Data Mining

(with many slides due to Gehrke, Garofalakis, Rastogi)

Raghu Ramakrishnan
Yahoo! Research
University of Wisconsin–Madison (on leave)

Introduction

Definition

Data mining is the exploration and analysis of large quantities of data in order to discover valid, novel, potentially useful, and ultimately understandable patterns in data.

Valid: The patterns hold in general.
Novel: We did not know the pattern beforehand.
Useful: We can devise actions from the patterns.
Understandable: We can interpret and comprehend the patterns.

Case Study: Bank

- Business goal: Sell more home equity loans
- Current models:
  – Customers with college-age children use home equity loans to pay for tuition
  – Customers with variable income use home equity loans to even out stream of income
- Data:
  – Large data warehouse
  – Consolidates data from 42 operational data sources

Case Study: Bank (Contd.)

1. Select subset of customer records who have received home equity loan offer
   – Customers who declined
   – Customers who signed up

Income	Number of Children	Average Checking Account Balance	…	Reponse
$40,000	2	$1500		Yes
$75,000	0	$5000		No
$50,000	1	$3000		No
…	…	…	…	…

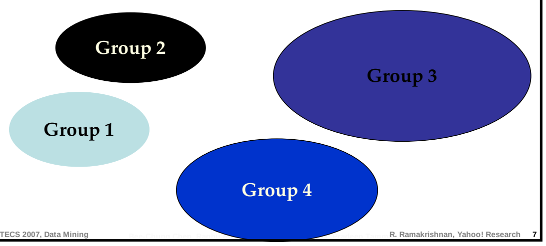
Case Study: Bank (Contd.)

2. Find rules to predict whether a customer would respond to home equity loan offer

IF (Salary < 40k) and
(numChildren > 0) and
(ageChild1 > 18 and ageChild1 < 22)
THEN YES
…

Case Study: Bank (Contd.)

3. Group customers into clusters and investigate clusters



Case Study: Bank (Contd.)

4. Evaluate results:
   - Many “uninteresting” clusters
   - One interesting cluster! Customers with both business and personal accounts; unusually high percentage of likely respondents

Example: Bank (Contd.)

Action:

- New marketing campaign

Result:

- Acceptance rate for home equity offers more than doubled

Example Application: Fraud Detection

- Industries: Health care, retail, credit card services, telecom, B2B relationships
- Approach:
  - Use historical data to build models of fraudulent behavior
  - Deploy models to identify fraudulent instances

Fraud Detection (Contd.)

- Examples:
  - Auto insurance: Detect groups of people who stage accidents to collect insurance
  - Medical insurance: Fraudulent claims
  - Money laundering: Detect suspicious money transactions (US Treasury's Financial Crimes Enforcement Network)
  - Telecom industry: Find calling patterns that deviate from a norm (origin and destination of the call, duration, time of day, day of week).

Other Example Applications

- CPG: Promotion analysis
- Retail: Category management
- Telecom: Call usage analysis, churn
- Healthcare: Claims analysis, fraud detection
- Transportation/Distribution: Logistics management
- Financial Services: Credit analysis, fraud detection
- Data service providers: Value-added data analysis

What is a Data Mining Model?

A data mining model is a description of a certain aspect of a dataset. It produces output values for an assigned set of inputs.

Examples:
- Clustering
- Linear regression model
- Classification model
- Frequent itemsets and association rules
- Support Vector Machines

Data Mining Methods

Overview

- Several well-studied tasks
  - Classification
  - Clustering
  - Frequent Patterns
- Many methods proposed for each
- Focus in database and data mining community:
  - Scalability
  - Managing the process
  - Exploratory analysis

Classification

Goal:

Learn a function that assigns a record to one of several predefined classes.

Requirements on the model:
- High accuracy
- Understandable by humans, interpretable
- Fast construction for very large training databases

Classification

Example application: telemarketing

Classification (Contd.)

- Decision trees are one approach to classification.
- Other approaches include:
  - Linear Discriminant Analysis
  - *k*-nearest neighbor methods
  - Logistic regression
  - Neural networks
  - Support Vector Machines

Classification Example

- Training database:
  - Two predictor attributes: Age and Car-type (**S**port, **M**inivan and **T**ruck)
  - Age is ordered, Car-type is categorical attribute
  - Class label indicates whether person bought product
  - Dependent attribute is *categorical*

Age	Car	Class
20	M	Yes
30	M	Yes
25	T	No
30	S	Yes
40	S	Yes
20	T	No
30	M	Yes
25	M	Yes
40	M	Yes
20	S	No

Types of Variables

- *Numerical*: Domain is ordered and can be represented on the real line (e.g., age, income)
- *Nominal* or *categorical*: Domain is a finite set without any natural ordering (e.g., occupation, marital status, race)
- *Ordinal*: Domain is ordered, but absolute differences between values is unknown (e.g., preference scale, severity of an injury)

Definitions

- Random variables $X_1, \ldots, X_k$ (*predictor variables*) and Y (*dependent variable*)
- X_i has domain dom(X_i), Y has domain dom(Y)
- P is a probability distribution on dom(X_1) x … x dom(X_k) x dom(Y)
  Training database D is a random sample from P
- A *predictor* d is a function
  d: dom(X_1) … dom(X_k) → dom(Y)

Classification Problem

- If Y is categorical, the problem is a *classification problem*, and we use C instead of Y. |dom(C)| = J, the number of classes.
- C is the *class label*, d is called a *classifier.*
- Let r be a record randomly drawn from P.
  Define the *misclassification rate* of d:
  RT(d,P) = P(d(r.X_1, …, r.X_k) != r.C)
- Problem definition: Given dataset D that is a random sample from probability distribution P, find classifier d such that RT(d,P) is minimized.

Regression Problem

- If Y is numerical, the problem is a *regression problem.*
- Y is called the dependent variable, d is called a *regression function.*
- Let r be a record randomly drawn from P.
  Define mean squared error rate of d:
  RT(d,P) = E(r.Y - d(r.X_1, …, r.X_k))^2
- Problem definition: Given dataset D that is a random sample from probability distribution P, find regression function d such that RT(d,P) is minimized.

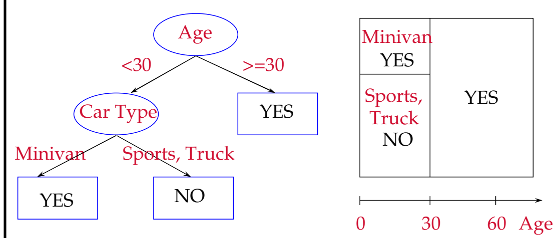
Regression Example

- Example training database
  - Two predictor attributes: Age and Car-type (**S**port, **M**inivan and **T**ruck)
  - Spent indicates how much person spent during a recent visit to the web site
  - Dependent attribute is *numerical*

Age	Car	Spent
20	M	$200
30	M	$150
25	T	$300
30	S	$220
40	S	$400
20	T	$80
30	M	$100
25	M	$125
40	M	$500
20	S	$420

Decision Trees

What are Decision Trees?



Decision Trees

- A *decision tree* T encodes d (a classifier or regression function) in form of a tree.
- A node t in T without children is called a *leaf node*. Otherwise t is called an *internal node*.

Internal Nodes

- Each internal node has an associated splitting predicate. Most common are binary predicates. Example predicates:
  - Age <= 20
  - Profession in {student, teacher}
  - 5000*Age + 3*Salary – 10000 > 0

Internal Nodes: Splitting Predicates

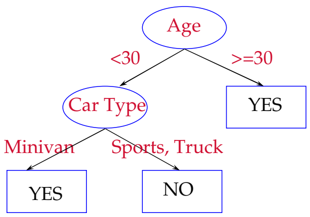
- Binary Univariate splits:
  - Numerical or ordered X: X <= c, c in dom(X)
  - Categorical X: X in A, A subset dom(X)
- Binary Multivariate splits:
  - Linear combination split on numerical variables: $\Sigma a_i X_i <= c$
- k-ary (k>2) splits analogous

Leaf Nodes

Consider leaf node t:

- Classification problem: Node t is labeled with one class label c in dom(C)
- Regression problem: Two choices
  - Piecewise constant model: t is labeled with a constant y in dom(Y).
  - Piecewise linear model: t is labeled with a linear model $Y = y_t + \Sigma a_i X_i$

Example



Encoded classifier:
If (age<30 and
carType=Minivan)
Then YES
If (age <30 and
(carType=Sports or
carType=Truck))
Then NO
If (age >= 30)
Then YES

Issues in Tree Construction

- Three algorithmic components:
  - Split Selection Method
  - Pruning Method
  - Data Access Method

Top-Down Tree Construction

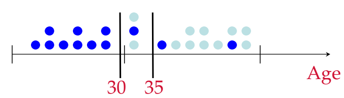
BuildTree(Node *n*, Training database *D*,
Split Selection Method ***S***)

[(1) Apply ***S*** to *D* to find splitting criterion]
(1a) **for** each predictor attribute *X*
(1b) Call ***S***.findSplit(AVC-set of *X*)
(1c) **endfor**
(1d) ***S***.chooseBest();
(2) **if** (*n* is not a leaf node) ...

S: C4.5, CART, CHAID, FACT, ID3, GID3, QUEST, etc.

Split Selection Method

- Numerical Attribute: Find a split point that separates the (two) classes



(Yes: ● No: ●)

Split Selection Method (Contd.)

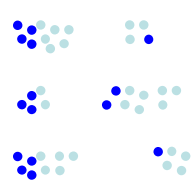
- Categorical Attributes: How to group?

Sport:  Truck: Minivan:

(Sport, Truck) -- (Minivan)

(Sport) --- (Truck, Minivan)

(Sport, Minivan) --- (Truck)



Impurity-based Split Selection Methods

- Split selection method has two parts:
  - Search space of possible splitting criteria. Example: All splits of the form "age <= c".
  - Quality assessment of a splitting criterion
- Need to quantify the quality of a split: Impurity function
- Example impurity functions: Entropy, gini-index, chi-square index

Data Access Method

- Goal: Scalable decision tree construction, using the complete training database

AVC-Sets

Training Database

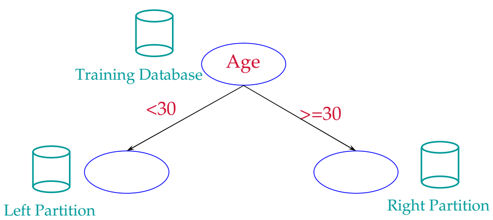
Age	Car	Class
20	M	Yes
30	M	Yes
25	T	No
30	S	Yes
40	S	Yes
20	T	No
30	M	Yes
25	M	Yes
40	M	Yes
20	S	No

AVC-Sets

Age	Yes	No
20	1	2
25	1	1
30	3	0
40	2	0

Car	Yes	No
Sport	2	1
Truck	0	2
Minivan	5	0

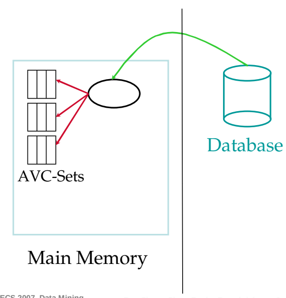
Motivation for Data Access Methods



In principle, one pass over training database for each node.
Can we improve?

RainForest Algorithms: RF-Hybrid

First scan:

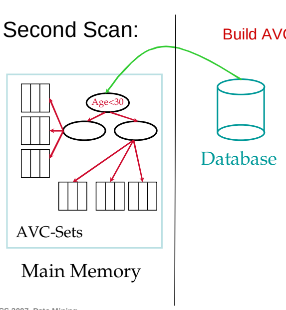


Build AVC-sets for root

RainForest Algorithms: RF-Hybrid

Second Scan:

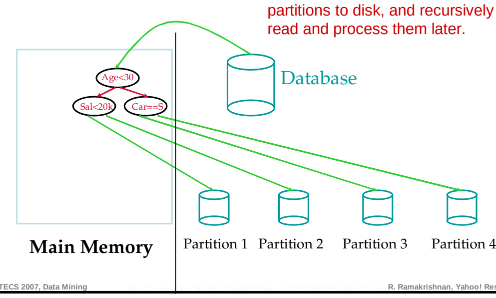
Build AVC sets for children of the root



RainForest Algorithms: RF-Hybrid

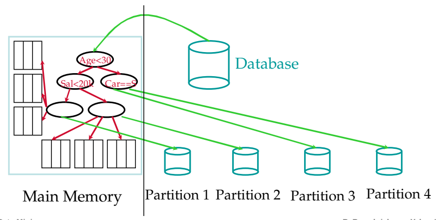
Third Scan:

As we expand the tree, we run out Of memory, and have to “spill” partitions to disk, and recursively read and process them later.



RainForest Algorithms: RF-Hybrid

Further optimization: While writing partitions, concurrently build AVC-groups of as many nodes as possible in-memory. This should remind you of Hybrid Hash-Join!



CLUSTERING

Problem

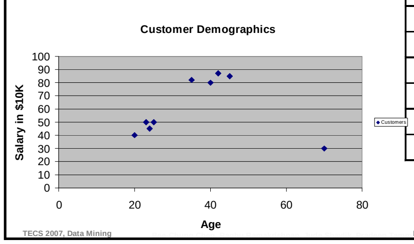
- Given points in a multidimensional space, group them into a small number of clusters, using some measure of "nearness"
  - E.g., Cluster documents by topic
  - E.g., Cluster users by similar interests

Clustering

- Output: (k) groups of records called clusters, such that the records within a group are more similar to records in other groups
  - Representative points for each cluster
  - Labeling of each record with cluster number
  - Other description of each cluster
- *This is unsupervised learning*: No record labels are given to learn from
- Usage:
  - Exploratory data mining
  - Preprocessing step (e.g., outlier detection)

Clustering (Contd.)

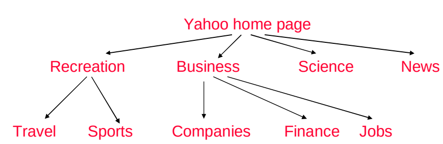
- Example input database: Two numerical variables
- How many groups are here?



Age	Salary
20	40
25	50
24	45
23	50
40	80
45	85
42	87
35	82
70	30

Improve Search Using Topic Hierarchies

- Web directories (or topic hierarchies) provide a hierarchical classification of documents (e.g., Yahoo!)



- Searches performed in the context of a topic restricts the search to only a subset of web pages related to the topic
- Clustering can be used to generate topic hierarchies

Clustering (Contd.)

- Requirements: Need to define "similarity" between records
- Important: Use the "right" similarity (distance) function
  - Scale or normalize all attributes. Example: seconds, hours, days
  - Assign different weights to reflect importance of the attribute
  - Choose appropriate measure (e.g., L1, L2)

Distance Measure D

- For 2 pts x and y:
  - D(x,x) = 0
  - D(x,y) = D(y,x)
  - D(x,y) <= D(x,z)+D(z,y), for all z
- Examples, for x,y in k-dim space:
  - L1: Sum of |xi-yi| over I = 1 to k
  - L2: Root-mean squared distance

Approaches

- Centroid-based: Assume we have k clusters, guess at the centers, assign points to nearest center, e.g., K-means; over time, centroids shift
- Hierarchical: Assume there is one cluster per point, and repeatedly merge nearby clusters using some distance threshold

Scalability: Do this with fewest number of passes over data, ideally, sequentially

K-means Clustering Algorithm

- Choose *k* initial means
- Assign each point to the cluster with the closest mean
- Compute new mean for each cluster
- Iterate until the *k* means stabilize

Agglomerative Hierarchical Clustering Algorithms

- Initially each point is a distinct cluster
- Repeatedly merge closest clusters until the number of clusters becomes *k*
  - Closest: $d_{mean}(C_i, C_j) = \|m_i - m_j\|$

    $d_{min}(C_i, C_j) = \min_{p \in C_i, q \in C_j} \|p - q\|$

    Likewise $d_{ave}(C_i, C_j)$ and $d_{max}(C_i, C_j)$

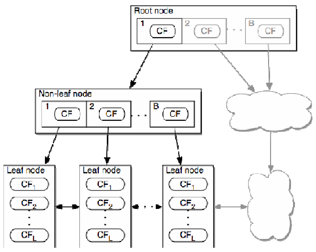
Scalable Clustering Algorithms for Numeric Attributes

CLARANS
DBSCAN
BIRCH
CLIQUE
CURE
…….

- Above algorithms can be used to cluster documents after reducing their dimensionality using SVD

Birch [ZRL96]

Pre-cluster data points using “CF-tree” data structure



BIRCH [ZRL 96]

- Pre-cluster data points using “CF-tree” data structure
  - CF-tree is similar to R-tree
  - For each point
    - CF-tree is traversed to find the closest cluster
    - If the cluster is within epsilon distance, the point is absorbed into the cluster
    - Otherwise, the point starts a new cluster
- Requires only single scan of data
- Cluster summaries stored in CF-tree are given to main memory clustering algorithm of choice

Background

Given a cluster of instances $\{\vec{X}_i\}$, *we define:*

Centroid $\vec{X}0 = \frac{\sum_{i=1}^{N} \vec{X}_i}{N}$

Radius $R = (\frac{\sum_{i=1}^{N} (\vec{X}_i - \vec{X}0)^2}{N})^{\frac{1}{2}}$

Diameter $D = (\frac{\sum_{i=1}^{N} \sum_{j=1}^{N} (\vec{X}_i - \vec{X}_j)^2}{N(N-1)})^{\frac{1}{2}}$

(Euclidean) Distance $D0 = ((\vec{X}0_1 - \vec{X}0_2)^2)^{\frac{1}{2}}$

The Algorithm: Background

We define the ***Euclidean*** *and* ***Manhattan*** *distance between any two clusters as:*

$$D0 = ((\vec{X}0_1 - \vec{X}0_2)^2)^{\frac{1}{2}}$$

$$D1 = |\vec{X}0_1 - \vec{X}0_2| = \sum_{i=1}^{d} |\vec{X}0_1^{(i)} - \vec{X}0_1^{(i)}|$$

Clustering Feature (CF)

Given a cluster $\{\vec{X}_1, \vec{X}_2, \ldots, \vec{X}_N\}$ $\quad \mathbf{CF} = (N, \vec{LS}, SS)$

N is the number of data points

$\vec{LS} = \sum_{i=1}^{N} \vec{X}_i$

$SS = \sum_{i=1}^{N} \vec{X}_i^{\,2}$

$$\mathbf{CF_1} + \mathbf{CF_2} = (N_1 + N_2, \vec{LS}_1 + \vec{LS}_2, SS_1 + SS_2)$$

Allows incremental merging of clusters!

Points to Note

- Basic algorithm works in a single pass to condense metric data using spherical summaries
  - Can be incremental
- Additional passes cluster CFs to detect non-spherical clusters
- Approximates density function
- Extensions to non-metric data

CURE [GRS 98]

- Hierarchical algorithm for dicovering arbitrary shaped clusters
  - Uses a small number of representatives per cluster
  - Note:
    - Centroid-based: Uses 1 point to represent a cluster => Too little information … Hyper-spherical clusters
    - MST-based: Uses every point to represent a cluster =>Too much information ... Easily mislead
- Uses random sampling
- Uses Partitioning
- Labeling using representatives

Cluster Representatives

A representative set of points:

- Small in number : c
- Distributed over the cluster
- Each point in cluster is close to one representative
- Distance between clusters:

smallest distance between representatives

Market Basket Analysis: Frequent Itemsets

Market Basket Analysis

- Consider shopping cart filled with several items
- Market basket analysis tries to answer the following questions:
  - Who makes purchases
  - What do customers buy

Market Basket Analysis

- Given:
  - A database of customer transactions
  - Each transaction is a set of items
- Goal:
  - Extract rules

TID	CID	Date	Item	Qty
111	201	5/1/99	Pen	2
111	201	5/1/99	Ink	1
111	201	5/1/99	Milk	3
111	201	5/1/99	Juice	6
112	105	6/3/99	Pen	1
112	105	6/3/99	Ink	1
112	105	6/3/99	Milk	1
113	106	6/5/99	Pen	1
113	106	6/5/99	Milk	1
114	201	7/1/99	Pen	2
114	201	7/1/99	Ink	2
114	201	7/1/99	Juice	4

Market Basket Analysis (Contd.)

- Co-occurrences
  - 80% of all customers purchase items X, Y and Z together.
- Association rules
  - 60% of all customers who purchase X and Y also buy Z.
- Sequential patterns
  - 60% of customers who first buy X also purchase Y within three weeks.

Confidence and Support

We prune the set of all possible association rules using two interestingness measures:

- Confidence of a rule:
  - X => Y has confidence c if P(Y|X) = c
- Support of a rule:
  - X => Y has support s if P(XY) = s

We can also define

- Support of a co-ocurrence XY:
  - XY has support s if P(XY) = s

Example

- Example rule:
  {Pen} => {Milk}
  Support: 75%
  Confidence: 75%
- Another example:
  {Ink} => {Pen}
  Support: 100%
  Confidence: 100%

TID	CID	Date	Item	Qty
111	201	5/1/99	Pen	2
111	201	5/1/99	Ink	1
111	201	5/1/99	Milk	3
111	201	5/1/99	Juice	6
112	105	6/3/99	Pen	1
112	105	6/3/99	Ink	1
112	105	6/3/99	Milk	1
113	106	6/5/99	Pen	1
113	106	6/5/99	Milk	1
114	201	7/1/99	Pen	2
114	201	7/1/99	Ink	2
114	201	7/1/99	Juice	4

Exercise

- Can you find all itemsets with support >= 75%?

TID	CID	Date	Item	Qty
111	201	5/1/99	Pen	2
111	201	5/1/99	Ink	1
111	201	5/1/99	Milk	3
111	201	5/1/99	Juice	6
112	105	6/3/99	Pen	1
112	105	6/3/99	Ink	1
112	105	6/3/99	Milk	1
113	106	6/5/99	Pen	1
113	106	6/5/99	Milk	1
114	201	7/1/99	Pen	2
114	201	7/1/99	Ink	2
114	201	7/1/99	Juice	4

Exercise

- Can you find all association rules with support >= 50%?

TID	CID	Date	Item	Qty
111	201	5/1/99	Pen	2
111	201	5/1/99	Ink	1
111	201	5/1/99	Milk	3
111	201	5/1/99	Juice	6
112	105	6/3/99	Pen	1
112	105	6/3/99	Ink	1
112	105	6/3/99	Milk	1
113	106	6/5/99	Pen	1
113	106	6/5/99	Milk	1
114	201	7/1/99	Pen	2
114	201	7/1/99	Ink	2
114	201	7/1/99	Juice	4

Extensions

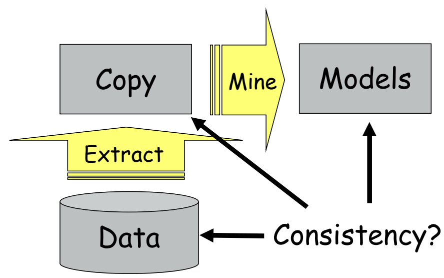
- Imposing constraints
  - Only find rules involving the dairy department
  - Only find rules involving expensive products
  - Only find rules with "whiskey" on the right hand side
  - Only find rules with "milk" on the left hand side
  - Hierarchies on the items
  - Calendars (every Sunday, every 1st of the month)

Market Basket Analysis: Applications

- Sample Applications
  - Direct marketing
  - Fraud detection for medical insurance
  - Floor/shelf planning
  - Web site layout
  - Cross-selling

DBMS Support for DM

Why Integrate DM into a DBMS?



Integration Objectives

- Avoid isolation of querying from mining
  - Difficult to do “ad-hoc” mining
- Provide simple programming approach to creating and using DM models

Analysts (users)

- Make it possible to add new models
- Make it possible to add new, scalable algorithms

DM Vendors

SQL/MM: Data Mining

- A collection of classes that provide a standard interface for invoking DM algorithms from SQL systems.
- Four data models are supported:
  - Frequent itemsets, association rules
  - Clusters
  - Regression trees
  - Classification trees

DATA MINING SUPPORT IN MICROSOFT SQL SERVER *

* Thanks to Surajit Chaudhuri for permission to use/adapt his slides

Key Design Decisions

- Adopt relational data representation
  - A Data Mining Model (DMM) as a “tabular” object (externally; can be represented differently internally)
- Language-based interface
  - Extension of SQL
  - Standard syntax

DM Concepts to Support

- Representation of input (*cases*)
- Representation of *models*
- Specification of *training step*
- Specification of *prediction step*

Should be independent of specific algorithms

What are "Cases"?

- DM algorithms analyze "cases"
- The "case" is the entity being categorized and classified
- Examples
  - Customer credit risk analysis: Case = Customer
  - Product profitability analysis: Case = Product
  - Promotion success analysis: Case = Promotion
- Each case encapsulates all we know about the entity

Cases as Records: Examples

Cust ID	Age	Marital Status	Wealth
1	35	M	380,000
2	20	S	50,000
3	57	M	470,000

Age	Car	Class
20	M	Yes
30	M	Yes
25	T	No
30	S	Yes
40	S	Yes
20	T	No
30	M	Yes
25	M	Yes
40	M	Yes
20	S	No

Types of Columns

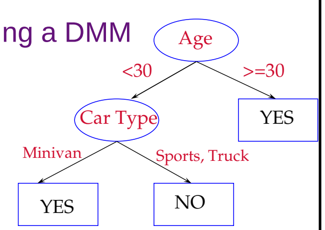
Cust ID	Age	Marital Status	Wealth	Product Purchases		
				Product	Quantity	Type
1	35	M	380,000	TV	1	Appliance
				Coke	6	Drink
				Ham	3	Food

- Keys: Columns that uniquely identify a case
- Attributes: Columns that describe a case
  - Value: A state associated with the attribute in a specific case
  - Attribute Property: Columns that describe an attribute
    - Unique for a specific attribute value (TV is always an appliance)
  - Attribute Modifier: Columns that represent additional "meta" information for an attribute
    - Weight of a case, Certainty of prediction

More on Columns

- Properties describe attributes
  - Can represent generalization hierarchy
- Distribution information associated with attributes
  - Discrete/Continuous
  - Nature of Continuous distributions
    - Normal, Log_Normal
  - Other Properties (e.g., ordered, not null)

Representing a DMM



- Specifying a Model
  - Columns to predict
  - Algorithm to use
  - Special parameters
- Model is represented as a (nested) table
  - Specification = Create table
  - Training = Inserting data into the table
  - Predicting = Querying the table

CREATE MINING MODEL

Name of model

```
CREATE MINING MODEL [Age Prediction]
(
[Gender]          TEXT   DISCRETE   ATTRIBUTE,
[Hair Color]       TEXT  DISCRETE   ATTRIBUTE,
[Age]             DOUBLE CONTINUOUS ATTRIBUTE PREDICT,
)
USING [Microsoft Decision Tree]
```

Name of algorithm

CREATE MINING MODEL

```
CREATE MINING MODEL [Age Prediction]
(
[Customer ID]  LONG    KEY,
[Gender]             TEXT    DISCRETE   ATTRIBUTE,
[Age]                DOUBLE CONTINUOUS ATTRIBUTE PREDICT,
[ProductPurchases] TABLE (
[ProductName]  TEXT   KEY,
[Quantity]           DOUBLE NORMAL CONTINUOUS,
[ProductType]  TEXT DISCRETE RELATED TO [ProductName]
)
)
USING [Microsoft Decision Tree]
```

Note that the ProductPurchases column is a nested table.
SQL Server computes this field when data is "inserted".

Training a DMM

- Training a DMM requires passing it "known" cases
- Use an INSERT INTO in order to "insert" the data to the DMM
  - The DMM will usually not retain the inserted data
  - Instead it will analyze the given cases and build the DMM content (decision tree, segmentation model)

```
• INSERT [INTO] <mining model name>
  [(columns list)]
    <source data query>
```


INSERT INTO

```
INSERT INTO [Age Prediction]
(
[Gender],[Hair Color], [Age]
)
OPENQUERY([Provider=MSOLESQL…,
'SELECT
     [Gender], [Hair Color], [Age]
 FROM [Customers]'
)
```


Executing Insert Into

- The DMM is trained
  - The model can be retrained or incrementally refined
- Content (rules, trees, formulas) can be explored
- Prediction queries can be executed

What are Predictions?

- Predictions apply the trained model to estimate missing attributes in a data set
- Predictions = Queries
- Specification:
  - Input data set
  - A trained DMM (think of it as a truth table, with one row per combination of predictor-attribute values; this is only conceptual)
  - Binding (mapping) information between the input data and the DMM

Prediction Join

```
SELECT [Customers].[ID],
      MyDMM.[Age],
      PredictProbability(MyDMM.[Age])
FROM
  MyDMM PREDICTION JOIN [Customers]
  ON MyDMM.[Gender] = [Customers].[Gender] AND
  MyDMM.[Hair Color] =
                        [Customers].[Hair Color]
```


Exploratory Mining: Combining OLAP and DM

Databases and Data Mining

- What can database systems offer in the grand challenge of understanding and learning from the flood of data we've unleashed?
  - The plumbing
  - Scalability

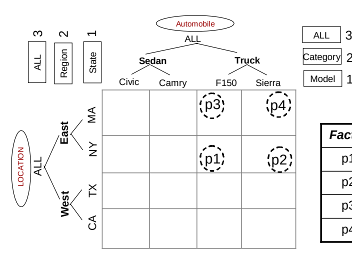
Databases and Data Mining

- What can database systems offer in the grand challenge of understanding and learning from the flood of data we've unleashed?
  - The plumbing
  - Scalability
  - Ideas!
    - Declarativeness
    - Compositionality
    - **Ways to conceptualize your data**

Multidimensional Data Model

- One fact table Δ=(**X**,**M**)
  - **X**=X_1, X_2, ... Dimension attributes
  - **M**=M_1, M_2,… Measure attributes
- Domain hierarchy for each dimension attribute:
  - Collection of domains Hier(X_i)= ($D_i^{(1)}$,..., $D_i^{(k)}$)
  - The extended domain: $EX_i = \cup_{1 \leq k \leq t} DX_i^{(k)}$
- Value mapping function: $\gamma_{D1 \rightarrow D2}(x)$
  - e.g., $\gamma_{month \rightarrow year}(12/2005) = 2005$
  - Form the value hierarchy graph
  - Stored as dimension table attribute (e.g., week for a time value) or conversion functions (e.g., month, quarter)

Multidimensional Data



DIMENSION ATTRIBUTES

FactID	*Auto*	*Loc*	*Repair*
p1	F150	NY	100
p2	Sierra	NY	500
p3	F150	MA	100
p4	Sierra	MA	200

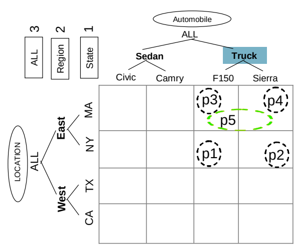
Cube Space

- Cube space: $C = EX_1 \times EX_2 \times \ldots \times EX_d$
- Region: Hyper rectangle in cube space
  - $c = (v_1, v_2, \ldots, v_d)$, $v_i \in EX_i$
- Region granularity:
  - gran(c) = (d_1, d_2, ..., d_d), d_i = Domain($c.v_i$)
- Region coverage:
  - coverage(c) = all facts in c
- Region set: All regions with same granularity

OLAP Over Imprecise Data

with Doug Burdick, Prasad Deshpande, T.S. Jayram, and Shiv Vaithyanathan
In VLDB 05, 06 joint work with IBM Almaden

Imprecise Data

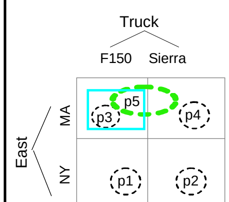


FactID	Auto	Loc	Repair
p1	F150	NY	100
p2	Sierra	NY	500
p3	F150	MA	100
p4	Sierra	MA	200
p5	Truck	MA	100

Querying Imprecise Facts

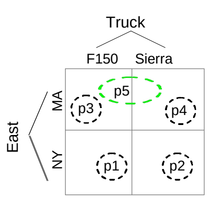
Auto = F150
Loc = MA
SUM(Repair) = ???

How do we treat p5?



FactID	Auto	Loc	Repair
p1	F150	NY	100
p2	Sierra	NY	500
p3	F150	MA	100
p4	Sierra	MA	200
p5	Truck	MA	100

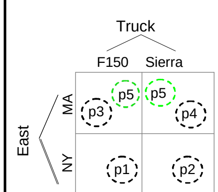
Allocation (1)



FactID	Auto	Loc	Repair
p1	F150	NY	100
p2	Sierra	NY	500
p3	F150	MA	100
p4	Sierra	MA	200
p5	Truck	MA	100

Allocation (2)

(Huh? Why 0.5 / 0.5?
- Hold on to that thought)

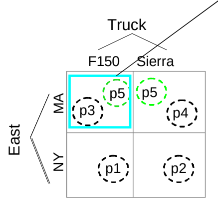


ID	FactID	Auto	Loc	Repair	Weight
1	p1	F150	NY	100	1.0
2	p2	Sierra	NY	500	1.0
3	p3	F150	MA	100	1.0
4	p4	Sierra	MA	200	1.0
5	p5	F150	MA	100	0.5
6	p5	Sierra	MA	100	0.5

Allocation (3)

Auto = F150
Loc = MA
SUM(Repair) = 150

Query the Extended Data Model!

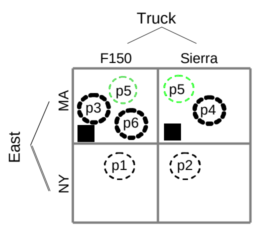


ID	FactID	Auto	Loc	Repair	Weight
1	p1	F150	NY	100	1.0
2	p2	Sierra	NY	500	1.0
3	p3	F150	MA	100	1.0
4	p4	Sierra	MA	200	1.0
5	p5	F150	MA	100	0.5
6	p5	Sierra	MA	100	0.5

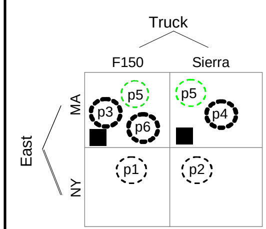
Allocation Policies

- The procedure for assigning allocation weights is referred to as an allocation policy:
  - Each allocation policy uses different information to assign allocation weights
  - Reflects assumption about the correlation structure in the data
    - Leads to EM-style iterative algorithms for allocating imprecise facts, maximizing likelihood of observed data

Allocation Policy: ***Count***

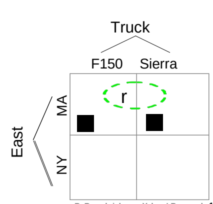


Allocation Policy: ***Measure***



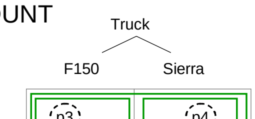
ID	Sales
p1	100
p2	150
p3	300
p4	200
p5	250
p6	400

Allocation Policy Template



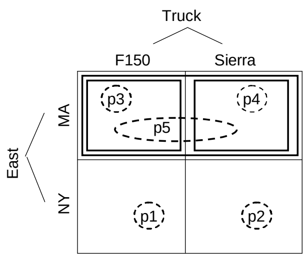
What is a Good Allocation Policy?

Query: COUNT



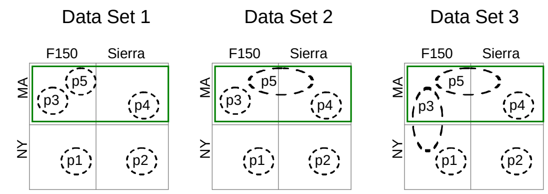
- We propose desiderata that enable appropriate definition of query semantics for imprecise data

Desideratum I: Consistency



- Consistency specifies the relationship between answers to related queries on a fixed data set

Desideratum II: Faithfulness



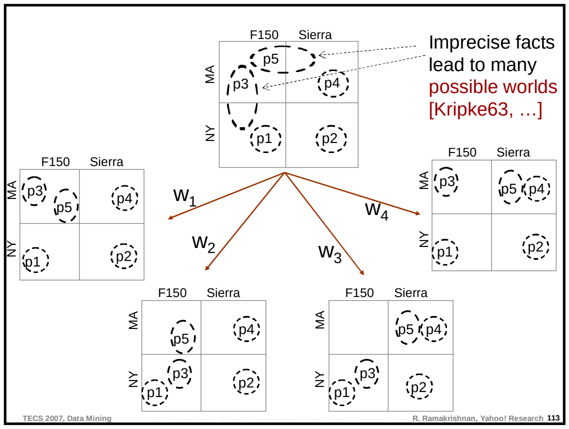
- Faithfulness specifies the relationship between answers to a fixed query on related data sets

Results on Query Semantics

- Evaluating queries over extended data model yields expected value of the aggregation operator over all possible worlds
- Efficient query evaluation algorithms available for SUM, COUNT; more expensive dynamic programming algorithm for AVERAGE
  - Consistency and faithfulness for SUM, COUNT are satisfied under appropriate conditions
  - (Bound-)Consistency does not hold for AVERAGE, but holds for E(SUM)/E(COUNT)
    - Weak form of faithfulness holds
  - Opinion pooling with LinOP: Similar to AVERAGE

Allocation Policies

- Procedure for assigning allocation weights is referred to as an allocation policy
  - Each allocation policy uses different information to assign allocation weight
- Key contributions:
  - Appropriate characterization of the large space of allocation policies (VLDB 05)
  - Designing efficient algorithms for allocation policies that take into account the correlations in the data (VLDB 06)



Imprecise facts lead to many possible worlds [Kripke63, …]

Query Semantics

- Given all possible worlds together with their probabilities, queries are easily answered using expected values
  - But number of possible worlds is exponential!
- Allocation gives facts weighted assignments to possible completions, leading to an extended version of the data
  - Size increase is linear in number of (completions of) imprecise facts
  - Queries operate over this extended version

Exploratory Mining: Prediction Cubes

with Beechun Chen, Lei Chen, and Yi Lin
In VLDB 05; EDAM Project

The Idea

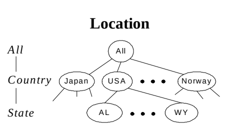
- Build OLAP data cubes in which cell values represent decision/prediction behavior
  - In effect, build a tree for each cell/region in the cube—observe that this is not the same as a collection of trees used in an ensemble method!
  - The idea is simple, but it leads to promising data mining tools
  - Ultimate objective: Exploratory analysis of the entire space of "data mining choices"
    - Choice of algorithms, data conditioning parameters …

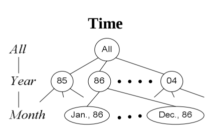
Example (1/7): Regular OLAP

Goal: Look for patterns of unusually high numbers of applications:

Z: Dimensions *Y*: Measure

Location	Time	*# of App.*
…	…	…
AL, USA	Dec, 04	2
…	…	…
WY, USA	Dec, 04	3





Example (2/7): Regular OLAP

Goal: Look for patterns of unusually high numbers of applications:

Z: Dimensions *Y*: Measure

Location	Time	*# of App.*
…	…	…
AL, USA	Dec, 04	2
…	…	…
WY, USA	Dec, 04	3

Coarser regions

	04	03	…
CA	100	90	…
USA	80	90	…
…	…	…	…

Roll up

	2004			2003			…
	Jan	…	Dec	Jan	…	Dec	…
CA	30	20	50	25	30	…	…
USA	70	2	8	10	…	…	…
…	…	…	…	…	…	…	…

Drill down

		2004			…
		Jan	…	Dec	…
CA	AB	20	15	15	…
	…	5	2	20	…
	YT	5	3	15	…
USA	AL	55	…	…	…
	…	5	…	…	
	WY	10	…	…	…
…	…	…	…	…	…

Cell value: Number of loan applications

Finer regions

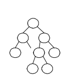
Example (3/7): Decision Analysis

Goal: Analyze a bank's loan decision process w.r.t. two dimensions: *Location* and *Time*

Fact table **D**

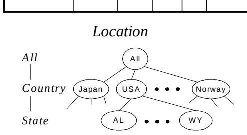
Z: Dimensions ***X***: Predictors *Y*: Class

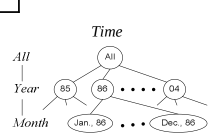
Location	Time	*Race*	*Sex*	*…*	*Approval*
AL, USA	Dec, 04	White	M	…	Yes
…	…	…	…	…	…
WY, USA	Dec, 04	Black	F	…	No



Cube subset

Model $h(\boldsymbol{X}, \sigma_Z(\mathbf{D}))$
E.g., decision tree

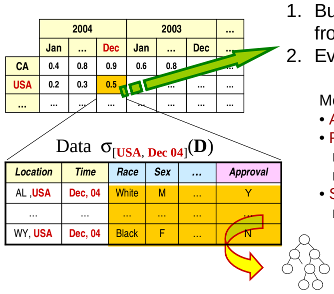




Example (3/7): Decision Analysis

- Are there branches (and time windows) where approvals were closely tied to sensitive attributes (e.g., race)?
  - Suppose you partitioned the training data by location and time, chose the partition for a given branch and time window, and built a classifier. You could then ask, "Are the predictions of this classifier closely correlated with race?"
- Are there branches and times with decision making reminiscent of 1950s Alabama?
  - Requires comparison of classifiers trained using different subsets of data.

Example (4/7): Prediction Cubes



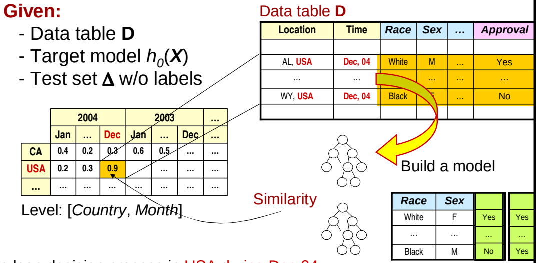
1. Build a model using data from USA in Dec., 1985
2. Evaluate that model

Measure in a cell:
- Accuracy of the model
- Predictiveness of *Race* measured based on that model
- Similarity between that model and a given model

Model $h(\boldsymbol{X}, \sigma_{[USA, Dec 04]}(\mathbf{D}))$
E.g., decision tree

Example (5/7): Model-Similarity

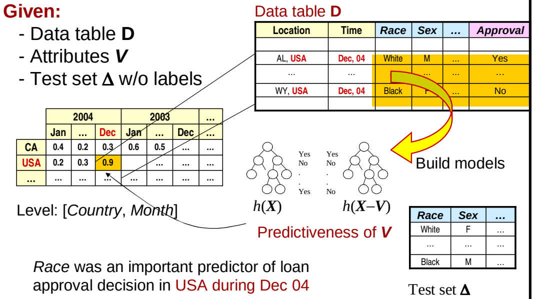
Given:
- Data table **D**
- Target model $h_0(\boldsymbol{X})$
- Test set **Δ** w/o labels



The loan decision process in USA during Dec 04 was similar to a discriminatory decision model

Example (6/7): Predictiveness

Given:
- Data table **D**
- Attributes ***V***
- Test set **Δ** w/o labels



Race was an important predictor of loan approval decision in USA during Dec 04

Model Accuracy

- A probabilistic view of classifiers: A dataset is a random sample from an underlying pdf $p^*(\boldsymbol{X}, Y)$, and a classifier
  $h(\boldsymbol{X}; \mathbf{D}) = \text{argmax}_y\, p^*(Y=y \mid \boldsymbol{X}=\mathbf{x}, \mathbf{D})$
  – i.e., A classifier approximates the pdf by predicting the "most likely" y value
- Model Accuracy:
  – $E_{x,y}[\, I(\, h(\mathbf{x}; \mathbf{D}) = y\,)\,]$, where $(\mathbf{x}, y)$ is drawn from $p^*(\boldsymbol{X}, Y \mid \mathbf{D})$, and $I(\Psi) = 1$ if the statement Ψ is true; $I(\Psi) = 0$, otherwise
  – In practice, since p* is an unknown distribution, we use a set-aside test set or cross-validation to estimate model accuracy.

Model Similarity

- The prediction similarity between two models, h1(**X**) and h2(**X**), on test set **Δ** is

$$\frac{1}{|\Delta|}\sum_{\mathbf{x}\in\Delta} I(h_1(\mathbf{x}) = h_2(\mathbf{x}))$$

- The KL-distance between two models, h1(**X**) and h2(**X**), on test set **Δ** is

$$\frac{1}{|\Delta|}\sum_{\mathbf{x}\in\Delta}\sum_{y} p_{h_1}(y \mid x)\log\frac{p_{h_1}(y \mid x)}{p_{h_2}(y \mid x)}$$

Attribute Predictiveness

- Intuition: $\boldsymbol{V} \subseteq \boldsymbol{X}$ is not predictive if and only if $\boldsymbol{V}$ is independent of Y given the other attributes $\boldsymbol{X} - \boldsymbol{V}$; i.e.,
  $p^*(Y \mid \boldsymbol{X} - \boldsymbol{V}, \mathbf{D}) = p^*(Y \mid \boldsymbol{X}, \mathbf{D})$
- In practice, we can use the distance between $h(\boldsymbol{X}; \mathbf{D})$ and $h(\boldsymbol{X} - \boldsymbol{V}; \mathbf{D})$
- Alternative approach: Test if $h(\boldsymbol{X}; \mathbf{D})$ is more accurate than $h(\boldsymbol{X} - \boldsymbol{V}; \mathbf{D})$ (e.g., by using cross-validation to estimate the two model accuracies involved)

Example (7/7): Prediction Cube

	2004			2003			...
	Jan	...	Dec	Jan	...	Dec	...
CA	0.4	0.1	0.3	0.6	0.8	...	...
USA	0.7	0.4	0.3	0.3	...	...	...
...	...	...	...	...	...	...	...

Roll up

	04	03	...
CA	0.3	0.2	...
USA	0.2	0.3	...
...	...	...	...

Cell value: Predictiveness of *Race*

Drill down

		2004			2003			...
		Jan	...	Dec	Jan	...	Dec	...
CA	AB	0.4	0.2	0.1	0.1	0.2	...	...
	...	0.1	0.1	0.3	0.3	...	...	...
	YT	0.3	0.2	0.1	0.2	...	...	...
USA	AL	0.2	0.1	0.2	...	...	...	...
	...	0.3	0.1	0.1	...	...	...	
	WY	0.9	0.7	0.8	...	...	...	...
...	...	...	...	...	...	...	...	...

Efficient Computation

- Reduce prediction cube computation to data cube computation
  – Represent a data-mining model as a distributive or algebraic (bottom-up computable) aggregate function, so that data-cube techniques can be directly applied

Bottom-Up Data Cube Computation

	1985	1986	1987	1988
All	47	107	76	67

	All
All	297

	1985	1986	1987	1988
Norway	10	30	20	24
...	23	45	14	32
USA	14	32	42	11

	All
Norway	84
...	114
USA	99

Cell Values: Numbers of loan applications

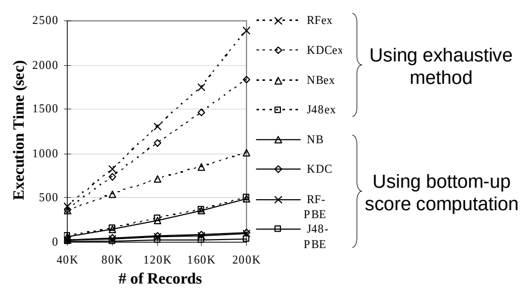
Scoring Function

- Represent a model as a function of sets
- Conceptually, a machine-learning model $h(\boldsymbol{X}; \sigma_Z(\mathbf{D}))$ is a scoring function $Score(y, \boldsymbol{x}; \sigma_Z(\mathbf{D}))$ that gives each class y a score on test example $\boldsymbol{x}$
  – $h(\boldsymbol{x}; \sigma_Z(\mathbf{D})) = \text{argmax}_y\, Score(y, \boldsymbol{x}; \sigma_Z(\mathbf{D}))$
  – $Score(y, \boldsymbol{x}; \sigma_Z(\mathbf{D})) \approx p(y \mid \boldsymbol{x}, \sigma_Z(\mathbf{D}))$
  – $\sigma_Z(\mathbf{D})$: The set of training examples (a cube subset of **D**)

Machine-Learning Models

- Naïve Bayes:
  – Scoring function: algebraic
- Kernel-density-based classifier:
  – Scoring function: distributive
- Decision tree, random forest:
  – Neither distributive, nor algebraic
- PBE: Probability-based ensemble (new)
  – To make any machine-learning model distributive
  – Approximation

Efficiency Comparison



Bellwether Analysis: Global Aggregates from Local Regions

with Beechun Chen, Jude Shavlik, and Pradeep Tamma
In VLDB 06

Motivating Example

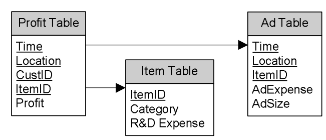
- A company wants to predict the first year worldwide profit of a new item (e.g., a new movie)
  - By looking at features and profits of previous (similar) movies, we predict expected total profit (1-year US sales) for new movie
    - Wait a year and write a query! If you can't wait, stay awake …
  - The most predictive "features" may be based on sales data gathered by releasing the new movie in many "regions" (different locations over different time periods).
    - Example "region-based" features: 1st week sales in Peoria, week-to-week sales growth in Wisconsin, etc.
    - Gathering this data has a cost (e.g., marketing expenses, waiting time)
- **Problem statement:** Find the most predictive region features that can be obtained within a given "cost budget"

Key Ideas

- Large datasets are rarely labeled with the targets that we wish to learn to predict
  - But for the tasks we address, we can readily use OLAP queries to generate features (e.g., 1st week sales in Peoria) and even **targets** (e.g., profit) for mining
- We use data-mining models as building blocks in the mining process, rather than thinking of them as the end result
  - The central problem is to find data subsets **("bellwether regions")** that lead to predictive features which can be gathered at low cost for a new case

Motivating Example

- A company wants to predict the first year's worldwide profit for a new item, by using its historical database
- Database Schema:

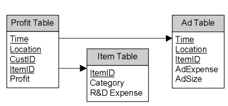


- The combination of the underlined attributes forms a key

A Straightforward Approach

- Build a regression model to predict item profit

By joining and aggregating tables in the historical database we can create a training set:



Item-table features | Target

ItemID	Category	R&D Expense	Profit
1	Laptop	500K	12,000K
2	Desktop	100K	8,000K
…	…	…	…

An Example regression model:
$Profit = \beta_0 + \beta_1\, Laptop + \beta_2\, Desktop + \beta_3\, RdExpense$

- There is much room for accuracy improvement!

Using Regional Features

- Example region: [1st week, HK]
- Regional features:
  - Regional Profit: The 1st week profit in HK
  - Regional Ad Expense: The 1st week ad expense in HK
- A possibly more accurate model:

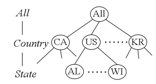
$$Profit_{[1yr,\ All]} = \beta_0 + \beta_1\, Laptop + \beta_2\, Desktop + \beta_3\, RdExpense + \beta_4\, \mathbf{Profit_{[1wk,\ KR]}} + \beta_5\, \mathbf{AdExpense_{[1wk,\ KR]}}$$

- Problem: Which region should we use?
  - The smallest region that improves the accuracy the most
  - We give each candidate region a cost
  - The most "cost-effective" region is the **bellwether region**

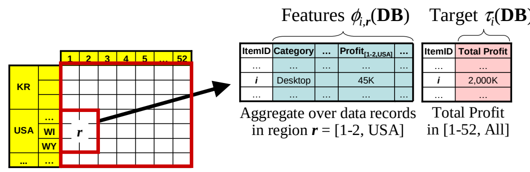
Basic Bellwether Problem

- Historical database: **DB**
- Training item set: **I**
- Candidate region set: **R**
  - E.g., { [1-*n* week, *Location*] }
- Target generation query: τ_i(**DB**) returns the target value of item $i \in$ **I**
  - E.g., $\alpha_{sum(Profit)}\ \sigma_{i,\ [1\text{-}52,\ All]}$ ProfitTable
- Feature generation query: $\phi_{i,r}$(**DB**), $i \in$ **I**$_r$ and $r \in$ **R**
  - **I**$_r$: The set of items in region *r*
  - E.g., [$Category_i$, $RdExpense_i$, $Profit_{i,\ [1\text{-}n,\ Loc]}$, $AdExpense_{i,\ [1\text{-}n,\ Loc]}$]
- Cost query: κ_r(**DB**), $r \in$ **R**, the cost of collecting data from *r*
- Predictive model: $h_r(x)$, $r \in$ **R**, trained on {($\phi_{i,r}$(**DB**), τ_i(**DB**)) : $i \in$ **I**$_r$}
  - E.g., linear regression model

Location domain hierarchy



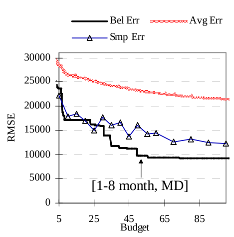
Basic Bellwether Problem



For each region **r**, build a predictive model $h_r(x)$; and then choose bellwether region:
- *Coverage*(**r**) ≡ fraction of all items in region ≥ minimum *coverage support*
- *Cost*(**r**, **DB**) ≤ *cost threshold*
- *Error*(h_r) is minimized

Experiment on a Mail Order Dataset

Error-vs-Budget Plot

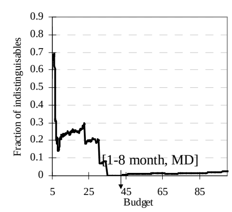


- **Bel Err:** The error of the bellwether region found using a given budget
- **Avg Err:** The average error of all the cube regions with costs under a given budget
- **Smp Err:** The error of a set of randomly sampled (non-cube) regions with costs under a given budget

(RMSE: Root Mean Square Error)

Experiment on a Mail Order Dataset

Uniqueness Plot

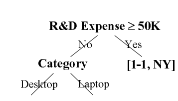


- ***Y*-axis:** Fraction of regions that are as good as the bellwether region
  - The fraction of regions that satisfy the constraints and have errors within the 99% confidence interval of the error of the bellwether region
- We have 99% confidence that that [1-8 month, MD] is a quite unusual bellwether region

Subset-Based Bellwether Prediction

- Motivation: Different subsets of items may have different bellwether regions
  - E.g., The bellwether region for laptops may be different from the bellwether region for clothes
- Two approaches:

Bellwether Tree



Bellwether Cube

R&D Expenses

Category		Low	Medium	High
Software	OS	[1-3,CA]	[1-1,NY]	[1-2,CA]
	...	...	...	...
Hardware	Laptop	[1-4,MD]	[1-1, NY]	[1-3,WI]
	...	...	...	...
...	...	...	...	...

Conclusions

Related Work: Building models on OLAP Results

- Multi-dimensional regression [Chen, VLDB 02]
  - Goal: Detect changes of trends
  - Build linear regression models for cube cells
- Step-by-step regression in stream cubes [Liu, PAKDD 03]
- Loglinear-based quasi cubes [Barbara, J. IIS 01]
  - Use loglinear model to approximately compress dense regions of a data cube
- NetCube [Margaritis, VLDB 01]
  - Build Bayes Net on the entire dataset of approximate answer count queries

Related Work (Contd.)

- Cubegrades [Imielinski, J. DMKD 02]
  - Extend cubes with ideas from association rules
  - How does the measure change when we rollup or drill down?
- Constrained gradients [Dong, VLDB 01]
  - Find pairs of similar cell characteristics associated with big changes in measure
- User-cognizant multidimensional analysis [Sarawagi, VLDBJ 01]
  - Help users find the most informative unvisited regions in a data cube using max entropy principle
- Multi-Structural DBs [Fagin et al., PODS 05, VLDB 05]

Take-Home Messages

- Promising exploratory data analysis paradigm:
  - Can use models to identify interesting subsets
  - Concentrate only on subsets in cube space
    - Those are meaningful subsets, tractable
  - Precompute results and provide the users with an interactive tool
- A simple way to plug "something" into cube-style analysis:
  - Try to describe/approximate "something" by a distributive or algebraic function

Big Picture

- Why stop with decision behavior? Can apply to other kinds of analyses too
- Why stop at browsing? Can mine prediction cubes in their own right
- Exploratory analysis of mining space:
  - Dimension attributes can be parameters related to algorithm, data conditioning, etc.
  - Tractable evaluation is a challenge:
    - Large number of "dimensions", real-valued dimension attributes, difficulties in compositional evaluation
    - Active learning for experiment design, extending compositional methods