For written notes on this lecture, please read chapter 3 of *The Practical Bioinformatician*,

CS2220: Introduction to Computational Biology
Unit 1: Essence of Knowledge Discovery

Li Xiaoli
11 August 2016
About the instructor

- **Instructor: Adj A/Prof Li Xiaoli**
  - Email
    - xlli@i2r.a-star.edu.sg
    - lixl@comp.nus.edu.sg
  - Web
    - http://www1.i2r.a-star.edu.sg/~xlli
  - Offices
    - NUS COM2-03-33
    - Fusionopolis (One North MRT station)
      Institute for Infocomm Research, A*Star
Outline

1. Knowledge discovery & Data Mining

2. Key techniques on knowledge discovery
   – Association rule mining
   – Clustering
   – Classification (focuses on decision tree)
Knowledge discovery

• The first two lectures aim to introduce some concepts, techniques on knowledge discovery, including
  – Association rule mining
  – Clustering/unsupervised machine learning
  – Classification/supervised machine learning
  – Classification performance measures
  – Feature selection techniques
Benefits of learning knowledge discovery

- **At the end of the course, students should**
  - Have good knowledge of basic data mining concepts
  - Be familiar with representative data mining techniques
  - Understand how to use a data mining tool, Weka
  - Given a biological dataset or task, have some knowledge of appropriate data mining techniques and tools for analyzing the data and address the task
What is knowledge discovery?

Jonathan’s blocks
- Hexagon
- Circle
- Triangle
- Cylinder

Jessica’s blocks
- Triangle
- Square
- Cylinder
- Diamond

What is this block?

Jonathan’s rules: Blue or Circle
Jessica’s rules: All the rest
What is knowledge discovery?

We have no problem distinguishing man/woman (even when the faces are half covered). But can you explain how? e.g. rules you have used?

no one can write down a clear comprehensive set of rules that work with high precision. However we can do fairly accurate predictions using data mining algos

http://how-old.net/
The steps of data mining

• Training data gathering
• Feature generation
  – k-grams, colour, texture, domain know-how, ...
• Feature selection
  – Entropy, $\chi^2$, CFS, t-test, domain know-how...
• Feature integration and model construction
  – SVM, ANN, PCL, CART, C4.5, kNN, ...

Some classifiers / machine learning methods
We are drowning in data
But starving for knowledge

Data Mining
What is knowledge discovery?

- **Knowledge Discovery from Databases (KDD):** The overall process of non-trivial extraction of implicit, previously unknown and potentially useful knowledge from large amounts of data.

- Exploration & analysis, by automatic or semi-automatic means, of large quantities of data in order to discover meaningful patterns.
Data mining: Core step of KDD process

Although “Data Mining” is just one of many steps in KDD, it is usually used to refer to the whole process of KDD.
Origins of data mining

• **Draws ideas from machine learning/AI, pattern recognition, statistics, and database systems**

• **Traditional Techniques may be unsuitable due to**
  – Enormity of data
  – High dimensionality of data
  – Heterogeneous, distributed nature of data
Data mining tools

- Weka
- RapidMiner
- R
- Python
- Orange
- IBM SPSS modeler
- SAS Data Mining
- Oracle Data Mining
- .......
Data mining tasks

• Prediction Methods
  – Use some variables to predict unknown or future values of other variables

• Description Methods
  – Find human-interpretable patterns that describe the data

From [Fayyad, et.al.] Advances in Knowledge Discovery and Data Mining, 1996
Data mining tasks

- Association Rule Discovery [Descriptive]
- Clustering [Descriptive]
- Classification [Predictive]
- Sequential Pattern Discovery [Descriptive]
- Regression [Predictive]
- Outlier Detection (Deviation Detection) [Predictive]
Outline

1. Knowledge discovery & Data Mining

2. Key techniques on knowledge discovery
   - Association rule mining
   - Clustering
   - Classification (focuses on decision tree)
Association rule mining: “Market basket analysis”
1. Association rules (Descriptive)

- Finding groups of items that tend to occur together

- Also known as “frequent itemset mining” or “market basket analysis”
  - Grocery store: Beer and Diaper
  - Amazon.com: People who bought this book also bought other books
  - Find gene mutations that are frequently occurred in patients
A real-world application:
Amazon book recommendations

Ian H. Witten (Author), Eibe Frank (Author), Mark A. Hall (Author)

List Price: $69.95
Price: $42.12 & this item ships for FREE with Super Saver Shipping. Details
You Save: $27.83 (40%)
Special Offers Available

In Stock.
Ships from and sold by Amazon.com. Gift-wrap available.

Want it delivered Tuesday, June 19? Order it in the next 37 hours and 24 minutes, and choose One-Day Shipping at checkout. Details

51 new from $33.24  29 used from $27.98
Frequent patterns & association rules

Frequently Bought Together

- **This item**: Data Mining: Practical Machine Learning Tools and Techniques, Third Edition
  - Author: Ian H. Witten
  - Edition: Paperback
  - Price: $42.12

- **Mining the Social Web**: Analyzing Data from Facebook, Twitter, LinkedIn, and Other Soci.
  - Author: Matthew A. Russell
  - Edition: Paperback
  - Price: $26.39

- **Data Analysis with Open Source Tools**: by Philipp K. Janert
  - Edition: Paperback
  - Price: $24.08

Association Rules

{data mining} \(\rightarrow\) {machine learning in action}

- **Machine Learning in Action**
  - Author: Peter Harrington
  - Edition: Paperback
  - Price: $26.99

- **Mining the Social Web**: Analyzing Data from...
  - Author: Matthew A. Russell
  - Edition: Paperback
  - Price: $26.39

- **Data Mining with R Learning with Case**
  - Author: Luis Torgo
  - Edition: Hardcover
  - Price: $67.70

- **Programming Collective Intelligence**: Building...
  - Author: Toby Segaran
  - Edition: Paperback
  - Price: $26.39
**Intuitive example:**
Association rules from transactional data

<table>
<thead>
<tr>
<th>Shopping Cart (transaction)</th>
<th>Items Bought</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>{fruits, , milk, beef, , eggs, …}</td>
</tr>
<tr>
<td>2</td>
<td>{sugar, , toothbrush, ice-cream, …}</td>
</tr>
<tr>
<td>3</td>
<td>{ , pacifier, formula, , blanket, …}</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
<tr>
<td>n</td>
<td>{battery, juice, beef, egg, chicken, …}</td>
</tr>
</tbody>
</table>
What is association rule mining?

• Given a set of transactions, find rules that predict the occurrence of items based on the occurrences of other items in the transaction.

Market-Basket transactions

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bread, Milk</td>
</tr>
<tr>
<td>2</td>
<td>Bread, Diaper, Beer, Eggs</td>
</tr>
<tr>
<td>3</td>
<td>Milk, Diaper, Beer, Coke</td>
</tr>
<tr>
<td>4</td>
<td>Bread, Milk, Diaper, Beer</td>
</tr>
<tr>
<td>5</td>
<td>Bread, Milk, Diaper, Coke</td>
</tr>
</tbody>
</table>

Example of Association Rules

\{\text{Diaper}\} \rightarrow \{\text{Beer}\},
\{\text{Milk, Bread}\} \rightarrow \{\text{Coke}\},
\{\text{Bread}\} \rightarrow \{\text{Milk}\},

Explanation and usefulness:

\{\text{Bread}\} \rightarrow \{\text{Milk}\}

“People who buy bread may also buy milk.”

• Stores might want to offer specials on bread to get people to buy more milk.
• Stores might want to put bread and milk close each other.

Note that Implication means co-occurrence, not causality!
Definition: Frequent itemset

- **Itemset**
  - A collection of \( n \geq 1 \) items
    - E.g. \{Milk, Bread, Diaper\}
  - \( k \)-itemset
    - An itemset with \( k \) items

- **Support count (\( \sigma \))**
  - Frequency of occurrence of an itemset
    - E.g. \( \sigma(\{\text{Milk}, \text{Bread}, \text{Diaper}\}) = 2 \)

- **Support (\( s \))**
  - Fraction of transactions that contain an itemset
    - E.g. \( s(\{\text{Milk, Bread, Diaper}\}) = 2/5 = 40\% \)

### Table

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bread, Milk</td>
</tr>
<tr>
<td>2</td>
<td>Bread, Diaper, Beer, Eggs</td>
</tr>
<tr>
<td>3</td>
<td>Milk, Diaper, Beer, Coke</td>
</tr>
<tr>
<td>4</td>
<td>Bread, Milk, Diaper, Beer</td>
</tr>
<tr>
<td>5</td>
<td>Bread, Milk, Diaper, Coke</td>
</tr>
</tbody>
</table>

- **Frequent itemset**
  - An itemset whose support is greater than or equal to a minsup threshold, which is usually defined by a user, possibly interactively
Definition: Association rule

- **Association Rule**
  - Expression of the form $X \rightarrow Y$, where $X$, $Y$ are itemsets, and $X \cap Y = \emptyset$
  - E.g. $\{\text{Milk, Diaper}\} \rightarrow \{\text{Beer}\}$

- **Rule Evaluation Metrics**
  - Support ($s$) is fraction of transactions containing both $X$ and $Y$
    \[
    P(X \cup Y) = \frac{\# \text{trans containing}(X \cup Y)}{\# \text{trans in } T}
    \]
  - Confidence ($c$) measures how often items in $Y$ appear in transactions that contain $X$
    \[
    P(Y \mid X) = \frac{\# \text{trans containing}(X \cup Y)}{\# \text{trans containing } X}
    \]

<table>
<thead>
<tr>
<th>TID</th>
<th>Items</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Bread, Milk</td>
</tr>
<tr>
<td>2</td>
<td>Bread, Diaper, Beer, Eggs</td>
</tr>
<tr>
<td>3</td>
<td>Milk, Diaper, Beer, Coke</td>
</tr>
<tr>
<td>4</td>
<td>Bread, Milk, Diaper, Beer</td>
</tr>
<tr>
<td>5</td>
<td>Bread, Milk, Diaper, Coke</td>
</tr>
</tbody>
</table>

\[
\begin{align*}
  s &= \frac{\sigma(\text{Milk, Diaper, Beer})}{|T|} = \frac{2}{5} = 0.4 \\
  c &= \frac{\sigma(\text{Milk, Diaper, Beer})}{\sigma(\text{Milk, Diaper})} = \frac{2}{3} = 0.67
\end{align*}
\]

- **Support**: How applicable is the rule?
- **Confidence**: How reliable is the rule?
Association rule mining task

• An association rule $r$ is strong if
  – $\text{Support}(r) \geq \text{min}\_\text{sup}$
  – $\text{Confidence}(r) \geq \text{min}\_\text{conf}$

• Given a set of transactions $T$, the goal of association rule mining is to find all rules having
  – support $\geq \text{minsup}$ threshold
  – confidence $\geq \text{minconf}$ threshold

• ARM is descriptive, which aims to identify patterns and rules from data automatically. It has been implemented in main data mining tools
2. Clustering

- Finding groups of objects such that the objects in a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups.
- Automatically learn the structure of data.
The input of clustering

A set of objects (instances, data points) $D_1, \ldots, D_m$. Each $D_i$ ($1 \leq i \leq m$) is represented as a $n$-dimensional feature vector: $D_i = (V_{i1}, V_{i2}, \ldots, V_{in})$

$f_1, f_2, \ldots, f_n$ could be various features like race, gender, income, education level, medical features (e.g. weight, blood pressure, total cholesterol, LDL (low-density lipoprotein cholesterol), HDL (high-density lipoprotein cholesterol), triglycerides (fats carried in the blood), x-ray, and biological features (e.g. DNA sequence, gene expression, protein interaction, gene functions).
Unsupervised learning: Purely learn from unlabeled data

<table>
<thead>
<tr>
<th>ID number</th>
<th>Clump Thickness</th>
<th>Uniformity of Cell Size</th>
<th>Uniformity of Cell Shape</th>
<th>Marginal Adhesion</th>
<th>Single Epithelial Cell Size</th>
<th>Bare Nuclei</th>
<th>Bland Chromatin</th>
<th>Normal Nucleoli</th>
<th>Mitoses</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ID2</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>10</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>ID3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>ID4</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td>1</td>
</tr>
</tbody>
</table>

- We can learn the relationship or structures from data by clustering; e.g. maybe ID1 and ID3 should be in 1 cluster?
Clustering: Definition

• Unsupervised learning: data without labels - different from supervised learning/classification (descriptive vs predictive)

• Given a set of objects or instances, each having a set of attributes, and a similarity measure among them, find clusters such that
  – Objects in one cluster are more similar to one another
  – Objects in separate clusters are less similar to one another

• Similarity Measures:
  – Euclidean Distance
  – Cosine Similarity
Clustering: Techniques and applications

- We will learn some standard clustering methods, e.g. K-means and hierarchical clustering, in the coming weeks.

- We will demonstrate how to apply clustering methods for gene expression data analysis.
3. Classification

- Also called supervised learning
- Learn from past experience/labels, and use the learned knowledge to classify new data
- Knowledge learned by intelligent machine learning algorithms
- Examples:
  - Clinical diagnosis for patients
  - Cell type classification
  - Secondary structure classification of protein as alpha-helix, beta-sheet, or random coil in biology
### An Classification Example

#### Attributes

<table>
<thead>
<tr>
<th>id number</th>
<th>Clump Thickness</th>
<th>Uniformity of Cell Size</th>
<th>Uniformity of Cell Shape</th>
<th>Marginal Adhesion</th>
<th>Single Epithelial Cell Size</th>
<th>Bare Nuclei</th>
<th>Bland Chromatin</th>
<th>Normal Nucleoli</th>
<th>Mitoses</th>
<th>Class: (2 for benign, 4 for malignant)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>ID2</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>10</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>ID3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>ID4</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

#### Find a model for class attribute as a function of the values of other attributes.

\[ f(\text{Clump Thickness}, \text{Uniformity of Cell Size}, \ldots, \text{Mitoses}) = \text{Class} \]

\[ f(4, 6, 5, 6, 8, 9, 2, 4) = ? \]

A test example
### Training Examples

<table>
<thead>
<tr>
<th>id number</th>
<th>Clump Thickness</th>
<th>Uniformity of Cell Size</th>
<th>Uniformity of Cell Shape</th>
<th>Marginal Adhesion</th>
<th>Single Epithelial Cell Size</th>
<th>Bare Nuclei</th>
<th>Bland Chromatin</th>
<th>Normal Nucleoli</th>
<th>Mitoses</th>
<th>Class: (2 for benign, 4 for malignant)</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID1</td>
<td>5</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>ID2</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>7</td>
<td>10</td>
<td>3</td>
<td>2</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>ID3</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>ID4</td>
<td>8</td>
<td>10</td>
<td>10</td>
<td>8</td>
<td>7</td>
<td>10</td>
<td>9</td>
<td>7</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

Find a model for class attribute as a function of the values of other attributes.

\[ f(\text{clump Thickness}, \text{Uniformity of Cell Size}, \ldots, \text{Mitoses}) = \text{Class} \]

**An test example**

\[ f(4, 6, 5, 6, 8, 9, 2, 4) = ? \]
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 (unknown)
Classification: Definition

• Given a collection of records (training set; history)
  – Each record contains a set of attributes, one of the key attributes is the class (i.e. class attribute)

• Find a model for class attribute as a function of the values of other attributes

• Goal: Previously unseen records should be assigned a class as accurately as possible (Future)

Prediction based on the past examples. Let History tell us the future!
Classification: Evaluation

• How good is your prediction? How do you evaluate your results?

• A test set is used to determine the accuracy of the model (Just like exam scenario)

• Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it

• Usually, the overlap between training and test is empty
Construction of a classifier

Training samples → Build Classifier → Classifier

Test instance → Apply Classifier → Prediction
Estimate accuracy: Wrong way

Exercise: Why is this way of estimating accuracy wrong?

We will elaborate how to estimate the classification accuracy in next lecture.
Typical notations

• Training data
  \{\langle x_1, y_1 \rangle, \langle x_2, y_2 \rangle, \ldots, \langle x_m, y_m \rangle \},
  where
  – \( x_i \ (1 \leq i \leq m) \) are \( n \)-dimensional vectors,
    e.g. \( x_i = (V_{i1}, V_{i2}, \ldots, V_{in}) \),
  – \( y_i \) are from a discrete space \( Y \),
    e.g. \( Y = \{\text{normal, disease}\} \), \( y_i = \text{normal} \)

• Test data
  \{\langle u_1, ? \rangle, \langle u_2, ? \rangle, \ldots, \langle u_k, ? \rangle \}
  – \( u_j \ (1 \leq j \leq k) \) are \( n \)-dimensional vectors
How to learn $f$, which is called classifier, classification model, a mapping, or a hypothesis

$Y = f(X)$

$\text{Training data: } X \quad \rightarrow \quad \text{Class labels } Y$

$\text{Test data: } U \quad \rightarrow \quad \text{Predicted class labels}$
Relational representation of gene expression data

$n$ features (order of 1000)

gene$_1$  gene$_2$  gene$_3$  gene$_4$  …  gene$_n$

$m$ samples

- $x_{11}$  $x_{12}$  $x_{13}$  $x_{14}$  …  $x_{1n}$
- $x_{21}$  $x_{22}$  $x_{23}$  $x_{24}$  …  $x_{2n}$
- $x_{31}$  $x_{32}$  $x_{33}$  $x_{34}$  …  $x_{3n}$

- $x_{m1}$  $x_{m2}$  $x_{m3}$  $x_{m4}$  …  $x_{mn}$

class

- P
- N
Features (aka attributes)

• **Categorical features**: describe *qualitative* aspects of an object
  – color = \{red, blue, green\}
  – Countries = \{Singapore, China, India, USA, …\}

• **Continuous or numerical features**: describe *quantitative* aspects of an object
  – gene expression
  – age
  – blood pressure
  – Discretization: Transform a continuous attribute into a categorical attribute (e.g. age)
Discretization of continuous attribute

• Transformation of a continuous attribute to a categorical attribute involves two subtasks

1. Decide how many categories
   • After the values are sorted, they are then divided into $n$ intervals by specifying $n-1$ split points (equal interval bin, equal frequency bin, or clustering)

2. Determine how to map the values of the continuous attribute to these categories
   • All the values in one interval are mapped to the same categorical value
An example of classification task

<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>
Overall picture of classification/supervised learning

Labelled Data + Algorithms

- Biomedical
- Financial
- Government
- Scientific
- ...

Classifiers

- Decision trees
- KNN
- SVM
- Bayesian Classifier
- Neural networks

Applications

- Disease diagnosis
- Stock market prediction
- Terrorist identification
- Drug prediction

Copyright 2016 © Wong Limsoon, Li Xiaoli
Requirements of biomedical classification

• **Accurate**
  – High accuracy
  – High sensitivity
  – High specificity
  – High precision

• **High comprehensibility**
Classification techniques

• Decision Tree based Methods
• Rule-based Methods
• Memory-based or Instance-based Methods
• Naïve Bayes Classifiers
• Neural Networks
• Support Vector Machines
• Ensemble Classification
Illustrating classification task

<table>
<thead>
<tr>
<th>Tid</th>
<th>Attr1</th>
<th>Attr2</th>
<th>Attr3</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Large</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Medium</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Small</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Medium</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Large</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Medium</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Large</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Small</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Medium</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Small</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Tid</th>
<th>Attr1</th>
<th>Attr2</th>
<th>Attr3</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>No</td>
<td>Small</td>
<td>55K</td>
<td>?</td>
</tr>
<tr>
<td>12</td>
<td>Yes</td>
<td>Medium</td>
<td>80K</td>
<td>?</td>
</tr>
<tr>
<td>13</td>
<td>Yes</td>
<td>Large</td>
<td>110K</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>No</td>
<td>Small</td>
<td>95K</td>
<td>?</td>
</tr>
<tr>
<td>15</td>
<td>No</td>
<td>Large</td>
<td>67K</td>
<td>?</td>
</tr>
</tbody>
</table>
Importance of rule-based methods

- Systematic selection of a small number of features used for the decision making
  ⇒ Increase 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. Incorporated in existing tools
Elegance of decision trees

Every path from root to a leaf forms a decision rule

Q: How many rules do we have in the tree?
### Example of a decision tree

#### Training Data

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Married</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Divorced</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Married</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Divorced</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Single</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Married</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Single</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

#### Splitting Attributes

- **Refund**
  - Yes: Leave nodes: Class
  - No: Split on MarSt
- **MarSt**
  - Single, Divorced
  - Split on TaxInc
- **TaxInc**
  - < 80K
  - > 80K

**Leave nodes: Class**
Another example of decision tree

Training Data

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Married</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Divorced</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Married</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Divorced</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Single</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Married</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Single</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

MarSt

Refund

TaxInc

There could be more than one tree that fits the same data!
Decision tree classification task

### Training Set

<table>
<thead>
<tr>
<th>Tid</th>
<th>Attr1</th>
<th>Attr2</th>
<th>Attr3</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Large</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Medium</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Small</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Medium</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Large</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Medium</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Large</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Small</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Medium</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Small</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

### Test Set

<table>
<thead>
<tr>
<th>Tid</th>
<th>Attr1</th>
<th>Attr2</th>
<th>Attr3</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>No</td>
<td>Small</td>
<td>55K</td>
<td>?</td>
</tr>
<tr>
<td>12</td>
<td>Yes</td>
<td>Medium</td>
<td>80K</td>
<td>?</td>
</tr>
<tr>
<td>13</td>
<td>Yes</td>
<td>Large</td>
<td>110K</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>No</td>
<td>Small</td>
<td>95K</td>
<td>?</td>
</tr>
<tr>
<td>15</td>
<td>No</td>
<td>Large</td>
<td>67K</td>
<td>?</td>
</tr>
</tbody>
</table>
Apply model to test data

Start from the root of tree.

Refund

Yes
NO

No
MarSt

Married

Single, Divorced

TaxInc

< 80K
NO

> 80K
YES

Test Data

<table>
<thead>
<tr>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Married</td>
<td>80K</td>
<td>?</td>
</tr>
</tbody>
</table>
Apply model to test data

Test Data

<table>
<thead>
<tr>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Married</td>
<td>80K</td>
<td>?</td>
</tr>
</tbody>
</table>

Refund

MarSt

TaxInc

< 80K

> 80K

Single, Divorced

Married

Yes No

Yes No

No

Copyright 2016 © Wong Limsoon, Li Xiaoli
Apply model to test data

Test Data

<table>
<thead>
<tr>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Married</td>
<td>80K</td>
<td>?</td>
</tr>
</tbody>
</table>

Refund → MarSt
- Single, Divorced
- Married

MarSt → TaxInc
- < 80K → No
- > 80K → YES
Apply model to test data

Test Data

<table>
<thead>
<tr>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Married</td>
<td>80K</td>
<td>?</td>
</tr>
</tbody>
</table>

Refund: No, Yes
Marital Status: Single, Divorced, Married
Taxable Income: < 80K, > 80K
Cheat: Yes, No
Apply model to test data

Test Data

<table>
<thead>
<tr>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Married</td>
<td>80K</td>
<td>?</td>
</tr>
</tbody>
</table>

- **Refund**
  - Yes: NO
  - No: MarSt

- **Marital Status**
  - Single, Divorced: TaxInc
  - Married

- **Taxable Income**
  - < 80K: NO
  - > 80K: YES
Apply model to test data

Test Data

<table>
<thead>
<tr>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>No</td>
<td>Married</td>
<td>80K</td>
<td>?</td>
</tr>
</tbody>
</table>

Assign Cheat to “No”
Decision tree classification task

Training Set

<table>
<thead>
<tr>
<th>Tid</th>
<th>Attrib1</th>
<th>Attrib2</th>
<th>Attrib3</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Large</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Medium</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Small</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Medium</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Large</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Medium</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Large</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Small</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Medium</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Small</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Test Set

<table>
<thead>
<tr>
<th>Tid</th>
<th>Attrib1</th>
<th>Attrib2</th>
<th>Attrib3</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>11</td>
<td>No</td>
<td>Small</td>
<td>55K</td>
<td>?</td>
</tr>
<tr>
<td>12</td>
<td>Yes</td>
<td>Medium</td>
<td>80K</td>
<td>?</td>
</tr>
<tr>
<td>13</td>
<td>Yes</td>
<td>Large</td>
<td>110K</td>
<td>?</td>
</tr>
<tr>
<td>14</td>
<td>No</td>
<td>Small</td>
<td>95K</td>
<td>?</td>
</tr>
<tr>
<td>15</td>
<td>No</td>
<td>Large</td>
<td>67K</td>
<td>?</td>
</tr>
</tbody>
</table>
Decision tree induction

• **Many Algorithms:**
  – Hunt’s Algorithm (one of the earliest)
  – CART
  – ID3, C4.5
  – SLIQ, SPRINT

We want to learn a classification tree model from training data and we want to apply it to predict **future** test data as accurate as possible.

How to learn a decision tree from data automatically?
Most discriminatory feature and partition

• **Discriminatory Feature**
  – Each 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

• **Partition**
  – 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

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

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

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

Copyright 2016 © Wong Limsoon, Li Xiaoli
A numerical feature is generally partitioned by choosing a “cutting point”.

The question is how to choose a good cutting point.

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
Steps of decision tree construction

- Select the “best” feature as root node of the whole tree
- Partition 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 root node of this sub-tree
- Recursively, until the partitions become pure or almost pure
How to determine the best split?

Before Splitting: 10 records of class 0 (C0), 10 records of class 1 (C1)

Suppose we have three features, i.e. Own Car, Car Type, Student ID?

Which test condition is the best?

The split can help us to better distinguish different classes
How to determine the best split

- **Greedy approach**
  - Nodes with **homogeneous** class distribution are preferred

- **Need a measure of node impurity**

  - Non-homogeneous
    - High degree of impurity
  - Homogeneous
    - Low degree of impurity

  Provide certainty and confidence for classifying future test cases
Measures of node impurity

- Gini Index
- Entropy *(Do it yourself)*
- Misclassification error *(Do it yourself)*
The key idea of the best split

Before Splitting:

<table>
<thead>
<tr>
<th></th>
<th>C0</th>
<th>C1</th>
</tr>
</thead>
<tbody>
<tr>
<td>M0</td>
<td>10</td>
<td>10</td>
</tr>
</tbody>
</table>

**Impurity of parents**

Node N

A?  
Yes  
Node N1

<table>
<thead>
<tr>
<th></th>
<th>C0</th>
<th>C1</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1</td>
<td>1</td>
<td>9</td>
</tr>
</tbody>
</table>

No  
Node N2

<table>
<thead>
<tr>
<th></th>
<th>C0</th>
<th>C1</th>
</tr>
</thead>
<tbody>
<tr>
<td>M2</td>
<td>9</td>
<td>1</td>
</tr>
</tbody>
</table>

B?  
Yes  
Node N3

<table>
<thead>
<tr>
<th></th>
<th>C0</th>
<th>C1</th>
</tr>
</thead>
<tbody>
<tr>
<td>M3</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

No  
Node N4

<table>
<thead>
<tr>
<th></th>
<th>C0</th>
<th>C1</th>
</tr>
</thead>
<tbody>
<tr>
<td>M4</td>
<td>5</td>
<td>5</td>
</tr>
</tbody>
</table>

Which one is good? Attribute A or attribute B? Why?

How to compute a score for A (or B) as it has been divided into two branches?

For A, we definite the impurity measure for nodes N1 and N2 (to get M1, M2), and then merge them (e.g. via weighed average of M1 and M2) to get attribute A’s impurity score.
How to find the best split

Which attribute (A or B) should we choose? We choose the one that can reduce more impurity and bring more gain. Choose A if Gain1 > Gain2; Otherwise choose B.

Q1: How to compute the impurity of a node, e.g. M0 for node N, M1 (N1), M2 (N2), M3 (N3), M4 (N4)?
Q2: How to compute the impurity of a attribute, e.g. M12 for A, N1, N2, M34 for B, N3, N4?
Measure of impurity: GINI for a node

\[ GINI(t) = 1 - \sum_{j} [p(j | t)]^2 \]

\( p(j | t) \) is the relative frequency of class \( j \) at node \( t \).

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>0</td>
</tr>
<tr>
<td>C2</td>
<td>6</td>
</tr>
</tbody>
</table>

P(C1) = 0/6 = 0 \quad P(C2) = 6/6 = 1

Gini = 1 – P(C1)^2 – P(C2)^2 = 1 – 0 – 1 = 0

Smallest impurity

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>1</td>
</tr>
<tr>
<td>C2</td>
<td>5</td>
</tr>
</tbody>
</table>

P(C1) = 1/6 \quad P(C2) = 5/6

Gini = 1 – (1/6)^2 – (5/6)^2 = 0.278

Examples for computing Impurity using GINI for a node

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>2</td>
</tr>
<tr>
<td>C2</td>
<td>4</td>
</tr>
</tbody>
</table>

P(C1) = 2/6 \quad P(C2) = 4/6

Gini = 1 – (2/6)^2 – (4/6)^2 = 0.444

Highest impurity

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>3</td>
</tr>
<tr>
<td>C2</td>
<td>3</td>
</tr>
</tbody>
</table>

P(C1) = 3/6 \quad P(C2) = 3/6

Gini = 1 – (3/6)^2 – (3/6)^2 = 0.5
Splitting based on GINI: GINI for a subtree

- Used in CART, SLIQ, SPRINT
- When a node $p$ is split into $k$ partitions (children), the quality of split is computed as (weighted average):

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where

- $n_i = \text{number of records at child } i$
- $n = \text{number of records at node } p$
Binary attributes: Computing GINI index

- **Splits into two partitions**

- **Effect of weighing partitions**
  - Larger and purer partitions are sought

Before use B

<table>
<thead>
<tr>
<th></th>
<th>Parent</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>6</td>
</tr>
<tr>
<td>C2</td>
<td>6</td>
</tr>
</tbody>
</table>

\[ \text{Gini} = 0.500 \]

After using attribute B (weighted average of N1 & N2)

<table>
<thead>
<tr>
<th></th>
<th>N1</th>
<th>N2</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>C2</td>
<td>2</td>
<td>4</td>
</tr>
</tbody>
</table>

\[ \text{Gini} = 0.3715 \]

What is the gain?

\[ \text{Gini}(\text{Children}) = \frac{7}{12} \times 0.4082 + \frac{5}{12} \times 0.32 = 0.3715 \]

\[ \text{Gini}(N1) = 1 - \left(\frac{5}{7}\right)^2 - \left(\frac{2}{7}\right)^2 \]

\[ = 0.4082 \]

\[ \text{Gini}(N2) = 1 - \left(\frac{1}{5}\right)^2 - \left(\frac{4}{5}\right)^2 \]

\[ = 0.3200 \]
Categorical attributes: Computing Gini index

- For each distinct value, gather counts for each class in the dataset
- Use the count matrix to make decisions

<table>
<thead>
<tr>
<th>CarType</th>
<th>Family</th>
<th>Sports</th>
<th>Luxury</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>1</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>C2</td>
<td>4</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Gini</td>
<td>0.393</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CarType</th>
<th>{Sports, Luxury}</th>
<th>{Family}</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>C2</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>Gini</td>
<td>0.400</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CarType</th>
<th>{Sports}</th>
<th>{Family, Luxury}</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>C2</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>Gini</td>
<td>0.419</td>
<td></td>
</tr>
</tbody>
</table>

Which one do you want to choose to split?
Continuous attributes: Computing Gini index

- Use Binary Decisions based on one value
- Several choices for the splitting value and each splitting value has a count matrix associated with it
  - Class counts in each of the partitions, \( A < v \) and \( A \geq v \)
  - \( v \) could be any value (e.g. 97)
- Simple method to choose best \( v \)
  - For each \( v \), scan the database to gather count matrix and compute its Gini index

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Cheat</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Married</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Divorced</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Married</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Divorced</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Single</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Married</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Single</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>
Continuous attributes: Computing Gini index

- **For efficient computation**
  - For each continuous attribute,
    - Sort the attribute on values
    - Linearly scan these values, each time updating the count matrix and computing Gini index
    - Pick split position that has the least Gini index (largest gain)
Characteristics of decision trees

- Easy to implement
- Single coverage of training data (elegance)
- Divide-and-conquer splitting strategy
- Easy to explain to medical doctors or biologists
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)
Thank You

Contact: xlli@i2r.a-star.edu.sg if you have questions
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/

**CS2220: Introduction to Computational Biology**

**Unit 1: Essence of Knowledge Discovery**

Li Xiaoli

18 August 2016
Outline

1. Decision tree ensembles
2. Other machine learning approaches
   – kNN (k-Nearest Neighbors)
   – NB (Naïve Bayesian Classifier)
   – SVM (Support Vector Machines)
3. Classification model evaluation
4. Feature selection
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 $C_1$
- $h(t) = \text{argmax}_{C \in \{C_1, C_2\}} |\{h_j \in \{h_1, h_2, h_3\} | h_j(t) = C\}|$
- Then $\text{prob}(h(t) = C_1)$
  
  \begin{align*}
  &= \text{prob}(h_1(t) = C_1 \& h_2(t) = C_1 \& h_3(t) = C_1) + \\
  &\quad \text{prob}(h_1(t) = C_1 \& h_2(t) = C_1 \& h_3(t) = C_2) + \\
  &\quad \text{prob}(h_1(t) = C_1 \& h_2(t) = C_2 \& h_3(t) = C_1) + \\
  &\quad \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\% + \\
  &\quad 60\% \times 40\% \times 60\% + 40\% \times 60\% \times 60\% = 64.8\%
  \end{align*}
Bagging

• Proposed by Breiman (1996)

• Also called Bootstrap aggregating

• Make use of randomness injected to training data
Main idea

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 committee $H$ of classifiers:

$h_1$  $h_2$  ...  $h_k$

base inducer such as C4.5
Decision making by Bagging

Given a new test sample $T$

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

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

In practice, we can build a random forest by building multiple decision trees. Each decision tree can be built by randomly choosing training examples and/or features.

http://en.wikipedia.org/wiki/Random_forest
Outline

1. Decision tree ensembles

2. Other machine learning approaches
   – kNN (k-Nearest Neighbors)
   – NB (Naïve Bayesian Classifier)
   – SVM (Support Vector Machines)

3. Classification model evaluation

4. Feature selection
Instance-based classifiers

- Store the training records (without training explicit models) – no induction step
- Use training records directly to predict the class label of unseen cases: deduction step

Set of Stored Cases

<table>
<thead>
<tr>
<th>Atr1</th>
<th>......</th>
<th>AtrN</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>A</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>C</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>B</td>
</tr>
</tbody>
</table>

Unseen Case

<table>
<thead>
<tr>
<th>Atr1</th>
<th>......</th>
<th>AtrN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Instance-based classifiers

- **Rote-learner**
  - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly

- **k-Nearest Neighbors (k-NN)**
  - Uses $k$ “closest” points (nearest neighbors) for performing classification
1) Nearest-neighbor classifiers

- **Basic idea:**
  - If it walks like a duck, quacks like a duck, then it’s probably a duck
Nearest-neighbor classifiers

- **Requires three things**
  - A set of stored training records
  - Distance metric to compute distance between records
  - The value of $k$, the number of nearest neighbors to retrieve

- **To classify an unknown record**
  - Compute distance to other training records
  - Identify $k$ nearest neighbors
  - Use class labels of the nearest neighbors to determine the class label of the unknown record (e.g., by majority vote)
Definition of nearest neighbor

• K-nearest neighbors of a record \( x \) are data points that have the \( k \) smallest distance (or largest similarity) to \( x \).

(a) 1-nearest neighbor  
(b) 2-nearest neighbor  
(c) 3-nearest neighbor
Nearest-neighbor classification

• Compute distance between two points $p$ & $q$
  – Euclidean distance
    \[ d(p, q) = \sqrt{\sum_i (p_i - q_i)^2} \]

• Determine the class from nearest neighbor list
  – Take the majority vote of class labels among the $k$-nearest neighbors (odd vs even number)
  – Weigh the vote according to distance
    • weight factor, $w = 1/d$ or $w = 1/d^2$
Nearest-neighbor classification

- Choosing the value of $k$
  - If $k$ is too small, sensitive to noise points (e.g. $k=1, 2, 3$)
  - If $k$ is too large, neighborhood may include points from other classes

What if $k=n$, i.e. the number of all the data points?
Then it becomes majority class in the training data
If $k$ is too large, the prediction will be depended on the data points that are not really related to the current data point

How to decide the value of $k$? cross validation or separate validation set
Nearest-neighbor classification

• Scaling issues
  – Attributes may have to be scaled or normalized to prevent distance measures from being dominated by one of the attributes
  – Example:
    • F1: height of a person may vary from 1.4m to 2.4m
    • F2: weight of a person may vary from 40kg to 442kg
    • F3: Annual income of a person may vary from $10K to $5,000K

\[
p = (p_1, p_2, p_3) = (1.65, 48, 6000) \]
\[
q = (q_1, q_2, q_3) = (1.82, 75, 8000) \]

\[
d(p, q) = \sqrt{\sum_{i} (p_i - q_i)^2} = \sqrt{(1.65 - 1.82)^2 + (48 - 75)^2 + (6000 - 8000)^2}
\]
Normalization

• Min-max normalization
  – $[\text{min}A, \text{max}A] \rightarrow [\text{new}\_\text{min}A, \text{new}\_\text{max}A]$

\[
\nu' = \frac{\nu - \text{min}_A}{\text{max}_A - \text{min}_A}(\text{new}\_\text{max}_A - \text{new}\_\text{min}_A) + \text{new}\_\text{min}_A
\]

– E.g. Income range [$12,000, 98,000$] normalized to [0.0, 1.0]. Then $73,000$ is mapped to

\[
\frac{73,000 - 12,000}{98,000 - 12,000} (1.0 - 0) + 0 = 0.71
\]

\[
\frac{12,000 - 12,000}{98,000 - 12,000} (1.0 - 0) + 0 = 0
\]

\[
\frac{98,000 - 12,000}{98,000 - 12,000} (1.0 - 0) + 0 = 1
\]
Nearest-neighbor classification (cont')

• **Pros of kNN**
  – Easy to implement
  – Incremental addition of training data trivial

• **Cons**
  – k-NN classifiers are lazy learners, which do not build models explicitly. This can be relatively more expensive than eager learners (such as decision tree) when classifying a test/unknown record
  – Unlike decision tree that attempts to find a global model that fits the entire input space, nearest-neighbor classifiers make the prediction based on local information, which can be more susceptible to noise.
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
Outline

1. Decision tree ensembles
2. Other machine learning approaches
   - kNN (k-Nearest Neighbors)
   - NB (Naïve Bayesian Classifier)
   - SVM (Support Vector Machines)
3. Classification model evaluation
4. Feature selection
2) Bayesian classifier

- A probabilistic framework for classification problems
- Conditional Probability:
  \[
  P(C \mid A) = \frac{P(A, C)}{P(A)} \quad (1)
  \]
  \[
  P(A \mid C) = \frac{P(A, C)}{P(C)} \quad (2)
  \]

- Bayes Theorem: replace \( P(A,C) \) in (1) with \( P(A \mid C) \times P(C) \) from (2)
  \[
  P(C \mid A) = \frac{P(A \mid C)P(C)}{P(A)} \quad (3)
  \]
Applying Bayes Theorem: Basic idea

C = \{Y, N\}
A is a test case, we want its probability belonging to class C

A=(Refund=Yes, Marital=Married, Taxable=79)

Which class A belong to, Y or N?

P(Y|Refund=Yes, Marital=Married, Taxable=79)
P(N|Refund=Yes, Marital=Married, Taxable=79)

We choose the bigger one

\begin{align*}
P(C|A) &= \frac{P(A|C)P(C)}{P(A)} \\
\end{align*}

• Bayes Theorem
  – To compute P(C|A), we need to estimate P(A|C) and P(C)
  – P(C) is easy to compute. There is no need to compute P(A). Why?
Applying Bayes Theorem: Example

\[ P(C | A) = \frac{P(A | C)P(C)}{P(A)} \]

<table>
<thead>
<tr>
<th>Tid</th>
<th>Refund</th>
<th>Marital Status</th>
<th>Taxable Income</th>
<th>Evade</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>No</td>
<td>Single</td>
<td>70K</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>Yes</td>
<td>Married</td>
<td>120K</td>
<td>No</td>
</tr>
<tr>
<td>5</td>
<td>No</td>
<td>Divorced</td>
<td>95K</td>
<td>Yes</td>
</tr>
<tr>
<td>6</td>
<td>No</td>
<td>Married</td>
<td>60K</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>Yes</td>
<td>Divorced</td>
<td>220K</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>No</td>
<td>Single</td>
<td>85K</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>No</td>
<td>Married</td>
<td>75K</td>
<td>No</td>
</tr>
<tr>
<td>10</td>
<td>No</td>
<td>Single</td>
<td>90K</td>
<td>Yes</td>
</tr>
</tbody>
</table>

- \( C = \{Y, N\} \)
- Objective
  - \( P(Y|\text{Refund}=?, \text{Marital}=?, \text{Taxable}=79) \)
  - \( P(N|\text{Refund}=?, \text{Marital}=?, \text{Taxable}=79) \)

- What we need to compute?
  1. \( P(C): P(Y) = 3/10, P(N) = 7/10 \)
  2. \( P(A|C): \text{Use indep assumption} \)
     - \( P(\text{Refund}=?|Y), P(\text{Refund}=?|N) \)
     - \( P(\text{Marital}=?|Y), P(\text{Marital}=?|N) \)
     - \( P(\text{Taxable}=79|Y), P(\text{Taxable}=79|N) \)
  3. \( P(A|C) \times P(C): \text{Use Bayes Theorem} \)
     \[ = P(A_1|C) \times P(A_2|C) \times \cdots \times P(C) \]

\[ e.g. P(\text{Marital}=\text{Married}|N) = 4/7 \text{ (out of 7 } N \text{ examples, we have } 4 \text{ Married)} \]

However, \( P(\text{Taxable Income } =79|Y) = ? \)

\[ P(\text{Taxable Income } =179|Y) = ? \]
How to estimate probabilities from data for continuous attributes

• **Discretize the range into bins**

• **Probability density estimation**
  – Assume attribute follows a normal distribution
  – Use data to estimate parameters of distribution (e.g., mean and standard deviation)
  – Once probability distribution is known, can use it to estimate the conditional probability $P(A_i|C)$
Estimating probabilities from data

- **Normal distribution**
  
  \[
P(A_i | c_j) = \frac{1}{\sqrt{2\pi\sigma^2_{ij}}} e^{-\frac{(A_i - \mu_{ij})^2}{2\sigma^2_{ij}}}
\]
  
  - One for each \((A_i, C_i)\) pair

- For (Taxable Income, Class=No):
  
  Class=No
  
  - sample mean = 110
  - sample variance = 2975
  - Std Deviation \(\sigma = \sqrt{2975} = 54.54\)

\[
P(\text{Income} = 120 \mid \text{No}) = \frac{1}{\sqrt{2\pi(54.54)}} e^{-\frac{(120 - 110)^2}{2(2975)}} = 0.0072
\]
Naïve Bayes classifier: Example

Given a Test Record:
\[ X = (\text{Refund} = \text{No}, \text{Married}, \text{Income} = 120k) \]

Naïve Bayes Classifier:

\[
P(\text{Refund=Yes}|\text{No}) = \frac{3}{7} \\
P(\text{Refund=No}|\text{No}) = \frac{4}{7} \\
P(\text{Refund=Yes}|\text{Yes}) = 0 \\
P(\text{Refund=No}|\text{Yes}) = 1 \\
P(\text{Marital Status=Single}|\text{No}) = \frac{2}{7} \\
P(\text{Marital Status=Divorced}|\text{No}) = \frac{1}{7} \\
P(\text{Marital Status=Married}|\text{No}) = \frac{4}{7} \\
P(\text{Marital Status=Single}|\text{Yes}) = \frac{2}{7} \\
P(\text{Marital Status=Divorced}|\text{Yes}) = \frac{1}{7} \\
P(\text{Marital Status=Married}|\text{Yes}) = 0
\]

For taxable income:
If class=No: sample mean=110, sample variance=2975
If class=Yes: sample mean=90, sample variance=25

\[
P(\text{X}|\text{Class=No}) = P(\text{Refund=No}|\text{Class=No}) \times P(\text{Married}| \text{Class=No}) \times P(\text{Income=}120K|\text{Class=No}) = \frac{4}{7} \times \frac{4}{7} \times 0.0072 = 0.0024
\]

\[
P(\text{X}|\text{Class=Yes}) = P(\text{Refund=No}|\text{Class=Yes}) \times P(\text{Married}| \text{Class=Yes}) \times P(\text{Income=}120K| \text{Class=Yes}) = 1 \times 0 \times 1.2 \times 10^{-9} = 0
\]

Clearly, \( P(\text{X}|\text{No})P(\text{No}) > P(\text{X}|\text{Yes})P(\text{Yes}) \)

Therefore \( P(\text{No}|\text{X}) > P(\text{Yes}|\text{X}) \)
\[ \Rightarrow \text{Class} = \text{No} \]
Naïve Bayes classifier: Smoothing

• If one of the conditional probability is zero, then the entire expression becomes zero

• Probability estimation:

Original: \[ P(A_i | C) = \frac{N_{ic}}{N_c} \]

Laplace: \[ P(A_i | C) = \frac{N_{ic} + 1}{N_c + c} \]

\(N_{ic}\): The number of times of feature \(A_i\) occurred in \(C\)

\(N_c\): the number of examples in \(C\)

\(c\): number of classes
Naïve Bayes classifier: Pros & cons

• Pros
  – Easy to implement
  – Very efficient
  – Good results obtained in many applications
  – Robust to isolated noise points
  – Handle missing values by ignoring the instance during probability estimate calculations
  – Robust to irrelevant attributes

• Cons
  – Independence assumption may not hold for some attributes (Could therefore loss of accuracy)
  – Use other techniques such as Bayesian Belief Networks (BBN)
Outline

1. Decision tree ensembles
2. Other machine learning approaches
   - kNN (k-Nearest Neighbors)
   - NB (Naïve Bayesian Classifier)
   - SVM (Support Vector Machines)
3. Classification model evaluation
4. Feature selection
3) Support Vector Machine (SVM)

- Find linear hyperplane (decision boundary) that separates the data
SVM

• One Possible Solution
Another possible solution
SVM

• Many other possible solutions
SVM

- Which one is better? B1 or B2? How do you define better?
Find a hyperplane maximizing the margin $\Rightarrow$ B1 is better than B2
SVM

\[ \mathbf{w} \cdot \mathbf{x} + b = +1 \]

\[ \mathbf{w} \cdot \mathbf{x} + b = 0 \]

\[ \mathbf{w} \cdot \mathbf{x} + b = -1 \]

\[ f(\mathbf{x}) = \begin{cases} 
1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b \geq 1 \\
-1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b \leq -1 
\end{cases} \]

Margin = \[ \frac{2}{||\mathbf{w}||^2} \]
SVM

• We want to maximize:  
  \[
  \text{Margin} = \frac{2}{\| \vec{w} \|^2}
  \]
  – Which is equivalent to minimizing:  
  \[
  L(w) = \frac{\| \vec{w} \|^2}{2}
  \]
  – But subjected to the following constraints:
  \[
  f(\vec{x}_i) = \begin{cases} 
  1 & \text{if } \vec{w} \cdot \vec{x}_i + b \geq 1 \\
  -1 & \text{if } \vec{w} \cdot \vec{x}_i + b \leq -1
  \end{cases}
  \]

• This is a constrained optimization problem, which can be solved by some numerical approaches, e.g., quadratic programming (QP)
SVM

• What if the problem is not linearly separable?
Nonlinear SVM

- What if decision boundary is not linear?
Nonlinear SVM

• **Transform data into higher dimensional space**
  – Using “kernel trick”, actual transformation need not be known
  – Just compute similarity between two vectors in original space

• **Some Kernels:**

\[
K(x, y) = (x^T y + 1)^p
\]

\[
K(x, y) = \exp\left(-\left|x - y\right|^2 /(2\sigma^2)\right)
\]

http://svmlight.joachims.org/
http://www.csie.ntu.edu.tw/~cjlin/libsvm/
Outline

1. Decision tree ensembles
2. Other machine learning approaches
   - kNN (k-Nearest Neighbors)
   - NB (Naïve Bayesian Classifier)
   - SVM (Support Vector Machines)
3. Classification model evaluation
4. Feature selection
3. Classification model evaluation

• **All models are wrong, but some are useful!**

  - Wrong because it is a simplification of reality
  - Useful if it may reach certain prediction accuracy
Model evaluation

• **Metrics for Performance Evaluation**
  – How to evaluate the performance of a model?

• **Methods for Performance Evaluation**
  – How to obtain reliable estimates?

• **Methods for Model Comparison**
  – How to compare the relative performance among competing models?
Metrics for performance evaluation

I need ground truth to compare with the prediction results to do fair evaluation.

Should I trust the fortune teller’s prediction?

I need ground truth to compare with the prediction results to do fair evaluation.
Metrics for performance evaluation

- **Focus on the predictive capability of a model**
  - Rather than how fast it takes to classify or build models, scalability, etc.

- **Confusion Matrix (element -> #cases in test set):**

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class =Yes</td>
<td>Class =Yes</td>
</tr>
<tr>
<td>Class =Yes</td>
<td>a</td>
</tr>
<tr>
<td>Class =No</td>
<td>c</td>
</tr>
</tbody>
</table>

a: TP (true positive)  
b: FN (false negative)  
c: FP (false positive)  
d: TN (true negative)
Metrics for performance evaluation

- **In the test set**

<table>
<thead>
<tr>
<th>Actual</th>
<th>Prediction</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Class = Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Class = No</td>
</tr>
<tr>
<td>No</td>
<td>Yes</td>
<td>Class = Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Class = No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Class = Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Class = Yes</td>
</tr>
<tr>
<td>No</td>
<td>No</td>
<td>Class = No</td>
</tr>
<tr>
<td>Yes</td>
<td>No</td>
<td>Class = No</td>
</tr>
<tr>
<td>Yes</td>
<td>Yes</td>
<td>Class = Yes</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

- a: TP (true positive)
- b: FN (false negative)
- c: FP (false positive)
- d: TN (true negative)
## Metrics for performance evaluation

- **Most widely-used metric, Accuracy:**

\[
\text{Accuracy} = \frac{a + d}{a + b + c + d} = \frac{TP + TN}{TP + TN + FP + FN}
\]

- **Error Rate = 1 - Accuracy**

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th>Class = Yes</th>
<th>Class = No</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class = Yes</td>
<td>a (TP)</td>
<td>b (FN)</td>
<td></td>
</tr>
<tr>
<td>Class = No</td>
<td>c (FP)</td>
<td>d (TN)</td>
<td></td>
</tr>
</tbody>
</table>
Limitation of accuracy

• Consider a 2-class problem (imbalanced classification)
  – spam detection
  – fraud detection
  – disease diagnostic

• Usually negative class = OK class
  positive class = not-OK class

• Assume in the test set
  – Number of negative examples = 9990
  – Number of positive examples = 10
Limitation of accuracy

• Number of negative examples = 9990
• Number of positive examples = 10
• If model predicts everything to be negative class, accuracy is $\frac{9990}{9990+10} = 99.9\%$
  
  $(TP=0, \ TN=9990, \ FP=0, \ FN=10)$
  
  – Accuracy is misleading because model does not detect any positive class example
• In the imbalanced cases, accuracy is not really a reliable metric
Cost matrix

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C(i</td>
</tr>
<tr>
<td>Class =Yes</td>
<td>C(Yes</td>
</tr>
<tr>
<td>Class =No</td>
<td>C(Yes</td>
</tr>
</tbody>
</table>

C(i|j): Cost of misclassifying class j example as class i

Cost/penalty means how much you need to pay if you suffer misclassification
Cost matrix in medical domain

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C(i</td>
<td>j)</td>
</tr>
<tr>
<td>Class =Cancer</td>
<td>C(Cancer</td>
<td>Cancer)</td>
</tr>
<tr>
<td>Class =Normal</td>
<td>C(Cancer</td>
<td>Normal)</td>
</tr>
<tr>
<td></td>
<td>C(100)</td>
<td>C(99999?)</td>
</tr>
</tbody>
</table>

It is **NOT acceptable** to misclassify cancer patients into normal, as it could delay the treatment.

It is also **Not that acceptable** to misclassify normal patients into cancer – why?
Computing cost of classification

<table>
<thead>
<tr>
<th>Cost Matrix</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACTUAL CLASS</td>
<td>+</td>
</tr>
<tr>
<td>+</td>
<td>-1</td>
</tr>
<tr>
<td>-</td>
<td>1</td>
</tr>
</tbody>
</table>

Model M₁

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>150</td>
<td>40</td>
</tr>
<tr>
<td>-</td>
<td>60</td>
<td>250</td>
</tr>
</tbody>
</table>

Accuracy = 80%
Cost = \(-1*150 + 100*40 + 60*1 + 0*250\)
=3910   M₁ is better

Model M₂

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>+</td>
<td>250</td>
<td>45</td>
</tr>
<tr>
<td>-</td>
<td>5</td>
<td>200</td>
</tr>
</tbody>
</table>

Accuracy = 90%
Cost = \(-1*250 + 100*45 + 1*5 + 0*200\)
=4255
## Precision, recall & F-measure

**Precision** (p) = \[ \frac{a}{a + c} \]

**Recall** (r) = \[ \frac{a}{a + b} \]

<table>
<thead>
<tr>
<th>ACTUAL CLASS</th>
<th>PREDICTED CLASS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class = Yes</td>
<td>a</td>
</tr>
<tr>
<td>Class = No</td>
<td>c</td>
</tr>
</tbody>
</table>

What is the precision and recall (by default wrt positive class)?

**Precision**
We predict \( a+c \) cases as positives, out of which \( a \) cases are correct.

**Recall**
There are \( a+b \) positive cases, out of which \( a \) cases are classified as positive correctly.
Precision-recall trade-off

- A predicts better than B if A has better recall and precision than B.
- There is a trade-off between recall and precision.
- In some apps, once you reach satisfactory precision, you optimize for recall.
- In some apps, once you reach satisfactory recall, you optimize for precision.

Exercise: Why is there a trade-off between recall and precision?
Example of precision, recall & F-measure

Precision (p) = \frac{a}{a + c}

Recall (r) = \frac{a}{a + b}

p = \frac{a}{a+c} = \frac{40}{40+160} = 20\%

r = \frac{a}{a+b} = \frac{40}{40+60} = 40\%

• r is also called sensitivity or true positive rate (TPR)

• Specificity or true negative rate = \frac{d}{c+d}

  = \frac{5000}{160+5000} = 96.9\%
Model evaluation

• Metrics for Performance Evaluation
  – How to evaluate the performance of a model?

• Methods for Performance Evaluation
  – How to obtain reliable estimates?

• Methods for Model Comparison
  – How to compare the relative performance among competing models?
Methods for performance evaluation

• How to obtain a reliable estimate of performance?

• Performance of a model may depend on other factors besides the learning algorithm:
  – Class distribution
  – Cost of misclassification
  – Size of training and test sets
Learning curve

- Learning curve shows how accuracy changes with varying sample size

- Requires a sampling schedule for creating learning curve:
  - Arithmetic sampling
    - Langley, et al.
    - E.g. 10, 20, 30, ...
  - Geometric sampling
    - Provost et al
    - E.g. 2, 4, 8, 16, 32, ...

Effect of small sample size:
- Bias in the estimate
- Variance of estimate
Methods of estimation

• **Holdout**
  – Reserve 2/3 for training and 1/3 for testing

• **Random subsampling**
  – Repeated holdout

• **Cross validation**
  – Partition data into k disjoint subsets
  – k-fold: train on k-1 partitions, test on the remaining one
  – Leave-one-out: k=n

• **Bootstrap**
  – Sampling with replacement
Cross validation

5-fold cross validation


Leave-one-out cross validation

- Divide samples into k roughly equal disjoint parts
- Each part has similar proportion of samples from different classes
- Use each part to test other parts
- Average accuracy and F-measure etc
Requirements of biomedical classification

• High accuracy, sensitivity, specificity, precision

• High comprehensibility
Model evaluation

• **Metrics for Performance Evaluation**
  – How to evaluate the performance of a model?

• **Methods for Performance Evaluation**
  – How to obtain reliable estimates?

• **Methods for Model Comparison**
  – How to compare the relative performance among competing models?
All the measures that we have mentioned can be used

- Accuracy
- Error Rate
- Precision
- Recall/sensitivity
- Specificity
- F-measure
- ROC
ROC (Receiver Operating Characteristic)

- Developed in 1950s for signal detection theory to analyze noisy signals
  - Characterize the trade-off between positive hits and false alarms [You hope you can escape from fire but do not want to be disturbed by false alarm]
- ROC curve plots TP rate (on the y-axis) against FP rate (on the x-axis)
- Performance of each classifier represented as a point on the ROC curve
  - Changing the threshold of algorithm (prob) changes the location of the point
How to construct ROC curve

- Probabilistic classifier (IR system) can generate a probability value to indicate how likely a case belongs to positive class.

| Instance | P(+|A) | True Class |
|----------|-------|------------|
| 1        | 0.95  | +          |
| 2        | 0.93  | +          |
| 3        | 0.87  | -          |
| 4        | 0.85  | -          |
| 5        | 0.85  | -          |
| 6        | 0.85  | +          |
| 7        | 0.76  | -          |
| 8        | 0.53  | +          |
| 9        | 0.43  | -          |
| 10       | 0.25  | +          |

- Use classifier that produces posterior probability for each test instance P(+|A).

- Sort the instances according to P(+|A) in decreasing order.

- Apply threshold $t$ at each unique value of P(+|A).

- Count the number of TP, FP, TN, FN at each threshold.

- TP rate, TPR = TP/(TP+FN).

- FP rate, FPR = FP/(FP + TN).
TP rate and FP rate

• TP rate, $\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \text{Recall (pos)}$
  – Fraction of positives that I get back (high recall means we do not miss real fire alarms)

• FP rate (false-alarm ratio), $\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}$
  – Out of all the negatives, what is the fraction of mistakes (i.e. got misclassified as positives, false alarms)
ROC curves

- By changing $t$, we get a range of sensitivities and specificities of a classifier.
- A predicts better than B if A has better sensitivities than B at most specificities.
- Leads to ROC curve that plots sensitivity vs. $(1 - \text{specificity})$.

Exercise: Draw a typical curve of sensitivity vs specificity.
Using ROC for model comparison

- No model consistently outperforms the other
  - M1 is better for small FPR
  - M2 is better for large FPR

- Area under the ROC curve
  - Ideal:
    - Area = 1
  - Random guess:
    - Area = 0.5
Outline

1. Decision tree ensembles

2. Other machine learning approaches
   – kNN (k-Nearest Neighbors)
   – NB (Naïve Bayesian Classifier)
   – SVM (Support Vector Machines)

3. Classification model evaluation

4. Feature selection
Recall kNN …

Image credit: Zaki
Curse of Dimensionality

• How much of each dimension is needed to cover a proportion \( r \) of total sample space?

• Calculate by \( e_p(r) = r^{1/p} \)

• So, to cover 10% of a 15-D space, need 85% of each dimension!

Exercise: Why \( e_p(r) = r^{1/p} \)?
Consequence of the Curse

• Suppose the number of samples given to us in the total sample space is fixed

• Let the dimension increase

• Then the distance of the $k$ nearest neighbours of any point increases

• Then the $k$ nearest neighbours are less and less useful for prediction, and can confuse the $k$-NN classifier
Feature selection

- Given a sample space of $p$ dimensions
- It is possible that some dimensions are irrelevant
- Need to find ways to separate those dimensions (aka features) that are relevant (aka signals) from those that are irrelevant (aka noise)
Signal selection: Basic idea

- Choose a feature with low intra-class distance (variance is smaller)
- Choose a feature with high inter-class distance
Signal selection (e.g., t-statistics)

The t-stats of a signal is defined as

\[ t = \frac{|\mu_1 - \mu_2|}{\sqrt{(\sigma_1^2/n_1) + (\sigma_2^2/n_2)}} \]

where \( \sigma_i^2 \) is the variance of that signal in class \( i \), \( \mu_i \) is the mean of that signal in class \( i \), and \( n_i \) is the size of class \( i \).

- A feature \( f_1 \) can be considered better than a feature \( f_2 \) if \( t(f_1, C_1, C_2) > t(f_2, C_1, C_2) \). Thus given a collection of candidate features in samples of \( C_1 \) and \( C_2 \), we simply sort them by their \( t \)-test statistical measure, and pick those with the largest \( t \)-test statistical measures.
Self-fulfilling oracle

- Construct **artificial dataset** with 100 samples, each with 100,000 randomly generated features and randomly assigned class labels.

- Select 20 features with the best t-statistics (or other methods).

- Evaluate accuracy by cross validation using the 20 selected features.

- The resulting accuracy can be ~90%.

- But the true accuracy should be 50%, as the data were derived randomly.
What went wrong?

• The 20 features were selected from whole dataset

• Information in the held-out testing samples has thus been “leaked” to the training process

• The correct way is to re-select the 20 features at each fold; better still, use a totally new set of samples for testing
While **dimensionality reduction** is an important tool in machine learning/data mining, we must always be aware that it can *distort* the data in misleading ways.

Above is a two dimensional projection of an intrinsically three dimensional world....
Original photographer unknown/
See also www.cs.gmu.edu/~jessica/DimRedcDanger.htm

© Eamonn Keogh
A cloud of points in 3D

Can be projected into 2D

**XY or XZ or YZ**

In 2D **XZ** we see a triangle

In 2D **YZ** we see a square

In 2D **XY** we see a circle

Thank You

Contact: xlli@i2r.a-star.edu.sg if you have questions
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


• T. G. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization, Machine Learning, 40:139-157, 2000

• Naïve Bayesian Classification, Wikipedia, http://en.wikipedia.org/wiki/Naive_Bayesian_classification