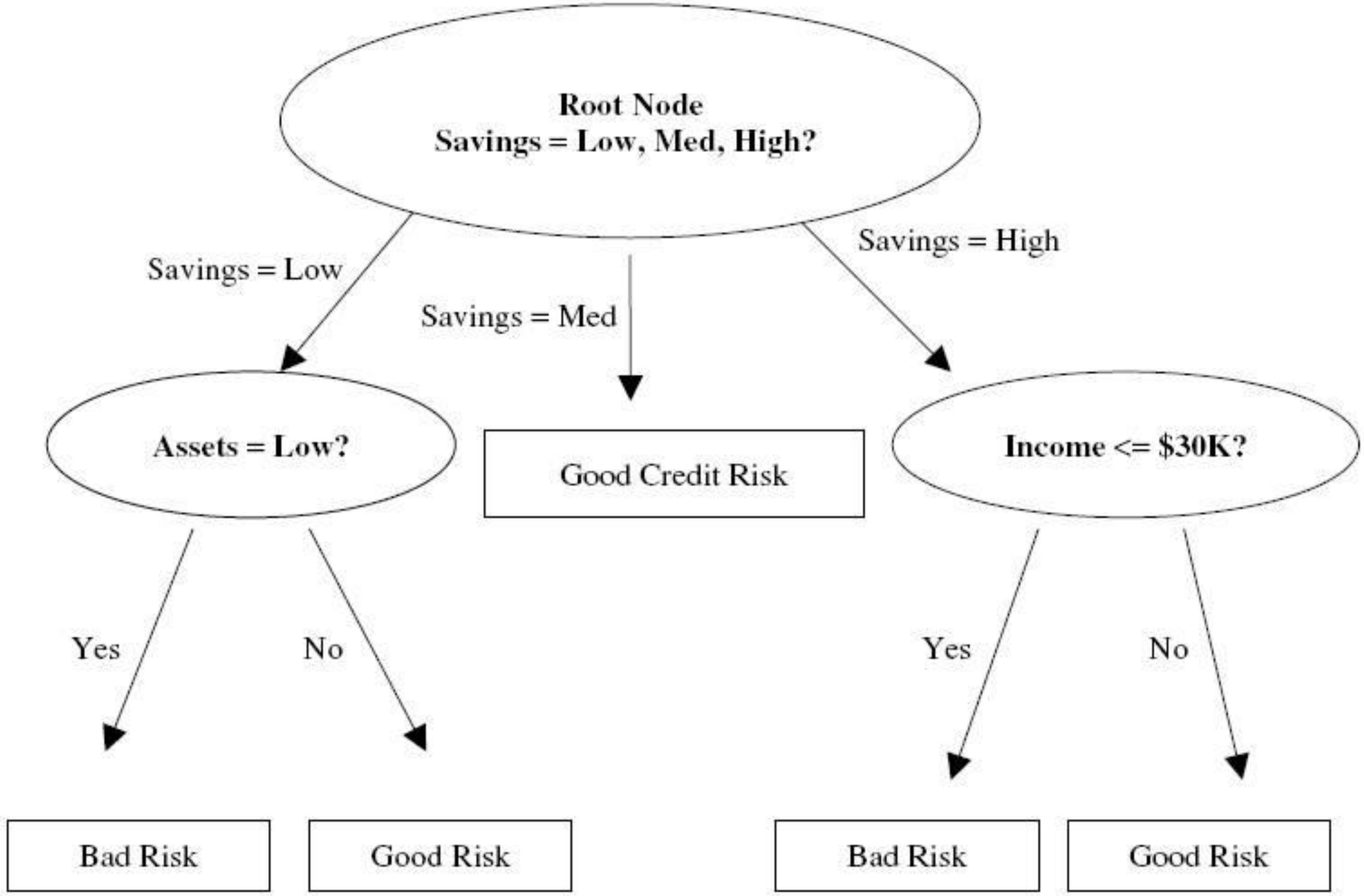
DECISION TREES

Pronalaženje skrivenog znanja

Bojan Furlan

DECISION TREES - Goal



DECISION TREES - Requirements

Creating Decision Trees

- Manual - Based on expert knowledge
- Automated - Based on training data (DM)

- Requirements so decision tree algorithms may be applied:
  - 1. A training data set must be supplied which provides the algorithm with the values of the target variable. (supervised learning)
  - 2. Training data set should be rich and varied
  - 3. The target attribute classes must be discrete (or discretized)

DECISION TREES - Properties

- **Issue #1**: Which attribute to take for a split?
  - Decision trees seek to create a set of leaf nodes that are as "pure" as possible
    - each of the records in a leaf node has the same classification.
  - This provide classification with the highest measure of confidence!
  - E.g. in the example above, the decision tree choose the *savings* attribute for the root node split. Why?
    - Because it partitions the training data set as "pure" as possible!

DECISION TREES - Properties

- **Issue #2**: When to stop splitting?
  - When there is no need for another decision node
    I. All of the records have been classified within same class.
    II. All splits are exhausted.
  - E.g. Why a leaf node and not another decision node for Savings=Med?
    - Because, all of the records with medium savings levels have been classified as good credit risks.

    => if customer has medium savings - predict good credit with 100% accuracy in the data set.
- Two algorithms for constructing decision trees:
  - Classification and regression trees (CART) algorithm
  - C4.5 algorithm

CLASSIFICATION AND REGRESSION TREES

- CART trees are binary, containing exactly **two** branches for each decision node.
- CART recursively partitions the records into subsets with same values for the target attribute.

CLASSIFICATION AND REGRESSION TREES

- Let Φ(s |t) be a measure of the "goodness" of a candidate split s at node t , where:

$$\Phi(s|t) = 2P_L P_R \sum_{j=1}^{\#\text{ classes}} |P(j|t_L) - P(j|t_R)|$$

$$t_L = \text{left child node of node } t$$

$$t_R = \text{right child node of node } t$$

$$P_L = \frac{\text{number of records at } t_L}{\text{number of records in training set}}$$

$$P_R = \frac{\text{number of records at } t_R}{\text{number of records in training set}}$$

$$P(j|t_L) = \frac{\text{number of class } j \text{ records at } t_L}{\text{number of records at } t}$$

$$P(j|t_R) = \frac{\text{number of class } j \text{ records at } t_R}{\text{number of records at } t}$$

- Then the optimal split maximizes this Φ(s |t) measure over all possible splits at node t .

CLASSIFICATION AND REGRESSION TREES

- Φ(s |t) is large when both of its main components are large: $2P_L P_R$ and $\sum_{j=1}^{\#\text{ classes}} |P(j|t_L) - P(j|t_R)|$

1. $2P_L P_R$ - Maximum value if child nodes are equal size (same support): E.g. 0.5*0.5 = 0.25 and 0.9*0.1= 0.09

2. Q (s |t)= $\sum_{j=1}^{\#\text{ classes}} |P(j|t_L) - P(j|t_R)|$

    - Maximum value if for each class the child nodes are completely uniform (pure).
    - Theoretical maximum value for Q (s|t) is k, where k is the number of classes for the target variable.

CART Example

Customer	Savings	Assets	Income ($1000s)	Credit Risk
1	Medium	High	75	Good
2	Low	Low	50	Bad
3	High	Medium	25	Bad
4	Medium	Medium	50	Good
5	Low	Medium	100	Good
6	High	High	25	Good
7	Low	Low	25	Bad
8	Medium	Medium	75	Good

Training Set of Records for Classifying Credit Risk

CART Example – Candidate Splits

Candidate Split	Left Child Node, t_L	Right Child Node, t_R
1	$Savings = low$	$Savings \in \{medium, high\}$
2	$Savings = medium$	$Savings \in \{low, high\}$
3	$Savings = high$	$Savings \in \{low, medium\}$
4	$Assets = low$	$Assets \in \{medium, high\}$
5	$Assets = medium$	$Assets \in \{low, high\}$
6	$Assets = high$	$Assets \in \{low, medium\}$
7	$Income \leq \$25{,}000$	$Income > \$25{,}000$
8	$Income \leq \$50{,}000$	$Income > \$50{,}000$
9	$Income \leq \$75{,}000$	$Income > \$75{,}000$

Candidate Splits for t = Root Node

- CART is restricted to binary splits

CART Primer

- Split 1. -> Savings=low (L-true, R-false)
  - Right:1,3,4,6,8
  - Left:2,5,7

- P_R=5/8 = 0.625 P_L=3/8=0.375 -> 2*$P_L P_R$=15/64=0.46875

- za j(klasu) = Bad
  - $P(Bad|t_R)$= 1/5=0.2
  - $P(Bad|t_L)$= 2/3=0.67
- za j(klasu) = Good
  - $P(Good|t_R)$= 4/5 = 0.8
  - $P(Good|t_L)$= 1/3 = 0.33
- $Q(s|t)$= |0.67-0.2|+|0.8-0.33| = 0.934

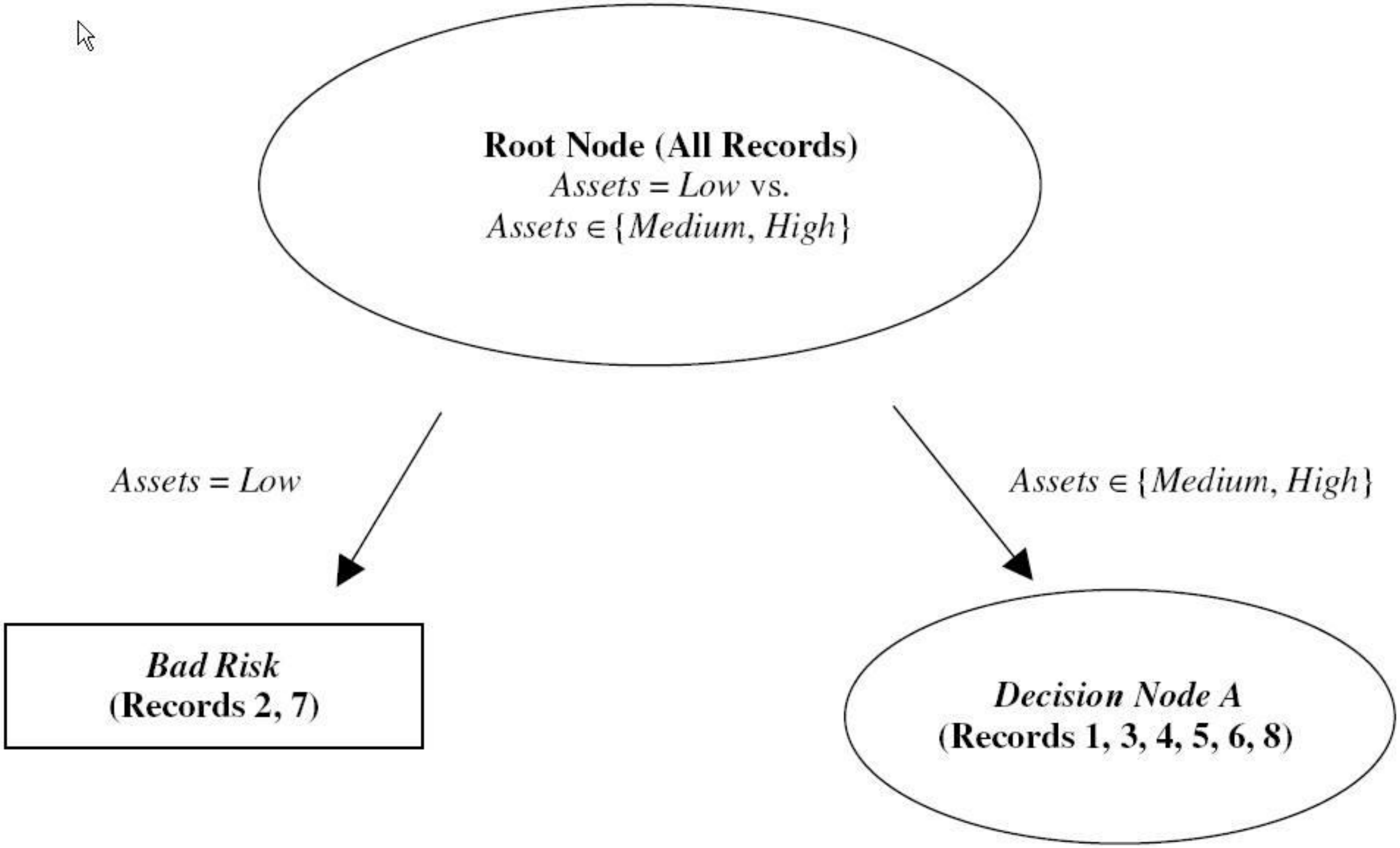
CART Example

Split	P_L	P_R	$P(j\|t_L)$	$P(j\|t_R)$	$2P_LP_R$	$Q(s\|t)$	$\Phi(s\|t)$
1	0.375	0.625	G: .333 B: .667	G: .8 B: .2	0.46875	0.934	0.4378
2	0.375	0.625	G: 1 B: 0	G: 0.4 B: 0.6	0.46875	1.2	0.5625
3	0.25	0.75	G: 0.5 B: 0.5	G: 0.667 B: 0.333	0.375	0.334	0.1253
4	0.25	0.75	G: 0 B: 1	G: 0.833 B: 0.167	0.375	1.667	0.6248
5	0.5	0.5	G: 0.75 B: 0.25	G: 0.5 B: 0.5	0.5	0.5	0.25
6	0.25	0.75	G: 1 B: 0	G: 0.5 B: 0.5	0.375	1	0.375
7	0.375	0.625	G: 0.333 B: 0.667	G: 0.8 B: 0.2	0.46875	0.934	0.4378
8	0.625	0.375	G: 0.4 B: 0.6	G: 1 B: 0	0.46875	1.2	0.5625
9	0.875	0.125	G: 0.571 B: 0.429	G: 1 B: 0	0.21875	0.858	0.1877

Values of Components of Optimality Measure Φ(s |t) for Each Candidate Split, for the Root Node

- For each candidate split, examine the values of the various components of the measure Φ(s |t).

CART Example - Tree



CART decision tree after initial split

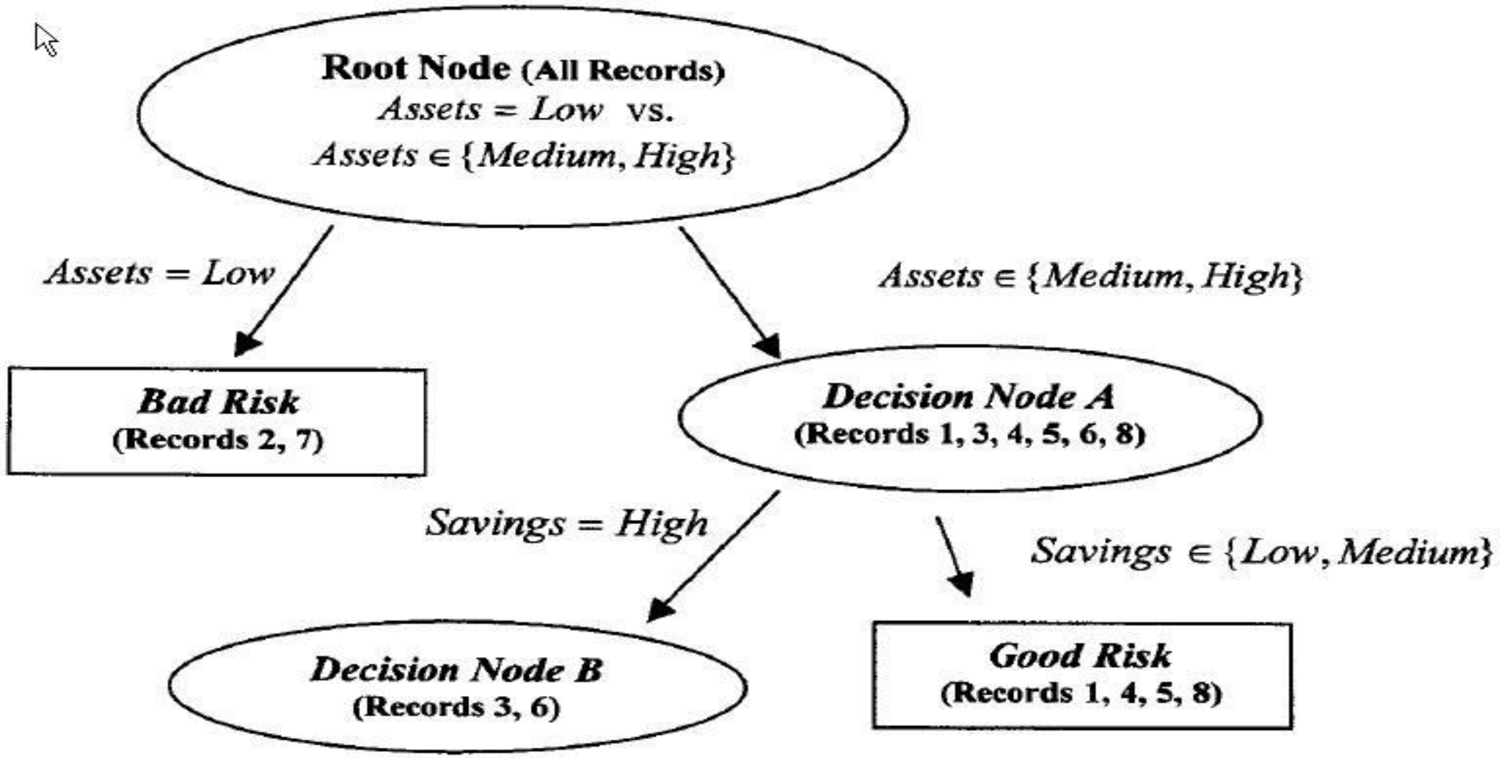
CART Example

Split	P_L	P_R	$P(j\|t_L)$	$P(j\|t_R)$	$2P_L P_R$	$Q(s\|t)$	$\Phi(s\|t)$
1	0.167	0.833	G: 1 B: 0	G: .8 B: .2	0.2782	0.4	0.1112
2	0.5	0.5	G: 1 B: 0	G: 0.667 B: 0.333	0.5	0.6666	0.3333
3	0.333	0.667	G: 0.5 B: 0.5	G: 1 B: 0	0.4444	1	0.4444
5	0.667	0.333	G: 0.75 B: 0.25	G: 1 B: 0	0.4444	0.5	0.2222
6	0.333	0.667	G: 1 B: 0	G: 0.75 B: 0.25	0.4444	0.5	0.2222
7	0.333	0.667	G: 0.5 B: 0.5	G: 1 B: 0	0.4444	1	0.4444
8	0.5	0.5	G: 0.667 B: 0.333	G: 1 B: 0	0.5	0.6666	0.3333
9	0.167	0.833	G: 0.8 B: 0.2	G: 1 B: 0	0.2782	0.4	0.1112

Values of Components of Optimality Measure $\Phi(s|t)$ for Each Candidate Split, for Decision Node A

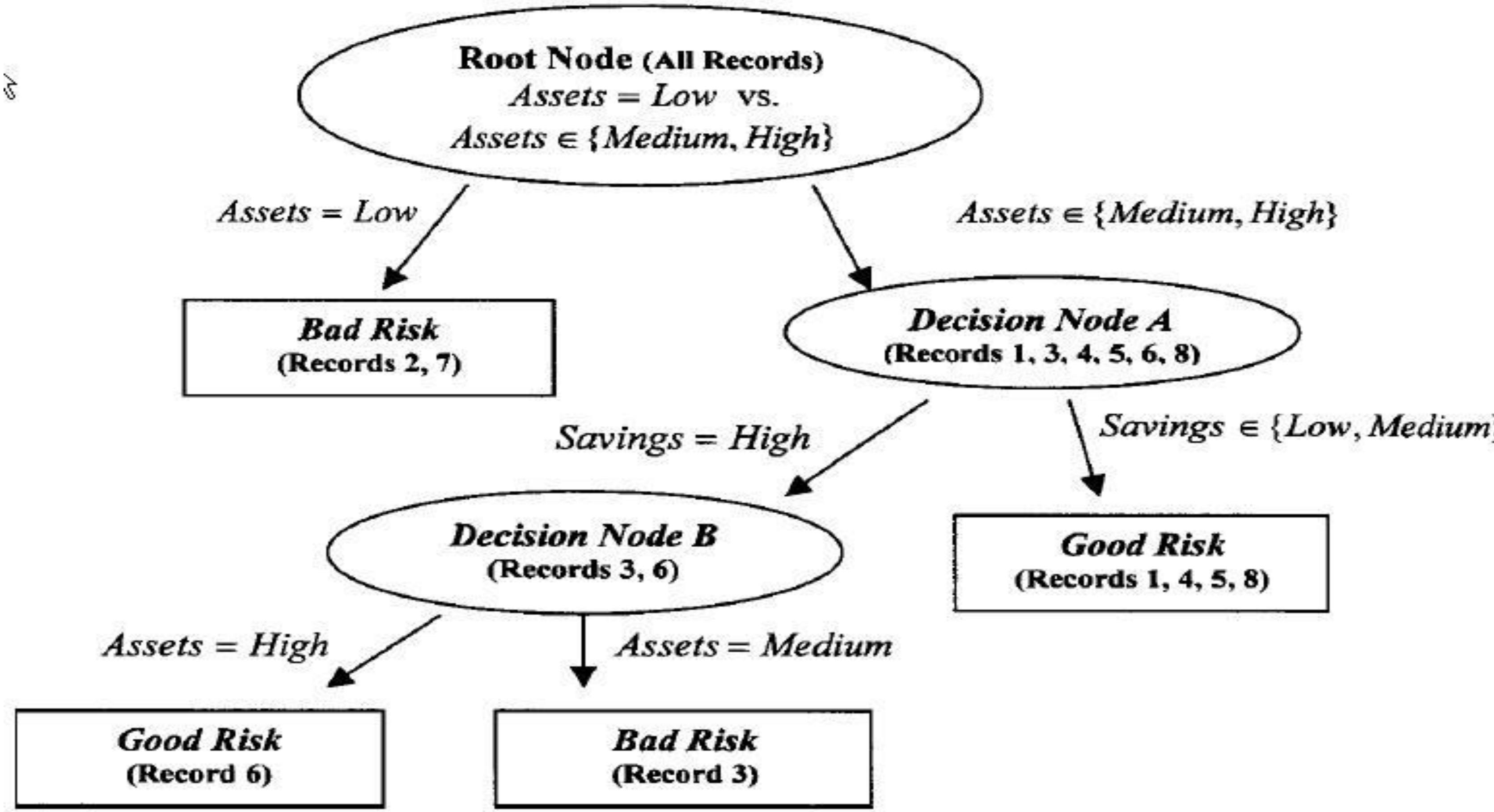
- Two candidate splits (3 and 7) share the highest value for $\Phi(s|t)$,0.4444.

CART Example - Tree



CART decision tree after decision node *A* split

CART Example - Tree



CART decision tree, fully grown form

CLASSIFICATION AND REGRESSION TREES

- Eventually, no decision nodes remain, and the “full tree” has been grown.
- Fully grown tree has the lowest error rate, but can result in <u>overfitting</u>.
- Pruning the tree will increase the <u>generalizability</u> of results.

DECISION TREES - purity

- Not all leaf nodes are homogeneous,
  which leads to a certain level of classification error.

Customer	Savings	Assets	Income	Credit Risk
004	High	Low	≤$30,000	Good
009	High	Low	≤$30,000	Good
027	High	Low	≤$30,000	Bad
031	High	Low	≤$30,000	Bad
104	High	Low	≤$30,000	Bad

Sample of Records That Cannot Lead to Pure Leaf Node

- When no further splits can be made,
  the decision tree algorithm stops growing new nodes.
- Decision tree may report that the classification for such customers is "bad," with 60% confidence

C4.5 ALGORITHM

- Differences between CART and C4.5:
  - Unlike CART, the C4.5 algorithm is not restricted to binary splits.
    - It produces a separate branch for each value of the categorical attribute.
  - C4.5 method for measuring node homogeneity is different from the CART.

C4.5 ALGORITHM - Measure

- We have a candidate split S, which partitions the training data set T into several subsets, $T_1, T_2, \ldots, T_k$.
- C4.5 uses the concept of *entropy reduction* to select the optimal split.
- entropy_reduction(S) = $H(T) - H_S(T)$, where entropy H(X) is:

$$H(X) = -\sum_{j} p_j \log_2(p_j)$$

- The weighted sum of the entropies for the individual subsets $T_1, T_2, \ldots, T_k$

$$H_S(T) = \sum_{i=1}^{k} P_i H_S(T_i)$$

- Where P_i represents the proportion of records in subset i .
- C4.5 chooses the optimal split - the split with greatest *entropy reduction*

C4.5 ALGORITHM

Customer	Savings	Assets	Income ($1000s)	Credit Risk
1	Medium	High	75	Good
2	Low	Low	50	Bad
3	High	Medium	25	Bad
4	Medium	Medium	50	Good
5	Low	Medium	100	Good
6	High	High	25	Good
7	Low	Low	25	Bad
8	Medium	Medium	75	Good

Training Set of Records for Classifying Credit Risk

Candidate Split	Child Nodes		
1	$Savings = low$	$Savings = medium$	$Savings = high$
2	$Assets = low$	$Assets = medium$	$Assets = high$
3	$Income \leq \$25{,}000$		$Income > \$25{,}000$
4	$Income \leq \$50{,}000$		$Income > \$50{,}000$
5	$Income \leq \$75{,}000$		$Income > \$75{,}000$

Candidate Splits at Root Node for C4.5 Algorithm

C4.5 ALGORITHM

- 5/8 records are classified as good credit risk and 3/8 are classified as bad credit risk
  the entropy before splitting is:

$$H(T) = -\sum_j p_j \log_2(p_j) = -\frac{5}{8}\log_2\left(\frac{5}{8}\right) - \frac{3}{8}\log_2\left(\frac{3}{8}\right) = 0.9544$$

- Compare the entropy of each candidate split against this H(T)=0.9544, to see which split results in the greatest reduction in entropy.

C4.5 ALGORITHM

- For candidate split 1 (savings):
  $P_{high} = \frac{2}{8}$, $P_{medium} = \frac{3}{8}$, $P_{low} = \frac{3}{8}$.
- Entropy for high savings is

$$-\frac{1}{2}\log_2\left(\frac{1}{2}\right) - \frac{1}{2}\log_2\left(\frac{1}{2}\right) = 1$$

- Entropy for medium is

$$-\frac{3}{3}\log_2\left(\frac{3}{3}\right) - \frac{0}{3}\log_2\left(\frac{0}{3}\right) = 0$$

- Entropy for low savings is

$$-\frac{1}{3}\log_2\left(\frac{1}{3}\right) - \frac{2}{3}\log_2\left(\frac{2}{3}\right) = 0.9183$$

C4.5 ALGORITHM

- We combine the entropies of these three and the proportions of the subsets Pi:

$$H_{\text{savings}}(T) = \tfrac{2}{8}(1) + \tfrac{3}{8}(0) + \tfrac{3}{8}(0.9183) = 0.5944$$

- Then the information gain represented by the split on the savings attribute is calculated as

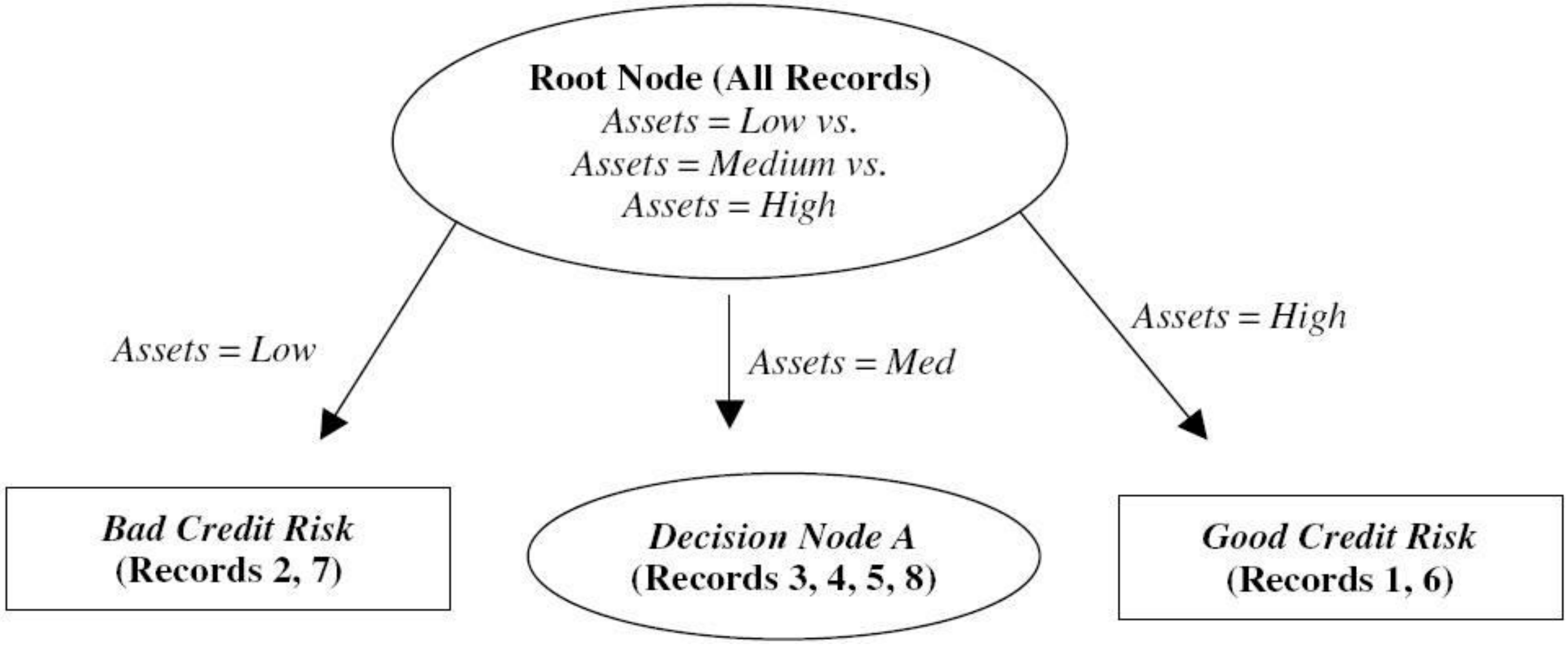
$$H(T) - H_{\text{savings}}(T) = 0.9544 - 0.5944 = 0.36$$

C4.5 ALGORITHM

Candidate Split	Child Nodes	Information Gain (Entropy Reduction)
1	$Savings = low$ $Savings = medium$ $Savings = high$	0.36 bits
2	$Assets = low$ $Assets = medium$ $Assets = high$	0.5487 bits
3	$Income \leq \$25{,}000$ $Income > \$25{,}000$	0.1588 bits
4	$Income \leq \$50{,}000$ $Income > \$50{,}000$	0.3475 bits
5	$Income \leq \$75{,}000$ $Income > \$75{,}000$	0.0923 bits

Information Gain for Each Candidate Split at the Root Node

C4.5 ALGORITHM



Partial decision tree resulting from C4.5's initial split

C4.5 ALGORITHM

Customer	Savings	Assets	Income ($1000s)	Credit Risk
3	High	Medium	25	Bad
4	Medium	Medium	50	Good
5	Low	Medium	100	Good
8	Medium	Medium	75	Good

Records Available at Decision Node A for Classifying Credit Risk

- the entropy before splitting is

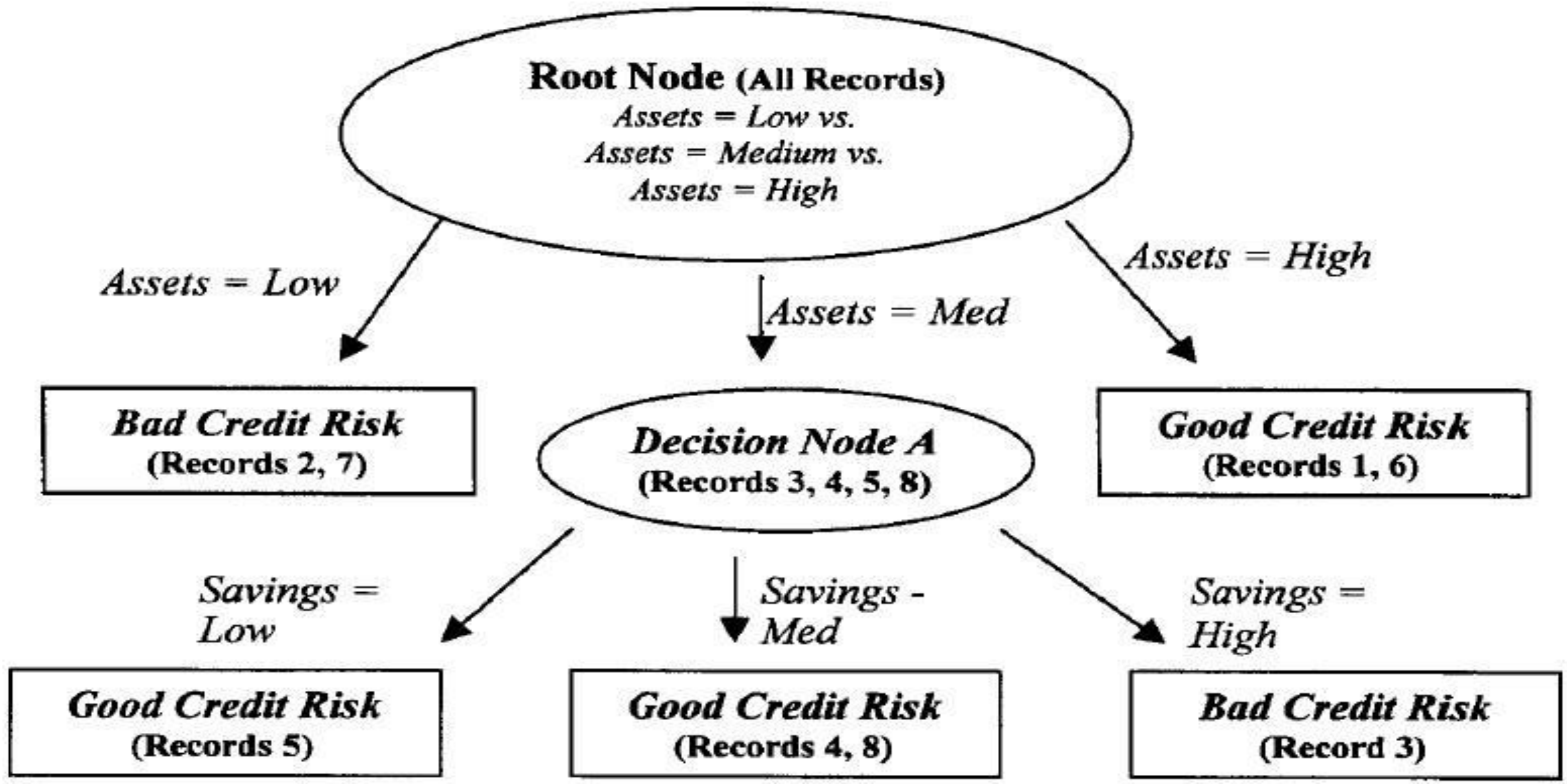
$$H(A) = -\sum_j p_j \log_2(p_j) = -\frac{3}{4}\log_2\left(\frac{3}{4}\right) - \frac{1}{4}\log_2\left(\frac{1}{4}\right) = 0.8113$$

C4.5 ALGORITHM

Candidate Split	Child Nodes		
1	$Savings = low$	$Savings = medium$	$Savings = high$
3	$Income \leq \$25{,}000$		$Income > \$25{,}000$
4	$Income \leq \$50{,}000$		$Income > \$50{,}000$
5	$Income \leq \$75{,}000$		$Income > \$75{,}000$

Candidate Splits at Decision Node A

C4.5 ALGORITHM



C4.5 Decision tree: fully grown form

DECISION RULES

Antecedent	Consequent	Support	Confidence
If *assets* = *low*	then *bad credit risk.*	$\frac{2}{8}$	1.00
If *assets* = *high*	then *good credit risk.*	$\frac{2}{8}$	1.00
If *assets* = *medium* and *savings* = *low*	then *good credit risk.*	$\frac{1}{8}$	1.00
If *assets* = *medium* and *savings* = *medium*	then *good credit risk.*	$\frac{2}{8}$	1.00
If *assets* = *medium* and *savings* = *high*	then *bad credit risk.*	$\frac{1}{8}$	1.00

Decision Rules Generated from Decision Last Tree

Tutorial

- SQL Server Data mining tutorial
  - Basic Data Mining Tutorial http://technet.microsoft.com/en-us/library/ms167167.aspx (Lessons 1-6)