Lecture 6
Classification and Prediction

(Part B)

Zhou Shuigeng

May 7, 2007
Outline

- Instance-Based Method
- Classification based on concepts from association rule mining
- Classification by Neural Networks
- Support Vector Machines
- Prediction
- Summary
Outline

- **Instance-Based Method** (基于案例的方法)
- Classification based on concepts from association rule mining
- Classification by Neural Networks
- Support Vector Machines
- Prediction
- Summary
Instance-Based Classifiers

- Store the training records
- Use training records to predict the class label of unseen cases
Instance-Based Classifiers

- Examples:
  - Rote-learner
    - Memorizes entire training data and performs classification only if attributes of record match one of the training examples exactly
  - Nearest neighbor (kNN)
    - Uses k “closest” points (nearest neighbors) for performing classification
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
  - The set of stored records
  - Distance metric
  - 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 nearest neighbors to determine the class label of unknown record (e.g., by taking majority vote)
Definition of Nearest Neighbor

- K-nearest neighbors of a record $x$ are data points that have the $k$ smallest distance to $x$
1 Nearest Neighbor

- Voronoi diagram
Nearest Neighbor Classification

- Compute distance between two points:
  - Example: 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
  - Weigh the vote according to distance
    - weight factor, \( w = 1/d^2 \)
Nearest Neighbor Classification...

- **Choosing the value of k:**
  - If $k$ is too small, sensitive to noise points
  - If $k$ is too large, neighborhood may include points from other classes
Nearest Neighbor Classification...

- Scaling issues
  - Attributes may have to be scaled to prevent distance measures from being dominated by one of the attributes
  - Example:
    - height of a person may vary from 1.5m to 1.8m
    - weight of a person may vary from 90lb to 300lb
    - income of a person may vary from $10K to $1M
Nearest Neighbor Classification…

- Problem with Euclidean measure:
  - High dimensional data
    - curse of dimensionality
  - Can produce counter-intuitive results

![Example vectors with Euclidean distances](image)

- Solution: Normalize the vectors to unit length
Characteristics of kNN Classifiers

- **Lazy learners**
  - No model building
    - Unlike eager learners such as decision tree induction and rule-based systems
  - Classifying unknown records are relatively expensive

- **Predictions are made based on local information**
  - Decision tree and rule-based classifiers attempt to find global model
  - More susceptible to outliers
Characteristics of kNN Classifiers

- Arbitrary-shaped boundaries
  - More flexible representation than the rectilinear boundaries of decision tree and rule-based classifiers

- High variability decision boundaries
  - depend on composition of training examples
  - More stable prediction if number of nearest neighbors increases
Example: PEPLS

- PEBLS: Parallel Examplar-Based Learning System
  - Works with both continuous and nominal features
    - Use modified value difference metric (MVDM) to compute distance for nominal features
  - Each record is also assigned a weight factor
  - Number of nearest neighbor, $k = 1$
Example: PEPLS

Distance between nominal attribute values:

\[ d(\text{Single, Married}) = \frac{|2/4 - 0/4| + |2/4 - 4/4|}{2} = 1 \]

\[ d(\text{Single, Divorced}) = \frac{|2/4 - 1/2| + |2/4 - 1/2|}{2} = 0 \]

\[ d(\text{Married, Divorced}) = \frac{|0/4 - 1/2| + |4/4 - 1/2|}{2} = 1 \]

\[ d(\text{Refund=Yes, Refund=No}) = \frac{|0/3 - 3/7| + |3/3 - 4/7|}{2} = 6/7 \]

\[
d(V_1, V_2) = \sum_{i} \left| \frac{n_{1i}}{n_1} - \frac{n_{2i}}{n_2} \right|
\]
Example: PEPLS

Distance between record X and record Y:

$$\Delta(X, Y) = w_X w_Y \sum_{i=1}^{d} d(X_i, Y_i)^2$$

where:

$$w_X = \frac{\text{Number of times } X \text{ is used for prediction}}{\text{Number of times } X \text{ predicts correctly}}$$

$$w_X \approx 1 \text{ if } X \text{ makes accurate prediction most of the time}$$

$$w_X > 1 \text{ if } X \text{ is not reliable for making predictions}$$
kNN based text classification (1)

- Vector space model
  - A document $d$ is represented by a $n$-dimensional vector
    $$
    \vec{d} = \{w_1, w_2, ..., w_n\}
    $$
  - Similarity between two documents $d_i$ and $d_j$ is defined as the inner product of the corresponding two documents $d_i$ and $d_j$
    $$
    Sim(d_i, d_j) = \vec{d}_i \cdot \vec{d}_j
    $$
Given a test document $d$, $k$NN method uses the classes of the $k$ nearest neighbors to weight class candidates.

Formally, the decision rule in $k$NN classification can be written as:

$$score(d, c_i) = \sum_{d_j \in kNN(d)} \text{Sim}(d, d_j) \delta(d_j, c_i) - s_i$$

Document $d$ belongs to class $c$ with the largest score among all classes.
kNN based text classification (3)

- kNN is the simplest classification method of high performance
  - Needs no training phase
  - Easy to be implemented
- kNN is inefficient
  - Require to calculate the similarity between a test document (unlabelled) to each training document
Ways to Speeding kNN up

- Reducing the number of classification features (dimension reduction)
- Reducing the size of training corpus
  - Uses the centroid of each class as the only representative of the entire class
  - Randomly selects some training instances as representatives of the class
  - Uses a set of generalized instances to represent the training corpus
    - kNN+linear classification
Training-corpus pruning based approach

- To speedup kNN for text classification by pruning training corpus
- A robust and controllable approach of training-corpus pruning is proposed
- The proposed approach can cut down the size of training corpus sharply, while keeping classification performance at the level comparable to that without training-corpus pruning
Problem Statement

- Given a training corpus $TC$, pruning $TC$ is to find a subset $S$ of $TC$, such that
  - Comparing to $|TC|$, $|S|$ is as less as possible, and
  - Classification competence of $S$ is nearly equal to that of $TC$
Observation (1)

- Training documents of the same class tends to form cluster in document vector space
- Different clusters (class) may overlay with each other
Observation (2)

- As far as $k$NN classification is concerned
  - inner documents or central documents (locating in the interior area) of a class are less important than outer documents or boundary documents (locating near the boundary), because their contribution to classification decision can be obtained from the outer documents
  - In this sense, inner documents of a class can be seen as *superfluous* documents. Here, *superfluous documents* are those that just not tell us much about making classification decision
Two documents Sets

- Given a training document $d$ in training corpus $D$, there are two sets of documents in $D$ that are related to $d$ in different ways.
  - Documents in one of the two sets are critical to the classification decision on $d$ if $d$ is seen as a test document (Reachibility set).
  - For documents in the other set, $d$ can contribute to the classification decisions on these documents if they are treated as test documents (Coverage set).
**k-reachability (d)**

- **Definition:** Given a document $d$ in training corpus $D$ and a natural number $k$,
  
  - $k$-reachability$(d) =$ \{ $d_i | d_i \in D$ and $d_i \in k\hat{NN}(d)$ \}

- **Implication:** $k$-reachability$(d)$ contains documents, whose effect on classification reaches document $d$
\textit{k-coverage}(d)

- Definition: Given a document \(d\) in training corpus \(D\) and a natural number \(k\),
  - \(K\)-coverage\((d) = \{d_i | d_i \in D \text{ and } d_i \in \text{class}(d) \text{ and } d \in k\text{-reachability}(d_i)\}\)

- Implication: \(d\)'s effect on classification can cover all documents in \(k\)-coverage\((d)\)
An Example

- k=2
  - 2-reachability(a)={b, e}
  - 2-coverage(a)={b}
Two Types of Documents

- **Potential superfluous documents**: A document $d$ is potential superfluous if it can be correctly classified with $k$-reachability($d$) by kNN method.

- **Critical documents**: A document $d$ is critical document if
  - At least one document in $k$-coverage($d$) cannot be correctly classified with its $k$-reachbility set.
  - After $d$ is deleted, at least one document in $k$-coverage($d$) will not be correctly classified with its $k$-reachbility set.
Pruning Rule

- A document $d$ is prunable (or removable) from training corpus if
  - Document $d$ is potential superfluous, and
  - Document $d$ is not critical document
The order of pruning is crucial because the pruning of one document may influence the decision on whether other documents can be pruned.

Intuitively, inner documents of a class in the training corpus should be pruned before outer documents for inner documents are prone to being superfluous documents.
Heuristic rules of pruning priority

- Document whose k-reachability set contains more documents of its own class should be pruned earlier.
- Document closer to the centroid of its class should be pruned earlier.
- Document further from the documents of other classes should be pruned earlier.
An Example

- d1, d2, d3 and d4 are in class c1 and prunable, k=4,
  - d1 is nearer to the centroid of class c1 than others
  - d2 is nearer to documents of class c2 than others
  - d1, d3 and d4 contain more documents of class c1 in their k-reachability sets than d2 does
- Pruning priority is
  - d1, d4, d3 and d2
The Pruning Algorithm

Algorithm 1 PruningTrainingCorpus(T, P)

**Input:** training corpus \( T = \{d_i| i = 1 \sim |T|\} \)

**Output:** result corpus \( P \) (a subset of \( T \))

1. \( P=T; S=\emptyset \) (empty set);
2. For each document \( d \) in \( T \)
3. \hspace{1em} Compute \( k\)-reachability\((d)\);
4. \hspace{1em} Compute \( k\)-coverage\((d)\);
5. \hspace{1em} For each document \( d \) in \( T \) but not in \( S \)
6. \hspace{2em} If \( d \) can be pruned (according to rule 1) and has the highest priority to be pruned (according to rule 2) then
7. \hspace{3em} \( S=S \cup \{d\}; P=P-\{d\}; \)
8. \hspace{3em} For each document \( d_i \) in \( k\)-coverage\((d)\)
9. \hspace{4em} Remove \( d \) from \( k\)-reachability\((d_i)\);
10. \hspace{4em} Add the \( k+1 \)st nearest neighbor \( d_j \) of \( d_i \) to \( k\)-reachability\((d_i)\)
11. \hspace{4em} Add \( d_i \) to \( k\)-coverage\((d_j)\)
12. Return \( P \).
Outline

- Instance-Based Method
- Classification based on concepts from association rule mining
- Classification by Neural Networks
- Support Vector Machines
- Prediction
- Summary
Association-Based Classification

- Several methods for association-based classification
  - ARCS: Quantitative association mining and clustering of association rules (Lent et al’97)
    - It beats C4.5 in (mainly) scalability and also accuracy
  - Associative classification: (Liu et al’98)
    - It mines high support and high confidence rules in the form of “cond_set => y”, where y is a class label
  - CAEP (Classification by aggregating emerging patterns) (Dong et al’99)
    - Emerging patterns (EPs): the itemsets whose support increases significantly from one class to another
    - Mine Eps based on minimum support and growth rate
Outline

- Instance-Based Method
- Classification based on concepts from association rule mining
- Classification by Neural Networks
- Support Vector Machines
- Prediction
- Summary
Principle of Neural Networks

- Use neural models to approximate multiple variable functions like

\[ y = f(X_1, X_2, \ldots, X_n, W_1, W_2, \ldots, W_k) \]

- Here, the form of \( f(...) \) is unknown. \( X_i \) are input variables, \( W_i \) are coefficients.
Output $Y$ is 1 if at least two of the three inputs are equal to 1.
Artificial Neural Networks

\[ Y = \text{sign}(0.3X_1 + 0.3X_2 + 0.3X_3 - 0.4) \]

where \( \text{sign}(x) = \begin{cases} 
1 & \text{if } x \geq 0 \\
-1 & \text{if } x < 0
\end{cases} \)
Neural Network Applications

- NN can be used in
  - Classification and prediction
  - Regression
  - Forecasting
  - Clustering

- NN usually works well on numeric data
  - to take numeric fields as input and
  - to estimate or forecast continuous values
Neural Network Models

- Multi-layer Perceptrons (ANN)
- Radial Basis Function Networks (RBFN)
- Probabilistic Neural Networks (PNN)
Artificial Neural Network

- Model is an assembly of inter-connected nodes and weighted links

- Output node sums up each of its input value according to the weights of its links

- Compare output node against some threshold $t$

$$Y = \text{sign} \left( \sum_{i=1}^{d} w_i X_i - t \right)$$

$$= \text{sign} \left( \sum_{i=0}^{d} w_i X_i \right)$$
General Structure of ANN

Input Layer

Hidden Layer

Output Layer

Neuron $i$

Activation function $g(S_i)$

Output $O_i$

Training ANN means learning the weights of the neurons

$x_1$, $x_2$, $x_3$, $x_4$, $x_5$

$I_1$, $I_2$, $I_3$

$w_{i1}$, $w_{i2}$, $w_{i3}$

$S_i$, threshold, $t$

$O_i$
Neural Network Training

- Given a training data set
- Specify a network architecture
- Run a training algorithm to learn a set of weights
Network Training

- The ultimate objective of training
  - obtain a set of weights that makes almost all the tuples in the training data classified correctly

- Steps
  - Initialize weights with random values
  - Feed the input tuples into the network one by one
  - For each unit
    - Compute the net input to the unit as a linear combination of all the inputs to the unit
    - Compute the output value using the activation function
    - Compute the error
    - Update the weights and the bias
Network Training

- Initialize the weights \((w_0, w_1, \ldots, w_d)\)
- Repeat
  - For each training example \((x_i, y_i)\)
    - Compute \(f(w, x_i)\)
    - Update the weight
      \[
      w^{(k+1)} = w^{(k)} + \lambda [y_i - f(w^{(k)}, x_i)]x_i
      \]
  - Until stopping condition is met
Properties of Neural Network Training

- Network architectures are not unique and have to be specified
- Initial weights have impact on the final model
- Each training converges to a locally optimal solution
- The backpropagation training algorithm is slow when training data is large
Basic Parameters

- Network architecture, including the number of hidden layers and the number of nodes in each hidden layer.

- Learning rate $\lambda$, e.g., Simple Error-Correction Learning rule.

  $$w(t+1) = w(t) + \lambda (d - y) x_i$$

- Stop condition, e.g.,

  Iterations, accuracy, time
Advanced Parameters

- Activation function, e.g.,
  - sigmoid (default), logistic,
  - hyperbolic tangent
- Momentum factor $\alpha < 1$ (default 0.9), (speed up the convergence of the learning process).
- Varied learning rate
  - Initial $\lambda$, high $\lambda$, low $\lambda$, $\lambda$ decay
Advanced Training Methods

- Multiple network architectures
- Dynamically changing a network architecture, e.g., growing or shrinking
- Evolutionary training
Outline

- Instance-Based Method
- Classification based on concepts from association rule mining
- Classification by Neural Networks
- Support Vector Machines
- Prediction
- Summary
Support Vector Machines

- Find a linear hyperplane (decision boundary) that will separate the data
Support Vector Machines

- One possible solution
Another possible solution

Support Vector Machines
Support Vector Machines

- Other possible solutions
Support Vector Machines

- Which is better, $B_1$ or $B_2$?
- How to define the better?
Support Vector Machines

- Find hyperplane \textbf{maximizes} the margin => \( B_1 \) is better than \( B_2 \)
SVM vs. Perceptron

Perceptron

SVM

Minimizes least square error (gradient descent)  Maximizes margin
Support Vector Machines

\[ \vec{w} \cdot \vec{x} + b = 0 \]
\[ \vec{w} \cdot \vec{x} + b = -1 \]

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

Margin = \frac{2}{\|\vec{w}\|}
Support Vector Machines

- We want to maximize: \( \text{Margin} = \frac{2}{\|\mathbf{w}\|} \)
- Which is equivalent to minimizing: \( L(w) = \frac{\|\mathbf{w}\|^2}{2} \)
- But subjected to the following constraints:

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

- Or

\[y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1, \quad i = 1, 2, \ldots, N\]

- This is a constrained optimization problem
  - Solve it using Lagrange multiplier method
Learning Linear SVM

- Lagrange multiplier:
  
  \[ L(w) = \frac{||\mathbf{w}||^2}{2} - \sum_{i=1}^{n} \lambda_i [y_i (\mathbf{w} \cdot \mathbf{x}_i + b) - 1] \]

  - Take derivative w.r.t \( \lambda \) and \( b \):
    
    \[ \frac{\partial L}{\partial \mathbf{w}} = 0 \Rightarrow \mathbf{w} = \sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i \]
    
    \[ \frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i} \lambda_i y_i = 0 \]

  - Additional constraints:
    
    \[ \lambda_i \geq 0 \]
    
    \[ \lambda_i [y_i (\mathbf{w} \cdot \mathbf{x}_i + b) - 1] = 0 \]

  - Dual problem:
    
    \[ L(w) = \sum_{i=1}^{n} \lambda_i - \sum_{i=1}^{n} \lambda_i \lambda_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \]
Learning Linear SVM

- Bigger picture:
  - Learning algorithm needs to find $\mathbf{w}$ and $b$
  - To solve for $\mathbf{w}$:
    $$\mathbf{w} = \sum_{i=1}^{n} \lambda_i y_i \mathbf{x}_i$$

- But:
  $$\lambda_i [y_i (\mathbf{w} \cdot \mathbf{x}_i + b) - 1] = 0$$

  - $\lambda$ is zero for points that do not reside on
    $$y_i (\mathbf{w} \cdot \mathbf{x}_i + b) = 1$$

  - Data points where $\lambda$s are not zero are called support vectors
Example of Learning Linear SVM
Bigger picture...
- Decision boundary depends only on support vectors
  - If you have data set with same support vectors, decision boundary will not change
- How to classify using SVM once $\mathbf{w}$ and $b$ are found? Given a test record, $\mathbf{x}_i$

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

- Instance Based Method
- Classification based on concepts from association rule mining
- Classification by Neural Networks
- Support Vector Machines
- Prediction
- Summary
What is Prediction?

- Prediction is similar to classification
  - First, construct a model
  - Second, use model to predict unknown value
    - Major method for prediction is regression
      - Linear and multiple regression
      - Non-linear regression
  - Prediction is different from classification
    - Classification refers to predict categorical class label
    - Prediction models continuous-valued functions
Predictive Modeling in Databases

- Predictive modeling: Predict data values or construct generalized linear models based on the database data.

- Method outline:
  - Minimal generalization
  - Attribute relevance analysis
  - Generalized linear model construction
  - Prediction

- Determine the major factors which influence the prediction
  - Data relevance analysis: uncertainty measurement, entropy analysis, expert judgement, etc.

- Multi-level prediction: drill-down and roll-up analysis
Regress Analysis and Log-Linear Models in Prediction

- **Linear regression**: $Y = \alpha + \beta X$
  - Two parameters, $\alpha$ and $\beta$ specify the line and are to be estimated by using the data at hand.
  - using the least squares criterion to the known values of $Y_1, Y_2, ..., X_1, X_2, ...$

- **Multiple regression**: $Y = b_0 + b_1 X_1 + b_2 X_2$.
  - Many nonlinear functions can be transformed into the above.

- **Log-linear models**:
  - The multi-way table of joint probabilities is approximated by a product of lower-order tables.
Linear Regression

- Find the line whose prediction is as close as possible to every point

![Graph showing linear regression with points scattered along a line and a dotted line indicating the prediction error to be minimized.](image)
Linear Regression Shortcomings

- It's only linear. If the data's not linear, you're out of luck
- It can be heavily influenced by just a few errors in the data. Not robust
Locally Weighted Regression

- Construct an explicit approximation to cover a local region surrounding query instance $x_q$.
- Locally weighted linear regression:
  - The target function $f$ is approximated near $x_q$ using the linear function:
    $$\bar{f}(x) = w_0 + w_1a_1(x) + \ldots + w_na_n(x)$$
  - minimize the squared error: distance-decreasing weight $K$
    $$E(x_q) = \frac{1}{2} \sum_{x \in \text{k_nearest_neighbors_of}_x} (f(x) - \bar{f}(x))^2 K(d(x_q,x))$$
  - the gradient descent training rule:
    $$\Delta w_j = \eta \sum_{x \in \text{k_nearest_neighbors_of}_x} K(d(x_q,x))(f(x) - \bar{f}(x))a_j(x)$$
- In most cases, the target function is approximated by a constant, linear, or quadratic function.
# Prediction: Numerical Data

## Relevance Analysis

<table>
<thead>
<tr>
<th>#</th>
<th>Predictive Name</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Sale_Price</td>
<td>-4560.00 - 91950.000</td>
</tr>
<tr>
<td>2</td>
<td>Channel</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Cost_of_Goods_Sold</td>
<td>-3500.00 - 69900.000</td>
</tr>
<tr>
<td>4</td>
<td>Advertising_Cost</td>
<td>0.000 - 1716.000</td>
</tr>
<tr>
<td>5</td>
<td>Average_Sales_Area</td>
<td>1130.000 - 4230.000</td>
</tr>
</tbody>
</table>

## Profit

-365.00 - 460.00
805.00 - 1000.00
1260.00 - 6005.00
430.00 - 605.00
1130.00 - 1260.00
Prediction: Categorical Data

Relevance Analysis

Profit:
-366.00~480.00
480.00~805.00
805.00~1000.00
1130.00~1260.00
1260.00~6005.00

Camping Chain
GO Outlet
Independent

Mass Marketer
Sports Chain
Outline

- Instance Based Method
- Classification based on concepts from association rule mining
- Classification by Neural Networks
- Support Vector Machines
- Prediction
- Summary
Summary

- Classification is an extensively studied problem (mainly in statistics, machine learning & neural networks)
- Classification is probably one of the most widely used data mining techniques with a lot of extensions
- Scalability is still an important issue for database applications: thus combining classification with database techniques should be a promising topic
- Research directions: classification of non-relational data, e.g., text, spatial, multimedia, steaming data etc.
References (1)

References (2)


- W. Li, J. Han, and J. Pei, CMAR: Accurate and Efficient Classification Based on Multiple Class-Association Rules, Proc. 2001 Int. Conf. on Data Mining (ICDM '01), San Jose, CA, Nov. 2001.


References (3)