5. Introduction to Data Mining

5.1 Introduction
5.2 Building a classification tree
  5.2.1 Information theoretic background
  5.2.2 Building the tree
5.3. Mining association rule
[5.4 Clustering]

using material from A. Kemper,
A.W. Moore, CMU www.cs.cmu.edu/~awm (excellent intro to data mining!)

Data Mining

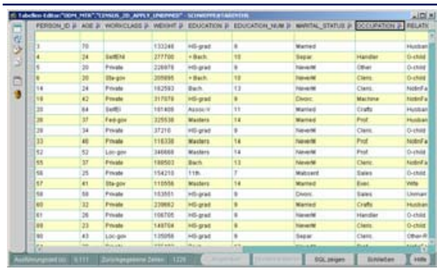
- Data Mining is all about automating the process of searching for patterns in the data.

A. Moore

Which patterns are interesting?

- How do we find them?

5.1 Introduction



- Large amount of data
- Find "hidden knowledge" – e.g. correlations between attributes
- Statistical techniques
- Challenge for DB technology: scalable algorithms for very large data sets

Introduction

- Typical Mining tasks

Classification

Given set of data and a set of classes. Assign data object to one class according to its characteristics (e.g. values)

find risk dependent on age, sex, make, horsepower
risk = 'high' or 'low' in db of car insurance

Methods: Decision tree of data set
Naïve Bayes
Adaptive Bayes

Goal: prediction of attribute value x=c dependent on predictor attributes

F(a1,...,an) = c

Sometimes written as classification rule :

(age<40) ∧ (sex =`m´) ∧ (make=`Golf GTI´) ∧ (hp > 100)
⇨ (risk=´high´)

Introduction

Association rules

Market basket analysis:
customer transaction data: `tid, time, {articles}`
Find rules X ⇨Y , with particular confidence
e.g. those buying sauce, meat and spaghetti
buy red wine with 0.7 probability.

Clustering

Group homogenous data into clusters
according to some similarity measure.
Not predefined as opposed to classification.

Data Mining

- **Which patterns are interesting?**

  What means "interesting"?
  Some quantitative measure?
- Which might be mere illusions?
- And how can they be exploited?

see A. Moore

- Data mining uses Machine Learning algorithms
- Well known since the 80's
- Challenge: apply to very large data sets

Introduction

- Data mining process
  - Data gathering, joining, reformatting
    e.g. Oracle: max 1000 attributes ⇨ transform into "transactional format": (id, attr_name, value)
  - Data cleansing
    - eliminate outliers
    - check correctness based on domain specific heuristics
    - check values in case of redundancy, ...
  - Build model (training phase). (Example: Decision tree)
  - Apply to new data

5.2 Building a decision tree

mpg	cylinders	displacement	horsepower	weight	acceleration	modelyear	maker
good	4	low	low	low	high	75to78	asia
bad	6	medium	medium	medium	medium	70to74	america
bad	4	medium	medium	medium	low	75to78	europe
bad	8	high	high	high	low	70to74	america
bad	6	medium	medium	medium	medium	70to74	america
bad	4	low	medium	low	medium	70to74	asia
bad	4	low	medium	low	low	70to74	asia
bad	8	high	high	high	low	75to78	america
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
:	:	:	:	:	:	:	:
bad	8	high	high	high	low	70to74	america
good	8	high	medium	high	high	79to83	america
bad	8	high	high	high	low	75to78	america
good	4	low	low	low	low	79to83	america
bad	6	medium	medium	medium	high	75to78	america
good	4	medium	low	low	low	79to83	america
good	4	low	low	medium	high	79to83	america
bad	8	high	high	high	low	70to74	america
good	4	low	medium	low	medium	75to78	europe
bad	5	medium	medium	medium	medium	75to78	europe

40 records

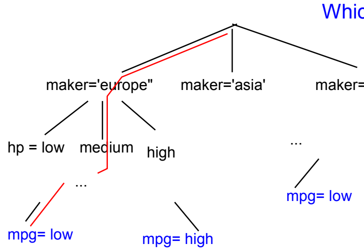
Miles per gallon: how can we predict mpg ("bad", "good") from the other attributes

example by A.Moore, data by R. Quinlan

Building a decision tree

- Wanted: tree which allows to predict value of an x given the values of the other attributes a1,...an
- Given: a training set – attribute value of x known

How to construct the tree?
Which attribute to start with?



Building a decision tree

Simple binary partitioning
D = Data set, n = node (root), a attribute
Prediction attribute x

```
BuildTree(n,D,a)
  split D according to a into D1, D2   -- binary!
   for each child D_i {
       if (x==const for all records in D_i
          OR no attribute can split D_i) make leaf node
       else
       { Chose "good" attribute b
         create children n1 and n2
         Partition Di into D_i1 und D_i2
         BuildTree(n1,D_i1,b)
        BuildTree(n2,D_i2,b) }
```

What is a "good" discriminating attribute?

5.2.1 Data mining and Information Theory

A short introduction to Information Theory
by Andrew W. Moore

Information theory:
- originally a "Theory of Communication" (C. Shannon)
- useful for data mining

Information Theory

- Huffman – Code
  Given an alphabet A = {a1,....,an} and probabilities of occurrence pi = p(ai) in a text for each ai.

  Find a binary code for A which minimizes
  $H'(A) = \Sigma$ pi * length (cw_i), cw_i = binary codeword of ai

  H'(A) is minimized for length$(cw_i) = \lceil \log_2 1/pi \rceil$

  well known how to construct it... ⇨ intro to algorithms

  $H(A) = - \Sigma\ pi * \log_2 pi$: important characterization of A
  what does it mean?

Entropy: interpretations

- Entropy

  $H(A) = - \Sigma\ pi * \log_2 pi$

  – minimal number of bits to encode A

  sender → receiver

  – amount of uncertainty of receiver before seeing an event (a character transmitted)

  – amount of surprise when seeing the event

  – the amount of information gained after receiving the event.

Information Theory and alphabets

- Example

  L = {A,C,T,G}, p(A) = p(C) = p(T) = p(G) = ¼,

  Boring: seeing a "T" in a sequence is as interesting as seeing a "G" or seeing an "A".

  H(L) = - ¼ * Σ log 1 – log 4 = 2

  But:

  L' = {A,C,T,G} , p(A) = 0.7, p(C) = 0.2 , p(T)= p(G) = 0.05

  Seeing a "T" or a "G" is exciting as opposed to "A"

  H(L') = -(-0.7*0,514 – 0.2*2.31- 2* 0.05*4.32)
  = 0.36 + 0.464 + 0.432 = 1.256

  Low entropy more interesting

Histograms and entropy

```
SELECT Count(*), education
FROM Census_2d_apply_unbinned
GROUP BY education;
```

29	10th
36	11th
15	12th
7	1st-4th
13	5th-6th
17	7th-8th
21	9th
241	< Bach.
44	Assoc-A
40	Assoc-V
202	Bach.
433	HS-grad
88	Masters
6	PhD
3	Presch.
31	Profsc

H(education) = 2.872

```
SELECT Count(*), Marital_status
FROM Census_2d_apply_unbinned
GROUP BY Marital_status;
```

161	Divorc.
20	Mabsent
3	Mar-AF
587	Married
380	NeverM
43	Separ.
32	Widowed

H(status)= 1.842

COUNT(*)	SEX
406	Female
820	Male

0.916

taken from Oracle DM data set / census data

	X	Y
14	9th	Male
7	9th	Female
6	PhD	Male
19	10th	Male
10	10th	Female
23	11th	Male
13	11th	Female
9	12th	Male
6	12th	Female
137	Bach.	Male
65	Bach.	Female
26	Profsc	Male
5	Profsc	Female
3	1st-4th	Male
4	1st-4th	Female
9	5th-6th	Male
4	5th-6th	Female
13	7th-8th	Male
4	7th-8th	Female
158	< Bach.	Male
83	< Bach.	Female
27	Assoc-A	Male
17	Assoc-A	Female
33	Assoc-V	Male
7	Assoc-V	Female
287	HS-grad	Male
146	HS-grad	Female
55	Masters	Male
33	Masters	Female
1	Presch.	Male
2	Presch.	Female

What can we say about Y if we know X?

Special conditional entropy:
H(Y | X= val) is entropy for those records having X= val

e.g. H(Y | X = 'Profsc')
= 26/31 * log 31/26 + 5/31 * log 31/5 = 0.637
(31 records)

Conditional entropy:
Σ Prob (X=xi) * H(Y | X= xi) is the average conditional entropy of Y

e.g. H(Y | X) = H(education|sex) = 0.909

Information gain

- What does the knowledge of X tell us about the value of Y?
- Or: Given the value of X, how much does the surprise of seeing an Y event decrease?
- Or: If sender and receiver know value of X, how much bits are required to encode Y?

$$\boxed{IG (Y \mid X) = H(Y) - H(Y|X)}$$

e.g. IG (education | sex) =
H(education) - H(education|sex) = 2.872 - 0,909 = 1.86

e.g. IG (maritalStatus | sex)
= H(status) - H(status|sex) = 1.842 – 0.717 = 1.125

Information gain: what for?

- Suppose you are trying to predict whether someone is going live past 80 years. From historical data you might find…

  – IG(LongLife | HairColor) = 0.01
  – IG(LongLife | Smoker) = 0.2
  – IG(LongLife | Gender) = 0.25
  – IG(LongLife | LastDigitOfSSN) = 0.00001

- IG tells you how interesting a 2-d contingency table is going to be.

Contingency tables

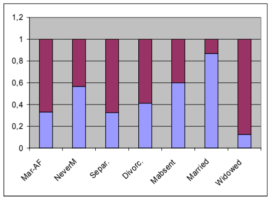
For each pair of values for attributes (status, sex) we can see how many records match (2-dimensional)

What is a k-dim contingency table? Any difference to data cube?

SQL groups normalized visualization

COUNT(*)	MARITAL_STAT	SEX
1	Mar-AF	Male
2	Mar-AF	Female
214	NeverM	Male
166	NeverM	Female
14	Separ.	Male
29	Separ.	Female
66	Divorc.	Male
95	Divorc.	Female
12	Mabsent	Male
8	Mabsent	Female
509	Married	Male
78	Married	Female
4	Widowed	Male
28	Widowed	Female

Normalized contingency table for census data



5.2.2 Building a decision tree

Remember

Decision tree is a plan to test attribute values in a particular sequence in order to predict the binary target value

Example: predict miles per gallon (low, high) depending on horse power, number of cylinders, make, ...

Constructing the tree from training set

In each step:

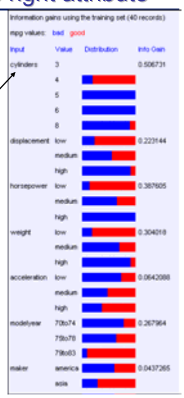
- chose attribute which has highest information gain

Construction of DT: choosing the right attribute

Contingency tables and information gain for mpg and a second attribute

The winner is:

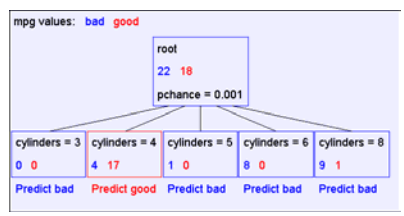
IG (cyl) = H(mpg) – H(mpg | cyl)



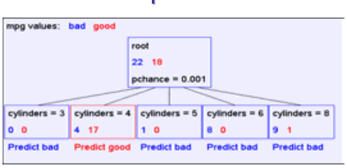
0 0
4 17
1 0
8 0
9 1

example and graphics by A. Moore

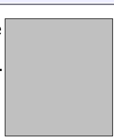
Building the tree



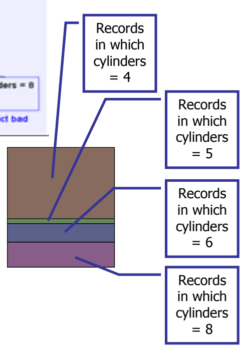
Recursion Step



Take the Original Dataset..

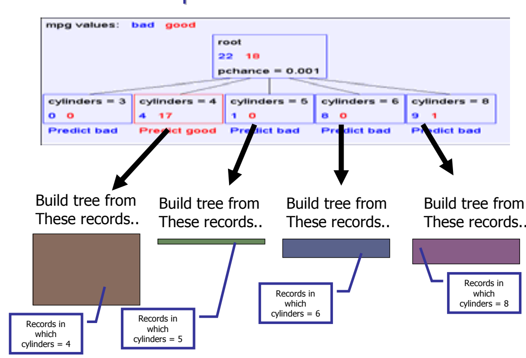


And partition it according to the value of the attribute we split on



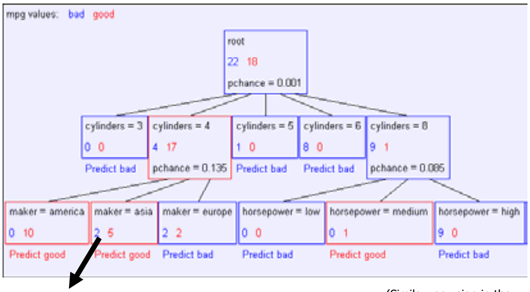
slide by A. Moore

Recursion Step



slide by A. Moore

Second level of tree

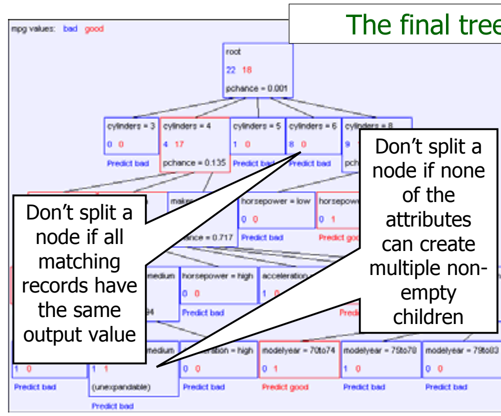


Recursively build a tree from the seven records in which there are four cylinders and the maker was based in Asia

(Similar recursion in the other cases)

slide by A. Moore

The final tree



slide by A. Moore

DT construction algorithm

BuildTree(*DataSet,Output*)

- If all output values are the same in *DataSet*, return a leaf node that says "predict this unique output"
- If all input values are the same, return a leaf node that says "predict the majority output"
- Else find attribute *X* with highest Info Gain
- Suppose *X* has n_X distinct values (i.e. X has arity n_X).
  - Create and return a non-leaf node with n_X children.
  - The *i*'th child should be built by calling

    BuildTree(DS_i,*Output*)

    Where DS_i built consists of all those records in DataSet for which X = *i*th distinct value of X.

slide by A. Moore

Errors

Training set error

- Check with records of training set if predicted value equals known value in record

Test set error

- use only subset of training set for tree construction
- Predict output value ("mpg") and compare with the known value
- Check attribute to be predicted in training set

  If prediction wrong: test set error

- For detailed analysis of errors etc see tutorial of A. Moore

Training set error much smaller than test set error – why?

	Num Errors	Set Size	Percent Wrong
Training Set	1	40	2.50
Test Set	74	352	21.02

Decision trees: conclusion

- Simple, important data mining tool
- Easy to understand, construct, use
- no prior assumptions on data
- predicts categorial date from categorial and / or numerical data
- applied to real life problems
- produce rules which can be easily interpreted

But:

- only categorial output value
- overfitting: paying too much attention to irrelevant attributes

  ... but not known in advance, which data are "noise"

  ⇨ statistical tests

5.3 Association rules: a short introduction

- Goal: discover co-occurence of items in large volumes of data ("market basket analysis")

  Example: how many customers by a printer together with their PC

- Non supervised learning
- Measures:
  - support (A ⇨ B) = P(A,B)

    how often co-occur A and B in the data set

    e.g. 0.05 if 5 % of all customers bought a printer and a PC
  - confidence (A ⇨ B) = P (B | A)

    fraction of customers, who bought a PC and also bought a printer , e.g. 0.8: 4 of 5 bought also printer

A Priori algorithm for finding associations

Transactionen	
TransID	Product
111	printer
111	paper
111	PC
111	toner
222	PC
222	scanner
333	printer
333	paper
333	toner
444	printer
444	PC
555	printer
555	paper
555	PC
555	scanner
555	toner

Goal: Find all rules A ⇨ B with support >= minSupport and confidence >= minConfidence

Algorithm first finds all frequent items :
FI = { p | p occurs in at least minSupport transactions}

All subsets of FI are also frequent item sets.

example adapted from Kemper

A Priori Algorithm

for all products p {
 if p occurs more than minSupport make frequent item set with one element: $F_1^p = \{p\}$ }
k = 1
repeat {
 for each Fk with k products generate candidates Fk+1 with k+1 products and Fk $\subseteq$ Fk+1.
 check in database, which candidates occur at least minSupport times; (sequential scan of DB)
 k = k+1 }
until no new frequent item set found

Transactionen	
TransID	Product
111	printer
111	paper
111	PC
111	toner
222	PC
222	scanner
333	printer
333	paper
333	toner
444	printer
444	PC
555	printer
555	paper
555	PC
555	scanner
555	toner

minSupport =3

Temporary results	
FI-candidate	#
{printer}	4
{paper }	3
{PC}	4
{scanner}	2
{toner}	3
{printer, paper}	3
{printer, PC}	3
{printer, Scanner}	
{printer, Toner}	3
{paper, PC}	2
{paper, Scanner}	
{paper, toner}	3
{PC, scanner}	
{PC,toner}	2
{scanner, toner}	

example adapted from Kemper

A Priori-Algorithmus

Transactionen	
TransID	Product
111	printer
111	paper
111	PC
111	toner
222	PC
222	scanner
333	printer
333	paper
333	toner
444	printer
444	PC
555	printer
555	paper
555	PC
555	scanner
555	toner

Zwischenergebnisse	
FI-Kandidat	Anzahl
{printer, paper}	3
{printer, PC}	3
{printer, acanner}	
{printer, toner}	3
{paper, PC}	2
{paper, scanner}	
{paper, toner}	3
{PC, acanner}	
{PC,toner}	2
{scanner, toner}	
{printer, paper, PC}	2
{printer, paper, toner}	3
{printer, PC, toner}	2
{paper, PC, toner}	2

example adapted from Kemper

Generate association rules

Given: set of FI of frequent items
for each FI with support >= minSupport:
 { for each subset L $\subset$ FI
  define rule R : L ⇨ FI \ L
  confidence (R) = support FI / support L
  if confidence(R) >= minConfidence: keep L
 }

Example:
 FI = {printer, paper, toner}
  Support = 3
 Rule: {printer} ⇨ {paper, toner},
  Confidence = Support({printer, paper, toner}) / Support({printer})
  = (3/5) / (4/5)
  = ¾ = 75 %

example adapted from Kemper

Increase of confidence

- Increase of Left hand side (i.e. decrease of right hand side) of a rule increases confidence
  $L \subset L^+, \mathbf{R} \subset \mathbf{R}^- \Rightarrow$ Confidence(L ⇨ R) <= C(L^+ ⇨ $\mathbf{R}^-$)
- Rule: {printer} ⇨ {paper, toner}
  confidence = support({printer, paper, toner}) / support({printer})
  = (3/5) / (4/5)
  = ¾ = 75%
- Rule: {printer,paper} ⇨ {toner}
  confidence = S({printer, paper, Toner}) / S({printer,paper})
  = (3/5) / (3/5)
  = 1 = 100%

example adapted from Kemper

Summary data mining

- important statistical technique
- basis algorithms from machine learning
- many different methods and algorithms
- distinction supervised versus unsupervised learning
- efficient implementation on very large data sets essential
- Enormous commercial interest
  (business transactions, web logs, ...)