Ersch. in: Data security and security data : revised selected papers / 27th British National Conference on Databases, BNCOD 27, Dundee, UK, June 28 - 30, 2010. Lachlan M. MacKinnon (ed.). - Berlin [u.a.] : Springer, 2012. - S. 118-121. - (Lecture Notes in Computer Science ; 6121). - ISBN 978-3-642-25703-2

http://dx.doi.org/10.1007/978-3-642-25704-9_13

Enabling Decision Tree Classification in Database Systems through Pre-computation

Nafees Ur Rehman and Marc H. Scholl

University of Konstanz, Box D 188,
78457 Konstanz, Germany
{Nafees.Rehman,Marc.Scholl}@Uni-Konstanz.de

Abstract. Integration of data mining in database systems is an open topic of research. The DBMS's power of dealing with lots of data and maintaining data integrity adds to the motivation of integrating it with data mining. We propose a method to integrate decision tree classification to do the required pre-computations and store it in database objects for later use. These pre-computed values get updated with the introduction of new data or change in the existing data for classification. Decision tree classification can readily make use of these pre-computed values to build classification models. Our approach is based on the column database to use it effectively for feature oriented calculations. This comparatively improves performance if classification is deemed to be performed on a high dimensional data.

Keywords: Data Mining, Decision Tree Classification, Database Systems.

1 Introduction

The tight coupling of data mining with database systems is an interesting challenge. The first study about integrating data analysis methods into DBMSs came with the development of data warehousing and On-Line Analysis Processing (OLAP) in particular [1]. Most DBMS vendors included data mining features into their products[2,3,4]. The motivation to integrate data mining with databases comes from the facts that these both fields are data oriented. Data mining is itself a merged field of various domains to analyze mostly large volumes of data and find useful patterns in it. While database systems store, manipulate and retrieve tons of data efficiently. DM lacks the scalability feature to be able to deal with volumes of data. It then depends on feature selection [5] or sampling [6] to reduce the size of the data in order to perform in- memory data mining. On the other hand, database management system is meant to handle large volume of data and maintain data integrity. Bringing the best of the both worlds together would help data analysts to perform complex analysis with least concern for scalability of data mining and integrity of the data.

2 Pre-computations

Databases and Data Warehouses are natural repositories to store data for different purposes. These data are processed using a variety of ways. One common approach is

to query data using SQL statements. Complex queries can take considerable time. One method to answer queries efficiently is to use pre-compute data and *materialize* it in an object, called Materialized View. Materialized views are query results that have been stored in advance so long-running calculations are not necessary when SQL statements are actually executed. From a physical design point of view, materialized views resemble tables or partitioned tables and behave like indexes [7]. This improves performance of the queries significantly.

Our idea of pre-computations for data mining is partially based on this concept of materialized views. As data mining is not a one shot activity but rather an iterative and interactive process. Data mining can be thought of set of various algorithms. Each algorithm operates on a data set by performing various computations over it. Decision tree classification algorithms are re-cursive in its operation. To build a decision tree model, at each step of the algorithm, all features (attributes) are evaluated to choose the best feature as a root node. We consider the ID3 algorithm to show how various values can be pre-computed for it.

ID3 algorithm takes in a set of categorical variables and evaluates each variable to find the best split variable using the *information gain* method. This process is then repeated recursively until all variables are tested and/or all instances are classified according to a given output variable. The evaluation of each variable involves performing a series of calculations from counting instances to calculating entropy and information gain. This operation can become costlier when numbers of variables are more. And this is the motivation for our method of pre-computations. All these mentioned calculations can be performed in advance and can be stored in a materialized view. And later, can be exploited by the algorithm without re-computations.

As soon as a decision tree model is declared and the number of classes is known, Entropy and Information gain can be calculated and stored. And when the algorithm is actually run, it can make use of these values. This saves time and makes the algorithm work efficiently.

3 Storage of Pre-computations for DTC

In order to store potentially pre-computable values that would later be fed to DTC program, there can be two simple ways to do it. One way is to have a separate storage structure for each of the table that contains observations for the training of the decision tree. Figure 1 depicts this method where for each base table there is one associated dependent MV to hold these pre-computed values. This method is simpler but it will require having the same number of MV as the number of base tables. That would contribute to the maintenance efforts in a negative way.

But as new data is pumped into the base tables, only corresponding MVs would be updated to reflect the change in the source data. The second approach is to have a central database-wide table to hold potentially pre-computable values for all base tables. Figure 2 represents this approach. This single table approach would reduce the maintenance efforts.

Base_tbl_1

ObsNo	Form	Parenting		Class
1	1	usual		Rec
2	2	Great		Ntr
3	1	Prêt		Ntr
---	---	---		---

DPMCtbl_1

CNo	FieldID	CalcType	Value
1	Form	E	0.45
2	Parent	E	0.32

Base_tbl_2

ObsNo	Outlook	Temp		Class
1	cloudy	High		1
2	sunny	Low		0
3	over	Med		0
---	---	---		---

DPMCtbl_2

CNo	FieldID	Calc_Type	Value
1	Outlook	E	0.693
2	Outlook	E	0.247

Base_tbl_n

ObsNo	F1	F2		Class
1	A	X		Y
2	B	Y		N
3	C	Z		N
---	---	---		---

DPMCtbl_n

CNo	FieldID	CalcType	Value
1	---	---	---
2	---	---	---

Fig. 1. Representation of pre-computation structures: One MV for one each table containing observations for DTC training

CNo	TableID	FieldID	C_Type	Value
1	PlayGOLF	PlayGOLF	Entropy	0.940
2	PlayGOLF	Outlook	Entropy	0.693
3	PlayGOLF	Outlook	Gain	0.247

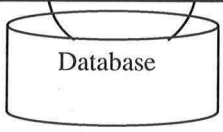

Fig. 2. Representation of pre-computation structures: A database-wide table for all datasets containing observations for training DTCs

4 Storage of DM Results into an Appropriate Storage Structure

Data Mining is a complex set of operations and takes a lot of efforts and resources to perform it. The output of almost each phase of DM is input to another phase. The time taken for processing at each phase can vary from microseconds to hours depending upon the size of the data, machine and type of tasks. Results of each phase and of the whole DM process can be stored in the existing DB/DWH structures to provide for efficient retrieval at a later stage.

There can be two ways to store trees. One is to use a separate table for each tree that is constructed. This may take more space on the disk and will result in more maintenance efforts as not all trees will be useful. Trees not useful will be discarded. The other method is to use a database-wide table to store all trees that are constructed. Each node of each tree will be inserted as a record in this table along with information about the node's level number, the SplitVar (attribute in database vocabulary), Split_Value, Operator, Parent Node, Version and TableID to keep track of the table for which the tree is constructed.

5 Summary and Conclusion

In this paper we proposed a method of integrating Decision Tree Classifiers in Database Systems. The integration aims to figure out pre-computable values required for the construction DTCs and store these values in MV or in a central database-wide table. These values can then be fed to the ID 3 algorithm allowing not only for integration but also to make it efficient. All computations requires for to choose the root node, are already performed. The algorithm will carry out its process further at the sub-tree levels. Furthermore, storage of the DTCs is also proposed which allows for interactive and iterative data mining model construction.

References

1. Codd, E., Codd, S., Salley, C.: Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. Technical report, EF Codd and Associates (1993)
2. IBM. DB2 intelligent miner scoring (2001), `http://www4.ibm.com/software/data/iminer/scoring`
3. Oracle, Oracle 9i data mining. White Paper (2001)
4. Soni, S., Tang, Z., Yang, J.: Performance Study of Microsoft Data-Mining Algorithms. Microsoft White Paper pages (October 2000)
5. Liu, H., Motoda, H.: Feature Selection for Knowledge Discovery and Data Mining, p. 214 (1998)
6. Chauchat, J., Rakotomalala, R.: A new sampling strategy for building decision trees from large databases (2000)
7. Oracle, Oracle 9i Data Warehousing Guide (2002)