K-NN Classifier, Decision Tree and Fuzzy Classification

Prof. Daniel Yeung
School of Computer Science and Engineering
South China University of Technology

Outline

- K-Nearest Neighbor (K-NN) (4.5)
- Decision Tree (DT) (8.2 – 8.4)
- Fuzzy Classification (4.7)
K-Nearest Neighbor (K-NN)

A new pattern is classified by a majority vote of its neighbors, with the pattern being assigned to the class most common amongst its $k$ nearest neighbors.

Algorithm:

- Given an unseen sample
- Calculate the distance between unseen sample and each training sample
- Select the $K$ nearest training samples
- Majority vote from these $K$ samples
K-Nearest Neighbor (K-NN)

- Target function for the entire space may be described as a combination of less complex local approximations.

Noisy Data

- A simple linearly separable dataset
- Obviously, unseen sample in question should be identified as Triangle
- However, if there is a noise sample next to this unseen sample, then a 1-NN classifier will classify it wrongly as a Circle
- Larger $K$ can solve this problem
K-Nearest Neighbor (K-NN)

- Is larger K better?
- Obviously, this unseen sample will be identified as a Circle by a 5-NN classifier.
- But a 19-NN classifier will classify it as a Triangle.
- However, 5-NN classifier will classify it correctly.

K-NN: Characteristic

Advantages:
- Very simple
- All computations deferred until classification
  - No training is needed

Disadvantages:
- Difficult to determine K
- Affected by noisy training data
- Classification is time consuming
  - Need to calculate the distance between the unseen sample and each training sample.
Decision Tree (DT)

- One of the most widely used and practical methods for inductive inference
- Approximates discrete-valued functions (including disjunctions)

DT: Example

- Do we go to play tennis today?

If Outlook is Sunny AND Humidity is Normal  Yes
If Outlook is Overcast  Yes
If Outlook is Rain AND Wind is Weak  Yes
Other situation?  No

Humidity  Sunny  Overcast  Rain
High  Yes  No
Normal  No  Weak
Yes  Yes  Yes
DT: Example

- **Internal node** corresponds to a test
- **Branch** corresponds to a result of the test
- **Leaf node** assigns a classification result

DT: Classification

- **Decision Region:**
  - Internal nodes can be *univariate*
  - Only one feature is used
DT: Classification

- Internal nodes can be *multivariate*
  - More than one features are used
  - Shape of Decision Region is irregular

\[ ax_1 + bx_2 + c > 0 \]

DT: Learning Algorithm

- LOOP:
  1. Select the *best feature* \((A)\)
  2. For each value of \(A\), create new descendant of node
  3. Sort training samples to leaf nodes

- STOP when training samples perfectly classified
DT: Learning Algorithm

- **Observation**
  - For a given training set, many trees may code it without any error
  - Finding the *smallest tree* is a NP-hard problem

- Local search algorithm to find reasonable solutions
  - What is the best feature?

DT: Feature Measurement

- **Entropy** can be used as a feature measurement
  - Measure of uncertainty
  - Range: 0 - 1
  - Smaller value, less uncertainty

\[
H(X) = - \sum_{i=1}^{n} p(x_i) \log_2 p(x_i)
\]

where \(X\) is a random variable with \(n\) outcomes \(\{x_i: i = 1, \ldots, n\}\)

\(p(x)\) is the probability mass function of outcome \(x\).

if \(X \in \text{class } x_i\), then \(p(x_i) = 1\), and all other \(p(x_i) = 0\) for \(i \neq 1\).

Thus \(H(X) = 0\), the smallest value possible (no uncertainty).
DT: Feature Measurement

- **Information Gain**
  - Reduction in entropy (reduce uncertainty) due to sorting on a feature $A$

$$Gain(X, A) = H(X) - H(X | A)$$

Current entropy  Entropy after using feature $A$

DT: Example

Recall:

$$H(X) = - \sum_{i=1}^{n} p(x_i) \log_2 p(x_i)$$

$x_1 = \text{yes} \quad x_2 = \text{No}$

$$H(X) = - \sum_{i=1}^{2} p(x_i) \log_2 p(x_i)$$

$$= - \frac{9}{14} \log_2 \left( \frac{9}{14} \right) - \frac{5}{14} \log_2 \left( \frac{5}{14} \right)$$

$$= 0.410 + 0.531$$

$$= 0.941$$

Which feature is the best?

**Current:** $H(X) = 0.941$

Uncertainty is high w/o any sorting by feature
DT: Example

Let $A = \text{Outlook}$

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>01</td>
<td>Sunny</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
<td>No: 3</td>
</tr>
<tr>
<td>01</td>
<td>Sunny</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
<td>No: 3</td>
</tr>
<tr>
<td>02</td>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
<td>No: 2</td>
</tr>
<tr>
<td>03</td>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
<td>No: 2</td>
</tr>
<tr>
<td>04</td>
<td>Sunny</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
<td>No: 0</td>
</tr>
<tr>
<td>05</td>
<td>Sunny</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
<td>No: 0</td>
</tr>
<tr>
<td>06</td>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
<td>Yes: 4</td>
</tr>
<tr>
<td>07</td>
<td>Rain</td>
<td>High</td>
<td>High</td>
<td>Yes</td>
<td>Yes: 4</td>
</tr>
</tbody>
</table>

Recall:

$Gain(X, A) = H(X) - H(X \mid A)$

$H(X \mid A) = H(X \mid A = \text{sunny})P(A = \text{sunny}) + H(X \mid A = \text{Rain})P(A = \text{Rain}) + H(X \mid A = \text{overcast})P(A = \text{overcast})$

$H(X \mid \text{sunny}) = -rac{3}{5}\log_2\left(\frac{3}{5}\right) - rac{2}{5}\log_2\left(\frac{2}{5}\right) = 0.971$

$H(X \mid \text{Rain}) = -rac{2}{5}\log_2\left(\frac{2}{5}\right) - rac{3}{5}\log_2\left(\frac{3}{5}\right) = 0.971$

$H(X \mid \text{overcast}) = -\frac{0}{4}\log_2\left(\frac{0}{4}\right) - \frac{4}{4}\log_2\left(\frac{4}{4}\right) = 0$

$H(X \mid A) = 0.971\times\left(\frac{5}{14}\right) + 0.971\times\left(\frac{5}{14}\right) + 0\times\left(\frac{4}{14}\right)$

$= 0.694$
DT: Example

Outlook is the best feature and
Should be used as the first node

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>Hot</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>Hot</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>Cool</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>Mild</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>Cool</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>Mild</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>Mild</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>Hot</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>Mild</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

Current: \( H(X) = 0.941 \)

Similarly, for each feature
\[
H(X \mid \text{Outlook}) = 0.694
\]
\[
H(X \mid \text{Temperature}) = 0.911
\]
\[
H(X \mid \text{Humidity}) = 0.789
\]
\[
H(X \mid \text{Wind}) = 0.892
\]

Information Gain is:
\[
\text{Gain}(X, \text{Outlook}) = 0.247
\]
\[
\text{Gain}(X, \text{Temperature}) = 0.030
\]
\[
\text{Gain}(X, \text{Humidity}) = 0.152
\]
\[
\text{Gain}(X, \text{Wind}) = 0.049
\]

Recall:
\[
\text{Gain}(X, A) = H(X) - H(X \mid A)
\]

DT: Example

- Next Step
  - Repeat the steps for each sub-branch
  - Until there is no ambiguity
    (all samples are of the same class)

Outlook

- Sunny
- Rain
- Overcast

No: 3
Yes: 2

No: 2
Yes: 3

No: 0
Yes: 4

Continues to select next features

Done
DT: Continuous-Valued Feature

- So far, we handle features with nominal values
- How to build a decision tree whose features have continuous values?

<table>
<thead>
<tr>
<th>Day</th>
<th>Outlook</th>
<th>Temperature</th>
<th>Humidity</th>
<th>Wind</th>
<th>PlayTennis</th>
</tr>
</thead>
<tbody>
<tr>
<td>D1</td>
<td>Sunny</td>
<td>29.9</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D2</td>
<td>Sunny</td>
<td>28.2</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D3</td>
<td>Overcast</td>
<td>35.2</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D4</td>
<td>Rain</td>
<td>26.4</td>
<td>High</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D5</td>
<td>Rain</td>
<td>18.9</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D6</td>
<td>Rain</td>
<td>21.2</td>
<td>Normal</td>
<td>Strong</td>
<td>No</td>
</tr>
<tr>
<td>D7</td>
<td>Overcast</td>
<td>20.4</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D8</td>
<td>Sunny</td>
<td>24.4</td>
<td>High</td>
<td>Weak</td>
<td>No</td>
</tr>
<tr>
<td>D9</td>
<td>Sunny</td>
<td>17.0</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D10</td>
<td>Rain</td>
<td>25.1</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D11</td>
<td>Sunny</td>
<td>24.0</td>
<td>Normal</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D12</td>
<td>Overcast</td>
<td>24.5</td>
<td>High</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>D13</td>
<td>Overcast</td>
<td>27.7</td>
<td>Normal</td>
<td>Weak</td>
<td>Yes</td>
</tr>
<tr>
<td>D14</td>
<td>Rain</td>
<td>25.5</td>
<td>High</td>
<td>Strong</td>
<td>No</td>
</tr>
</tbody>
</table>

Accomplished by partitioning the continuous attribute value into a discrete set of intervals

In particular, a new Boolean feature $A_c$ ($A < c$) can be created

Question: how to select the best value for $c$
DT: Continuous-Valued Feature

- Recall, the objective is to minimize the entropy (or maximize the information gain)
- Entropy only needs to be evaluated between points of different classes

<table>
<thead>
<tr>
<th>Value</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>17.0</td>
<td>Yes</td>
</tr>
<tr>
<td>18.9</td>
<td>Yes</td>
</tr>
<tr>
<td>20.4</td>
<td>Yes</td>
</tr>
<tr>
<td>21.2</td>
<td>No</td>
</tr>
<tr>
<td>24.0</td>
<td>Yes</td>
</tr>
<tr>
<td>24.4</td>
<td>No</td>
</tr>
<tr>
<td>24.5</td>
<td>Yes</td>
</tr>
<tr>
<td>25.1</td>
<td>No</td>
</tr>
<tr>
<td>25.5</td>
<td>Yes</td>
</tr>
<tr>
<td>26.4</td>
<td>Yes</td>
</tr>
<tr>
<td>27.7</td>
<td>No</td>
</tr>
<tr>
<td>28.2</td>
<td>Yes</td>
</tr>
<tr>
<td>29.9</td>
<td>No</td>
</tr>
<tr>
<td>35.2</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Best $c$ is:

$$c^* = \arg \max_c \text{Gain}(X, A_c)$$

DT: Pruning

- Goal: Prevent overfitting and noisy data
- Two strategies for “pruning” a decision tree:
  - Post-pruning
    - Take a fully-grown decision tree and discard unreliable parts
  - Pre-pruning
    - Stop growing a branch when information becomes unreliable
DT: Pruning

Pre-pruning

- Stop splitting during training of a tree.
  
  Why? Try to avoid overfitting (ie, grow a full tree until every leaf node corresponds to a class)

- Statistical significance test
  
  - Stop growing the tree when there is no statistically significant association between any attribute and the class at a particular node

- Only statistically significant attributes were allowed to be selected by information gain procedure

- Chi-squared test is the most popular one

However, pre-pruning may stop too early

Classic example: XOR/Parity-problem

- No individual attribute exhibits any significant association to the class
- Structure is only visible in fully expanded tree
- Pre-pruning won’t expand the root node

However, XOR-type problems rare in practice
DT: Pruning

Post-pruning

- Prune after the full tree is built
- Algorithm
  - Split data into training and validation set
  - Do until further pruning is harmful
    - Evaluate impact of pruning each possible node (plus those below it) on the validation set
    - Greedily remove the one that most improves validation set accuracy

Example: Labor negotiations

- When the DT is trained, the validation set is applied for post-pruning
**DT: Pruning Post-pruning**

**Bad**

3 samples are wrong
Total Error: 6/50
(including the error in the left subtree)

**Good**

12 samples are wrong
Total Error: 15/50
(including the error in the left subtree)

Take the minimum (treat all samples are bad)
Therefore, the error is 6/50 after pruning this tree

---

**DT: Pruning Post-pruning**

**Bad**

8 samples are wrong
Total Error: 11/50
(including the error in the left subtree)

**Good**

12 samples are wrong
Total Error: 15/50
(including the error in the left subtree)

Take the minimum (treat all samples are bad)
Therefore, the error is 11/50 after pruning this tree
DT: Pruning

Post-pruning

- Example: Labor negotiations

![Diagram showing decision tree prunes](image-url)
**DT: Pruning**

**Post-pruning**

- Example: Labor negotiations

**Original Tree**

- **wage increase 1st year**
  - $\leq 2.5$
  - $> 2.5$
- **working hours per week**
  - $\leq 36$
  - $> 36$
- **statutory holidays**
  - $> 10$
  - $\leq 10$
- **health plan contribution**
  - none
  - half
  - full
- **Outcome**
  - bad
  - good

**Tree after post-pruning**

- **wage increase 1st year**
  - $\leq 2.5$
  - $> 2.5$
- **working hours per week**
  - $\leq 36$
  - $> 36$
- **statutory holidays**
  - $> 10$
  - $\leq 10$
- **health plan contribution**
  - none
  - half
  - full
- **Outcome**
  - bad
  - good

Error: 9/50

Error: 6/50

---

**DT: Rule Extraction from Tree**

- Each distinct path through the decision tree node produces a distinct rule

R1: IF (age > 38.5) AND (years-in-job > 2.5) THEN $\gamma = 0.8$
**DT: Rule Extraction from Tree**

- Each distinct path through the decision tree node produces a distinct rule

```
R1: IF (age > 38.5) AND (years-in-job > 2.5) THEN \( y = 0.8 \)
R2: IF (age > 38.5) AND (years-in-job ≤ 2.5) THEN \( y = 0.6 \)
R3: IF (age ≤ 38.5) AND (job-type='A') THEN \( y = 0.4 \)
R4: IF (age ≤ 38.5) AND (job-type='B') THEN \( y = 0.3 \)
R5: IF (age ≤ 38.5) AND (job-type='C') THEN \( y = 0.2 \)
```

**DT: Binary Split vs Multi-Split**

- Each decision outcome at a node is called a **Split**

- Multi-Split (two or more branches at a node)
  - Exhausts all information in that attribute
  - Nominal attribute is tested (at most) once on any path in the tree
  - Tree is hard to read

- Binary Split (always two branches at a node)
  - Numeric attribute may be tested several times along a path in the tree
DT: Characteristics

- Rule extraction from trees
  - A decision tree can be used for feature extraction (e.g. seeing which features are useful)
- Interpretability
  - Easy to understand
- Compact and fast classification method

SMC 2009
San Antonio, Texas, USA

Father of Fuzzy Logic
Lotfi A. Zadeh
Fuzzy Classification

- Till now, patterns are measured very precisely
  - E.g. Salmon is 43.2cm long
- However, in real world, many things can not be measured precisely. They are fuzzy

Fuzzy Logic

- E.g. Is he tall?

Conventional (Boolean) logic
- Yes / No
- E.g. He is tall

Fuzzy logic
- Partial truth
  - i.e. truth values between “completely true” and “completely false”
    - denoted by a value between 0 and 1
- E.g. He is really not very tall
Fuzzy Classification

Fuzzy Rule

- A fuzzy rule is a linguistic expression of causal dependencies between linguistic variables in form of if-then statements.

- General form of rule:
  \[
  \text{IF antecedent THEN consequence}
  \]

- It is called fuzzy rule if either antecedent or consequence is a fuzzy term

- Example:
  - If temperature is cold and oil price is cheap
  - Then heating is high

  **Linguistic variables**
  **Linguistic values**

Fuzzy Classification: Fuzzy Rule

Example

- A simple two-input one-output problem that includes three rules:

  Rule: 1 IF x is A3 OR y is B1 THEN z is C1
  Rule: 2 IF x is A2 AND y is B2 THEN z is C2
  Rule: 3 IF x is A1 THEN z is C3

- Real-life example for these kinds of rules:

  Rule: 1 IF project_funding is adequate OR project_staffing is small THEN risk is low
  Rule: 2 IF project_funding is marginal AND project_staffing is large THEN risk is normal
  Rule: 3 IF project_funding is inadequate THEN risk is high
Fuzzy Classification System

1. Fuzzification

- The first step is to take the crisp inputs and determine the degree to which these inputs belong to each of the appropriate fuzzy sets.

<table>
<thead>
<tr>
<th>Person</th>
<th>Height</th>
</tr>
</thead>
<tbody>
<tr>
<td>Billy</td>
<td>3' 2&quot;</td>
</tr>
<tr>
<td>Yoke</td>
<td>5' 5&quot;</td>
</tr>
<tr>
<td>Drew</td>
<td>5' 9&quot;</td>
</tr>
<tr>
<td>Erik</td>
<td>5'10&quot;</td>
</tr>
<tr>
<td>Mark</td>
<td>6’1&quot;</td>
</tr>
<tr>
<td>Kareem</td>
<td>7’ 2”</td>
</tr>
</tbody>
</table>

The degree of truth of the statement “Mark is TALL” is 0.54.
Fuzzy Classification System: 1. Fuzzification

Fuzzy Set

- Crisp Set
  - The characteristic function assigns a number 1 or 0 to each element in the set
    - Depending on whether the element is in the subset A or not

- Fuzzy Set
  - The membership function is a graphical representation of the magnitude of participation of each input
    - Associates weighting with each of the inputs that are processed

Standard Membership Functions

- Z function
- Π function
- S function
- Singleton
- Triangle
- Trapezoidal
Fuzzy Classification System: 1. Fuzzification

Example

Fuzzy Classification System: 2. Rule Evaluation

- The second step is to apply the fuzzified inputs to the antecedents of the fuzzy rules.
- If a given fuzzy rule has multiple antecedents, fuzzy operator is used.
Fuzzy Classification System: 2. Rule Evaluation

Operations on Fuzzy Sets

\[
\begin{align*}
\text{AND} & : \mu_{A \cap B}(x) = \min \{\mu_A(x), \mu_B(x)\} \\
\text{OR} & : \mu_{A \cup B}(x) = \max \{\mu_A(x), \mu_B(x)\} \\
\text{NOT} & : \mu_{\sim A}(x) = 1 - \mu_A(x)
\end{align*}
\]

Properties

- The following rules which are common in classical set theory also apply to Fuzzy Logic

- **De Morgan’s Law**
  \[
  \begin{align*}
  (A \cap B) & = \overline{A} \cup \overline{B} \\
  (A \cup B) & = \overline{A} \cap \overline{B}
  \end{align*}
  \]

- **Associativity**
  \[
  \begin{align*}
  (A \cap B) \cap C & = A \cap (B \cap C) \\
  (A \cup B) \cup C & = A \cup (B \cup C)
  \end{align*}
  \]

- **Commutativity**
  \[
  \begin{align*}
  A \cap B & = B \cap A \\
  A \cup B & = B \cup A
  \end{align*}
  \]

- **Distributivity**
  \[
  \begin{align*}
  A \cap (B \cup C) & = (A \cap B) \cup (A \cap C) \\
  A \cup (B \cap C) & = (A \cup B) \cap (A \cup C)
  \end{align*}
  \]
Fuzzy Classification System: 2. Rule Evaluation

Example

Rule: 1 IF x is A3 OR y is B1 THEN z is C1

Rule: 2 IF x is A2 AND y is B2 THEN z is C2

Rule: 3 IF x is A1 THEN z is C3

Fuzzy Classification System

3. Defuzzification

- The final output of a fuzzy classification system has to be a class
- Back to our example:
  - After the rule evaluation:
    - C1 0.1
    - C2 0.2
    - C3 0.5
  - Therefore, the result is C3
- Take the maximum of all rules describing a class