A Novel Approach of Horizontal Aggregation in SQL for Data Mining

M.Saravanan¹, P.Mythili²

¹Research Scholar, Sathyabama University, Chennai, India
²Post Graduate Student, Sathyabama University, Chennai, India

Abstract— In data mining, an important goal is to generate efficient data. To analyse data efficiently data mining uses horizontal tabular layout. There are three fundamental methods to evaluate horizontal layout, they are SPJ, CASE and PIVOT which has its own advantages. Preparing datasets in data mining requires many SQL queries for joining tables and aggregated columns. In Data mining projects, Classification is one of the most significant tasks which consumes more time. This paper presents an efficient implementation technique for SQL by using optimised C4.5 algorithm to perform classification. By using optimised C4.5 algorithm we can handle high dimensional records with minimal time.

Keywords— Aggregation, Horizontal Aggregation, SPJ, CASE, PIVOT, Optimised C4.5

Introduction

Data mining is a process of extracting information from data set and transforms it into a logical structure. Data mining is mainly used for analysing data. It allows users to examine data from different dimensions and acknowledges the relationships identified.

In general, data mining is a process of finding patterns among relational databases and also repository of information into the databases where data sources can include database, web, and data warehouse.

In a relational database, all data are stored into the database and accessed via relationships. A special work is needed to prepare data set that can be used as input for data mining algorithms. Most algorithms need input as data sets in horizontal layout with one value per column. Each and every study group uses different terms to represent data sets. In data mining most commonly used terms are point-dimensions where as in statistics literature uses observation-variables and in machine learning research uses instance feature.

There are three methods to evaluate horizontal aggregations. Those are SPI, CASE, and PIVOT. SPI: This Method is based on relational algebra operators; CASE: This compares the values using IF-ELSE; The PIVOT and UNPIVOT are the two functions used to exchange rows and columns values. Classification is one of the significant problems in data mining. This paper proposes an efficient SQL implementation of the optimised C4.5 algorithm to perform classification in large database. The Objective of the paper is to handle high dimensional data by using optimised C4.5 algorithm.

In General, the study of mined databases on worksheets is more flexible to have aggregated functions on one group per row. The advantage of horizontal aggregation is to reduce manual work in data preparations and it automatically generates codes for datasets. The optimised C4.5 algorithms are used to generate decision tree for the given data sets which later undergoes horizontal aggregation function.

I. RELATED WORK

C.Cunningalam[1] introduces an optimization technique by using relational database management system. PIVOT and UNPIVOT operators, exchanges row values into column values. This system is easy to implement and reduce complexity. Xiang Lan[2] proposed an technique to reduce cost model for query performance. It checks an effectiveness of GDR, ADR and LDR by using partition based randomized search for selected queries C.Ordenez[3] proposed a technique for statistical models in DBMS. To compute summary matrices, linear sum of points and quadratic cross products are introduces. It helps to provide an efficient SQL queries. M.Madhavi [4] proposed query evaluator methods. In this paper, it compares CASE and PIVOT method for speed complexity and SPI method are used only for theoretical concepts. Rajesh Reddy [5] experiments an different way to mine the database.Introduced a vertical and horizontal view to analyse the data sets. It helps to reduce workload and time complexity. Durka.c [6] introduced horizontal aggregations a new standard of pivoting function is incorporated in data mining. This can be achieved using SQL server analysis services, where data are transformed into knowledge cubes by using MDX queries. The knowledge data are customised based on generalized and suppression algorithm. It increases the performance and efficiency.

II. AGGREGATION

To generate data sets for data mining projects, an efficient and well summarized data are needed. Databases are said to be a collection of information about data. To extract information from database SQL is used. Aggregation function plays a...
major role in summarization of rows and columns. Commonly used Aggregation functions are sum(), avg(), min(), max() and count(). There are two types of aggregation. One is vertical aggregation and other is horizontal aggregation.

Vertical aggregation produces result in form of vertical layout and contains many rows. There are methods which help to produce results in vertically aggregated form. This is similar to standard SQL functions.

Horizontal aggregation produces result in form of horizontal layout. Extended SQL is needed to produce result in horizontal layout. The syntax for horizontal aggregation is the following,

```
SELECT columns, aggregation function (parameters)
FROM GROUP-BY columns
```

IV. PROPOSED METHODOLOGY

A. Dataset Creation

Data set is an important for all operations in data mining. To retrieve a data from database we normally use SQL queries to those data. After retrieval, various extractions and transformations are made for suitable application.

Dataset creation involves four major steps. First step is to select relevant data for analysis. Second is data integration, which collects data from various resources. They are combined and stored in form of table. Third is data transformation which transforms the data into specified format. The last step is data reduction. In data reduction, data is compressed for easiness of operations.

The main problem occurs in creation and transformation of operands for analysis is the selection of relevant record from large database. Sometimes analysis need summarized details. So there is a need of summarized data. Horizontal aggregation uses extended SQL syntax for transformations.

There is framework for efficient dataset. Create a dataset by update, delete, insert and modify SQL queries. The optimised C4.5 algorithm helps to build decision tree and classify the data. The summarized and aggregated data undergoes decision tree which helps to understand easily. It is simple for user and reduces the time complexity. If the tree is large then they are less comprehensible. A decision tree transformed into IF-THEN rules that is one of most important forms for representation. Even though data set is costly and complexity, data set can be prepared and data are stored in database.

B. SQL Code Generation

The main aim is to identify the patterns to generate SQL code for operations. The SELECT statements can be expanded with the clause. In a horizontal aggregation there are four input parameters:

a. The input table P1, P2...Pd
b. The list of GROUP BY columns S1, S2...Sk
c. The column to aggregate (X)
d. The list of transposing K1...Kz

This aggregation query will results a large table with n+1 column with one group for each values. Query optimizer estimates the query and shows the result.

C. Query evaluation method

There are three methods

1. SPJ method
2. CASE method
3. PIVOT method

1. SPJ method:

SPJ method compares the values between two operands. It is important in theoretical point of view. The syntax is as follows,

```
SELECT(1_column,2_column.....n_column) aggregation function
FROM Table
GROUP BY(1_column,2_column....3_column)
```

2. Pivot method:

PIVOT exchange the series of rows into columns. Pivot used to determine the new set of columns in table. The syntax is as follows,

```
SELECT column
FROM tablename
PIVOT(aggregation function)
FOR(column name)
IN(first_pivotcolumn,Second_pivot column.........last_pivot column)
```

3. Case method:

Case method combines GROUP BY and CASE functions. It is more efficient and widely used. This helps us to minimize the complexity and results in horizontal layout.

The syntax is as follows,

```
CASE Expression
WHEN Expression THEN result expression
[....n]
ELSE result_expression
End
```

V. CLASSIFICATION BASED ON DECISION TREE

A Decision trees is one of the most powerful tools for classification. In SQL, a classification is the significant task. There are many algorithms for creating Decision trees. A decision tree is in the form of tree structure, where each inner node represents an attribute, each branch represents the result set and leaf node represents distributions. The top of the node in a tree is said to be a root node. An example of a tree is
given on Figure 1. The optimised C4.5 algorithm helps to build decision tree based on classification.

![Decision Tree Diagram](image)

**Figure 1: Example for Decision Tree**

VI. PRUNING TREE

When a decision tree is developed, other branches will reflect anomaly. Tree pruning methods reduce the problem of over flowing of the data. There are two common methods in tree pruning. They are pre-pruning and post pruning. In the pre-pruning method, a tree is pruned by stopping its development. Upon halting, the node becomes a leaf. In the post pruning method, a tree is pruned after it is grown. A complete tree node gets pruned by removing its unwanted branches. The lowest un-pruned node becomes a leaf.

VII. OPTIMISED C4.5 ALGORITHM

**INPUT:** An Attribute-Valued dataset, AD

**STEPS:**

1. Tree [ ]
2. If AD is “NULL” or other stopping criteria met then
3. Terminate
4. End If
5. For all attribute a € AD
6. Do
7. Compute information, if we split on a
8. End For
9. \( a_{\text{best}} \) Best attribute according to above computed criteria
10. Tree = create a decision node that test \( a_{\text{best}} \) into the root.
11. AD = Induced sub datasets from AD based on \( a_{\text{best}} \)
12. For all AD, do
13. Tree = C4.5 (AD,)
14. Attach tree to corresponding branch of tree
15. End For
16. Return tree

VIII. EXPLANATION

The Algorithm works in the following manner. A sample datasets are collected. If the samples are of relevant group, then the node becomes a leaf. Otherwise, the algorithm finds an entropy-based measure. It is also known as information gain and used for selecting the attribute. The resultant attribute is known as decision attribute. The attribute result in the form of discredited or in categorical form. A branch is created for known value of the test attribute and the samples are partitioned. The optimised C4.5 algorithm uses the same process repeatedly to form a decision tree for the samples. Once an attribute has generated at a node, it need not be considered in any of the node. The repeated partitioning stops only when any one of the following conditions is true. The distribution of the node, samples are stored. There are no samples for the branch of test-attribute. Hence the leaf is created along with the class in samples.

IX. EXPERIMENTAL RESULT AND ANALYSIS

In this paper, initially extraction of information is collected as data sets. First step is to upload the dataset into the database and also data can be added manually by the end user. Then there is a possibility of modification, update and deletion also can be done in data sets. Horizontal aggregated method is performed by SPJ, CASE method and PIVOT method. The optimised C4.5 algorithm used to classify the datasets. It includes decision tree based on set of rules from datasets. After performing aggregated functions on data sets, a decision tree is generated. It makes easy to understand and can handle huge amount of data. It reduces time complexity. Thus optimised C4.5 performs well in classifying the data sets.

X. CONCLUSION

An optimised C4.5 algorithm builds decision tree and classifies the given data. It supports us to understand easily and interpret the results set. In future, the work can be extended to examine the each and every classification models and prepare dataset with more information and high quality.

REFERENCES


[2] Xiang Lian, “General cost model for evaluating dimensional reduction in high dimensional space”. IEEE Trans on Knowledge and Data Engineering, 22(1), 139-144, 2010

[3] C.Ordonez Statistical model computation with UDFs. IEEE Trans on Knowledge and Data Engineering, 22(1), 139-144, 2010


edge and Data Engineering, 22, 2010.