

# Learning

<b>Learning</b>	<b>2</b>
Learning Definitions . . . . .	2
More Learning Definitions . . . . .	3
Example of Examples . . . . .	4
More about Inductive Learning . . . . .	5
Error in Learning . . . . .	6
<b>Decision Trees</b>	<b>7</b>
Definition . . . . .	7
Example of a Decision Tree . . . . .	8
Algorithm for Growing Decision Trees . . . . .	9
Comparing Attributes: Information Gain . . . . .	10
Plot of Information Function . . . . .	11
Plot of Information Gain . . . . .	12
Example of Attribute Selection . . . . .	13
Attribute Selection, Continued . . . . .	14
Alternative Attributes Measures . . . . .	15
Special Cases in Decision Trees . . . . .	16
Pruning Decision Trees . . . . .	17
Estimating Error. . . . .	18
Algorithm for Pruning Decision Trees . . . . .	19
<b>Ensemble Learning</b>	<b>20</b>
Definition . . . . .	20
Boosting . . . . .	21
Example Boosting Algorithm . . . . .	22
Example Run of AdaBoost. . . . .	23
Example Run of AdaBoost, Continued . . . . .	24

## Learning

### Learning Definitions

- Learning* is improvement of performance (time, accuracy).
- Inductive inference* is improving accuracy by generalizing from experience.
- An *example* is a single, specific experience.
- In *supervised learning*, each example is an input/output pair.
  - *Regression* is when the output is continuous.
  - *Classification* is when the output is discrete.
    - *Concept learning* has two possible outputs (positive or negative).

### More Learning Definitions

- In *unsupervised learning*, examples do not always have outputs.
- In *reinforcement learning*, an agent performs a series of actions, receiving intermittent feedback.
- In *batch* learning, the learner receives all the examples at the same time.
- In *online* learning, the learner receives the examples one at a time.

### Example of Examples

No.	Attributes				Class
	Outlook	Temp	Humidity	Windy	
1	sunny	hot	high	false	neg
2	sunny	hot	high	true	neg
3	overcast	hot	high	false	pos
4	rain	mild	high	false	pos
5	rain	cool	normal	false	pos
6	rain	cool	normal	true	neg
7	overcast	cool	normal	true	pos
8	sunny	mild	high	false	neg
9	sunny	cool	normal	false	pos
10	rain	mild	normal	false	pos
11	sunny	mild	normal	true	pos
12	overcast	mild	high	true	pos
13	overcast	hot	normal	false	pos
14	rain	mild	high	true	neg

## More about Inductive Learning

- The learner learns a *hypothesis*  $h$  from a set of *training* examples.
- $h$  can be evaluated empirically on a set of *test* examples or theoretically on the *probability distribution* of the examples.
- *Inductive bias* refers to the hypotheses that the learner prefers.
- One kind of inductive bias is to restrict the *hypothesis space*, the set of hypotheses to be considered.

CS 5233 Artificial Intelligence

Learning – 5

## Error in Learning

- Perfect learning cannot be guaranteed by any learning algorithm from a finite set of training examples.
- The training examples might not cover all the possibilities, or might not be representative.
- No learning algorithm is best. All learning algorithms are forced to make assumptions which might not be true.
- The goal of *PAC* learning (PAC = “probably approximately correct”) is to find a hypothesis that is unlikely ( $\delta$  or less) to have high error ( $\epsilon$  or more).

CS 5233 Artificial Intelligence

Learning – 6

## Decision Trees

7

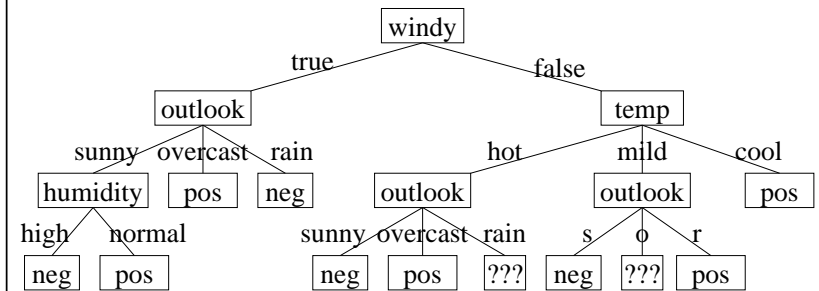
### Definition

- Decision trees are a representation for classification.
  - The root is labeled by an attribute.
  - Edges are labeled by attribute values.
  - Edges go to decision trees or leaves.
  - Each leaf is labeled by a class.
- Growth Phase: Construct the tree top-down.
  - Find the “best” attribute.
  - Split examples based on attribute’s values.
- Pruning Phase: Prune the tree bottom-up.
  - For each node, keep subtree or change to leaf.

CS 5233 Artificial Intelligence

Learning – 7

## Example of a Decision Tree



CS 5233 Artificial Intelligence

Learning – 8

## Algorithm for Growing Decision Trees

### Grow\_DT(*examples*)

1.  $N \leftarrow$  a new node
2.  $N.class \leftarrow$  most common class in *examples*
3. **if** *examples* have identical class or values
4.     **then return**  $N$
5.  $N.test \leftarrow$  best attribute (or test)
6. **for each** value  $v_j$  of  $N.test$
7.      $examples_j \leftarrow$  *examples* with  $N.test = v_j$
8.     **if**  $examples_j$  is empty
9.         **then**  $N.branch_j \leftarrow N.class$
10.     **else**  $N.branch_j \leftarrow$  **Grow\_DT**( $examples_j$ )
11. **return**  $N$

CS 5233 Artificial Intelligence

Learning – 9

### Comparing Attributes: Information Gain

- $p$  positive examples and  $n$  negative examples
- The information contained is:  

$$I(p, n) = -\frac{p}{p+n} \log_2 \frac{p}{p+n} - \frac{n}{p+n} \log_2 \frac{n}{p+n}$$
- Attribute  $A$  has  $v$  values,  $p_j$  positive examples and  $n_j$  negative examples when  $A = v_j$
- The *Remainder* of  $A$  is:

$$\text{Remainder}(A) = \sum_{j=1}^v \frac{p_j + n_j}{p+n} I(p_j, n_j)$$

- The information gain of  $A$  is:

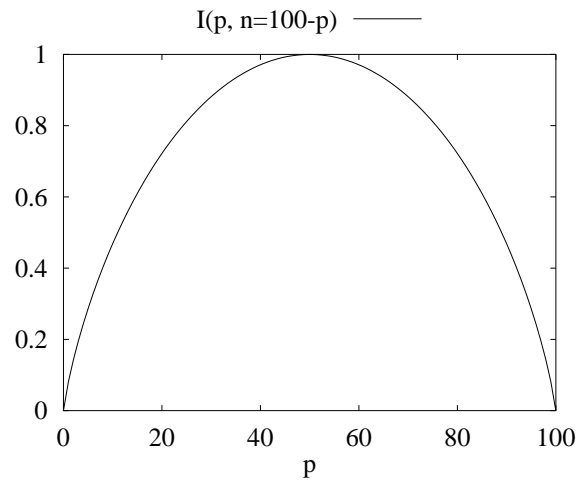
$$\text{Gain}(A) = I(p, n) - \text{Remainder}(A)$$

CS 5233 Artificial Intelligence

Learning - 10

### Plot of Information Function

$p$  positive examples and  $n$  negative examples

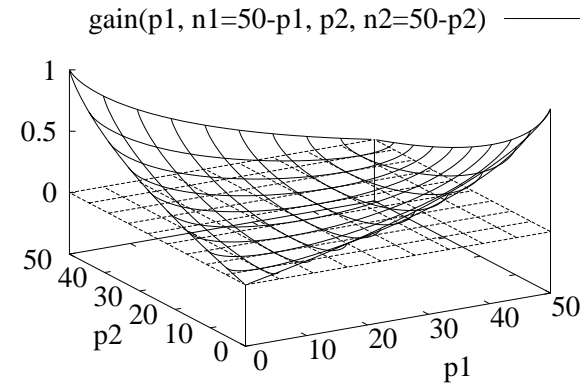


CS 5233 Artificial Intelligence

Learning - 11

### Plot of Information Gain

$p_1$  positive and  $n_1$  negative exs. when attr.= $v_1$   
 $p_2$  positive and  $n_2$  negative exs. when attr.= $v_2$

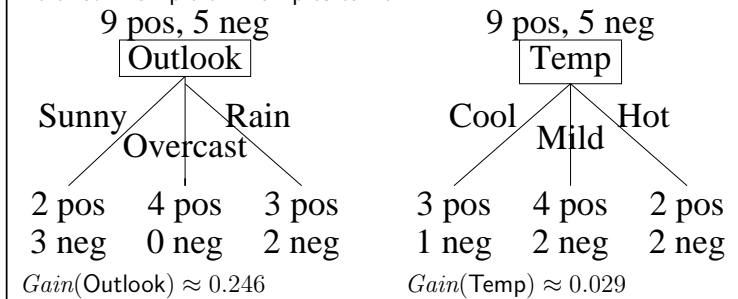


CS 5233 Artificial Intelligence

Learning - 12

### Example of Attribute Selection

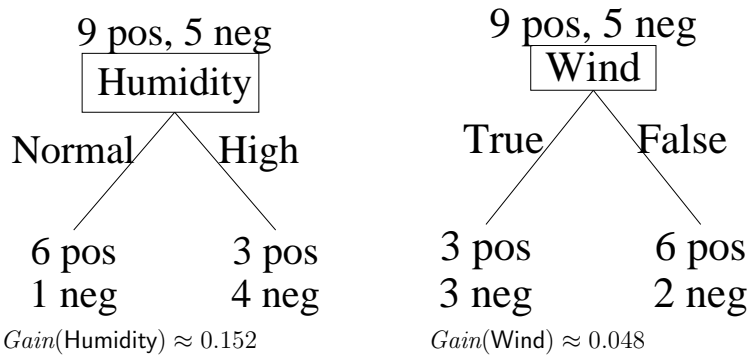
Refer to Example of Examples earlier.



CS 5233 Artificial Intelligence

Learning - 13

## Attribute Selection, Continued



Outlook has the highest gain.  
 Overcast branch is pure.  
 Need to construct DTs for two branches.

CS 5233 Artificial Intelligence

Learning - 14

## Alternative Attributes Measures

- Maximize Information Gain Ratio  
 $GainRatio(A) = Gain(A) / I(p_1 + n_1, \dots, p_v + n_v)$

- Minimize Gini Index

$$Gini(p, n) = 1 - \left(\frac{p}{p+n}\right)^2 - \left(\frac{n}{p+n}\right)^2$$

$$GiniIndex(A) = \sum_{j=1}^v \frac{p_j + n_j}{p+n} Gini(p_j, n_j)$$

- "Maximize" Chi-Squared Statistic

$$\chi^2 = \sum_{j=1}^v \frac{(p_j - p s_j)^2}{p s_j} + \frac{(n_j - n s_j)^2}{n s_j}$$

$$\text{where } s_j = (p_j + n_j) / (p + n)$$

CS 5233 Artificial Intelligence

Learning - 15

## Special Cases in Decision Trees

- Attribute  $A$  is numeric.
  - Find best  $A \leq v$  test. Requires sorting.
  - Or: Discretization. Partition  $A$  into ranges.
- Attribute  $A$  has missing values.
  - Pretend missing is just another value.
  - Or: Ignore missing values. Split examples with missing values across branches.
- Attribute  $A$  has many discrete values.
  - Find best  $A = v$  test. Forms binary tree.
  - Or: Partition values into subsets.

CS 5233 Artificial Intelligence

Learning - 16

## Pruning Decision Trees

- Why are there errors?
  - Statistical fluctuations.
  - Examples might have noise and/or outliers.
  - DT approximates decision boundary.
- Results in overfitting at lower levels of DT
- Pruning
  - Prepruning: Avoid creation of subtrees based on number of examples or attribute relevance.
  - Postpruning: Create overfitting DT and substitute subtrees with leaves if estimated error is reduced.

CS 5233 Artificial Intelligence

Learning - 17

## Estimating Error

- Use a “validation” set of examples.  
(training set, validation set, test set should be disjoint)
- Minimum Description Length principle  
(minimize size of tree and minimize size of errors)
- Add some error to each leaf (C4.5).
  - Suppose a leaf has  $e$  errors on  $n$  examples.
  - Find 75% confidence interval using binomial dist.
  - Estimate true error as upper limit of interval.

CS 5233 Artificial Intelligence

Learning – 18

## Algorithm for Pruning Decision Trees

**Prune\_DT**( $N$ : node,  $examples$ )

1.  $leaferr \leftarrow$  number of  $examples \neq N.class$
2. increase  $leaferr$  if  $examples$  were training set
3. **if**  $N$  is a leaf **then return**  $leaferr$
4.  $treeerr \leftarrow 0$
5. **for** each value  $v_j$  of  $N.test$
6.      $examples_j \leftarrow examples$  with  $N.test = v_j$
7.      $suberr \leftarrow$  **Prune\_DT**( $N.branch_j$ ,  $examples_j$ )
8.      $treeerr \leftarrow treeerr + suberr$
9. **if**  $leaferr < treeerr$
10.    **then** make  $N$  a leaf; **return**  $leaferr$
11. **else return**  $treeerr$

CS 5233 Artificial Intelligence

Learning – 19

## Ensemble Learning

20

### Definition

- There are many algorithms for learning a single hypothesis.
- *Ensemble learning* will learn and combine a collection of hypotheses by running the algorithm on different training sets.
- *Bagging* (briefly mentioned in the book) runs a learning algorithm on repeated subsamples of the training set.
- If there are  $n$  examples, then a subsample of  $n$  examples is generated by sampling with replacement.
- On a test example, each hypothesis casts 1 vote for the class it predicts.

CS 5233 Artificial Intelligence

Learning – 20

### Boosting

- In *boosting*, the hypotheses are learned in sequence.
- Both hypotheses and examples have weights with different purposes.
- After each hypothesis is learned, its weight is based on its error rate, and the weights of the training examples (initially all equal) are also modified.
- On a test example, when each hypothesis predicts a class, its weight is the size of its vote. The ensemble predicts the class with the highest vote.

CS 5233 Artificial Intelligence

Learning – 21

### Example Boosting Algorithm

**AdaBoost**( $examples$ ,  $algorithm$ ,  $iterations$ )

1.  $n \leftarrow$  number of examples
2. initialize weights  $w[1 \dots n]$  to  $1/n$
3. **for**  $i$  **from** 1 **to**  $iterations$
4.     $h[i] \leftarrow algorithm(examples)$
5.     $error \leftarrow$  (sum of exs. misclassified by  $h[i]$ ) /  $n$
6.    **for**  $j$  **from** 1 **to**  $n$
7.      **if**  $h[i]$  is correct on example  $j$
8.        **then**  $w[j] \leftarrow w[j] * error / (1 - error)$
9.    normalize  $w[1 \dots n]$  so it sums to 1
10.   weight of  $h[i] \leftarrow \log((1 - error) / error)$
11. **return**  $h[1 \dots iterations]$  and their weights

CS 5233 Artificial Intelligence

Learning – 22

### Example Run of AdaBoost

Using the 14 examples as a training set:

- The hypothesis  $\text{windy} = \text{false} \leftrightarrow \text{class} = \text{pos}$  is wrong on 5 of the 14 examples.
- The weights of the correctly classified examples are multiplied by  $5/9$ , then all examples are multiplied by  $14/10$  so they sum up to 1 again.
- This hypothesis has a weight of  $\log(9/5)$ .
- Note that after weight updating, the sum of the correctly classified examples equals the sum of the incorrectly classified examples.

CS 5233 Artificial Intelligence

Learning – 23

### Example Run of AdaBoost, Continued

- The next hypothesis must be different from the previous one to have error less than  $1/2$ .
- Now the hypothesis  $\text{outlook} = \text{overcast} \leftrightarrow \text{class} = \text{pos}$  has an error rate of  $29/90 \approx 0.322$
- The weights of the correctly classified examples are multiplied times  $29/61 \approx 0.475$ , then all examples are multiplied by  $90/58 \approx 1.55$  so they sum up to 1 again.
- This hypothesis has a weight of  $\log(61/29)$ .

CS 5233 Artificial Intelligence

Learning – 24