Decision Trees

Nicholas Ruozzi
University of Texas at Dallas

Based on the slides of Vibhav Gogate and David Sontag

Supervised Learning

- Input: labeled training data
  - i.e., data plus desired output
- Assumption: there exists a function f that maps data items x to their correct labels
- Goal: construct an approximation to f

Today

- We’ve been focusing on linear separators
  - Relatively easy to learn (using standard techniques)
  - Easy to picture, but not clear if data will be separable
- Next two lectures we’ll focus on other hypothesis spaces
  - Decision trees
  - Nearest neighbor classification

Application: Medical Diagnosis

- Suppose that you go to your doctor with flu-like symptoms
  - How does your doctor determine if you have a flu that requires medical attention?

Application: Medical Diagnosis

- Suppose that you go to your doctor with flu-like symptoms
  - How does your doctor determine if you have a flu that requires medical attention?
  - Check a list of symptoms:
    - Do you have a fever over 100.4 degrees Fahrenheit?
    - Do you have a sore throat or a stuffy nose?
    - Do you have a dry cough?

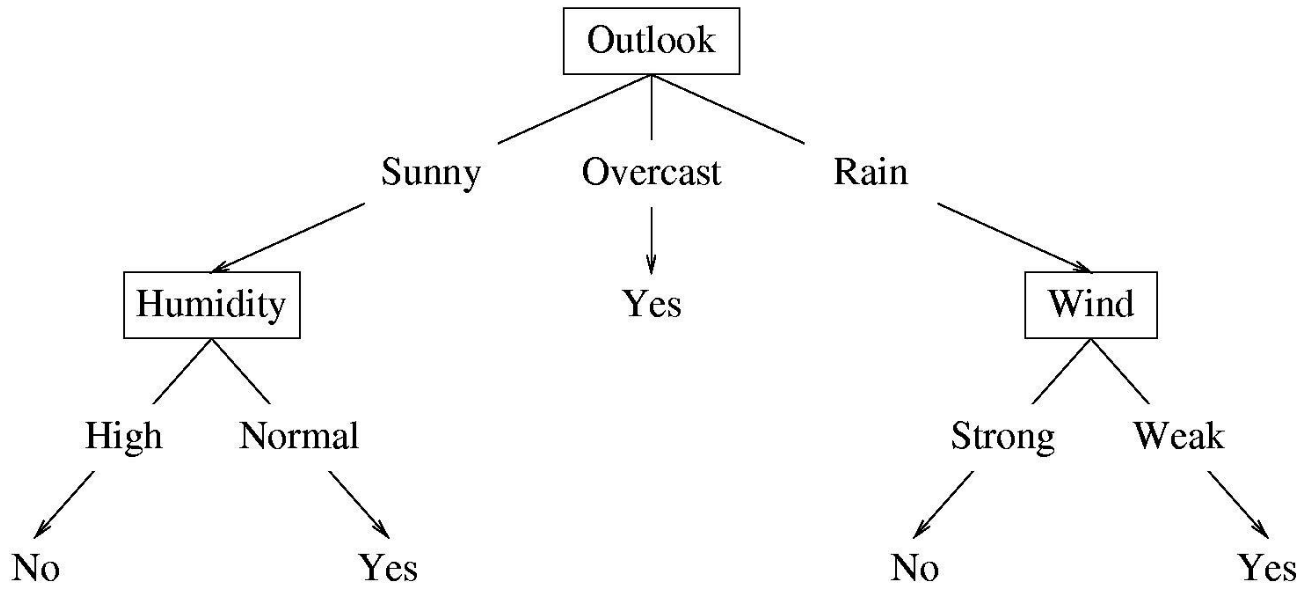
Application: Medical Diagnosis

- Just having some symptoms is not enough, you should also not have symptoms that are not consistent with the flu
- For example,
  - If you have a fever over 100.4 degrees Fahrenheit?
  - And you have a sore throat or a stuffy nose?
  - You probably do not have the flu (most likely just a cold)

Application: Medical Diagnosis

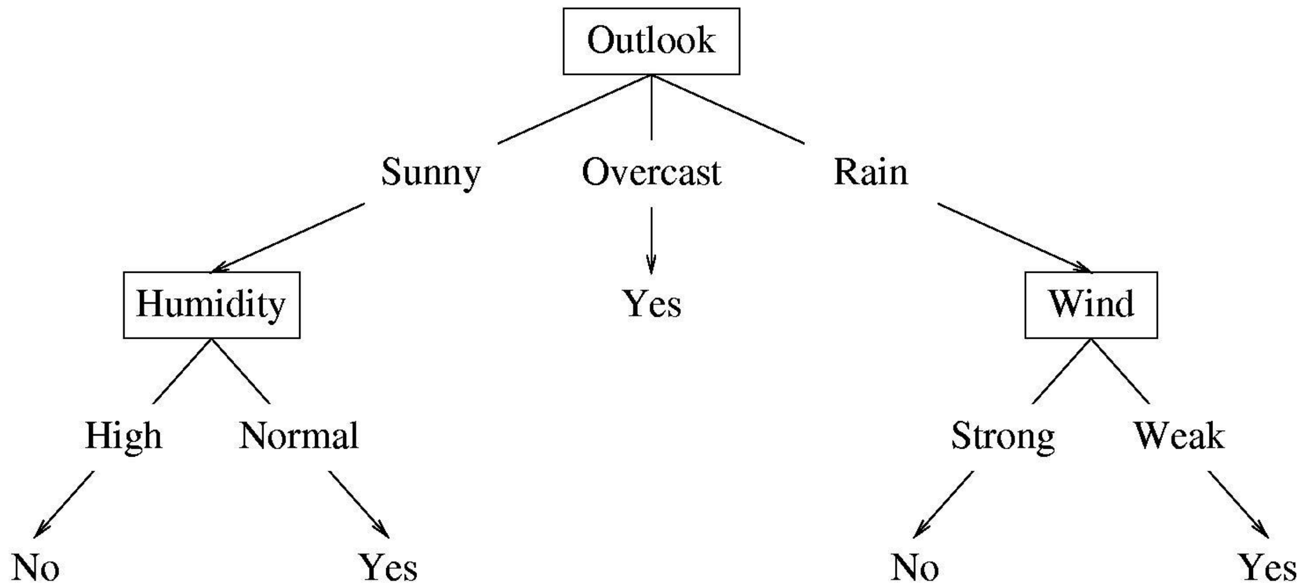
- In other words, your doctor will perform a series of tests and ask a series of questions in order to determine the likelihood of you having a severe case of the flu
- This is a method of coming to a diagnosis (i.e., a classification of your condition)
- We can view this decision making process as a tree

Decision Trees



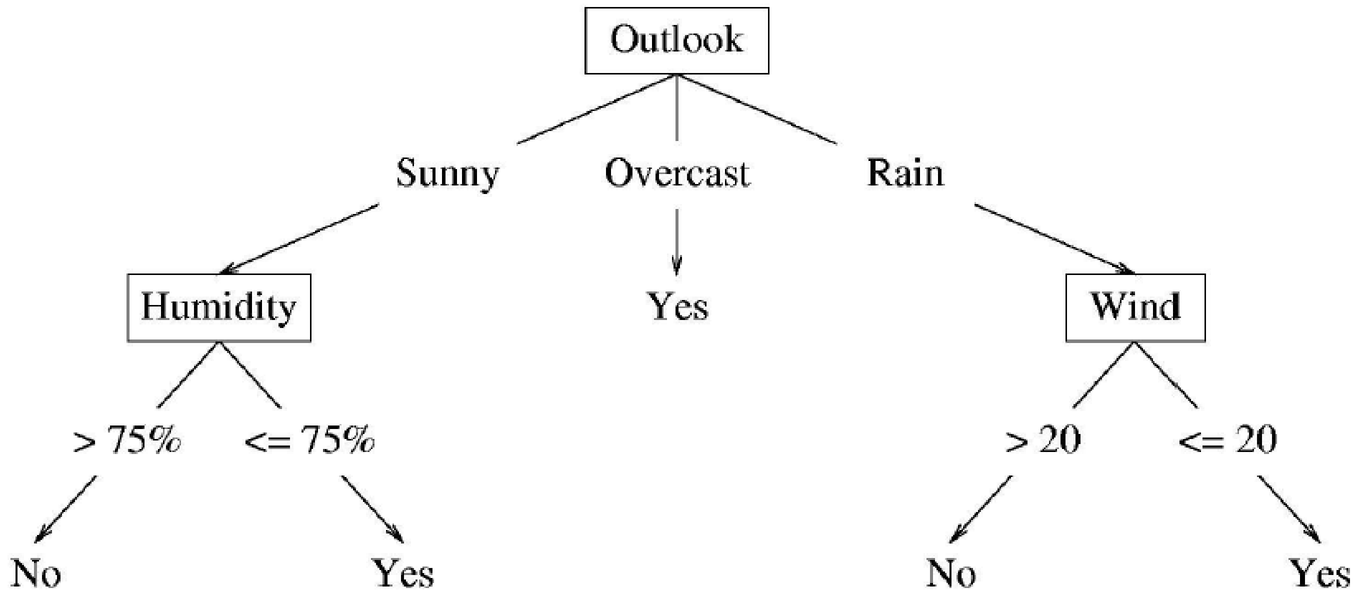
- A tree in which each internal (non-leaf) node tests the value of a particular feature
- Each leaf node specifies a class label (in this case whether or not you should play tennis)

Decision Trees



- Features: (Outlook, Humidity, Wind)
- Classification is performed root to leaf
  - The feature vector (Sunny, Normal, Strong) would be classified as a yes instance

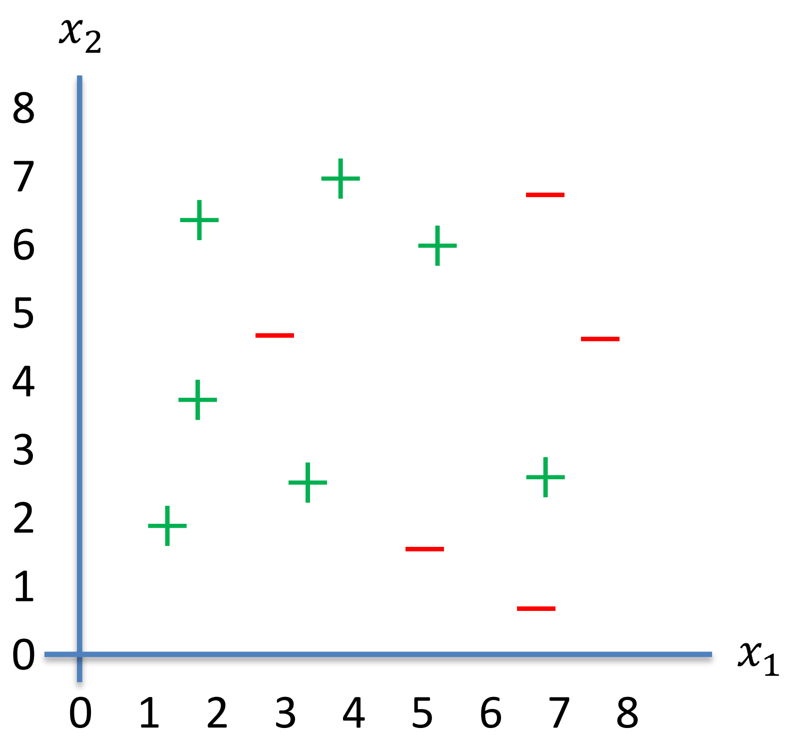
Decision Trees

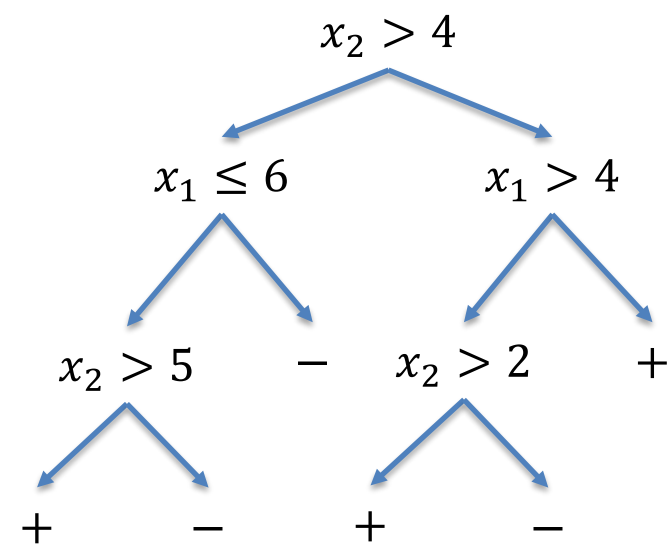


- Can have continuous features too
  - Internal nodes for continuous features correspond to thresholds

Decision Trees

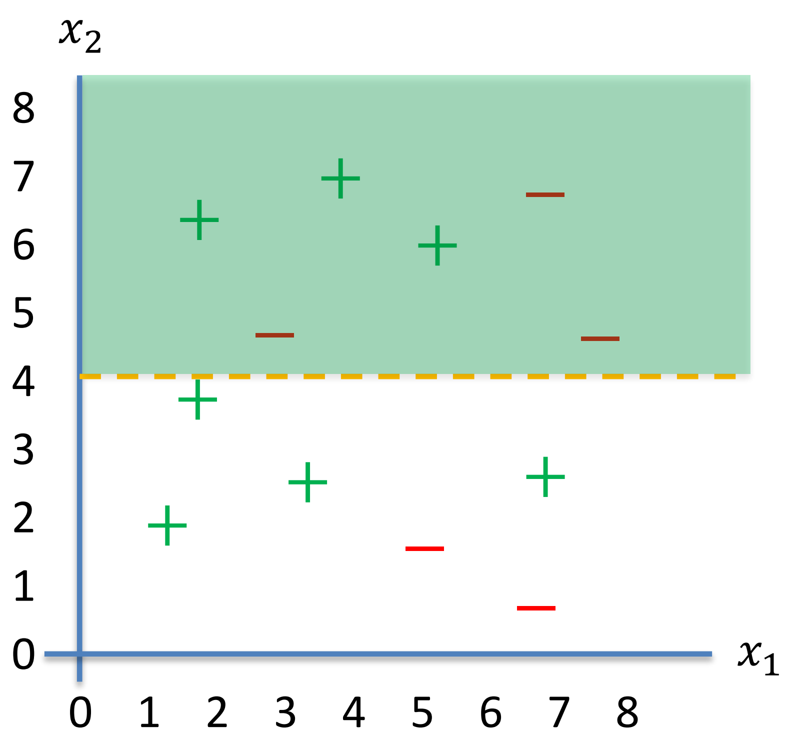
- Decision trees divide the feature space into axis parallel rectangles

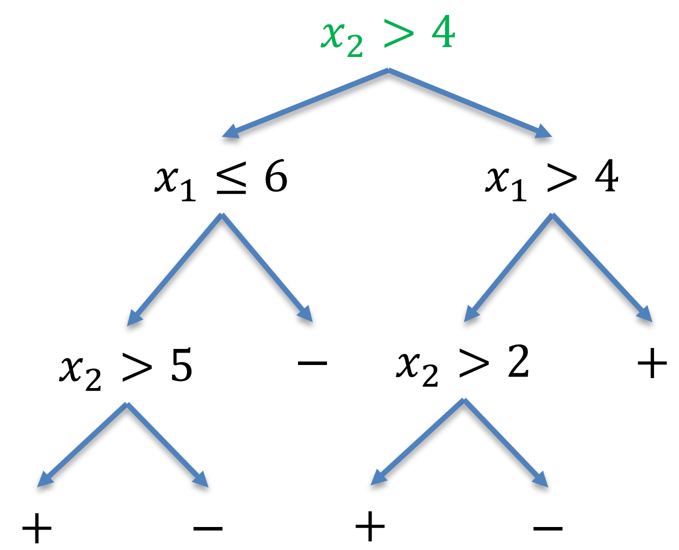




Decision Trees

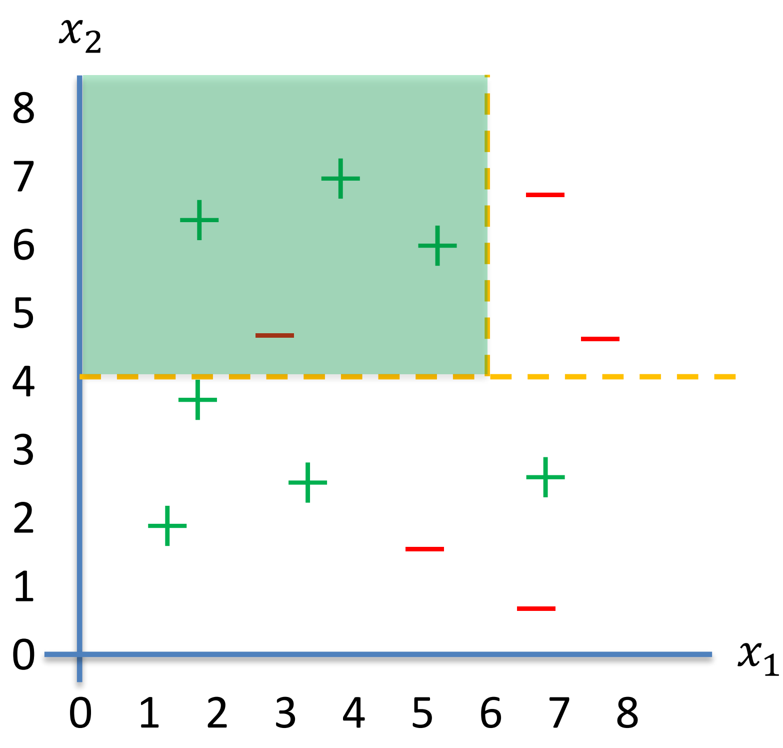
- Decision trees divide the feature space into axis parallel rectangles

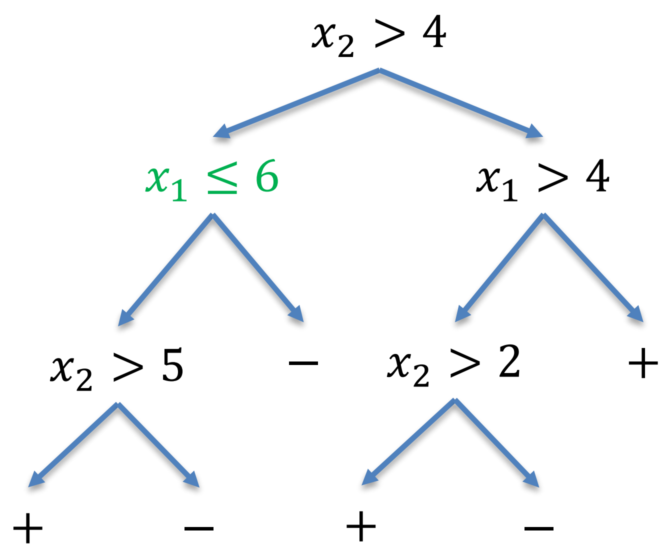




Decision Trees

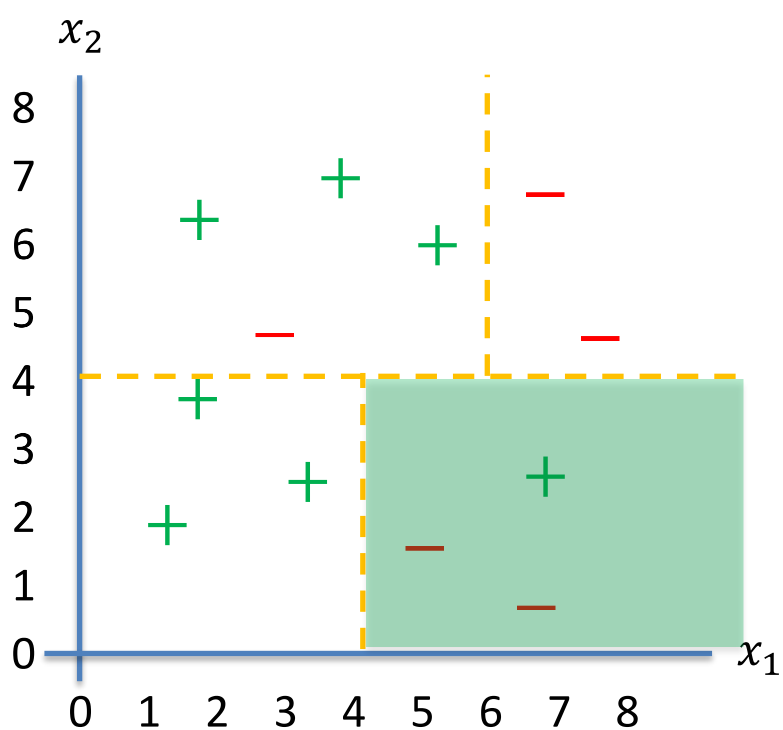
- Decision trees divide the feature space into axis parallel rectangles

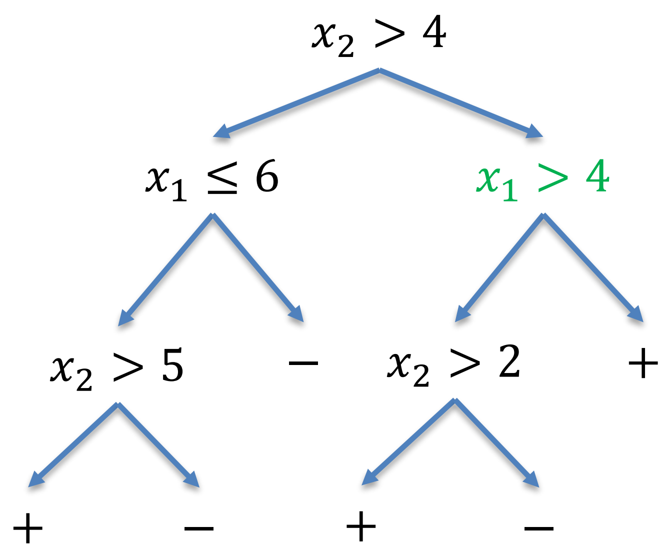




Decision Trees

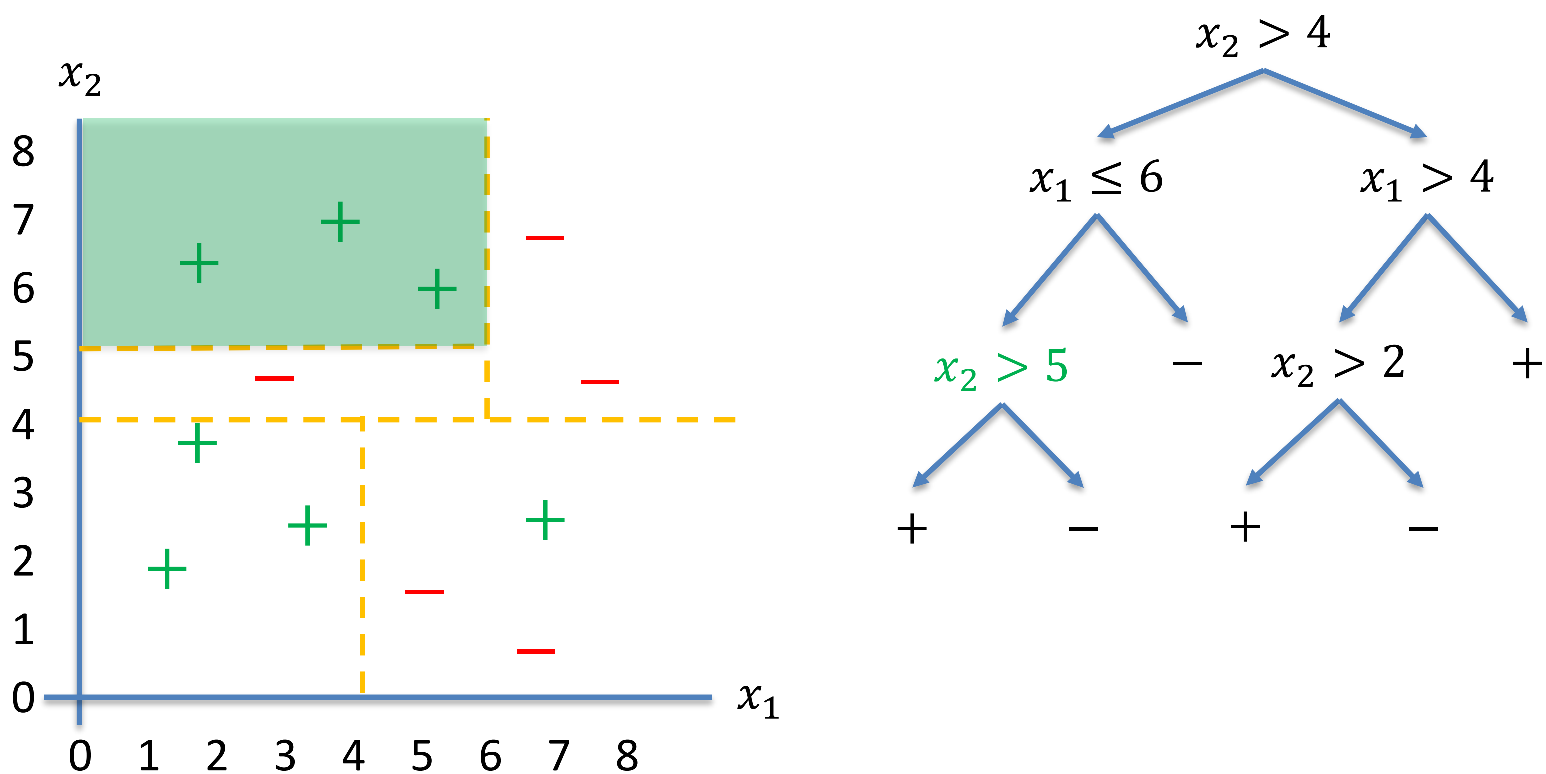
- Decision trees divide the feature space into axis parallel rectangles





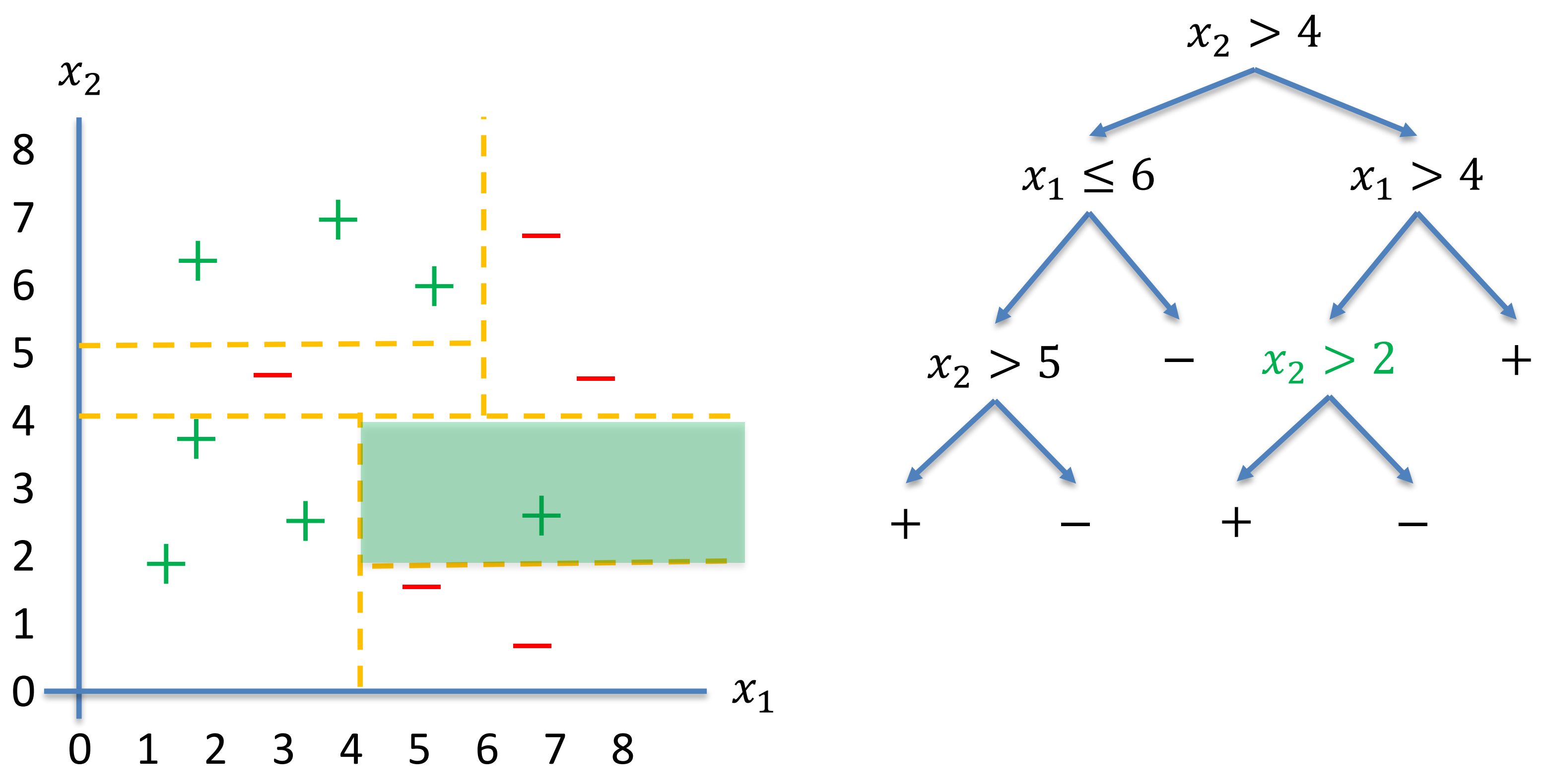
Decision Trees

- Decision trees divide the feature space into axis parallel rectangles



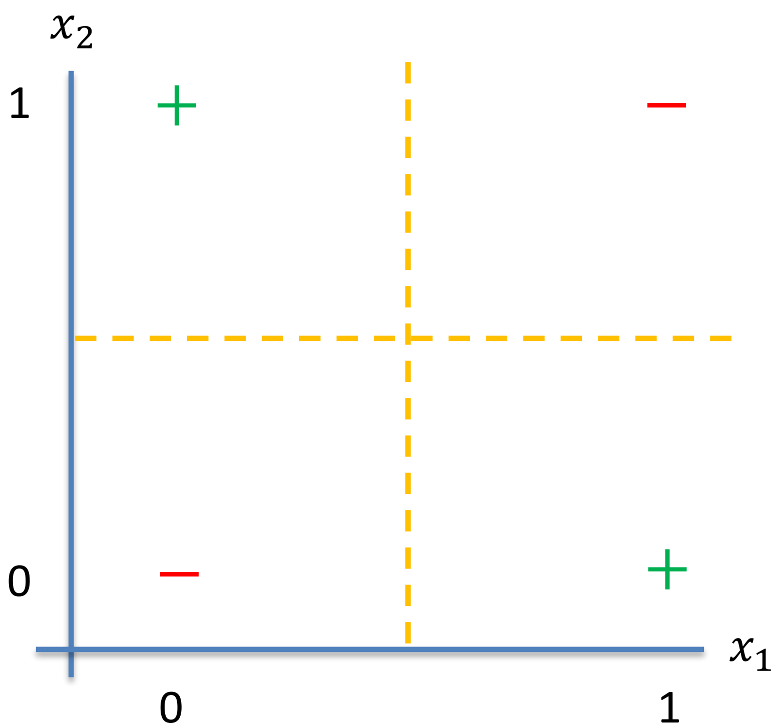
Decision Trees

- Decision trees divide the feature space into axis parallel rectangles



Decision Trees

- Worst case decision tree may require exponentially (in the dimension of the data) many nodes



Decision Tree Learning

- Basic decision tree building algorithm:
  - Pick some feature/attribute
  - Partition the data based on the value of this attribute
  - Recurse over each new partition

Decision Tree Learning

- Basic decision tree building algorithm:
  - Pick some feature/attribute (how to pick the “best”?)
  - Partition the data based on the value of this attribute
  - Recurse over each new partition (when to stop?)

We’ll focus on the discrete case first (i.e., each feature takes a value in some finite set)

Decision Trees

- What functions can be represented by decision trees?

- Are decision trees unique?

Decision Trees

- What functions can be represented by decision trees?
  - Every function of +/- can be represented by a sufficiently complicated decision tree
- Are decision trees unique?
  - No, many different decision trees are possible for the same set of labels

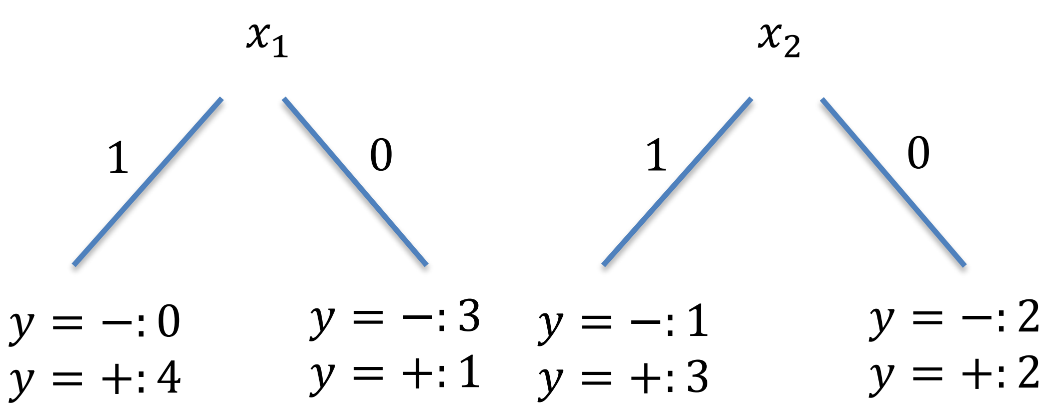
Choosing the Best Attribute

- Because the complexity of storage and classification increases with the size of the tree, should prefer smaller trees
  - Simplest models that explain the data are usually preferred over more complicated ones
  - Finding the smallest tree is an NP-hard problem
  - Instead, use a greedy heuristic based approach to pick the best attribute at each stage

Choosing the Best Attribute

$$x_1, x_2 \in \{0,1\}$$

Which attribute should you split on?

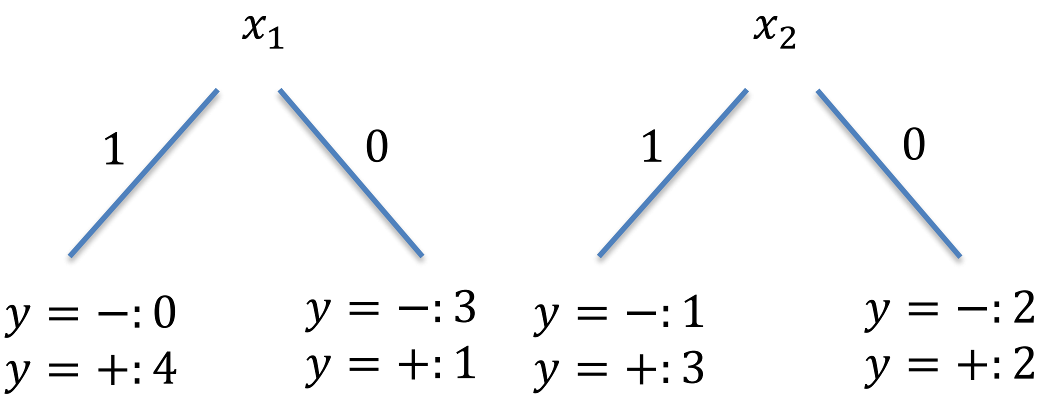


x_1	x_2	y
1	1	+
1	0	+
1	1	+
1	0	+
0	1	+
0	0	−
0	1	−
0	0	−

Choosing the Best Attribute

$$x_1, x_2 \in \{0,1\}$$

Which attribute should you split on?



x_1	x_2	y
1	1	+
1	0	+
1	1	+
1	0	+
0	1	+
0	0	−
0	1	−
0	0	−

Can think of these counts as probability distributions over the labels: if $x = 1$, the probability that $y = +$ is equal to 1

Choosing the Best Attribute

- The selected attribute is a good split if we are more "certain" about the classification after the split
  - If each partition with respect to the chosen attribute has a distinct class label, we are completely certain about the classification after partitioning
  - If the class labels are evenly divided between the partitions, the split isn't very good (we are very uncertain about the label for each partition)
  - What about other situations? How do you measure the uncertainty of a random process?

Discrete Probability

- **Sample space** specifies the set of possible outcomes
  - For example, $\Omega = \{\mathrm{H}, \mathrm{T}\}$ would be the set of possible outcomes of a coin flip
- Each element $\omega \in \Omega$ is associated with a number $\mathrm{p}(\omega) \in [0,1]$ called a **probability**

$$\sum_{\omega \in \Omega} p(\omega) = 1$$

  - For example, a biased coin might have $p(H) = .6$ and $p(T) = .4$

Discrete Probability

- An **event** is a subset of the sample space
  - Let $\Omega = \{1, 2, 3, 4, 5, 6\}$ be the 6 possible outcomes of a dice role
  - $A = \{1, 5, 6\} \subseteq \Omega$ would be the event that the dice roll comes up as a one, five, or six
- The probability of an event is just the sum of all of the outcomes that it contains
  - $p(A) = p(1) + p(5) + p(6)$

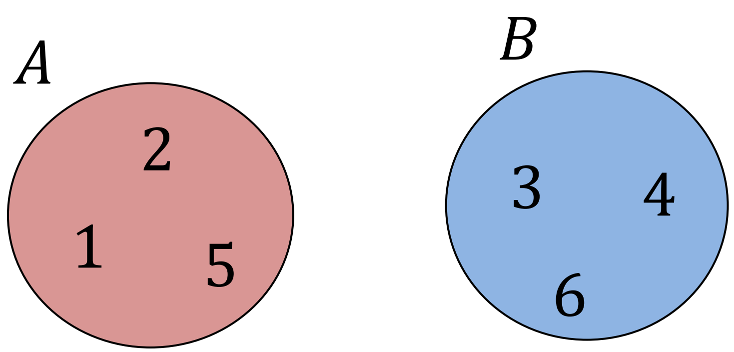
Independence

- Two events A and B are **independent** if

$$p(A \cap B) = p(A)P(B)$$

Let's suppose that we have a fair die: $p(1) = \ \ldots = p(6) = 1/6$

If $A = \{1, 2, 5\}$ and $B = \{3, 4, 6\}$ are A and B indpendent?



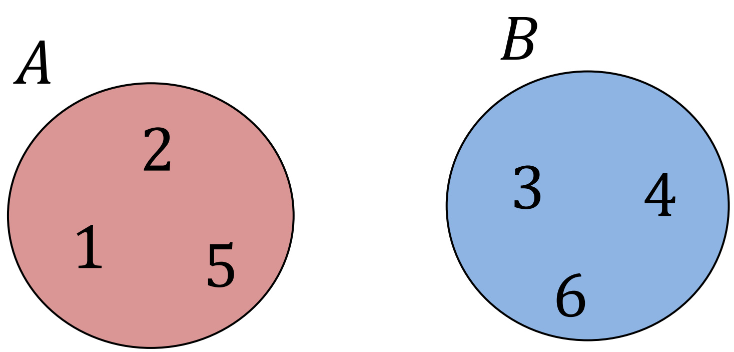
Independence

- Two events A and B are **independent** if

$$p(A \cap B) = p(A)P(B)$$

Let's suppose that we have a fair die: $p(1) = \ ... = p(6) = 1/6$

If $A = \{1, 2, 5\}$ and $B = \{3, 4, 6\}$ are A and B indpendent?



No!

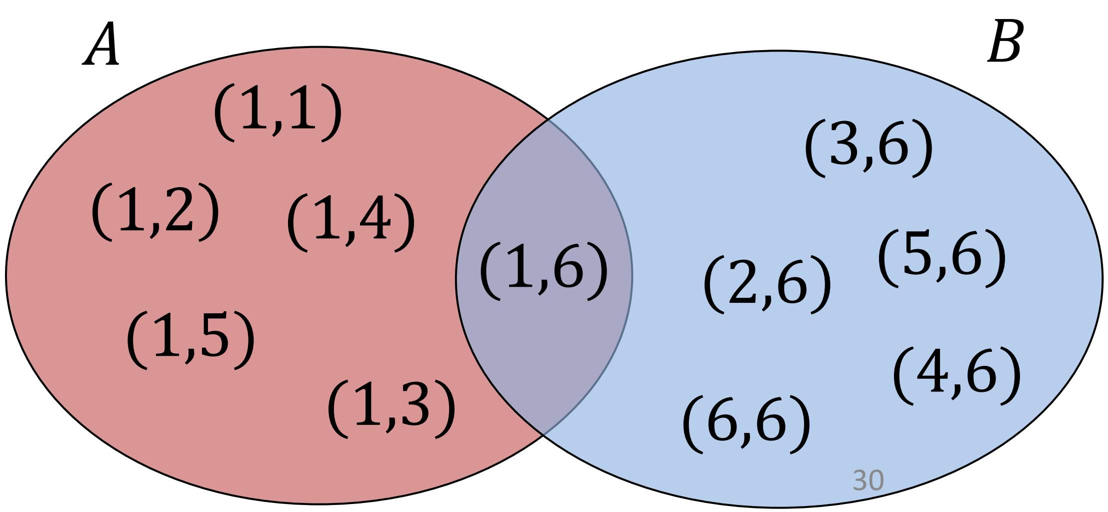
$$p(A \cap B) = 0 \neq \frac{1}{4}$$

Independence

- Now, suppose that $\Omega = \{(1,1), (1,2), \dots, (6,6)\}$ is the set of all possible rolls of two **unbiased** dice

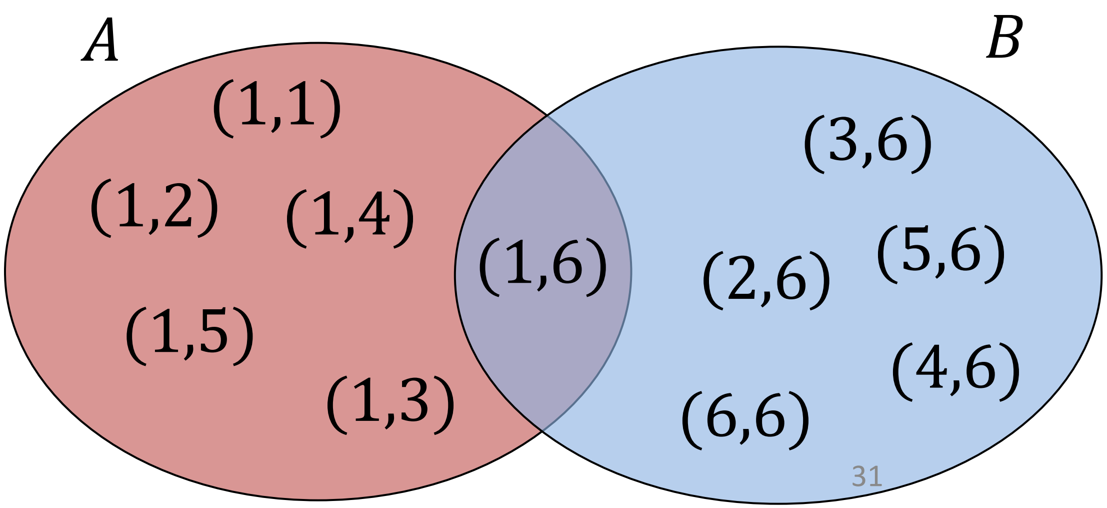
- Let $A = \{(1,1), (1,2), (1,3), \dots, (1,6)\}$ be the event that the first die is a one and let $B = \{(1,6), (2,6), \dots, (6,6)\}$ be the event that the second die is a six

- Are A and B independent?



Independence

- Now, suppose that $\Omega = \{(1,1), (1,2), \dots, (6,6)\}$ is the set of all possible rolls of two **unbiased** dice
- Let $A = \{(1,1), (1,2), (1,3), \dots, (1,6)\}$ be the event that the first die is a one and let $B = \{(1,6), (2,6), \dots, (6,6)\}$ be the event that the second die is a six
- Are A and B independent?



Yes!

$$p(A \cap B) = \frac{1}{36} = \frac{1}{6} * \frac{1}{6}$$

Conditional Probability

- The **conditional probability** of an event A given an event B with $p(B) > 0$ is defined to be

$$p(A|B) = \frac{p(A \cap B)}{P(B)}$$

- This is the probability of the event $A \cap B$ over the sample space $\Omega' = B$

- Some properties:

  - $\sum_{\omega \in \Omega} p(\omega|B) = 1$

  - If A and B are independent, then $p(A|B) = p(A)$

Discrete Random Variables

- A discrete **random variable**, X, is a function from the state space Ω into a discrete space D
  - For each $x \in D$,

$$p(X = x) \equiv p(\{\omega \in \Omega : X(\omega) = x\})$$

    is the probability that X takes the **value** x

  - $p(X)$ defines a probability distribution
    - $\sum_{x \in D} p(X = x) = 1$
- Random variables partition the state space into disjoint events

Example: Pair of Dice

- Let Ω be the set of all possible outcomes of rolling a pair of dice
- Let p be the uniform probability distribution over all possible outcomes in Ω
- Let $X(\omega)$ be equal to the sum of the value showing on the pair of dice in the outcome ω
  - $p(X = 2) = ?$
  - $p(X = 8) = ?$

Example: Pair of Dice

- Let Ω be the set of all possible outcomes of rolling a pair of dice
- Let p be the uniform probability distribution over all possible outcomes in Ω
- Let $X(\omega)$ be equal to the sum of the value showing on the pair of dice in the outcome ω
  - $p(X = 2) = \frac{\mathbf{1}}{\mathbf{36}}$
  - $p(X = 8) = ?$

Example: Pair of Dice

- Let Ω be the set of all possible outcomes of rolling a pair of dice
- Let p be the uniform probability distribution over all possible outcomes in Ω
- Let $X(\omega)$ be equal to the sum of the value showing on the pair of dice in the outcome ω
  - $p(X = 2) = \frac{\mathbf{1}}{\mathbf{36}}$
  - $p(X = 8) = \frac{\mathbf{5}}{\mathbf{36}}$

Discrete Random Variables

- We can have vectors of random variables as well

$$X(\omega) = [X_1(\omega), \dots, X_n(\omega)]$$

- The **joint distribution** is $p(X_1 = x_1, \dots, X_n = x_n)$ is

$$p(X_1 = x_1 \cap \cdots \cap X_n = x_n)$$

typically written as

$$p(x_1, \dots, x_n)$$

- Because $X_i = x_i$ is an event, all of the same rules from basic probability apply

Entropy

- A standard way to measure uncertainty of a random variable is to use the entropy

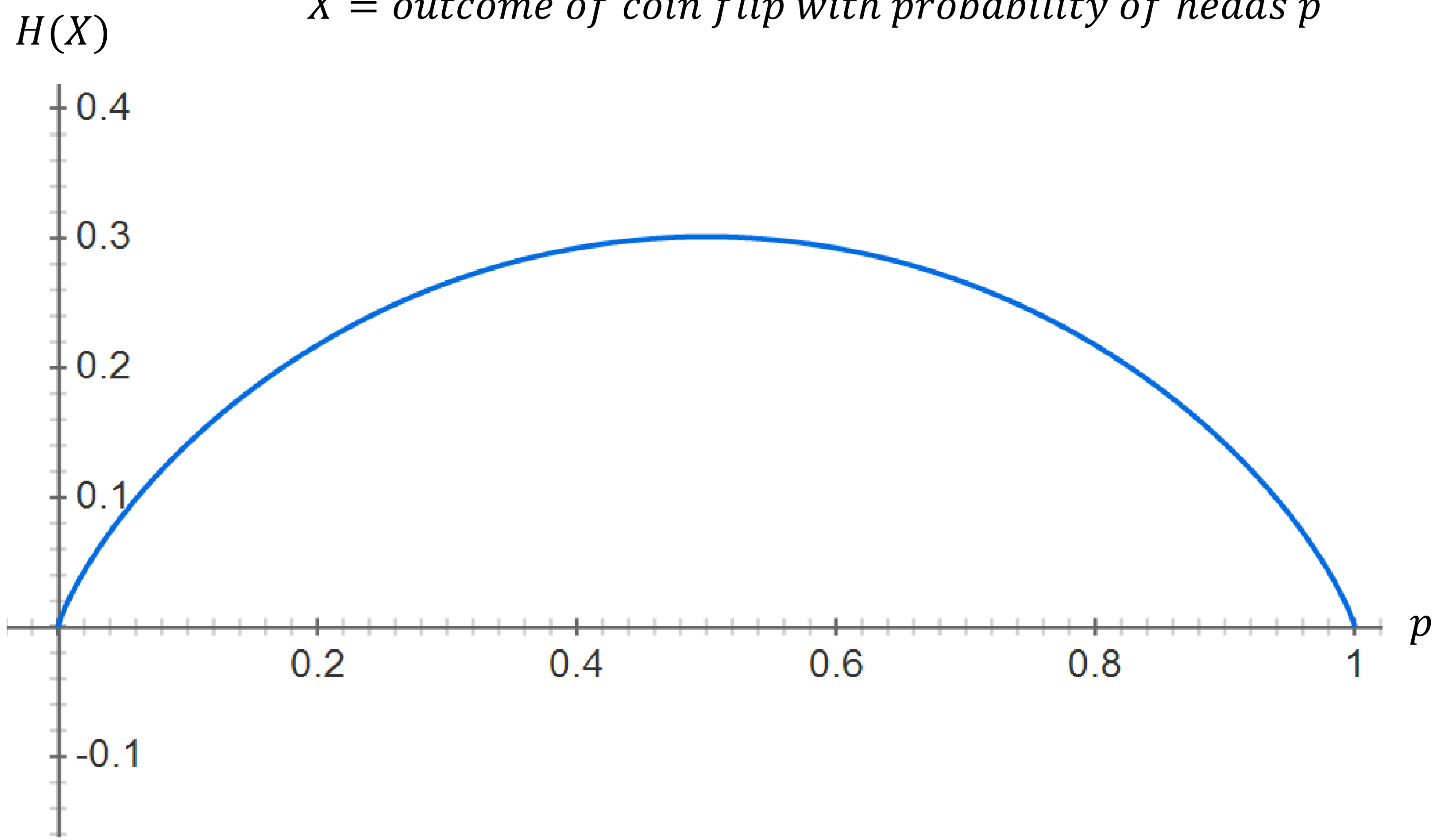
$$H(Y) = -\sum_{Y=y} p(Y = y) \log p(Y = y)$$

- Entropy is maximized for uniform distributions

- Entropy is minimized for distributions that place all their probability on a single outcome

Entropy of a Coin Flip

$$X = outcome\ of\ coin\ flip\ with\ probability\ of\ heads\ p$$



Conditional Entropy

- We can also compute the entropy of a random variable conditioned on a different random variable

$$H(Y|X) = -\sum_x p(X = x) \sum_y p(Y = y|X = x) \log p(Y = y|X = x)$$

  - This is called the conditional entropy
  - This is the amount of information needed to quantify the random variable Y given the random variable X

Information Gain

- Using entropy to measure uncertainty, we can greedily select an attribute that guarantees the largest expected decrease in entropy (with respect to the empirical partitions)

$$IG(X) = H(Y) - H(Y|X)$$

  - Called information gain
  - Larger information gain corresponds to less uncertainty about Y given X
    - Note that $H(Y|X) \leq H(Y)$

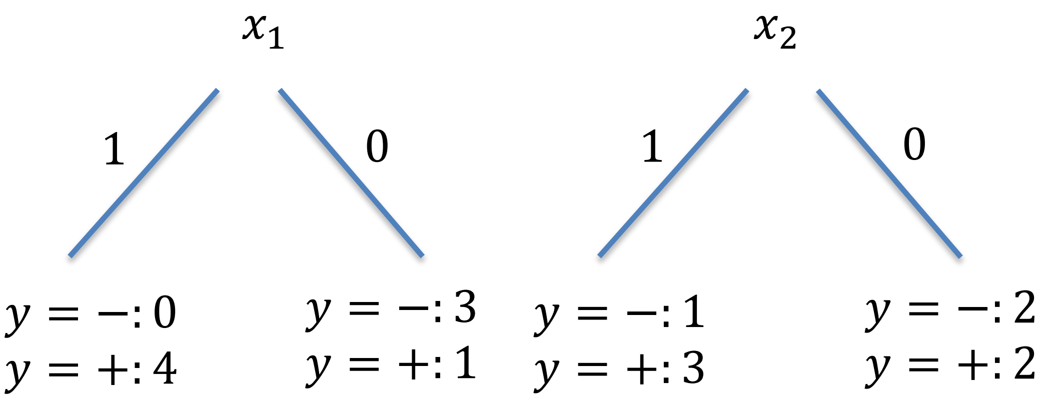
Decision Tree Learning

- Basic decision tree building algorithm:
  - Pick the feature/attribute with the highest information gain
  - Partition the data based on the value of this attribute
  - Recurse over each new partition

Choosing the Best Attribute

$$x_1, x_2 \in \{0,1\}$$

Which attribute should you split on?



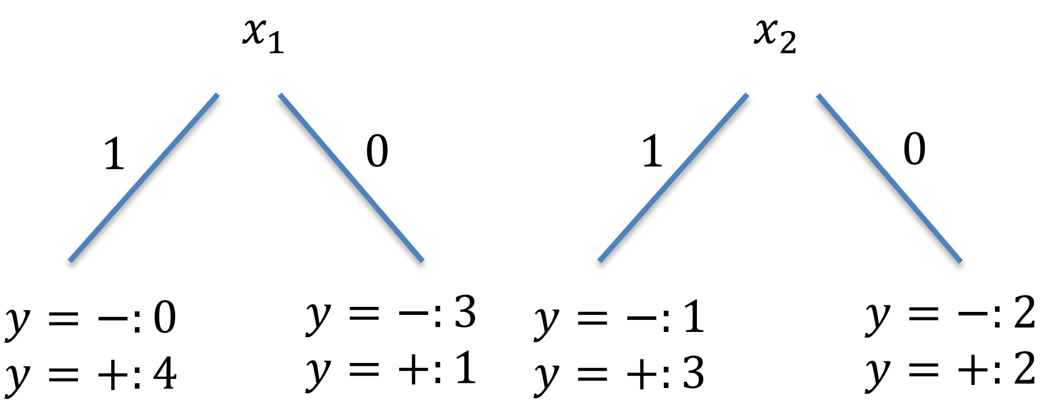
x_1	x_2	y
1	1	+
1	0	+
1	1	+
1	0	+
0	1	+
0	0	−
0	1	−
0	0	−

What is the information gain in each case?

Choosing the Best Attribute

$$x_1, x_2 \in \{0,1\}$$

Which attribute should you split on?



x_1	x_2	y
1	1	+
1	0	+
1	1	+
1	0	+
0	1	+
0	0	−
0	1	−
0	0	−

$$H(Y) = -\frac{5}{8}\log\frac{5}{8} - \frac{3}{8}\log\frac{3}{8}$$

$$H(Y|X_1) = .5[-0\log 0 - 1\log 1] + .5[-.75\log .75 - .25\log .25]$$

$$H(Y|X_2) = .5[-.5\log .5 - .5\log .5] + .5[-.75\log .75 - .25\log .25]$$

$$H(Y) - H(Y|X_1) - H(Y) + H(Y|X_2) = -.5\log .5 > 0$$

Should split on x_1

When to Stop

- If the current set is “pure” (i.e., has a single label in the output), stop
- If you run out of attributes to recurse on, even if the current data set isn’t pure, stop and use a majority vote
- If a partition contains no data points, use the majority vote at its parent in the tree
- If a partition contains no data items, nothing to recurse on
- For fixed depth decision trees, the final label is determined by majority vote

Handling Real-Valued Attributes

- For continuous attributes, use threshold splits
  - Split the tree into $x_k < t$ and $x_k \geq t$
  - Can split on the same attribute multiple times on the same path down the tree
- How to pick the threshold t?

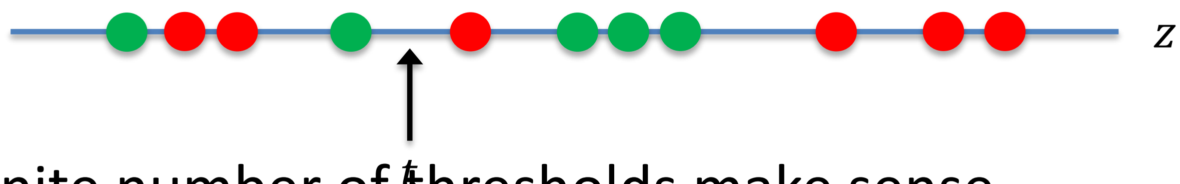
Handling Real-Valued Attributes

- For continuous attributes, use threshold splits
  - Split the tree into $x_k < t$ and $x_k \geq t$
  - Can split on the same attribute multiple times on the same path down the tree
- How to pick the threshold t?
  - Try every possible t

How many possible t are there?

Handling Real-Valued Attributes

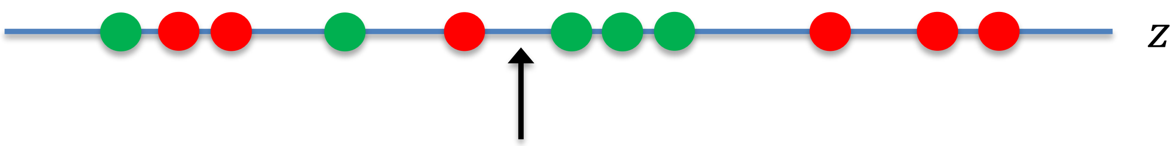
- Sort the data according to the k^{th} attribute: $z_1 > z_2 > \cdots > z_n$



- Only a finite number of thresholds make sense

Handling Real-Valued Attributes

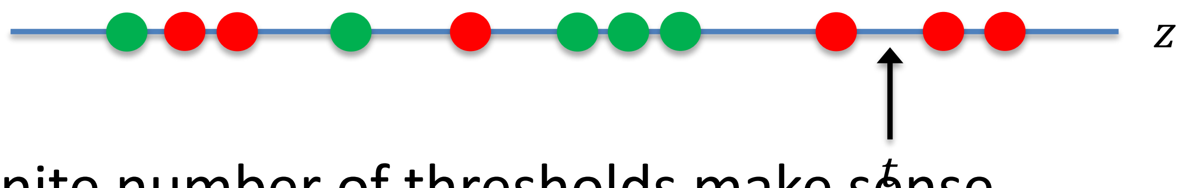
- Sort the data according to the k^{th} attribute: $z_1 > z_2 > \cdots > z_n$



- Only a finite number of thresholds make sense
  - Just split in between each consecutive pair of data points (e.g., splits of the form $t = \frac{z_i + z_{i+1}}{2}$)

Handling Real-Valued Attributes

- Sort the data according to the k^{th} attribute: $z_1 > z_2 > \cdots > z_n$



Does it make sense for a threshold to appear between two x's with the same class label?

- Only a finite number of thresholds make sense
  - Just split in between each consecutive pair of data points (e.g., splits of the form $t = \frac{z_i + z_{i+1}}{2}$)

Handling Real-Valued Attributes

- Compute the information gain of each threshold
- Let $X\!:t$ denote splitting with threshold t and compute

$$H(Y|X\!:t) = -p(X < t)\sum_y p(Y = y|X < t)\log p(Y = y|X < t) + \\ -p(X \geq t)\sum_y p(Y = y|X \geq t)\log p(Y = y|X \geq t)$$

- In the learning algorithm, maximize over all attributes and all possible thresholds of the real-valued attributes

$$\max_t H(Y) - H(Y|X\!:t), \text{ for real-valued } X$$

$$H(Y) - H(Y|X), \text{ for discrete } X$$

Decision Trees

- Because of speed/ease of implementation, decision trees are quite popular
    - Can be used for regression too
- Decision trees will **always** overfit!
    - It is always possible to obtain zero training error on the input data with a deep enough tree (if there is no noise in the labels)
    - Solution?