CS2220: Introduction to Computational Biology
Lecture 2: Essence of Knowledge Discovery

Limsoon Wong
21 January 2010

Outline

- Overview of Supervised Learning
  - Decision Trees

- Decision Trees Ensembles
  - Bagging
  - CS4

- Other Methods
  - K-Nearest Neighbour
  - Support Vector Machines
  - Bayesian Approach
  - Hidden Markov Models
Overview of Supervised Learning

Computational Supervised Learning

• Also called classification

• Learn from past experience, and use the learned knowledge to classify new data

• Knowledge learned by intelligent algorithms

• Examples:
  – Clinical diagnosis for patients
  – Cell type classification
Data

• Classification application involves > 1 class of data. E.g.,
  – Normal vs disease cells for a diagnosis problem

• Training data is a set of instances (samples, points) with known class labels

• Test data is a set of instances whose class labels are to be predicted

Typical Notations

• Training data
  \{ (x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m) \}
  where \( x_j \) are n-dimensional vectors
  and \( y_j \) are from a discrete space \( Y \).
  E.g., \( Y = \{ \text{normal, disease} \} \)

• Test data
  \{ (u_1, ?), (u_2, ?), \ldots, (u_k, ?) \}
Process

Training data: $X$ \[ f(X) \] Class labels $Y$

A classifier, a mapping, a hypothesis

Test data: $U$ \[ f(U) \] Predicted class labels

Relational Representation of Gene Expression Data

$n$ features (order of 1000)

$m$ samples

<table>
<thead>
<tr>
<th>gene_1</th>
<th>gene_2</th>
<th>gene_3</th>
<th>gene_4</th>
<th>\cdots</th>
<th>gene_n</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_{11}$</td>
<td>$x_{12}$</td>
<td>$x_{13}$</td>
<td>$x_{14}$</td>
<td>\cdots</td>
<td>$x_{1n}$</td>
<td>$P$</td>
</tr>
<tr>
<td>$x_{21}$</td>
<td>$x_{22}$</td>
<td>$x_{23}$</td>
<td>$x_{24}$</td>
<td>\cdots</td>
<td>$x_{2n}$</td>
<td>$N$</td>
</tr>
<tr>
<td>$x_{31}$</td>
<td>$x_{32}$</td>
<td>$x_{33}$</td>
<td>$x_{34}$</td>
<td>\cdots</td>
<td>$x_{3n}$</td>
<td>$P$</td>
</tr>
<tr>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
<td>\cdots</td>
</tr>
<tr>
<td>$x_{m1}$</td>
<td>$x_{m2}$</td>
<td>$x_{m3}$</td>
<td>$x_{m4}$</td>
<td>\cdots</td>
<td>$x_{mn}$</td>
<td>$N$</td>
</tr>
</tbody>
</table>
Features (aka Attributes)

- **Categorical features**
  - color = \{red, blue, green\}

- **Continuous or numerical features**
  - gene expression
  - age
  - blood pressure

- **Discretization**

### An Example

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Windy</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>75</td>
<td>70</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Sunny</td>
<td>80</td>
<td>90</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>85</td>
<td>85</td>
<td>false</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>72</td>
<td>95</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>69</td>
<td>70</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>72</td>
<td>90</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>83</td>
<td>78</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>64</td>
<td>65</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>81</td>
<td>72</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>71</td>
<td>80</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Rain</td>
<td>65</td>
<td>70</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Rain</td>
<td>75</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>68</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>70</td>
<td>96</td>
<td>false</td>
<td>Play</td>
</tr>
</tbody>
</table>
Overall Picture of Supervised Learning

Labelled Data + Algorithms

Biomedical
- Financial
- Government
- Scientific

Decision trees
- Emerging patterns
- SVM
- Neural networks

Classifiers (Medical Doctors)

Evaluation of a Classifier

- Performance on independent blind test data
- K-fold cross validation: Given a dataset, divide it into k even parts, k-1 of them are used for training, and the rest one part treated as test data
- LOOCV, a special case of K-fold CV

- Accuracy, error rate
- False positive rate, false negative rate, sensitivity, specificity, precision
Requirements of Biomedical Classification

- High accuracy/sensitivity/specificity/precision
- High comprehensibility

Importance of Rule-Based Methods

- Systematic selection of a small number of features used for the decision making
  ⇒ Increase the comprehensibility of the knowledge patterns
- C4.5 and CART are two commonly used rule induction algorithms---a.k.a. decision tree induction algorithms
Structure of Decision Trees

- If $x_1 > a_1$ & $x_2 > a_2$, then it's A class
- C4.5, CART, two of the most widely used
- Easy interpretation, but accuracy generally unattractive

Elegance of Decision Trees

Every path from root to a leaf forms a decision rule
Brief History of Decision Trees

CLS (Hunt et al. 1966) --- cost driven

CART (Breiman et al. 1984) --- Gini Index

ID3 (Quinlan, 1986) --- Information-driven

C4.5 (Quinlan, 1993) --- Gain ratio + Pruning ideas

A Simple Dataset

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Windy</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>75</td>
<td>70</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Sunny</td>
<td>80</td>
<td>90</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>85</td>
<td>85</td>
<td>false</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>72</td>
<td>95</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>69</td>
<td>70</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>72</td>
<td>90</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>83</td>
<td>78</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>64</td>
<td>65</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>81</td>
<td>75</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>71</td>
<td>80</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Rain</td>
<td>65</td>
<td>70</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Rain</td>
<td>75</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>68</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>70</td>
<td>96</td>
<td>false</td>
<td>Play</td>
</tr>
</tbody>
</table>

9 Play samples
5 Don’t
A total of 14.
A Decision Tree

- Construction of a tree is equivalent to determination of the root node of the tree and the root node of its sub-trees

Exercise: What is the accuracy of this tree?
Most Discriminatory Feature

• Every feature can be used to partition the training data

• If the partitions contain a pure class of training instances, then this feature is most discriminatory

Example of Partitions

• **Categorical feature**
  – Number of partitions of the training data is equal to the number of values of this feature

• **Numerical feature**
  – Two partitions
<table>
<thead>
<tr>
<th>Instance #</th>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Windy</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sunny</td>
<td>75</td>
<td>70</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>2</td>
<td>Sunny</td>
<td>80</td>
<td>90</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>3</td>
<td>Sunny</td>
<td>85</td>
<td>85</td>
<td>false</td>
<td>Don’t</td>
</tr>
<tr>
<td>4</td>
<td>Sunny</td>
<td>72</td>
<td>95</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>5</td>
<td>Sunny</td>
<td>69</td>
<td>70</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>6</td>
<td>Overcast</td>
<td>72</td>
<td>90</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>7</td>
<td>Overcast</td>
<td>83</td>
<td>78</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>8</td>
<td>Overcast</td>
<td>64</td>
<td>65</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>9</td>
<td>Overcast</td>
<td>81</td>
<td>75</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>10</td>
<td>Rain</td>
<td>71</td>
<td>80</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>11</td>
<td>Rain</td>
<td>65</td>
<td>70</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>12</td>
<td>Rain</td>
<td>75</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>13</td>
<td>Rain</td>
<td>68</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>14</td>
<td>Rain</td>
<td>70</td>
<td>96</td>
<td>false</td>
<td>Play</td>
</tr>
</tbody>
</table>

A categorical feature is partitioned based on its number of possible values:

- **Outlook = sunny**
  - 1, 2, 3, 4, 5
  - P, D, D, D, P

- **Outlook = overcast**
  - 6, 7, 8, 9
  - P, P, P

- **Outlook = rain**
  - 10, 11, 12, 13, 14
  - D, D, P, P, P

**Total 14 training instances**
A numerical feature is generally partitioned by choosing a “cutting point”

Steps of Decision Tree Construction

• Select the “best” feature as the root node of the whole tree

• Partition the dataset into subsets using this feature so that the subsets are as “pure” as possible

• After partition by this feature, select the best feature (wrt the subset of training data) as the root node of this sub-tree

• Recursively, until the partitions become pure or almost pure
Let's Construct a Decision Tree Together

Ask the class to pick root node and construct the tree recursively with them… How good is that tree?

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>windy</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>75</td>
<td>70</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Sunny</td>
<td>80</td>
<td>90</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>85</td>
<td>85</td>
<td>false</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>72</td>
<td>95</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>69</td>
<td>70</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>72</td>
<td>90</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>83</td>
<td>78</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>64</td>
<td>65</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>81</td>
<td>75</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>71</td>
<td>80</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Rain</td>
<td>65</td>
<td>70</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Rain</td>
<td>75</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>68</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>70</td>
<td>96</td>
<td>false</td>
<td>Play</td>
</tr>
</tbody>
</table>

Three Measures to Evaluate Which Feature is Best

- Gini index
- Information gain
- Information gain ratio
Gini Index

Let \( U = \{ C_1, ..., C_k \} \) be all the classes. Suppose we are currently at a node and \( D \) is the set of those samples that have been moved to this node. Let \( f \) be a feature and \( df \) be the value of the feature \( f \) in a sample \( d \). Let \( S \) be a range of values that the feature \( f \) can take. Then the Gini index for \( f \) in \( D \) for the range \( S \) is defined as

\[
gini_f^r(S) = 1 - \sum_{C_i \in U} \left( \frac{|\{ d \in D \mid df \in S \cap C_i \}|}{|D|} \right)^2
\]

The purity of a split of the value range \( S \) of an attribute \( f \) by some split-point into subranges \( S_1 \) and \( S_2 \) is then defined as

\[
gini_f(S_1, S_2) = \sum_{S \subseteq \{S_1, S_2\}} \frac{|\{ d \in D \mid df \in S \}|}{|D|} \cdot gini_f^r(S)
\]

we choose the feature \( f \) and the split-point \( p \) that minimizes \( gini_f(S_1, S_2) \) over all possible alternative features and split-points.

Gini index can be thought of as the expected value of the ratio of the diff of two arbitrary specimens to the mean value of all specimens. Thus the closer it is to 1, the closer you are to the expected “background distribution” of that feature. Conversely, the closer it is to 0, the more “unexpected” the feature is.

\[
gini(S) = \frac{\text{diff of two arbitrary specimen in } S}{\text{mean specimen in } S} = \frac{\text{prob(getting two specimen of diff class in } S)}{\text{prob(getting specimen of some class in } S)}
\]

\[
= \sum_{i} \text{prob(getting specimen of class } i \text{ in } S) \cdot \text{prob(getting specimen of class } j \text{ in } S)^2
\]

\[
= 1 - \sum_{i} \text{prob(getting specimen of class } i \text{ in } S)^2
\]

Gini index can be thought of as the expected value of the ratio of the diff of two arbitrary specimens to the mean value of all specimens. Thus the closer it is to 1, the closer you are to the expected “background distribution” of that feature. Conversely, the closer it is to 0, the more “unexpected” the feature is.
Gini Index of Outlook

<table>
<thead>
<tr>
<th>Outlook</th>
<th>Temp</th>
<th>Humidity</th>
<th>Windy</th>
<th>class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sunny</td>
<td>75</td>
<td>70</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Sunny</td>
<td>80</td>
<td>90</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>85</td>
<td>85</td>
<td>false</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>72</td>
<td>95</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Sunny</td>
<td>69</td>
<td>70</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>72</td>
<td>90</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>83</td>
<td>78</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>64</td>
<td>65</td>
<td>true</td>
<td>Play</td>
</tr>
<tr>
<td>Overcast</td>
<td>83</td>
<td>75</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>71</td>
<td>80</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Rain</td>
<td>65</td>
<td>70</td>
<td>true</td>
<td>Don’t</td>
</tr>
<tr>
<td>Rain</td>
<td>75</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>68</td>
<td>80</td>
<td>false</td>
<td>Play</td>
</tr>
<tr>
<td>Rain</td>
<td>70</td>
<td>96</td>
<td>false</td>
<td>Play</td>
</tr>
</tbody>
</table>

\[
gini_D(S) = 1 - \sum_{C \in \text{class}} \left( \frac{|d \in D \mid d \in C \land d \in f(S)|}{|D|} \right)^2
\]

\[
gini_{D}(S, S_0) = \sum_{S_i \in \mathcal{S}} \left( \frac{|d \in D \mid d \in S \land d \notin S_i|}{|D|} \right) \cdot gini_D(S_i)
\]

- $\text{gini}(\text{Sunny}) = 1 - (2/5)^2 - (3/5)^2 = 0.48$
- $\text{gini}(\text{Overcast}) = 1 - (4/4)^2 - (0/5)^2 = 0$
- $\text{gini}(\text{Rain}) = 1 - (3/5)^2 - (2/5)^2 = 0.48$
- $\text{gini}(\text{Outlook}) = 5/14 \times 0.48 + 4/14 \times 0 + 5/14 \times 0.48 = 0.34$

Characteristics of C4.5/CART Trees

- Single coverage of training data (elegance)
- Divide-and-conquer splitting strategy
- Fragmentation problem $\Rightarrow$ Locally reliable but globally insignificant rules

Missing many globally significant rules; mislead the system
Example Use of Decision Tree Methods: Proteomics

Approaches to Biomarker Discovery

- In prostate and bladder cancers (Adam et al. *Proteomics*, 2001)
- In serum samples to detect breast cancer (Zhang et al. *Clinical Chemistry*, 2002)
- In serum samples to detect ovarian cancer (Petricoin et al. *Lancet*; Li & Rao, *PAKDD* 2004)

Decision Tree Ensembles
Motivating Example

- $h_1, h_2, h_3$ are independent classifiers with accuracy = 60%
- $C_1, C_2$ are the only classes
- $t$ is a test instance in $C_1$
- $h(t) = \arg\max_{C \in \{C_1, C_2\}} \{h_j \in \{h_1, h_2, h_3\} | h_j(t) = C\}$
- Then $\Pr(h(t) = C_1) =$ 
  
  $\Pr(h_1(t)=C_1 & h_2(t)=C_1 & h_3(t)=C_1) + \Pr(h_1(t)=C_1 & h_2(t)=C_1 & h_3(t)=C_2) + \Pr(h_1(t)=C_1 & h_2(t)=C_2 & h_3(t)=C_1) + \Pr(h_1(t)=C_2 & h_2(t)=C_1 & h_3(t)=C_1) = 60\% \times 60\% \times 60\% + 60\% \times 60\% \times 40\% + 60\% \times 40\% \times 60\% + 40\% \times 60\% \times 60\% = 64.8\%$

Bagging

- Proposed by Breiman (1996)
- Also called Bootstrap aggregating
- Make use of randomness injected to training data
Main Ideas

Original training set

50 p + 50 n

Draw 100 samples with replacement

48 p + 52 n
49 p + 51 n
53 p + 47 n

A base inducer such as C4.5

A committee $H$ of classifiers:

$h_1 \quad h_2 \quad \ldots \quad h_k$

Decision Making by Bagging

Given a new test sample $T$

$$\text{bagged}(T) = \arg\max_{C_j \in \mathcal{U}} \{ h_i \in H \mid h_i(T) = C_j \}$$

where $\mathcal{U} = \{C_1, \ldots, C_r\}$

Exercise: What does the above formula mean?
CS4

- CS4: Cascading and Sharing for decision trees
- Doesn't make use of randomness

Main Ideas

Selection of root nodes is in a cascading manner!
Decision Making by CS4

\[ \text{rule}_{1}^{\text{pos}}, \text{rule}_{2}^{\text{pos}}, \ldots, \text{rule}_{k_1}^{\text{pos}}, \]
\[ \text{rule}_{1}^{\text{neg}}, \text{rule}_{2}^{\text{neg}}, \ldots, \text{rule}_{k_2}^{\text{neg}}. \]

\[ \text{Score}^{\text{pos}}(T) = \sum_{i=1}^{k_1} \text{coverage}(\text{rule}_{i}^{\text{pos}}) \]
\[ \text{Score}^{\text{neg}}(T) = \sum_{i=1}^{k_2} \text{coverage}(\text{rule}_{i}^{\text{neg}}) \]

Not equal voting

Summary of Ensemble Classifiers

- Bagging
- Random Forest

AdaBoost.M1

Rules may not be correct when applied to training data

- Randomization Trees
- CS4

Rules correct

Exercise: Describe the 3 decision tree ensemble classifiers not explained in this ppt
Other Machine Learning Approaches

Outline

• K-Nearest Neighbour
• Support Vector Machines
• Bayesian Approach
• Hidden Markov Models

Exercise: Name and describe one other commonly used machine learning method
K-Nearest Neighbours

How kNN Works

• Given a new case

• Find k “nearest” neighbours, i.e., k most similar points in the training data set

• Assign new case to the same class to which most of these neighbours belong

• A common “distance” measure between samples $x$ and $y$ is

$$\sqrt{\sum_f (x[f] - y[f])^2}$$

where $f$ ranges over features of the samples

Exercise: What does the formula above mean?
Some Issues

- Simple to implement
- But need to compare new case against all training cases
  ⇒ May be slow during prediction

- No need to train
- But need to design distance measure properly
  ⇒ May need expert for this

- Can’t explain prediction outcome
  ⇒ Can’t provide a model of the data
Example Use of kNN: **Segmentation of White Lesion Matter in MRI**

- Use kNN to automated segmentation of white matter lesions in cranial MR images
- Rely on info from T1-weighted, inversion recovery, proton density-weighted, T2-weighted, & fluid attenuation inversion recovery scans

![Fig. 3. Classification of a patient with moderate lesion load. (A) FLAIR image, (B) manual segmentation, (C) probability map, (D) segmentations derived from probability map with different thresholds: black: probability \( P = 0 \), blue: \( 0 < P \leq 0.3 \), green: \( 0.3 < P \leq 0.5 \), yellow: \( 0.5 < P \leq 0.8 \), red: \( 0.8 < P \leq 1 \).](image)

Example Use of kNN: **Ovarian Cancer Diagnosis Based on SELDI Proteomic Data**

- Use kNN to diagnose ovarian cancers using proteomic spectra
- Data set is from Petricoin et al., *Lancet* 359:572-577, 2002

![Fig. 1. Minimum, median and maximum of percentages of correct prediction as a function of the number of top-ranked m/z ratios in 50 independent partitions into learning and validation sets.](image)
Example Use of kNN: Prediction of Compound Signature Based on Gene Expr Profiles


- Store gene expression profiles corr to biological responses to exposures to known compounds whose toxicological and pathological endpoints are well characterized

- Use kNN to infer effects of unknown compound based on gene expr profiles induced by it

Support Vector Machines
Basic Idea

(a) Linear separation not possible w/o errors
(b) Better separation by nonlinear surfaces in input space
(c) Nonlinear surface corr to linear surface in feature space.

Map from input to feature space by “kernel” function $\Phi$

$\Rightarrow$ “Linear learning machine” + kernel function as classifier

Linear Learning Machines

- Hyperplane separating the x’s and o’s points is given by $(W \cdot X) + b = 0$, with $(W \cdot X) = \sum_j W[j] X[j]$

$\Rightarrow$ Decision function is $\text{llm}(X) = \text{sign}((W \cdot X) + b)$
Linear Learning Machines

- Solution is a linear combination of training points $X_k$ with labels $Y_k$
  
  $$W[j] = \sum_k \alpha_k Y_k X_k[j],$$
  
  with $\alpha_k > 0$, and $Y_k = \pm 1$

  $$\Rightarrow \text{llm}(X) = \text{sign}(\sum_k \alpha_k Y_k (X_k \cdot X) + b)$$

  “data” appears only in dot product!

Kernel Function

- $\text{llm}(X) = \text{sign}(\sum_k \alpha_k Y_k (X_k \cdot X) + b)$

- $\text{svm}(X) = \text{sign}(\sum_k \alpha_k Y_k (\Phi X_k \cdot \Phi X) + b)$

  $$\Rightarrow \text{svm}(X) = \text{sign}(\sum_k \alpha_k Y_k K(X_k, X) + b)$$
  
  where $K(X_k, X) = (\Phi X_k \cdot \Phi X)$
Kernel Function

- \( \text{svm}(X) = \text{sign}(\sum_k \alpha_k Y_k^* K(X_k, X) + b) \)
  \( \Rightarrow K(A, B) \) can be computed w/o computing \( \Phi \)

- In fact replace it w/ lots of more “powerful” kernels besides \((A \cdot B)\). E.g.,
  - \( K(A, B) = (A \cdot B)^d \)
  - \( K(A, B) = \exp(-||A \cdot B||^2 / (2*\sigma)), ... \)

How SVM Works

- \( \text{svm}(X) = \text{sign}(\sum_k \alpha_k Y_k^* K(X_k, X) + b) \)
- To find \( \alpha_k \) is a quadratic programming problem
  \[
  \max: \sum_k \alpha_k - 0.5 \sum_k \sum_h \alpha_k \alpha_h Y_k^* Y_h^* K(X_k, X_h) \\
  \text{subject to: } \sum_k \alpha_k^* Y_k = 0 \\
  \text{and for all } \alpha_k, C \geq \alpha_k \geq 0 \\
  \]
- To find \( b \), estimate by averaging
  \[
  Y_h - \sum_k \alpha_k Y_k^* K(X_h, X_k) \\
  \text{for all } \alpha_h \geq 0 \\
  \]
Example Use of SVM: Prediction of Protein-Protein Interaction Sites From Sequences

- Koike et al, Protein Engineering Design & Selection 17:165-173, 2004

- Identification of protein-protein interaction sites is impt for mutant design & prediction of protein-protein networks

- Interaction sites were predicted here using SVM & profiles of sequentially/spatially neighbouring residues

Legend: green=TP, white=TN, yellow=FN, red=FP
A: human macrophage migration inhibitory factor
B & C: the binding proteins

Example Use of SVM: Prediction of Gene Function From Gene Expression

- Brown et al., PNAS 91:262-267, 2000

- Use SVM to identify sets of genes w/ a c'mon function based on their expression profiles

- Use SVM to predict functional roles of uncharacterized yeast ORFs based on their expression profiles

Fig. 1: Expression profile of YPL037C compared with the MYOG class of cytoplasmic ribosomal proteins. YPL037C is classified as an rDNA protein by the SVM, but is not included in the class by MYOG. The figure shows the expression profile for YPL037C, along with standard deviation bars for the class of cytoplasmic ribosomal proteins. Ticks along the x-axis represent the beginning of experimental series.
Example Use of SVM: Recognition of Protein Translation Initiation Sites

- Use SVM to recognize protein translation initiation sites from genomic sequences
- Raw data set is same as Liu & Wong, *JBCB* 1:139-168, 2003

Bayesian Approach
Bayes Theorem

\[ P(h|d) = \frac{P(d|h) \times P(h)}{P(d)} \]

- \(P(h)\) = prior prob that hypothesis \(h\) holds
- \(P(d|h)\) = prob of observing data \(d\) given \(h\) holds
- \(P(h|d)\) = posterior prob that \(h\) holds given observed data \(d\)

Bayesian Approach

- Let \(H\) be all possible classes. Given a test instance w/ feature vector \(\{f_1 = v_1, \ldots, f_n = v_n\}\), the most probable classification is given by
  \[ \arg\max_{h_j \in H} P(h_j | f_1 = v_1, \ldots, f_n = v_n) \]
- Using Bayes Theorem, rewrites to
  \[ \arg\max_{h_j \in H} \frac{P(f_1 = v_1, \ldots, f_n = v_n | h_j) \times P(h_j)}{P(f_1 = v_1, \ldots, f_n = v_n)} \]
- Since denominator is independent of \(h_j\), this simplifies to
  \[ \arg\max_{h_j \in H} P(f_1 = v_1, \ldots, f_n = v_n | h_j) \times P(h_j) \]
An Example

Training samples

Prior probability for GREEN = \frac{\text{Number of GREEN objects}}{\text{Total number of objects}} = \frac{40}{60}

Prior probability for RED = \frac{\text{Number of RED objects}}{\text{Total number of objects}} = \frac{20}{60}

A testing instance X

Likelihood of X given GREEN = \frac{\text{Number of GREEN in the vicinity of X}}{\text{Total number of GREEN cases}} = \frac{1}{40}

Likelihood of X given RED = \frac{\text{Number of RED in the vicinity of X}}{\text{Total number of RED cases}} = \frac{3}{20}

Posterior probability of X being GREEN = \frac{\text{Prior probability of GREEN} \times \text{Likelihood of X given GREEN}}{\text{Prior probability of RED} \times \text{Likelihood of X given RED}}

= \frac{\frac{1}{6} \times \frac{1}{40}}{\frac{1}{6} \times \frac{1}{20}}

we classify X as RED since its class membership achieves the largest posterior probability

Source: http://www.statsoft.com/textbook/stnaiveb.html

Copyright 2010 © Limsoon Wong

Naïve Bayes

- But estimating \( P(f_1=v_1, \ldots, f_n=v_n|h_j) \) accurately may not be feasible unless training data set is sufficiently large
- “Solved” by assuming \( f_1, \ldots, f_n \) are conditionally independent of each other
- Then \( \arg\max_{h_j \in H} P(f_1 = v_1, \ldots, f_n = v_n|h_j) \cdot P(h_j) = \arg\max_{h_j \in H} \prod_{i} P(f_i = v_i|h_j) \cdot P(h_j) \)
- where \( P(h_j) \) and \( P(f_i=v|h_j) \) can often be estimated reliably from typical training data set

Exercise: How do you estimate \( P(h_j) \) and \( P(f_i=v|h_j) \)?

Copyright 2010 © Limsoon Wong
Independence vs Conditional Independence

• Independence: \( P(A, B) = P(A) \times P(B) \)
• Conditional Independence: \( P(A, B|C) = P(A|C) \times P(B|C) \)
• Indep does not imply conditional indep
  – Consider tossing a fair coin twice
    • \( A \) is event of getting head in 1st toss
    • \( B \) is event of getting head in 2nd toss
    • \( C \) is event of getting exactly one head
  – Then \( A = \{HT, HH\}, B = \{HH, TH\} \) and \( C = \{HT, TH\} \)
  – \( P(A, B|C) = P(\{HH\}|C) = 0 \)
  – \( P(A|C) = P(A, C)/P(C) = P(\{HT\})/P(C) = (1/4)/(1/2) = 1/2 \)
  – Similarly, \( P(B|C) = 1/2 \)
Example Use of Bayesian: Design of Screens Macromolecular Crystallization


- Xtablization of proteins requires search of expt settings to find right conditions for diffraction-quality xtals

- BMCD is a db of known xtablization conditions

- Use Bayes to determine prob of success of a set of expt conditions based on BMCD

**Figure 1**
Crystallization parameter dependency graph. The graph represents the parameters included in the calculation of the estimated probability of success and their dependencies. A connecting arc from pH to buffer indicates that the probability distribution for the buffer may depend on the value of the pH. The lack of a connecting arc between two parameters reflects conditional independence (the probability distribution for a parameter is independent of the value of the other parameter).

Hidden Markov Models
What is a HMM

- HMM is a stochastic generative model for sequences
- Defined by model parameters
  - finite set of states $S$
  - finite alphabet $A$
  - transition prob matrix $T$
  - emission prob matrix $E$
- Move from state to state according to $T$ while emitting symbols according to $E$

The Order of a HMM

- In $n$th order HMM, $T$ & $E$ depend on all $n$ previous states
- E.g., for 1st order HMM, given emissions $X = x_1, x_2, ..., $ & states $S = s_1, s_2, ...$, the prob of this seq is

$$P_{\text{prob}}(X, S) = \prod_i P_{\text{prob}}(x_i | s_i) = \prod_i E(x_i | s_i) \ast T(s_{i-1} \rightarrow s_i)$$

Copyright 2010 © Limsoon Wong
Using HMM

- Given the model parameters, compute the probability of a particular output sequence. Solved by the forward algorithm

- Given the model parameters, find the most likely sequence of (hidden) states which could have generated a given output sequence. Solved by the Viterbi algorithm

- Given an output sequence, find the most likely set of state transition and output probabilities. Solved by the Baum-Welch algorithm

Exercise: Describe these algorithms

Example: Dishonest Casino

- Casino has two dices:
  - Fair dice
    - $P(i) = 1/6$, $i = 1..6$
  - Loaded dice
    - $P(i) = 1/10$, $i = 1..5$
    - $P(i) = 1/2$, $i = 6$

- Casino switches between fair & loaded die with prob 1/2. Initially, dice is always fair

- Game:
  - You bet $1
  - You roll
  - Casino rolls
  - Highest number wins $2

- Question: Suppose we played 2 games, and the sequence of rolls was 1, 6, 2, 6. Were we likely to have been cheated?
"Visualization" of Dishonest Casino

Emission Matrix

<table>
<thead>
<tr>
<th>E(Fair)</th>
<th>E(Loaded)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/6</td>
<td>1/10</td>
</tr>
</tbody>
</table>

Transition Matrix

<table>
<thead>
<tr>
<th>T(Loaded, Loaded)</th>
<th>T(Loaded, Fair)</th>
<th>T(Fair, Fair)</th>
<th>T(Fair, Loaded)</th>
<th>T(?, Fair)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1.0</td>
</tr>
<tr>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>1/2</td>
<td>0.0</td>
</tr>
</tbody>
</table>

1, 6, 2, 6?
We were probably cheated...

\[
\Pr(X, S = \text{Fair, Fair, Fair, Fair}) = E(\text{Fair}) \times T(?, \text{Fair}) \times E(\text{Fair}) \times T(\text{Fair, Fair}) \times E(\text{Fair}) \times T(\text{Fair, Fair}) \times E(\text{Fair}) \times T(\text{Fair, Fair}) \\
= \frac{1}{6} \times \frac{1}{6} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{6} \times \frac{1}{2} \\
= 9.0451 \times 10^{-5}
\]

\[
\Pr(X, S = \text{Fair, Loaded, Fair, Loaded}) = E(\text{Fair}) \times T(?, \text{Fair}) \times E(\text{Loaded}) \times T(\text{Fair, Loaded}) \times E(\text{Fair}) \times T(\text{Loaded, Fair}) \times E(\text{Loaded}) \times T(\text{Fair, Loaded}) \\
= \frac{1}{6} \times \frac{1}{6} \times \frac{1}{2} \times \frac{1}{2} \times \frac{1}{6} \times \frac{1}{2} \\
= 5.6996 \times 10^{-4}
\]
Example Use of HMM: Protein Families Modelling

- Baldi et al., *PNAS* 91:1059-1063, 1994
- HMM is used to model families of biological sequences, such as kinases, globins, & immunoglobulins
- Bateman et al., *NAR* 32:D138-D141, 2004
- HMM is used to model 6190 families of protein domains in Pfam

![HMM Architecture](Image)

**Fig. 1.** HMM architecture. $S$ and $E$ are the start and end states. Sequence of main states $m_i$ is the backbone. Side states $d_i$ (resp. $i$) correspond to deletions (resp. insertions).

---

Example Use of HMM: Gene Finding in Bacterial Genomes

- Borodovsky et al., *NAR* 23:3554-3562, 1995
- Investigated statistical features of 3 classes (wrt level of codon usage bias) of E. coli genes
- HMM for nucleotide sequences of each class was developed

![Histogram of Codon Usage](Image)

**Figure 4.** Distributions of Codon usage scores for 126 new genes. The x-axis represents the scores assigned by GORNEW, y-axis gives the score computed by QAA-DILCO program. The green line is the ideal, $p < 0.05$ for each gene a threshold of the was utilized.
Concluding Remarks…

What have we learned?

• Decision Trees

• Decision Trees Ensembles
  – Bagging
  – CS4

• Other Methods
  – K-Nearest Neighbour
  – Support Vector Machines
  – Bayesian Approach
  – Hidden Markov Models
Any Question?

http://www.cs.waikato.ac.nz/ml/weka

Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization.

Exercise: Download a copy of WEKA. What are the names of classifiers in WEKA that correspond to C4.5 and SVM?
Acknowledgements

• Most of the slides used in this ppt came from a tutorial that I gave with Jinyan Li at the 8th European Conference on Principles and Practice of Knowledge Discovery in Databases, Pisa, Italy, 20-24 September 2004

• The dishonest casino example came from slides I inherited from Ken Sung

• The “indep vs conditional indep” example came from Kwok Pui Choi

References

• L. Breiman, Bagging predictors, Machine Learning, 24:123--140, 1996
• L. Breiman, Random forests, Machine Learning, 45:5-32, 2001
• J. R. Quinlan, Induction of decision trees, Machine Learning, 1:81--106, 1986
• Jinyan Li et al., Data Mining Techniques for the Practical Bioinformatician, The Practical Bioinformatician, Chapter 3, pages 35—70, WSPC, 2004
References