

Course 395: Machine Learning – Lectures

- Lecture 1-2: Concept Learning
- Lecture 3-4: Decision Trees & CBC Intro
- Lecture 5-6: Evaluating Hypotheses
- Lecture 7-8: Artificial Neural Networks I
- Lecture 9-10: Artificial Neural Networks II
- Lecture 11-12: Artificial Neural Networks III
- Lecture 13-14: Instance Based Learning & Genetic Algorithms

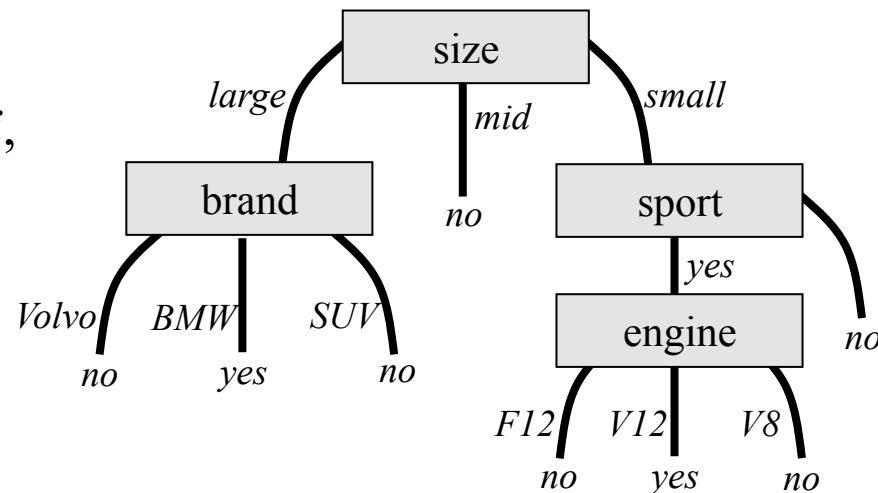
Decision Trees – Lecture Overview

- Problem Representation using a Decision Tree
- ID3 algorithm
- The problem of overfitting

Problem Representation using a Decision Tree

- Decision Tree learning is a method for approximating discrete classification functions by means of a tree-based representation
- A learned Decision Tree classifies a new instance by sorting it down the tree
 - tree node \leftrightarrow classification OR test of a specific attribute of the instance
 - tree branch \leftrightarrow possible value for the attribute in question

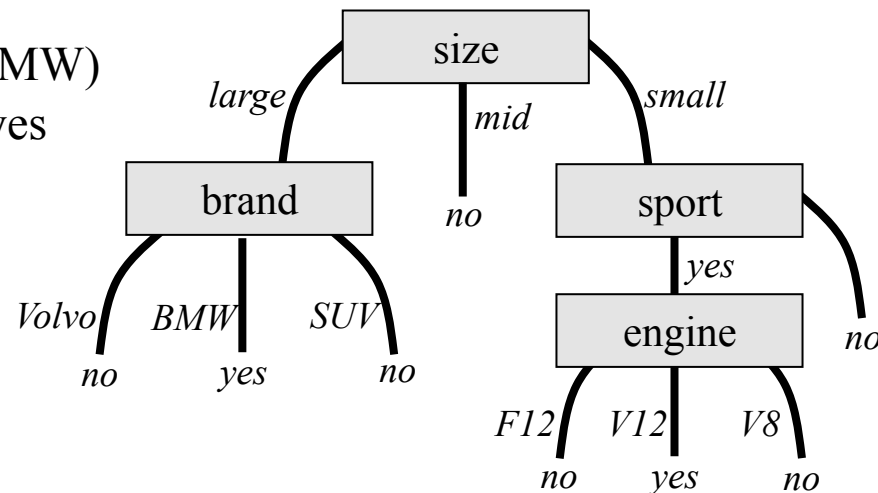
- Concept: *Good Car*
 \langle size = *small*, brand = *Ferari*,
model = *Enzo*, sport = *yes*,
engine = *V12*, colour = *red* \rangle



Problem Representation using a Decision Tree

- A learned Decision Tree can be represented as a set of *if-then* rules
- To ‘read out’ the rules from a learned Decision Tree
 - tree \leftrightarrow disjunction (\vee) of sub-trees
 - sub-tree \leftrightarrow conjunction (\wedge) of constraints on the attribute values
- Rule: *Good Car*

IF (size = large AND brand = BMW)
OR (size = small AND sport = yes
AND engine = V12)
THEN Good Car = yes
ELSE Good Car = no;



Decision Tree Learning Algorithm

- Decision Tree learning algorithms employ top-down greedy search through the space of possible solutions.
- A general Decision Tree learning algorithm:
 1. perform a statistical test of each attribute to determine how well it classifies the training examples when considered alone;
 2. select the attribute that performs best and use it as the root of the tree;
 3. to decide the descendant node down each branch of the root (parent node), sort the training examples according to the value related to the current branch and repeat the process described in steps 1 and 2.
- ID3 algorithm is one of the most commonly used Decision Tree learning algorithms and it applies this general approach to learning the decision tree.

ID3 Algorithm

- ID3 algorithm uses so-called *Information Gain* to determine how informative an attribute is (i.e., how well it classifies the training examples).
- *Information Gain* is based on a measure that we call *Entropy*, which characterizes the impurity of a collection of examples S (i.e., impurity $\uparrow \rightarrow E(S)\uparrow$):

$$E(S) \equiv - p^{\oplus} \log_2 p^{\oplus} - p^{\otimes} \log_2 p^{\otimes},$$

where p^{\oplus} (p^{\otimes}) is the proportion of positive (negative) examples in S .

(Note: $E(S) = 0$ if S contains only positive or only negative examples

$$\Leftrightarrow p^{\oplus} = 1, p^{\otimes} = 0, E(S) = -1 \cdot 0 - 0 \cdot \log_2 p^{\otimes} = 0$$

(Note: $E(S) = 1$ if S contains equal amount of positive and negative examples

$$\Leftrightarrow p^{\oplus} = 1/2, p^{\otimes} = 1/2, E(S) = -1/2 \cdot (-1) - 1/2 \cdot (-1) = 1$$

In the case that that the target attribute can take n values:

$$E(S) \equiv - \sum_i p_i \log_2 p_i, i = [1..n]$$

where p_i is the proportion of examples in S having the target attribute value i .

ID3 Algorithm

- *Information Gain* is based on a measure that we call *Entropy*, which characterizes the impurity of a collection of examples S (impurity $\uparrow \rightarrow E(S) \uparrow$):
$$E(S) \equiv -p^{\oplus} \log_2 p^{\oplus} - p^{\otimes} \log_2 p^{\otimes},$$
where p^{\oplus} (p^{\otimes}) is the proportion of positive (negative) examples in S .
(Note: $E(S) = 0$ if S contains only positive or only negative examples
 $\Leftrightarrow p^{\oplus} = 1, p^{\otimes} = 0, E(S) = -1 \cdot 0 - 0 \cdot \log_2 p^{\otimes} = 0$
(Note: $E(S) = 1$ if S contains equal amount of positive and negative examples
 $\Leftrightarrow p^{\oplus} = 1/2, p^{\otimes} = 1/2, E(S) = -1/2 \cdot (-1) - 1/2 \cdot (-1) = 1$
In the case that that the target attribute can take n values:
$$E(S) \equiv -\sum_i p_i \log_2 p_i, i = [1..n]$$
where p_i is the proportion of examples in S having the target attribute value i .
- *Information Gain* – Reduction in $E(S)$ caused by partitioning S according to attribute A
$$IG(S, A) = E(S) - \sum_{v \in \text{values}(A)} (|S_v| / |S|) E(S_v)$$
where $\text{values}(A)$ are all possible values for attribute A , $S_v \in S$ contains all examples for which attribute A has the value v , and $|S_v|$ is the cardinality of set S_v .

ID3 Algorithm – Example

1. For each attribute A of the training examples in set S calculate:
$$IG(S, A) = E(S) - \sum_{v \in \text{values}(A)} (|S_v| / |S|) E(S_v), E(S_v) \equiv \sum_v - p_v \log_2 p_v, v = [1..n].$$
2. Select the attribute with the maximal $IG(S, A)$ and use it as the root of the tree.
3. To decide the descendant node down each branch of the root (i.e., parent node), sort the training examples according to the value related to the current branch and repeat the process described in steps 1 and 2.

Target concept: *Play Tennis* (Mitchell's book, p. 59)

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
1	0	sunny	hot	high	weak
2	0	sunny	hot	high	strong
...
13	1	overcast	hot	normal	weak
14	0	rain	mild	high	strong

$$IG(D, Outlook) = E(D) - 5/14 E(D_{sunny}) - 4/14 E(D_{overcast}) - 5/14 E(D_{rain})$$

ID3 Algorithm – Example

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
1	0	sunny	hot	high	weak
2	0	sunny	hot	high	strong
3	1	overcast	hot	high	weak
4	1	rain	mild	high	weak
5	1	rain	cool	normal	weak
6	0	rain	cool	normal	strong
7	1	overcast	cool	normal	strong
8	0	sunny	mild	high	weak
9	1	sunny	cool	normal	weak
10	1	rain	mild	normal	weak
11	1	sunny	mild	normal	strong
12	1	overcast	mild	high	strong
13	1	overcast	hot	normal	weak
14	0	rain	mild	high	strong

ID3 Algorithm – Example

1. For each attribute A of the training examples in set S calculate:

$$IG(S, A) = E(S) - \sum_{v \in \text{values}(A)} (|S_v| / |S|) E(S_v), E(S_v) \equiv \sum_v - p_v \log_2 p_v, v = [1..n].$$
2. Select the attribute with the maximal $IG(S, A)$ and use it as the root of the tree.
3. To decide the descendant node down each branch of the root (i.e., parent node), sort the training examples according to the value related to the current branch and repeat the process described in steps 1 and 2.

Target concept: *Play Tennis* (Mitchell's book, p. 59)

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
...

$$IG(D, Outlook) = E(D) - 5/14 E(D_{sunny}) - 4/14 E(D_{overcast}) - 5/14 E(D_{rain})$$

$$= 0.940 - 0.357 \cdot 0.971 - 0 - 0.357 \cdot 0.971 = 0.246$$

$$IG(D, Temperature) = E(D) - 4/14 E(D_{hot}) - 6/14 E(D_{mild}) - 4/14 E(D_{cool})$$

$$= 0.940 - 0.286 \cdot 1 - 0.429 \cdot 0.918 - 0.286 \cdot 0.811 = 0.029$$

$$IG(D, Humidity) = E(D) - 7/14 E(D_{high}) - 7/14 E(D_{normal}) = 0.940 - \frac{1}{2} \cdot 0.985 - \frac{1}{2} \cdot 0.591 = 0.151$$

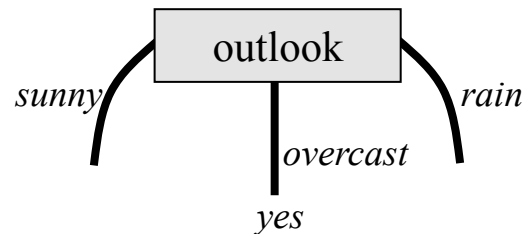
$$IG(D, Wind) = E(D) - 8/14 E(D_{weak}) - 6/14 E(D_{strong}) = 0.940 - 0.571 \cdot 0.811 - 0.429 \cdot 1 = 0.048$$

ID3 Algorithm – Example

1. For each attribute A of the training examples in set S calculate:
$$IG(S, A) = E(S) - \sum_{v \in \text{values}(A)} (|S_v| / |S|) E(S_v), E(S_v) \equiv \sum_v - p_v \log_2 p_v, v = [1..n].$$
2. Select the attribute with the maximal $IG(S, A)$ and use it as the root of the tree.
3. To decide the descendant node down each branch of the root (i.e., parent node), sort the training examples according to the value related to the current branch and repeat the process described in steps 1 and 2.

Target concept: *Play Tennis* (Mitchell's book, p. 59)

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
...



ID3 Algorithm – Example

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
1	0	sunny	hot	high	weak
2	0	sunny	hot	high	strong
3	1	overcast	hot	high	weak
4	1	rain	mild	high	weak
5	1	rain	cool	normal	weak
6	0	rain	cool	normal	strong
7	1	overcast	cool	normal	strong
8	0	sunny	mild	high	weak
9	1	sunny	cool	normal	weak
10	1	rain	mild	normal	weak
11	1	sunny	mild	normal	strong
12	1	overcast	mild	high	strong
13	1	overcast	hot	normal	weak
14	0	rain	mild	high	strong

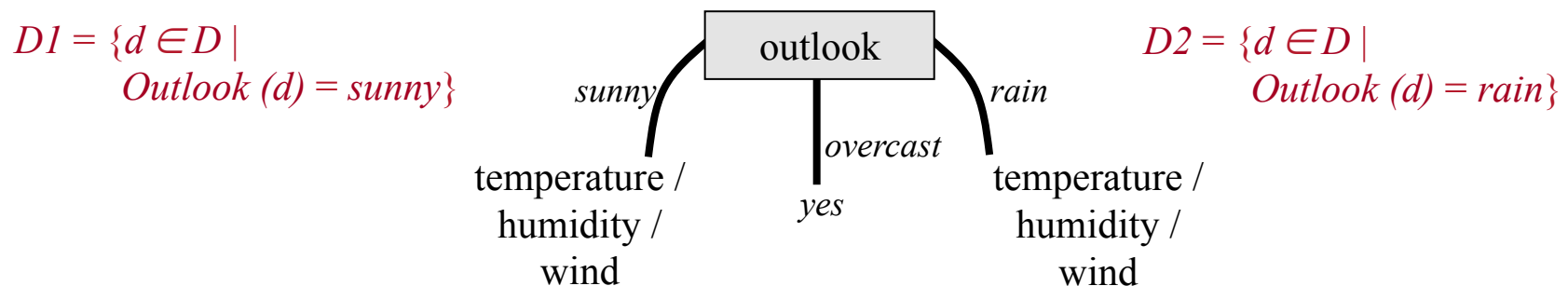
ID3 Algorithm – Example

1. For each attribute A of the training examples in set S calculate:

$$IG(S, A) = E(S) - \sum_{v \in \text{values}(A)} (|S_v| / |S|) E(S_v), E(S_v) \equiv \sum_v - p_v \log_2 p_v, v = [1..n].$$
2. Select the attribute with the maximal $IG(S, A)$ and use it as the root of the tree.
3. To decide the descendant node down each branch of the root (i.e., parent node), sort the training examples according to the value related to the current branch and repeat the process described in steps 1 and 2.

Target concept: *Play Tennis* (Mitchell's book, p. 59)

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
...




ID3 Algorithm – Example

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>	
1	0	sunny	hot	high	weak	} <i>D1</i>
2	0	sunny	hot	high	strong	
8	0	sunny	mild	high	weak	
9	1	sunny	cool	normal	weak	
11	1	sunny	mild	normal	strong	
4	1	rain	mild	high	weak	} <i>D2</i>
5	1	rain	cool	normal	weak	
6	0	rain	cool	normal	strong	
10	1	rain	mild	normal	weak	
14	0	rain	mild	high	strong	
3	1	overcast	hot	high	weak	
7	1	overcast	cool	normal	strong	
12	1	overcast	mild	high	strong	
13	1	overcast	hot	normal	weak	

ID3 Algorithm – Example

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
1	0	sunny	hot	high	weak
2	0	sunny	hot	high	strong
8	0	sunny	mild	high	weak
9	1	sunny	cool	normal	weak
11	1	sunny	mild	normal	strong
...



$$E(D1) = \text{abs}(-2/5 \log_2 2/5 - 3/5 \log_2 3/5) = 0.971$$

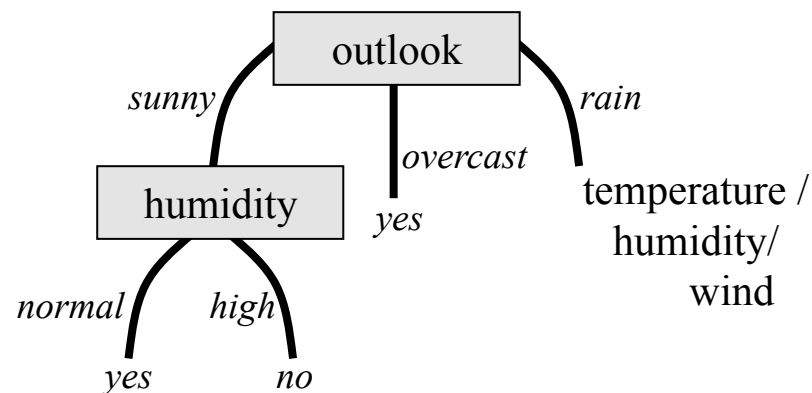

$$\begin{aligned} IG(D1, \text{Temperature}) &= E(D1) - 2/5 E(D1_{hot}) - 2/5 E(D1_{mild}) - 1/5 E(D1_{cool}) \\ &= 0.971 - 0.4 \cdot 0 - 0.4 \cdot 1 - 0.4 \cdot 0 = 0.571 \end{aligned}$$

$$\begin{aligned} IG(D1, \text{Humidity}) &= E(D1) - 3/5 E(D1_{high}) - 2/5 E(D1_{normal}) \\ &= 0.971 - 0.6 \cdot 0 - 0.4 \cdot 0 = \mathbf{0.971} \end{aligned}$$

$$\begin{aligned} IG(D1, \text{Wind}) &= E(D1) - 3/5 E(D1_{weak}) - 2/5 E(D1_{strong}) \\ &= 0.971 - 0.6 \cdot 0.918 - 0.4 \cdot 1 = 0.02 \end{aligned}$$


ID3 Algorithm – Example

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
1	0	sunny	hot	high	weak
2	0	sunny	hot	high	strong
8	0	sunny	mild	high	weak
9	1	sunny	cool	normal	weak
11	1	sunny	mild	normal	strong
...



ID3 Algorithm – Example

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
4	1	rain	mild	high	weak
5	1	rain	cool	normal	weak
6	0	rain	cool	normal	strong
10	1	rain	mild	normal	weak
14	0	rain	mild	high	strong
...



$$E(D2) = \text{abs}(-3/5 \log_2 3/5 - 2/5 \log_2 2/5) = 0.971$$

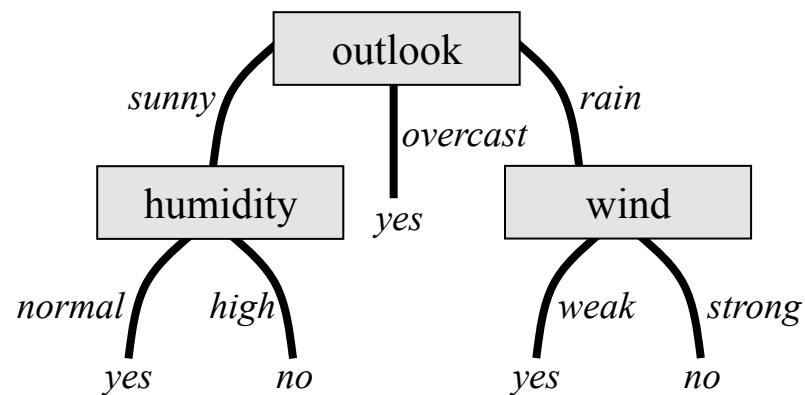
$$\begin{aligned} IG(D2, \text{Temperature}) &= E(D2) - 0/5 E(D2_{hot}) - 3/5 E(D2_{mild}) - 2/5 E(D2_{cool}) \\ &= 0.971 - 0 - 0.6 \cdot 0.918 - 0.4 \cdot 1 = 0.02 \end{aligned}$$

$$\begin{aligned} IG(D2, \text{Humidity}) &= E(D2) - 2/5 E(D2_{high}) - 3/5 E(D2_{normal}) \\ &= 0.971 - 0.4 \cdot 1 - 0.6 \cdot 0.918 = 0.02 \end{aligned}$$

$$\begin{aligned} IG(D2, \text{Wind}) &= E(D2) - 3/5 E(D2_{weak}) - 2/5 E(D2_{strong}) \\ &= 0.971 - 0.6 \cdot 0 - 0.4 \cdot 0 = \mathbf{0.971} \end{aligned}$$

ID3 Algorithm – Example

	<i>PlayTennis(d)</i>	<i>outlook</i>	<i>temperature</i>	<i>humidity</i>	<i>wind</i>
4	1	rain	mild	high	weak
5	1	rain	cool	normal	weak
6	0	rain	cool	normal	strong
10	1	rain	mild	normal	weak
14	0	rain	mild	high	strong
...



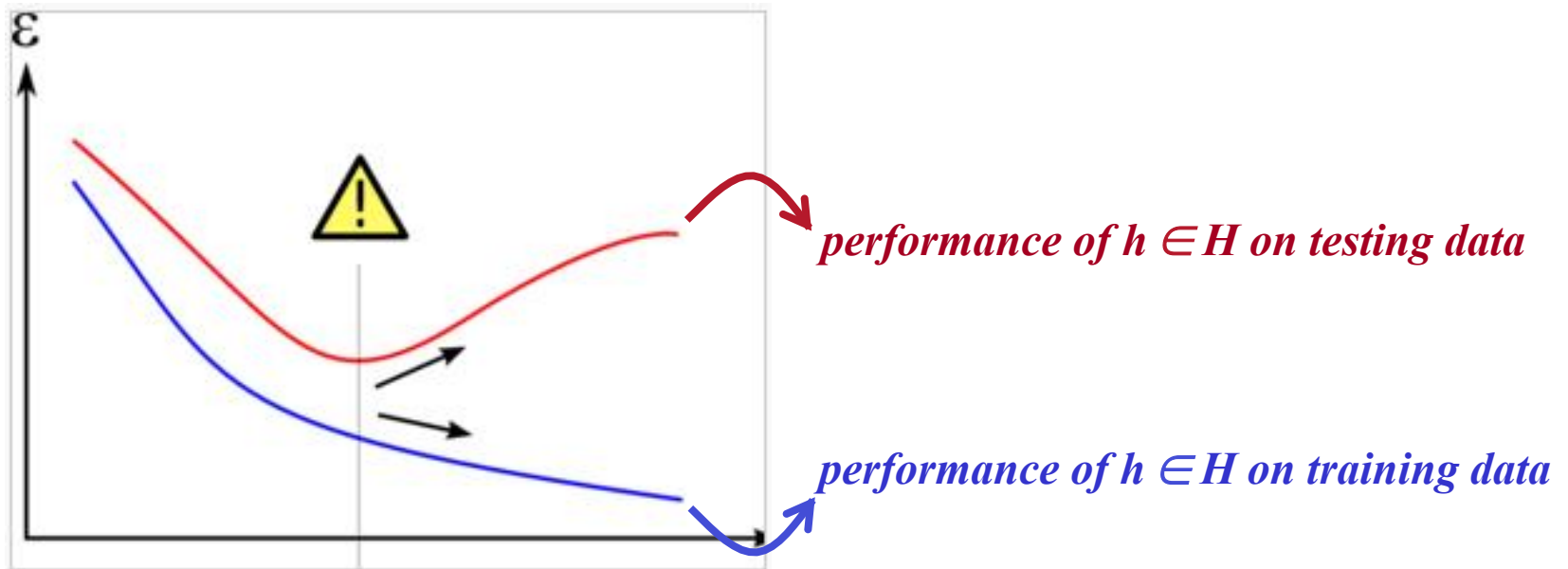
ID3 Algorithm – Advantages & Disadvantages

- Advantages of ID3 algorithm:
 1. Every discrete classification function can be represented by a decision tree → it cannot happen that ID3 will search an incomplete hypothesis space.
 2. Instead of making decisions based on individual training examples (as is the case by Find-S and Candidate-Elimination algorithms), ID3 uses statistical properties of all examples (information gain) → resulting search is much less sensitive to errors in individual training examples.
- Disadvantages of ID3 algorithm:
 1. ID3 determines a single hypothesis, not a space of consistent hypotheses (as is the case by Candidate-Elimination algorithm) → ID3 cannot determine how many different decision trees are consistent with the available training data.
 2. ID3 grows the tree to perfectly classify the training examples without performing a backtracking in its search → ID3 may overfit the training data and converge to locally optimal solution that is not globally optimal.

The Problem of Overfitting

- Def (Mitchell 1997):

Given a hypothesis space H , $h \in H$ overfits the training data if $\exists h' \in H$ such that h has smaller error over the training examples, but h' has smaller error than h over the entire distribution of instances.



The Problem of Overfitting

- Ways to avoid overfitting:
 1. Stop the training process before the learner reaches the point where it perfectly classifies the training data.
 2. Apply backtracking in the search for the optimal hypothesis. In the case of Decision Tree Learning, backtracking process is referred to as ‘post-pruning of the overfitted tree’.
- Ways to determine the correctness of the learner’s performance:
 1. Use two different sets of examples: training set and validation set.
 2. Use all examples for training, but apply a statistical test to estimate whether a further training will produce a statistically significant improvement of the learner’s performance. In the case of Decision Tree Learning, the statistical test should estimate whether expanding / pruning a particular node will result in a statistically significant improvement of the performance.
 3. Combine 1. and 2.

Decision Tree Learning – Exam Questions

- Tom Mitchell's book –chapter 3
- Relevant exercises from chapter 3: 3.1, 3.2, 3.3, 3.4

Course 395: Machine Learning – Lectures

- Lecture 1-2: Concept Learning
- Lecture 3-4: Decision Trees & CBC Intro
- Lecture 5-6: Evaluating Hypotheses
- Lecture 7-8: Artificial Neural Networks I
- Lecture 9-10: Artificial Neural Networks II
- Lecture 11-12: Artificial Neural Networks III
- Lecture 13-14: Instance Based Learning & Genetic Algorithms