CSE4334/5334 Data Mining

K Nearest-Neighbor Classifier

Chengkai Li
University of Texas at Arlington
Spring 2015

(Slides courtesy of Pang-Ning Tan, Michael Steinbach and Vipin Kumar)
Instance-Based Classifiers

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>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>

• Store the training records
• Use training records to predict the class label of unseen cases

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

Examples:

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

- Nearest neighbor
  - 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

![Diagram showing the process of nearest neighbor classification](image)
Nearest-Neighbor Classifiers

- Requires three things
  - The set of stored 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 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

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

Voronoi Diagram
Nearest Neighbor Classification

Compute distance between two points:
- 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

\[
\begin{array}{cccccccccccc}
1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 0 \\
0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\
\end{array}
\quad \text{vs} \quad \begin{array}{cccccccccccc}
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
\end{array}
\]

\[d = 1.4142 \quad \text{vs} \quad d = 1.4142\]

- Solution: Normalize the vectors to unit length
Nearest neighbor Classification...

**k-NN classifiers are lazy learners**

- It does not build models explicitly
- Unlike eager learners such as decision tree induction and rule-based systems
- Classifying unknown records are relatively expensive
**Example: PEBLS**

### Distance between nominal attribute values:

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

#### Example 1:

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

- **Taxable Income**:
  - 125K
  - 100K
  - 70K
  - 120K
  - 95K
  - 60K
  - 220K
  - 85K
  - 75K
  - 90K

### Table:

<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>

#### Class:

<table>
<thead>
<tr>
<th>Class</th>
<th>Marital Status</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Single</td>
</tr>
<tr>
<td>Yes</td>
<td>2</td>
</tr>
<tr>
<td>No</td>
<td>2</td>
</tr>
</tbody>
</table>

### Class:

<table>
<thead>
<tr>
<th>Class</th>
<th>Refund</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td>
<td>0</td>
</tr>
<tr>
<td>No</td>
<td>3</td>
</tr>
</tbody>
</table>
**Example: PEBLS**

<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>X</td>
<td>Yes</td>
<td>Single</td>
<td>125K</td>
<td>No</td>
</tr>
<tr>
<td>Y</td>
<td>No</td>
<td>Married</td>
<td>100K</td>
<td>No</td>
</tr>
</tbody>
</table>

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:

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

- \( w_X \approx 1 \) if X makes accurate prediction most of the time
- \( w_X > 1 \) if X is not reliable for making predictions
CSE4334/5334 Data Mining
Support Vector Machines

Chengkai Li
University of Texas at Arlington
Spring 2015

(Slides courtesy of Pang-Ning Tan, Michael Steinbach and Vipin Kumar)
Support Vector Machines

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

One Possible Solution
Support Vector Machines

Another possible solution
Support Vector Machines

Other possible solutions
Support Vector Machines

Which one is better? B1 or B2?
How do you define better?
Find hyperplane maximizes the margin => B1 is better than B2
Support Vector Machines

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

Margin: \( \frac{2}{\| \vec{w} \|} \)

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

We want to maximize: \[ \text{Margin} = \frac{2}{\| \vec{w} \|} \]

- Which is equivalent to minimizing: \[ L(w) = \frac{\| \vec{w} \|^2}{2} \]

- But subjected to the following constraints:

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

- This is a constrained optimization problem
  - Numerical approaches to solve it (e.g., quadratic programming)
Support Vector Machines

What if the problem is not linearly separable?
Support Vector Machines

What if the problem is not linearly separable?

- Introduce slack variables
  - Need to minimize:
    $$L(w) = \frac{\| \tilde{w} \|^2}{2} + C \left( \sum_{i=1}^{N} \xi_i \right)$$

- Subject to:
  $$f(\tilde{x}_i) = \begin{cases} 
1 & \text{if } \tilde{w} \cdot \tilde{x}_i + b \geq 1 - \xi_i \\
-1 & \text{if } \tilde{w} \cdot \tilde{x}_i + b \leq -1 + \xi_i
\end{cases}$$
Nonlinear Support Vector Machines

What if decision boundary is not linear?
Nonlinear Support Vector Machines

Transform data into higher dimensional space