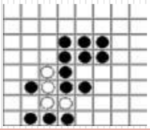
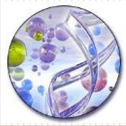



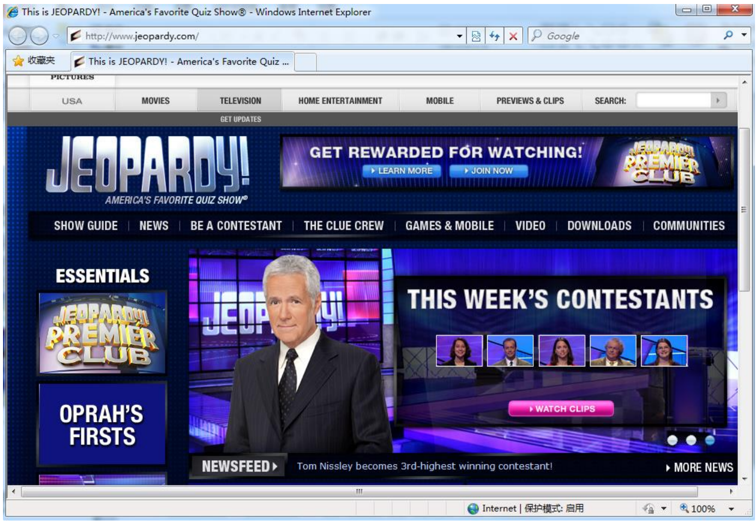

Welcome to

Introduction to Machine Learning!



Coffee Time

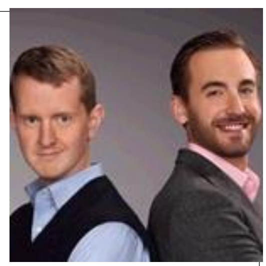
IBM Watson DeepQA



Jeopardy: An American TV show

Requires the players to suss out the subtleties of language from jokes and puns to irony and anagrams

IBM Watson @ Jeopardy



- February 14, 15, and 16, 2011
  - *Jeopardy*'s two biggest champions
  - Brad Rutter (right):
    - Won a whopping $3.25 million playing *Jeopardy*, the most cash ever awarded on the show.
    - He is a Johns Hopkins University dropout
  - Ken Jennings (left):
    - Holds the title for longest *Jeopardy* winning streak, with 74 consecutive wins in 2004.
    - He holds degrees in computer science and English, from Brigham Young University, and an international BA diploma from Seoul Foreign School.

IBM Watson won the Jeopardy

- http://domino.watson.ibm.com/library/cyberdig.nsf/papers/D12791EAA13BB952852575A1004A055C/$File/rc24789.pdf

Towards the Open Advancement of Question Answering Systems

Final:

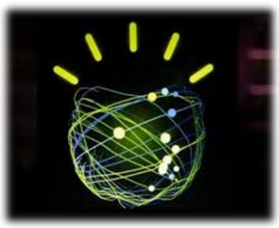
$77,147

to

$21,600 &

$24,000.

IBM Watson



- In development for 4 years
- Runs on 90 servers
- Does not connect to the Internet
- Search on a large scale knowledge base
- Trained with previous questions and games
  - With Jeopardy players: 77 (2009) + 55 (2010, winners)
  - E.g. Category: US Cities
    - *Q: "Its largest airport was named for a World War II hero; its second largest, for a World War II battle."*
    - *A: "What is Chicago / Toronto?"*

Technical requirements

- Answers to questions on any topic
  - Science, geography, popular culture …
- Accuracy: not only an answer, but a confident right answer
- Speed: within 3 second or less

- Advanced linguistic understanding
  - Parser complex sentences, recognize and understand jokes, metaphors, puns and riddles
- Real time analysis of questions
- Learn from mistakes
- Be prepared to handle the unexpected …

Techniques involved -- DeepQA

- A massively parallel probabilistic evidence-based architecture for answer questions
  - Non-database approach
  - Deep text analytics
    - NLP and statistical NLP
  - Machine learning
    - Formulating parallel hypotheses with confidence score
    - Voting, Question interpretation…
  - Search
  - Risk assessment
  - Hadoop and UIMA

Topic 2: Decision Trees

ZHANG Min
z-m@tsinghua.edu.cn

Part I. Basic Decision tree learning

An example: Enjoy Sport

- Known:

Sky	Temp	Humid	Wind	Water	Forecst	Enjoy
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes

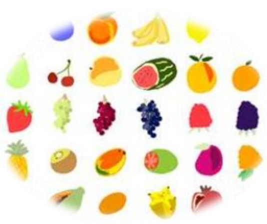
In a coming new day,

will the one enjoy the sport?

Problems

- Classification problem involving nominal data.
- Discrete
- No natural notion of similarity
- No order, in general

- Another Example: Fruit
  - Color: red, green, yellow, …
  - Size: small, medium, big
  - Shape: round, thin
  - Taste: sweet, sour



Representation

- Lists of attributes instead of vectors of real numbers.

e.g.

- EnjoySport:
  - 6-tuple on *Sky, AirTemp, Humidity, Wind, Water, Forecast*
  - {Sunny, Warm, Normal, Strong, Warm, Same}
- Fruit:
  - 4-tuple on *color*, *size*, *shape*, *taste*

    {red, round, sweet, small}

Basic Concepts

- Given:
  - Instance Space X e.g. possible days, each described by the attributes *Sky, AirTemp, Humidity, Wind, Water, Forecast*
  - Hypothesis Class H e.g.

    if (Temp = cold AND humidity = high) *then* play tennis = no.
  - Training Examples D Positive and negative examples of the Target Function C $<x_1, c(x_1)>, \ldots\ldots, <x_m, c(x_m)>$
- Determine: A hypothesis $h \in H$ *such that*

$$h(x) = c(x) \text{ for all } x \in X$$

Basic Concepts

- Typically X is exponentially or infinitely larger, so in general we can never be sure that $h(x)=c(x)$ for all $x \in X$
- Instead, settle for a good approximation, e.g. $h(x)=c(x)$ for all $x \in D$

Suppose: n binary attributes/features（e.g. true/false, warm/cold）

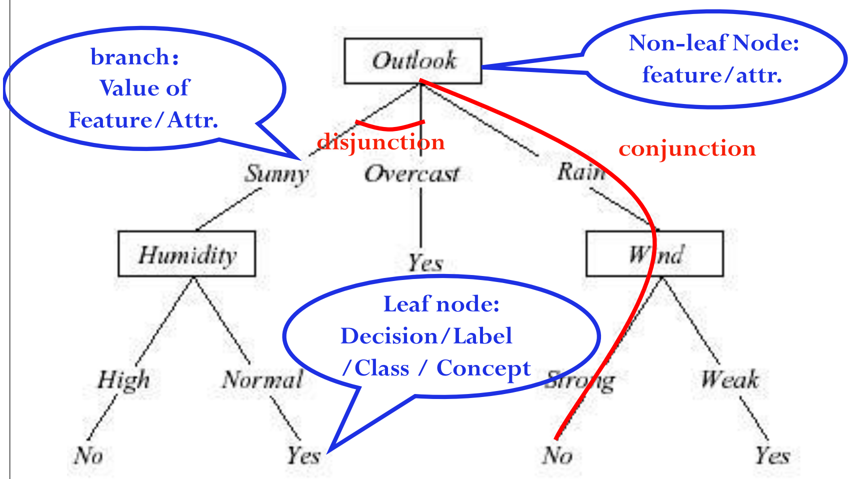
- **Instance Space X： 2^n elements**
- **Concept (Hypothesis) Space H： at most 2^{2^n} elements (why?)**

Training examples

Sky	Temp	Humid	Wind	Water	Forecst	Enjoy
Sunny	Warm	Normal	Strong	Warm	Same	Yes
Sunny	Warm	High	Strong	Warm	Same	Yes
Rainy	Cold	High	Strong	Warm	Change	No
Sunny	Warm	High	Strong	Cool	Change	Yes
			…			…

- Banana: yellow, thin, medium, sweet
- Watermelon: green, round, big, sweet
- Banana: yellow, thin, medium, sweet
- Grape: green, round, small, sweet
- Grape: red, round, small, sour
- …

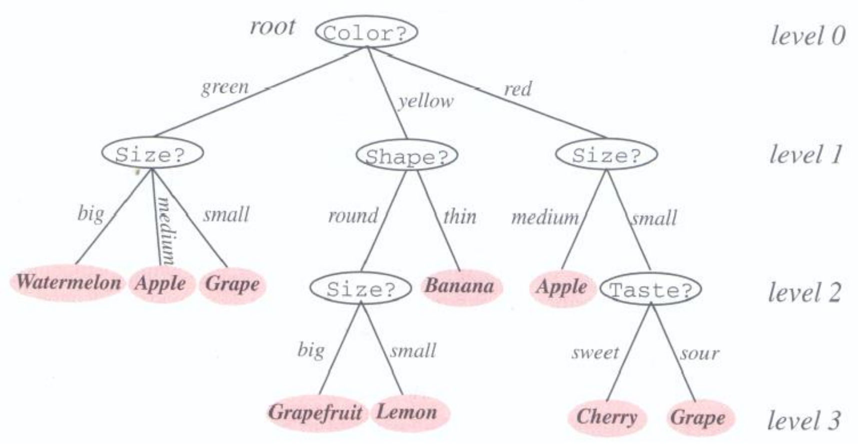
Decision tree – Concepts



17

Introduction to Machine Learning: Decision Tree Learning

Example 2: Fruits



18

Introduction to Machine Learning: Decision Tree Learning

Decision tree – Milestones

- In 1966, first proposed by Hunt
- In 1970's~1980's
  - CART by Friedman, Breiman
  - ID3 by Quinlan
- Since 1990's
  - Comparative study (Mingers, Dietterich, Quinlan, etc)
  - Most popular DTree algorithm: C4.5 by Quinlan in 1993

Classical Decision Tree Algorithms

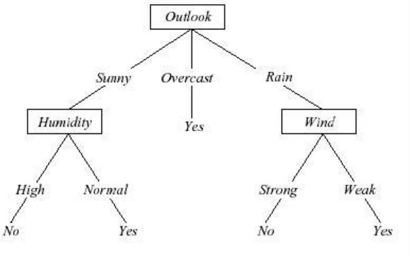
CART (classification and regression trees)

A general framework:

- Create or grow a decision tree using training data
- Decision tree will progressively split the set of training examples into smaller and smaller subsets
- Stop splitting if each subset is pure
- Or accept an imperfect decision

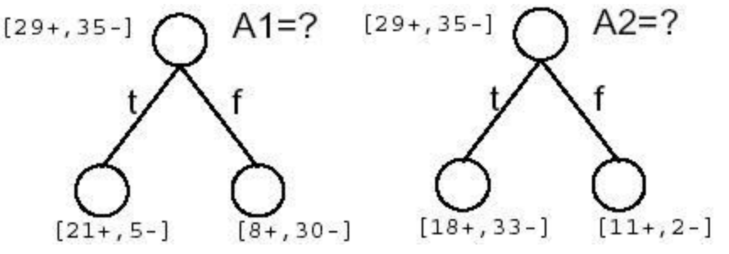
Many DTree algorithms follow this framework, including ID3, C4.5, etc.

Classical DTree Alg. – ID3



- Top-down, greedy search
- Recursive algorithm
- Main Cycle:
  - A : the best? decision attribute for the next step
  - Assign A as decision attribute for node
  - For each value of A (v_i), create new descendant of node
  - Sort training examples to leaf nodes
  - If training examples perfectly classified?, Then RETURN, Else drill down to new leaf nodes

ID3 Q1: Which attribute is the best one?

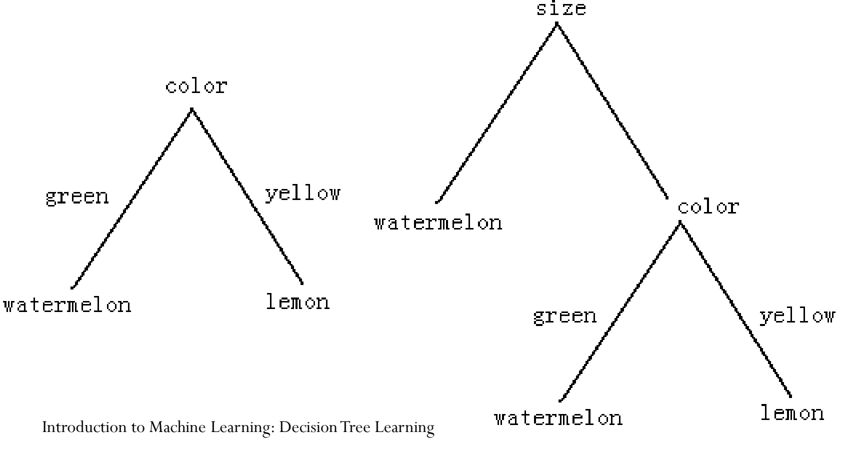


Outlook, Humidity, Wind, ….. ?

Query selection and node impurity

- Fundamental principle: **simplicity**

  We prefer decisions that lead to a simple, compact tree with few nodes



Query selection and node impurity

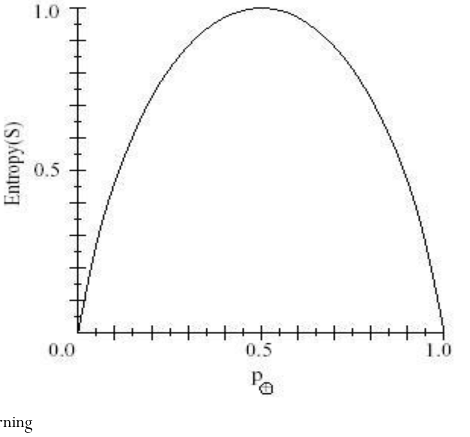
- Fundamental principle: **simplicity**
  - We prefer decisions that lead to a simple, compact tree with few nodes
- We seek a property query T at each node N that makes the data reaching the immediate descendent nodes as "pure" as possible
- Purity – Impurity

How to measure impurity?

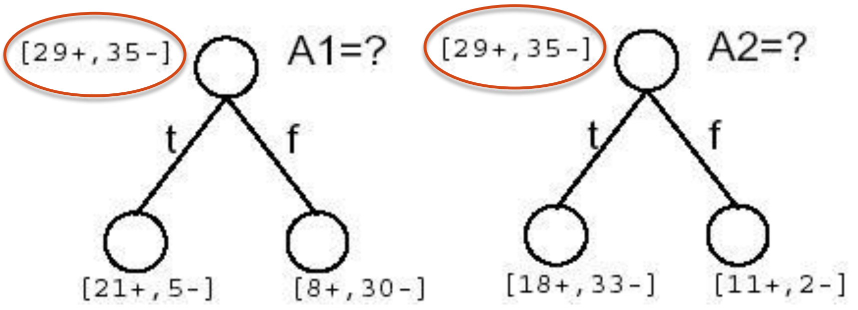
Entropy impurity (is frequently used)

$$Entropy(N) = -\sum_j P(w_j)\log_2 P(w_j)$$

- Define: 0log0=0
- In information theory, entropy measures the **purity/impurity** of information, or the uncertainty of information
- Uniform distribution – Maximum value of entropy



Entropy



$$Entropy(S) = -\frac{29}{64} \times \log_2 \frac{29}{64} - \frac{35}{64} \times \log_2 \frac{35}{64} = 0.993$$

Besides Entropy Impurity

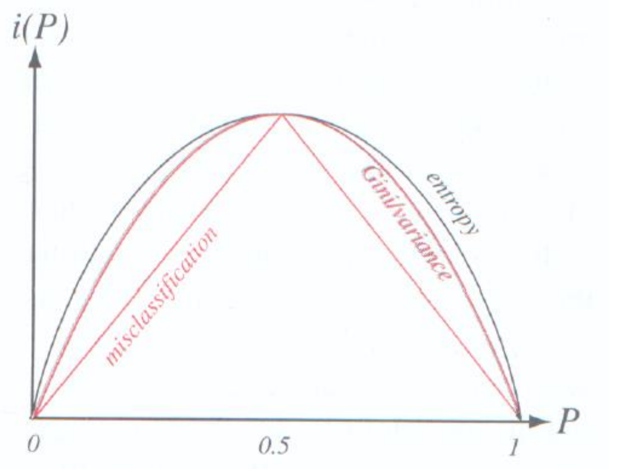
- Gini impurity (Duda prefers Gini impurity)

$$i(N) = \sum_{i \neq j} P(w_i)P(w_j) = 1 - \sum_j P^2(w_j)$$

- Misclassification impurity

$$i(N) = 1 - \max_j P(w_j)$$

Impurity



Measuring the change of impurity ΔI(N) — Information Gain (IG), for example

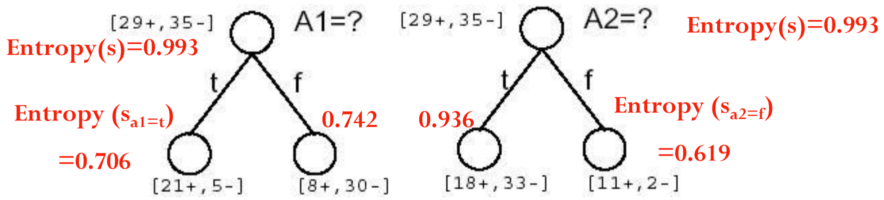
- Expected reduction in entropy due to sorting on A

$$Gain(S,A) \equiv \underbrace{Entropy(S)}_{\text{Entropy of Original S}} - \underbrace{\sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)}_{\text{Expected entropy after sorting on A}}$$

Information Gain, IG

- Expected reduction in entropy due to sorting on A

$$Gain(S,A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

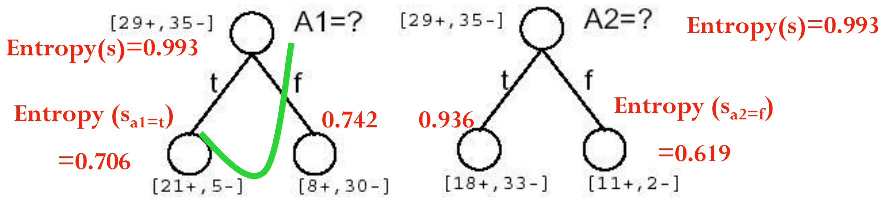


$$Entropy(S) = -\frac{29}{64} \times \log_2 \frac{29}{64} - \frac{35}{64} \times \log_2 \frac{35}{64} = 0.993$$

Information Gain, IG

- Expected reduction in entropy due to sorting on A

$$Gain(S,A) \equiv Entropy(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$



$$Gain(S,A_1) = 0.993 - (\frac{26}{64} \times 0.706 + \frac{38}{64} \times 0.742) = 0.266, \quad Gain(S,A_2) = 0.121$$

ID3 Q2: when to RETURN (stop splitting) ?

- "If training examples perfectly classified"
- Condition 1: if all the data in the current subset has the same output class, then stop
- Condition 2: if all the data in the current subset has the same input value, then stop

Possible condition 3: if all the attributes' IG scores are 0, then stop

A good idea?

ID3 Q2 : when to RETURN (stop splitting) ?

- y=a XOR b

a	b	y
0	0	0
0	1	1
1	0	1
1	1	0

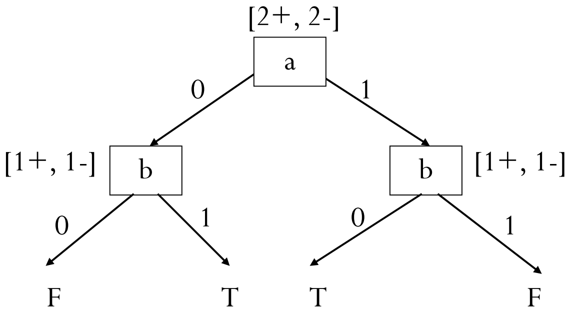
Information Gain:

Attr	value	probability	IG
a	0	50%	0
	1	50%	
b	0	50%	0
	1	50%	

- According to the proposed condition 3, No attribute could be chosen even at the first step.

ID3 Q2 : when to RETURN ?

- If we ignore the proposed condition 3



There're ONLY 2 conditions for stopping splitting in ID3:

- The same output class or The same input value

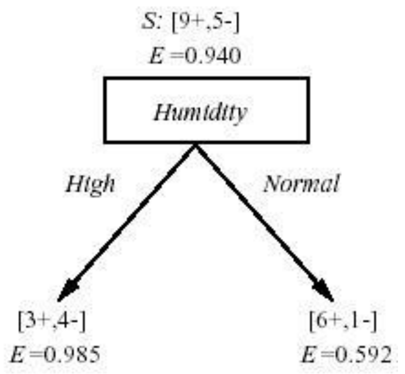
Discussion: If they have same input but diff. output, what does it mean?

ID3 example: training samples

High: 3+,4 -; Normal: 6+,1- Total: 9+, 5- ;

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

ID3 example: feature selection



Gain(S,Humidity)

=0.940－(7/14)*0.985-
(7/14)*0.592

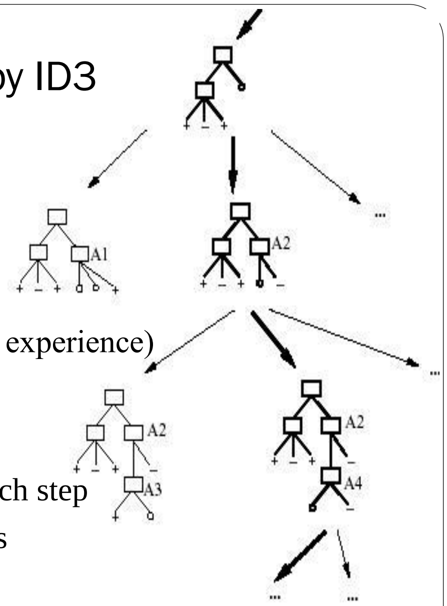
=0.151

Gain(S,Outlook)=0.246

Gain(S,Wind)=0.048

Gain(S,Temperature)=0.029

Hypothesis space search by ID3



- Hypothesis space is complete
  - Target function surely in there
- Output a single hypothesis
  - Can't play over 20 questions (by experience)
- No back tracking
  - Local minima…
- Use all the data in the subset for each step
  - Statistically-based search choices
  - Robust to noisy data

Inductive bias in ID3

- Note H is the power set of instances X
  - No restriction on the hypothesis space
- Preference for trees with high IG attributes near the root
  - Attempt to find the shortest tree
  - Bias is a ***preference*** for some hypotheses (search bias), rather than a ***restriction*** of hypothesis space H (language bias).
  - Occam's razor: prefer the shortest hypothesis that fits the data

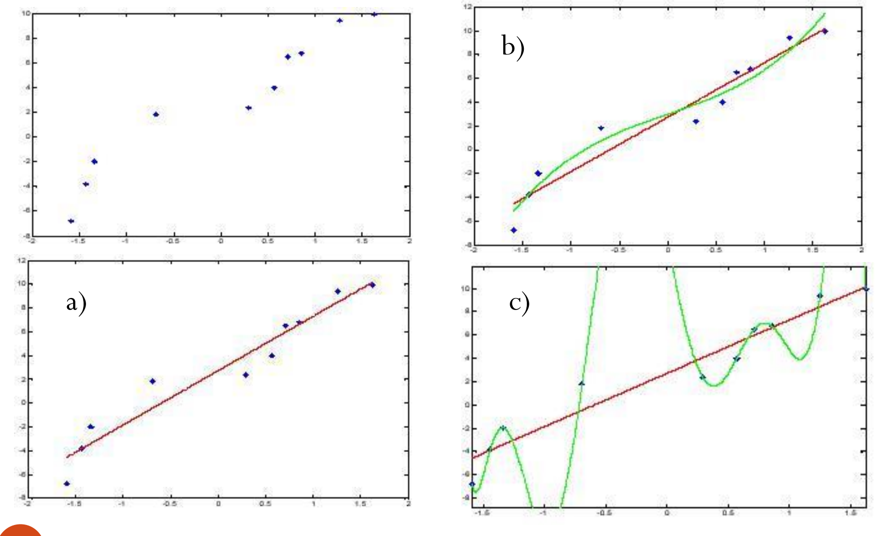
Occam's razor

- Just gives an idea here, no detail discussion
- For more information:
  - Domingos, The role of Occam's Razor in knowledge discovery. Journal of Data Mining and Knowledge Discovery, 3(4), 1999.

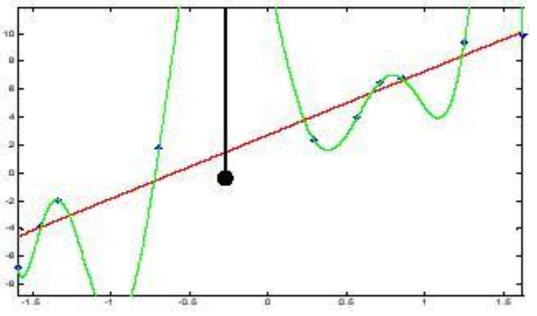
Decision Tree

- Introduction -- basic concepts
- ID3 algorithm as an example
  - Algorithm description
  - Feature selection
  - Stop conditions
  - Inductive bias for ID3
- Over-fitting and Pruning

What's over-fitting ?



What's over-fitting ?



- $h \in$ H overfits training data if there's an alternative h' $\in$ H such that:

  $err_{train}(h) < err_{train}(h')$

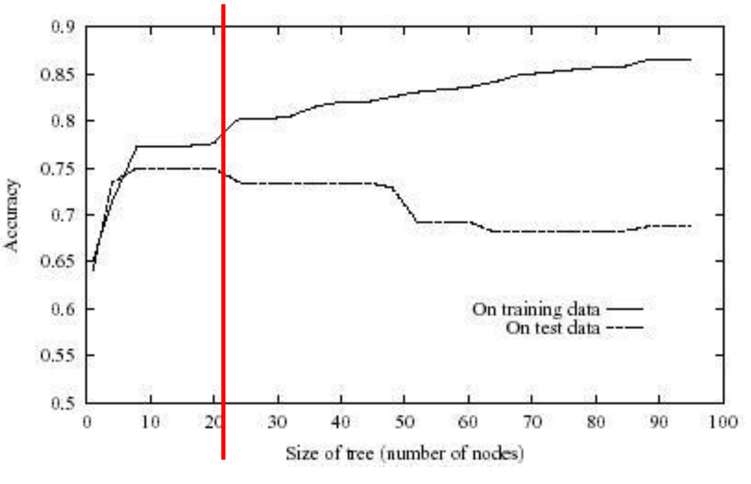
  AND

  $err_{test}(h) > err_{test}(h')$

An example of over-fitting in DTree

- Each leaf corresponds to a single training point and the full tree is merely a convenient implementation of a lookup table

Over-fitting in Decision Tree Learning



Avoid over-fitting

- Two ways of avoid over-fitting for DTree
  I. Stop growing when data split not statistically significant (pre-pruning)
  II. Grow full tree, then post-pruning

For Option I:

Hard to estimate the size of the tree

Pre-Pruning: When to stop splitting (I) Number of instances

- Frequently, a node is not split further if
  - The number of training instances reaching a node is smaller than a certain percentage of the training set
    - (e.g. 5%)
  - Regardless the impurity or error.
  - Any decision based on too few instances causes variance and thus generalization error.

Pre-Pruning: When to stop splitting (2) Threshold of information gain value

- Set a small threshold value, splitting is stopped if $\Delta i(s) \leq \beta$
- Benefits: Use all the training data. Leaf nodes can lie in different levels of the tree.
- Drawback: Difficult to set a good threshold

Avoid over-fitting

- Two ways of avoid over-fitting for D-Tree

  I. Stop growing when data split not statistically significant (pre-pruning)

  II. Grow full tree, then post-pruning

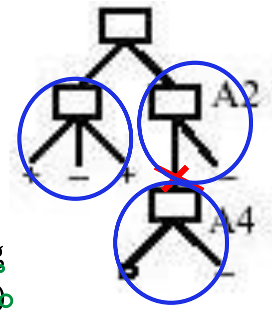
For option II:

- How to select "best" tree？
  - Measure performance over training data (statistical pruning)
    - Confidence level (will be introduced later)
  - Measure performance over separate validation data set
- MDL (Minimize Description Length 最小描述长度):

  minimize (*size*(tree) + *size*(misclassifications(tree)))

Post-pruning (1). Reduced-Error pruning

- Split data into training set and validation set
  - Validation set:
    - Known label
    - Test performance
    - No model updates during this test!
- Do until further pruning is harmful:
  - Evaluate impact on validation set of pruning each possible node (plus the subtree it roots)
  - Greedily remove the one that most improves validation set accuracy

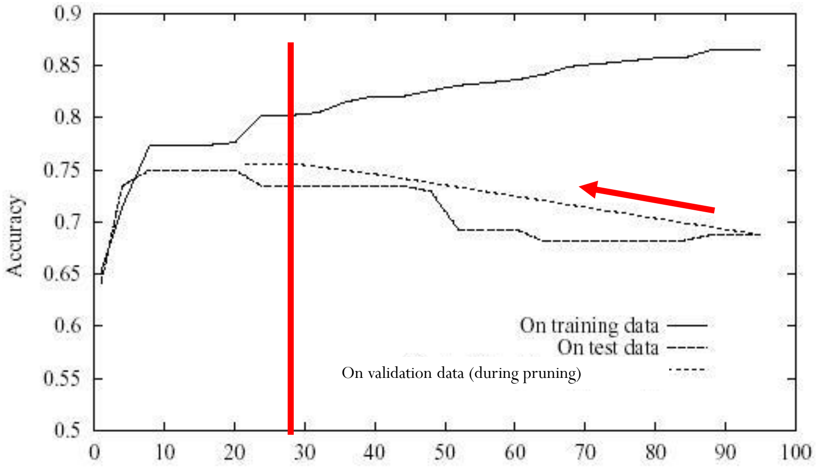


How to assign the label of the new leaf node?

Supplement: strategies of the new leaf node label after pruning

- Assign the most common class.
- Give the node multiple-class labels
  - Each class has a support degree (based on the number of the training data with each label)
  - On test: select one class with probability, or select multiple classes
- If it is the regression tree (numeric labels), can be averaged, or weighted average.
- ……

Effect of Reduced-Error pruning



Post-pruning (2). Rule Post-pruning

1, Convert tree to equivalent set of rules

- e.g. if (outlook=sunny)$\wedge$(humidity=high) then playTennis = no

2, Prune each rule by removing any preconditions that result in improving its estimated accuracy

- i.e. (outlook=sunny), (humidity=high)

3, Sort rules into desired sequence (by their estimated accuracy) .

4, Use the final rules in the same sequence when classifying instances.

(after the rules are pruned, it may not be possible to write them back as a tree anymore.)

One of the most frequently used methods, e.g. in C4.5.

Why convert the decision tree to rule before pruning?

- Independent to contexts.
  - Otherwise, if the tree were pruned, two choices:
    - Remove the node completely, or
    - Retain it there.
- No difference between root node and leaf nodes.
- Improve readability

Brief overview of Decision Tree Learning (Part 1)

- Introduction -- basic concepts
- ID3 algorithm as an example
  - Algorithm description
  - Feature selection
  - Stop conditions
  - Inductive bias for ID3
- Over-fitting and Pruning
  - Pre-pruning
  - Post-pruning: Reduced-Error pruning, Rule post-pruning
  - In practice, pre-pruning is faster, post-pruning generally leads to more accurate trees

Brief overview of Decision Tree Learning (Part 1)

- The basic idea come from human's decision procedure
- Simple, easy to understand: If…Then…
- Robust to noise data
- Widely used in research and application
  - Medical Diagnosis (Clinical symptoms → disease)
  - Credit analysis (personal information → valuable custom?)
  - Schedule
  - ……
- A decision tree is generally tested as the benchmark before more complicated algorithms are employed.

Part 2: Advanced Topics in Decision Tree

Problems & improvements

1. Continuous attribute value

$$x_l < x_s < x_u$$

Temperature	40	48	60	72	80	90
decision	No	No	Yes	Yes	Yes	No

- Create a set of discrete attribute value
- Options:
  - I. Get the medium of the adjacent values with different decisions

$$x_s = (x_l + x_u)/2$$

(Fayyad proved that thresholds lead to max IG satisfies the condition in 1991)

  - II. Take into account the probability $x_s = (1-P)x_l + Px_u$

2. Attributes with many values

Problem:

- Bias: If attribute has many values, IG will select it
  - e.g. Date as an attribute
- One possible solution: use *GainRatio* instead

$$GainRatio(S, A) \equiv \frac{Gain(S, A)}{SplitInformation(S, A)}$$

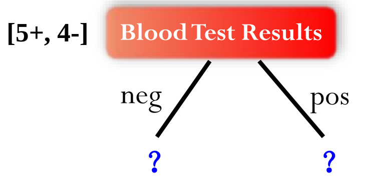
$$SplitInformation(S, A) \equiv -\sum_{i=1}^{c} \frac{|S_i|}{|S|} \log_2 \frac{|S_i|}{|S|}$$

Punish factor, entropy of S on A

3. Unknown attribute values

BTR	Temp	…	label
neg	normal	…	-
neg	normal	…	-
neg	normal	…	-
neg	normal	…	-
neg	high	…	+
pos	normal	…	+
pos	high	…	+
pos	high	…	+
?	normal	…	+

With missing data



Most common training: neg

[2+, 4-] **[3+, 0-]**

Most common according to the label: pos

[1+, 4-] **[4+, 0-]**

Assign probability: neg 5/8, pos 3/8

[(1+5/8)+, 4-] **[(3+3/8)+, 0-]**

4. Attributes with costs

- Tan & Schlimmer (1990)

$$\frac{Gain^2(S,A)}{Cost(A)}$$

- Nunez (1988)

$$\frac{2^{Gain(S,A)}-1}{(Cost(A)+1)^w}$$

  - w:[0,1] importance of cost

What's more ...

- Perhaps the simplest and the most frequently used algorithm
  - Easy to understand
  - Easy to implement
  - Easy to use
  - Small computation costs
- Decision Forest:
  - Many decision trees by C4.5
- For More information about C4.5 (C5.0):
  - http://www.rulequest.com/Personal/
    Ross Quinlan's homepage

Experiment 1

Decision Tree Algorithm and Analysis

Deadline: March 23 (Sunday), 2014