Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Handling Different Kinds of Cases in Classification
- Summary
Supervised vs. Unsupervised Learning

- **Supervised learning (classification) (học có thấy)**
  - 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) (học không thấy)**
  - 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
Prediction Problems: Classification vs. Numeric Prediction

- **Classification**
  - predicts categorical class labels (discrete or nominal)
  - classifies data (constructs a model) based on the training set and the values (**class labels**) in a classifying attribute and uses it in classifying new data

- **Numeric Prediction**
  - models continuous-valued functions, i.e., predicts unknown or missing values

- **Typical applications**
  - Credit/loan approval: xem xét giao dịch safe/risk
  - Medical diagnosis: if a tumor is cancerous or benign
  - Fraud detection: if a transaction is fraudulent
  - Web page categorization: which category it is
Classification—A Two-Step Process

- Model construction (xây dựng mô hình): 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 (sử dụng mô hình): for classifying future or unknown objects
  - Estimate accuracy (ước lượng độ chính xác) 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 data tuples whose class labels are not known
Process (1): Model Construction

```
<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mike</td>
<td>Assistant Prof</td>
<td>3</td>
<td>no</td>
</tr>
<tr>
<td>Mary</td>
<td>Assistant Prof</td>
<td>7</td>
<td>yes</td>
</tr>
<tr>
<td>Bill</td>
<td>Professor</td>
<td>2</td>
<td>yes</td>
</tr>
<tr>
<td>Jim</td>
<td>Associate Prof</td>
<td>7</td>
<td>yes</td>
</tr>
<tr>
<td>Dave</td>
<td>Assistant Prof</td>
<td>6</td>
<td>no</td>
</tr>
<tr>
<td>Anne</td>
<td>Associate Prof</td>
<td>3</td>
<td>no</td>
</tr>
</tbody>
</table>
```

IF rank = ‘professor’ OR years > 6
THEN tenured = ‘yes’

Training Data

Classification Algorithms

Classifier (Model)
Process (2): Using the Model in Prediction

Testing Data

Classifier

Unseen Data

(Jeff, Professor, 4)

<table>
<thead>
<tr>
<th>NAME</th>
<th>RANK</th>
<th>YEARS</th>
<th>TENURED</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tom</td>
<td>Assistant Prof</td>
<td>2</td>
<td>no</td>
</tr>
<tr>
<td>Merlisa</td>
<td>Associate Prof</td>
<td>7</td>
<td>no</td>
</tr>
<tr>
<td>George</td>
<td>Professor</td>
<td>5</td>
<td>yes</td>
</tr>
<tr>
<td>Joseph</td>
<td>Assistant Prof</td>
<td>7</td>
<td>yes</td>
</tr>
</tbody>
</table>

Tenured? Yes
Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Handling Different Kinds of Cases in Classification
- Summary
**Decision Tree Induction: Training Dataset**

<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>&lt;=30</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>high</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>31…40</td>
<td>high</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>no</td>
</tr>
<tr>
<td>31…40</td>
<td>low</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>medium</td>
<td>no</td>
<td>fair</td>
<td>no</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>low</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&lt;=30</td>
<td>medium</td>
<td>yes</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>31…40</td>
<td>medium</td>
<td>no</td>
<td>excellent</td>
<td>yes</td>
</tr>
<tr>
<td>31…40</td>
<td>high</td>
<td>yes</td>
<td>fair</td>
<td>yes</td>
</tr>
<tr>
<td>&gt;40</td>
<td>medium</td>
<td>no</td>
<td>excellent</td>
<td>no</td>
</tr>
</tbody>
</table>

This follows an example of Quinlan’s ID3 (Playing Tennis)
Output: A Decision Tree for “buys_computer”

age?

<=30

student?

no

yes

31..40

yes

credit rating?

excellent

no

fair

>40

no

yes
Algorithm for 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
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 $D$:
  \[
  \text{Info}_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times I(D_j)
  \]
- **Information gained** by branching on attribute $A$
  \[
  \text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D)
  \]
**Attribute Selection: Information Gain**

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

\[
Info(D) = I(9,5) = -\frac{9}{14} \log_2 \left( \frac{9}{14} \right) - \frac{5}{14} \log_2 \left( \frac{5}{14} \right) = 0.940
\]

<table>
<thead>
<tr>
<th>age</th>
<th>( p_i )</th>
<th>( n_i )</th>
<th>( I(p_i, n_i) )</th>
</tr>
</thead>
<tbody>
<tr>
<td>&lt;=30</td>
<td>2</td>
<td>3</td>
<td>0.971</td>
</tr>
<tr>
<td>31...40</td>
<td>4</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>&gt;40</td>
<td>3</td>
<td>2</td>
<td>0.971</td>
</tr>
</tbody>
</table>

\[
Info_{age}(D) = \frac{5}{14} I(2,3) + \frac{4}{14} I(4,0) + \frac{5}{14} I(3,2) = 0.694
\]

\[
\frac{5}{14} I(2,3) \text{ means “age } \leq 30\text{” has 5 out of 14 samples, with 2 yes’es and 3 no’s. Hence}
\]

\[
Gain(\text{age}) = Info(D) - Info_{age}(D) = 0.246
\]

\[
Gain(\text{income}) = 0.029
\]

\[
Gain(\text{student}) = 0.151
\]

\[
Gain(\text{credit_rating}) = 0.048
\]
Computing Information-Gain for Continuous-Valued 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*
    - \((a_i+a_{i+1})/2\) 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:
  - \(D_1\) is the set of tuples in D satisfying \(A \leq \text{split-point}\), and
  - \(D_2\) is the set of tuples in D satisfying \(A > \text{split-point}\)
Gain Ratio for Attribute Selection (C4.5)

- Information gain measure is biased towards attributes with a large number of values.
- C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain).

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

- Example:

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

- 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 $gini(D)$ is defined as

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

Reduction in Impurity:

$$\Delta gini(A) = gini(D) - gini_A(D)$$

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).
Gini index (CART, IBM IntelligentMiner)

- Ex. 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 \( D_2 \)

\[
gini_{\text{income} \in \{\text{low,medium}\}}(D) = \left( \frac{10}{14} \right) Gini(D_1) + \left( \frac{4}{14} \right) Gini(D_1)
\]

\[
= \frac{10}{14} \left( 1 - \left( \frac{7}{10} \right)^2 - \left( \frac{3}{10} \right)^2 \right) + \frac{4}{14} \left( 1 - \left( \frac{2}{4} \right)^2 - \left( \frac{2}{4} \right)^2 \right)
\]

\[= 0.443\]

\[= Gini_{\text{income} \in \{\text{high}\}}(D).\]

\( Gini_{\{\text{low,high}\}} \) is 0.458; \( Gini_{\{\text{medium,high}\}} \) is 0.450. Therefore, split on the \{low,medium\} (and \{high\}) since it 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
Comparing Attribute Selection Measures

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
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 decision tree induction in data mining?
  - 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
Scalable Decision Tree Induction Methods

- **SLIQ** (EDBT’96 — Mehta et al.)
  - Builds an index for each attribute and only class list and the current attribute list reside in memory
- **SPRINT** (VLDB’96 — J. Shafer et al.)
  - Constructs an attribute list data structure
- **PUBLIC** (VLDB’98 — Rastogi & Shim)
  - Integrates tree splitting and tree pruning: stop growing the tree earlier
- **RainForest** (VLDB’98 — Gehrke, Ramakrishnan & Ganti)
  - Builds an AVC-list (attribute, value, class label)
- **BOAT** (PODS’99 — Gehrke, Ganti, Ramakrishnan & Loh)
  - Uses bootstrapping to create several small samples
Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Handling Different Kinds of Cases in Classification
- Summary
Bayesian Classification: Why?

- **A statistical classifier**: performs *probabilistic prediction*, *i.e.*, predicts class membership probabilities
- **Foundation**: Based on Bayes’ Theorem.
- **Performance**: A simple Bayesian classifier, *naïve Bayesian classifier*, has comparable performance with decision tree and selected neural network classifiers
- **Incremental**: Each training example can incrementally increase/decrease the probability that a hypothesis is correct — prior knowledge can be combined with observed data
- **Standard**: Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured
Bayesian Theorem: Basics

- Let $X$ be a data sample ("evidence"): class label is unknown.
- Let $H$ be a hypothesis that $X$ belongs to class $C$.
- Classification is to determine $P(H|X)$, (posteriori probability), the probability that the hypothesis holds given the observed data sample $X$.
- $P(H)$ (prior probability), the initial probability.
  - E.g., $X$ will buy computer, regardless of age, income, ...
- $P(X)$: probability that sample data is observed.
- $P(X|H)$ (likelyhood), the probability of observing the sample $X$, given that the hypothesis holds.
  - E.g., Given that $X$ will buy computer, the prob. that $X$ is 31..40, medium income.
Bayesian Theorem

- Given training data $\mathbf{X}$, posterior probability of a hypothesis $H$, $P(H|\mathbf{X})$, follows the Bayes theorem

$$P(H | \mathbf{X}) = \frac{P(\mathbf{X}|H)P(H)}{P(\mathbf{X})}$$

- Informally, this can be written as

  \[ \text{posteriori = likelihood x prior/evidence} \]

- Predicts $\mathbf{X}$ belongs to $C_2$ iff the probability $P(C_i|\mathbf{X})$ is the highest among all the $P(C_k|\mathbf{X})$ for all the $k$ classes

- Practical difficulty: require initial knowledge of many probabilities, significant computational cost
Towards Naïve Bayesian Classifier

- Let $D$ be a training set of tuples and their associated class labels, and each tuple is represented by an $n$-D attribute vector $X = (x_1, x_2, ..., x_n)$
- Suppose there are $m$ classes $C_1, C_2, ..., C_m$.
- Classification is to derive the maximum posteriori, i.e., the maximal $P(C_i|X)$
- This can be derived from Bayes’ theorem

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}$$

- Since $P(X)$ is constant for all classes, only $P(C_i|X) = P(X|C_i)P(C_i)$ needs to be maximized
Derivation of Naïve Bayes Classifier

- A simplified assumption: attributes are conditionally independent (i.e., no dependence relation between attributes):
  \[
P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i) = P(x_1 \mid C_i) \times P(x_2 \mid C_i) \times \ldots \times P(x_n \mid C_i)
  \]

- This greatly reduces the computation cost: Only counts the class distribution.

- If \( A_k \) is categorical, \( P(x_k \mid C_i) \) is the # of tuples in \( C_i \) having value \( x_k \) for \( A_k \) divided by \( |C_i, D| \) (# of tuples of \( C_i \) in \( D \)).

- If \( A_k \) is continuous-valued, \( P(x_k \mid C_i) \) is usually computed based on Gaussian distribution with a mean \( \mu \) and standard deviation \( \sigma \):
  \[
g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi \sigma}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}
  \]
  and \( P(x_k \mid C_i) \) is
  \[
P(X \mid C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})
  \]
Naïve Bayesian Classifier: Training Dataset

Class:
C1:buys_computer = 'yes'
C2:buys_computer = 'no'

Data sample
X = (age <=30, Income = medium, Student = yes, Credit_rating = Fair)
Naïve Bayesian Classifier: An Example

- **P(C_i):**
  - P(buys_computer = “yes”) = 9/14 = 0.643
  - P(buys_computer = “no”) = 5/14 = 0.357

- Compute P(X|C_i) for each class
  - P(age = “<=30” | buys_computer = “yes”) = 2/9 = 0.222
  - P(age = “<=30” | buys_computer = “no”) = 3/5 = 0.6
  - P(income = “medium” | buys_computer = “yes”) = 4/9 = 0.444
  - P(income = “medium” | buys_computer = “no”) = 2/5 = 0.4
  - P(student = “yes” | buys_computer = “yes”) = 6/9 = 0.667
  - P(student = “yes” | buys_computer = “no”) = 1/5 = 0.2
  - P(credit_rating = “fair” | buys_computer = “yes”) = 6/9 = 0.667
  - P(credit_rating = “fair” | buys_computer = “no”) = 2/5 = 0.4

- **X = (age <= 30, income = medium, student = yes, credit_rating = fair)**

- **P(X|C_i):**
  - P(X|buys_computer = “yes”) = 0.222 x 0.444 x 0.667 x 0.667 = 0.044
  - P(X|buys_computer = “no”) = 0.6 x 0.4 x 0.2 x 0.4 = 0.019

- **P(X|C_i) * P(C_i):**
  - P(X|buys_computer = “yes”) * P(buys_computer = “yes”) = 0.028
  - P(X|buys_computer = “no”) * P(buys_computer = “no”) = 0.007

**Therefore, X belongs to class (“buys_computer = yes”)**
Avoiding the 0-Probability Problem

- Naïve Bayesian prediction requires each conditional prob. be non-zero. Otherwise, the predicted prob. will be zero.
  \[
P(X \mid C_i) = \prod_{k=1}^{n} P(x_k \mid C_i)
\]

- Ex. Suppose a dataset with 1000 tuples, income=low (0), income=medium (990), and income = high (10),
- Use Laplacian correction (or Laplacian estimator)
  - Adding 1 to each case
    - Prob(income = low) = 1/1003
    - Prob(income = medium) = 991/1003
    - Prob(income = high) = 11/1003
  - The “corrected” prob. estimates are close to their “uncorrected” counterparts.
Naïve Bayesian Classifier: Comments

- Advantages
  - Easy to implement
  - Good results obtained in most of the cases

- Disadvantages
  - Assumption: class conditional independence, therefore loss of accuracy
  - Practically, dependencies exist among variables
    - E.g., hospitals: patients: Profile: age, family history, etc.
      Symptoms: fever, cough etc., Disease: lung cancer, diabetes, etc.
    - Dependencies among these cannot be modeled by Naïve Bayesian Classifier

- How to deal with these dependencies?
  - Bayesian Belief Networks (Chapter 9)
Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Handling Different Kinds of Cases in Classification
- Summary
Using IF-THEN Rules for Classification

- Represent the knowledge in the form of IF-THEN rules
  
  \[ \text{R: IF } \text{age} = \text{youth AND student} = \text{yes } \text{THEN buys_computer} = \text{yes} \]
  
  - Rule antecedent/precondition vs. rule consequent

- Assessment of a rule: coverage and accuracy
  
  - \( n_{\text{covers}} = \# \) of tuples covered by \( R \)
  
  - \( n_{\text{correct}} = \# \) of tuples correctly classified by \( R \)
  
  \[
  \text{coverage}(R) = \frac{n_{\text{covers}}}{|D|} \quad \text{// * D: training data set */}
  \]
  
  \[
  \text{accuracy}(R) = \frac{n_{\text{correct}}}{n_{\text{covers}}}
  \]

- If more than one rule are triggered, need conflict resolution
  
  - Size ordering: assign the highest priority to the triggering rules that has the “toughest” requirement (i.e., with the most attribute tests)
  
  - Class-based ordering: decreasing order of prevalence or misclassification cost per class
  
  - Rule-based ordering (decision list): rules are organized into one long priority list, according to some measure of rule quality or by experts
Rule Extraction from a Decision Tree

- Rules are \textit{easier to understand} than large trees
- One rule is created \textit{for each path} from the root to a leaf
- Each attribute-value pair along a path forms a conjunction: the leaf holds the class prediction
- Rules are mutually exclusive and exhaustive

Example: Rule extraction from our \textit{buys\_computer} decision-tree

- IF \textit{age} = young AND \textit{student} = no THEN \textit{buys\_computer} = no
- IF \textit{age} = young AND \textit{student} = yes THEN \textit{buys\_computer} = yes
- IF \textit{age} = mid-age THEN \textit{buys\_computer} = yes
- IF \textit{age} = old AND \textit{credit\_rating} = excellent THEN \textit{buys\_computer} = yes
- IF \textit{age} = young AND \textit{credit\_rating} = fair THEN \textit{buys\_computer} = no
Rule Induction: Sequential Covering Method

- Sequential covering algorithm: Extracts rules directly from training data
- Typical sequential covering algorithms: FOIL, AQ, CN2, RIPPER
- Rules are learned *sequentially*, each for a given class $C_i$ will cover many tuples of $C_i$ but none (or few) of the tuples of other classes
- Steps:
  - Rules are learned one at a time
  - Each time a rule is learned, the tuples covered by the rules are removed
  - The process repeats on the remaining tuples unless *termination condition*, e.g., when no more training examples or when the quality of a rule returned is below a user-specified threshold
- Comp. w. decision-tree induction: learning a set of rules *simultaneously*
Sequential Covering Algorithm

while (enough target tuples left)
  generate a rule
  remove positive target tuples satisfying this rule
How to Learn-One-Rule?

- Star with the *most general rule* possible: condition = empty
- *Adding new attributes* by adopting a greedy depth-first strategy
  - Picks the one that most improves the rule quality
- Rule-Quality measures: consider both coverage and accuracy
  - Foil-gain (in FOIL & RIPPER): assesses info_gain by extending condition
    \[
    \text{FOIL} \_ \text{Gain} = pos \times (\log_2 \frac{pos}{pos + neg} - \log_2 \frac{pos}{pos + neg})
    \]
    It favors rules that have high accuracy and cover many positive tuples
- Rule pruning based on an independent set of test tuples

\[
\text{FOIL} \_ \text{Prune}(R) = \frac{pos - neg}{pos + neg}
\]

Pos/neg are # of positive/negative tuples covered by R.
If \text{FOIL} \_ \text{Prune} is higher for the pruned version of R, prune R
Rule Generation

- To generate a rule
  
  while(true)
  
  find the best predicate $p$

  if foil-gain($p$) > threshold then add $p$ to current rule

  else break

\[ A3 = 1 \land A1 = 2 \land A2 = 1 \land A8 = 5 \land A3 = 1 \]

Positive examples

Negative examples
Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Handling Different Kinds of Cases in Classification
- Summary
Model Evaluation and Selection

- Evaluation metrics: How can we measure accuracy? Other metrics to consider?
- Use test set of class-labeled tuples instead of training set when assessing accuracy
- Methods for estimating a classifier’s accuracy:
  - Holdout method, random subsampling
  - Cross-validation
  - Bootstrap
- Comparing classifiers:
  - Confidence intervals
  - Cost-benefit analysis and ROC Curves
Classifier Evaluation Metrics: Accuracy & Error Rate

Confusion Matrix:

<table>
<thead>
<tr>
<th>Actual class\Predicted class</th>
<th>$C_1$</th>
<th>$\sim C_1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$C_1$</td>
<td>True Positives (TP)</td>
<td>False Negatives (FN)</td>
</tr>
<tr>
<td>$\sim C_1$</td>
<td>False Positives (FP)</td>
<td>True Negatives (TN)</td>
</tr>
</tbody>
</table>

Classifier Accuracy, or recognition rate: percentage of test set tuples that are correctly classified,

\[
\text{accuracy} = \frac{TP + TN}{TP + TN + FP + FN}
\]

Error rate: $1 - \text{accuracy}$, or

\[
\text{error rate} = \frac{FP + FN}{TP + TN + FP + FN}
\]
### Classifier Evaluation Metrics: Example - Confusion Matrix

<table>
<thead>
<tr>
<th>Actual class\Predicted class</th>
<th>buy_computer = yes</th>
<th>buy_computer = no</th>
<th>Total</th>
<th>Recognition(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>buy_computer = yes</td>
<td>6954</td>
<td>46</td>
<td>7000</td>
<td>99.34</td>
</tr>
<tr>
<td>buy_computer = no</td>
<td>412</td>
<td>2588</td>
<td>3000</td>
<td>86.27</td>
</tr>
<tr>
<td>Total</td>
<td>7366</td>
<td>2634</td>
<td>10000</td>
<td>95.42</td>
</tr>
</tbody>
</table>

- Given $m$ classes, an entry, $C_{Mi,j}$ in a **confusion matrix** indicates the number of tuples in class $i$ that were labeled by the classifier as class $j$.
- May be extra rows/columns to provide totals or recognition rate per class.
Classifier Evaluation Metrics: Sensitivity and Specificity

- **Class Imbalance Problem:**
  - one class may be *rare*, e.g. fraud detection data, medical data
  - significant *majority of the negative class* and minority of the positive class

- **Sensitivity**: True Positive recognition rate,
  \[
  \text{sensitivity} = \frac{TP}{P}
  \]

- **Specificity**: True Negative recognition rate,
  \[
  \text{specificity} = \frac{TN}{N}
  \]
Classifier Evaluation Metrics: Precision and Recall

- **Precision**: exactness – what % of tuples that the classifier labeled as positive are actually positive?
  
  \[
  \text{precision} = \frac{TP}{TP + FP}
  \]

- **Recall**: completeness – what % of positive tuples did the classifier label as positive?
  
  \[
  \text{recall} = \frac{TP}{TP + FN}
  \]

- Perfect score is 1.0
- Inverse relationship between precision & recall
## Classifier Evaluation Metrics: Example

<table>
<thead>
<tr>
<th>Actual class\Predicted class</th>
<th>cancer = yes</th>
<th>cancer = no</th>
<th>Total</th>
<th>Recognition(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cancer = yes</td>
<td>90</td>
<td>210</td>
<td>300</td>
<td>30.00 sensitivity</td>
</tr>
<tr>
<td>cancer = no</td>
<td>140</td>
<td>9560</td>
<td>9700</td>
<td>98.56 specificity</td>
</tr>
<tr>
<td>Total</td>
<td>230</td>
<td>9770</td>
<td>10000</td>
<td>96.40 accuracy</td>
</tr>
</tbody>
</table>

*Precision* = 90/230 = 39.13%; *Recall* = 90/300 = 30.00%
Classifier Evaluation Metrics: \( F \) and \( F_\beta \) Measures

- **\( F \) measure (\( F_1 \) or \( F \)-score):** harmonic mean of precision and recall,
  \[
  F = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}
  \]

- **\( F_\beta \):**
  - weighted measure of precision and recall
  - assigns \( \beta \) times as much weight to recall as to precision,
  \[
  F_\beta = \frac{(1 + \beta^2) \times \text{precision} \times \text{recall}}{\beta^2 \times \text{precision} + \text{recall}}
  \]
Evaluating Classifier Accuracy: Holdout & Cross-Validation Methods

- **Holdout method**
  - Given data is randomly partitioned into two independent sets
    - Training set (e.g., 2/3) for model construction
    - Test set (e.g., 1/3) for accuracy estimation
  - Random sampling: a variation of holdout
    - Repeat holdout k times, accuracy = avg. of the accuracies obtained

- **Cross-validation** (k-fold, where k = 10 is most popular)
  - Randomly partition the data into \( k \) mutually exclusive subsets, each approximately equal size
  - At \( i \)-th iteration, use \( D_i \) as test set and others as training set
  - Leave-one-out: \( k \) folds where \( k = \# \text{ of tuples} \), for small sized data
  - *Stratified cross-validation*: folds are stratified so that class dist. in each fold is approx. the same as that in the initial data
Evaluating the Classifier Accuracy: Bootstrap

- **Bootstrap**
  - Works well with small data sets
  - Samples the given training tuples uniformly *with replacement*
    - i.e., each time a tuple is selected, it is equally likely to be selected again and re-added to the training set
  - Several bootstrap methods, and a common one is the **.632 bootstrap**
    - A data set with \(d\) tuples is sampled \(d\) times, with replacement, resulting in a training set of \(d\) samples. The data tuples that did not make it into the training set end up forming the test set. About 63.2% of the original data end up in the bootstrap, and the remaining 36.8% form the test set (since \((1 - 1/d)^d \approx e^{-1} = 0.368\))
    - Repeat the sampling procedure \(k\) times, overall accuracy of the model:
      \[
      acc(M) = \sum_{i=1}^{k} (0.632 \times acc(M_i)_{test\_set} + 0.368 \times acc(M_i)_{train\_set})
      \]
Estimating Confidence Intervals: Classifier Models $M_1$ vs. $M_2$

- Suppose we have 2 classifiers, $M_1$ and $M_2$. Which is best?
- Use 10-fold cross-validation to obtain $\overline{err}(M_1)$ and $\overline{err}(M_2)$
- These mean error rates are just estimates of error on the true population of future data cases
- What if the difference between the 2 error rates is just attributed to chance?
  - Use a test of statistical significance
  - Obtain confidence limits for our error estimates
Estimating Confidence Intervals: Null Hypothesis

- Perform 10-fold cross-validation
- Assume samples follow a t distribution with \( k-1 \) degrees of freedom (here, \( k=10 \))
- Use t-test (or Student’s t-test)
- **Null Hypothesis**: \( M_1 \) & \( M_2 \) are the same, i.e.
  \[
  \left| \bar{\text{err}}(M_1) - \bar{\text{err}}(M_2) \right| = 0
  \]
- If we can reject null hypothesis, then
  - conclude that the difference between \( M_1 \) & \( M_2 \) is statistically significant
  - Chose model with lower error rate
Estimating Confidence Intervals: \textit{t-test}

- If only 1 test set available: \textbf{pairwise comparison}
  - For $i^{th}$ round of 10-fold cross-validation, the same cross partitioning is used to obtain $\text{err}(M_1)_i$ and $\text{err}(M_2)_i$
  - Average over 10 rounds to get $\overline{\text{err}}(M_1)$ and $\overline{\text{err}}(M_2)$
  - \textit{t-test} computes \textit{t-statistic} with $k-1$ degrees of freedom:
    \[
    t = \frac{\overline{\text{err}}(M_1) - \overline{\text{err}}(M_2)}{\sqrt{\text{var}(M_1 - M_2)/k}},
    \]
    where
    \[
    \text{var}(M_1 - M_2) = \frac{1}{k} \sum_{i=1}^{k} \left[ \text{err}(M_1)_i - \text{err}(M_2)_i - (\overline{\text{err}}(M_1) - \overline{\text{err}}(M_2)) \right]^2
    \]

- If 2 test sets available: use \textbf{non-paired t-test}
  where
  \[
  \text{var}(M_1 - M_2) = \sqrt{\frac{\text{var}(M_1)}{k_1} + \frac{\text{var}(M_2)}{k_2}},
  \]
  where $k_1$ & $k_2$ are # of cross-validation samples used for $M_1$ & $M_2$, resp.
Estimating Confidence Intervals: Table for t-distribution

Table B: t-Distribution Critical Values

<table>
<thead>
<tr>
<th>df</th>
<th>.25</th>
<th>.20</th>
<th>.15</th>
<th>.10</th>
<th>.05</th>
<th>.025</th>
<th>.02</th>
<th>.01</th>
<th>.005</th>
<th>.0025</th>
<th>.001</th>
<th>.0005</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.000</td>
<td>1.376</td>
<td>1.963</td>
<td>3.078</td>
<td>6.314</td>
<td>12.71</td>
<td>15.89</td>
<td>31.82</td>
<td>63.66</td>
<td>127.3</td>
<td>318.3</td>
<td>636.6</td>
</tr>
<tr>
<td>2</td>
<td>1.812</td>
<td>1.686</td>
<td>2.332</td>
<td>3.061</td>
<td>9.925</td>
<td>19.68</td>
<td>25.71</td>
<td>42.96</td>
<td>69.13</td>
<td>144.3</td>
<td>329.1</td>
<td>648.6</td>
</tr>
<tr>
<td>3</td>
<td>2.317</td>
<td>1.960</td>
<td>2.776</td>
<td>3.602</td>
<td>16.81</td>
<td>28.96</td>
<td>35.71</td>
<td>55.88</td>
<td>86.96</td>
<td>184.2</td>
<td>429.0</td>
<td>833.7</td>
</tr>
<tr>
<td>4</td>
<td>2.776</td>
<td>2.145</td>
<td>3.182</td>
<td>4.604</td>
<td>22.01</td>
<td>38.92</td>
<td>44.18</td>
<td>66.04</td>
<td>94.08</td>
<td>206.0</td>
<td>475.3</td>
<td>945.7</td>
</tr>
<tr>
<td>5</td>
<td>3.106</td>
<td>2.306</td>
<td>3.552</td>
<td>5.394</td>
<td>27.28</td>
<td>45.42</td>
<td>52.99</td>
<td>74.39</td>
<td>106.8</td>
<td>236.0</td>
<td>501.1</td>
<td>993.7</td>
</tr>
<tr>
<td>6</td>
<td>3.390</td>
<td>2.447</td>
<td>3.944</td>
<td>6.515</td>
<td>32.20</td>
<td>51.84</td>
<td>61.04</td>
<td>81.65</td>
<td>118.6</td>
<td>267.1</td>
<td>557.0</td>
<td>1083.0</td>
</tr>
<tr>
<td>7</td>
<td>3.607</td>
<td>2.571</td>
<td>4.317</td>
<td>7.378</td>
<td>37.37</td>
<td>57.87</td>
<td>68.97</td>
<td>88.21</td>
<td>131.2</td>
<td>299.8</td>
<td>613.3</td>
<td>1173.7</td>
</tr>
<tr>
<td>8</td>
<td>3.771</td>
<td>2.681</td>
<td>4.671</td>
<td>7.925</td>
<td>42.36</td>
<td>63.96</td>
<td>76.48</td>
<td>94.08</td>
<td>143.8</td>
<td>333.7</td>
<td>670.4</td>
<td>1262.2</td>
</tr>
<tr>
<td>9</td>
<td>3.889</td>
<td>2.796</td>
<td>4.988</td>
<td>8.307</td>
<td>47.02</td>
<td>70.07</td>
<td>83.44</td>
<td>99.27</td>
<td>157.1</td>
<td>370.0</td>
<td>728.7</td>
<td>1352.0</td>
</tr>
<tr>
<td>10</td>
<td>3.989</td>
<td>2.904</td>
<td>5.282</td>
<td>8.594</td>
<td>51.44</td>
<td>75.45</td>
<td>90.17</td>
<td>104.1</td>
<td>170.8</td>
<td>406.5</td>
<td>788.3</td>
<td>1441.9</td>
</tr>
<tr>
<td>12</td>
<td>4.221</td>
<td>3.081</td>
<td>5.860</td>
<td>9.109</td>
<td>57.43</td>
<td>85.95</td>
<td>101.26</td>
<td>115.8</td>
<td>190.2</td>
<td>455.3</td>
<td>873.8</td>
<td>1623.5</td>
</tr>
<tr>
<td>15</td>
<td>4.541</td>
<td>3.356</td>
<td>6.545</td>
<td>9.713</td>
<td>63.66</td>
<td>97.99</td>
<td>116.46</td>
<td>129.7</td>
<td>216.0</td>
<td>516.0</td>
<td>982.6</td>
<td>1788.7</td>
</tr>
<tr>
<td>20</td>
<td>4.988</td>
<td>3.645</td>
<td>7.278</td>
<td>10.273</td>
<td>70.54</td>
<td>109.88</td>
<td>135.81</td>
<td>148.3</td>
<td>248.0</td>
<td>583.3</td>
<td>1093.0</td>
<td>1972.6</td>
</tr>
<tr>
<td>30</td>
<td>5.505</td>
<td>4.029</td>
<td>8.090</td>
<td>10.894</td>
<td>78.07</td>
<td>122.40</td>
<td>159.08</td>
<td>168.5</td>
<td>286.1</td>
<td>659.0</td>
<td>1201.2</td>
<td>2180.8</td>
</tr>
</tbody>
</table>

- **Symmetric**
- **Significance level**, e.g., $\text{sig} = 0.05$ or 5% means $M_1$ & $M_2$ are significantly different for 95% of population
- **Confidence limit**, $z = \text{sig}/2$
Estimating Confidence Intervals: Statistical Significance

- Are $M_1$ & $M_2$ **significantly different**?
  - Compute $t$. Select *significance level* (e.g. $sig = 5\%$)
  - Consult table for t-distribution: Find *t value* corresponding to *$k-1$ degrees of freedom* (here, 9)
  - $t$-distribution is symmetric – typically upper % points of distribution shown → look up value for **confidence limit** $z=sig/2$ (here, 0.025)
  - If $t > z$ or $t < -z$, then $t$ value lies in rejection region:
    - **Reject null hypothesis** that mean error rates of $M_1$ & $M_2$ are same
    - Conclude: statistically significant difference between $M_1$ & $M_2$
  - **Otherwise**, conclude that any difference is chance.
Model Selection: ROC Curves

- **ROC** (Receiver Operating Characteristics) curves: for visual comparison of classification models
- Originated from signal detection theory
- Shows the trade-off between the true positive rate and the false positive rate
- The area under the ROC curve is a measure of the accuracy of the model
- Rank the test tuples in decreasing order: the one that is most likely to belong to the positive class appears at the top of the list
- The closer to the diagonal line (i.e., the closer the area is to 0.5), the less accurate is the model
- Vertical axis represents the true positive rate
- Horizontal axis rep. the false positive rate
- The plot also shows a diagonal line
- A model with perfect accuracy will have an area of 1.0
Issues Affecting Model Selection

- **Accuracy**
  - classifier accuracy: predicting class label

- **Speed**
  - time to construct the model (training time)
  - time to use the model (classification/prediction time)

- **Robustness**: handling noise and missing values

- **Scalability**: efficiency in disk-resident databases

- **Interpretability**
  - understanding and insight provided by the model

- Other measures, e.g., goodness of rules, such as decision tree size or compactness of classification rules
Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Handling Different Kinds of Cases in Classification
- Summary
Ensemble Methods: Increasing the Accuracy

- **Ensemble methods**
  - Use a combination of models to increase accuracy
  - Combine a series of \( k \) learned models, \( M_1, M_2, \ldots, M_k \), with the aim of creating an improved model \( M^* \)

- **Popular ensemble methods**
  - Bagging: averaging the prediction over a collection of classifiers
  - Boosting: weighted vote with a collection of classifiers
  - Ensemble: combining a set of heterogeneous classifiers
Bagging: Bootstrap Aggregation

- Analogy: Diagnosis based on multiple doctors’ majority vote
- Training
  - Given a set $D$ of $d$ tuples, at each iteration $i$, a training set $D_i$ of $d$ tuples is sampled with replacement from $D$ (i.e., bootstrap)
  - A classifier model $M_i$ is learned for each training set $D_i$
- Classification: classify an unknown sample $X$
  - Each classifier $M_i$ returns its class prediction
  - The bagged classifier $M^*$ counts the votes and assigns the class with the most votes to $X$
- Prediction: can be applied to the prediction of continuous values by taking the average value of each prediction for a given test tuple
- Accuracy
  - Often significant better than a single classifier derived from $D$
  - For noise data: not considerably worse, more robust
  - Proved improved accuracy in prediction
Boosting

- Analogy: Consult several doctors, based on a combination of weighted diagnoses—weight assigned based on the previous diagnosis accuracy.

- How boosting works?
  - **Weights** are assigned to each training tuple.
  - A series of $k$ classifiers is iteratively learned.
  - After a classifier $M_i$ is learned, the weights are updated to allow the subsequent classifier, $M_{i+1}$, to pay more attention to the training tuples that were misclassified by $M_i$.
  - The final $M^*$ combines the votes of each individual classifier, where the weight of each classifier's vote is a function of its accuracy.

- Boosting algorithm can be extended for numeric prediction.

- Comparing with bagging: Boosting tends to achieve greater accuracy, but it also risks overfitting the model to misclassified data.
Adaboost (Freund and Schapire, 1997)

- Given a set of $d$ class-labeled tuples, $(x_{1}, y_{1}), \ldots, (x_{d}, y_{d})$
- Initially, all the weights of tuples are set the same ($1/d$)
- Generate $k$ classifiers in $k$ rounds. At round $i$,
  - Tuples from $D$ are sampled (with replacement) to form a training set $D_i$ of the same size
  - Each tuple’s chance of being selected is based on its weight
  - A classification model $M_i$ is derived from $D_i$
  - Its error rate is calculated using $D_i$ as a test set
  - If a tuple is misclassified, its weight is increased, o.w. it is decreased
- Error rate: $err(x_j)$ is the misclassification error of tuple $x_j$. Classifier $M_i$ error rate is the sum of the weights of the misclassified tuples:
  \[
  error(M_i) = \sum_{j} w_j \times err(x_j)
  \]
- The weight of classifier $M_i$’s vote is
  \[
  \log \frac{1 - error(M_i)}{error(M_i)}
  \]
Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Handling Different Kinds of Cases in Classification
- Summary
Handling Different Kinds of Cases in Classification (under construction!)

- **Class Imbalance Problem**: Consider sensitivity and specificity measures.
- **Multiclass problem**: Instead of assigning a class label, assign a probability class distribution.
- **Active learning**
- **Transfer learning**
Chapter 8. Classification: Basic Concepts

- Classification: Basic Concepts
- Decision Tree Induction
- Bayes Classification Methods
- Rule-Based Classification
- Model Evaluation and Selection
- Techniques to Improve Classification Accuracy: Ensemble Methods
- Handling Different Kinds of Cases in Classification
- Summary
Summary (I)

- **Classification** is a form of data analysis that extracts models describing important data classes.
- Effective and scalable methods have been developed for decision tree induction, Naive Bayesian classification, rule-based classification, and many other classification methods.
- **Evaluation metrics** include: accuracy, sensitivity, specificity, precision, recall, $F$ measure, and $F_\beta$ measure.
- **Stratified k-fold cross-validation** is recommended for accuracy estimation. Bagging and boosting can be used to increase overall accuracy by learning and combining a series of individual models.
Summary (II)

- **Significance tests** and **ROC curves** are useful for model selection.
- There have been numerous **comparisons of the different classification methods**; the matter remains a research topic.
- No single method has been found to be superior over all others for all data sets.
- Issues such as accuracy, training time, robustness, scalability, and interpretability must be considered and can involve trade-offs, further complicating the quest for an overall superior method.
References (1)

- C. M. Bishop, **Neural Networks for Pattern Recognition.** Oxford University Press, 1995.
- H. Cheng, X. Yan, J. Han, and C.-W. Hsu, **Discriminative Frequent Pattern Analysis for Effective Classification**, ICDE'07.
- H. Cheng, X. Yan, J. Han, and P. S. Yu, **Direct Discriminative Pattern Mining for Effective Classification**, ICDE'08.
- W. Cohen. **Fast effective rule induction.** ICML'95.
- G. Cong, K.-L. Tan, A. K. H. Tung, and X. Xu. **Mining top-k covering rule groups for gene expression data.** SIGMOD'05.
References (2)

- W. Li, J. Han, and J. Pei, *CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules*, ICDM'01.
References (3)


References (4)

- R. Rastogi and K. Shim. **Public: A decision tree classifier that integrates building and pruning.** VLDB’98.
- J. Shafer, R. Agrawal, and M. Mehta. **SPRINT: A scalable parallel classifier for data mining.** VLDB’96.
- P. Tan, M. Steinbach, and V. Kumar. **Introduction to Data Mining.** Addison Wesley, 2005.
- X. Yin and J. Han. **CPAR: Classification based on predictive association rules.** SDM’03
- H. Yu, J. Yang, and J. Han. **Classifying large data sets using SVM with hierarchical clusters.** KDD’03.