



Chapter 5: Concept Description: Characterization and Comparison

- What is concept description?
- Data generalization and summarization-based characterization
- Analytical characterization: Analysis of attribute relevance
- Mining class comparisons: Discriminating between different classes
- Summary



Concept Description

- A concept refers to a collection of data such as frequent_buyers, graduate_students etc.
- Concept description is close to data generalization.
- Example:
 - Instead of examining individual customer transactions, sales manager may prefer to view the data generalized to higher levels such as summarized by customer groups according to geographic regions, frequency of purchases per group and customer income.

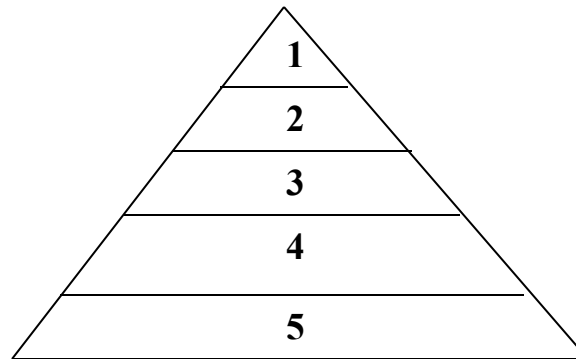


What is Concept Description?

- Descriptive vs. predictive data mining
 - **Descriptive mining**: describes concepts or task-relevant data sets in concise, summative, informative, discriminative forms
 - **Predictive mining**: Based on data and analysis, constructs models for the database, and predicts the trend and properties of unknown data
- Concept description:
 - **Characterization**: provides a concise, brief and clear summarization of the given collection of data
 - **Comparison**: provides descriptions comparing two or more collections of data

Data Generalization and Summarization-based Characterization

- Data generalization
 - A process which abstracts a large set of task-relevant data in a database from a low conceptual levels to higher ones.



Conceptual levels

- Approaches:
 - Data cube approach(OLAP approach)
 - Attribute-oriented induction approach



Characterization: Data Cube Approach (without using AO-Induction)

- Perform computations and store results in data cubes
- Strength
 - An efficient implementation of data generalization
 - Computation of various kinds of measures
 - e.g., count(), sum(), average(), max()
 - Generalization and specialization can be performed on a data cube by *roll-up* and *drill-down*
- Limitations
 - handle only dimensions of *simple nonnumeric data* and measures of *simple aggregated numeric values*.
 - Lack of intelligent analysis, can't tell which dimensions should be used and what levels should the generalization reach



Attribute-Oriented Induction

- Proposed in 1989 (KDD '89 workshop)
- Not confined to categorical data nor particular measures.
- How it is done?
 - Collect the task-relevant data(*initial relation*) using a relational database query
 - Perform generalization by attribute removal or attribute generalization.
 - Apply aggregation by merging identical, generalized tuples and accumulating their respective counts.
 - Interactive presentation with users.



Attribute-Oriented Induction

- Attribute Removal:

If there is a large set of distinct values for an attribute of the initial working relation, but either (1) there is no generalization operator on the attribute (no concept hierarchy) or (2) its higher-level concepts are expressed in terms of other attributes then the attribute should be removed.

- Attribute Generalization:

If there is a large set of distinct values for an attribute in initial working relation and there exists a set of generalization operators on the attribute then a generalization operator should be selected and applied to the attribute.



Attribute-Oriented Approach: Attribute Generalization

- Attribute Generalization Threshold Control
 - Either sets one generalization threshold for all of the attributes or sets one threshold for each attribute. If the number of distinct values in an attribute is greater than the attribute threshold, attribute removal or attribute generalization should be performed.
- Generalized Relation Threshold Control
 - Sets a threshold for the generalized relation. If the number of tuples in the generalized relation is greater than the threshold, further generalization should be performed.

Basic Principles of Attribute-Oriented Induction

- Data focusing: task-relevant data, including dimensions, and the result is the *initial relation*.
- **A data mining query for characterization.** Suppose that a user would like to describe the general characteristics of graduate students in the **Big University database**, given the attributes *name*, *gender*, *major*, *birth place*, *birth date*, *residence*, *phone#* (telephonenumber), and *gpa* (grade point average).

<i>name</i>	<i>gender</i>	<i>major</i>	<i>birth_place</i>	<i>birth_date</i>	<i>residence</i>	<i>phone#</i>	<i>gpa</i>
Jim Woodman	M	CS	Vancouver, BC, Canada	8-12-76	3511 Main St., Richmond	687-4598	3.67
Scott Lachance	M	CS	Montreal, Que, Canada	28-7-75	345 1st Ave., Richmond	253-9106	3.70
Laura Lee	F	physics	Seattle, WA, USA	25-8-70	125 Austin Ave., Burnaby	420-5232	3.83
...

Initial working relation: a collection of task-relevant data



Basic Principles of Attribute-Oriented Induction

- Attribute-removal:
- Remove attribute A if there is a large set of distinct values for A but either
 1. There is no generalization operator on A (*e.g. there is no concept hierarchy*, or
 2. A 's higher level concepts are expressed in terms of other attributes.
- E.g: attribute **street**, {city, province_or state, country} any of the other attribute is present in initial working relation then we can drop **street** attribute.



Basic Principles of Attribute-Oriented Induction

- Attribute-generalization: If there is a large set of distinct values for A , and there exists a set of generalization operators on A , then select an operator and generalize A .
- Attribute-generalization control
 - If there is a large set of distinct values=> “how high an attribute should be generalized?”
 - Attribute-threshold control:
 - typical 2-8, specified/default.
 - Set one threshold for all attribute
 - Set threshold for individual attribute
 - Generalized relation threshold control: control the final relation/rule size.



Basic Algorithm for Attribute-Oriented Induction

- InitialRel: Query processing of task-relevant data, deriving the *initial relation*.
- PreGen: Based on the analysis of the number of distinct values in each attribute, determine generalization plan for each attribute: removal? or how high to generalize?
- PrimeGen: Based on the PreGen plan, perform generalization to the right level to derive a “prime generalized relation”, accumulating the counts.
- Presentation: User interaction: (1) adjust levels by drilling, (2) pivoting, (3) mapping into rules, cross tabs, visualization presentations.



Input:

- *DB*, a relational database;
- *DMQuery*, a data mining query;
- *a_list*, a list of attributes (containing attributes, a_i);
- *Gen*(a_i), a set of concept hierarchies or generalization operators on attributes, a_i ;
- *a_gen_thresh*(a_i), attribute generalization thresholds for each a_i .

Output: *P*, a *Prime_generalized_relation*.

Method:

1. $W \leftarrow \text{get_task_relevant_data} (DMQuery, DB)$; // Let *W*, the working relation, hold the task-relevant data.
2. $\text{prepare_for_generalization} (W)$; // This is implemented as follows.
 - (a) Scan *W* and collect the distinct values for each attribute, a_i . (Note: If *W* is very large, this may be done by examining a sample of *W*.)
 - (b) For each attribute a_i , determine whether a_i should be removed. If not, compute its minimum desired level L_i based on its given or default attribute threshold, and determine the mapping pairs (v, v') , where v is a distinct value of a_i in *W*, and v' is its corresponding generalized value at level L_i .

3. $P \leftarrow \text{generalization} (W)$,

The *Prime_generalized_relation*, *P*, is derived by replacing each value v in *W* by its corresponding v' in the mapping while accumulating `count` and computing any other aggregate values.

This step can be implemented efficiently using either of the two following variations:

- (a) For each generalized tuple, insert the tuple into a sorted prime relation *P* by a binary search: if the tuple is already in *P*, simply increase its `count` and other aggregate values accordingly; otherwise, insert it into *P*.
- (b) Since in most cases the number of distinct values at the prime relation level is small, the prime relation can be coded as an m -dimensional array, where m is the number of attributes in *P*, and each dimension contains the corresponding generalized attribute values. Each array element holds the corresponding `count` and other aggregation values, if any. The insertion of a generalized tuple is performed by measure aggregation in the corresponding array element.



Example

- **DMQL:** Describe general characteristics of graduate students in the Big-University database

```
use Big_University_DB
```

```
mine characteristics as "Science_Students"  
in relevance to name, gender, major, birth_place,  
birth_date, residence, phone#, gpa
```

```
from student
```

```
where status in "graduate"
```

- **Corresponding SQL statement:**

```
Select name, gender, major, birth_place, birth_date,  
residence, phone#, gpa
```

```
from student
```

```
where status in {"Msc", "MBA", "PhD" }
```

Class Characterization: An Example

**Initial
Relation**

Name	Gender	Major	Birth-Place	Birth_date	Residence	Phone #	GPA
Jim Woodman	M	CS	Vancouver,BC, Canada	8-12-76	3511 Main St., Richmond	687-4598	3.67
Scott Lachance	M	CS	Montreal, Que, Canada	28-7-75	345 1st Ave., Richmond	253-9106	3.70
Laura Lee	F	Physics	Seattle, WA, USA	25-8-70	125 Austin Ave., Burnaby	420-5232	3.83
...
Removed	Retained	Sci,Eng, Business	Country	Age range	City	Removed	Excl, VG,...

**Prime
Generalized
Relation**

Gender	Major	Birth region	Age range	Residence	GPA	Count
M	Science	Canada	20-25	Richmond	Very-good	16
F	Science	Foreign	25-30	Burnaby	Excellent	22
...

Birth_Region Gender	Canada	Foreign	Total
M	16	14	30
F	10	22	32
Total	26	36	62



Attribute Relevance Analysis

- The general idea behind attribute relevance analysis is to compute some measure which is used to quantify the relevance of an attribute with respect to given class or concept.



Attribute Relevance Analysis : Attribute Selection

- Attribute selection is a term commonly used in data mining to describe the tools and techniques available for reducing inputs to a manageable size for processing and analysis.
- Attribute selection implies not only cardinality reduction but also the choice of attributes based on their usefulness for analysis.
- Find a subset of attributes that is most likely to describe / predict the class best. Filter type methods suppress the least interesting variables. These methods are effective in computation time and robust to over fitting.



Attribute Relevance Analysis

- Why?
 - Which dimensions should be included?
 - How high level of generalization?
 - Automatic vs. interactive
 - Reduce # attributes; easy to understand patterns
- What?
 - statistical method for preprocessing data
 - filter out irrelevant or weakly relevant attributes
 - retain or rank the relevant attributes
 - relevance related to dimensions and levels
 - analytical characterization, analytical comparison



Attribute relevance analysis (cont'd)

- How?
 - Data Collection
 - Target class and contrasting class
 - Analytical Generalization
 - Use information gain analysis (e.g., entropy or other measures) to identify highly relevant dimensions and levels.
 - Relevance Analysis
 - Sort and select the most relevant dimensions and levels.
 - Attribute-oriented Induction for class description
 - On selected dimension/level
 - OLAP operations (e.g. drilling, slicing) on relevance rules



Relevance Measures

- Quantitative relevance measure determines the classifying power of an attribute within a set of data.
- Methods
 - information gain (ID3)
 - gain ratio (C4.5)
 - gini index
 - χ^2 contingency table statistics
 - uncertainty coefficient



Information-Theoretic Approach

- Decision tree
 - each internal node tests an attribute
 - each branch corresponds to attribute value
 - each leaf node assigns a classification
- ID3 algorithm
 - build decision tree based on training objects with known class labels to classify testing objects
 - rank attributes with information gain measure
 - minimal height
 - the least number of tests to classify an object



Decision Tree

Play or not?

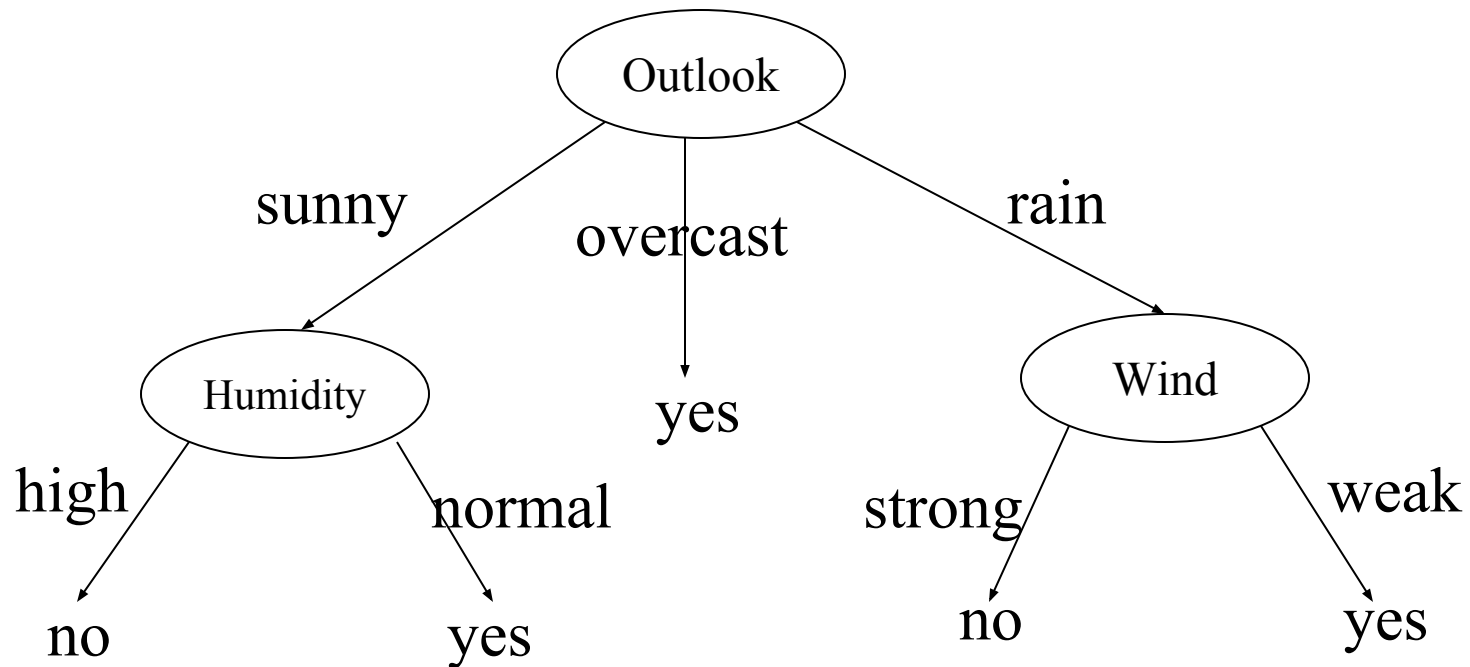
- The weather dataset

OUTLOOK	TEMP	HUMIDITY	WINDY	PLAY
-----	----	-----	-----	----
Sunny	Hot	High	False	No
Sunny	Hot	High	True	No
Overcast	Hot	High	False	Yes
Rainy	Mild	High	False	Yes
Rainy	Cool	Normal	False	Yes
Rainy	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Sunny	Mild	High	False	No
Sunny	Cool	Normal	False	Yes
Rainy	Mild	Normal	False	Yes
Sunny	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Rainy	Mild	High	True	No

Top-Down Induction of Decision Tree

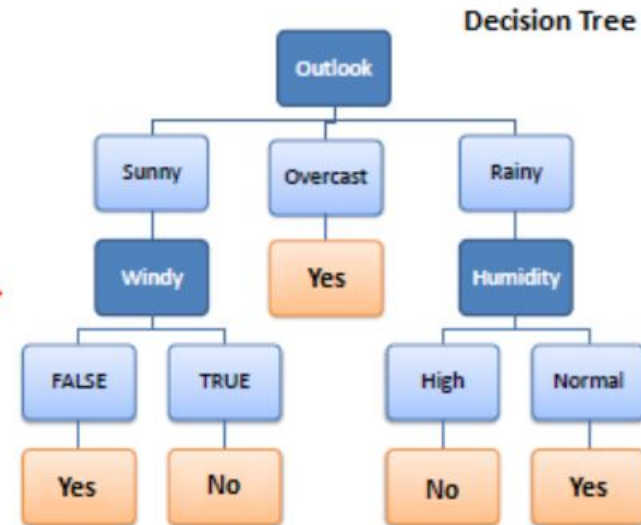
Attributes = {Outlook, Temperature, Humidity, Wind}

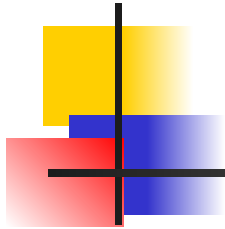
PlayTennis = {yes, no}



Decision Tree

Predictors				Target
Outlook	Temp.	Humidity	Windy	Play Golf
Rainy	Hot	High	False	No
Rainy	Hot	High	True	No
Overcast	Hot	High	False	Yes
Sunny	Mild	High	False	Yes
Sunny	Cool	Normal	False	Yes
Sunny	Cool	Normal	True	No
Overcast	Cool	Normal	True	Yes
Rainy	Mild	High	False	No
Rainy	Cool	Normal	False	Yes
Sunny	Mild	Normal	False	Yes
Rainy	Mild	Normal	True	Yes
Overcast	Mild	High	True	Yes
Overcast	Hot	Normal	False	Yes
Sunny	Mild	High	True	No





In decision tree, each node focuses on identifying an attribute and a split condition on that attribute which minimizing the class labels mixing, consequently giving relatively pure subsets.

In order to check “the goodness of splitting criterion” or for evaluating how well the splitting is, various splitting indices were proposed. Some of them are gini index and information gain.



Expected Information, Entropy and Information Gain

- S contains s_i samples of class C_i for $i = \{1, \dots, m\}$
- Information measures info required to classify any arbitrary tuple (**Expected Information**)

$$I(s_1, s_2, \dots, s_m) = - \sum_{i=1}^m \frac{s_i}{S} \log_2 \frac{s_i}{S}$$

- **Entropy** of attribute A with values $\{a_1, a_2, \dots, a_v\}$

$$E(A) = \sum_{j=1}^v \frac{s_{1j} + \dots + s_{mj}}{S} I(s_{1j}, \dots, s_{mj})$$

- **Information gained** by branching on attribute A

$$\text{Gain}(A) = I(s_1, s_2, \dots, s_m) - E(A)$$

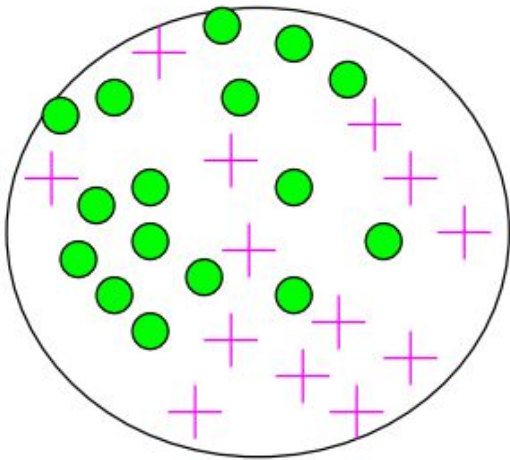


Expected Information, Entropy and Information Gain

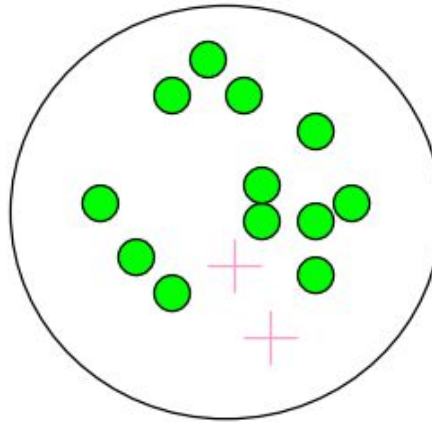
- Entropy
 - The entropy (very common in Information Theory) characterizes the (im)purity of an arbitrary collection of examples.
- Information Gain
 - Information Gain is the expected reduction in entropy caused by partitioning the examples according to a given attribute

Entropy

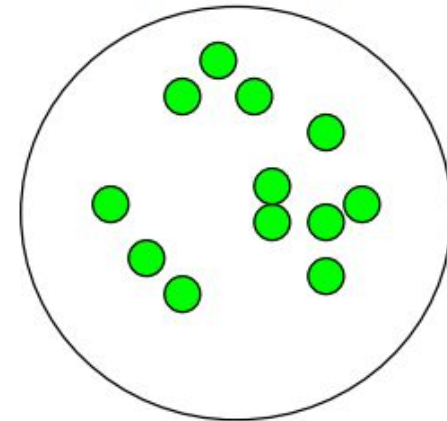
Very impure group



Less impure



Minimum impurity



Entropy – Measures the level of impurity in a group of examples

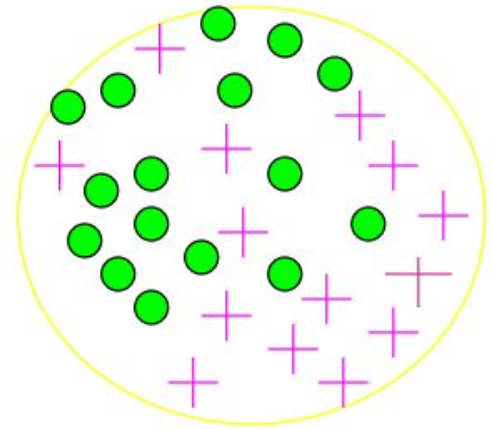
If all elements belong to a single class, then it is termed as “Pure”, and if not then the distribution is named as “Impurity”

Entropy

- Entropy =
$$\sum_i -p_i \log_2 p_i$$

p_i is the probability of class i

Compute it as the proportion of class i in the set.



16/30 are green circles; 14/30 are pink crosses

$\log_2(16/30) = -.9$; $\log_2(14/30) = -1.1$

Entropy = $-(16/30)(-.9) - (14/30)(-1.1) = .99$

Entropy : 2 Class Case

Entropy comes from information theory. The higher the entropy the more the information content.

- What is the entropy of a group in which all examples belong to the same class?

- $\text{entropy} = -1 \log_2 1 = 0$

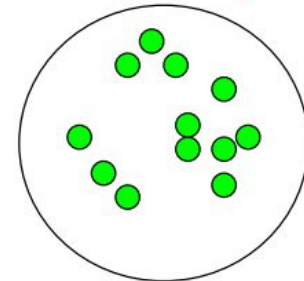
not a good training set for learning

- What is the entropy of a group with 50% in either class?

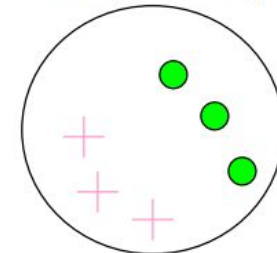
- $\text{entropy} = -0.5 \log_2 0.5 - 0.5 \log_2 0.5 = 1$

good training set for learning

**Minimum
impurity**



**Maximum
impurity**





Information Gain

- We want to determine **which attribute** in a given set of training feature vectors is **most useful** for discriminating between the classes to be learned.
- **Information gain** tells us how important a given attribute of the feature vectors is.
- We will use it to decide the ordering of attributes in the nodes of a decision tree.

Information gain computes the difference between entropy before and after split and specifies the impurity in class elements.



Example

Training Set: 3 features and 2 classes

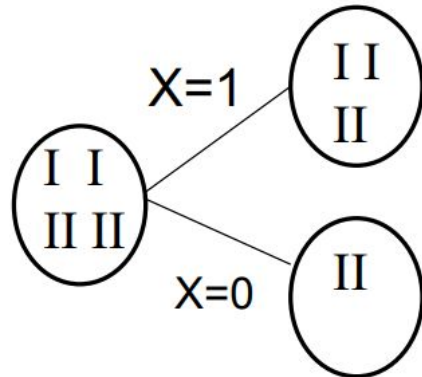
X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

How would you distinguish class I from class II?

Example

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Split on attribute X



If X is the best attribute,
this node would be further split.

$$\begin{aligned} E_{\text{child1}} &= -(1/3)\log_2(1/3) - (2/3)\log_2(2/3) \\ &= .5284 + .39 \\ &= .9184 \end{aligned}$$

$$E_{\text{child2}} = 0$$

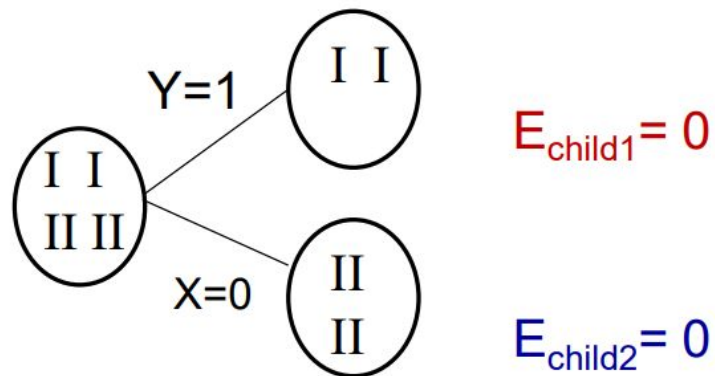
$$E_{\text{parent}} = 1$$

$$\text{GAIN} = 1 - (3/4)(.9184) - (1/4)(0) = .3112$$

Example

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Split on attribute Y



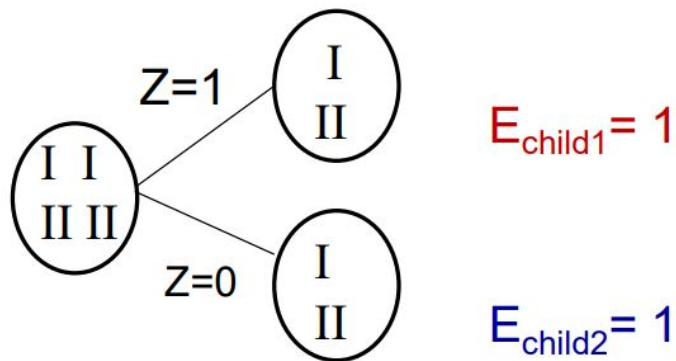
$$E_{\text{parent}} = 1$$

$$\text{GAIN} = 1 - (1/2)0 - (1/2)0 = 1; \text{ BEST ONE}$$

Example

X	Y	Z	C
1	1	1	I
1	1	0	I
0	0	1	II
1	0	0	II

Split on attribute Z



$$E_{\text{parent}} = 1$$

$$\text{GAIN} = 1 - (1/2)(1) - (1/2)(1) = 0 \quad \text{ie. NO GAIN; WORST}$$



Calculating Entropy of Dataset

Looking at some data

<u>Color</u>	<u>Size</u>	<u>Shape</u>	<u>Edible?</u>
Yellow	Small	Round	+
Yellow	Small	Round	-
Green	Small	Irregular	+
Green	Large	Irregular	-
Yellow	Large	Round	+
Yellow	Small	Round	+
Yellow	Small	Round	+
Yellow	Small	Round	+
Green	Small	Round	-
Yellow	Large	Round	-
Yellow	Large	Round	+
Yellow	Large	Round	-
Yellow	Large	Round	-
Yellow	Large	Round	-
Yellow	Small	Irregular	+
Yellow	Large	Irregular	+



Calculating Entropy of Dataset

Entropy for our data set

- 16 instances: 9 positive, 7 negative.


$$I(all_data) = -\left[\left(\frac{9}{16}\right)\log_2\left(\frac{9}{16}\right) + \left(\frac{7}{16}\right)\log_2\left(\frac{7}{16}\right)\right]$$

- This equals: 0.9836
- This makes sense - it's almost a 50/50 split; so, the entropy should be close to 1.



Example: Analytical Characterization

- Task
 - Mine general characteristics describing graduate students using analytical characterization
- Given
 - attributes *name, gender, major, birth_place, birth_date, phone#, and gpa*
 - $Gen(a_i)$ = concept hierarchies on a_i
 - U_i = attribute analytical thresholds for a_i
 - T_i = attribute generalization thresholds for a_i
 - R = attribute relevance threshold = 0.1



Example: Analytical Characterization (cont'd)

- 1. Data collection
 - target class: graduate student
 - contrasting class: undergraduate student
- 2. Analytical generalization using U_i
 - attribute removal
 - remove *name* and *phone#*
 - attribute generalization
 - generalize *major*, *birth_place*, *birth_date* and *gpa*
 - accumulate counts
 - **candidate relation**: *gender*, *major*, *birth_country*, *age_range* and *gpa*

Example: Analytical characterization (2)

gender	major	birth country	age range	gpa	count
M	Science	Canada	20-25	Very_good	16
F	Science	Foreign	25-30	Excellent	22
M	Engineering	Foreign	25-30	Excellent	18
F	Science	Foreign	25-30	Excellent	25
M	Science	Canada	20-25	Excellent	21
F	Engineering	Canada	20-25	Excellent	18

Candidate relation for Target class: Graduate students ($\Sigma=120$)

gender	major	birth country	age range	gpa	count
M	Science	Foreign	<20	Very_good	18
F	Business	Canada	<20	Fair	20
M	Business	Canada	<20	Fair	22
F	Science	Canada	20-25	Fair	24
M	Engineering	Foreign	20-25	Very_good	22
F	Engineering	Canada	<20	Excellent	24

Candidate relation for Contrasting class: Undergraduate students ($\Sigma=130$)

Example: Analytical characterization (3)

- 3. Relevance analysis
 - Calculate expected info required to classify an arbitrary tuple

$$I(s_1, s_2) = I(120, 130) = -\frac{120}{250} \log_2 \frac{120}{250} - \frac{130}{250} \log_2 \frac{130}{250} = 0.9988$$

- Calculate entropy of each attribute: e.g. **major**

For *major*="Science": $s_{11}=84$ $s_{21}=42$ $I(s_{11}, s_{21})=0.9183$

For *major*="Engineering": $s_{12}=36$ $s_{22}=46$ $I(s_{12}, s_{22})=0.9892$

For *major*="Business": $s_{13}=0$ $s_{23}=42$ $I(s_{13}, s_{23})=0$

Number of grad
students in "Science"

Number of undergrad
students in "Science"



Example: Analytical Characterization (4)

- Calculate expected info required to classify a given sample if S is partitioned according to the attribute

$$E(major) = \frac{126}{250} I(s_{11}, s_{21}) + \frac{82}{250} I(s_{12}, s_{22}) + \frac{42}{250} I(s_{13}, s_{23}) = 0.7873$$

- Calculate information gain for each attribute

$$Gain(major) = I(s_1, s_2) - E(major) = 0.2115$$

- Information gain for all attributes

$$Gain(\text{gender}) = 0.0003$$

$$Gain(\text{birth_country}) = 0.0407$$

$$Gain(\text{major}) = 0.2115$$

$$Gain(\text{gpa}) = 0.4490$$

$$Gain(\text{age_range}) = 0.5971$$



Example: Analytical characterization (5)

- 4. Initial working relation (W_0) derivation
 - (attribute relevance threshold) $R = 0.1$
 - remove irrelevant/weakly relevant attributes from candidate relation \Rightarrow drop *gender*, *birth_country*
 - remove contrasting class candidate relation

major	age_range	gpa	count
Science	20-25	Very_good	16
Science	25-30	Excellent	47
Science	20-25	Excellent	21
Engineering	20-25	Excellent	18
Engineering	25-30	Excellent	18

Initial target class working relation W_0 : Graduate students

- 5. Perform attribute-oriented induction on W_0 (using attribute generalization thresholds) T_i



Example

<i>RID</i>	<i>age</i>	<i>income</i>	<i>student</i>	<i>credit_rating</i>	<i>Class: buys_computer</i>
1	youth	high	no	fair	no
2	youth	high	no	excellent	no
3	middle_aged	high	no	fair	yes
4	senior	medium	no	fair	yes
5	senior	low	yes	fair	yes
6	senior	low	yes	excellent	no
7	middle_aged	low	yes	excellent	yes
8	youth	medium	no	fair	no
9	youth	low	yes	fair	yes
10	senior	medium	yes	fair	yes
11	youth	medium	yes	excellent	yes
12	middle_aged	medium	no	excellent	yes
13	middle_aged	high	yes	fair	yes
14	senior	medium	no	excellent	no



Example

- The class label attribute, buys computer, has two distinct values (namely, {yes, no}); therefore, there are two distinct classes (i.e., $m = 2$).
- There are nine tuples of class yes and five tuples of class no.
- A (root) node N is created for the tuples in D. To find the splitting criterion for these tuples, we must compute the information gain of each attribute.
- $\text{Info}(D) = - 9/14 \log_2 9/14 - 5/14 \log_2 5 / 14 = 0.940$



Example

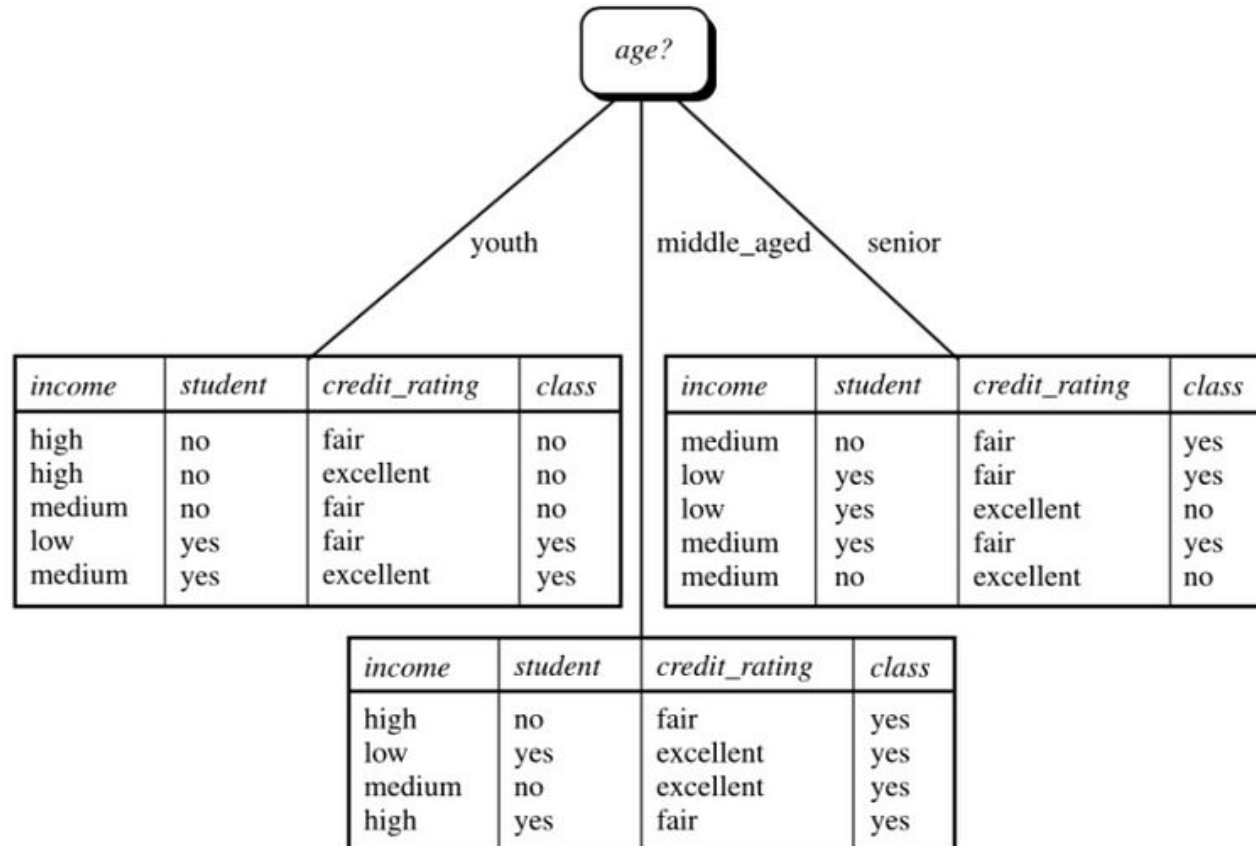
- Next, we need to compute the expected information requirement for each attribute. Let's start with the attribute age.
- For the age category "youth," there are two yes tuples and three no tuples. For the category "middle aged," there are four yes tuples and zero no tuples. For the category "senior," there are three yes tuples and two no tuples.
- The expected information needed to classify a tuple in D if the tuples are partitioned according to age is
- $\text{Info age}(D) = \frac{5}{14} \times (-\frac{2}{5} \log_2 (\frac{2}{5}) - \frac{3}{5} \log_2 (\frac{3}{5})) + \frac{4}{14} \times (-\frac{4}{4} \log_2 (\frac{4}{4})) + \frac{5}{14} \times (-\frac{3}{5} \log_2 (\frac{3}{5}) - \frac{2}{5} \log_2 (\frac{2}{5}))$
- Information gain of age attribute is: $\text{Gain}(\text{age}) = \text{Info}(D) - \text{Info age}(D) = 0.940 - 0.694 = 0.246$



Example

- $\text{Gain}(\text{age}) = 0.246$
- $\text{Gain}(\text{student}) = 0.151$
- $\text{Gain}(\text{credit rating}) = 0.048$
- $\text{Gain}(\text{income}) = 0.029$
- Because age has the highest information gain among the attributes, it is selected as the splitting attribute.

Example



The tuples falling into the partition for age = middle aged all belong to the same class. Because they all belong to class “yes,” a leaf should therefore be created at the end of this branch and labeled “yes.”



Mining Class Comparisons

- Comparison: Comparing two or more classes.
- Method:
 - Partition the set of relevant data into the **target class** and the **contrasting class(es)**
 - Generalize both classes to the same high level concepts
 - Compare tuples with the same high level descriptions
 - Present for every tuple its description and two measures:
 - support - distribution within single class
 - comparison - distribution between classes
 - Highlight the tuples with strong discriminant features
- Relevance Analysis:
 - Find attributes (features) which best distinguish different classes.



Example: Analytical comparison

- Task
 - Compare **graduate** and **undergraduate students** using **discriminant rule**.
 - DMQL query

use Big_University_DB
mine comparison as “grad_vs_undergrad_students”
in relevance to *name, gender, major, birth_place, birth_date, residence, phone#, gpa*
for “graduate_students”
where status in “**graduate**”
versus “undergraduate_students”
where status in “**undergraduate**”
analyze count%
from student



Example: Analytical comparison (2)

- Given
 - attributes *name, gender, major, birth_place, birth_date, residence, phone#* and *gpa*
 - **$Gen(a_i)$** = concept hierarchies on attributes a_i
 - **U_i** = attribute analytical thresholds for attributes a_i
 - **T_i** = attribute generalization thresholds for attributes a_i
 - **R** = attribute relevance threshold



Example: Analytical comparison (3)

- 1. Data collection
 - target and contrasting classes
- 2. Attribute relevance analysis
 - remove attributes *name*, *gender*, *major*, *phone#*
- 3. Synchronous generalization
 - controlled by user-specified dimension thresholds
 - prime target and contrasting class(es) relations

Example: Analytical comparison (4)

Prime generalized relation for the target class:

Graduate students

Birth_country	Age_range	Gpa	Count%
Canada	20-25	Good	5.53%
Canada	25-30	Good	2.32%
Canada	Over_30	Very_good	5.86%
...
Other	Over_30	Excellent	4.68%

Prime generalized relation for the contrasting class: Undergraduate students

Birth_country	Age_range	Gpa	Count%
Canada	15-20	Fair	5.53%
Canada	15-20	Good	4.53%
...
Canada	25-30	Good	5.02%
...
Other	Over_30	Excellent	0.68%



Example: Analytical comparison (5)

- 4. Drill down, roll up and other OLAP operations on target and contrasting classes to adjust levels of abstractions of resulting comparison
- 5. Presentation
 - as generalized relations, crosstabs, bar charts, pie charts, or rules
 - contrasting measures to reflect comparison between target and contrasting classes
 - e.g. **count%**



Presentation of Generalized Results using Generalized relation

- Relations where some or all attributes are generalized, with counts or other aggregation values accumulated.

After applying AOI on sales data the generalized relation
for the sales in 1999

<i>location</i>	<i>item</i>	<i>sales (in million dollars)</i>	<i>count (in thousands)</i>
Asia	TV	15	300
Europe	TV	12	250
North_America	TV	28	450
Asia	computer	120	1000
Europe	computer	150	1200
North_America	computer	200	1800

Presentation of Generalized Results using Cross tabulation

- Mapping results into cross tabulation form (similar to contingency tables).
- Pie charts, bar charts, curves, cubes, and other visual forms.

location	<i>item</i>					
	<i>TV</i>		<i>computer</i>		<i>both_items</i>	
	<i>sales</i>	<i>count</i>	<i>sales</i>	<i>count</i>	<i>sales</i>	<i>count</i>
Asia	15	300	120	1000	135	1300
Europe	12	250	150	1200	162	1450
North_America	28	450	200	1800	228	2250
<i>all_regions</i>	45	1000	470	4000	525	5000



Presentation of Generalized Results using **Quantitative Rule**

- Quantitative characteristic rule
 - t_weight
- Quantitative discriminant rule
 - d_weight



Quantitative characteristic rule

- Used to define/present quantitative characteristic.

$$t_weight = \frac{count(q_a)}{\sum_{i=1}^n count(q_i)}$$

- q_a is generalized tuple describing **target class**
 - n is total number of tuples
 - q_i is count of tuples for all distinct values/categories
- range: $[0, 100]\%$
- $\forall X, target_class(X) \Rightarrow condition_1(X) [t:w_1] \vee \dots condition_m(x) [t:w_m]$

Quantitative characteristic rule

$$t_weight = \frac{count(q_a)}{\sum_{i=1}^n count(q_i)}$$

	item					
	TV		computer		both_items	
location	sales	count	sales	count	sales	count
Asia	15	300	120	1000	135	1300
Europe	12	250	150	1200	162	1450
North_America	28	450	200	1800	228	2250
all_regions	45	1000	470	4000	525	5000

- $\forall X, \text{item}(X) = \text{"computer"} \Rightarrow$
 - $(\text{location}(X) = \text{"Asia"}) [t:25\%] V$
 - $(\text{location}(X) = \text{"Europe"}) [t:30\%] V$
 - $(\text{location}(X) = \text{"North_America"}) [t:45\%]$



Quantitative Discriminant Rules

- C_j = target class
- q_a = a generalized tuple covers some tuples of class
 - but can also cover some tuples of contrasting class
- d-weight
 - range: $[0, 100]\%$

$$d\text{-weight} = \frac{\text{count}(q_a \in C_j)}{\sum_{i=1}^m \text{count}(q_a \in C_i)}$$

- quantitative discriminant rule form

$$\forall X, \text{target_class}(X) \Leftarrow \text{condition}(X) \quad [d : d_weight]$$

Example: Quantitative Discriminant Rule

Status	Birth country	Age range	Gpa	Count
Graduate	Canada	25-30	Good	90
Undergraduate	Canada	25-30	Good	210

Count distribution between graduate and undergraduate students for a generalized tuple

- Quantitative discriminant rule

$\forall X, \text{graduate_student}(X) \Leftarrow$

$\text{birth_country}(X) = \text{"Canada"} \wedge \text{age_range}(X) = \text{"25-30"} \wedge \text{gpa}(X) = \text{"good"} \quad [d : 30\%]$

- where $90/(90+210) = 30\%$



Can we combine both?

- Quantitative characteristic rule

$\forall X, \text{target_class}(X) \Rightarrow \text{condition}(X) \quad [t : t_weight]$

- necessary

- Quantitative discriminant rule

$\forall X, \text{target_class}(X) \Leftarrow \text{condition}(X) \quad [d : d_weight]$

- sufficient

- Quantitative description rule

$\forall X, \text{target_class}(X) \Leftrightarrow$

$\text{condition}_1(X) [t : w_1, d : w'_1] \vee \dots \vee \text{condition}_n(X) [t : w_n, d : w'_n]$

- necessary and sufficient

Example: Quantitative Description Rule

Item->	TV			Computer			Both_items		
Location	Count	t-wt	d-wt	Count	t-wt	d-wt	Count	t-wt	d-wt
Europe	80	25%	40%	240	75%	30%	320	100%	32%
North_America	120	17.65%	60%	560	82.35%	70%	680	100%	68%
Both_region	200	20%	100%	800	80%	100%	1000	100%	100%

Crosstab showing associated **t-weight**, **d-weight** values and total number (in thousands) of **TVs** and **computers** sold at AllElectronics in **1998**

- Quantitative description rule for target class ***Europe***

$\forall X, location(X) = "Europe" \Leftrightarrow$

$(item(X) = "TV") [t : 25\%, d : 40\%] \vee (item(X) = "computer") [t : 75\%, d : 30\%]$



Summary

- Concept description: **characterization** and **discrimination**
- OLAP-based vs. attribute-oriented induction
- Efficient implementation of AOI
- Analytical characterization and comparison



<http://www.cs.sfu.ca/~han/dmbook>



Thank you !!!



Concept Description vs. OLAP

- OLAP:
 - restricted to a small number of dimension and measure types
 - Measures (count(), sum(), avg()) applicable to numeric data)
 - Drill-down, roll-up all functions are user-controlled process
- Concept description:
 - Can handle complex data types of the attributes and their aggregations
 - More automated process



Characterization vs. OLAP

- Similarity:

- Presentation of data summarization at multiple levels of abstraction.
- Interactive drilling, pivoting, slicing and dicing.

- Differences:

- Automated desired level allocation.
- Dimension relevance analysis and ranking when there are many relevant dimensions.
- Sophisticated typing on dimensions and measures.
- Analytical characterization: data dispersion analysis.



Characterization vs. OLAP

