Decision Trees

- Leaves: Denoted by values of the Goal variable
- Inner nodes: Denoted by an input attribute, for each possible value an edge denoted by the value

*Rozhodovací strom* pro daný cílový atribut $G$ je kořen tvořený $z$. 
Prediction of a New Example

- Start in the root
- According the value of the root attribute choose an edge
- Follow edges according current values to a leaf
- Predict the label of the leaf.

Rozhodovací strom pro daný cílový atribut $G$ je kořen tvořený $z$. 
Create a Classification Tree

decision_tree(data,G)

• if all G in the same class:
  • create a leaf labeled with this class

• else
  • select an split attribute A
  • if (no split possible) create a leaf return
  • for each possible value a of A
    – subtree=decision_tree(data[A==a,],G)
    – return a tree with the root A and all sub-trees.
Selecting the Split Attribute

- Miopic – highest decrease of an error measure:
  - $\hat{p}_{mk} = \frac{1}{N_m} \sum_{x_i \in R_m} I(y_i = k)$.
  - $m$-current region, $k$- element $G$

- Missclassification error (usually not used)
  - $err_{MISC} = 1 - \max_k(\hat{p}_{mk})$

- Cross-entropy, deviance:
  - $D = - \sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$

- Gini
  - $G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$

Cross-entropy and Gini similarly good.
Gini, Cross-entropy, Misclass. error

\[
2p(1 - p)
\]

\[
1 - \max(p, 1 - p)
\]

\[
-p \log p - (1 - p) \log (1 - p)
\]

**FIGURE 9.3.** Node impurity measures for two-class classification, as a function of the proportion \( p \) in class 2. Cross-entropy has been scaled to pass through (0.5, 0.5).

**Misclassification error:** \[
\frac{1}{N_m} \sum_{i \in R_m} I(y_i \neq k(m)) = 1 - \hat{p}_{mk(m)}
\]

**Gini index:** \[
\sum_{k \neq k'} \hat{p}_{mk} \hat{p}_{mk'} = \sum_{k=1}^{K} \hat{p}_{mk} (1 - \hat{p}_{mk})
\]

**Cross-entropy or deviance:** \[
-\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}
\]
Entropy

- Is a function, that:
  - Is 0 if all labels are equal
    - for example: YES, YES, YES, YES, YES, YES, YES, YES.
  - Is maximal if all labels are equally probable
    - for example: YES, YES, YES, NO, NO, NO, NO.
- Two step split leads to the same result as one split:
  \[ E([2,3,4]) = E([2,7]) + \frac{7}{9}E([3,4]) \]
- This fulfills only function
  \[ D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \]
numbers: #errors on validation dataset
Prunning Trees

• Decision trees are usually overfitted.

• We reduce the size of the tree to reduce test error.

• Different methods:
  • reduced error prunning on validation dataset
  • (J4.5 – estimate derived from train data)
  • cost complexity prunning, crossvalidation.
Reduced error pruning

- We need validation data, not used for training.
- Calculate prediction error on validation data.
- From the leaves up, select variant with smaller error:
  - keep inner node
  - replace the inner node with a leaf.
- We can consider to replace inner node by a subtree
  - computationally intensive
  - Weka: only on the most frequent path.
What Shell We Cut Off?
Cost Complexity Pruning

- Penalized error measure, parameter $\alpha$:
  \[
  C_\alpha(T) = \sum_{\text{leaf} = 1}^{\left| T \right|} err_{MISC}(\text{leaf}) + \alpha \left| T \right|
  \]
- Estimate this value on set hidden by crossvalidation.
- For every tree starting with $T$, then always prune the node less decreasing training misc. error.
- For each $\alpha$, an optimal subtree of $T$ is in this sequence.
Cost Complexity Prunning

- Learn a tree, create cost-complexity sequence.
  - We have a sequence of tree sizes and corresp. $\alpha$.
- In K-fold crossvalidation for each $\alpha$ we have K estimates of error,
- select the best $\alpha$. 

![Graph showing the relationship between tree size and misclassification rate]
Numerical Attributes, Classification

• Consider all binary splits.
• Select the one minimizing the weighted entropy after the split.

<table>
<thead>
<tr>
<th></th>
<th>64</th>
<th>65</th>
<th>68</th>
<th>69</th>
<th>70</th>
<th>71</th>
<th>72</th>
<th>75</th>
<th>80</th>
<th>81</th>
<th>83</th>
<th>85</th>
</tr>
</thead>
<tbody>
<tr>
<td>a1</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>no,</td>
<td>yes</td>
<td>yes</td>
<td>no</td>
<td>yes</td>
<td>yes</td>
</tr>
</tbody>
</table>

$E([9,5])=0.940$

• Split at 71.5:

$(6/14)*E([4,2])+(8/14)*E([5,3])=0.939.$
Numerical Attributes, Regression

- Binary splits, attribute may be selected repeatedly.
- CART: for each attribute, each split point:
  - calculate decrease in square error loss
  - select the best cut.

Hledá proměnnou \( j \) a bod řezu \( s \) na oblasti \( R_1(j,s) = \{X | X_j \leq s \} \) a \( R_2(j,s) = \{X | X_j > s \} \) takové, aby minimalizovaly

\[
\min_{j,s} = \left[ \min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \right]
\]

vnitřní minima jsou průměry, tj. \( \hat{c}_1 = \text{avg}(y_i | x_i \in R_1(j,s)) \).
Algorithm 8.1 Building a Regression Tree

1. Use recursive binary splitting to grow a large tree on the training data, stopping only when each terminal node has fewer than some minimum number of observations.

2. Apply cost complexity pruning to the large tree in order to obtain a sequence of best subtrees, as a function of $\alpha$.

3. Use $K$-fold cross-validation to choose $\alpha$. That is, divide the training observations into $K$ folds. For each $k = 1, \ldots, K$:
   (a) Repeat Steps 1 and 2 on all but the $k$th fold of the training data.
   (b) Evaluate the mean squared prediction error on the data in the left-out $k$th fold, as a function of $\alpha$.

   Average the results for each value of $\alpha$, and pick $\alpha$ to minimize the average error.

4. Return the subtree from Step 2 that corresponds to the chosen value of $\alpha$. 
Decision Trees

- Historically, the most popular method of machine learning, many (similar) algorithms, names:
  - CART, ID3, C4.5, J4.5.

Today
- very useful for understanding, interpretation,
- other methods (gam, forests, MARS...) give better predictions.
Missing Data

- Trees can handle missing data well.
- Ignore missing data – may be nothing left.

**Are values missing at random?**
- Not random: treat 'missing' as attribute value.
- Missing at random: 'Fill the value it.'
  - Split the instance to pieces with different value
  - weighted according the value probability
    - at the current level of the tree.
Missing Data - Example
Linear methods or Trees
Loss Matrix – Error Costs

- Sensitivity: $P(\text{predict disease} \mid \text{true disease})$
- Specificity: $P(\text{predict OK} \mid \text{true OK})$

- weights on leaves
  $$k(m) = \arg \min_k \sum_{\ell} L_{\ell k} \hat{p}_{m\ell}$$
  or better

- learning on weighted data.
Decision Rules

• Historically, specific algorithms, still useful in Inductive Logic Programming.

• The tree can be converted to rules AND SIMPLIFIED and rules ordered according to precision.

• Specific algorithm: BUMP HUNTING.
Algorithm 9.3 Patient Rule Induction Method.

1. Start with all of the training data, and a maximal box containing all of the data.

2. Consider shrinking the box by compressing one face, so as to peel off the proportion \( \alpha \) of observations having either the highest values of a predictor \( X_j \), or the lowest. Choose the peeling that produces the highest response mean in the remaining box. (Typically \( \alpha = 0.05 \) or 0.10.)

3. Repeat step 2 until some minimal number of observations (say 10) remain in the box.

4. Expand the box along any face, as long as the resulting box mean increases.

5. Steps 1–4 give a sequence of boxes, with different numbers of observations in each box. Use cross-validation to choose a member of the sequence. Call the box \( B_1 \).

6. Remove the data in box \( B_1 \) from the dataset and repeat steps 2–5 to obtain a second box, and continue to get as many boxes as desired.
Spam Example (ESL Book)

<table>
<thead>
<tr>
<th>Rule 1</th>
<th>Global Mean</th>
<th>Box Mean</th>
<th>Box Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>0.3931</td>
<td>0.9607</td>
<td>0.1413</td>
</tr>
<tr>
<td>Test</td>
<td>0.3958</td>
<td>1.0000</td>
<td>0.1536</td>
</tr>
</tbody>
</table>

Rule 1

\[
\begin{align*}
\text{ch!} & > 0.029 \\
\text{CAPAVE} & > 2.331 \\
\text{your} & > 0.705 \\
\text{1999} & < 0.040 \\
\text{CAPTOT} & > 79.50 \\
\text{edu} & < 0.070 \\
\text{re} & < 0.535 \\
\text{ch;} & < 0.030
\end{align*}
\]

<table>
<thead>
<tr>
<th>Rule 2</th>
<th>Remain Mean</th>
<th>Box Mean</th>
<th>Box Support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>0.2998</td>
<td>0.9560</td>
<td>0.1043</td>
</tr>
<tr>
<td>Test</td>
<td>0.2862</td>
<td>0.9264</td>
<td>0.1061</td>
</tr>
</tbody>
</table>

Rule 2

\[
\begin{align*}
\text{remove} & > 0.010 \\
\text{george} & < 0.110
\end{align*}
\]