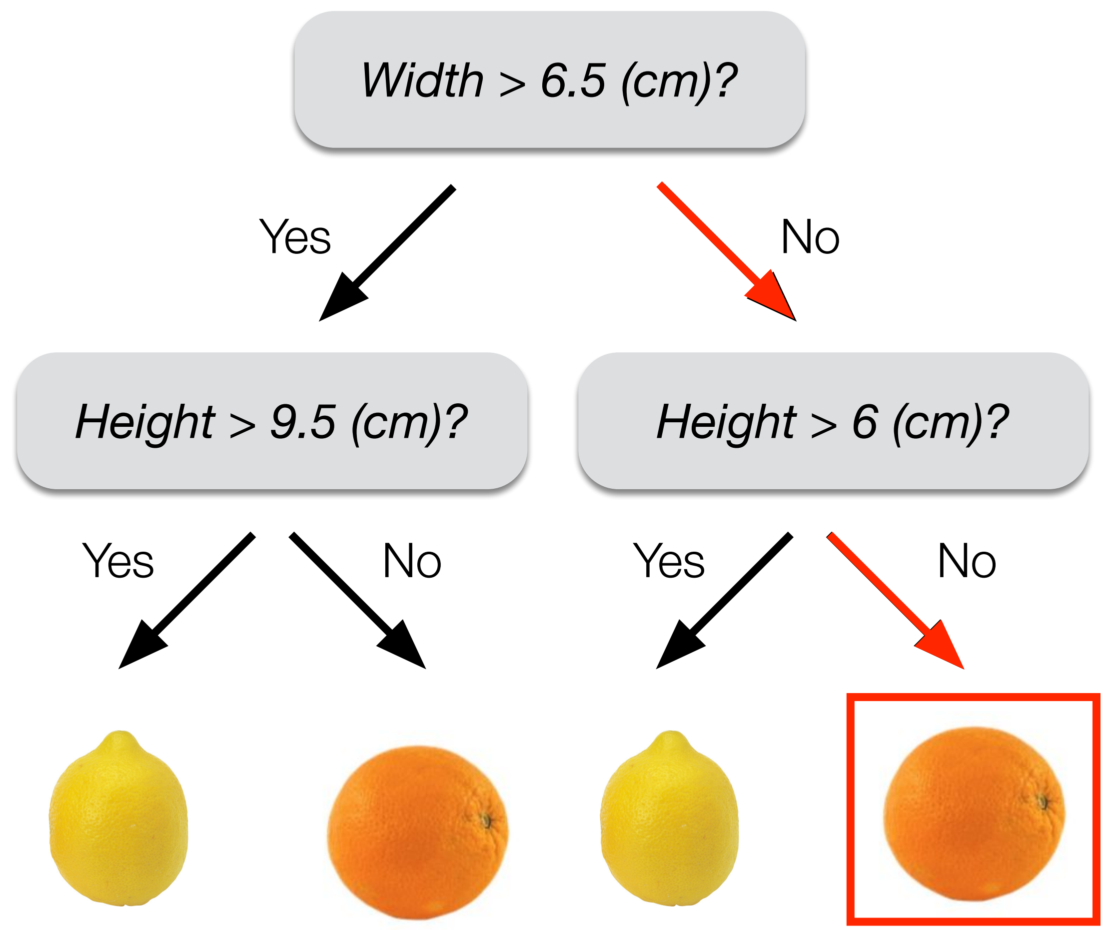
- We can define condition also for continuous variables
  - By using a threshold



Decision Boundary

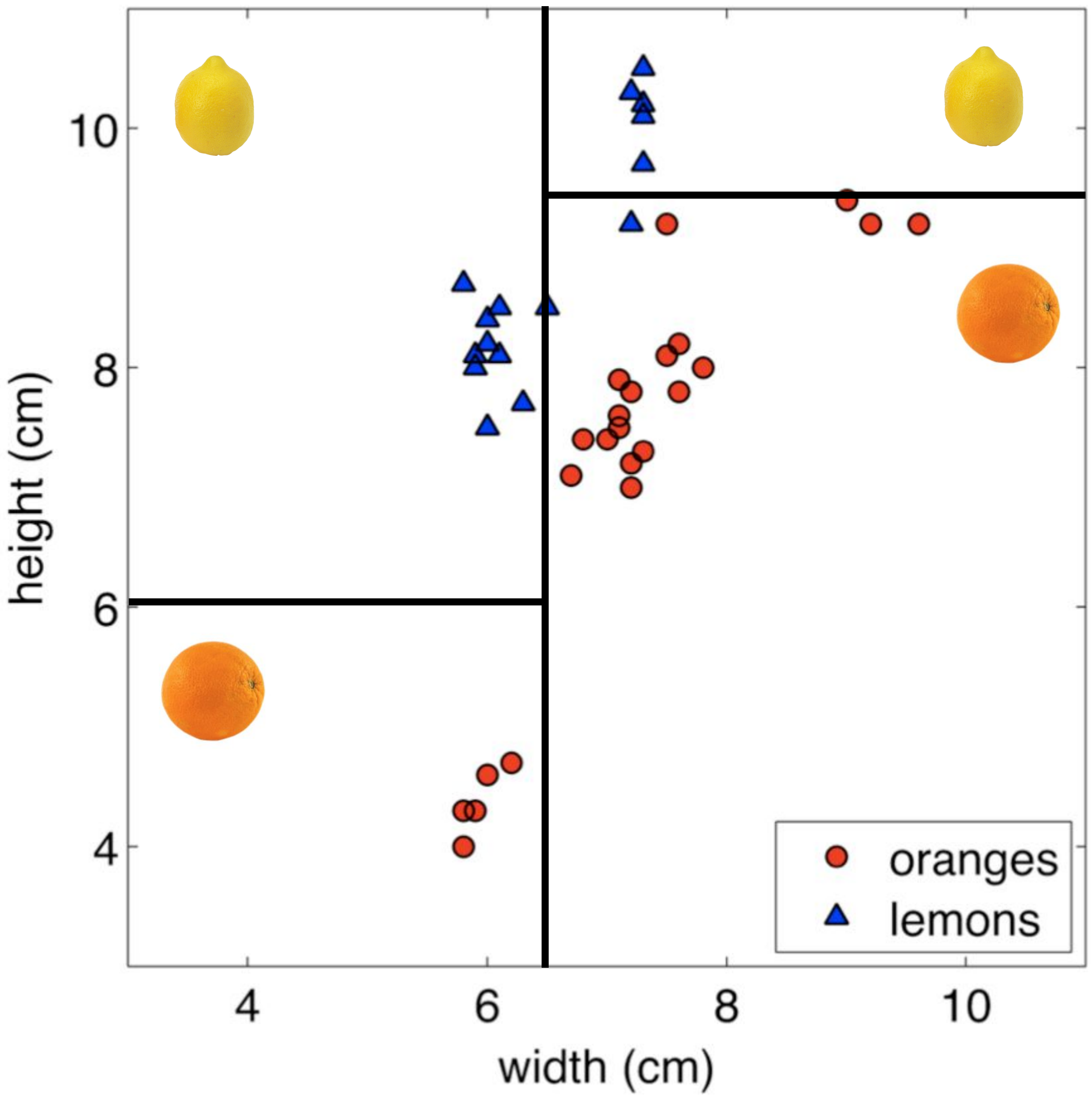
- Decision trees divide the feature space into axis-parallel (hyper-)rectangles
- Each rectangular region is labeled with one label

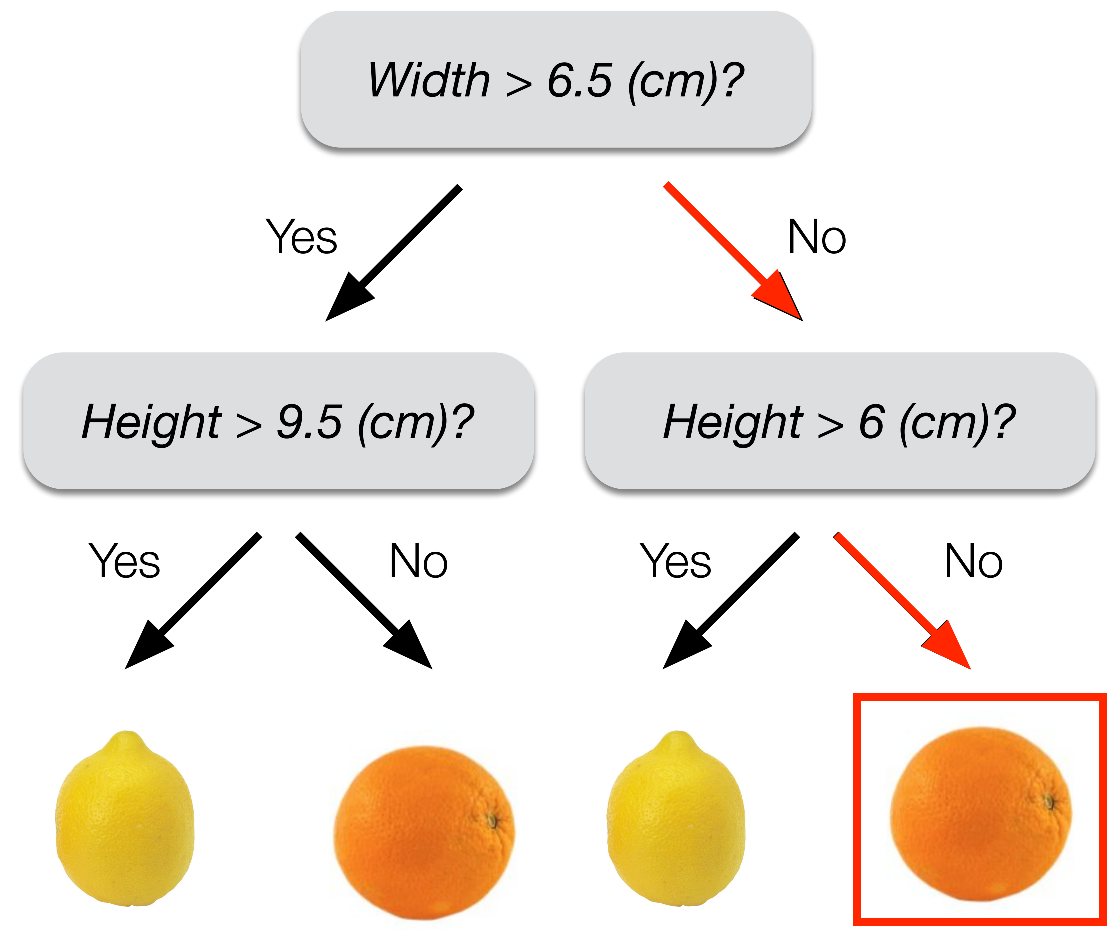




Decision Boundary

- Decision trees divide the feature space into axis-parallel (hyper-)rectangles
- Each rectangular region is labeled with one label





Human Interpretable!!

Hypothesis Space

- How many hypotheses?
- What type of functions can be learned?

Expressiveness

- Decision trees can represent any function of the input attribute
- Ingredients:
    - Nodes: Related to test attributes
    - Leaves: Classification output
    - Branches: determined by the attribute values (on node condition)
- We can represent **any** training data with enough nodes
- The higher the tree, the more expressive
    - Depth 1: 2 distinct separations
    - Depth 2: 4 distinct separations
    - Depth 3: 2^3 = 8

Learning a DT

- We utilize heuristics
  a. Not too small → risk of underfitting
  b. Not too big → risk of overfitting
- We can use the following algorithm
  a. We start from an empty tree
  b. We split the data accordingly to the **best feature**
  c. Check the two new sets:
    - If a set contains only one label → **leaf node**
    - If a set contantains multiple labels → restart from (b)
      - <u>Recursion</u>

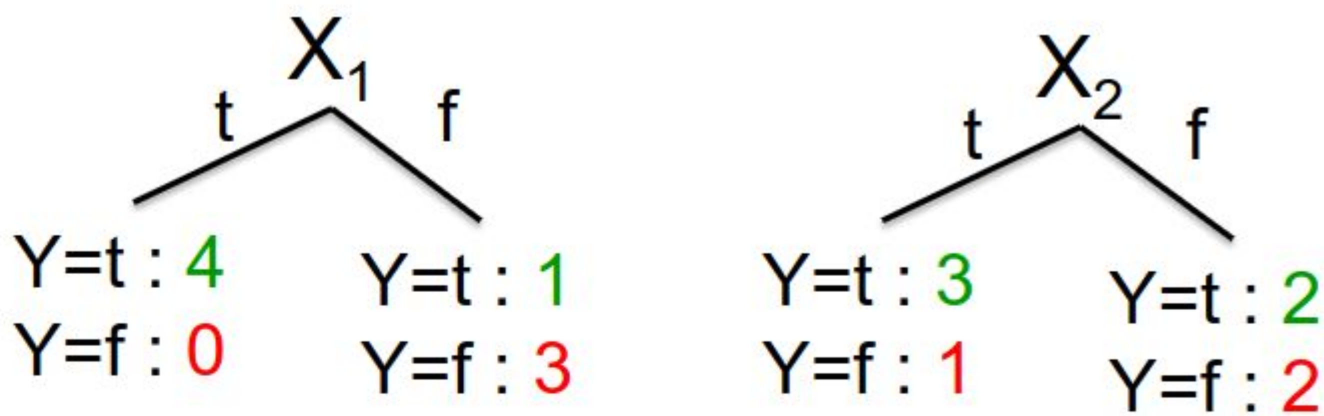
Learning a DT

- What is a good feature?
  a. Would you prefer to split with x1 or x2?

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

Learning a DT

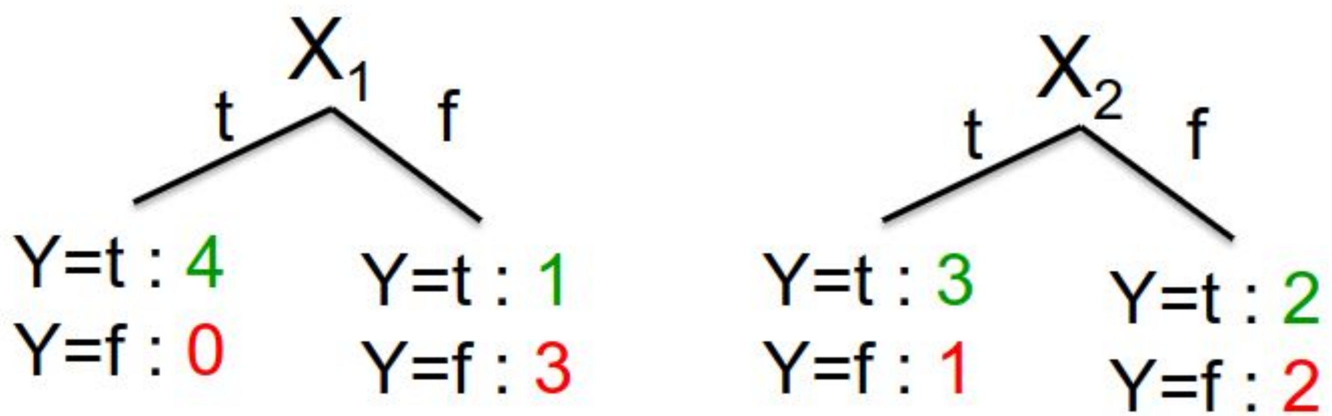
- What is a good feature?
  a. Would you prefer to split with x1 or x2?



X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

Learning a DT

- What is a good feature?
  a. Would you prefer to split with x1 or x2?



X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

Idea: we look at the leaves node and we measure the <u>uncertainty</u>

Learning a DT

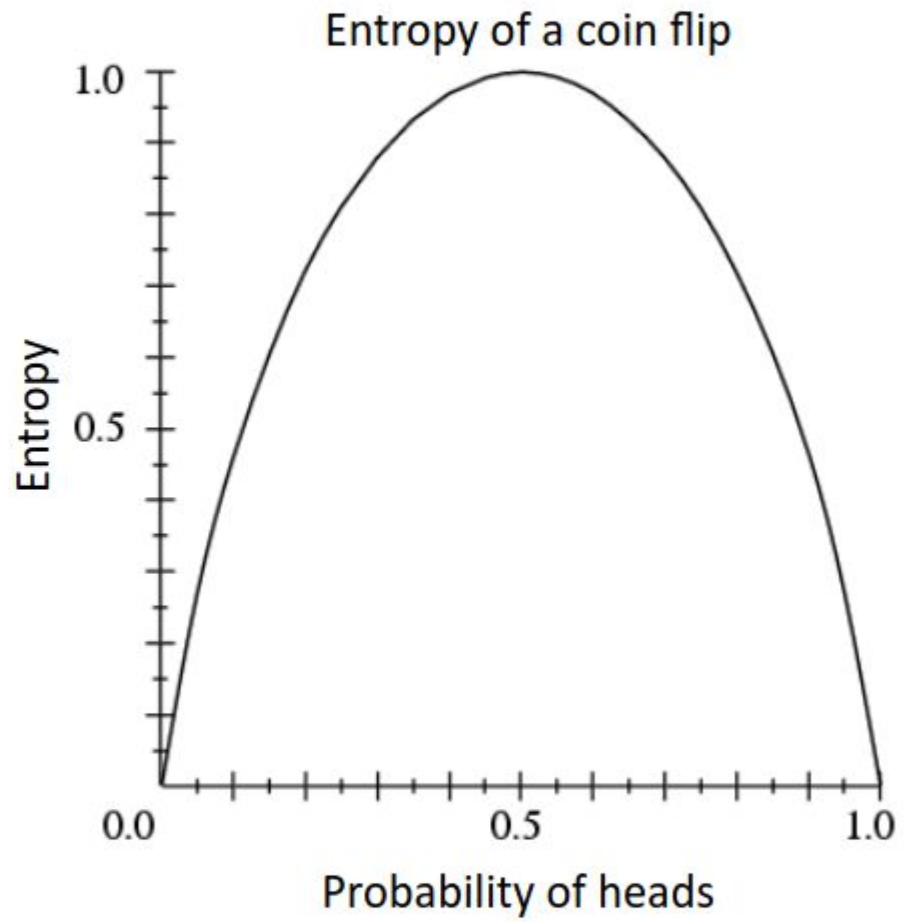
- What is a good feature?
- A good split if we are more certain about classification
  a. They are all true or false: GOOD
  b. They are uniformly distributed: BAD
  c. What about distribution in between?

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F
F	T	F
F	F	F

Entropy

$$H(Y) = -\sum_{i=1}^{k} P(Y = y_i) \log_2 P(Y = y_i)$$

- More entropy → more uncertainty
- High entropy:
  - After the split, the data samples are uniformly distributed on y
  - “Not useful for a prediction”
- Low entropy:
  - After the split, the distribution of the ground truth is not uniform
  - Value samples are more predictable



Learning a DT

- We can use the entropy to select the most suitable attribute
  a. **Information gain** → we pick the attribute that makes the entropy decrease the most

$$IG(X) = H(Y) - H(Y \mid X)$$

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F

Learning a DT

$$IG(X) = H(Y) - H(Y \mid X)$$

$$H(Y) = -\sum_{i=1}^{k} P(Y = y_i) \log_2 P(Y = y_i)$$

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F

P(Y=t) = 5/6

P(Y=f) = 1/6

$$H(Y) = -\ 5/6 \log_2 5/6 - 1/6 \log_2 1/6$$
$$= 0.65$$

Learning a DT

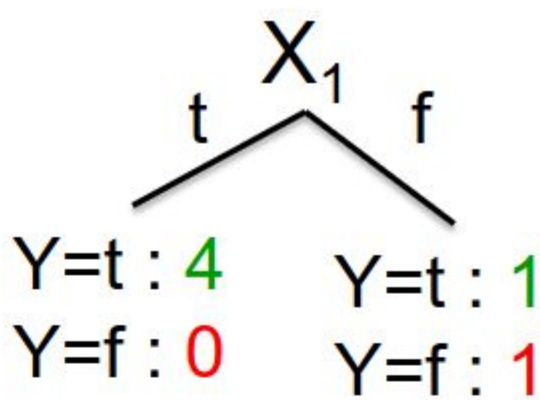
$$IG(X) = H(Y) - H(Y \mid X)$$

$$H(Y \mid X) = -\sum_{j=1}^{v} P(X = x_j) \sum_{i=1}^{k} P(Y = y_i \mid X = x_j) \log_2 P(Y = y_i \mid X = x_j)$$

Example:

$P(X_1=t) = 4/6$

$P(X_1=f) = 2/6$



X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F

$H(Y|X_1) = - 4/6\ (1 \log_2 1 + 0 \log_2 0)$

$\quad - 2/6\ (1/2 \log_2 1/2 + 1/2 \log_2 1/2)$

$= 2/6$

Learning a DT

$$IG(X) = H(Y) - H(Y \mid X)$$

In our running example:

$IG(X_1) = H(Y) – H(Y|X_1)$
$= 0.65 – 0.33$

$IG(X_1) > 0 \rightarrow$ we prefer the split!

X_1	X_2	Y
T	T	T
T	F	T
T	T	T
T	F	T
F	T	T
F	F	F

Learning a DT

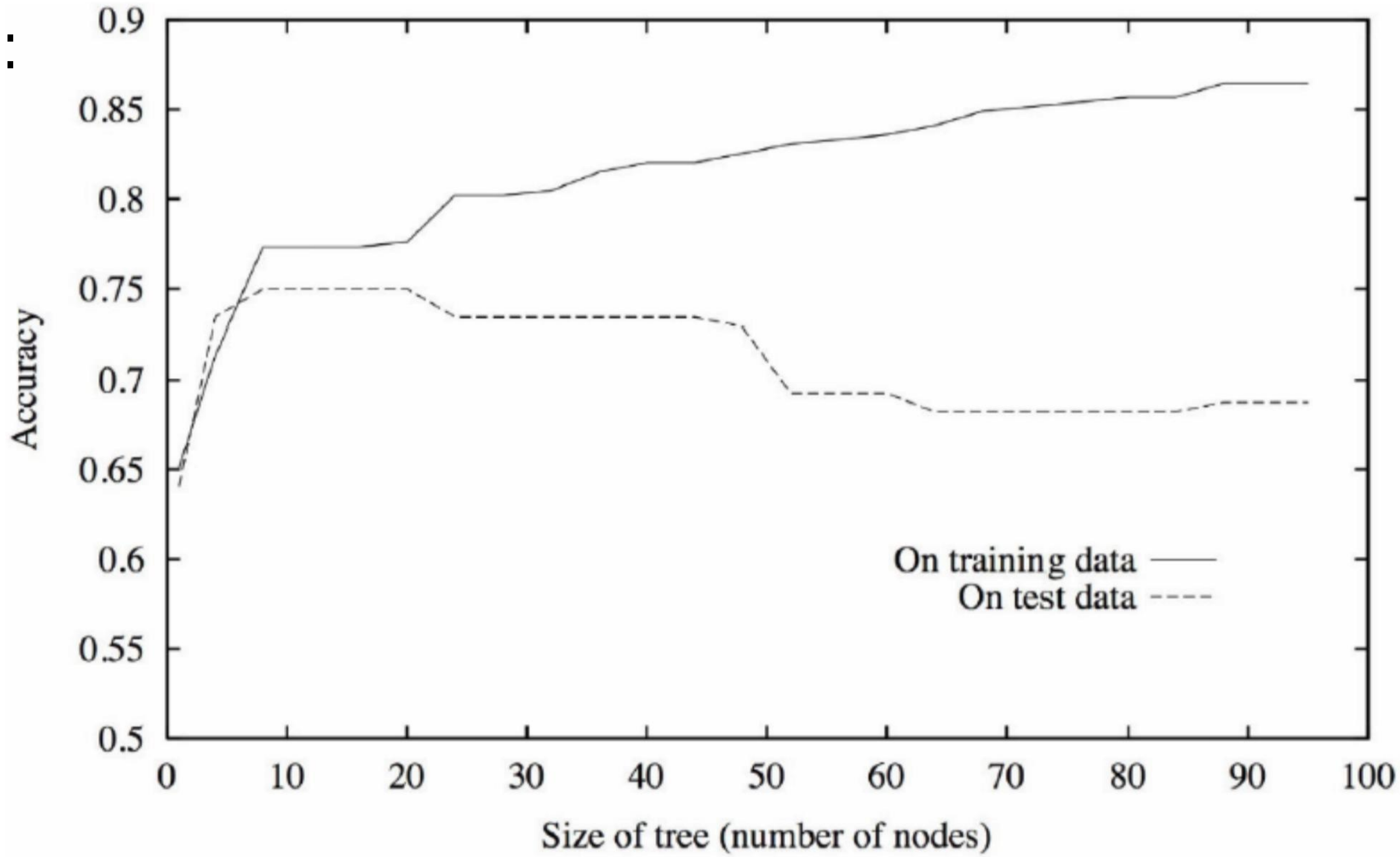
- We utilize heuristics
  a. Not too small → risk of underfitting
  b. Not too big → risk of overfitting
- We can use the following algorithm
  a. We start from an empty tree
  b. We split the data accordingly to the **best feature**
     - Selection Strategy: use the Information Gain

$$\arg\max_i IG(X_i) = \arg\max_i H(Y) - H(Y \mid X_i)$$

  c. Check the two new sets:
     - If a set contains only one label → **leaf node**
     - If a set contantains multiple labels → restart from (b)
       - Recursion

Decision Trees: issues, challenges

- Towards preventing overfitting (through pruning)
  - E.g.:

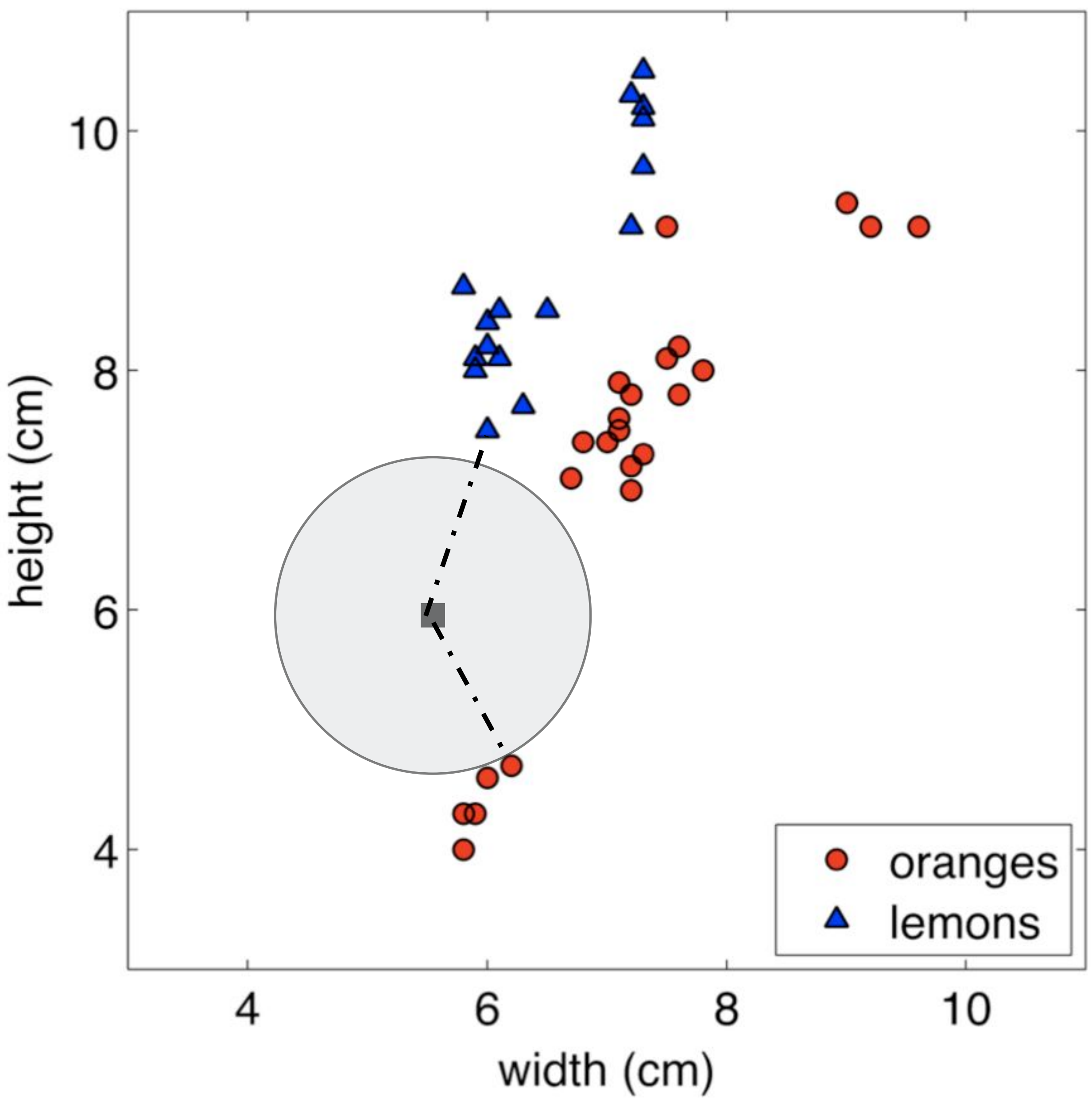


  - <u>Note</u>: in `sklearn` you can set a threshold on the number of examples in the leaf nodes or on the depth of the tree

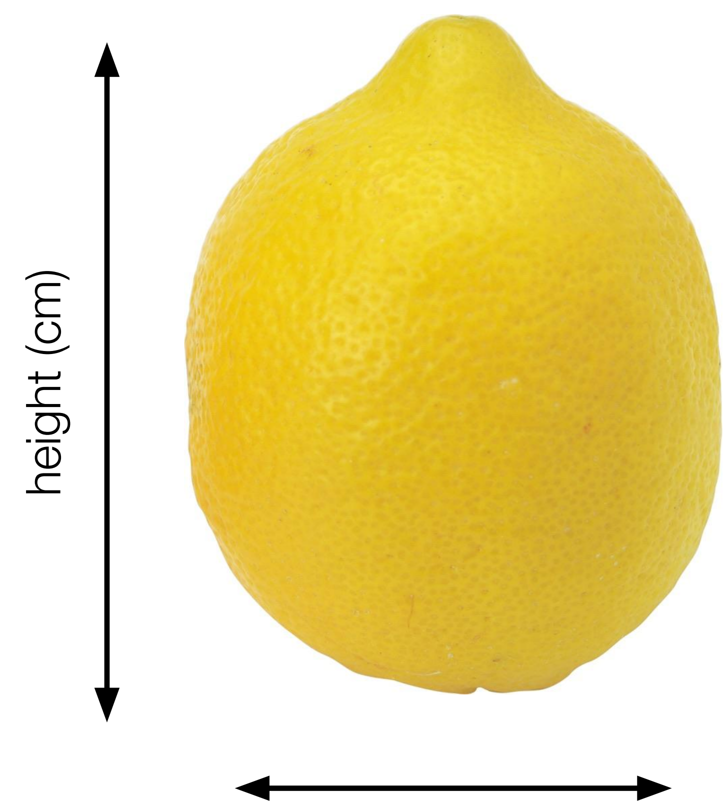
Decision Trees: issues, challenges

- Towards preventing overfitting
- Stop growing the tree earlier, before it reaches the point where it perfectly classifies the training data
  - Note: in `sklearn` you can set a threshold on the number of examples in the leaf nodes or on the depth of the tree
- overfit the data and then prune the tree afterwards
  - pruning: transform a subtree into a leaf node labelled with the majority label
  - Use a separate set of examples to evaluate the utility of post-pruning nodes from the tree

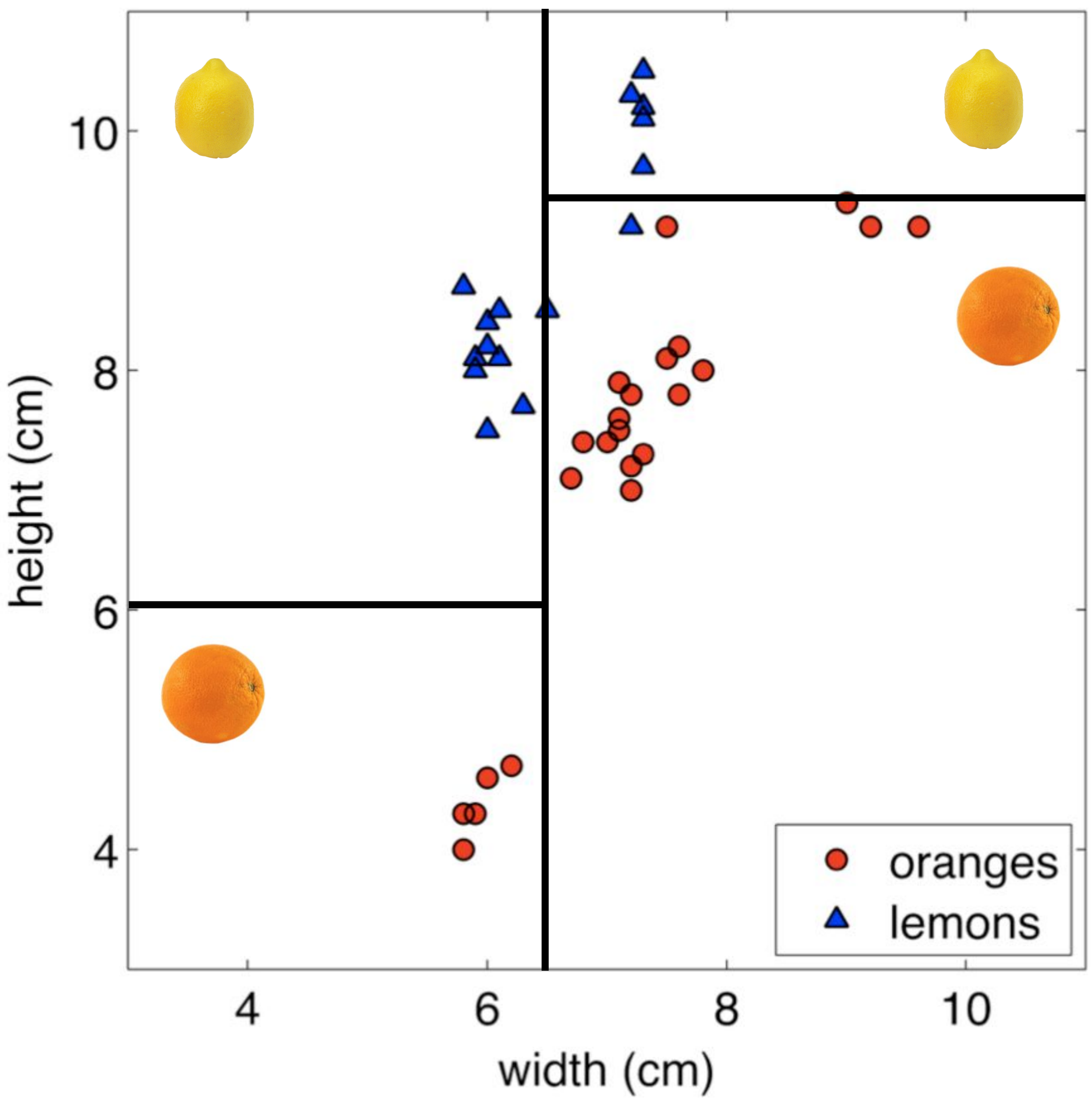
KNN

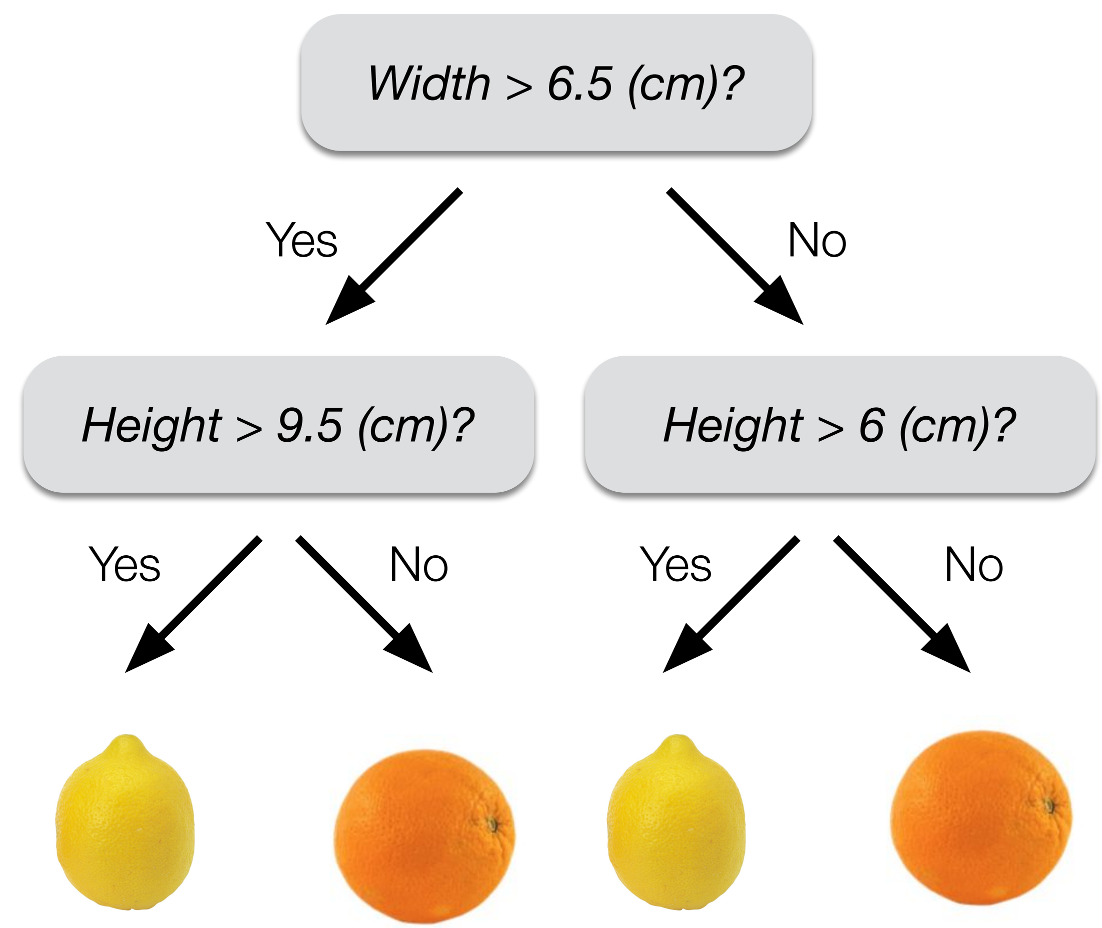


Binary classifier based on two simple features:



Decision Trees





Restriction Bias

- How we Limit the Hypothesis Space

- Representing functions that split the feature space into axis-aligned rectangular regions

- Only modeling stepwise functions—it cannot inherently represent smooth or continuous decision boundaries unless the tree is extremely deep.

Preference Bias

- How we Order the Hypothesis Space
    - Changing the order of “nodes” impact the set of possible representable functions

- Prioritize features with high information gain (e.g., entropy or Gini impurity reduction). This means it prefers splits that quickly separate the data.

- Favor simpler trees (via pruning techniques) to prevent overfitting