Outline

Previous class
Ch 8: Decision Trees
This class:
Ch 9: Decision Trees
CHAPTER 9: Decision Trees

Slides from: Blaž Zupan and Ivan Bratko
magix.fri.uni-lj.si/predavanja/uisp
Color?
- red
  - Dot?
    - yes
    - Outline?
      - solid
      - dashed
  - green
    - Dot?
      - no
      - Outline?
        - solid
        - dashed
- yellow
  - Dot?
    - no
    - Outline?
      - solid
      - dashed
Classification Trees

- What is the good split function?
- Use Impurity measure
- Assume $N_m$ training samples reach node $m$
  - $N^i_m$ of $N_m$ belong to class $C_i$, with $\sum_i N^i_m = N_m$.
  
  $$\hat{P}(C_i|\mathbf{x}, m) \equiv p^i_m = \frac{N^i_m}{N_m}$$

- Node $m$ is pure if for all classes either 0 or 1

---

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)
Entropy

- Measure amount of uncertainty on a scale from 0 to 1
- Example: 2 events
- If $p_1 = p_2 = 0.5$, entropy is 1 which is maximum uncertainty
- If $p_2 = 1 = 1 - p_1$, entropy is 0, which is no uncertainty (Eq 9.3)

$$I_m = - \sum_{i=1}^{K} p^i_m \log_2 p^i_m$$
Entropy
Best Split

- Node is impure, need to split more
- Have several split criteria (coordinates), have to choose optimal
- Minimize impurity (uncertainty) after split
- Stop when impurity is small enough
  - Zero stop impurity => complex tree with large variance
  - Larger stop impurity => small tress but large bias
Best Split

- Impurity after split: $N_{mj}$ of $N_m$ take branch $j$.
- $N_{mj}$ belong to $C_i$

$$\hat{P}(C_i | x, m, j) = p_{mj}^i = \frac{N_{mi}^i}{N_{mj}}$$

(Eq 9.8)

$$I'_m = -\sum_{j=1}^{n} \sum_{i=1}^{K} p_{mj}^i \log_2 p_{mj}^i$$

- Find the variable and split that min impurity
  - among all variables
  - split positions for numeric variables

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)
Information Gain

\[ I = 0.940 \]
\[ I(\text{red}) = 0.971 \text{ bits} \]
\[ I(\text{green}) = 0.971 \text{ bits} \]
\[ I(\text{yellow}) = 0.0 \text{ bits} \]

\[ \text{Gain}(\text{Color}) = I - I_{res}(\text{Color}) = 0.940 - 0.694 = 0.246 \text{ bits} \]
GenerateTree($\mathcal{X}$)

If NodeEntropy($\mathcal{X}$) < $\theta_I$ /* eq. 9.3
Create leaf labelled by majority class in $\mathcal{X}$
Return

$i \leftarrow$ SplitAttribute($\mathcal{X}$)
For each branch of $x_i$
Find $\mathcal{X}_i$ falling in branch
GenerateTree($\mathcal{X}_i$)

SplitAttribute($\mathcal{X}$)
MinEnt$\leftarrow$ MAX
For all attributes $i = 1, \ldots, d$

If $x_i$ is discrete with $n$ values
Split $\mathcal{X}$ into $\mathcal{X}_1, \ldots, \mathcal{X}_n$ by $x_i$

$e \leftarrow$ SplitEntropy($\mathcal{X}_1, \ldots, \mathcal{X}_n$) /* eq. 9.8 */
If $e < $MinEnt MinEnt$\leftarrow e$; bestf$\leftarrow i$

Else /* $x_i$ is numeric */
For all possible splits
Split $\mathcal{X}$ into $\mathcal{X}_1, \mathcal{X}_2$ on $x_i$

$e \leftarrow$ SplitEntropy($\mathcal{X}_1, \mathcal{X}_2$)
If $e <$MinEnt MinEnt$\leftarrow e$; bestf$\leftarrow i$

Return bestf
Regression Trees

- Value not a label in a leaf nodes
- Need other impurity measure
- Use Average Error

\[ b_m(x) = \begin{cases} 
1 & \text{if } x \in X_m : x \text{ reaches node } m \\
0 & \text{otherwise} 
\end{cases} \]

\[ E_m = \frac{1}{N_m} \sum_t \left( r^t - g_m \right)^2 b_m(x^t) \]

\[ g_m = \frac{\sum_t b_m(x^t)r^t}{\sum_t b_m(x^t)} \]
Regression Trees

- After splitting:

\[ b_{mj}(x) = \begin{cases} 
1 & \text{if } x \in X_{mj} : x \text{ reaches node } m \text{ and branch } j \\
0 & \text{otherwise} 
\end{cases} \]

\[ E'_m = \frac{1}{N_m} \sum_j \sum_t (r^t - g_{mj})^2 b_{mj}(x^t) \quad g_{mj} = \frac{\sum_t b_{mj}(x^t) r^t}{\sum_t b_{mj}(x^t)} \]
Example
Pruning Trees

- Number of data instances reach a node is small
  - Less than 5% of training data
  - Don’t want to split further regardless of impurity
- Remove subtrees for better generalization
  - Prepruning: Early stopping
  - Postpruning: Grow the whole tree then prune subtrees
    - Set aside pruning set
    - Make sure pruning does not significantly increase error
Decision Trees and Feature Extraction

• Univariate Tree uses only certain variable
• Some variables might not get used
• Features closer to the root have greater importance

Based on E. Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)
Interpretability

- Conditions that are simple to understand
- Path from the root => one conjunction of test
- All paths can be defined using set of IF_THEN rules
  - Form a rule base
- Percentage of training data covered by the rule
  - Rule support
- Tool for a Knowledge Extraction
- Can be verified by experts
Rule Extraction from Trees

C4.5 Rules
(Quinlan, 1993)

\[
\text{R1: IF (age} \geq 38.5 \text{) AND (years-in-job} \geq 2.5 \text{) THEN } y = 0.8 \\
\text{R2: IF (age} \geq 38.5 \text{) AND (years-in-job} \leq 2.5 \text{) THEN } y = 0.6 \\
\text{R3: IF (age} \leq 38.5 \text{) AND (job-type} = \text{‘A’} \text{) THEN } y = 0.4 \\
\text{R4: IF (age} \leq 38.5 \text{) AND (job-type} = \text{‘B’} \text{) THEN } y = 0.3 \\
\text{R5: IF (age} \leq 38.5 \text{) AND (job-type} = \text{‘C’} \text{) THEN } y = 0.2
\]
Learning Rules

- Rule induction is similar to tree induction **but**
  - tree induction is breadth-first,
  - rule induction is depth-first; one rule at a time

- Rule set contains rules; rules are conjunctions of terms
  - A rule **covers** an example if all terms of the rule evaluate to true for the example.
  - A rule is said to **cover** an example if the example satisfies all the conditions of the rule.

- **Sequential covering:**
  - Generate rules one at a time until all positive examples are covered
Rule-Based Classifier

Classify records by using a collection of “if… then…” rules

Rule: \((\text{Condition}) \rightarrow y\)

where

\(\text{Condition}\) is a conjunctions of attributes
\(y\) is the class label

Examples of classification rules:

\((\text{Blood Type}=\text{Warm}) \land (\text{Lay Eggs}=\text{Yes}) \rightarrow \text{Birds}\)

\((\text{Taxable Income} < 50\text{K}) \land (\text{Refund}=\text{Yes}) \rightarrow \text{Evade}=\text{No}\)
### Rule-based Classifier

#### (Example)

<table>
<thead>
<tr>
<th>Name</th>
<th>Blood Type</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>human</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>python</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>reptiles</td>
</tr>
<tr>
<td>salmon</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>fishes</td>
</tr>
<tr>
<td>whale</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>frog</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>amphibians</td>
</tr>
<tr>
<td>komodo</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>reptiles</td>
</tr>
<tr>
<td>bat</td>
<td>warm</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>pigeon</td>
<td>warm</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>birds</td>
</tr>
<tr>
<td>cat</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>leopard shark</td>
<td>cold</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>fishes</td>
</tr>
<tr>
<td>turtle</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>reptiles</td>
</tr>
<tr>
<td>penguin</td>
<td>warm</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>birds</td>
</tr>
<tr>
<td>porcupine</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>eel</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>fishes</td>
</tr>
<tr>
<td>salamander</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>sometimes</td>
<td>amphibians</td>
</tr>
<tr>
<td>gila monster</td>
<td>cold</td>
<td>no</td>
<td>no</td>
<td>yes</td>
<td>reptiles</td>
</tr>
<tr>
<td>platypus</td>
<td>warm</td>
<td>no</td>
<td>no</td>
<td>no</td>
<td>mammals</td>
</tr>
<tr>
<td>owl</td>
<td>warm</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>birds</td>
</tr>
<tr>
<td>dolphin</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>mammals</td>
</tr>
<tr>
<td>eagle</td>
<td>warm</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>birds</td>
</tr>
</tbody>
</table>

**R1:** $(\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{yes}) \rightarrow \text{Birds}$

**R2:** $(\text{Give Birth} = \text{no}) \land (\text{Live in Water} = \text{yes}) \rightarrow \text{Fishes}$

**R3:** $(\text{Give Birth} = \text{yes}) \land (\text{Blood Type} = \text{warm}) \rightarrow \text{Mammals}$

**R4:** $(\text{Give Birth} = \text{no}) \land (\text{Can Fly} = \text{no}) \rightarrow \text{Reptiles}$

**R5:** $(\text{Live in Water} = \text{sometimes}) \rightarrow \text{Amphibians}$

---

From: Luke Huan 2006
Application of Rule-Based Classifier

- A rule is said to cover an example if the example satisfies all the conditions of the rule.

R1: (Give Birth = no) ∧ (Can Fly = yes) → Birds
R2: (Give Birth = no) ∧ (Live in Water = yes) → Fishes
R3: (Give Birth = yes) ∧ (Blood Type = warm) → Mammals
R4: (Give Birth = no) ∧ (Can Fly = no) → Reptiles
R5: (Live in Water = sometimes) → Amphibians

<table>
<thead>
<tr>
<th>Name</th>
<th>Blood Type</th>
<th>Give Birth</th>
<th>Can Fly</th>
<th>Live in Water</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>hawk</td>
<td>warm</td>
<td>no</td>
<td>yes</td>
<td>no</td>
<td>?</td>
</tr>
<tr>
<td>grizzly bear</td>
<td>warm</td>
<td>yes</td>
<td>no</td>
<td>no</td>
<td>?</td>
</tr>
</tbody>
</table>

The rule R1 covers a hawk => Bird
The rule R3 covers the grizzly bear => Mammal

From: Luke Huan 2006
Example of Sequential Covering

(ii) Step 1

From: Luke Huan 2006
Example of Sequential Covering…

(iii) Step 2

(iv) Step 3

From: Luke Huan 2006
Ripper Algorithm

- There are two kinds of loop in Ripper algorithm (Cohen, 1995):
  - **Outer loop**: adding one rule at a time to the rule base
  - **Inner loop**: adding one condition at a time to the current rule
    - Conditions are added to the rule to maximize an information gain measure.
    - Conditions are added to the rule until it covers *no negative* example.
Ripper Algorithm

- In Ripper, conditions are added to the rule to
  - **Maximize** an information gain measure

\[
Gain(R', R) = s \cdot (\log_2 \frac{N'}{N'} - \log_2 \frac{N}{N})
\]

- \(R\) : the original rule
- \(R'\) : the candidate rule after adding a condition
- \(N\) (\(N'\)) : the number of instances that are covered by \(R\) (\(R'\))
- \(N_+\) (\(N'_+\)) : the number of true positives in \(R\) (\(R'\))
- \(s\) : the number of true positives in \(R\) and \(R'\) (after adding the condition)
- Until it covers no negative example
Stopping Criterion and Rule Pruning

- **Stopping criterion**
  - Compute the gain
  - If gain is not significant, discard the new rule

- **Rule Pruning**
  - Similar to post-pruning of decision trees
  - Reduced Error Pruning:
    - Remove one of the conjuncts in the rule
    - Compare error rate on validation set before and after pruning
    - If error improves, prune the conjunct
Summary of Direct Method

- Grow a single rule
- Remove Instances from rule
- Prune the rule (if necessary)
- Add rule to Current Rule Set
- Repeat
Direct Method: RIPPER

- For 2-class problem, choose one of the classes as positive class, and the other as negative class
  - Learn rules for positive class
  - Negative class will be default class
- For multi-class problem
  - Order the classes according to increasing class prevalence (fraction of instances that belong to a particular class)
  - Learn the rule set for smallest class first, treat the rest as negative class
  - Repeat with next smallest class as positive class
Multivariate Trees

\[ w_{11}x_1 + w_{12}x_2 + w_{10} = 0 \]

Based on E Alpaydın 2004 Introduction to Machine Learning © The MIT Press (V1.1)
Likelihood- vs. Discriminant-based Classification

- **Likelihood-based**: Assume a model for $p(x|C_i)$, use Bayes’ rule to calculate $P(C_i|x)$
  \[ g_i(x) = \log P(C_i|x) \]

- **Discriminant-based**: Assume a model for $g_i(x|\Phi_i)$; no density estimation
  - Estimating the boundaries is enough; no need to accurately estimate the densities inside the boundaries
  - Learning is the optimization of the model parameters $\Phi_i$ to maximize the classification accuracy on a given labeled training set.
Linear Discriminant

- **Linear discriminant:**
  \[ g_i(x | w_i, w_{i0}) = w_i^T x + w_{i0} = \sum_{j=1}^{d} w_{ij} x_j + w_{i0} \]

- **Advantages:**
  - **Simple:** \( O(d) \) space/computation
  - **Easy to understand:**
    - The final output is weighted sum of several factors.
  - **Accurate:**
    - When \( p(x | C_i) \) are Gaussian with a shared covariance matrix, the optimal discriminant is linear.
    - The linear discriminant can be used even when this assumption does not hold.
Generalized Linear Model

- **Quadratic discriminant:**
  \[ g_i(x \mid W_i, w_i, w_{i0}) = x^T W_i x + w_i^T x + w_{i0} \]

- **Higher-order (product) terms:**
  \[
  z_1 = x_1, \quad z_2 = x_2, \quad z_3 = x_1^2, \quad z_4 = x_2^2, \quad z_5 = x_1 x_2
  \]
  Map from \( x \) to \( z \) using **nonlinear basis functions** and use a linear discriminant in \( z \)-space
  \[
  g_i(x) = \sum_{j=1}^{k} w_j \phi_{ij}(x)
  \]
  \( \phi_{ij}(x) \) : basis functions
Two Classes

\[ g(x) = g_1(x) - g_2(x) \]
\[ = (w_1^T x + w_{10}) - (w_2^T x + w_{20}) \]
\[ = (w_1 - w_2)^T x + (w_{10} - w_{20}) \]
\[ = w^T x + w_0 \]

Choose
\[
\begin{cases} 
C_1 & \text{if } g(x) > 0 \\
C_2 & \text{otherwise}
\end{cases}
\]
Geometry

\[ g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + w_0 \]

- \( w_0 \): determines the location of the hyper-plane with respect to the origin
- \( \mathbf{w} \): determines its orientation
Multiple Classes

\[ g_i(x \mid w_i, w_{i0}) = w_i^T x + w_{i0} \]

Choose \( C_i \) if \( g_i(x) = \max_{j=1}^{k} g_j(x) \)

\( H_i : \) the hyperplane separate the examples of \( C_i \) from the examples of all other classes

Classes are linearly separable
Pairwise Separation

\[
g_{ij}(x | w_{ij}, w_{ij0}) = w_{ij}^T x + w_{ij0}
\]

\[
g_{ij}(x) = \begin{cases} 
> 0 & \text{if } x \in C_i \\
\leq 0 & \text{if } x \in C_j \\
\text{don't care} & \text{otherwise}
\end{cases}
\]

Choose \( C_i \) if \( \forall j \neq i, g_{ij}(x) > 0 \)

\( H_{ij} \): the hyperplane separate the examples of \( C_i \) and the examples of \( C_j \)