ULB

UNIVERSITÉ LIBRE DE BRUXELLES

Implementation of Decision Tree Classifiers
ID3 versus C4.5

Depuydt Antoine
Dany Efila
Mudura Mircea

May, 2017

Introduction

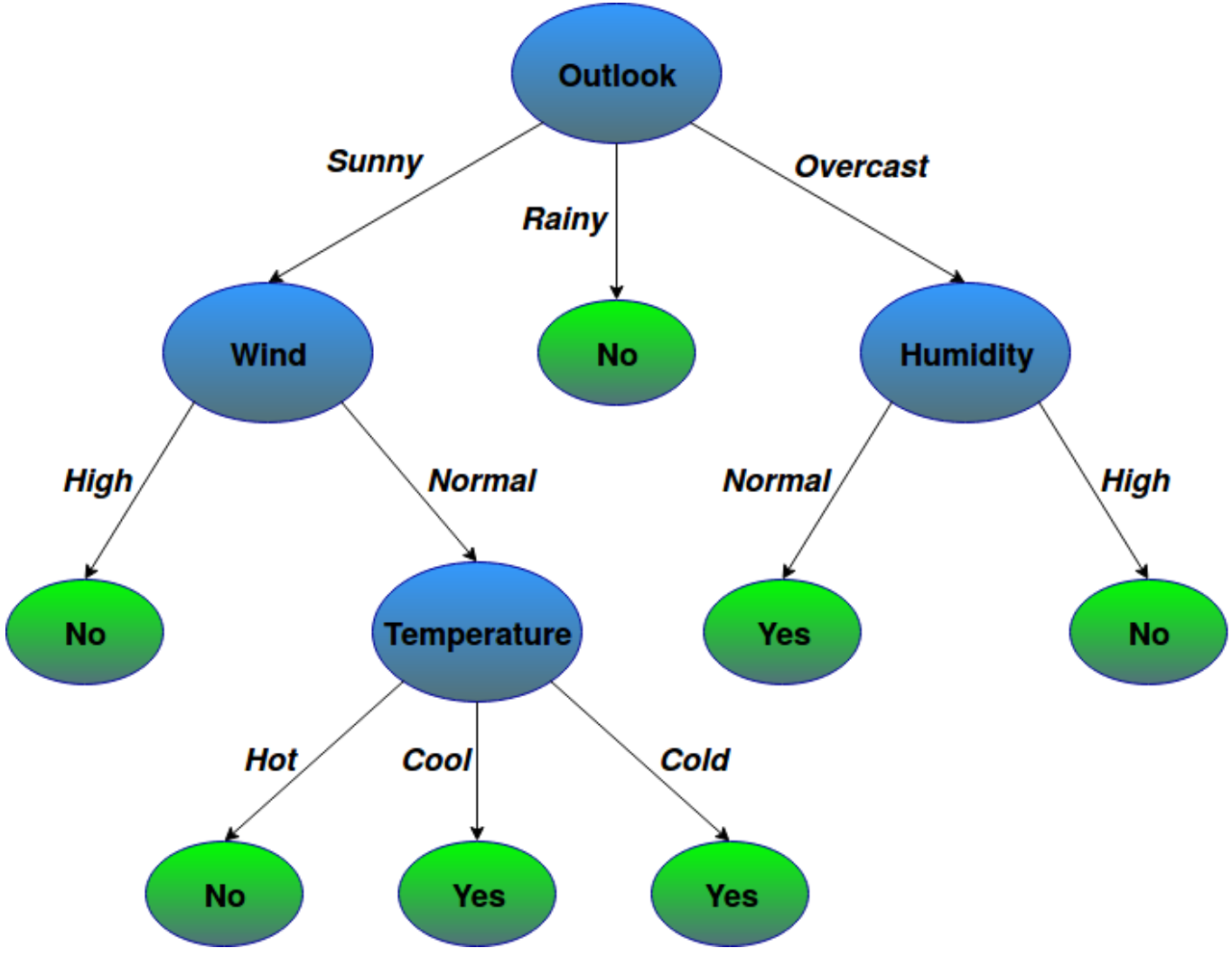
- Data mining: compress, understand and predict
  - Clustering
  - Classification
  - Regression
  - ...
- Techniques to find links
  - Linear Regression
  - Decision Trees
  - Neural Networks
  - ...

Classification

- Classical example: play tennis today?
  - **Features**:
    - Outlook: sunny, overcast, rainy
    - Temperature: hot, cool, cold
    - Wind: high, weak
    - Humidity: high, normal
  - **Class labels**:
    - Yes
    - No

Decision Tree

- Visual model, easily understandable
- Model: tree with decision and leaf nodes



Premise

- Given a training data-set
- Recursively split on a node:
- If node is pure return leaf (class value)
- Else compute entropy & info gain:
  - Shannon's entropy: $E(S) = \sum_i -p_i log_2(p_i)$
  - Subtree gain: $Gain(T, X) = E(T) - E(T, X)$

ID3 versus C4.5

- Goal: implement ID3 and C4.5 algorithms
- Objectives: compare ID3 and C4.5 output
  - Compare ID3 and C4.5
  - Create an application that classifies any data using both algorithms

The math

- Shannon's entropy: $H(S) = -\sum_i (p_i * log_2(p_i))$
  - Higher entropy more information gathered
- Example coin toss:
  - Entropy $= -P(heads)log_2P(heads) - P(tails)log_2P(tails)$
- Information Gain: $IG(A) = H(S) - \sum_t p(t)H(t)$
  - H(S) = entropy of set S
  - t Subset of S obtained by splitting S with A (T)
  - $p(t)$ proportion of elements in t to the no elements in S
  - $H(t)$ entropy of subset t

Getting to know the data

Attributes, Classifier, Class features

Outlook	Temperature	Humidity	Windy	Play
sunny	hot	high	false	no
sunny	hot	high	true	no
overcast	hot	high	false	yes
rainy	mild	high	false	yes
rainy	cool	normal	false	yes
rainy	cool	normal	true	no
overcast	cool	normal	true	yes
sunny	mild	high	false	no
sunny	cool	normal	false	yes
rainy	mild	normal	false	yes
sunny	mild	normal	true	yes
overcast	mild	high	true	yes
overcast	hot	normal	false	yes
rainy	mild	high	true	no

Applied math

- $Entropy(S) = -\frac{9}{14} log_2(\frac{9}{14}) - \frac{5}{14} log_2(\frac{5}{14})$
- $Gain(S, Windy) =$
  $Entropy(S) - \frac{8}{14} Entropy(S_{false}) - \frac{6}{14} Entropy(S_{true})$
- Windy:
  - False 8 $\rightarrow (6_+, 2_-)$
  - True 6 $\rightarrow (3_+, 3_-)$
- $Entropy(S_{false}) = -\frac{6}{8} log_2(\frac{6}{8}) - \frac{2}{8} log_2(\frac{2}{8})$=0.811
- $Entropy(S_{true}) = -\frac{3}{6} log_2(\frac{3}{6}) - \frac{3}{6} log_2(\frac{3}{6})$=1
- $\rightarrow Gain(S, Windy) = 0.940 - \frac{8}{14} * 0.811 - \frac{6}{14} * 1 = 0.048$
- $Gain(S, Humidity) = 0.151$
- $Gain(S, Temperature) = 0.029$
- $Gain(S, Outlook) = 0.246 \leftarrow$

Stopping

- Split the set according to the maximum gain
- Repeat until homogeneous or pure nodes (ie only leaves or entropy $=0$)

Pseudo-code shamelesly stolen from wikipedia

- If there is a mix of positive and negative examples ID3 works as:
- ID3 (Examples, Target Attribute, Attributes)
  - A The Attribute that best classifies examples.
  - Decision Tree attribute for Root = A.
  - For each possible value, vi, of A,
  - Add a new tree branch below Root, corresponding to the test A = vi.
  - Let Examples(vi) be the subset of examples that have the value vi for A
  - If Examples(vi) is empty
  - Then below this new branch add a leaf node with label = most common target value in the examples
  - Else below this new branch add the subtree ID3 (Examples(vi), Target Attribute, Attributes A)
  - End
  - Return Root

Improvements?

- Entropy & information gain not sufficient metrics
- Missing data has to be handled
- Numerical values could provide order or dimension to a problem set
- Tree can be simplified

Missing data I

2,*,*,*,*,*,2
1,2,*,*,*,*,1
1,1,2,*,*,*,1
1,1,1,*,*,*,1
1,1,3,2,2,*,1
1,*,*,*,*,4,1
2,1,4,*,*,1,1
2,1,4,*,*,2,1
2,1,4,*,*,3,1
2,1,3,1,1,1,1
2,1,3,1,1,2,1
2,1,3,1,2,1,1
2,1,3,1,2,2,1
1,1,3,1,1,3,1
2,1,3,1,2,3,1

Missing data II

- Datatypes can co-exist (eg. strings, integer/float)

- Solutions
  - Replace missing values in column with most frequent
  - For numerical values replace with mean/mode/median
- Column 2:
  - No instances = 15
  - Card(2) = 1
  - Card(1) = 12
  - $\rightarrow$ safest choice replace missing values with 1
- Column 3:
  - No instances = 15 (of course)
  - Card(2) = 1
  - Card(1) = 1
  - Card(3) = 7
  - Card(4) = 3
  - Missing = 3
  - $\rightarrow$ replace missing values with 3

Numerical & continuous variables

- General approach separate categorical and continuous
- Our implementation:
  - Treat all numerical variables as continuous
  - C45 implementation based on a binary tree (computational gain)
  - $\rightarrow$ Everything equal or smaller than node value to the left
  - $\rightarrow$ Everything else to the right

C4.5 I

- Simplifying a tree
  - Given a target gain level (generic or user-defined)
  - Prune (condense) the subtree
    - Might induce overclassification or errors
    - Decreases the depth of the tree
- 2 strategies:
  - **Pre-prune:**
    - Using statistical signifiance
    - $\rightarrow$ stop growing/building when no statistical significant association between any attribute and class at a node
    - chi-squared test (too much statistics for us)
    - Pre-pruning may stop growing prematurely (eg. XOR stops at root node)

C4.5 II

- **Post-prune:**
  - Subtree replacement
  - → Replace subtree with leaf
  - → Stop when additional pruning is harmful
  - → Accuracy default/user given
  - → Usinga validation test-set (derived from original)
  - Subtree raising
  - → Delete node & redistribute instances
  - → Slower than replacement strategy

Pruning

- Example avec notre code

C4.5 - last slide, we promise

- Implementation differences to ID3
  - First test for missing values and replace
  - When splitting apply rule for numerical values
  - After growing the tree: prune

ULB

UNIVERSITÉ LIBRE DE BRUXELLES

K-fold cross validation

ULB

UNIVERSITÉ LIBRE DE BRUXELLES

Demonstration