Machine Learning

- A completely different way to have an agent acquire the appropriate abilities to solve a particular goal is via *machine learning.*
What is Machine Learning?
- Programs that get better with experience given a task and some performance measure.
  - Learning to classify news articles
  - Learning to recognize spoken words
  - Learning to play board games
  - Learning to navigate a virtual world

Usually involves some sort of inductive reasoning step.

Read Chaps. 17 & 26 in Alex’ Book
Inductive Reasoning

- Deductive reasoning (rule based reasoning)
  - From the general to the specific
- Inductive reasoning
  - From the specific to the general
Example

- Facts: every time you see a swan you notice that the swan is white.
- Inductive step: you infer that all swans are white.

Inference is the act or process of drawing a conclusion based solely on what one already knows.
Observation

- Deduction is “truth preserving”
  - If the rules employed in the deductive reasoning process are sound, then, what holds in the theory will hold for the deduced facts.

- Induction is NOT “truth preserving”
  - It is more of a statistical argument
  - The more swans you see that are white, the more probable it is that all swans are white. But this does not exclude the existence of black swans.
Observation

$D \equiv$ observations
$X \equiv$ universe of all swans
Different Styles of Machine Learning

- **Supervised Learning**
  - The learning needs explicit examples of the concept to be learned (e.g. white swans...)

- **Unsupervised Learning**
  - The learner discovers autonomously any structure in the domain that might represent an interesting concept
Knowledge - Representing what has been learned

- **Symbolic Learners** (transparent models)
  - If-then-else rules
  - Decision trees
  - Association rules

- **Sub-Symbolic Learners** (non-transparent models)
  - Neural Networks
  - Clustering (Self-Organizing Maps, k-Means)
  - Support Vector Machines
Why Learning?

- Scripting works well if there is a well understood relationship between the input (senses) and the actions to be taken.

- Learning works well where no such clear relationship exists:
  - Perhaps there are too many special cases to consider.
  - Perhaps there is a non-linear numerical relationship between the input and the output that is difficult to characterize.

- Learning can be adaptive…online learning where the agent constantly evaluates its actions and adjusts its acquired knowledge:
  - Very difficult to achieve in scripting.
Decision Trees

- Learn from labeled observations - supervised learning
- Represent the knowledge learned in form of a tree

Example: learning when to play tennis.
- Examples/observations are days with their observed characteristics and whether we played tennis or not
### Play Tennis Example

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Windy</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>True</td>
<td>No</td>
</tr>
<tr>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>High</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Cool</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Cool</td>
<td>Normal</td>
<td>True</td>
<td>No</td>
</tr>
<tr>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>True</td>
<td>Yes</td>
</tr>
<tr>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>True</td>
<td>Yes</td>
</tr>
<tr>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>High</td>
<td>True</td>
<td>No</td>
</tr>
</tbody>
</table>

![Decision Tree for Play Tennis Example](Image)
Decision Tree Learning

Facts or Observations

Induction

Theory
A DT uses the attributes of an observation table as nodes and the attribute values as links. All attribute values of a particular attribute need to be represented as links. The target attribute is special - its values show up as leaf nodes in the DT.
Interpreting a DT

Each path from the root of the DT to a leaf can be interpreted as a decision rule.

IF Outlook = Sunny AND Humidity = Normal THEN Playtennis = Yes
IF Outlook = Overcast THEN Playtennis = Yes
IF Outlook = Rain AND Wind = Strong THEN Playtennis = No
**Explanation:** the DT summarizes (explains) all the observations in the table perfectly ⇒ 100% Accuracy

**Prediction:** once we have a DT (or model) we can use it to make predictions on observations that are not in the original training table, consider:

Outlook = Sunny, Temperature = Mild, Humidity = Normal, Windy = False, Playtennis = ?
Constructing DTs

- How do we choose the attributes and the order in which they appear in a DT?
  - Recursive partitioning of the original data table
  - Heuristic - each generated partition has to be “less random” (entropy reduction) than previously generated partitions
S is a sample of training examples

\( p^+ \) is the proportion of positive examples in \( S \)

\( p^- \) is the proportion of negative examples in \( S \)

Entropy measures the impurity (randomness) of \( S \)

\[
\text{Entropy}(S) = \text{Entropy}([9+, 5-]) = .94
\]
## Partitioning the Data Set

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Windy</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>True</td>
<td>No</td>
</tr>
<tr>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>True</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Windy</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>True</td>
<td>Yes</td>
</tr>
<tr>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>True</td>
<td>Yes</td>
</tr>
<tr>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Windy</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>High</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Cool</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Cool</td>
<td>Normal</td>
<td>True</td>
<td>No</td>
</tr>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>High</td>
<td>True</td>
<td>No</td>
</tr>
</tbody>
</table>

E = 0.97  
E = 0  
Average Entropy = 0.64
Partitioning in Action

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Windy</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>True</td>
<td>Yes</td>
</tr>
<tr>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>High</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Cool</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Cool</td>
<td>Normal</td>
<td>True</td>
<td>No</td>
</tr>
<tr>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>True</td>
<td>Yes</td>
</tr>
<tr>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>False</td>
<td>No</td>
</tr>
<tr>
<td>Rainy</td>
<td>Cool</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>True</td>
<td>Yes</td>
</tr>
<tr>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>True</td>
<td>Yes</td>
</tr>
<tr>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>False</td>
<td>Yes</td>
</tr>
<tr>
<td>Rainy</td>
<td>Mild</td>
<td>High</td>
<td>True</td>
<td>No</td>
</tr>
</tbody>
</table>

E = .640

E = .789

E = .892

E = .911
Recursive Partitioning

\[
\text{Partition}(\text{Examples, TargetAttribute, Attributes}) \]

\[
\text{Examples are the training examples. TargetAttribute is a binary (+/-) categorical dependent variable and Attributes is the list of independent variables which are available for testing at this point. This function returns a decision tree.}
\]

\begin{itemize}
  \item Create a Root node for the tree.
  \item If all Examples are positive then return Root as a leaf node with label = +.
  \item Else if all Examples are negative then return Root as a leaf node with label = -.
  \item Else if Attributes is empty then return Root as a leaf node with label = most common value of TargetAttribute in Examples.
  \item Otherwise
    \begin{itemize}
      \item \( A := \) the attribute from Attributes that reduces entropy the most on the Examples.
      \item \( Root := A \)
      \item \( \text{For each } v \in \text{values}(A) \)
        \begin{itemize}
          \item Add a new branch below the Root node with value \( A = v \)
          \item Let \( \text{Examples}_v \) be the subset of Examples where \( A = v \)
          \item If \( \text{Examples}_v \) is empty then add new leaf node to branch with label = most common value of TargetAttribute in Examples.
          \item Else add new subtree to branch
            \begin{itemize}
              \item \( \text{Partition}(\text{Examples}_v, \text{TargetAttribute, Attributes} - \{A\}) \)
            \end{itemize}
        \end{itemize}
    \end{itemize}
  \end{itemize}
\end{itemize}

Return \( Root \)