Distributed Data Mining vs. Sampling Techniques: A Comparison*

Mohamed Aounallah, Sébastien Quirion, and Guy W. Mineau

Laboratory of Computational Intelligence,
Computer Science and Software Engineering Department,
Laval University, Sainte-Foy, Québec, G1K 7P4, Canada;
{Mohamed.Aoun-Allah, Guy.Mineau}@ift.ulaval.ca, SQuirion@gel.ulaval.ca

Abstract. To address the of mining a huge volume of geographically distributed databases, we propose two approaches. The first one is to download only a sample of each database. The second option is to mine each distributed database remotely and to download the resulting models to a central site and then aggregate these models. In this paper, we present an overview of the most common sampling techniques. We then present a new technique of distributed data-mining based on rule set models, where the aggregation technique is based on a confidence coefficient associated with each rule and on very small samples from each database. Finally, we present a comparison between the best sampling techniques that we found in the literature, and our approach of model aggregation.

1 Introduction

This paper deals with the problem of mining several huge geographically distributed databases, proposing and comparing two data mining techniques. The first one that we examined uses a sampling of each individual database to which, once gathered, we apply some data mining technique. This technique is based on the aggregation of data. With this intention, we studied the existing sampling techniques. The most promising methods, based on our tests, will be detailed later in this paper.

The second technique of data mining, that we introduce (based on the aggregation of models), consists of applying data mining techniques on each individual database. The models resulting from these techniques are then gathered and some aggregated model is produced by a technique described in what follows. In this work, models, either produced individually on a subset of the data or from the aggregation technique that we propose are a set of classification rules.

This paper proceeds as follows. In Section 2, an overview of the most common sampling techniques is presented. Then, in Section 3, we present our solution to distributed data mining (DDM) using model aggregation (DDM-MA). In Section 4, we present our experimentation results. We finally present a conclusion and our future work.

* This work is sponsored by NSERC.

A.Y. Tawfik and S.D. Goodwin (Eds.): Canadian AI 2004, LNAI 3060, pp. 454–460, 2004.

2 Sampling

The sampling approach consists in creating a representative sample of a large database under the hypothesis that a classifier trained from that sample will not perform significantly worse than a classifier trained on the entire database. In our context, sampling is used on each remote database, generating distinct samples at each site. We then merge these samples to finally train a classifier on the resulting sample. The literature in data mining is filled with various sampling algorithms [3] [6] [4], which can be grouped, with respect to how the samples are formed, to form three distinct families: static, dynamic and active sampling.

Static Sampling refers to a sampling that is performed without any knowledge other than what the database provides. The most common algorithm for static sampling could be called random sampling. As presented in [3], for a database D, an initial sample size n_0 and a schedule of sample size increments (typically either an arithmetic ($\Delta n_i = \lambda$) or geometric ($\Delta n_i = n_{i-1}$) schedule [6]), we first form an initial sample S of n_0 random items from D and, while the distribution of the attributes of S differs significantly from that of D, add to S an additional Δn_i random items from D $\setminus$ S.

Dynamic sampling differs from static sampling only in the sample validation process. At each iteration, a classifier is built from the current sample and evaluated. If the resulting classifier has not reached satisfactory accuracy, i.e. reaches a plateau in its learning curve, the algorithm must iterate once more. There are three common techniques to detect convergence: Local Detection (LD)(stopping when $acc(n_i) \leq acc(n_{i-1})$) [3], Learning Curve Estimation (LCE) [3] and Linear Regression with Local Sampling (LRLS) [6].

Active sampling differs from dynamic sampling only in the way the items are picked at each iteration. In the literature, active sampling is used in contexts where classified items are not provided to the learner *a priori*. The learner has to pick amongst unclassified items and ask an expert for their classification. The purpose of active sampling is then to minimize the number of items needed by the learner to acquire the concept correctly. They achieve this by estimating which items would provide the greatest *knowledge gain* (formally, an effectiveness score) and including them in the sample. In general, the effectiveness scores (ES) are computed either by a probabilistic classifier or by a committee of classifiers. In our context (using classified items), the different active sampling methods can be summarized by this algorithm:

1. `i` $\Leftarrow$ `0`
2. `S` $\Leftarrow$ {n_0 `random items from D`}
3. `Generate {C}, a set of classifiers, from S`
4. `While {C} has not converged`
   a) `i` $\Leftarrow$ `i + 1`

b) For each $x_j \in$ D $\setminus$ S, compute ES_j, the ES, with {C}
c) S $\Leftarrow$ S $\cup$ {Δn_i items chosen from D $\setminus$ S with respect to ES}
d) Generate {C} from S

Generally, the Δn_i items added to S at each iteration are the items with the highest ES. However, [8] proposes to use the ES as the item's weight in a random selection in order to add robustness to noisy data. Also interesting, [4] proposes, for a small cost in accuracy, to build the sample using a quick probabilistic classifier and then use that sample to train any other classifier. Our implementation of active sampling is an uncertainty sampling integrating these two approaches (Weighted Heterogeneous Uncertainty Sampling) for speed and robustness purposes.

3 Distributed Data Mining Using Model Aggregation

To construct our aggregated model, hereafter called the meta-classifier, we use two types of software agents: *miner agents* which mine individual (distributed) databases and *collector agents*, that are responsible for aggregating information produced by miner agents. There is typically only one collector agent in our system. Roughly speaking, our technique goes through the following algorithm. A detailed description with justifications of our choices can be found in [1].

1. Do, by miner agents, in parallel at different remote sites, for each database DB_i with $i = 1 \ldots nd$, where nd is the number of remote sites:
   a) Apply C4.5 over DB_i then transform the decision tree obtained to a rule set $R_i = \{r_{ik} \mid k \in [1..nr_i]\}$, where nr_i is the number of rules;
   b) Compute for each r_{ik} a confidence coefficient $c_{r_{ik}}$ as one minus the error rate of r_{ij} and minus one half the width of the confidence interval of the error rate computed based on the Central Limit theorem
   c) Extract a sample S_i from DB_i.
2. Do, by a collector agent, at a central site:
   a) Create R and S as follows:
   $R = \bigcup_{i=1\ldots nd} R_i$
   $S = \bigcup_{i=1\ldots nd} S_i$;
   b) From R, eliminate rules which have a confidence coefficient lower than a certain threshold t: $R_t = \{r_{ik} \in R \mid c_{r_{ik}} \geq t\}$;
   c) Create a binary relation $\mathcal{I}$ defined over $R_t \times S$ where, at each intersection (r_i, s_j), we find 0 if r_i does not cover s_j, 1 if r_i covers s_j correctly, or -1 otherwise ;
   d) For each rule $r \in R_t$, compute an error rate $Err^r_{\mathcal{I}}$ using S as test set, i.e. the number of -1 in each row of $\mathcal{I}$ divided by the number of non-zero values in the same row;
   e) Construct the rule set $R_{t_\mathcal{I}}$ using a threshold $t_\mathcal{I}$ as follows: $R_{t_\mathcal{I}} = \{r_p \in R_t | Err^{r_p}_{\mathcal{I}} \leq t_\mathcal{I}\}$.

4 A Comparative Experimentation

For our experiments, we have tested techniques proposed on nine data sets: adult, chess end-game (King+Rook versus King+Pawn), house-votes-84, ionosphere, mushroom, pima-indians-diabetes, tic-tac-toe, Wisconsin Breast Cancer (BCW)[5] and Wisconsin Diagnostic Breast Cancer (WDBC), taken from the UCI repository [2]. The size of these data sets varies from 351 to 45222 objects.

In order to determine the best sampling techniques, we divided each database into a training set (2/3) and a test set (1/3), when they are not already divided in the UCI repository. On the other hand, to test the DDM technique proposed and in order to simulate a distributed environment, firstly, we divided each database into two data subsets with a proportion of 1/4 and 3/4. The first subset is used as test set for the meta-classifier (aggregated model) or for the classifier built on the samples aggregation. The second subset is randomly divided into two, three or four data subsets of random sizes, which are, in turn, each divided into two sets with proportion of 2/3 (a data file) and 1/3 (the associated .test file) for training and test sets respectively. For the meta-classifier, a random sample set (an associated .sple file) is extracted from the training set (the .data file) with a size of 10% its size and a maximum of 50 objects. This maximum size is needed to bound the meta-classifier technique to very small data samples, which is in accordance to one of our assumptions.

4.1 Comparing the Sampling Methods

In order to compare sampling followed by a data mining technique on the aggregated samples, with distributed data mining as proposed in this paper, we decided to compare the various sampling methods to determine the one that succeeds best on our test data. So we compared both methods, dynamic and active sampling, using each of the three convergence detection mentioned earlier (*LD*, *LCE* and *LRLS*), testing each of these six methods with an arithmetic (*Arith.*) and geometric (*Geo.*) schedule. We also compared these methods to random sampling, with an arithmetic and geometric schedule, and to samples of 50 items formed by random picking and by weighted uncertainty sampling, for a total of 16 competing sampling methods.

Experiments have shown that "Active - LCE - Geo" and "Dynamic - LCE - Geo." are almost always among the three best methods. For each data set where this is not the case (i.e., one of these two methods does not appear among the best three methods), experiments have also shown that the methods "Active/Dynamic - LCE - Geo." are always within 5% of the most accurate method on any of the nine databases. These results suggest that these are the two most effective sampling techniques, at least for data sets that would resemble ours.

4.2 The DDM-MA Experiment

For the construction of the base classifiers we used C4.5 release 8 [7] which produces a decision tree that is then directly transformed into a set of rules. The

confidence coefficient of each rule is computed on the basis of 95% confidence interval (i.e., $N = 95$). For threshold $t_{\mathcal{I}}$, we used 5% and 10% respectively, but these two values gave exactly the same results. For threshold t we used i) all values ranging from 0.95 to 0.20, with decrements of 0.05, ii) 0.01, iii) and, μ with $\mu = 1/nd \sum_{i=1}^{nd} \mu_i$ and $\mu_i = 1/nr_i \sum_{k=1}^{nr_i} c_{r_{ik}}$. The value μ is used in order to get an automatic way to compute the threshold based on the average confidence that we have in the produced rules.

Experimentations of the meta-classifier with the different values of threshold t, previously listed, showed that μ gave the best performance. This is predictable since it is not an absolute value but rather it is a threshold that finds a consensus between the different μ_i by finding their closest value.

4.3 Comparison between Meta-classifier and Sampling

We base our comparison on the results obtained in sections 4.1 and 4.2. Consequently, in this section, we only compare Dynamic/Active - LCE - Geo. sampling techniques with the meta-classifier with $N = 95$, threshold $t = \mu$ and threshold $t_{\mathcal{I}} = 5\%$. Comparison is conducted on the basis of error rate and execution time. In order to assess the importance of the error rates obtained by the meta-classifier and sampling techniques, we compare them to error rates obtained on C4.5 applied to the whole database $DB = \cup_i DB_i$; it is used only as a reference to assess the loss accuracy since, by assumption, we stated that we could not process DB because of download/processing time constraints.

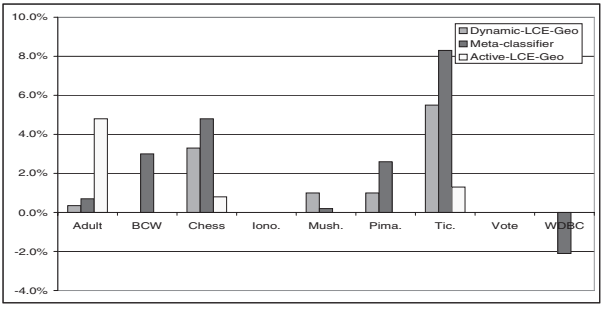

Fig. 1. Error rate comparison between the meta-classifier and the Active/Dynamic LCE Geo. sampling techniques assuming C4.5 error rate as reference

Figure 1 shows the difference between what we obtained using C4.5, versus the sampling techniques and the meta-classifier. The Dynamic - LCE - Geo sampling technique is represented by a light gray histogram, the Active - LCE - Geo sampling technique is represented by a white histogram and the meta-classifier approach is represented by a dark gray histogram. The first conclusion that we can extract from this chart is that all these error rates could be assessed to be acceptable since they are no more than 8.5% worse than C4.5 performance.

Comparing sizes. comparing the the size of each database to on one hand the size of the samples obtained with Active/Dynamic-LCE-Geo. sampling techniques and on the other hand the meta-classifier[1], we can conclude that in 4 cases (BCW, Iono., Vote and WDBC) the size of the samples issued from Active/Dynamic-LCE-Geo. sampling is the same as the database. This explains the fact that in Fig. 1 the error rate of the sampling techniques is the same as that of the C4.5 algorithm for these 4 cases producing a difference of 0. Surprisingly, in these 4 data sets, our meta-classifier gives the same error rate as the C4.5 in two cases (Iono. and Vote), 3% worse (BCW) or even better (WDBC) with samples as small as 34 items or less. In the 5 other cases, our meta-classifier has an error rate comparable to sampling techniques: better than one of the two techniques or worse than the worse sampling technique by no more than 2.80%. This performance is quite interesting since the meta-classifier sample sizes are much smaller than those required by the sampling techniques.

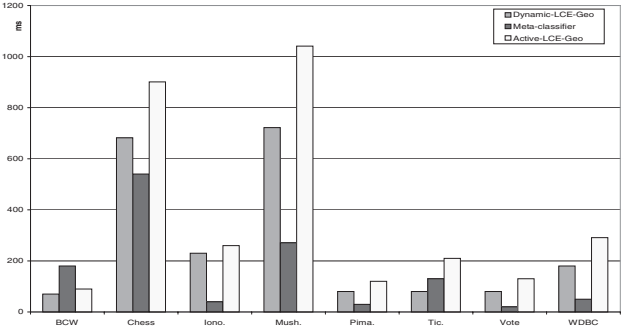

Fig. 2. Execution time comparison between the meta-classifier and the Active/Dynamic LCE Geo. sampling techniques

Comparing processing time. Finally, in order to compare the sampling techniques with the DDM-MA method on the basis of execution time, we can look at Fig. 2. Noting that these programs were programmed in C++, compiled by the same compiler and executed (single task dedicated CPU) on the same machine. While not presented on the chart for readability reasons, the execution times of the Dynamic/Active-LCE-Geo. sampling and the meta-classifier on the adult data set are respectively 20078ms, 38905ms and 9852ms. From this, one can easily conclude that, apart from the BCW and TIC data sets, that DDM-MA is always faster and sometimes, much faster than sampling techniques. Furthermore, the asymptotic analysis of the algorithm given in Fig. 2 shows that this would remain the case as N, the size of DB, grows.

[1] As reminder, the samples S_i used to produce the meta-classifier have a maximal limit of 50 items, but can be less if the size of DB_i is less than 500.

5 Conclusion

In this paper, we presented an overview of the most common sampling techniques as well as a new technique for distributed data mining based on the aggregation of models. Experiments highlighted the constantly high accuracy obtained by Active/Dynamic LCE Geo. sampling.

Our tests also provide us with the best parameters for the meta-classifier, which are used in our comparison with the most promising sampling techniques. Experiments showed that meta-classifier error rates are quite acceptable compared to those of sampling techniques or to that of a C4.5 applied on the whole database. Moreover, the meta-classifier uses samples of very small size compared to those produced by sampling techniques. A comparison of the processing time, showed that the meta-classifier is significantly more efficient than sampling techniques as data set size becomes important.

We can conclude that the meta-classifier presented in this paper is a promising technique. We are currently considering different approaches to improve its performance. For example, an importance coefficient could be affected to each object in a sample since the distributed databases could have significant differences in size. Moreover, we plan on integrating some efficient sampling techniques to extract the samples S_i used in the production of $\mathcal{I}$. Their impact on the various model aggregation techniques will be carefully assessed.

References

1. Mohamed Aounallah and Guy Mineau. Rule Confidence Produced From Disjoint Databases: a Statistically Sound Way to Regroup Rules Sets. *Accepted in IADIS international conference, Applied Computing*. 2004
2. C.L. Blake and C.J. Merz. UCI Repository of machine learning databases. http://www.ics.uci.edu/∼mlearn/MLRepository.html, 1998
3. George John and Pat Langley. Static Versus Dynamic Sampling for Data Mining. In Evangelos Simoudis and Jiawei Han and Usama M. Fayya, editors, *Proceedings of the Second International Conference on Knowledge Discovery in Databases and Data Mining*, pages 367–370, Portland, Oregon, August 1996. AAAI/MIT Press.
4. David D. Lewis and William A. Gale. A sequential algorithm for training text classifiers. In *Proceedings of SIGIR-94, 17th ACM International Conference on Research and Development in Information Retrieval*, pages 3–12, Dublin, IE, 1994. Springer Verlag, Heidelberg, DE.
5. O.L. Mangasarian and W.H. Wolberg. Cancer diagnosis via linear programming. *SIAM News*, 23(5):1–18, September 1990.
6. Foster Provost, David Jensen, and Tim Oates. Efficient Progressive Sampling. In Surajit Chaudhuri and David Madigan editors, *Proceedings of the Fifth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 23–32, N.Y., August 15–18 1999.ACM Press.
7. J. Ross Quinlan. *Improved Use of Continuous Attributes in C4.5*. Journal of Artificial Intelligence Research, 4:77–90, 1996.
8. Maytal Saar-Tsechansky and Foster Provost. Active Sampling for Class Probability Estimation and Ranking. http://www.mccombs.utexas.edu/faculty/Maytal.Saar-Tsechansky/home/MLJ-BootstrapLV-final.pdf, 2001.