For written notes on this lecture, please read chapter 3 of *The Practical Bioinformatician*. Alternatively, please read “Rule-Based Data Mining Methods for Classification Problems in Biomedical Domains”, a tutorial at PKDD04 by Jinyan Li and Limsoon Wong, September 2004. http://www.comp.nus.edu.sg/~wongls/talks/pkdd04/

Bioinformatics and Biomarker Discovery

*Part 2: Tools*

Limsoon Wong

3 September 2014
Outline

• Overview of Supervised Learning

• Decision Trees Ensembles
  – Bagging
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
Training data: $X$

$f(X)$

Class labels $Y$

Test data: $U$

$f(U)$

Predicted class labels

$f(\bullet)$: A classifier, a mapping, a hypothesis
Relational Representation of Patient Data

$n$ features (order of 1000)

\[
\begin{array}{ccccccc}
\text{gene}_1 & \text{gene}_2 & \text{gene}_3 & \text{gene}_4 & \ldots & \text{gene}_n \\
\hline
X_{11} & X_{12} & X_{13} & X_{14} & \ldots & X_{1n} \\
X_{21} & X_{22} & X_{23} & X_{24} & \ldots & X_{2n} \\
X_{31} & X_{32} & X_{33} & X_{34} & \ldots & X_{3n} \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
X_{m1} & X_{m2} & X_{m3} & X_{m4} & \ldots & X_{mn}
\end{array}
\]

$m$ samples

class

P
N
P
N
Importance of Rule-Based Methods

- Systematic selection of a small number of features used for the decision making
  \[\Rightarrow\] 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

- Every path from root to a leaf forms a decision rule
  - 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
## 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?
An Example

Source: Anthony Tung
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.
<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>

**Total 14 training instances**

**Temperature <= 70**

- 5, 8, 11, 13, 14
- P, P, D, P, P

**Temperature > 70**

- 1, 2, 3, 4, 6, 7, 9, 10, 12
- P, D, D, D, P, P, P, D, P

A numerical feature is generally partitioned by choosing a “cutting point”
Steps of Decision Tree Construction

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

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

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

4. Recursively, until the partitions become pure or almost pure.
Measures to Evaluate Which Feature is Best

- Gini index
- Information gain
- Information gain ratio
- T-statistics
- $\chi^2$
- ...

Copyright 2014 © Limsoon Wong
Gini Index

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

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

- Gini index is the expected value of the ratio of the diff of two arbitrary specimens to the mean value of all specimens
- Closer to 0, means the samples are “pure”
Gini Index

Let $\mathcal{U} = \{C_1, ..., C_k\}$ 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 $d[f]$ 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

$$gini^D_f(S) = 1 - \sum_{C_i \in \mathcal{U}} \left( \frac{|\{d \in D \mid d \in C_i, \ d[f] \in S\}|}{|D|} \right)^2$$

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

$$gini^D_f(S_1, S_2) = \sum_{s \in \{S_1, S_2\}} \frac{|\{d \in D \mid d[f] \in S\}|}{|D|} \cdot gini^D_f(S)$$

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

- **Gini index of a node:** the weighted average of the purity (measured by Gini) of subtrees at the node

  $\Rightarrow$ If each subtree is “pure”, this node is good
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 $C_1$
- $h(t) = \arg\max_{C \in \{C_1, C_2\}} |\{h_j \in \{h_1, h_2, h_3\} \mid h_j(t) = C\}|$
- Then $\text{prob}(h(t) = C_1) = \text{prob}(h_1(t)=C_1 \& h_2(t)=C_1 \& h_3(t)=C_1) + \text{prob}(h_1(t)=C_1 \& h_2(t)=C_1 \& h_3(t)=C_2) + \text{prob}(h_1(t)=C_1 \& h_2(t)=C_2 \& h_3(t)=C_1) + \text{prob}(h_1(t)=C_2 \& h_2(t)=C_1 \& h_3(t)=C_1) = 60\% \times 60\% \times 60\% + 60\% \times 60\% \times 40\% + 60\% \times 40\% \times 60\% + 40\% \times 60\% \times 60\% = 64.8\%$
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

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

A base inducer such as C4.5

A committee \( H \) of classifiers:

\[ h_1 \quad h_2 \quad \ldots \quad h_k \]

Draw 100 samples with replacement
Decision Making by Bagging

Given a new test sample $T$

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

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

Exercise: What does the above formula mean?
Significantly improves cross-batch prediction accuracy in gene expression profile analyses.
Concluding Remarks...
What have we learned?

• Decision Trees

• Decision Trees Ensembles
  – Bagging

• There are many other approaches of interest, but no time to cover here …
  – Support vector machine (SVM)
  – Nearest neighbour (kNN)
  – Naïve Bayes
References

• Jinyan Li et al., Data Mining Techniques for the Practical Bioinformatician, *The Practical Bioinformatician*, Chapter 3, pages 35—70, WSPC, 2004