Introduction to ML

Two examples of Learners:

Naïve Bayesian Classifiers
Decision Trees
Why Bayesian learning?

• **Probabilistic learning**: Calculate explicit probabilities for hypothesis, among the most practical approaches to certain types of learning problems.

• **Incremental (online)**: Each training example can incrementally increase/decrease the probability that a hypothesis is correct. Prior knowledge can be combined with observed data.

• **Standard**: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured.
Bayesian classification: the basic idea

- The classification problem may be formalized using *a-posteriori* probabilities:

- \( P(v|X) = \) prob. that the sample tuple \( X = <a_1,\ldots,a_k> \) is of class \( v \).

- E.g. \( P(\text{class}=\text{No} \mid \text{outlook}=\text{sunny},\text{windy}=\text{true},\ldots) \)

- Idea: assign to instance \( X \) the class label \( v \) such that \( P(v|X) \) is maximal
Estimating a-posteriori probabilities

- **Bayes theorem:**
  \[ P(v|X) = \frac{P(X|v) \cdot P(v)}{P(X)} \]
- \( P(X) \) is constant for all classes
- \( P(v) = \) relative freq of class vs all samples
- \( v \) such that \( P(v|X) \) is maximum = \( v \) such that \( P(X|v) \cdot P(v) \) is maximum
- Problem: computing \( P(X|v) = P(a_1, \ldots, a_k | v) \) is unfeasible (requires too many probabilities \( 2 \times 2^k \! \))
- Practical difficulty: required initial knowledge of many probabilities, significant computational costs
Naïve Bayesian Classification

- Naïve assumption: attribute independence
  \[ P(a_1,\ldots,a_k|v) = P(a_1|v) \cdot \ldots \cdot P(a_k|v) \]

- If i-th attribute is categorical (discrete-valued):
P(a_i|v) is estimated as the relative freq of samples having value a_i as i-th attribute in class v

- If i-th attribute is continuous:
  1) it could be discretized and treated as categorical;
  2) or \( P(a_i|v) \) is estimated assuming it has Gaussian (normal) distribution – requires only mean and variance

- Computationally easy in both cases
## The Play Tennis Dataset

**[Day, Outlook, Temp, Humidity, Wind, **PlayTennis**]**

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

Predict the target value (yes or no) of the target concept PlayTennis for the new instance

Outlook = sunny,
Temperature = cool,
Humidity = high,
Wind = strong
Naïve Bayes: PlayTennis Example

\( v_{NB} \) is the target value output by the Naïve Bayes classifier

\[
v_{NB} = \arg\max_{v_j \in \{\text{yes}, \text{no}\}} P(v_j) \prod_i P(a_i | v_j)
\]

\[
= \arg\max_{v_j \in \{\text{yes}, \text{no}\}} P(v_j) \cdot P(\text{outlook}=\text{sunny} | v_j) \cdot P(\text{Temperature}=\text{cool} | v_j)
\]

\[
\quad \quad \quad \cdot P(\text{Humidity}=\text{high} | v_j) \cdot P(\text{Wind}=\text{strong} | v_j)
\]

Prior Probabilities

Conditional Probabilities
Estimating Probabilities

Prior Probabilities:
Probabilities of different target values estimated from frequencies over 14 training examples

\[ P(\text{PlayTennis} = \text{yes}) = \frac{9}{14} = .64 \]
\[ P(\text{PlayTennis} = \text{no}) = \frac{5}{14} = .36 \]

Conditional Probabilities:
Similarly, can estimate conditional probabilities. For example, those for \( \text{Wind} = \text{strong} \) are:

\[ P(\text{Wind} = \text{strong}|\text{PlayTennis} = \text{yes}) = \frac{3}{9} = .33 \]
\[ P(\text{Wind} = \text{strong}|\text{PlayTennis} = \text{no}) = \frac{3}{5} = .60 \]
The Result

- Using these and similar probability estimates for remaining attribute values, $v_{NB}$ can be calculated as follows

$$P(\text{yes}) \times P(\text{sunny}|\text{yes}) \times P(\text{cool}|\text{yes}) \times P(\text{high}|\text{yes}) \times P(\text{strong}|\text{yes}) = .0053$$

$$P(\text{no}) \times P(\text{sunny}|\text{no}) \times P(\text{cool}|\text{no}) \times P(\text{high}|\text{no}) \times P(\text{strong}|\text{no}) = .0206$$

- Thus, the naïve Bayes classifier assigns the target value $\text{PlayTennis} = \text{no}$ to this new instance
Normalizing class probabilities

- By normalizing the above quantities to sum to one, we can calculate the conditional probability that the target value is \textit{no}, given the observed attribute values.

- For the current example, this probability is

\[
\frac{.0206}{.0206 + .0053} = .795
\]
An Approach to Represent Arbitrary Text Documents

• Given a text document, we define
  – an attribute for each word position in the document
  – the value of that attribute to be the English word found in that position

• thus, this paragraph beginning with sentence “Given a text document …,” would be described by 74 attribute values, corresponding to the 74 word positions

• value of the first attribute is the word “Given” the value of the second attribute is “a” etc.
Reducing Number of Probability Terms

• Assume positional independence

  - Probability of encountering a specific word $w_k$ is independent of the specific word position being encountered ($a_{23}$ versus $a_{95}$)

  - This amounts to assuming that attributes are independent and identically distributed

  - $P(a_i = w_k | v_j) = P(a_m = w_k | v_j)$ for all $i,j,k,m$
Electronic Newsgroups considered in Text Classification Experiment

• Problem of classifying news articles
  – 20 electronic newsgroups (usenet) considered
    • comp.graphics, alt.atheism, etc

• Target classification
  – Name of newsgroup in which article appeared
    – Task is one of Newsgroup Posting Service that learns to assign documents to appropriate newsgroup
Experiment Dataset

• Data Set
  - 1,000 articles collected from each newsgroup, forming data set of 20,000 documents

• Vocabulary
  - 100 most frequent words were removed (the, of, ...)
  - any word occurring fewer than 3 times was removed
  - resulting Vocabulary consisted of 38,500 words

• Naïve Bayes was applied using
  - 2/3 of these 20,000 documents as training examples
  - performance measured over remaining 1/3
Text Classification Experimental Results (Joachims 1996)

• Accuracy achieved by the program was 89%

• Random guessing would yield 5% accuracy

• Variant of Naïve Bayes
  − NewsWeeder system
    • Training: user rates some news articles as interesting
    • Based on user profile Newsweeder then suggests subsequent articles of interest to user
    • NewsWeeder suggests top 10% of its automatically rated articles each day
    • Result: 59% of articles presented were interesting as opposed to 16% in overall pool
Summary

• The naïve Bayes classifier is useful in many practical applications.
  - Called “naïve” because it incorporates the simplifying assumption that attribute values are conditionally independent, given the classification of the instance.
• Suitable for incremental (online) learning

• In some domains performance is comparable to that of neural network and decision tree learning

• Was successfully applied to text categorization problems
Decision Trees

• Decision Trees classify instances by sorting them down the tree itself
• Leaf nodes provide the classification of the instance
• Each node in the tree specify a test of some attribute (feature value) of the instance
• Each branch descending from the node correspond to one possible value of the attribute
A Sample Tree: Playing tennis

- **Outlook**
  - Sunny
  - Overcast
  - Rainy
  - **Humidity**
    - High
      - No
    - Normal
      - Yes
  - Wind
    - Strong
    - Weak
Representation Power

• Typically
  - Examples presented as an array of features
  - Each node tests one single attribute
  - One child node for each possible value of the attribute
  - The target function has discrete output values
  - Robust methodology in presence of noise in the training data.
Induction of Decision Trees

• The core algorithm uses a greedy search through the space of possible trees

• The technique is a top-down search starting at the root node and ending at the leaf nodes.
  – Start with full dataset
  – Apply a statistical test to evaluate how good attributes are in partitioning the examples (dividing examples into groups according to their class)
  – For each outcome of the test create a child node
  – Move examples to children according to the outcome of test
  – Repeat procedure for each node
Evaluating the Attributes

- The central choice in top-down induction of decision trees is selecting which attribute to test at each node.

- We need to select the attribute that is most useful in classification.

- *Information Gain* is a statistical property that measures how well a given attribute separates the examples according to the target classification.

- *Information Gain* is based on the concept of entropy.
Entropy function

• Entropy characterizes the purity/impurity of an arbitrary collection against a target concept.

• The entropy function is defined as:

  \[ \text{Entropy} (S) = - \sum p_i \log_2 p_i \]

  \( S \): is the set of examples
  
  \( p_i \): is the ratio of examples belonging to class \( i \)
Entropy of a 2 class problem

Entropy \( S \) = - p_a \log_2 p_a - p_b \log_2 p_b

Possible Values:

- \( E (S) = 0 \) if all the members belong to the same class
- \( E (S) = 1 \) if each class is represented in equal number
- \( E (S) = (0,1) \) if the classes are unequally represented
Information Gain

- Information gain is defined as the expected reduction in entropy caused by partitioning the examples according to the given attribute.

\[
\text{Gain}(S, A) = \text{Entropy}(S) - \sum \frac{|S_v|}{|S|} \text{Entropy}(S_v)
\]

- S: is the given set of examples (training set)
- A: is the attribute we are testing
- \(S_v\): is the subset of elements of S having value v for attribute A
Example 1

Assume S has 9 + and 5 - examples; partition according to **Wind** or **Humidity** attribute.

\[
\begin{align*}
\text{Gain(S, Humidity)} &= 0.940 - \frac{7}{14} \cdot 0.985 - \frac{7}{14} \cdot 0.592 \\
&= 0.151
\end{align*}
\]

\[
\begin{align*}
\text{Gain(S, Wind)} &= 0.940 - \frac{8}{14} \cdot 0.811 - \frac{6}{14} \cdot 1.0 \\
&= 0.048
\end{align*}
\]
Example 2

Assuming Outlook is chosen, the partitioning continues until the leaf nodes are reached.
Search in TDIDT
(top-down induction of decision trees)

- Hypothesis space $H = \text{set of all trees}$
- $H$ is searched in a hill-climbing fashion, from simple to complex
Inductive Bias in TDIDT

What is the policy by which TDIDT generalizes from the observations? (aka what is its inductive bias?)

- Preference for short trees, and for those with high information gain attributes near the root

- Occam’s razor: prefer the shortest (simplest) hypothesis that fits the data (1320)
  - A 5-node tree fitting the data is less likely to be a statistical coincidence than a 500-node tree
Overfitting the Data

• Overfitting:
  - keep improving a model, making it better and better on training set by making it more complicated ...
  - increases risk of modelling noise and coincidences in the data set
  - may actually harm predictive power of theory on unseen cases

• Example: fitting a curve with too many parameters:
Effect of Overfitting

- Typical phenomenon when overfitting
  - training accuracy keeps increasing
  - accuracy on unseen test set starts decreasing

![Graph showing the effect of overfitting](chart.png)
Avoiding Overfitting in DTs

• Option 1
  - stop adding nodes to tree when overfitting starts occurring
  - need stopping criterion

• Option 2
  - don’t bother about overfitting when growing the tree
  - after the tree has been built, start pruning it
Stopping Criteria

• How do we know when overfitting starts?
  – use a *validation set*: separated set extracted from the training set
    • when accuracy goes down on validation set: stop adding nodes to this branch
    • Evaluate the generalization accuracy on test set
  – use some statistical test
    • significance test: e.g., is the change in class distribution still significant? ($\chi^2$-test)
    • ...
Pluses of Decision Trees

- Easy to generate; simple algorithm
- Easy to read small trees; can be converted to simple rules
- Fast and easy to construct
- Good performance on many tasks
- A wide variety of problems can be recast as classification problems
Minuses of Decision Trees

- Not always sufficient to learn complex concepts
- Can be hard to understand if the trees are large
- Some problems with continuously-valued attributes or classes may not be easily discretized
References

• Chapter 3 of “Machine Learning”, T. Mitchell