

SMLO12: Data Mining

Statistical Machine Learning Overview

Douglas Aberdeen

Canberra Node, RSISE Building
Australian National University

4th November 2004

Outline

- 1 Introduction
- 2 Association Rule Mining
- 3 Decision Trees
- 4 Summary

News

- Last Lecture!
- All assignments online
- Assignments due Fri Nov 12. Email me if you have a good reason for an extension.
- Tute at RSISE, tomorrow 3pm

Large Data Sets

- All the algorithms we've seen to date are $O(O(n^2))$ [sic]
- Normally getting enough data is the hard part!
- What if we have trillions of records?
 - Every sale at McDonalds
 - Every set of products (items) sold by Coles-Myer group
 - Every tax return ever submitted
- Each record draws maybe 1 to 200 items from a set of 100,000
- Holy overflowing RAM batman!
- Can we do useful things in close to linear time?

Setting

Suppose we have the following data...

Trasaction	Items
1	A B D
2	A B C D E
3	A C D
4	B D
5	D
6	B C D

Association Rule Mining

- Extremely simple, therefore fast
- $O(n)$ in the amount of data
- One **full** pass through the data
- Reading from slow media is the main expense
- Easy for executives to understand and trust

A-Priori Algorithm

- Based on idea of frequent **itemsets**
- Itemsets S are sets of items
- Frequent itemset if it appears in $> s\%$ of transactions, i.e.,

$$\text{support}(S) = \frac{\text{occurrences of } S}{\# \text{ transactions}}$$

A-Priori Trick

- If a set of items is frequent, all subsets of that set are also frequent
- Candidates for larger frequent itemsets are constructed from frequent subset

A-Priori Algorithm: s =support threshold

- 1 $k = 1$
- 2 Candidate set = all single items
- 3 For each transaction, count instance of a candidate set
- 4 For each candidate set...
 - 5 Divide candidate count by # transactions
 - 6 If result is $> s$, candidate is a frequent itemset
 - 7 End For
- 8 Generate new $k = k + 1$ candidates from k frequent itemsets
- 9 $k = k + 1$
- 10 If no candidates, end
- 11 Goto 3

What To Do With Frequent Itemsets

- A frequent item set by itself might be meaningful
- We can infer rules such as $A, B \leftarrow B$
- S is a frequent itemset X is a single item
- Compute all possible rules $\{S\} \rightarrow X$ from frequent itemsets
- If $\frac{\text{support}(S)}{\text{support}(X)} > c$, we are **confident** this rule means something

Example

- 6 Transactions. 50% minimum support
- First pass.. $k=1$

Candidate	Count	Support
A	3	50%
B	4	67%
C	3	50%
D	6	100%
E	1	17%

- $k = 2$ candidates: AB AC AD BC BD CD

Example

- 6 Transactions. 50% minimum support
- Second pass.. $k=2$

Candidate	Count	Support
AB	2	33%
AC	2	33%
AD	3	50%
BC	2	33%
BD	4	67%
CD	3	50%

- $k = 3$ candidates: ABD ACD BCD
- ABC cannot be a candidate (AB BC AC not frequent)

Example

- 6 Transactions. 50% minimum support
- Second pass.. $k=2$

Candidate	Count	Support
ABD	2	33%
ACD	1	17%
BCD	2	33%

- no $k = 4$ candidates. Stop
- Should have observed AB, BC, AC not frequent

Confident Rules

- Confidence of $B \rightarrow D = \frac{\text{support}(BD)}{\text{support}(D)} = 100\%$
- Confidence of $D \rightarrow B = \frac{\text{support}(BD)}{\text{support}(B)} = 67\%$
- Many more possible rules to test

Beer and Nappies

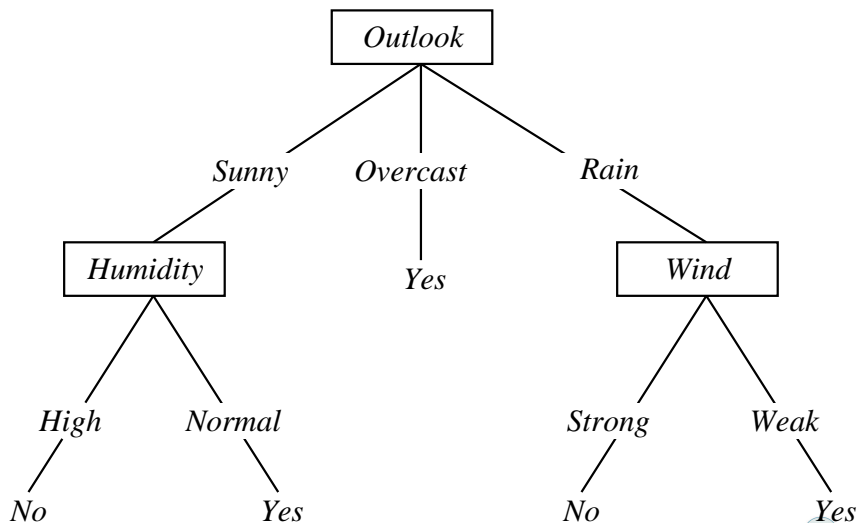
- Data Mining Urban Myth
- Men on Friday night buy beer and nappies
- Doing so on request from partners
- Put beer and nappies together, voila! \$\$\$
- Seems to be due to an IBM sales pitch?

A Real Example

- HIC commissioned DM project
- Associations on episode database for pathology services
- 6.8 million records X 120 attributes (3.5GB)
- 15 months preprocessing then 2 weeks data mining
- Goal: find associations between tests
- $cmin = 50\%$ and $smin = 1\%, 0.5\%, 0.25\%$ (1% of 6.8 million = 68,000)
- Unexpected/unnecessary combination of services
- Refuse cover saves \$550,000 per year

Decision Tree for *PlayTennis*

Content modified from Tom Mitchell



A Tree to Predict C-Section Risk

Learned from medical records of 1000 women

Negative examples are C-sections

```
[833+,167-] .83+ .17-
Fetal_Presentation = 1: [822+,116-] .88+ .12-
| Previous_Csection = 0: [767+,81-] .90+ .10-
| | Primiparous = 0: [399+,13-] .97+ .03-
| | Primiparous = 1: [368+,68-] .84+ .16-
| | | Fetal_Distress = 0: [334+,47-] .88+ .12-
| | | | Birth_Weight < 3349: [201+,10.6-] .95+ .05-
| | | | Birth_Weight >= 3349: [133+,36.4-] .78+ .22-
| | | Fetal_Distress = 1: [34+,21-] .62+ .38-
| Previous_Csection = 1: [55+,35-] .61+ .39-
Fetal_Presentation = 2: [3+,29-] .11+ .89-
Fetal_Presentation = 3: [8+,22-] .27+ .73-
```

Decision Trees

Decision tree representation:

- Each internal node tests an attribute
- Each branch corresponds to attribute value
- Each leaf node assigns a classification

When to Consider Decision Trees

- Instances describable by attribute–value pairs
- Target function is discrete valued
- Disjunctive hypothesis may be required
- Possibly noisy training data

Examples:

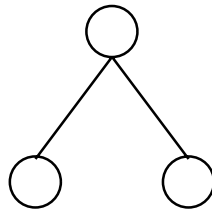
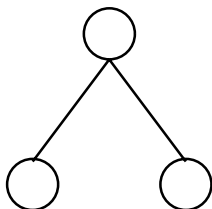
- Equipment or medical diagnosis
- Credit risk analysis
- Modeling calendar scheduling preferences

Top-Down Induction of Decision Trees

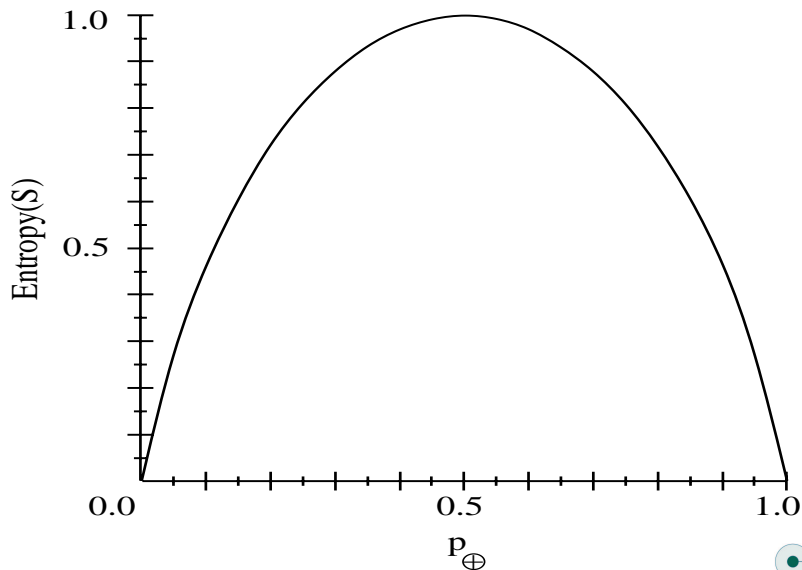
Main loop:

- 1 $A \leftarrow$ the “best” decision attribute for next *node*
- 2 Assign A as decision attribute for *node*
- 3 For each value of A , create new descendant of *node*
- 4 Sort training examples to leaf nodes
- 5 If training examples perfectly classified, Then STOP, Else iterate over new leaf nodes

Which attribute is best?



Entropy



Entropy Cont.

- S is a sample of training examples
- p_{\oplus} is the proportion of positive examples in S
- p_{\ominus} is the proportion of negative examples in S
- Entropy measures the impurity of S

$$\text{Entropy}(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

Entropy Cont

$Entropy(S)$ = expected number of bits needed to encode class (\oplus or \ominus) of randomly drawn member of S (under the optimal, shortest-length code)

Why?

Information theory: optimal length code assigns $-\log_2 p$ bits to message having probability p .

So, expected number of bits to encode \oplus or \ominus of random member of S :

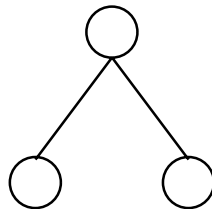
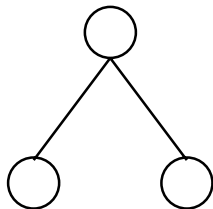
$$p_{\oplus}(-\log_2 p_{\oplus}) + p_{\ominus}(-\log_2 p_{\ominus})$$

$$Entropy(S) \equiv -p_{\oplus} \log_2 p_{\oplus} - p_{\ominus} \log_2 p_{\ominus}$$

Information Gain

$Gain(S, A) =$ expected reduction in entropy due to sorting on A

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

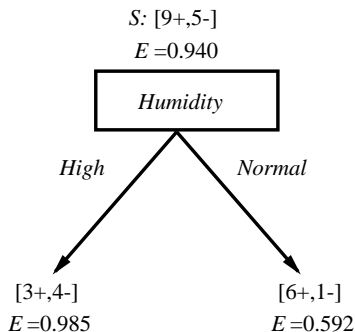


Training Examples

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

Selecting the Next Attribute

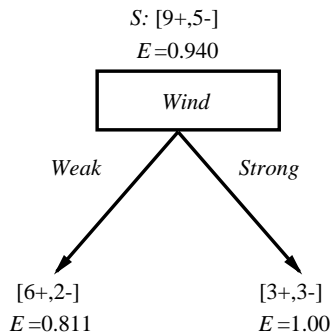
Which attribute is the best classifier?



$Gain(S, Humidity)$

$$= .940 - (7/14).985 - (7/14).592$$

$$= .151$$

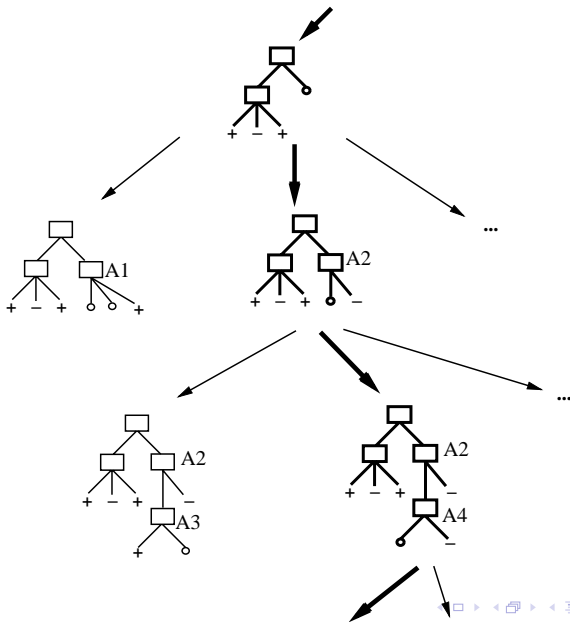


$Gain(S, Wind)$

$$= .940 - (8/14).811 - (6/14)1.0$$

$$= .048$$

Hypothesis Space Search by ID3



Hypothesis Space Search by ID3

- Hypothesis space is complete!
 - Target function surely in there...
- Outputs a single hypothesis (which one?)
 - Can't play 20 questions...
- No back tracking
 - Local minima...
- Statically-based search choices
 - Robust to noisy data...
- Inductive bias: approx “prefer shortest tree”

Inductive Bias in ID3

Note H is the power set of instances X

→ Unbiased?

Not really...

- Preference for short trees, and for those with high information gain attributes near the root
- Bias is a *preference* for some hypotheses, rather than a *restriction* of hypothesis space H
- Occam's razor: prefer the shortest hypothesis that fits the data

Occam's Razor

Why prefer short hypotheses?

Argument in favor:

- Fewer short hyps. than long hyps.
- a short hyp that fits data unlikely to be coincidence
- a long hyp that fits data might be coincidence

Argument opposed:

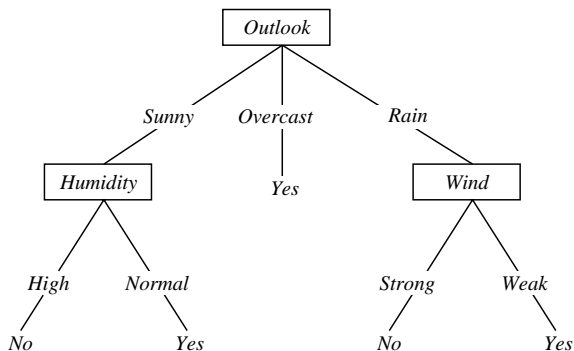
- There are many ways to define small sets of hyps
- e.g., all trees with a prime number of nodes that use attributes beginning with "Z"
- What's so special about small sets based on size of hypothesis??

Overfitting in Decision Trees

Consider adding noisy training example #15:

Sunny, Hot, Normal, Strong, PlayTennis = No

What effect on earlier tree?



Overfitting

Consider error of hypothesis h over

- training data: $error_{train}(h)$
- entire distribution \mathcal{D} of data: $error_{\mathcal{D}}(h)$

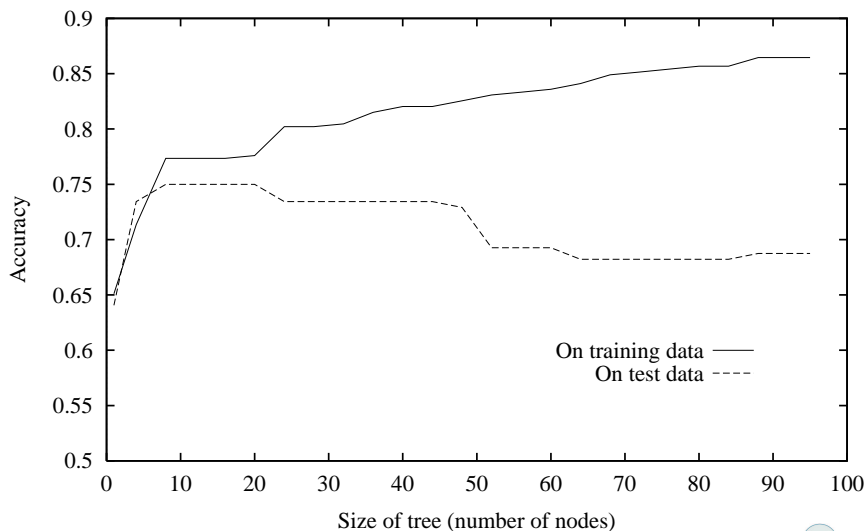
Hypothesis $h \in H$ **overfits** training data if there is an alternative hypothesis $h' \in H$ such that

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

and

$$error_{\mathcal{D}}(h) > error_{\mathcal{D}}(h')$$

Overfitting in Decision Tree Learning



Avoiding Overfitting

How can we avoid overfitting?

- stop growing when data split not statistically significant
- grow full tree, then post-prune

How to select “best” tree:

- Measure performance over training data
- Measure performance over separate validation data set
- MDL: minimize $size(tree) + size(misclassifications(tree))$

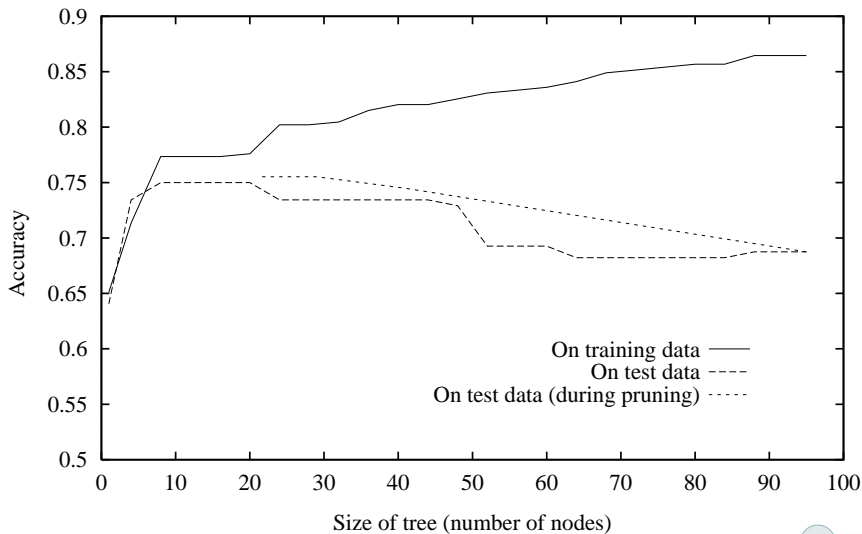
Reduced-Error Pruning

Split data into *training* and *validation* set

Do until further pruning is harmful:

- 1 Evaluate impact on *validation* set of pruning each possible node (plus those below it)
 - 2 Greedily remove the one that most improves *validation* set accuracy
- produces smallest version of most accurate subtree
 - What if data is limited?

Effect of Reduced-Error Pruning

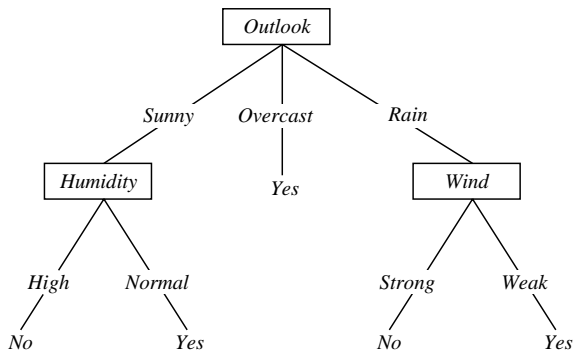


Rule Post-Pruning

- 1 Convert tree to equivalent set of rules
- 2 Prune each rule independently of others
- 3 Sort final rules into desired sequence for use

Perhaps most frequently used method (e.g., C4.5)

Converting A Tree to Rules



IF $(\text{Outlook} = \text{Sunny}) \wedge (\text{Humidity} = \text{High})$
 THEN $\text{PlayTennis} = \text{No}$

IF $(\text{Outlook} = \text{Sunny}) \wedge (\text{Humidity} = \text{Normal})$
 THEN $\text{PlayTennis} = \text{Yes}$

...

Continuous Valued Attributes

Create a discrete attribute to test continuous

- $Temperature = 82.5$
- $(Temperature > 72.3) = t, f$

<i>Temperature:</i>	40	48	60	72	80	90
<i>PlayTennis:</i>	No	No	Yes	Yes	Yes	No

Unknown Attribute Values

What if some examples missing values of A ?

Use training example anyway, sort through tree

- If node n tests A , assign most common value of A among other examples sorted to node n
- assign most common value of A among other examples with same target value
- assign probability p_i to each possible value v_i of A
 - assign fraction p_i of example to each descendant in tree

Classify new examples in same fashion

We Covered

- Review of some matrix stuff
- Evolutionary Algorithms, Gradient Methods, EM, Convex Optimisation
- Optimal Bayes, Naive Bayes, MAP vs ML,
- Aggregation Clustering, MST, kd-trees, k-means, KNN
- Perceptron, Multi-Layer NN, SOFM
- Hebbian learning, PCA, ICA
- Kernel perceptron, SVMs
- Graphical models (HMMs, Bayes Nets, Kalman Filter)
- Dynamic Programming, Temporal Difference Learning
- Association Rule Mining, Decision Trees

We Did NOT Cover

- Deductive methods: forward chaining, backward chaining, planning
- Inductive logic programming
- Learning Theory (mistake bounds, sample bounds)
- Radial basis functions
- Gaussian Processes
- Exponential families
- Boosting methods