Augmenting OLAP Exploration with Dynamic Advanced Analytics

Benjamin Leonhardi
IBM Deutschland
Schönaicher Str. 220
71032 Böblingen, Germany
bleon@de.ibm.com

Bernhard Mitschang
Universität Stuttgart
Universitätsstrasse 38
70569 Stuttgart, Germany
Bernhard.Mitschang@ipvs.uni-stuttgart.de

Rubén Pulido
IBM Deutschland
Schönaicher Str. 220
71032 Böblingen, Germany
rpulido@de.ibm.com

Christoph Sieb
IBM Deutschland
Schönaicher Str. 220
71032 Böblingen, Germany
christoph.sieb@de.ibm.com

Michael Wurst
IBM Deutschland
Schönaicher Str. 220
71032 Böblingen, Germany
mwurst@de.ibm.com

ABSTRACT

Online Analytical Processing (OLAP) is a popular technique for explorative data analysis. Usually, a fixed set of dimensions (such as time, place, etc.) is used to explore and analyze various subsets of a given, multi-dimensional data set. These subsets are selected by constraining one or several of the dimensions, for instance, showing sales only in a given year and geographical location. Still, such aggregates are often not enough. Important information can only be discovered by combining several dimensions in a multidimensional analysis. Most existing approaches allow to add new dimensions either statically or dynamically. These approaches support, however, only the creation of global dimensions that are not interactive for the user running the report. Furthermore, they are mostly restricted to data clustering and the resulting dimensions cannot be interactively refined.

In this paper we propose a technique and an architectural solution that is based on an interaction concept for creating OLAP dimensions on subsets of the data dynamically, triggered interactively by the user, based on arbitrary multi-dimensional grouping mechanisms. This approach allows combining the advantages of both, OLAP exploration and interactive multidimensional analysis. We demonstrate the industry-strength of our solution architecture using a setup of IBM® InfoSphere™ Warehouse data mining and Cognos® BI as reporting engine. Use cases and industrial experiences are presented showing how insight derived from data mining can be transparently presented in the reporting front end, and how data mining algorithms can be invoked from the front end, achieving closed-loop integration.

EDBT 2010, March 22–26, 2010, Lausanne, Switzerland.
Copyright 2010 ACM 978-1-60558-945-9/10/0003 ...$10.00

1. INTRODUCTION

1.1 Motivation

Online Analytical Processing is a very popular technique for explorative data analysis. Usually, a fixed set of dimensions (such as time, place, etc.) is used to analyze various subsets of a given, multi-dimensional data set. These subsets are selected by constraining one or several of the dimensions, for instance, showing sales for a given year and geographical location. Still, such aggregates are often not enough. Important information can only be discovered by combining several dimensions in a multidimensional analysis.

Assume we have sales data describing when customers bought products in any of 20 product categories. Analysing each category together with other OLAP dimensions is almost impossible in any real-life situation, as the number of different combinations is too large. However, customers usually can be grouped in a few groups of customers with similar buying habits. Such clusters represent a single new OLAP dimension that can easily be combined with existing dimensions such as time or geography. Another important case in which users would like to combine several dimensions dynamically is to define local semantic meaning to certain groups. For instance, income and age of customers can be combined to define groups like ”elderly rich customers in Germany”.

1.2 Requirements and Related Work

Existing approaches use multi-dimensional analysis to create such additional OLAP dimensions for the complete set of records. This is, however, not sufficient.

- Some dimensions may only be defined on a subset of data (in the given example: we may find different typical customer groups in different regions or at different points in time). Using a grouping mechanism to create a global new dimension would not be able to deliver appropriate results in this case.
- The subsets of data for which we want to create an additional dimension will often emerge in the course of the analysis, i.e., ad hoc and not beforehand.

An appropriate solution must allow the user to create local ad hoc OLAP dimensions:

1. dynamically (after the warehouse has been deployed)
2. interactively (during the process of analysis, triggered by the user)
3. locally, on subsets of the data (on a slice/dice)
4. using arbitrary grouping mechanisms (e.g. data mining, rules, etc.)
5. recursively (newly defined groups can be further refined)

Most existing approaches to create OLAP dimensions automatically use static approaches [6, 2]. They create additional dimensions statically on all data records, as the warehouse is designed, violating (1). In [2], prior knowledge of the data set is obtained before performing OLAP operations, not fulfilling (2). Furthermore, the method for obtaining this prior knowledge is restricted to data clustering using neural network technology which additionally violates (4). None of the existing approaches allows adding dimensions locally, violating (3).

Some approaches allow adding new dimensions dynamically [8, 3]. These approaches only allow creating global dimensions (violating (3)) and are not interactive for the user that runs the report (the dimensions have to be added by the database administrator, violating (2)).

The approach presented in [7] provides a new operator for multidimensional on-line analysis. It uses Agglomerative Hierarchical Clustering to achieve a semantic aggregation on the attributes of a data cube dimension. This approach, however does not allow dynamically creating new dimensions and thus has a different scope than our approach that combines different dimensions to create a new one.

Another approach that is related to this work, but too preliminary is On-line Analytical Mining (OLAM) [4, 5]. The idea is, to select an arbitrary subset of data and then to apply data mining only to this data and to visualize the result to the user in a data mining tool. The same approach is also followed by [9]. The purpose for this integration is similar to ours, has, however, the important drawback that no new OLAP dimension is created. Therefore, the grouping cannot be applied recursively (violating (5)) and the newly created dimension cannot be combined with the existing ones. Furthermore, the user has to switch between two visualization modes.

1.3 Contribution and Overview

In this paper we propose a technique and an architectural solution that is based on an interaction concept for dynamically creating OLAP dimensions on subsets of data. It can be triggered interactively by the user and supports arbitrary multidimensional grouping mechanisms. This approach allows combining the advantages of both, OLAP exploration and interactive multidimensional analysis. We demonstrate the industry strength of our solution architecture using a setup of IBM InfoSphere Warehouse data mining and Cognos BI as reporting engine. Industrial experiences are presented showing both how insight derived from customer segmentation can be transparently presented in the reporting OLAP front end, and how data mining algorithms can be invoked from the reporting front end, achieving thus a closed loop integration.

In Section 2 we outline the method and interaction concept underlying augmented OLAP exploration and we give an illustrative example. Section 3 describes our solution architecture and an implementation based on IBM InfoSphere Warehouse and Cognos BI. In Section 4 we report on industrial experiences that assess the benefit of our augmentation approach to OLAP exploration. Finally, Section 5 presents a short summary of the achievements and an outlook containing topics for future research.

2. AUGMENTED OLAP EXPLORATION

We propose a method and an interaction concept for dynamically creating OLAP dimensions on subsets of the data. It can be, triggered interactively by the user, based on arbitrary multi-dimensional grouping mechanisms. Our method consists of the following basic steps to solve the problem of interactively creating local OLAP dimensions. First, the user may select any subset of the data that is analyzed. This could be achieved, for instance, by setting constraints on one or several OLAP dimensions (e.g. by selecting only customers from a given region). Then a multi-dimensional grouping function can be applied to the selected subset of the data that divides this dataset into a number of partitions (e.g. the customers could be grouped into clusters of similar customers). This grouping is then imported as a new OLAP dimension along with the other dimensions and can be used just as one of them (e.g. the user can analyze how much money each of the typical customer groups spends on average). Additionally, our method allows the user to refine automatically created dimensions by recursively partitioning all the data records in a specific group until a desired level of detail is reached (e.g. the user could further split the group of users that have an over average spending on computer equipment to find customer subgroups in this group).

This approach allows combining the advantages of both, OLAP and interactive multidimensional analysis: first, the ability for ad hoc data exploration known from traditional OLAP analysis and second, the ability to group datasets according to several variables at once.

2.1 Formal Description

OLAP allows users to analyze data by inspecting and aggregating over several dimensions. Typical dimensions are, for instance, time, location, etc. OLAP allows drilling down on such dimensions, e.g. the user can select to see only sales in a given region and in a given time frame.

Formally, an OLAP analysis consists of a set of dimensions D on a set of data records X. Each dimension is associated with a hierarchy of dimension values (e.g. dimension "location" may have several regions subdivided into individual cities). These dimensions are used to formulate constraints C on the underlying data that can be represented as conjunction of basic conditions $c*$ on the underlying dimensions

$d_i \in D$ of the datasets,

$$C := c_1(d_1) \wedge \ldots \wedge c_n(d_n)$$

thus C then uniquely defines a subset of data records.

We propose a method that allows defining one or more additional dimensions (let $d*$ denote a single new dimension) relative to a subset of data records defined by a constraint set C.

This method consists of the following steps:

1. Select a subset of data $X_c \subseteq X$ by defining a constraint set C. This step is achieved by traditional OLAP analysis.
2. Apply any (multidimensional) grouping mechanism that creates a new dimension $d*$ as a multivariate combination of existing dimensions. This can be represented as function $d^* = f(d_1, ..., d_n, X_c)$ with d_i in D and X_c being the set of data records that fulfils constraint C. Function f can be implemented in a variety of ways. We propose the use of
   (a) Rules that define a new dimension: A set of rules defines a grouping based on existing dimensions by associating each group with a constraint set (for example a rule that defines elderly, rich customers as such who have an average balance over a certain threshold and are older than 60).
   (b) Multidimensional data clustering methods that group data records according to a predefined similarity, such that the result of the analysis are groups of records similar to each other. (Customers can, for example, be grouped automatically into clusters based on which products they tend to buy).
3. The newly created dimension d^* is integrated into the OLAP analysis engine locally, thus it is only displayed if the data records X_c (or any subset) are selected.
4. Also, the user may refine local dimension d^* by recursively splitting up its values. For example, the top-level of a dimension may be formed by "frequent users" and "non-frequent users". The user may choose to split up the first group dynamically to obtain sub groups, such as "frequent users with many international and data calls" and "others".

These steps can be freely interrelated in order to support our method and interaction concept for creating OLAP dimensions on subsets of the data dynamically, triggered interactively by the user, based on arbitrary multi-dimensional grouping mechanisms. In this way all the requirements stated in the introduction are fulfilled.

2.2 Example

Assume the user wants to analyze customer data from a telecommunication company based on buying habits. For each customer the following data is stored. The original OLAP definition contains five dimensions on customer data: place of residence, age, and total spending on local, international and data calls respectively, thus D= {place($=d_1$), age($=d_2$), spLocal($=d_3$), spInternational($=d_4$), spData($=d_5$) }.

The user is now especially interested in people younger than 30 that live in Berlin. Thus she uses the constraint C = (place is Berlin) $\wedge$ (age $<$ 30), this yields a subset of customers X_c (see Table 1).

Table 1: A subset of 95 customers is selected.

	Place: Berlin	Place: Munich	Place: Stuttgart
Age: 15-30	95	55	50
Age: 30-45	200	150	100
Age: 45-50	100	150	100

The user now wants to find typical customer segments in this subset of data records and applies a multidimensional clustering algorithm to get d* = f(spLocal, spInternational, spData, X_c), where f groups the customers in X_c according to how much local, international and data calls they use. This yields two groups of customers, the ones that use mainly local calls and a second one that use mostly data and international calls (Figure 1).

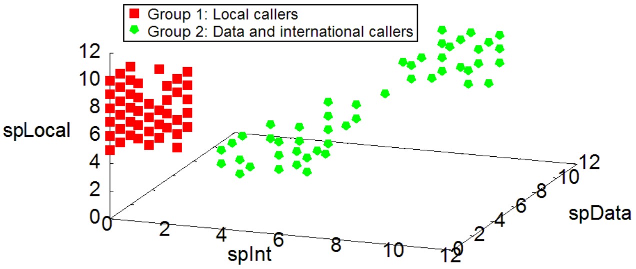


Figure 1: The selected subset of customers is locally clustered into two groups.

These groups are integrated as new dimension into the existing OLAP analysis. This integration can be used, for instance, to find out that the "local callers" are equally distributed among different age groups under 30, while the other customers are mostly between 25 and 30 (see Table 2 and Table 3).

Table 2: This subset of customers is locally clustered into two groups.

	Place: Berlin	
	d*: Group 1	**d*: Group 2**
Age: 15-30	45	50

The user now wants to find out more about the second group and thus refines it. This yields two subgroups, namely the customers using only data calls and the others using all three types of calls (Figure 2).

Table 3: The resulting dimension is locally integrated with the existing OLAP dimensions.

	Place: Berlin	
	d*: Group 1	**d*: Group 2**
Age: 15-20	16	1
Age: 20-25	14	3
Age: 25-30	15	46

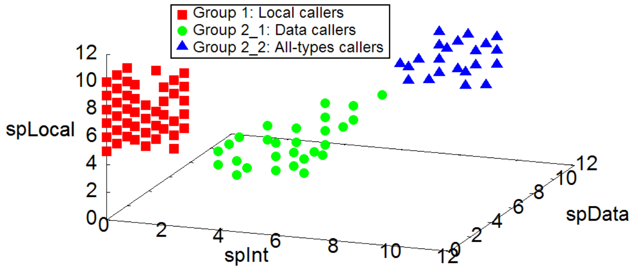


Figure 2: The selected subset of customers is locally clustered into two groups.

Again, these dimensions are added to the OLAP analysis (Table 4).

Table 4: The refined dimension is integrated into the OLAP analysis.

	Place: Berlin	
	d*: Group 2_1	**d*: Group 2_2**
Age: 15-30	28	22

This may reveal that the group of data callers is situated mostly in the centre of Berlin. (Table 5). Now, the user may select an entirely different subset using a second constraint set and repeat the analysis.

Table 5: The refined dimension is integrated into the OLAP analysis.

	Place: Berlin, center	**Place: Berlin, outskirts**	**Place: Berlin, center**	**Place: Berlin, outskirts**
	d*: Group 2_1		**d*: Group 2_2**	
Age: 15-30	26	2	11	11

3. SYSTEM ARCHITECTURE

In this section we present an architecture for an integrated reporting - mining solution enabling large scale delivery of mining results, as well as realizing augmented OLAP with current industry products. For data mining results to be widely consumed they must be accessible through thin clients, like web browsers. This reduces installation and configuration efforts as well as client hardware requirements, since data mining computations and report generation take place in the backend application or data tier. When a user asks for a report to be generated typically an HTTP request is sent to the reporting engine, as shown in Figure 3-1. For standard static reports, the reporting engine would then access the data in the relational data sources through SQL calls and generate the reports which are returned to the end user. To achieve a closed-loop integration, data mining products must allow reporting engines to invoke mining tasks while a report is generated. This way, reports can be enriched with insight dynamically derived from the underlying data. Since most reporting tools can communicate with the database, data mining algorithms can be made accessible to reporting tools by encapsulating them as stored procedures installed in the database. Thus the reporting engine can invoke mining tasks through stored procedure calls (Figure 3-2). This way data mining is dynamically invoked at report generation time making interactive analysis possible. Reports are no longer limited to presentation of precomputed mining results, but the user is also enabled including parameter values on her request to tune the data mining algorithms to her individual needs. An easy-to-use reporting interface completely abstracts the underlying complexity of the mining algorithms from the user. Complex parameters are usually determined by the mining engine, whereas only business related parameters are allowed to be adjusted by the user. The stored procedure calls then trigger the whole data mining process. Such a process includes modeling, i.e. creating a model of the extracted patterns in the data (Figure 3-3), and/or scoring, i.e. applying the model to new data records (Figure 3-4). Since both, the data and the mining algorithms reside in the database the whole data analysis process takes place inside this protected environment. This solves the security issues in a business environment. Moreover, inconvenient, time consuming data movement is avoided resulting in a more efficient mining process. Storing mining results along with the data in the database allows defining triggers that update the models once the data is modified. This way consistency is ensured, since the data mining models always correspond to the current data. Data security can be ensured through database authentication and encryption, preventing unauthorized access to both, data and computed mining models. Finally, mature database backup and recovery capabilities prevent loss of data and mining models when system failures occur. The data mining models computed during the modeling phase are usually stored in XML based files, defined by the PMML standard. This does achieve a degree of interoperability, since generation of mining models, and scoring of new data can be performed in different data mining products, as long as they both adhere to the PMML standard. However, very few reporting tools can read data directly from PMML files. Data mining tools are required to provide functionality to transform PMML mining models to a format that can be widely accessed by reporting tools. This is achieved through table extractors (Figure 3-5), which parse PMML models and store data mining results in a relational-table format. This way data mining results can be accessed by most reporting engines. This functionality is crucial for the integration of powerful data mining and reporting solutions. Both, model information and scored data can, thus, be easily accessed through SQL calls which return the data as result sets (Figure 3-6). The reporting engine can then use such data to generate tables, charts and scorecards. Typically HTML reports are then returned to the end user (Figure 3-7).

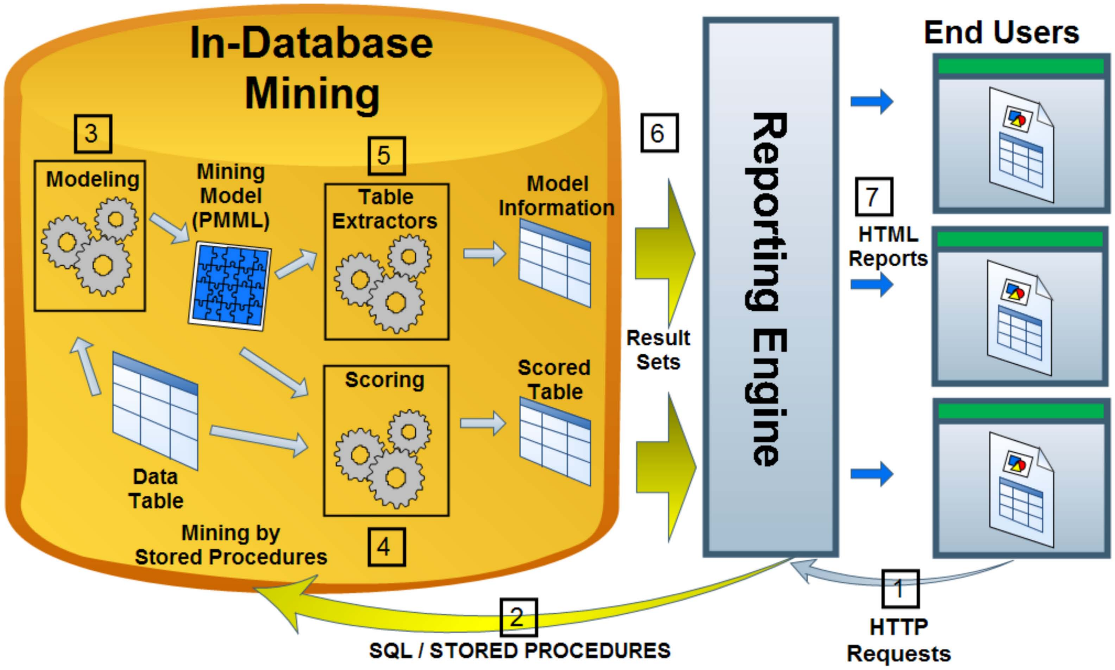
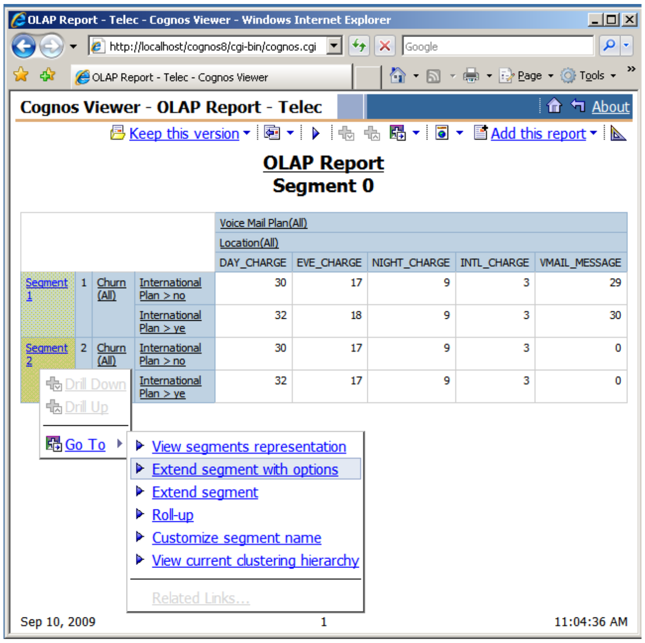


Figure 3: Architecture for an integrated industry strength reporting - mining solution

Reporting tools usually derive results by applying basic arithmetic operations to the information stored in a data warehouse. Most reporting tools provide also relational OLAP or come with an OLAP server for MOLAP functionality. While IBM Cognos is one of the market leaders in reporting it does not provide data mining, since these are beyond the scope of a reporting framework. In light of their complementing features, InfoSphere Warehouse and IBM Cognos BI can be combined in a flexible way to offer a comprehensive BI solution. We have devised an integration in which analysts are presented with traditional Cognos reports that contain the most up to date information stored in the database. Additionally analysts may benefit from insight buried into the data, which is extracted by data mining functions of InfoSphere Warehouse, and integrated into the traditional Cognos reports. Invoking mining functionality dynamically from Cognos front end tools isolates analysts from the technical complexity of data mining, allowing them to repeatedly issue data mining queries without the assistance of a more advanced user, benefiting thus of a fully interactive and iterative knowledge discovery process. Cognos users may select within Cognos reports the parameters to be used by the data mining algorithms. InfoSphere Warehouse mining algorithms are invoked through stored procedure calls, directly from Cognos reports. Mining takes place in the data warehouse. First a model is generated containing the hidden patterns in the data derived by the corresponding data mining algorithm. The information in the model can then be put into relational tables through table extracting functions. The mining model is then applied to the data (scoring) and results are saved in the database. The extracted model information and the scored data records are then returned to the Cognos reports in the form of result sets. More information on the integration of InfoSphere Warehouse data mining with Cognos reporting can be found in [1].

4. INDUSTRY EXPERIENCE - CUSTOMER SEGMENTATION IN THE TELECOMMUNICATIONS DOMAIN

Customer segmentation allows grouping customers into segments of mutually similar customers. Telecommunication companies usually gather data about the demographic aspects of their customers (age, profession, place of residence, etc.) as well as data about their transactions (number of calls at different times, contracts, etc.). An analysis of this combined data could reveal customer groups that could not been predicted from common knowledge. For example, elderly customers that spend a lot on international calls but do not call at night time. Information about the typical behaviors of such groups can then be used by the marketing department to develop specialized products or services. In the given application, we created a report that did interactively present a customer segmentation model through a Cognos report. An important prerequesite was, that the user should be able to specify which aspects to use for the segmentation: only demographic information, usage data or both. It should also be possible for the user to specify the maxmimum number of clusters. The customer segments are shown in a report, as an additional dimension of an OLAP cube, which can be explored and extended from the reporting front end (Figure 4). This way, standard OLAP operations aggregating business figures, can benefit from a semantic dimension that groups customers not only according to a single attribute as location, product plan or age group, but to a combination of them. Again, this proved to be a key to make data mining more accessible to end-users already familiar with reporting and OLAP. The parameters of the


OLAP Report - Telec - Cognos Viewer - Windows Internet Explorer
http://localhost/cognos8/cgi-bin/cognos.cgi
Google
OLAP Report - Telec - Cognos Viewer
Page
Tools
Cognos Viewer - OLAP Report - Telec
About
Keep this version
Add this report
Drill Down
Drill Up
Go To
View segments representation
Extend segment with options
Extend segment
Roll-up
Customize segment name
View current clustering hierarchy
Related Links...

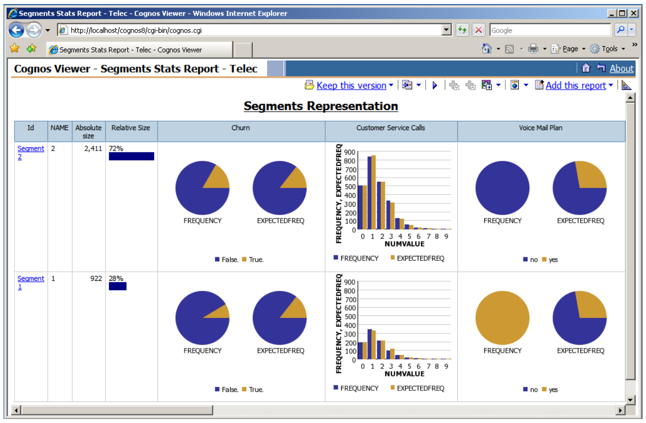
OLAP Report
Segment 0

				Voice Mail Plan(All)				
				Location(All)				
				DAY_CHARGE	EVE_CHARGE	NIGHT_CHARGE	INTL_CHARGE	VMAIL_MESSAGE
Segment 1	1	Churn (All)	International Plan > no	30	17	9	3	29
			International Plan > ye	32	18	9	3	30
Segment 2	2	Churn (All)	International Plan > no	30	17	9	3	0
			International Plan > ye	32	17	9	3	0

Sep 10, 2009 1 11:04:36 AM

Figure 4: Extending a segment through data clustering from a Cognos OLAP Report

clustering algorithm were reduced to the ones relevant in the application. The ability to visualize mining models in conjunction with other information in the report (Figure 5) led to a powerful combination that enables new usage scenarios.



Segments Representation

Id	NAME	Absolute size	Relative Size
Segment 2	2	2,411	72%
Segment 1	1	922	28%

Figure 5: Visualizing customer segments from a Cognos Report

5. CONCLUSIONS

In this work we discussed the benefits of augmenting OLAP analysis with insight dynamically extracted through advanced analytics. We explained, both formally and through an example, how the user can interactively trigger the dynamic creation of OLAP dimensions defined by arbitrary multidimensional grouping mechanisms on subsets of the data.

We introduced an architecture that uses SQL both to invoke mining in the database and to read the mining results using specialized table extractors. This solution is very simple, yet powerful, as it allows combining data mining with every reporting tool that is able to handle SQL input. Also, information from mining results can be freely combined with any other information in the report, as both use a table format.

We implemented this architecture based on InfoSphere Warehouse and Cognos BI. Also, we described our experiences in applying this system in the telecommunication domain. The ability to treat OLAP data and mining results in the same way (transparent to the reporting tool) proved to be a key to create complex, custom applications within very short time. The same holds for the ability to invoke mining with various parameters through stored procedures.

Besides further experience and fine tuning of the interaction concept, there are a number of issues that warrant further research. First, performance can be improved by reducing the number of round trips through caching and intelligent pre-processing. The latter can be achieved by means of dynamic index generation and exploitation, resulting in a more efficient report generation. Second, to enhance the usability of the interaction concept, a history function allowing the analyst to recapture and replay the latest steps is desirable.

IBM, Cognos and InfoSphere are registered trademarks or trademarks of International Business Machines Corp. in many jurisdictions worldwide.

6. REFERENCES

[1] Developer works article series, integrate infospehere warehouse data mining with ibm cognos reporting. http://www.ibm.com/developerworks/data/library/techarticle/dm-0810wurst/index.html.

[2] S. Asghar, D. Alahakoon, and A. Hsu. Enhancing olap functionality using self-organizing neural networks. *Neural, Parallel Sci. Comput.*, 12(1):1–20, 2004.

[3] A. Company. Extending cep with real-time olap. www.aleri.com/files/Aleri-Live-OLAP.pdf.

[4] J. Han. Olap mining: An integration of olap with data mining. In *Proceedings of the 7th IFIP 2.6 Working Conference on Database Semantics*, pages 1–9, 1997.

[5] J. Han. Towards on-line analytical mining in large databases. *ACM SIGMOD Record*, 27:97–107, 1998.

[6] N. Kerdprasop and K. Kerdprasop. Enhancing the power of olap with knowledge discovery. In *The 7th International Conference on Software Engineering and Applications (SEA 2003)*, 2003.

[7] R. B. Messaoud, O. Boussaid, and S. Rabaséda. A new olap aggregation based on the ahc technique. In *Proceedings of the 7th ACM international workshop on Data warehousing and OLAP*, pages 65–72. ACM, 2004.

[8] A. A. Vaisman, A. O. Mendelzon, W. Ruaro, and S. G. Cymerman. Supporting dimension updates in an olap server. In *Proceedings of the 14th International Conference on Advanced Information Systems Engineering*, London, UK, 2002. Springer-Verlag.

[9] M. G. Z. Liu. A proposal of integrating data mining and on-line analytical processing in data warehouse. *International Conf. on Info-tech and Info-net.*, 2001.