Lecture Outline

• Issues Regarding Classification & Prediction
• Decision Tree Induction
• Attribute Selection
  – Information Gain
  – Gain Ratio
  – Gini Index

• Summary
Basic Concepts

• The purpose of data analysis is to
  – Design *models* describing important data trends
  – What does a model look like?
    • A function
    • A decision-tree
    • An artificial neural network

• Two major forms of data analysis
  – Classification
    • Predicts categorical labels (class labels)
  – Prediction
    • Models continuous valued functions

• Applications
  – target marketing, performance prediction, medical diagnosis, manufacturing, fraud detection, webpage categorization…
Supervised vs. Unsupervised learning

• **Supervised learning (classification)**
  – Supervision: The training data (observations, measurements, etc.) are accompanied by labels indicating the class of the observations
  – New data is classified based on the training set

• **Unsupervised learning (clustering)**
  – The class labels of training data is unknown
  – Given a set of measurements, observations, etc. with the aim of establishing the existence of classes or clusters in the data
Classification – A two-step process

• **Model construction: describing a set of predetermined classes**
  – Each tuple/sample is assumed to belong to a predefined class, as determined by the class label attribute
  – The set of tuples used for model construction is **training set**
  – The model is represented as classification rules, decision trees, or mathematical formulae

• **Model usage: for classifying future or unknown objects**
  – Estimate accuracy of the model
  – The known label of test sample is compared with the classified result from the model
  – Accuracy rate is the percentage of test set samples that are correctly classified by the model
  – Test set is independent of training set (otherwise overfitting)
  – If the accuracy is acceptable, use the model to classify new data

• **Note:** If the test set is used to select models, it is called **validation (test) set**
Step 1: Build a classification model

<table>
<thead>
<tr>
<th>Name</th>
<th>Age</th>
<th>Income</th>
<th>Loan Decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sandy Jones</td>
<td>youth</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>Bill Lee</td>
<td>youth</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>Caroline Fox</td>
<td>middle-aged</td>
<td>high</td>
<td>safe</td>
</tr>
<tr>
<td>Rick Field</td>
<td>middle-aged</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>Susan Lake</td>
<td>senior</td>
<td>low</td>
<td>safe</td>
</tr>
<tr>
<td>Claire Phips</td>
<td>senior</td>
<td>medium</td>
<td>safe</td>
</tr>
<tr>
<td>Joe Smith</td>
<td>middle-aged</td>
<td>high</td>
<td>safe</td>
</tr>
</tbody>
</table>

IF age = youth THEN loan_decision = risky
IF income = high THEN loan_decision = safe
IF age = middle-aged AND income = low
   THEN loan_decision = risky

...
Step 2: Use the model on test data

<table>
<thead>
<tr>
<th>name</th>
<th>age</th>
<th>income</th>
<th>loan_decision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Juan Bello</td>
<td>senior</td>
<td>low</td>
<td>safe</td>
</tr>
<tr>
<td>Sylvia Crest</td>
<td>middle-aged</td>
<td>low</td>
<td>risky</td>
</tr>
<tr>
<td>Anne Yee</td>
<td>middle-aged</td>
<td>high</td>
<td>safe</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Classification rules

Test data

New data

(John Henry, middle-aged, low)

Loan decision?

risky
Lecture Outline

• Issues Regarding Classification & Prediction
• Decision Tree Induction
• Bayes Classification Methods
• Rule-Based Classification
• Summary
Decision Tree Induction

• **Basic algorithm (a greedy algorithm)**
  – Tree is constructed in a top-down recursive divide-and-conquer manner
  – At start, all the training examples are at the root
  – Attributes are categorical (if continuous-valued, they are discretized in advance)
  – Examples are partitioned recursively based on selected attributes
  – Test attributes are selected on the basis of a heuristic or statistical measure (e.g., information gain)

• **Conditions for stopping partitioning**
  – All samples for a given node belong to the same class
  – There are no remaining attributes for further partitioning – majority voting is employed for classifying the leaf
  – There are no samples left
From Data to Decision Trees

<table>
<thead>
<tr>
<th>age</th>
<th>income</th>
<th>student</th>
<th>credit_rating</th>
<th>buys_computer</th>
</tr>
</thead>
<tbody>
<tr>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>youth</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>middle_age</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>senior</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>middle_age</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>youth</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>youth</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>senior</td>
<td>medium</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>youth</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>middle_age</td>
<td>medium</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>middle_age</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>senior</td>
<td>medium</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
</tbody>
</table>

Decision Tree:

- **age?**
  - **youth**
    - **student?**
      - no
      - yes
  - **middle_aged**
    - yes
    - fair
    - excellent
  - **senior**
    - no
    - yes

- **credit_rating?**
  - no
  - yes
  - fair
  - excellent
What is Entropy?

- **Shannon's Entropy (Information Theory)**
  - A measure of uncertainty associated with a random variable
  - A way to quantify the potential reduction in our uncertainty once we have learnt the outcome of the probabilistic process.
    - Coins with a head and tail
    - Coins with both heads
  - Calculation: For a discrete random variable $Y$ taking $m$ distinct values $\{y_1, ..., y_m\}$,
    - $H(Y) = - \sum_{i=1}^{m} p_i \log_2(p_i)$, where $p_i = P(Y = y_i)$
  - Interpretation:
    - Higher entropy $\Rightarrow$ Higher uncertainty
    - Lower entropy $\Rightarrow$ Lower uncertainty

- **Conditional Entropy**
  - $H(Y|X) = \sum_x p(x)H(Y|X = x)$
Information and Entropy

- Entropy is maximized by a uniform distribution.
- For coin toss example (equally likely: max-entropy)
  \[ H(Y) = -0.5 \log_2(0.5) - 0.5 \log_2(0.5) = 0.3 \]
- Suppose coin is a biased coin and ‘Head’ is certain (min-entropy)
  \[ H(Y) = -1 \log_2(1) = 0 \]
- In information theory, entropy is the average amount of information contained in each message received.
  More uncertainty \[\rightarrow\] More information.

• Steps:
  – Establish Classification Attribute Ci in the database ‘D’.
  – Compute Classification Attribute Entropy.
  – For all other attributes in D, calculate Information Gain using the classification attribute Ci.
  – Select Attribute with the highest gain to be the next Node in the tree (starting from the Root node).
  – Remove Node Attribute, creating reduced table DS.
  – Repeat steps 3-5 until all attributes have been used, or the same classification value remains for all rows in the reduced table.
Information Gain (IG)

- IG calculates effective change in entropy after making a decision based on the value of an attribute.
- For decision trees, it’s ideal to base decisions on the attribute that provides the largest change in entropy, the attribute with the highest gain.
  - Information Gain for attribute $A$ on set $S$ is defined by taking the entropy of $S$ and subtracting from it the summation of the entropy of each subset of $S$, determined by values of $A$, multiplied by each subset’s proportion of $S$.

$$I_G(S, A) = I_E(S) - \sum_{i=1}^{n} (p(S_i^A) \times I_E(S_i^A))$$
Attribute Selection Measure: Information Gain (ID3/C4.5)

- Select the attribute with the highest information gain
- Let $p_i$ be the probability that an arbitrary tuple in $D$ belongs to class $C_i$, estimated by $|C_{i,D}|/|D|$
- **Expected information** (entropy) needed to classify a tuple in $D$:
  \[ \text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2(p_i) \]

- **Information** needed (after using $A$ to split $D$ into $v$ partitions) to classify $D$:
  \[ \text{Info}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \text{Info}(D_j) \]

- **Information gained** by branching on attribute $A$
  \[ \text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D) \]
Attribute Selection - Example

- **Binary classes:**
  - Class $P$: buys_computer = “yes”
  - Class $N$: buys_computer = “no”

- **Expected Information**
  - $Info(D) = I(9,5) = -\frac{9}{14} \log_2 \left(\frac{9}{14}\right) - \frac{5}{14} \log_2 \left(\frac{5}{14}\right) = 0.940$

- Information needed after using “age” to split $D$ into 3 partitions:
  - $Info_{age}(D) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.694$
  - where $\frac{5}{14} I(2,3)$ means “youth” has 5 out of 4 samples, with 2 “yes”es and 3 “no”s.

  - $I(2,3) = -\frac{2}{5} \log_2 \left(\frac{2}{5}\right) - \frac{3}{5} \log_2 \left(\frac{3}{5}\right) = 0.971$
  - $I(4,0) = -\frac{4}{4} \log_2 \left(\frac{4}{4}\right) = 0$
  - $I(3,2) = -\frac{3}{5} \log_2 \left(\frac{3}{5}\right) - \frac{2}{5} \log_2 \left(\frac{2}{5}\right) = 0.971$

- Information gained by branching on attribute “age”
  - $Gain(age) = Info(D) - Info_{age}(D) = 0.940 - 0.694 = 0.246$

- **Similarly, work out gains for other attributes:**
  - $Gain(income) = 0.029$
  - $Gain(student) = 0.151$
  - $Gain(credit\_rating) = 0.048$
Dealing with Continuous Value Attributes

• Let attribute A be a continuous-valued attribute
• Must determine the best split point for A
  – Sort the value A in increasing order
  – Typically, the midpoint between each pair of adjacent values is considered as a possible split point
    • \( \frac{1}{2} (a_i + a_{i+1}) \) is the midpoint between the values of \( a_i \) and \( a_{i+1} \)
    • The point with the minimum expected information requirement for A is selected as the split-point for A
• Split:
  – D1 is the set of tuples in D satisfying \( A \leq \text{split-point} \), and D2 is the set of tuples in D satisfying \( A > \text{split-point} \)
  – e.g. the age attribute was discretized into three values:
    • \( \leq 30 \) converted to youth
    • \( 30 \ldots 40 \) converted to middle_age
    • \( > 40 \) converted to senior
Gain Ratio (C4.5)

• Information gain measure is **biased towards attributes with a large number of unique values.**

• C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain)

  \[
  S_{\text{SplitInfo}}_A(D) = -\sum_{j=1}^{\nu} \frac{|D_j|}{|D|} \times \log_2(\frac{|D_j|}{|D|})
  \]

  \[
  \text{GainRatio}(A) = \frac{\text{Gain}(A)}{S_{\text{SplitInfo}}_A(D)}
  \]

• **Examples**

  \[
  S_{\text{SplitInfo}_{\text{income}}}(D) = -\frac{4}{14} \times \log_2\left(\frac{4}{14}\right) - \frac{6}{14} \times \log_2\left(\frac{6}{14}\right) - \frac{4}{14} \times \log_2\left(\frac{4}{14}\right) = 1.557
  \]

  \[
  \text{GainRatio(income)} = \frac{0.029}{1.557} = 0.019
  \]

• The attribute with the maximum gain ratio is selected as the splitting attribute
• If a data set $D$ contains examples from $n$ classes, gini index, $gini(D)$ is defined as
  
  $gini(D) = 1 - \sum_{j=1}^{n} p_j^2$
  
  where $p_j$ is the relative frequency of class $j$ in $D$.

• If a data set $D$ is split on $A$ into two subsets $D_1$ and $D_2$, the gini index $gini_A(D)$ is defined as

  $gini_A(D) = \frac{|D_1|}{|D|} gini(D_1) + \frac{|D_2|}{|D|} gini(D_2)$

• Reduction in Impurity:

  $gini_{split, A}(D) = \Delta gini(A) = gini(D) - gini_A(D)$

• The attribute provides the smallest $gini_{split}(D)$ or the largest reduction in impurity) is chosen to split the node.

• need to enumerate all the possible splitting points for each attribute
Example: Gini Index

- D has 9 tuples in `buys_computer = “yes”` and 5 in “no”
  - \( gini(D) = 1 - \left( \frac{9}{14} \right)^2 - \left( \frac{5}{14} \right)^2 = 0.459 \)
- Suppose the attribute income partitions D into 10 in \( D_1 \): \{low, medium\} and 4 in \( D_2 \)
  - \( gini_{\text{income} \in \{\text{low, medium}\}}(D) = \frac{10}{14} gini(D_1) + \frac{4}{14} gini(D_2) \)
    - \( gini(D_1) = 1 - \left( \frac{7}{10} \right)^2 - \left( \frac{3}{10} \right)^2 \), \( gini(D_2) = 1 - \left( \frac{2}{4} \right)^2 - \left( \frac{2}{4} \right)^2 \)
  - \( gini_{\text{income} \in \{\text{low, medium}\}}(D) = 0.443 = gini_{\text{income} \in \{\text{high}\}}(D) \)
  - \( gini_{\text{income} \in \{\text{low, high}\}}(D) = 0.458 = gini_{\text{income} \in \{\text{medium}\}}(D) \)
  - \( gini_{\text{income} \in \{\text{medium, high}\}}(D) = 0.450 = gini_{\text{income} \in \{\text{low}\}}(D) \)
- Thus, split on \{low, medium\} and \{high\} has the lowest Gini index.
- All attributes are assumed continuous-valued
- May need other tools, e.g., clustering, to get the possible split values
- Can be modified for categorical attributes
The three measures, in general, return good results but:

- **Information gain:**
  - biased towards multivalued attributes

- **Gain ratio:**
  - tends to prefer unbalanced splits in which one partition is much smaller than the others

- **Gini index:**
  - biased to multivalued attributes
  - has difficulty when # of classes is large
  - tends to favor tests that result in equal-sized partitions and purity in both partitions
The rest of the slides are optional
Other Attribute Selection Measures

- **CHAID**: a popular decision tree algorithm, measure based on $\chi^2$ test for independence
- **C-SEP**: performs better than info. gain and gini index in certain cases
- **G-statistic**: has a close approximation to $\chi^2$ distribution
- **MDL (Minimal Description Length) principle** (i.e., the simplest solution is preferred):
  - The best tree as the one that requires the fewest # of bits to both (1) encode the tree, and (2) encode the exceptions to the tree
- **Multivariate splits** (partition based on multiple variable combinations)
  - **CART**: finds multivariate splits based on a linear comb. of attrs.
- **Which attribute selection measure is the best?**
  - Most give good results, none is significantly superior than others
Overfitting and Tree Pruning

• **Overfitting**: An induced tree may overfit the training data
  – Too many branches, some may reflect anomalies due to noise or outliers
  – Poor accuracy for unseen samples

• **Two approaches to avoid overfitting**
  – **Prepruning**: *Halt tree construction early*—do not split a node if this would result in the goodness measure falling below a threshold
    • Difficult to choose an appropriate threshold
  – **Postpruning**: *Remove branches* from a “fully grown” tree—get a sequence of progressively pruned trees
    • Use a set of data different from the training data to decide which is the “best pruned tree”
Enhancements to Basic Decision Tree Induction

• **Allow for continuous-valued attributes**
  – Dynamically define new discrete-valued attributes that partition the continuous attribute value into a discrete set of intervals

• **Handle missing attribute values**
  – Assign the most common value of the attribute
  – Assign probability to each of the possible values

• **Attribute construction**
  – Create new attributes based on existing ones that are sparsely represented
  – This reduces fragmentation, repetition, and replication
Classification in Large Databases

• Classification—a classical problem extensively studied by statisticians and machine learning researchers

• Scalability: Classifying data sets with millions of examples and hundreds of attributes with reasonable speed

• Why is decision tree induction popular?
  – relatively faster learning speed (than other classification methods)
  – convertible to simple and easy to understand classification rules
  – can use SQL queries for accessing databases
  – comparable classification accuracy with other methods

• RainForest (VLDB’98 — Gehrke, Ramakrishnan & Ganti)
  – Builds an AVC-list (attribute, value, class label)