Inductive Learning or Concept Learning

All learning can be seen as learning the representation of a function.

Inductive learning: system tries to induce a “general rule” from a set of observed instances.

Supervised learning: learning algorithm is given the correct value of the function for particular inputs, and changes its representation of the function to try to match the information provided by the feedback.

An example is a pair $(x, f(x))$, where $x$ is the input and $f(x)$ is the output of the function applied to $x$.

Example: Work or Play?

<table>
<thead>
<tr>
<th>outlook</th>
<th>temp</th>
<th>humidity</th>
<th>windy</th>
<th>Saturday plan</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>cs472</td>
</tr>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>true</td>
<td>cs472</td>
</tr>
<tr>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>soccer</td>
</tr>
<tr>
<td>rain</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>soccer</td>
</tr>
<tr>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>soccer</td>
</tr>
<tr>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>true</td>
<td>cs472</td>
</tr>
</tbody>
</table>

- Each input observation, $x$, is a Saturday, described by the features outlook, temp, humidity, windy
- The target concept, $f$: day $\rightarrow$ \{soccer, cs472\}

Classification Tasks

Learning a discrete-valued function is called classification.

Steering a vehicle: image in windshield $\rightarrow$ direction to turn the wheel

Medical diagnosis: patient symptoms $\rightarrow$ has disease/ does not have disease

Forensic hair comparison: image of two hairs $\rightarrow$ match or not

Stock market prediction: closing price of last few days $\rightarrow$ market will go up or down tomorrow

Noun phrase coreference: description of two noun phrases in a document $\rightarrow$ do they refer to the same real world entity

Building Classifiers

1. Learn about the domain, write a program that maps inputs to outputs (eg., rule-based medical diagnosis systems).

2. Automate the process using data in the form of observations $(x_i, f(x_i))$.
   - cholesterol=170,bp=170/95,... $\rightarrow$ heart disease = N
   - cholesterol=250,bp=170/95,... $\rightarrow$ heart disease = Y
Inductive Learning

Given: collection of examples

Return: a function $h$ (hypothesis) that approximates $f$ (target concept).

OR

Given: a universe of objects described by a collection of attributes each labeled with one of a discrete number of classes

Return: a classification “rule” that can determine the class of any object from its attributes’ values

Inductive learning hypothesis: any hypothesis found to approximate the target function well over a sufficiently large set of training examples will also approximate the target function well over any other unobserved examples.

Assumptions for Inductive Learning Algorithms:
- The training sample represents the population
- The input features permit discrimination

$k$-nearest neighbor

Also called instance-based Learning; case-based learning.

$A$: set of features/attributes, $A_1, \ldots, A_n$ that describe the problem

$x = x_{a_1}x_{a_2} \ldots x_{a_n}$, where $x_{a_i}$ is the value of feature $A_i$ in example $x$

$f(x) : x \rightarrow c \epsilon C = \{c_1, \ldots, c_m\}$

The case base is the set of training examples $(x_1, f(x_1)), (x_2, f(x_2)), \ldots$
**k-nearest neighbor algorithm for computing** $f(x)$:

1. Compare new example, $x$, to each case, $y$, in the case base and calculate for each pair:

   $sim(x, y) = \sum_{i=1}^{n} match(x_{a_i}, y_{a_i})$

   where $match(a, b)$ is a function that returns 1 if $a$ and $b$ are equal and 0 otherwise.

2. Let $R$ = the top $k$ cases ranked according to $sim$

3. Return as $f(x)$ the class, $c$, that wins the majority vote among $f(R_1), f(R_2), \ldots, f(R_{|k|})$. Handle ties randomly.

---

**Types of Attributes**

1. Symbolic (nominal) – $EyeColor \in \{brown, blue, green\}$
2. Boolean – $anemic \in \{TRUE, FALSE\}$
3. Numeric (Integer, Real) – $age \in [0, 105]$

How do we compute the similarity between $EyeColor = brown$ and $EyeColor = green$?

---

**Example of case retrieval for k-nn**

<table>
<thead>
<tr>
<th>outlook</th>
<th>temp</th>
<th>humidity</th>
<th>windy</th>
<th>plan</th>
<th>sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>cs472</td>
<td></td>
</tr>
<tr>
<td>sunny</td>
<td>hot</td>
<td>high</td>
<td>true</td>
<td>cs472</td>
<td></td>
</tr>
<tr>
<td>overcast</td>
<td>hot</td>
<td>high</td>
<td>false</td>
<td>soccer</td>
<td></td>
</tr>
<tr>
<td>overcast</td>
<td>mild</td>
<td>normal</td>
<td>true</td>
<td>football</td>
<td></td>
</tr>
<tr>
<td>rain</td>
<td>mild</td>
<td>high</td>
<td>false</td>
<td>soccer</td>
<td></td>
</tr>
<tr>
<td>rain</td>
<td>cool</td>
<td>normal</td>
<td>false</td>
<td>soccer</td>
<td></td>
</tr>
</tbody>
</table>

$A$: outlook, temp, humidity, windy

$k = 1$, $C = \{soccer, cs472 football\}$

*test case*: $X = sunny$ cool high false
Advantages and Disadvantages

What constitutes the concept description?

Slide CS472 – Machine Learning 16

Poisonous Mushroom Decision Tree

Concept description: decision trees

Slide CS472 – Machine Learning 17

Another Poisonous Mushroom Decision Tree?

Finding a Decision Tree

Goal: find the best decision tree
where best means the smallest tree consistent with data

Ockham’s Razor: all other things being equal, choose the simplest

Problem: goal is computationally intractable
Solution: use heuristic search

Slide CS472 – Machine Learning 18

Slide CS472 – Machine Learning 19
Top Down Induction of Decision Trees

If all instances from same class
then tree is leaf with that class name
else

*pick test for decision node*

partition instances by test outcome
construct one branch for each possible outcome
build subtrees recursively

---

Example

CS Major Database

<table>
<thead>
<tr>
<th>Height</th>
<th>Eyes</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>short</td>
<td>brown</td>
<td>hacker</td>
</tr>
<tr>
<td>tall</td>
<td>blue</td>
<td>theoretician</td>
</tr>
<tr>
<td>short</td>
<td>brown</td>
<td>hacker</td>
</tr>
<tr>
<td>tall</td>
<td>blue</td>
<td>theoretician</td>
</tr>
</tbody>
</table>

---

A Concept Learning Task

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Wind</th>
<th>Play-Tennis?</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

---

Characteristics of Tests

Let $|P| = 20$, $|N| = 20$

A Boolean test splits the data into two subsets, $U_1$ and $U_2$
The best test: $U_1 = P$ and $U_2 = N$
The worst test: $U_1 = \frac{1}{2}P + \frac{1}{2}N$ and $U_2 = \frac{1}{2}P + \frac{1}{2}N$
**Information Gain**

average disorder =

\[ \sum_{b=1}^{n_{branches}} \frac{n_b}{n_t} \cdot Disorder(b) \]

average disorder =

\[ \sum_{b=1}^{n_{branches}} \frac{n_b}{n_t} \cdot \left( \sum_{c} - \frac{n_{bc}}{n_b} \log_2 \left( \frac{n_{bc}}{n_b} \right) \right) \]

\( n_b \) is the number of instances in branch \( b \)
\( n_t \) is the total number of instances
\( n_{bc} \) is the number of instances in branch \( b \) of class \( c \)

---

**Disorder Term**

Disorder = \( (\sum_{c} \frac{n_{bc}}{n_b} \log_2 \left( \frac{n_{bc}}{n_b} \right)) \)

Average disorder =

\[ \sum_{b=1}^{n_{branches}} \frac{n_b}{n_t} \cdot disorder(b) \]

---

**Calculation for Attribute Humidity**

<table>
<thead>
<tr>
<th>branch</th>
<th>value</th>
<th>( n_bp )</th>
<th>( n_{bn} )</th>
<th>disorder</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>high</td>
<td>3</td>
<td>4</td>
<td>.99</td>
</tr>
<tr>
<td>2</td>
<td>normal</td>
<td>6</td>
<td>1</td>
<td>.58</td>
</tr>
</tbody>
</table>

Disorder(high) = \(-\frac{3}{7}\log_2(\frac{3}{7}) - \frac{4}{7}\log_2(\frac{4}{7}) = .99\)
Disorder(normal) = \(-\frac{6}{7}\log_2(\frac{6}{7}) - \frac{1}{7}\log_2(\frac{1}{7}) = .58\)

Average Disorder of Humidity = \[ \frac{7}{14} \text{Disorder(high)} + \frac{7}{14} \text{Disorder(normal)} = \]
\[ \frac{7}{14}(.99) + \frac{7}{14}(.58) = .79 \]

---

**Selection of Attribute**

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Average Disorder</th>
</tr>
</thead>
<tbody>
<tr>
<td>outlook</td>
<td>0.69</td>
</tr>
<tr>
<td>temperature</td>
<td>0.91</td>
</tr>
<tr>
<td>humidity</td>
<td>0.79</td>
</tr>
<tr>
<td>windy</td>
<td>0.89</td>
</tr>
</tbody>
</table>
Information Gain and Entropy

- $S$ is a sample of training examples
- $p$ is the proportion of positive examples in $S$
- $n$ is the proportion of negative examples in $S$
- Entropy (our Disorder) measures the impurity of $S$

$$\text{Entropy}(S) \equiv -p \log_2 p - n \log_2 n$$

Information Gain measures the expected reduction in entropy caused by partitioning the examples according to attribute $A$.

$$\text{Gain}(S, A) = \text{Entropy}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Entropy}(S_v)$$

Decision Trees

Goal: Construct a decision tree that agrees (is consistent) with the training set.

Trivial solution: construct a decision tree that has one path to a leaf for every example.

Problem with trivial solution?

Non-trivial solution: find a concise decision tree that agrees with the training data.

Appropriate Problems for Decision Tree Learning

- Instances represented by attribute-value pairs
- Target function has a discrete number of output values
- Disjunctive descriptions may be required