Evaluating the scalability of data mining provider classifiers

C. L. Curotto¹ & N. F. F. Ebecken²
¹CESEC/UFPR – Civil Engineering Graduate Program, Brazil
²COPPE/UFRJ – Civil Engineering Graduate Program, Brazil

Abstract

Two classifiers implemented as Data Mining Providers are considered. These providers runs as a stand-alone servers or aggregated with Microsoft® SQL Server. One of these classifiers is the Microsoft® Decision Trees algorithm. The other is the Simple Naive Bayes incremental classifier, that supports continuous input attributes, multiple discrete predictable attributes and incremental updating of the training data set. The performance study carried out to verify the scalability of the classifiers includes factors of cardinality (number of training cases), number of input attributes, number of states of the input attributes and number of predictable attributes.

1 Introduction

A great effort has been spent to achieve the tight coupling of DM (Data Mining) and OLAP (On-Line Analytical Processing) techniques in database application development environments [1,2,3,4].

Toward this objective the Microsoft® OLE DB for DM technology [5] provides an industry standard for developing DM algorithms and although it has been created by an enterprise, it is totally independent of any proprietary software. This technology was included in the last release of the Microsoft® SQL Server. The Analysis Services component of this software includes a DM provider supporting two algorithms: one for decision trees classification [6] and other for clustering [7]. The DM Aggregator feature of this component and the OLE DB for DM Resource Kit [8] make possible for developers and researchers to implement new DM algorithms.
This paper describes an implementation of an incremental Simple Naive Bayes classifier [9] and shows the study carried out to verify the scalability and performance of this classifier.

2 The implementation

The SNB (Simple Naive Bayes) classifier is well known and has been described in many papers and books [10]. The main objective of a DM algorithm is to predict attributes based on a set of cases of input attributes. Succinctly, the SNB classifier uses counts of discrete and continuous attributes occurrences and means and standard deviations of continuous attributes to do this task.

To support incremental update of dataset, it is enough to store the sum and the square sum of continuous attribute values, computing means and deviations as necessary. Multiple prediction trees are supported by an adequate data structure. Details about this formulation are described by Curotto [11].

The DMcMlMine Provider implements the incremental SNB classifier starting from the source code included with Sample Provider of OLE DB for DM Resource Kit [8], which includes the complete implementation of an aggregated provider.

The OLE DB for DM technology [5] is described in detail by Netz et alii [12,13]. They stated precisely the key operations that must be supported by a DM provider algorithm on DM models, reproduced as follows:

- Define a mining model, identifying the set of predictable attributes, the set of input attributes, and the algorithm used to build the mining model;
- Populate a mining model from training data using the algorithm specified;
- Predict attributes for new data using a mining model that has been populated;
- Browse a mining model for reporting and visualization applications;

These key operations will be the key steps of the DM provider implementation. The well-known AllEletronics customer database [10] will be used as the training dataset to illustrate these steps.

2.1 Creating the mining model

The syntax of DM SQL commands of the new algorithm is defined using a parse analyzer. With a few modifications in the source code, the support for the new algorithm is ready. The Relational Mining Model Editor of Analysis Services Manager can be used to create DM SQL commands [5]. The Sample Provider source code includes all necessary steps to create the mining model. The DM SQL command to create a model with two predictable attributes of AllEletronics dataset is shown below:

```
```
2.2 Populating the mining model

The DM SQL command showed below is generated automatically by Analysis Services Manager and is used to populate the mining model using data stored on MS SQL Server. All support for this task must be developed for new algorithms. The model data structure is defined and all functions related to training the dataset, assembling the model tree, saving and loading this model are developed. This data structure must support the two following key steps.

```sql
INSERT INTO [AllElet-SNB] (SKIP, [Age], [Income], [Student], [CreditRating], [BuysComputer])
OPENROWSET ('SQLOLEDB.1', 'Provider=SQLOLEDB;Integrated Security=SSPI;Persist Security Info=False; Initial Catalog=AllElet; Data Source=CLC', 'SELECT NumReg, Age, Income, Student, CreditRating, BuysComputer FROM AllEletTrain')
```

2.3 Predicting attributes

The DM SQL command to predict attributes can be manually assembled or generated by the Data Transformation Services of MS SQL Server. All support for this task must be developed for new algorithms. An example of this command is shown below:

```sql
SELECT FLATTENED [T1].[NumReg], [T1].[Age], [T1].[Income], [T1].[Student], [T1].[CreditRating], [T1].[BuysComputer], [AllElet-SNB].[BuysComputer] as BuysComputer FROM [AllElet-SNB] PREDICTION JOIN OPENROWSET ('SQLOLEDB.1', 'Provider=SQLOLEDB.1;Integrated Security=SSPI;Persist Security Info=False; Initial Catalog=AllElet; Data Source=CLC', 'SELECT "NumReg", "Age", "Income", "Student", "CreditRating", "BuysComputer" FROM "AllElet" ORDER BY "NumReg"') AS [T1] ON [AllElet-SNB].[NumReg] = [T1].[NumReg] AND [AllElet-SNB].[Age] = [T1].[Age] AND [AllElet-SNB].[Income] = [T1].[Income] AND [AllElet-SNB].[Student] = [T1].[Student] AND [AllElet-SNB].[CreditRating] = [T1].[CreditRating] AND [AllElet-SNB].[BuysComputer] = [T1].[BuysComputer]
```

2.4 Browsing a mining model

The model of DM is exposed for visualization and report generator applications by means of rowsets. The content of these rowsets is retrieved by DM provider client applications by DM SQL Select queries. All support for this task must be developed for new algorithms. An example of this query is:

```sql
SELECT * FROM [AllElet_SNB].CONTENT
```
3 Performance study

Soni et al. [14] showed a performance study for MSDT (Microsoft® Decision Trees) classifier. This work is used as a template for experiments to evaluate the scalability and the performance of the DMclcMine SNB classifier. Four factors are considered: cardinality (number of cases), number of input attributes, number of states of the attributes and number of predictable attributes. The results of the experiments are compared with those produced by the MSDT classifier.

The experiments were made with the same equipment used to implement the DM provider: an IBM® PC compatible microcomputer, one Intel® Pentium® II 500 MHz processor, 512 MB RAM, 30 MB hard disk, virtual memory of 768 MB minimum and 2,048 MB maximum. Software installed: MS Windows® 2000 Advanced Server SP2; MS SQL Server 2000 Enterprise SP3; MS Visual Studio® 6.0 SP5 with Visual C++®, Visual J++® and Visual Basic® compilers; MS Platform SDK February 2001 Edition and Sandstone Visual++ Parse 4.00.

The datasets used by the experiments are generated artificially by a computer program [15]. Details of this generation are showed by Curotto [11].

In each experiment, the training time is measured varying one variable. The next table shows the parameters and the description of the classifiers used in these experiments. All input and predictable attributes are discrete.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>Description</th>
<th>Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSDT</td>
<td>Microsoft® Decision Trees Classifier [6]</td>
<td>COMPLEXITY_PENALTY = 0.9, MINIMUM_LEAF_CASES = 10, SCORE_METHOD = 4, SPLIT_METHOD = 3;</td>
</tr>
</tbody>
</table>

3.1 Varying the number of input attributes

In this experiment the number of training cases was fixed at 1,000,000. Each input attribute may have 25 different states. The predictable attribute may have 5 classes. Figure 1 shows the graph with the training times for the following sequence of number of attributes: 12; 25; 50; 100; 200.

The SNB classifier showed an absolutely linear behavior and a better performance than the MSDT classifier. It is interesting to note that as reported by Soni et al [14], the MSDT classifier spent 130 minutes (7,800 seconds) to train 1,000,000 cases in an equipment (4 Intel Xeon 550 MHz processors, 4 GB RAM) better than the one used in this experiment. The figure 1 shows that this time was only of 8,673 seconds.
3.2 Varying the number of training cases

This experiment was used to compare the performance of the classifiers varying the cardinality of the training dataset. The numbers of training cases were: 10,000; 25,000; 50,000; 75,000; 100,000; 1,000,000; 2,000,000; 10,000,000. The number of input attributes was fixed at 20 with 25 different states. The predictable attribute had 5 classes. The graph of the figure 2 displays the training processing times.

Again, the SNB classifier showed an absolutely linear behavior and a better performance than the MSDT classifier.

The graph of the figure 3 displays the processing times to predict the same datasets. This experiment made possible the evaluation of the processing capacity of the used equipment. When processing the prediction of 1,000,000 cases with the SNB classifier, there was a lack of memory after 2 hours and 16 elapsed minutes. At this time the installed virtual memory was only of 1,024 MB. Only after the virtual memory was increased to 2,048 MB this task could be completed. However this task exceeds the efficient capacity of the equipment due to the intensive use of virtual memory, which causes the non linear behavior in the results showed by the mentioned graph.

Another problem happened with the prediction task of 1,000,000 cases using MSDT classifier. DM providers retrieve the cases using a simple query of a table with 10,000,000 registers stored in MS SQL Server. However a non configurable time limit of 30 seconds exists to carry out query operations in this server. As queries of this size can exceed this limit, this task is impossible. This is a bug reported by Microsoft® that should be fixed in future releases of the server. To by-pass this problem, tables can be used instead of queries. If the use of queries is required, this can be done through SQL Query Analyzer, which allows the configuration of this limit. Surprisingly this problem was also solved with the increase of the virtual memory.
The graph of the figure 3 shows that the linear behavior of SNB classifier, for the prediction task, occurred only till the number of cases of 100,000. From this value on a non linear behavior occurred due to the high demand of memory required by the classifier. This fact can be explained by the same reasons exposed previously.
The use of memory of the SNB classifier to process the prediction of 1,000,000 cases was approximately of 1.6 MB. As expected considering this fact, the prediction task of 10,000,000 cases was impossible to be finished due to lack of memory (approximately 2.2 MB were in use after 6 hours and 34 minutes of processing). To make possible this processing, a table was used instead of a query. However the MSDT classifier presented an error exceeding the time limit. The difference of the two kinds of errors can be explained by the way that each one of the classifiers accesses the training data internally. As reported by Bernhardt et al [16], the classifier MSDT owns a processing module that makes queries directly to MS SQL Server, justifying the kind of error presented.

To by-pass the problem of the prediction task, a procedure was elaborated for prediction by parts. Thus the data was divided in parts of 10,000 cases. The lines of the graph in figure 3, representing the classifiers MSDTp and SNBp, shows the result of this test proving the linear behavior of the two classifiers for this task.

Experiments were made with continuous data and both classifiers presented results similar to the results presented for discrete data.

![Graph](image)

Figure 4: Incremental Training Time x Number of Training Cases.

To test the incremental resource of the classifier SNB, the experiment of the cardinality was repeated simulating the number of training cases increment. Thus the numbers of cases were: 10,000; 25,000; 50,000; 75,000; 100,000; 1,000,000; 2,000,000; 10,000,000. For comparison, the cardinality results of the non incremental SNB classifier are reproduced again in figure 4, labeled SNB. The results of the non incremental SNB classifier for this experiment correspond to the accumulated values of the previous results, represented in the graph by the line labeled SNBa. The results of the incremental SNB classifier are represented
in the graph by the line labeled SNBi. Obviously the results of processing times of the incremental method are inferior to those obtained by the non incremental ones. However the comparison of lines SNB and SNBi shows that the incremental method allowed the obtaining of results lightly better (~2.5%) than those obtained by the non incremental method.

3.3 Varying the number of states of the input attributes

In this experiment the number of training cases is fixed at 1,000,000. The number of data attributes is constant equal to 20. The predictable attribute may have 5 different classes. The states of the input attributes were: 2; 5; 10; 25; 50; 75; 100. The results for the processing times are shown in the graph of figure 5. This graph shows an evident linear behavior of the SNB classifier. The MSDT classifier oscillated in this same tendency, presenting linearity larger than the one reported by Soni et al [14].

3.4 Varying the number of predictable attributes

SNB and MSDT classifiers own the resource of generating multiple trees, considering multiple predictable attributes. This experiment studies the behavior of the two classifiers in face to the variation of the number of predictable attributes in the following sequence: 1; 2; 4; 16; 32. While the other values stay constant: 1,000,000 cases; 40 total attributes, including among these the predictable attributes for each task; and 25 states for each one of the attributes, including the predictable attribute.

The graph of figure 6 shows the linear behavior of the SNB classifier, which presented a slight increase of the training time when the number of predictable attributes increases. Again the classifier MSDT showed a linear behavior better than those reported by Soni et al [14].
Figure 6: Training Time x Number of Predictable Attributes.

4 Conclusion

As predicted, due to its statistical formulation, as well as the data structure of the implementation, the SNB classifier showed high scalability inside the limits of the equipment used in the experiments.

The good results considering the large datasets puts the classifier as a mandatory option in the study of these sorts of problems. Using the incremental training associated to the prediction by parts, virtually problems with any size can be solved.

The size of the dataset used in the experiments exceeds many of real life problems, proving the usefulness of the implementation.

The OLE DB DM technology showed to be efficient for implementation of algorithms of DM integrated with relational databases.

It was demonstrated that the use of microcomputers is possible for the solution of medium level DM tasks.

The non linear behavior of the classifier for large datasets suggests additional research using equipments with larger capacity, including the use of federations of databases servers, as well as multiprocessors computers.

References


[2] Han, J. et alii, DBMiner: A System for Mining Knowledge in Large Relational Databases, Proc. of the KDD'96, 2nd Int'l Conf. on Knowledge
Data Mining IV

Discovery in Databases and Data Mining, Portland, Oregon, USA, pp. 250-255, 1996.


