Handling Mutual Inconsistencies in Multiple Databases

Alexander Logvinovsky, Vagan Terziyan *, Seppo Puuronen **

(*) Department of Artificial Intelligence and Information Systems,
Kharkov State Technical University of Radioelectronics, UKRAINE
e-mail: vagan@milab.kharkov.ua
(**)Department of Computer Science and Information Systems,
University of Jyvðskylð, FINLAND,
e-mail: sepi@jytko.jyu.fi

Abstract

It is obvious that the results of knowledge discovery in a large database could vary depending on the data mining method used. Dynamic selection of the most appropriate method can be solved in several ways. The metamethod proposed in [1] results in a clusterization (segmentation) of the whole data area into so called "competence areas" for every of the available methods. In such a case we obtain a "competence map" of methods upon the data set. It means that the metamethod defines which method should be used with each database object. However the selection problem is still actual when one tries to extract knowledge from multiple databases (possibly heterogeneous). Such attempt may cause a conflicting result even if each separate database is consistent. There are at least two types of conflicts to arise. The first one lays in possible data inconsistency within the area of the intersection of the databases. Such ambiguity is often depended on the appropriate database context and assumes that an object from the intersected part of two databases is classified in a different manner in each database separately. The second type conflict appears when the metamethod, working separately with two databases, selects different methods for each of them to be used with the objects from the intersected part. Hence we can differ four cases of handling multiple databases. The first one means that there is no inconsistency in integrated data and there is not inconsistency in integrated competence maps, thus there are no conflicts. The second one means that we have inconsistent integrated data and no mutual inconsistency between the competence maps. In this case we offer to decontextualize conflicting databases before integration. The third one includes a mutual inconsistency between competence maps only. Here we offer a method to integrate competence maps by handling inconsistency in their intersection. And the last (the most general) case contains both types of inconsistencies and can be handled by the integration of the two previous methods.

1. Introduction

Although the modern database technology enables storage of large streams of data, we do not yet have the technology to help us analyze, understand, or even visualize this stored data. Data mining is the process of nontrivial extracting valid, previously unknown, comprehensive, and actionable information from databases, and discovering useful patterns in data that are not obvious to the data's user. Such rapidly emerging field of knowledge discovery in databases has grown significantly in the past few years.

Several data mining methods have recently been developed to extract knowledge from large databases. Very often the method is applied statically without analyzing each particular instance (read «not considering the context») . If such applying of method is done dynamically taking into account the characteristics of each instance then it is possible to make decision about mutual dependencies both within the concepts of the separate database and among several ones.

To apply any statistical method in a "brute–force" manner it is not the best way to solve this problem, because none of such methods take into account neither self-context nor context knowledge about data they process. Our goal is any data inconsistency conflicts to be vanished and so we should choose and use the most appropriate method (or methods) taking into consideration the type of conflicts we want to solve. The scope of the conflicts is wide and so we will just consider conflicts for the numerical data.

There was designed data mining method allowing to obtain solution for the single database and multiple statistical methods [1]. The metamethod for selecting the most appropriate statistical method consists of two steps: training and classification of the unknown sample. While the first step one can collect classification characteristics for each training sample of the input training set using the Jackknife method. These characteristics include classification errors. As the result of this step we obtain a matrix Q_{NxM}, where N is number of training samples; M is number of classification methods and q_{ij} is equal to 1 if the classification result of the *j*-method for the *i*-sample of the training set is incorrect and 0 otherwise. The original Jackknife method to obtain the matrix Q_{NxM} consists of the following steps:

1) unmark all the samples of the training set;
2) take an unmarked *i*-sample from the training set. If there is no unmarked samples then stop.
3) derive the classification result for the *i*-sample using the training set without *i*-sample as the input of *j*-method for each statistical method (*j*=1, …, M);
4) derive the value q_{ij} as 1 if the classification result of the *j*-method for the *i*-sample is incorrect and 0 otherwise;
5) continue from step 2.

On the second step there is performed the classification of a new sample using the weighted majority algorithm. For this purpose there are selected *k* nearest neighbors of the new sample. The weight of *k*-neighbor is evaluated as a function of the distance between this *k*-neighbor and the new sample. Based on the weights and the values of the matrix Q_{NxM} one can select the most appropriate method.

If one examine this method attentively it is possible to notice a little fault of it. For this purpose let us consider the work of the method over one of the training set samples. Let it be *i*-sample. While the evaluation of the classification result for *i*-sample on the second step of metamethod we take it into account. And it is obvious that the *i*-sample is the nearest neighbor for itself and so it has weight equal to 1. But on the training step we have excluded this sample from the training set and hence have misestimated the value q_{ij}. To solve this problem we have strengthen this method by considering the influences of each of the training set samples over the classification result through both steps of metamethod.

But there appear some problems when one tries to obtain the classification result using data from multiple databases. These problems arise within an area of intersection of two database data. There are at least two types of such conflicts. The first type conflicts happen due to the possible data inconsistency of the intersection area and mean that for the same sample there exist two different classification results. The second type deals with an inconsistency of the competence maps of the methods within the intersection area and means that for the same sample metamethod selects different classification method for each database.

In [2] Chan has proposed the arbiter meta-learning approach for inductive learning. As we can see, Chan uses conflicting intersection area of two subsets to generate more precise classification result. The method can be stated as following:

1) divide whole data set *T* into subsets of the smaller size $T_1, \ldots, T_n$: $T = T_1 \cup \ldots \cup T_n$;
2) generate classifiers $C_1, \ldots, C_n$ for the subsets $T_1, \ldots, T_n$ accordingly;
3) determine the conflicting area T_{ij} for the pair of classifiers *i, j* over the union of the subsets T_i, T_j (i.e. subsets the classifiers have been learned from);
4) generate arbiter *A* over the pair of classifiers of the previous level and the subset T_{ij}; obtained arbiter is considered as a classifier of the next level;
5) repeat steps 3 and 4 for each of $log_2(n)$ levels, where *n* is number of data subsets.

It is obvious that while determining the arbiter (classifier) we take into account only a subset of the whole data and do not – the influence of the rest. So the classifier of the first level handles the context of a single subset only. The classifiers of the next levels are built over the union of subsets of the lower ones and hence the higher the classifier the more subset contexts it handles. This is analogues to the slide exam method considered above. The difference is that a group of samples

(not a single sample) are considered. We generalized both cases to handle the influence of the context on the classification result.

The inconsistencies of the competence maps and data are shown in the chapter 2. It is devoted to the way these problems could be resolved. Chapter 3 describes the approach of contextualization and decontextualization to handle the inconsistency of data.

2. The Problems of Multiple Databases & Multiple Methods Technique

Given n databases and some samples with the precise classification results (hereafter training set) for every database. There are also m methods that are applied over the training sets to get a classification result for any sample. The classification results of the methods may differ over the same training set. And, by the analogy, the classification results of the same method may differ over the training sets of several databases. The primary problem is to obtain a classification result for unknown sample using the classification results of the method(s) over all of the training sets. One can consider four possible combinations of applied methods and databases to be processed:

1) there is one method to be applied over one database. It is obvious that there are no conflicts in such case;

2) there are *m* methods to be applied over one database. The metamethod proposed in [1] was advanced in [3] to estimate an area of data where the classification method results in the most precise value. Such areas, where the method is the most appropriate among the others, are called competence areas of the method. As it was mentioned earlier the metamethod distinguishes just two cases of the classification results of the method: either correct or not. So it does not take into account the degree of the error the method makes. This may cause that the minimal error will lead to the exclusion of the method out of the consideration. So we propose to extent this discrete case in a following manner.

While slide exam over the training set of *k* samples one can evaluate values of errors the method makes: $(\varepsilon_1,\ldots, \varepsilon_k)$, where the value ε_i is an absolute error of the method applied to the sample *i* (or, in other words, a difference between the calculated value and the value of the training set). Let us denote the weight of the method over the sample *i* as ω_i. For the discrete case the weight ω_i should coincide with the value of matrix Q_{NxM} and be equal to 1 if $\varepsilon_i = 0$ and 0 – otherwise. For the continuos case the weight ω_i should satisfy the following conditions: $\omega_i \to 1$ if $\varepsilon_i \to 0$ and $\omega_i \to 0$ if $\varepsilon_i \to \pm\infty$. These conditions guarantees that the discrete weight is a special case of the continuos one. It is obvious that there exist some families of functions which map the error ε_i into the weight ω_i and satisfy the above conditions. We propose to use the following function

$$\omega = \frac{1}{1+\varepsilon^2} \tag{2.1}$$

as one of the simple to evaluate (in comparison, for example, with the function of normal distribution). To estimate the weight of method for unknown sample it is necessary, first of all, to interpolate the errors of the training set values for the given sample and then to evaluate the weight using this error obtained by the interpolation.

Let us denote the error function of the method *i* for the sample *x* as $\varepsilon^i(x)$. By the analogy the weight function is denoted like $\omega^i(x)$. The function $\omega^i(x)$ represents a competence map of the method *i*. The union of the competence maps of all methods forms a competence map of the database. An example of the competence map is shown on the Figure 2.1.

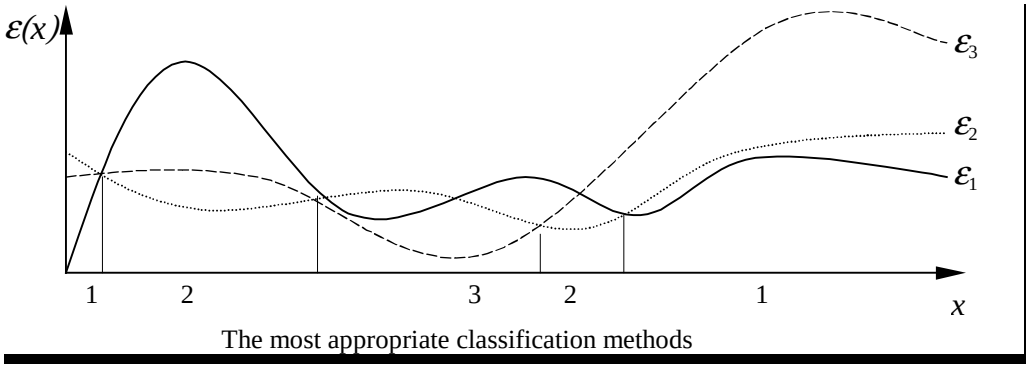

Figure 2.1 – An example of the competence map

If it is necessary to determine the most appropriate method for classification of the sample x then we have to evaluate functions $\omega^1(x)$, …, $\omega^m(x)$ and select the method with $\omega = \max \omega^i(x)$. Such selection guarantees that the error of the method is minimal, i.e. $\varepsilon = \min \varepsilon^i(x)$.

But if someone wants to determine the result value as a union of the results obtained by all of the available methods then it is possible to evaluate it as the following:

$$y = \frac{\sum_i y^i \omega^i}{\sum_i \omega^i}, \qquad (2.2);$$

where y^i – value obtained by the method i;
ω^i – the weight of the method i.

So the weight of the method over the sample allows either to evaluate the result value with the most appropriate classification method or to estimate the result using predictions of all available methods. And hence it is the way to resolve competence map inconsistency.

3) there is one method to be applied over n databases. Let us denote the classification result of the method for the sample x over the database j as $y_j(x)$. The weight of the method for the sample x over the database j is denoted like $\omega_j(x)$. Then an integral weight of the method over the database j we introduce as mean weight:

$$\nu_j = \frac{1}{b-a}\int_a^b \omega_j(x) \qquad (2.3)$$

where a, b – boundaries of database samples;
$\omega_j(x)$ – the weight of the method over the sample x of the database j;
ν_j – the integral weight of the method over the database j;
the integral sign means the sum of the weights of all samples of the database.

Such definition of the integral weight makes it possible to estimate the work of the same method over each database separately.

If someone needs to determine the result value as a union of the results obtained by the method over all available databases then it is possible to evaluate it as the following:

$$y = \frac{\sum_j y_j \nu_j}{\sum_j \nu_j}, \qquad (2.4);$$

where y_j – value obtained by the method over the sample x of the database j;
ν_j – the integral weight of the method over the database j.

Such approach for the evaluation of the result value allows to combine separate contradictory results obtained from several databases. So one can resolve data inconsistency conflicts using either the above formula to obtain certain mean result or the condition $\nu = \max \nu_j$ to select the most representative one.

4) there are *m* method to be applied over *n* databases. It is clear that this case is the combination of two previous cases and is the most general one.

Let us combine previous notations and denote the classification result of the method *i* applied over the database *j* for the sample *x* as $y_j^i(x)$. The weight of this value and the integral weight of the method are denoted like $\omega_j^i(x)$ and v_j^i accordingly.

The variants of application of the methods to the databases are shown in the table below.

	M_1	M_2	…	M_m	
DB_1	$y_1^1, \omega_1^1 (v_1^1)$	$y_1^2, \omega_1^2 (v_1^2)$	…	$y_1^m, \omega_1^m (v_1^m)$	y_1, ω_1
DB_2	$y_2^1, \omega_2^1 (v_2^1)$	$y_2^2, \omega_2^2 (v_2^2)$	…	$y_2^m, \omega_2^m (v_2^m)$	y_2, ω_2
…	…	…	…	…	…
DB_n	$y_n^1, \omega_n^1 (v_n^1)$	$y_n^2, \omega_n^2 (v_n^2)$	…	$y_n^m, \omega_n^m (v_n^m)$	y_n, ω_n
	y^1, ω^1	y^2, ω^2	…	y^m, ω^m	y, ω

Each row of the table describes the results and weights of all methods over single database. One can obtain the total classification result y_j and weight ω_j of the database j in accordance with the following formulae:

$$y_j = \frac{\sum_{i=1}^{m} y_j^i \omega_j^i}{\sum_{i=1}^{m} \omega_j^i}, \quad \omega_j = \frac{\sum_{i=1}^{m} \omega_j^i v_j^i}{\sum_{i=1}^{m} v_j^i}, \tag{2.5}$$

where y_j, ω_j – total classification result and weight of database *j* respectively;

y_j^i, ω_j^i, v_j^i – classification result, weight and integral weight of the method *i* over database *j* respectively.

Each column of the table describes the results and weights obtained by the same method through all databases. The classification result y^i and weight ω^i obtained by the method *i* over all databases are evaluated as following:

$$y^i = \frac{\sum_{j=1}^{n} y_j^i v_j^i}{\sum_{j=1}^{n} v_j^i}, \quad \omega^i = \frac{\sum_{j=1}^{n} \omega_j^i v_j^i}{\sum_{j=1}^{n} v_j^i}, \tag{2.6}$$

where y^i, ω^i – total classification result and weight evaluated by the method *i* over all databases respectively;

y_j^i, ω_j^i, v_j^i – classification result, weight and integral weight of the method *i* over database *j* respectively.

The result value *y* and weight ω in the bottom-left cell of the table is obtained as a combination of the classification results of all methods over all databases.

As one could notice, there are some variants of how to determine the total classification result and weight over all available databases using several methods. There are at least four of them:

1) to evaluate classification results y^i and weights ω^i, i=(1,m), using formulae (2.6) (i.e. through the columns of the matrix) and then determine the total result in accordance with the formula:

$$y = \frac{\sum_{i=1}^{m} y^i \omega^i}{\sum_{i=1}^{m} \omega^i} \tag{2.7};$$

2) to evaluate classification results y_j and weights ω_j, j=(1, n), using formulae (2.5) (i.e. through the rows of the matrix) and then determine the total result in accordance with the formula:

$$y = \frac{\sum_{j=1}^{n} y_j \omega_j}{\sum_{j=1}^{n} \omega_j} \qquad (2.8);$$

3) to evaluate the total result based on the values obtained by two previous methods (2.7, 2.8), for example as a mean value;
4) to evaluate the result through the whole matrix at once:

$$y = \frac{\sum_{i,j} y_j^i \omega_j^i}{\sum_{i,j} \omega_j^i} \qquad (2.9).$$

3. The Sliding Exam Technique with Decontextualization

Let us denote the value obtained while classification including all the examples of the training set as y^+, and the value obtained while classification excluding example i as y^{i-}. It is obvious that y^+ depends of the context of any training set example. And what is remarkable that the classification result y^{i-} obtained by applying the above evaluation procedure is beyond the influence of the example i. Consequently while compare y^{i-} with classification result y^+ it is possible to determine decontextualization trend of database values.

Decontextualization process of the values describing the trend of context decrease was proposed in [4] and can be evaluated as

$$y_{res} = \frac{y_i \cdot y_j}{y_i + y_j} \qquad (3.1)$$

The main property of (2.8) is the result decontextualized value y_{res} is less than the source values.

So knowing completely context–depending value y^+ and partially context–depending value y^{i-} at any point x we can find decontextualized value y' as:

$$y' = \frac{y^+ \cdot y^{i-}}{y^+ + y^{i-}}. \qquad (3.2)$$

To strengthen the result of decontextualization y' we should take into account all partially context–depending values y^i. In such case y' is like the following :

$$y^- = \frac{\prod_i y^{i-}}{\prod_i y^{i-} \cdot \sum_j \left(\frac{1}{y^{j-}} \right)} = \frac{1}{\sum_j \left(\frac{1}{y^{j-}} \right)}; \qquad (3.3)$$

$$y' = \frac{y^+ \cdot y^-}{y^+ + y^-} = \frac{y^+ \cdot \dfrac{1}{\sum_j \left(\dfrac{1}{y^{j-}} \right)}}{y^+ + \dfrac{1}{\sum_j \left(\dfrac{1}{y^{j-}} \right)}} = \frac{1}{\dfrac{1}{y^+} + \sum_j \left(\dfrac{1}{y^{j-}} \right)}. \quad (3.4)$$

Then we have obtained decontextualized value we are able to find the difference between the precise classification result *y* and decontextualized one:

$$\varepsilon = y - y' \quad (3.5)$$

So for each classification example y_i we could evaluate the difference $\varepsilon_i = y_i - y_i'$, $i=(1, k)$, where *k* is number of examples at the training set.

Such evaluation of differences enables the precision increase of the statistical method used at any point *x* and could be done by applying the following steps:

1) evaluation of decontextualized values y_i' and differences ε_i for every training set example *i*, $i=(1, k)$, where k is number of examples at the training set;
2) evaluation of decontextualized value $y'(x)$ at the point *x* as follows

$$y'(x) = \frac{1}{\dfrac{1}{y^+(x)} + \sum_j \left(\dfrac{1}{y^{j-}(x)} \right)}; \quad (3.6)$$

3) evaluation of difference $\varepsilon(x)$ at point *x* as the result of interpolation on the set of values $\varepsilon_1,\ldots,\varepsilon_k$, *k* where *k* is number of examples at the training set;
4) evaluation of $y(x)$ as $y(x) = y'(x) + \varepsilon(x)$.

The process of obtaining $y(x)$ by applying the difference $\varepsilon(x)$ to the decontextualized value $y'(x)$ is the contextualization process of last one.

So in the case of single database, $\varepsilon(x)$-function determine the context of this database. Let us call it as self-database difference or just self-difference. By analogy with the process of self-difference finding we can find inter-difference between two databases as follows

$$\varepsilon_{ij} = y_j - y_i', \quad (3.7)$$

where y_j – the precise value in the database *j*;

y_i' – decontextualized value in the database *i*;

ε_{ij} – mutual difference between databases.

Inter–difference ε_{ij} determine the context of database *j* in terms of database *i*. Hence based on the values of the database *i* it is possible to find them in the context of database *j:*

$$y_i(x) = y_j'(x) + \varepsilon_{ij}(x). \quad (3.8)$$

It is obvious that ε_{ii} is a special case of inter-difference and is the self–difference of database *i*.

4. Conclusion

This paper describes the approach of how to handle inconsistencies while multiple databases processing. We distinguish two types of inconsistencies: data inconsistency and inconsistency of competence map. We have considered four cases of applying methods over databases and pick out possible ways to determine the total classification result based on the results of all methods over each database. We have also considered deconstextualization process to increase the correctness of applying the classification method and to handle the mutual inconsistencies between databases.

References

[1] Terziyan V., Tsymbal A., Puuronen S., Decision Support Systems in Telemedicine Based on Multiple Expertise, In: Special Issue on “Computer Based Medical Systems”, International Journal of Medical Informatics, Elsevier, 1998 (to appear).
[2] P. Chan, Ac Extensible Meta-Learning Approach for Scalable and Accurate Inductivwe Learning, PhD Thesis, Columbia University, 1996
[3] Puuronen S., Terziyan V., Tsymbal A., Data Mining with Integration of Statistical Methods
[4] Terziyan V., Puuronen S., Kaikova H., A Decontextualization Method for Estimated Intervals