CMPS 4420

Advanced Database Systems

Dr. Chengwei Lei
CEECS
California State University, Bakersfield
Supervised Learning
Basic concepts
An example application

- An emergency room in a hospital measures 17 variables (e.g., blood pressure, age, etc) of newly admitted patients.
- **A decision is needed**: whether to put a new patient in an intensive-care unit.
- Due to the high cost of ICU, those patients who may survive less than a month are given higher priority.
- **Problem**: to predict high-risk patients and discriminate them from low-risk patients.
Another application

- A credit card company receives thousands of applications for new cards. Each application contains information about an applicant,
  - age
  - Marital status
  - annual salary
  - outstanding debts
  - credit rating
  - etc.
- **Problem**: to decide whether an application should be approved, or to classify applications into two categories, approved and not approved.
Machine learning and our focus

- Like human learning from past experiences.
- A computer does not have “experiences”.
- A computer system learns from data, which represent some “past experiences” of an application domain.
- **Our focus:** learn a target function that can be used to predict the values of a discrete class attribute, e.g., approve or not-approved, and high-risk or low risk.
- The task is commonly called: Supervised learning, classification, or inductive learning.
The data and the goal

- **Data:** A set of data records (also called examples, instances or cases) described by
  - **k attributes:** $A_1, A_2, \ldots, A_k$.
  - **a class:** Each example is labelled with a pre-defined class.

- **Goal:** To learn a classification model from the data that can be used to predict the classes of new (future, or test) cases/instances.
An example: data (loan application)

<table>
<thead>
<tr>
<th>ID</th>
<th>Age</th>
<th>Has_Job</th>
<th>Own_House</th>
<th>Credit_Rating</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>good</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>young</td>
<td>true</td>
<td>false</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>young</td>
<td>true</td>
<td>true</td>
<td>fair</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>middle</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>middle</td>
<td>false</td>
<td>false</td>
<td>good</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>middle</td>
<td>true</td>
<td>true</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>middle</td>
<td>false</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>middle</td>
<td>true</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>old</td>
<td>false</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>old</td>
<td>false</td>
<td>true</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>old</td>
<td>true</td>
<td>false</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>old</td>
<td>true</td>
<td>false</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>15</td>
<td>old</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
</tbody>
</table>
An example: the learning task

- Learn a classification model from the data
- Use the model to classify future loan applications into
  - Yes (approved) and
  - No (not approved)
- What is the class for following case/instance?

<table>
<thead>
<tr>
<th>Age</th>
<th>Has_Job</th>
<th>Own_house</th>
<th>Credit-Rating</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>young</td>
<td>false</td>
<td>false</td>
<td>good</td>
<td>?</td>
</tr>
</tbody>
</table>
Supervised vs. unsupervised Learning

- **Supervised learning**: classification is seen as supervised learning from examples.
  - **Supervision**: The data (observations, measurements, etc.) are labeled with pre-defined classes. It is like that a “teacher” gives the classes (supervision).
  - Test data are classified into these classes too.

- **Unsupervised learning (clustering)**
  - Class labels of the data are unknown
  - Given a set of data, the task is to establish the existence of classes or clusters in the data
Supervised learning process: two steps

- **Learning (training)**: Learn a model using the training data
- **Testing**: Test the model using unseen test data to assess the model accuracy

\[
Accuracy = \frac{\text{Number of correct classifications}}{\text{Total number of test cases}},
\]

![Diagram of supervised learning process]

Step 1: Training

Step 2: Testing
What do we mean by learning?

- Given
  - a data set $D$,
  - a task $T$, and
  - a performance measure $M$,

a computer system is said to learn from $D$ to perform the task $T$ if after learning the system’s performance on $T$ improves as measured by $M$.

- In other words, the learned model helps the system to perform $T$ better as compared to no learning.
An example

- **Data**: Loan application data
- **Task**: Predict whether a loan should be approved or not.
- **Performance measure**: accuracy.

**No learning**: classify all future applications (test data) to the majority class (i.e., *Yes*):

- Accuracy = $9/15 = 60\%$.

- We can do better than 60% with learning.
Assumption: The distribution of training examples is identical to the distribution of test examples (including future unseen examples).

- In practice, this assumption is often violated to certain degree.
- Strong violations will clearly result in poor classification accuracy.
- To achieve good accuracy on the test data, training examples must be sufficiently representative of the test data.
Decision tree induction
Introduction

- Decision tree learning is one of the most widely used techniques for classification.
  - Its classification accuracy is competitive with other methods, and
  - it is very efficient.
- The classification model is a tree, called decision tree.
The loan data (reproduced)

<table>
<thead>
<tr>
<th>ID</th>
<th>Age</th>
<th>Has_Job</th>
<th>Own_House</th>
<th>Credit_Rating</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>good</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>young</td>
<td>true</td>
<td>false</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>young</td>
<td>true</td>
<td>true</td>
<td>fair</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>middle</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>middle</td>
<td>false</td>
<td>false</td>
<td>good</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>middle</td>
<td>true</td>
<td>true</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>middle</td>
<td>false</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>middle</td>
<td>false</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>old</td>
<td>false</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>old</td>
<td>false</td>
<td>true</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>old</td>
<td>true</td>
<td>false</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>old</td>
<td>true</td>
<td>false</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>15</td>
<td>old</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
</tbody>
</table>
A decision tree from the loan data

- Decision nodes and leaf nodes (classes)
Use the decision tree

<table>
<thead>
<tr>
<th>Age</th>
<th>Has_Job</th>
<th>Own_house</th>
<th>Credit-Rating</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>young</td>
<td>false</td>
<td>false</td>
<td>good</td>
<td>No</td>
</tr>
</tbody>
</table>
Is the decision tree unique?

- **No.** Here is a simpler tree.
- **We want** smaller tree and accurate tree.
  - Easy to understand and perform better.

- Finding the best tree is NP-hard.
- All current tree building algorithms are heuristic algorithms
A decision tree can be converted to a set of rules.

Each path from the root to a leaf is a rule.

Own_house = true → Class = Yes [sup=6/15, conf=6/6]
Own_house = false, Has_job = true → Class = Yes [sup=5/15, conf=5/5]
Own_house = false, Has_job = false → Class = No [sup=4/15, conf=4/4]
Algorithm for decision tree learning

- Basic algorithm (a greedy **divide-and-conquer** algorithm)
  - Assume attributes are categorical now (continuous attributes can be handled too)
  - Tree is constructed in a **top-down recursive manner**
  - At start, all the training examples are at the root
  - Examples are partitioned recursively based on selected attributes
  - Attributes are selected on the basis of an impurity function (e.g., **information gain**)

- Conditions for stopping partitioning
  - All examples for a given node belong to the same class
  - There are no remaining attributes for further partitioning – majority class is the leaf
  - There are no examples left
Algorithm decisionTree(D, A, T)
1. if D contains only training examples of the same class c_j ∈ C then
   make T a leaf node labeled with class c_j;
2. else if A = ∅ then
   make T a leaf node labeled with c_j, which is the most frequent class in D
3. else // D contains examples belonging to a mixture of classes. We select a single
   // attribute to partition D into subsets so that each subset is purer
   p_0 = impurityEval-1(D);
4. for each attribute A_i ∈ {A_1, A_2, ..., A_k} do
   p_i = impurityEval-2(A_i, D)
5. end
6. Select A_g ∈ {A_1, A_2, ..., A_k} that gives the biggest impurity reduction, computed using p_0 - p_g;
7. if p_0 - p_g < threshold then // A_g does not significantly reduce impurity p_0
   make T a leaf node labeled with c_j, the most frequent class in D.
8. else // A_g is able to reduce impurity p_0
   Make T a decision node on A_g;
   Let the possible values of A_g be v_1, v_2, ..., v_m. Partition D into m disjoint subsets D_1, D_2, ..., D_m based on the m values of A_g.
9. for each D_j in {D_1, D_2, ..., D_m} do
   if D_j ≠ ∅ then
      create a branch (edge) node T_j for v_j as a child node of T;
   decisionTree(D_j, A-{A_g}, T_j) // A_g is removed
10. end
11. end
12. end
Choose an attribute to partition data

- The **key** to building a decision tree - which attribute to choose in order to branch.
- The objective is to reduce impurity or uncertainty in data as much as possible.
  - A subset of data is **pure** if all instances belong to the same class.
- We can choose the attribute with the maximum **Information Gain** or **Gain Ratio** based on information theory.
The loan data (reproduced)

<table>
<thead>
<tr>
<th>ID</th>
<th>Age</th>
<th>Has_Job</th>
<th>Own_House</th>
<th>Credit_Rating</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>2</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>good</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>young</td>
<td>true</td>
<td>false</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>4</td>
<td>young</td>
<td>true</td>
<td>true</td>
<td>fair</td>
<td>Yes</td>
</tr>
<tr>
<td>5</td>
<td>young</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>6</td>
<td>middle</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
<tr>
<td>7</td>
<td>middle</td>
<td>false</td>
<td>false</td>
<td>good</td>
<td>No</td>
</tr>
<tr>
<td>8</td>
<td>middle</td>
<td>true</td>
<td>true</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>9</td>
<td>middle</td>
<td>false</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>10</td>
<td>middle</td>
<td>false</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>11</td>
<td>old</td>
<td>false</td>
<td>true</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>12</td>
<td>old</td>
<td>false</td>
<td>true</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>13</td>
<td>old</td>
<td>true</td>
<td>false</td>
<td>good</td>
<td>Yes</td>
</tr>
<tr>
<td>14</td>
<td>old</td>
<td>true</td>
<td>false</td>
<td>excellent</td>
<td>Yes</td>
</tr>
<tr>
<td>15</td>
<td>old</td>
<td>false</td>
<td>false</td>
<td>fair</td>
<td>No</td>
</tr>
</tbody>
</table>
Two possible roots, which is better?

- Fig. (B) seems to be better.