CHAPTER-11

Specifying a data mining query for characterization with DMQL:

11.Specifying a data mining query for characterization with DMQL

11.1 . Introduction

Suppose that 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# (telephone number) , and gpa (grade_point_average) .A data mining query for this characterization can be expressed in the data mining query language DMQL as follows:

use Big_University_DB

mine characteristics as"Arts.Students"

in relevance to name , gender ,major, birth_place, birth_date, residence, phoned, gpa

from student where Status in "graduate" We will see how this example of a typical data-mining query can apply attribute- Oriented induction for mining characteristic descriptions.

The first step of attribute-oriented induction is data focusing and it should be

Performed prior to attribute-oriented induction . The data are collected based on the

Information provided in the data-mining query . Since a data mining query is usually

Relevant to only a portion of the database , selecting the relevant set of data not only

Makes mining more entire data base.efficient , but also derives more meaningful results than mining on

The Specifying the set of relevant attributes (i.e. , attributes for mining , as indicated in

DMQL with the in relevance to clause)may be difficult for the user. A user may select

Only a few attributes that she feels may be important, while missing others that could

also play a role in the description . For example , suppose that the dimension birth_place

is defined by the attributes city, province_or_state, and country . Of these attributes , the

user has only thought to specify city. In order to allow generalization on the birthplace

dimension , the other attributes defining this dimension should also be included . In other

words, having the system automatically include province_or_state and country as relevant attributes allows city to be generalized to these higher conceptual levels during

the induction process.

At the other extreme , a user may introduce too many attributes by specifying all of the possible attributes with the clause “in relevance to *”. In this case , all of the

Attributes in the relation specified by the from clause would be included in the analysis.

Many of these attributes are unlikely to contribute to an interesting description.

“ What does the ‘**where** status **in** “graduate “’ clause mean?” This **where** clause

implies that a concept hierarchy exists for the attribute status . Such a concept hierarchy

organizes primitive-level data values for status , such as “M.Sc. ‘;”M.A”, “M.B.A.’\

“Ph.D.”, “B.Sc.” “B.A.”, into higher conceptual levels , such as “graduate “and

“undergraduate “. This use of concept hierarchies does not appear in traditional

relation query languages , yet is a common feature in data mining query languages.

The essential operation of attribute-oriented induction is data generalization,

which can be performed in either of two ways on the initial working relation: attribute

Removal and attribute generalization .

11.2 Attribute removal is based on the following rule: 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(e.g., there is no concept hierarchy defined for

Attribute j, or (2) should be removed from the working relation.

An attribute-value pair represents a conjunct in a generalized tuple, or rule .The

Removal of a conjunct eliminates a constraint and thus generalizes the rule. If, as in case

1, there is a large set of distinct values for an attribute but there is no generalization operator for it, the attribute should be removed because it cannot be generalized, and preserving it would imply keeping a large number of disjuncts which contradicts the goal of generating concise rules. On the other hand, consider case 2, where the higher-level concepts of the attribute are expressed in terms of other attributes . For example, suppose that the attribute in question is street, whose higher-level concepts are represented by the attributes {city, province_or_state, country}. The removal of street is equivalent to the application of a generalization operator . This rule corresponds to the Generalization rule known as dropping conditions in the machine learning literature on learning from examples.

11.3 Attribute generalization is based on the following rule: If there is a large set of distinct values for an attribute in the 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, This rule is based on the following reasoning .Use of a generalization operator to generalize an attribute value within a tuple , or rule., in generalizing the concept it represents. This corresponds to the generalization rule known as climbing generalization trees in learning from examples , or concept tree ascension. Both rules, attribute removal and attribute generalization , claim that if there is a large set of distinct values for an attribute , further generalization should be applied. This raises the question :how large is " a large set of distinct values for an attribute "considered to be? Depending on the attributes or application involved , a user may prefer some attributes to remain at a rather low abstraction level while others are generalized to higher levels. The control of how high an attribute should be generalized is typically quite subjective . The control of this process is called attribute generalization control . If the attribute is generalized "too high," it may lead to overgeneralization , and the resulting rules may not be very informative . On the other hand , if the attribute is not generalized to a “sufficiently high level “ then under-generalization may result , where the rules obtained may not be informative either . Thus , a balance should be attained in

attribute-oriented generalization. There are many possible ways to control a generalization process . We will describe two common approaches. The first technique , called 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 , further attribute removal or attribute generalization should be performed. Data mining systems typically have a default attribute threshold value (typically ranging From 2 to 8) and should allow experts and users to modify the threshold values as well.

If a user feels that the generalization reaches too high a level for a particular attribute, the threshold can be increased . This corresponds to drilling down along the attribute. Also , to further generalize a relation , the user can reduce the threshold of a particularattribute , which corresponds to rolling up along the attribute.

The second techniques , called generalized relation threshold control, sets a threshold for the generalized relation . If the number of (distinct) tuples in the generalized relation is greater than the threshold , further generalization should be performed . Otherwise , no further generalization should be performed . Such a threshold may also be present in the data mining system (usually within a range o 10 to 30),or set by an expert or user, and should be adjustable . For example , if a use feels that the generalized relation is too small, she can increase the threshold , which implies drilling down . Otherwise , to further generalize a relation , she can reduce the threshold , which implies rolling up. These two techniques can be applied in sequence ; first apply the attribute Threshold control technique to generalize each attribute , and then apply relation threshold control to further reduce the size of the generalized relation. No matter which generalization control technique is applied , the user should be allowed to adjust the generalization thresholds in order to obtain interesting concept descriptions.

In many database-oriented induction processes , uses are interested in obtaining quantitative or statistical information about the data at different levels of abstraction . Thus , it is important to accumulate count and other aggregate values in the induction process. Conceptually , this is performed as follows . A social measure , or numerical attribute , that is associated with each database tuple the aggregate function , count. Its value for each tuple in the initial working relation is initialized to 1. Through attribute removal and attribute generalization tuples within the initial working relation may be generalized , resulting in-groups of identical tuples. In this case , all of the identical tuples forming a group should be merged into one tuple-The count of this new, generalized tuple is set to 1 total number of tuples from the initial working relation that are represented (i.e., were merged into) the new generalized tuple . For example , suppose that attribute-oriented induction , 52 data tuples from the initial working relation is all generalized to the same tuple , T. That is , the generalization of these 52 tuple resulted in 52 identical instances of tuple T. These 52 identical tuples are merge to form one instance of T, whose count is set to 52. Other popular aggregate functions include sum and avg. For a given generalized tuple , sum contains the sum of the values of a given numeric attribute for the initial working relation tuple making up the generalized tuple. Suppose that tuple T contained sum (units_sold) as n aggregate function . The sum value for tuple T would then be set to the total number of units sold for each of the 52 tuples . The aggregate avg(average)computed according to the formula , avg=sum/count.

11.4 Example:

Attribute-oriented induction :Here we show how attribute-oriented induction is performed on the initial working relation . For each attribute of the relation , the

Generalization proceeds as follows:

1. name : Since there are a large number of distinct values for name and there is no generalization operation defined on it , this attribute is removed.

2. gender : Since there are only two distinct values for gender , this attribute is retained and no generalization is performed on it.

3. major : Suppose that a concept hierarchy has been defined that allows the attribute major to be generalized to the values {arts ,science , engineering , business}.

   Suppose also that the attribute generalization threshold is set to 5, and that there

   are over 20 distinct values for major in the initial working relation . By attribute

   by climbing the given concept hierarchy.

4. birth_place: This attribute has a large number of distinct values ; therefore , we would like to generalize it . Suppose that concept hierarchy exists for birthplace,

   define as city <province _or_state_country>. If the number of distinct values for

   country in the initial working relation is greater than the ; attribute generalization

   operator exists for it, the generalization threshold would not be satisfied . If instead, the number of distinct values for country is less than the attribute generalization threshold , then birthplace should be generalized to birth_country.

5. birth_data: Suppose that hierarchy exists that can generalize birth_date to age, and age to age_range and that the number of age ranges for intervals) is small with respect to the attribute generalization threshold . Generalization of birth_date

   should therefore take place.

6. residence : Suppose that residence is defined by the attributes number ,street, residence_city , residence_province_or_state , and residence_country. The number

7. phone #: As with the attribute name above , this attribute contains too many distinct values and should therefore be removed in generalization.

8. gpa : Suppose that a concept hierarchy exists for gpa that groups values for grade point average into numerical intervals like {3,75-4.0,3.5-3.75,….},which in turn are grouped into descriptive values , such a [excellent , very good , …]. The attribute can therefore be generalized.

The generalization process will result in-groups of identical tuples. Based on the Vocabulary used in OLAP, we may view count as a measure , and the remaining Attributes as dimensions . Note that aggragate functions , such as sum , may be applied to Numerical attributes , like salary and sales . These attributes are referred to as measure attributes.

Implementation techniques and methods of presenting the derived generalization are discussed in the following subsections.

11.5 Review Questions

1.Expalin about Attribute removal

2 Explainb about Attribute generalization

11.6 References

[1]. Data Mining Techniques, Arun k pujari 1st Edition

[2] .Data warehousung,Data Mining and OLAP, Alex Berson ,smith.j. Stephen

[3].Data Mining Concepts and Techniques ,Jiawei Han and MichelineKamber

[4]Data Mining Introductory and Advanced topics, Margaret H Dunham PEA

[5] The Data Warehouse lifecycle toolkit , Ralph Kimball Wiley student Edition