Technical Report

Remote Patient Monitoring System

(RPMS)

"A Formal Usability Constraints Model for Watermarking of Outsourced Datasets"

June, 2012

FAST National University of Computer & Emerging Sciences, Islamabad, Pakistan

Ownership-preserving Data Mining: A Formal Usability Constraints Model for Watermarking of Outsourced Datasets

M. Kamran and Muddassar Farooq

Abstract

The large datasets are being mined to extract hidden knowledge and patterns that assist decision makers in making effective, efficient and timely decisions in an ever increasing competitive world. This type of "knowledge-driven" data mining activity is not possible without sharing the "datasets" between their owners and data mining experts (or corporations); as a consequence, protecting ownership (by embedding watermark) on the datasets is becoming relevant. The most important challenge in watermarking (to be mined) datasets is: how to preserve knowledge in features or attributes?. Usually, an owner needs to manually define "Usability constraints" for each type of dataset to preserve the contained knowledge. The major contribution of this paper is a novel formal model that facilitates a data owner to define usability constraints – to preserve the knowledge contained in the dataset – in an automated fashion. The model aims at preserving "classification potential" of each feature and other major characteristics of datasets that play an important role during the mining process of data; as a result, learning statistics and decision making rules also remain intact. We have implemented our model and integrated it with a new watermark embedding algorithm to prove that the inserted watermark not only preserves the knowledge contained in a dataset but also significantly enhances watermark security compared with existing techniques. We have tested our model on 25 different data-mining datasets to show its efficacy, effectiveness and the ability to adapt and generalize.

Index Terms

Watermarking datasets, database watermarking, data mining, right protection, data usability, knowledge-preserving watermarking.

I. Introduction

THE large datasets – generated from very large databases – are being mined to extract hidden knowledge and patterns that are proving useful for decision makers to make effective, efficient and timely decisions in a competitive world. This type of "knowledge-driven" data mining expert systems cannot be designed and developed until the owner of data is willing to share (or outsource) the dataset with data mining experts (or corporations). Recently, a startup company – Kaggle (www.kaggle.com) – has made a business case out of this need where organizations outsource their datasets and the associated business challenge to data mining experts with an objective to find novel solutions to the posted problem [1]. This validates the thesis that corporations with large databases want to get the optimized solution to a problem by levering the power of crowd-sourcing.

In the emerging field of "sharing datasets" with the intended recipients, protecting ownership on the datasets is becoming a challenge in itself. Recently, an article reported the illegal sale of patients data and the concerned patients have sued the original hospital for breaching their privacy [2]. An even bigger concern is that the recipient may try to take credit for contribution towards knowledge discovery and data mining (KDD) by claiming the false ownership of the shared data. To mitigate these threats, a non-disclosure agreement is usually signed with the recipient binding him that he will not sale (or further share) the dataset and will also not claim the ownership of the data. If the recipient breaches the agreement, the legitimate data owner can only sue him if he can prove in a court of Law his ownership in an unambiguous manner over the dataset. Watermarking is the commonly used mechanism to enforce and prove ownership for the digital data in different formats like audio, video, image, relational database, text and software [3], [4], [5]. The most important challenge in watermarking data mining datasets[1] is: how to preserve knowledge in features or attributes during the embedding of watermark bits?

In order to preserve the knowledge in the dataset, one has to ensure that the predictive ability of a feature or an attribute is preserved; as a result, the classification results (and associated rules) remain preserved as well. In order to meet this requirement,

A shortened version of this work has been submitted for review for possible publication in IEEE Transactions on Information Forensics and Security.

M. Kamran is with Next Generation Intelligent Networks Research Center (nexGIN RC), Department of Computer Science, National University of Computer and Emerging Sciences, Islamabad, Pakistan.
E-mail: m.kamran@nexginrc.org

Muddassar Farooq is director Next Generation Intelligent Networks Research Center (nexGIN RC) and also with the Department of Electrical Engineering, National University of Computer and Emerging Sciences, Islamabad, Pakistan.
E-mail: muddassar.farooq@nu.edu.pk

[1]Throughout this paper – unless otherwise specified – the terms datasets, (to be mined) datasets, outsourced datasets, and data mining datasets have been used interchangeably.

an owner is supposed to define the "usability constraints" that provide the distortion band – within which the values of a feature can change – for each feature. As a result, the classification accuracy of the dataset remains unaltered. In addition to this, the inserted watermark should be imperceptible and robust against any type of sophisticated attacks that can be launched on the watermarked dataset. To conclude, defining "usability constraints" is a challenge because a user has to strike a balance between "robustness of watermark" and "preserving knowledge contained in features". For example, biomedical datasets may tolerate only very small amount of change – during the embedding of a watermark – in their features' set to preserve the diagnosis rules.

At the moment, the process of defining "usability constraints" is manually repeated and is dependent on the dataset and its intended application or use. Moreover, if right protection is enforced using "fingerprinting", the owner of data may need to define different "usability constraints" on the same dataset because in fingerprinting a different watermark for each user is added. To the best of our knowledge, no technique has been proposed to model the "usability constraints" while watermarking data mining datasets in particular, and other relational datasets in general. In this paper, we propose a novel formal model for identifying the essential "usability constraints" which must be enforced while embedding watermark in a dataset. The major contributions of our paper are:

- We propose a generic formal model to define "usability constraints" on a dataset that not only ensures the robustness of an inserted watermark but also preserves the knowledge contained in the dataset. The proposed technique is independent of the type of a dataset i.e numeric, non-numeric or strings.
- We have integrated our model in a new knowledge-preserving watermarking scheme to validate its efficacy and effectiveness.
- We show that the new knowledge-preserving watermarking scheme has significantly enhanced the security – (in terms of) deleting or changing the watermark – compared with existing techniques.
- We have conducted experiments on 25 publicly available datasets to prove that our technique can generalize to any type of dataset and achieves its objective of preserving the classification accuracy when mined with a machine learning classifier. We have evaluated our scheme on 5 well known machine learning classifiers – J48, SMO, Naive Bayes, IBk, and JRip – to show that their classification accuracy is preserved.

We briefly describe the related work in Section II and then introduce our approach in Section III. We present the formal model in Section IV. Subsequently, we explain that how the "usability constraints", defined by our model, can be given as input to a watermark embedding scheme. We report the results of our experiments in Section VI. Finally, we conclude the paper with an outlook to our future work.

II. Related Work

To the best of our knowledge, no technique has (so far) been proposed for modeling "usability constraints" for watermarking data mining datasets. In the work of Agrawal et al. [6], the first well known technique for watermarking numeric attributes in a database has been proposed. In this technique, message authenticated code (MAC) is calculated with the help of a secret key to identify the candidate tuples. Sion et al.[7] presented a marker tuples based watermarking technique for relational databases but these techniques are not applicable to data mining datasets because they do not aim at preserving the knowledge contained in the dataset.

Shehab et al. [8] proposed a partitioning based database watermarking technique. They modeled the process of watermark insertion as a constraint optimization problem and tested genetic algorithm (GA) and pattern search (PS) [9] optimizers. They select PS because it is able to optimize in realtime. But this technique requires defining "usability constraints" manually and does not account for preserving the knowledge contained in the data mining datasets.

Recently, in [10], we have proposed a relevant technique protecting ownership of electronic medical records (EMR) system. In this technique, information gain is used to identify the predictive ability of all features present in the EMR. The numeric feature(s) with the least predictive ability are selected to embed watermark bits to ensure information-preserving characteristic. This technique is only limited to information gain and does not generalize to other feature selection schemes. Moreover, it does not take into account certain characteristics of dataset that play a vital role in classification of the dataset. Since the major motivation of the technique is information-preserving watermarking; therefore, it does not describe any mechanism to model the "usability constraints". Moreover, this watermarking technique is limited to numeric features only.

In comparison, the focus of our current work is on developing a formal model to define "usability constraints" for watermarking of data mining datasets in such a way that the watermark is not only robust but the knowledge contained in the dataset is also preserved. Furthermore, we also provide a mechanism to logically group the dataset into groups such that high ranked features might also be watermarked during watermarking. This is a significant enhancement because if only low ranked features are watermarked, an attacker can launch malicious attacks on low ranked features only without compromising the data quality to a great extent. In this context, our data grouping approach enables a data owner to embed a watermark in high ranked features as well while still satisfying the "usability constraints" imposed by our formal model. Last but not the least, we have significantly enhanced our recently proposed information-preserving watermarking scheme [10] for data mining datasets in such a way that it can now watermark any type of features – numeric, non-numeric or strings.

III. Approach Overview

In this paper, we present two contributions: (1) a novel framework model which derives usability constraints for all kinds of datasets; and (2) a new watermarking technique that works for numeric, non-numeric and strings datasets. Our system takes the dataset as an input, models the "usability constraints" to be enforced during the watermark embedding in the dataset. Later it uses three different optimizers to find an optimum watermark that meets the relative constraints. The top level architecture of the proposed framework is shown in Figure 1.

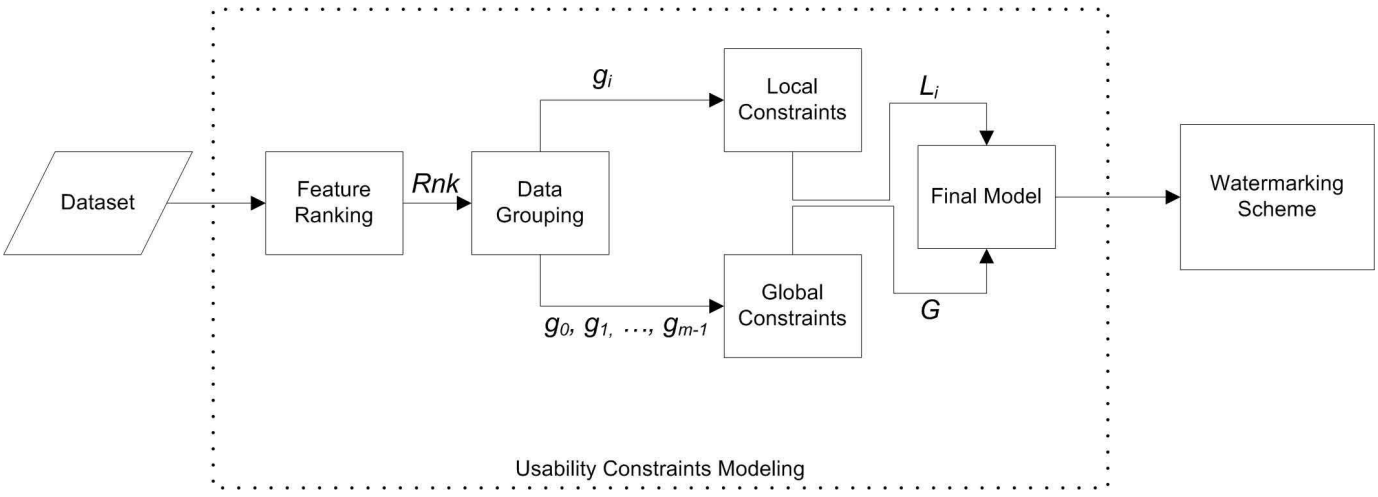

Fig. 1. The top level architecture of the proposed framework.

In the first step, the predictive ability of features, present in the dataset, are calculated and the features are ranked on the basis of computed predictive ability. Using these ranks, the next step is to generate the logical groups of features. In this step, "local usability constraints" are defined for each logical group. Similarly, the "global usability constraints" are also defined that are applicable for the whole dataset. Finally, both types of constraints are used to build a meta-constraints model that is given as an input to the watermarking scheme.

IV. A Formal Model for "Usability Constraints"

We now present our formal model to define "Usability Constraints" that preserve the knowledge during the process of inserting watermark in the dataset.

Definition 1. ***[Tuple.]*** *A tuple τ is an ordered list of elements.*

The tuple is used as a basic unit for referring different parameters of a dataset.

Definition 2. ***[Learning algorithm.]*** *Given a dataset D_O with M features, N instances, and a class attribute Y, a learning algorithm Γ, groups N instances into α different groups. Formally,*

$$\Gamma : D^M_{O_N} \rightarrow (C_\alpha, C_S) \tag{1}$$

where α is the number of distinct items in Y. A learning algorithm may be a classification algorithm or a clustering algorithm.

Definition 3. ***[Learning statistics.]*** *Learning statistics C_S is a tuple containing the classification statistics (or accuracy) of a particular learning algorithm. Formally:*

$$C_S \leftarrow \Gamma : D_O \tag{2}$$

These statistics include TP_{rate}, FP_{rate} and decision rule boundaries $\Re_b$. They are defined as:

$$TP_{rate} = \left(\frac{TP}{TP + FN}\right) \times 100 \tag{3}$$

$$FP_{rate} = \left(\frac{FP}{FP + TN}\right) \times 100 \tag{4}$$

Decision rule boundaries ($\Re_b$): $\Re_b$ denotes the threshold values that define the boundary of a particular decision rule.

TP (true positive): TP denotes the number of instances of a particular class correctly detected as instances of that class.

FP (false positive): For a particular class, the number of instances of other class(es) incorrectly detected as instances of that particular class.

TN (true negative): For a particular class, the number of instances correctly detected as instances of other class(es).

FN (false negative): For a particular class, the number of instances of that class incorrectly detected as instances of other class(es).

We store learning statistics in C_S, that is:

$$C_S = (TP_{rate}, FP_{rate}, \Re_b) \tag{5}$$

Definition 4. ***[Decision rules.]*** *Given a dataset D_O with M features, a rule r is a tuple constructed by mapping of m features, with $m \subseteq M$, based on C_S for identifying the class label y, where $y \epsilon Y$. $\Re$ contains all such rules as:*

$$\Re : (D_O, C_S) \rightarrow Y \tag{6}$$

Definition 5. ***[Feature selection scheme.]*** *A feature selection scheme S transforms M-dimensional data D_O, having N samples, M features and a class attribute Y, in m-dimensional space R^m (with $m \leq M$, such that $R^m \subseteq R^M$) that can yield "optimum" learning statistics. Formally,*

$$S : D_O^M \rightarrow R^m \tag{7}$$

In this paper, we have used 6 most commonly used different feature selection schemes (mutual information (I), information gain (IG), information gain ratio (IG_r), correlation based feature selection (CFS), consistency based feature subset evaluator (CBF), and principal components analysis (PCA)). All these feature selection schemes define the classification potential C_P of the features.

Axiom 1. *Different feature selection schemes can yield different learning statistics C_S even for the same learning algorithm Γ.*

Definition 6. ***[Mutual information.]*** *Let X and Y be two random variables, then their mutual information can be defined as:*

$$I(X;Y) = \sum_{y \epsilon Y} \sum_{x \epsilon X} p(x,y) log \left(\frac{p(x,y)}{p(x)p(y)} \right) \tag{8}$$

where $p(x,y)$ represents the joint probability distribution function of x and y, and $p(x)$ and $p(y)$ are marginal distributions of X and Y. Mutual information was introduced in [11]. In general, we denote the mutual information[2] $I(X;Y)$ (meaning the gain in information about X after observing Y) by $I(X)$.

Definition 7. ***[Watermark embedding.]*** *A watermark embedding is a transformation of a dataset D_O to D_W after embedding a watermark W.*

$$\Phi : (D_O, W) \rightarrow D_W \tag{9}$$

The dataset D_W is shared with an intended recipient, therefore; the information lost during the process of watermark embedding must be within allowed limits in order to preserve the knowledge contained in D_O.

Definitions 5, 6, and 7 provide the basis for formulating Theorem 1.

Theorem 1. *If for two datasets D_O and D_W each having N instances and M features, a feature $X_O \epsilon D_O$ has values tuple $x_O^N \epsilon X_O$ and a feature $X_W \epsilon D_W$ has values tuple $x_W^N \epsilon X_W$ then the classification potential of X_O and X_W will be the same if and only if the corresponding class label tuple for x_O^i and x_W^i is the same.*

Proof: Let Y_O and Y_W be the class label tuples for D_O and D_W respectively. Also suppose that the datasets D_O and D_W consist of only one feature X_O and X_W respectively. The percentage classification potential of X_O, can be calculated as:

$$C_{P_{X_O}} = \left(\frac{I(X_O)}{\sum_{i=1}^{M} I(X_{O_i})} \right) * 100 \tag{10}$$

Similarly, the percentage classification potential of X_W can be calculated as:

$$C_{P_{X_W}} = \left(\frac{I(X_W)}{\sum_{i=1}^{M} I(X_{W_i})} \right) * 100 \tag{11}$$

But classification potential of feature X_O is calculated using Definition 6 as:

$$I(X_O;Y_O) = \sum_{y_O \epsilon Y_O} \sum_{x_O \epsilon X_O} p(x_O, y_O) log \left(\frac{p(x_O, y_O)}{p(x_O)p(y_O)} \right) \tag{12}$$

[2]In this paper, unless otherwise specified, we use the terms predictive ability, mutual information, and classification potential interchangeably.

And for X_W:

$$I(X_W;Y_W) = \sum_{y_W \epsilon Y_W} \sum_{x_W \epsilon X_W} p(x_W, y_W) log\left(\frac{p(x_W, y_W)}{p(x_W)p(y_W)}\right) \tag{13}$$

Now, it is evident from equations (10), (11), (12), and (13) that $C_{P_{X_O}} = C_{P_{X_W}}$ if and only if for each tuple x_O^i and x_W^i, $y_O^i = y_W^i$.

Similarly, for datasets D_O^M and D_W^M, each consisting of M features, $C_{P_{X_O^k}} = C_{P_{X_W^k}}$ if and only if for each tuple $x_{k_O}^i$ and $x_{k_W}^i$, $y_{k_O}^i = y_{k_W}^i$, where $k = 1, 2, ..., M-1, M$. ■

So the watermarking process should not modify the value of a candidate feature in such a way that its class label is changed. This will ensure that the learning statistics of a learning algorithm are preserved.

For example, assume we have a dataset of pregnant women and we want to identify hypertensive patients. As a rule, a normal pregnant woman in between the age of 20 to 30 has a systolic BP less than 120; therefore, after watermarking, the BP of a normal pregnant woman must remain less than 120 to avoid misdiagnosis.

Definition 8. ***[Classification potential threshold.]*** *Given a dataset D_O with M features, the classification potential threshold (C_{P_T}) is a parameter for grouping of features – using γ as the secret grouping parameter that has a value between 0 and 1 – based on their classification potentials. It is calculated as:*

$$C_{P_T} = \gamma \times \frac{\sum_{i=1}^{M} I_i}{M} \tag{14}$$

This threshold is used to make different groups of a dataset so that the features with higher classification potentials are least modified during the process of watermarking. A data group g_i contains features possessing similar classification potential under the threshold C_{P_T}.

Definition 9. ***[Local usability Constraints.]*** *Local usability constraints L_i is a tuple constituting mutual information $I(X)$ of the feature X in a particular data group g_i.*

$$L_i = I(X) \tag{15}$$

The local usability constraints are used to watermark features in a group g_i and they are enforced at a group level only.

Definition 10. ***[Global usability Constraints.]*** *Given a dataset, global usability constraints G is a tuple that consists of features' set produced by different feature selection schemes on that dataset.*

In our case we are using 5 feature selection schemes – IG, (IG_r), (CFS), (CBF), and (PCA).

$$G = \left(R_{IG}^m, R_{IG_r}^m, R_{CFS}^m, R_{CBF}^m, R_{PCA}^m\right) \tag{16}$$

Global usability constraints are enforced both at a group level and at the global dataset level. The features' set, produced by applying a feature selection scheme to a group or a dataset, should remain unaltered.

The local usability constraints, global usability constraints and learning statistics lay the foundation for Lemma 1.

Lemma 1. *If for two datasets D_O and D_W, a feature selection schemes S yields tuples R_O^m and R_W^m such that $R_O^m \neq R_W^m$, then learning statistics C_{S_O} and C_{S_W} obtained by applying learning algorithms on R_O^m and R_W^m will not be same.*

Proof: Let $C_{S_{O1}}$ and $C_{S_{W1}}$ be the learning statistics of a learning algorithm Γ_1 after working on R_O^m and R_W^m respectively.

Now, as the results of the two feature selection schemes yielded R_O^m and R_W^m such that $R_O^m \neq R_W^m$; so according to our Definition 5 the "optimum" results of classification would be different for R_O^m and R_W^m; hence we can safely conclude $C_{S_{O1}} \neq C_{S_{W1}}$. Therefore,

$$R_O^m \neq R_W^m \Rightarrow C_{S_{O1}} \neq C_{S_{W1}} \tag{17}$$

■

So, we conclude from Lemma 1 that for two datasets D_O and D_W, a feature selection scheme S should yield the same features' set so that the learning statistics of an algorithm are preserved. The usability constraints on feature selection schemes are defined using Lemma 1.

Definition 11. ***[Data separability.]*** *Given a dataset D_O, data separability is division of dataset D_O into α sub-datasets such that:*

$$D_{O_1} \cap D_{O_2} \cap ... \cap D_{O_\alpha} = () \tag{18}$$

and

$$D_{O_1} \cup D_{O_2} \cup ... \cup D_{O_\alpha} = D_O \tag{19}$$

Axiom 2. *A dataset can be classified into α classes if and only if it is $(\alpha - 1)$ dimension separable.*

The data separability helps to define the rule boundary of a decision.

Definition 12. ***[Data distribution.]*** *Given a feature $X \epsilon D_O$, a data distribution Ω is the number of occurrences of certain data values $x_i \epsilon X$.*

$$\Omega \leftarrow (\forall i | H(x_i)) \tag{20}$$

where H represents a data distribution function.

An ideal watermarking algorithm should preserve the data distribution of original dataset.

Lemma 2. *If for two datasets D_O and D_W, the data distributions Ω_O and Ω_W are such that $\Omega_O \neq \Omega_W$, then the learning statistics C_{S_O} and C_{S_W} of datasets D_O and D_W respectively, will not be the same.*

Proof: Suppose a dataset D_O has a class attribute Y with α possible values, then D_O can be classified correctly if it is $(\alpha - 1)$ dimension separable. Now, consider $\alpha = 2$, that is, D_O has two class labels c_1 and c_2. According to Axiom 2 learning algorithm Γ can be applied on D_O to yield learning statistics C_{S_O}, if and only if D_O is 1 dimension separable into sub-datasets D_{O_1} and D_{O_2} (Definition 11) such that:

$$D_{O_1} \cap D_{O_2} = ()$$

and

$$D_{O_1} \cup D_{O_2} = D_O$$

But this is possible if data distribution Ω_O is such that a *boundary point* separates D_{O_1} from D_{O_2}.
Similarly for dataset D_W,

$$D_{W_1} \cap D_{W_2} = ()$$

and

$$D_{W_1} \cup D_{W_2} = D_W$$

Again, this is possible if data distribution Ω_W is such that a *boundary point* separates D_{W_1} from D_{W_2}. So, if position of this *boundary point* is different from the position of *boundary point* of D_{O_1} and D_{O_2}, then elements of D_{W_1} will be different from D_{O_1} and similarly elements of D_{W_2} will be different from D_{O_2}; and as a consequence, learning statistics $C_{S_W} \neq C_{S_O}$.
Now, if $\alpha > 2$, then learning algorithm Γ can be applied on D_O to yield learning statistics C_{S_O}, if and only if D_O is $(\alpha - 1)$ dimension separable into sub-datasets $D_{O_1}, D_{O_2}, ..., D_{O_\alpha}$ such that:

$$D_{O_1} \cap D_{O_2} \cap ... \cap D_{O_\alpha} = () \ \ and \ \ D_{O_1} \cup D_{O_2} \cup ... \cup D_{O_\alpha} = D_O$$

Similarly for D_W, the learning algorithm Γ can be applied to yield learning statistics C_{S_W}, if and only if D_W is $(\alpha - 1)$ dimension separable into sub-datasets $D_{W_1}, D_{W_2}, ..., D_{W_\alpha}$ such that:

$$D_{W_1} \cap D_{W_2} \cap ... \cap D_{W_\alpha} = () \ \ and \ \ D_{W_1} \cup D_{W_2} \cup ... \cup D_{W_\alpha} = D_W$$

But this is possible if data distribution Ω is such that $(\alpha - 1)$ *boundary points* separate $D_{O_1}, D_{O_2}, ..., D_{O_\alpha}$ from each other. So, if position of these *boundary points* is different for $D_{O_1}, D_{O_2}, ..., D_{O_\alpha}$ and $D_{W_1}, D_{W_2}, ..., D_{W_\alpha}$; then elements of $D_{O_1}, D_{O_2}, ..., D_{O_\alpha}$ and $D_{W_1}, D_{W_2}, ..., D_{W_\alpha}$ will not be same; and as a consequence, learning statistics $C_{S_O} \neq C_{S_W}$.
Hence, for two datasets D_O and D_W, learning statistics C_{S_O} and C_{S_W} produced by learning algorithm Γ will be different if the data distributions Ω_O and Ω_W of datasets D_O and D_W are not the same. ■

Definition 13. ***[Learning information loss.]*** *If C_{S_O} represent the learning statistics tuple obtained by applying a learning algorithm Γ on dataset D_O, and C_{S_W} represent the learning statistics' tuple obtained by applying a learning algorithm Γ on dataset D_W, then learning information loss $C_{S_{O_{Loss}}}$ is defined as:*

$$C_{S_{Loss}} = \frac{|C_{S_O} - C_{S_W}|}{C_{S_O}} * 100 \tag{21}$$

We use $C_{S_{Loss}}$ to quantify the amount of information lost during the process of inserting watermark in a dataset.

Definition 14. ***[Knowledge-preserving watermarking.]*** *Given two datasets D_O and D_W, the watermarking scheme will be knowledge-preserving if and only if learning statistics $C_{S_O} = C_{S_W}$, that is:*

$$C_{S_{Loss}} = 0 \tag{22}$$

The knowledge-preserving and lossless usability constraints model is derived from Theorem 1, Lemma 1, and Lemma 2 as follows.

Definition 15. ***[Knowledge-preserving and lossless usability constraints model.]*** *The usability constraints model $\hbar$ for knowledge-preserving and lossless watermarking of data mining datasets is a tuple meeting following constraints:*

$$Y_{D_O} = Y_{D_W}; C_{P_{D_O}} = C_{P_{D_W}};\ and\ \Omega_{D_O} = \Omega_{D_W} \tag{23}$$

The second constraint in (23) needs to be satisfied by using local constraints (Definition 9) and global constraints (Definition 10); as a result, the output of feature selection schemes must be same for both datasets D_O and D_W.

Theorem 2. *If a watermarking scheme meets the constraints in $\hbar$ only then it is defined as knowledge-preserving and lossless.*

Proof: If C_{S_O} and C_{S_W} are the learning statistics obtained by applying a learning algorithm Γ on original dataset D_O, and watermarked dataset D_W respectively.

Since, class labels, classification potentials and data distributions are preserved during watermarking; so according to Theorem 1, Lemma 1, and Lemma 2 the statistics C_{S_O} and C_{S_W} would be same, that is:

$$C_{S_{D_O}} = C_{S_{D_W}} \tag{24}$$

But the learning statistics tuple $C_{S_{D_O}}$ is formulated as:

$$C_{S_{D_O}} = (TP_{rate_O}, FP_{rate_O}, \Re_{b_O}) \tag{25}$$

and learning statistics tuple $C_{S_{D_W}}$ as:

$$C_{S_{D_W}} = (TP_{rate_W}, FP_{rate_W}, \Re_{b_W}) \tag{26}$$

Now, according to definition of learning information loss:

$$C_{S_{Loss}} = \frac{|C_S - C_{S_W}|}{C_S} * 100$$

So,

$$C_{S_{Loss}} = \frac{\left(|C_{S_{Loss_1}}|\right)\left(|C_{S_{Loss_2}}|\right)\left(|C_{S_{Loss_3}}|\right)}{C_S} * 100 \tag{27}$$

where

$$\begin{aligned} C_{S_{Loss_1}} &= TP_{rate_O} - TP_{rate_W}; \\ C_{S_{Loss_2}} &= FP_{rate_O} - FP_{rate_W};\ and \\ C_{S_{Loss_3}} &= \Re_{b_O} - \Re_{b_W} \end{aligned} \tag{28}$$

By substituting equations (24), (25), (26), and (28) in equation (27), we get:

$$\begin{aligned} C_{S_{Loss}} &= \frac{(0)\,(0)\,(0)}{C_S} * 100 \\ C_{S_{Loss}} &= \frac{0}{C_S} = 0 \end{aligned}$$

■

To conclude, the usability constraints, defined for the process of watermarking, must hold this theorem to ensure knowledge-preserving and lossless watermarking. We now discuss watermark embedding scheme in the following.

V. Watermarking Scheme

We now describe our watermarking scheme – with its foundation in the above-mentioned formal model – that not only preserves the classification potential of features but also results in (approximately) zero information loss. There are two main phases in our watermarking scheme: watermark encoding and watermark decoding.

A. Watermarking Encoding

The steps involved in the watermark encoding phase are:

Step 1: The classification potential of each feature is calculated using mutual information I (Definition 6) and it is stored in a vector Rnk. The threshold C_{P_T} (Definition 8) is computed using a vector of classification potentials. The classification potential of features (the vector Rnk) and C_{P_T} are then used to logically group features of the dataset into n non-overlapping groups $g_0, g_1, ..., g_{n-1}$.

Step 2: The watermark is optimized and embedded in this stage while enforcing the usability constraints modeled in Section IV.

The different steps of watermark encoding phase are shown in Figure 2.

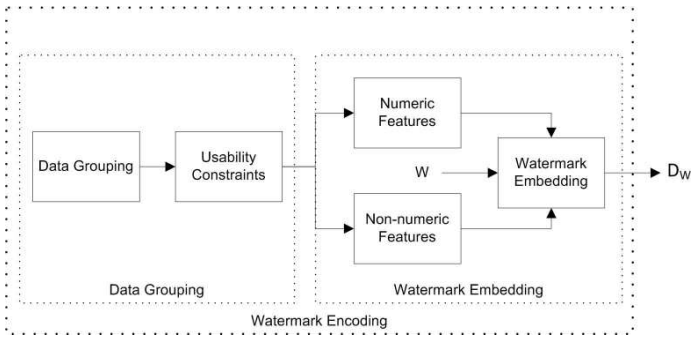

Fig. 2. Different steps of watermark encoding phase.

1) Feature Ranking: In this step, the features are ranked to: (1) logically group the data into n non-overlapping partitions; and (2) to define "usability constraints" in such a manner that the information loss is zero. The ranking is done, using a well known information measure, mutual information I, to understand the correlation of a feature on predicting a class label. The rank of all features, present in a dataset, are stored in a vector Rnk.

2) Classification potential threshold computation: Intuitively speaking, a feature that has a large classification potential is expected to tolerate only a small change (during embedding of watermark) in its values to ensure that the decision rules remain unaltered. In Figure 3, the tolerable alteration Δ in the value of a feature is plotted against its classification potential if the learning statistics were to be preserved as proved in [10]. The figure also concurs to our expectations: (1) the features with high classification potential can tolerate only small changes, if the information is to be preserved during watermarking; and (2) the top ranked features show approximately zero tolerance towards any change. Therefore, it is very important to compute the amount of change that a feature can tolerate during the watermarking process. The data groups are constructed using C_{P_T} and "tolerable alteration" is computed for each group.

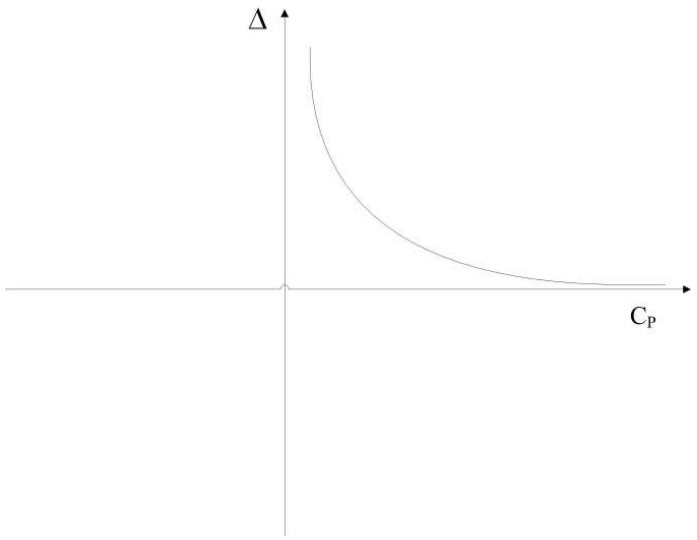

Fig. 3. Plot for C_P versus Δ.

3) Data Grouping: As mentioned before, C_{P_T} is used to group the features into n logical non-overlapping groups. The data grouping function is given in equation (29).

$$\begin{cases} insert\ \ f\ \ into\ \ g_0, & if\ \ Rnk_f = 0 \\ insert\ \ f\ \ into\ \ g_n, & if\ \ Rnk_f \geq Avg \\ insert\ \ f\ \ into\ \ g_i, & if\ \ Rnk_f > 0 \\ & and\ \ Rnk_f \geq i * C_{P_T} \\ insert\ \ f\ \ into\ \ g_{n-2}, & otherwise \end{cases} \tag{29}$$

where, $i = 1, 2, ..., n-3$; and

$$Avg = \frac{\sum_{u=1}^{n} I_u}{n} \tag{30}$$

The grouping function is applied on every feature of a given dataset. A group might be empty but then will be omitted during the optimization phase. Remember that the groups are only logical and hence cannot be truly separated from one another. Our pilot studies show that the proposed group assignment algorithm is simple yet effective. We use the groups to define the local and global "usability constraints". Our data grouping approach overcomes a significant shortcoming of our earlier work [10] in which only the lowest ranked features are watermarked. In the new approach, an attacker cannot easily build an attack vector by filtering low ranking features only.

4) Refined Usability Constraints: In this paper, the data usability constraints are defined by "knowledge-preserving and lossless usability constraints model" (Definition 15) presented in equation (23). In order to easily enforce constraints, we refine the constraint number 2 into two types: (1) global constraints G for the whole dataset; and (2) local L_i for a particular logical group g_i.

Local Usability Constraints. The local usability constraints (Definition 9) are defined by mutual information (see equation (15)). In order to enforce them, the constraint 2 of "knowledge-preserving and lossless usability constraints model" must be met. In our formal model, we preserve the information by trying to keep the data distribution (as much as possible) unaltered. As a result, we have to model the challenge as an optimization problem to ensure that under given constraints, the "tolerable alteration" is maximized for all features in general and minimized for high ranking features within each group while watermarking a group g_i.

Global Usability Constraints. In global usability constraints G (Definition 10), we have to ensure that the predictive ability of each feature – calculated by five well known feature selection schemes (IG, IG_r, CFS, CBF , and PC) remains preserved, that is ($R^m_{IG^W} = R^m_{IG^O}$, $R^m_{IG^W_r} = R^m_{IG^O_r}$, $R^m_{CFS^W} = R^m_{CFS^O}$, $R^m_{CBF^W} = R^m_{CBF^O}$, and $R^m_{PC^W} = R^m_{PC^O}$.

5) Selecting Data for Watermarking: An important step in watermarking of a dataset is to select relevant rows in which the watermark will be inserted. In this paper, we use a parameter ζ to store information about such rows. Its main purpose is to insert the watermark in the rows of the original data D_O in such a way that local and global usability constraints are not violated.

6) Watermark Embedding: **Watermarking non-numeric features.** Data grouping step is not performed for non-numeric features because our watermark embedding algorithm does not bring any change in the values of such features. To embed a watermark in the dataset, a sequence of binary bits is used as a watermark. For watermarking a non-numeric feature f, secret hash value for each row is calculated by seeding a pseudo random sequence generator $\eth$ with concatenation of a secret key K_s, class label of the row, and row value (ascii) using equation (31)

$$row.hash = \eth(K_s||y||row.val) \tag{31}$$

where, $row.val$ denotes a particular row value and y is the class label of that row.

The secret ordering of rows is computed using the watermark bits as depicted in Algorithm 1. This secret order does not bring any change in the underlying dataset. If a row value is repeated with the same class label then the same hash value would always be generated; consequently such rows are logically placed in the same cluster. The secret order of hash values after embedding the final bit is stored to use it during the watermark decoding stage.

Watermarking numeric features. We model the maximizing "tolerable alteration" as a constrained optimization problem subject to certain constraints on "tolerable alterations". We have significantly enhanced the information preserving watermarking technique of [10], using our formal model, to make it a generalized usability constraints modeling and knowledge-preserving watermarking scheme.

The lower and upper bound (denoted by Δ^f_{min} and Δ^f_{max}) for a feature f in the i^{th} group, are calculated using equations (32) and (33) respectively to define "tolerable alterations" for a data element.

$$\Delta^f_{min} = \frac{\left(\frac{1}{1+Rnk^f}\right) * \frac{\gamma}{(i+1)}}{i^2} \tag{32}$$

$$\Delta^f_{max} = \frac{\frac{\left(\frac{1}{1+Rnk^f}\right)}{\frac{\gamma}{(i+1)}}}{i^2} \tag{33}$$

Algorithm 1 Watermark non-numeric features

Input: D_O, K_s, W
Output: D_W, $temp$
  $temp \leftarrow D_O$
  for each row in D_O **do**
    $row.hash = \eth(K_s||y||row.val)$
    $temp(row) = D_O(row.hash)$
  end for
  for each bit b in W **do**
    if $b == 1$ **then**
      sort the rows in the descending order of hash values
    else
      sort the rows in the ascending order of hash values
    end if
    $D_W \leftarrow temp$
  end for
  for each row in D_W **do**
    $row.hash = \eth(K_s||y||row.val||)$
    $temp(row) = D_W(row.hash)$
  end for
  return D_W, $temp$

where, Rnk^f is the classification potential of the feature f and γ is the secret grouping parameter. Note that the relations for computing Δ^f_{min} and Δ^f_{max} are significantly different from the relations used in [10]. We incorporate the secret parameter γ with the objective of having more watermark security. Also note that in the equations (32) and (33), 1 is added in denominator to avoid division by zero if a feature has zero classification potential.

In our approach, the usability constraints given in equation (23) are enforced while maximizing the allowable alterations Δ_i in a group g_i. This ensures that both local (within a group) as well as global (across the whole dataset) constraints are satisfied.

To conclude, we use equation (34) to define a new objective function: to maximize Δ_i under user defined "tolerable alterations"($\Delta_{min_i} \leq \Delta_i \leq \Delta_{max_i}$).

The process of optimizing Δ_i follows two simultaneous steps: (1) In step 1, the constraints are verified locally for each logical group in an iterative manner; and (2) In step 2, the global usability constraints are also verified for the complete dataset. The objective function used for each type of optimization scheme is the same.

$$\begin{aligned} &\max\left(g_i + \Delta_i\right) \\ &subject\ \ to \\ &\hbar\ \ with\ \ L_i\ \ and\ \ G \end{aligned} \tag{34}$$

Note that in our objective function g_i is the part of data D_O; therefore, the only variable to be optimized is Δ_i. As mentioned before, for the features having high classification potential, a very small "tolerable alteration" is allowed; therefore, the value of Δ_i – for the groups containing such features – should be zero subject to meeting the constraints mentioned in the objective function of equation (34). Therefore, if Δ_{min} for such features violates any usability constraints then Δ_{min} is set to zero for those features. This situation is also handled during the optimization process. In the following, we evaluate the suitability of three optimization schemes to optimize the objective function mentioned in equation (34).

Our next target is to select an optimizer[3] that has the ability to locate the global optimum in the optimization search space in realtime. We have used two biological heuristic based optimizers – Particle Swarm Optimization (PS0) and Genetic Algorithms (GA) – and one classical optimizer – Mixed Integer Nonlinear programming.

Optimization with Particle Swarm Optimization Algorithm. Particle swarm optimization (PSO) [12] is a stochastic technique developed for continuous optimization using social behavior of flocking birds. The solutions to the problem are modeled as particles. PSO has been quite efficiently and effectively implemented for function optimization, artificial neural network training, fuzzy system control, and other areas. Moreover, it is proven to be better in many respects from evolutionary algorithms like genetic algorithms, memetic algorithms, ant-colony systems, and shuffled frog leaping [13].

In our implementation of PSO, we map the statistics contained in Δ_i using bit string having length l (the length of the watermark) as depicted in Figure 4. As our particle consists of 0 and 1 bits; therefore, our technique performs the different data alterations for each distinct bit. The used fitness function is given in equation (34).

Optimization with Genetic Algorithm. Genetic algorithm (GA) is inspired from the principles of biological evolution: survival of the fittest. The populations of candidate solutions – called chromosomes – compete to evolve to become better solutions. GA is heuristic based method and was introduced by John Holland [14].

We have used GA to create the watermark (chromosome) for embedding it into the partitioned dataset. In our implementation of GA, the chromosome consists of mapping of feasible Δ_is into a bit string consisting of 0s and 1s having length l. It uses

[3]We use the watermark optimization step for numeric features only because Δ is not required for non-numeric features in the proposed technique.

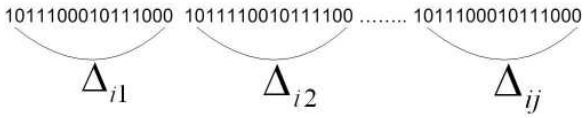

Fig. 4. PSO particle representation.

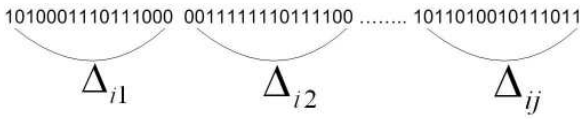

Fig. 5. GA chromosome representation.

the same fitness function as that of PSO given in equation (34). The representation of the chromosome (watermark) is given in Figure 5.

When a chromosome violates the usability constraints, it is penalized by decreasing its fitness value. The penalty function is given as:

$$\partial_{new} = \partial_{old} * \frac{1}{1+\vartheta} \tag{35}$$

where ∂_{new} and ∂_{old} are the new and old fitness values of the chromosome respectively, and ϑ is the number of violations of the usability constraints. If no violations occur for a chromosome, the fitness value (according to equation (35)) of the chromosome remains unchanged. Moreover, as the value of ϑ increases the fitness of chromosome decreases. The feasible solution set ρ contains all the chromosomes which satisfy the conditions specified in the objective function. Our chromosome representation consists of 0 and 1 bits so here again, our technique performs the different data alterations for each distinct bit. The algorithm stops once the termination criteria is met: (1) when a chromosome is found with the maximum allowable value of Δ_i; or (2) Δ_i does not further increase in a predefined number of iterations; or (3) when the predefined number of generations have been evaluated.

Optimization with Mixed Integer Nonlinear Programming. Mixed Integer Nonlinear Programming (MINLP) is a mathematical programming technique that uses continuous and discrete variables, and nonlinear objective function and constraints. MINLPs have been applied for optimizing constrained problems with applications in engineering, finance, and other scientific problems. There are several different approaches for solving MINLPs including but not limited to: Outer Approximation (OA) methods, Branch-and-Bound (B&B), Extended Cutting Plane methods, and Generalized Benders Decomposition (GBD) [15]. The motivation to use MINLP in our scheme came from its use in designing algorithms for constrained combinatorial non-linear mathematical problems in various disciplines. In this paper, we work with the OA method proposed by Fletcher et al. in [16]. OA divides MINLP into NLP subproblem and a master Mixed Integer Program (MIP). An interested reader is referred to [17] for a survey of different MINLP techniques and their possible applications.

We have used MINLP implemented in LINDO [18] software for optimizing our non linear objective function given in equation (34). We have incorporated our "usability constraint model" in LINDO.

The optimum value of Δ_i for a particular data group g_i is embedded in each row of the group subject to usability constraints as specified in the objective function given above. The numeric features in a group g_i are marked with positive Δ_i, if a watermark bit b is 1; and with negative Δ_i if a watermark bit b is 0. Algorithm 2 lists the steps of watermark embedding in numeric features.

7) Watermark Security: As discussed before, the current scheme further improves on the robustness level of our earlier technique [10] by using the data-grouping strategy. For brevity, we compute the probability of successfully attacking one watermark bit only. Let $Prob$ be the probability of successfully attacking one row of the data having n logical groups. Since, the attacker is not aware of the secret parameters that were used during the data-grouping stage; therefore, he cannot intentionally target a particular group for launching different types of attacks. As a result, he will have to select a random feature to launch his attacks. In this case, the probability of successfully corrupting (or deleting) a watermark bit is $(0.5)^n$ because different watermark is inserted in different groups. Since, we are watermarking all rows in the data and also using the majority voting as an error correction measure; therefore, the attacker has to target at least half the total number of rows to achieve his objective. If there are N rows in the dataset, the probability of successfully corrupting a watermark bit is $((0.5)^n)^{\frac{N}{2}}$. For large datasets and large values of n, this probability becomes significantly small. Consider, for example, a data has 100 rows and $n = 4$. The probability of successful launching an attack on this data is:

Algorithm 2 Watermark numeric features

Input: D_O, W
Output: D_W, Δ
construct data groups g_n using D_O
for each bit b in W **do**
  for each group g_i in g_n **do**
    for each row in g_i **do**
      if $b == 1$ **then**
        $D_W(row) \Leftarrow g_i(row) + \Delta_i(row)$ subject to $\hbar$ with L_i and G
      else
        $D_W(row) \Leftarrow g_i(row) + (-\Delta_i(row))$ subject to $\hbar$ with L_i and G
      end if
    end for
  end for
end for
return D_W, Δ

$$Prob = ((0.5)^4)^{50} = (0.0625)^{50} = 6.22 \times 10^{-61}$$

This probability in our earlier technique [10] is 8.88×10^{-16}; therefore, the new technique has significantly reduced the proability of deleting a watermark from the dataset.

B. Watermark Decoding

The architecture of the watermark decoding phase is shown in Figure 6.

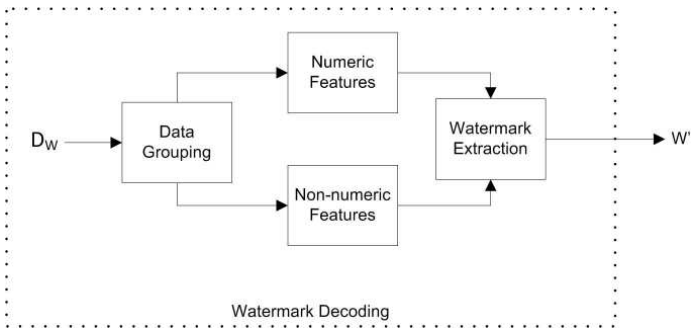

Fig. 6. Steps of watermark decoding phase.

1) Watermark decoding from non-numeric features:: In the watermark detection phase, the hash value of a feature for each row is calculated using the same steps of watermark embedding. The secret ordering, on the basis of this hash value is calculated by comparing it with the ordering based on the hash values in $temp$ and the embedded bit is decoded from this ordering. It is important to mention here that the sole purpose of using $temp$ for ordering the rows during watermark decoding stage is to combat attacks which may change the order of rows in the watermarked data. Since, the $temp$ contains the ordering after embedding the last watermark bit b; therefore, the last embedded bit is decoded first. The steps of this phase are listed in Algorithm 3 that are repeated for each bit of watermark W.

2) Watermark decoding from numeric features:: To decode the watermark from numeric features, a decoding threshold T^* value is calculated for each group by using the same method as in [10]. For decoding a watermark from a numeric feature in a group g_{i_W}, a parameter val is computed from the watermarked dataset D_W as:

$$val = \Delta i * g_{i_W} \tag{36}$$

The value of the parameter val is compared with the decoding threshold T^* and if val is found to be greater than T^*, the watermark bit b is decoded as 1; otherwise it is decoded as 0. Here, as an error correction mechanism, a majority voting step is performed after decoding a watermark bit b from all the rows. The steps of this phase are shown in Algorithm 4.

VI. Experiments and Results

We have performed our experiments on 25 different datasets[4]. These biomedical and biomedicine datasets are carefully chosen from different domains so that we test our technique for two class datasets, multi-class datasets, high dimensional datasets, datasets with missing values, datasets with various type of features, imbalanced datasets, and datasets with large

[4]These datasets are available online at www.ics.uci.edu/mlearn/MLRepository.html and http://home.ccr.cancer.gov/ncifdaproteomics/ppatterns.asp

Algorithm 3 Watermark decoding from non-numeric features

Input: D_W, K_s, $temp$
Output: $W^{'}$
  $temp^{'} \leftarrow D_W$
  for each row in D_W **do**
    $row.hash = \eth(K_s||y||row.val)$
    $temp^{'}(row) = D_W(row.hash)$
  end for
  compute secret order using $row.hash$
  update the order of rows in $temp^{'}$ is according to their order in $temp$
  for $j =$ length of W to 1 **do**
    if the rows are sorted in the descending order of hash values **then**
      $b = 1$
    else
      $b = 0$
    end if
    if $b == 1$ **then**
      sort the rows in the ascending order of hash values
    else
      sort the rows in the descending order of hash values
    end if
    $W^{'} \leftarrow b$
  end for
  return $W^{'}$

Algorithm 4 Watermark decoding from numeric features

Input: D_W, $T*$
Output: Decoded watermark $W^{'}$
  construct data groups g_n using D_W
  for $j =$ length of W to 1 **do**
    for each group g_i in g_n **do**
      for each row in g_i **do**
        compute val using equation (36)
        if $val > T^*$ **then**
          $b = 1$
        else
          $b = 0$
        end if
      end for
    end for
    perform majority voting to finally decode b
    $W^{'}_j \leftarrow b$
  end for
  return $W^{'}$

number of instances. We do not report the results of our study on the robustness of proposed scheme in this paper because this is done elsewhere in [10] but our current approach further improves the robustness level using the data grouping strategy discussed in Section V.

The experiments were carried out on a computer with a 1.73 Core 2 processor and a 1 GB of RAM. We set the value of $\zeta = 100\%$, which means that all rows were selected for watermarking. A data owner can choose any watermark length but we set its length to $l = 16$ bits. Five well known machine learning schemes are used to analyze their learning statistics on both original and altered datasets to show the relevance of Theorem 2 in the proposed scheme.

A. Working with non-numeric features

The example run of watermark encoding and decoding for non-numeric features is reported in [19].

For brevity, we report the test run of watermark embedding and decoding algorithms on non-numeric features in only six rows of the Protein dataset in Figure 7.

Since the proposed watermark embedding algorithm does not bring any change in the data; therefore, the results of feature selection schemes and all the learning statistics will definitely be the same for original and watermarked data. Hence, our model of Definition 15 is validated and the proposed watermarking scheme for non-numeric datasets is knowledge-preserving & lossless (Theorem 2).

B. Working with numeric features

For numeric features, we have used all 3 optimization schemes on 24[5] datasets for optimizing the value of Δ. We have used two metrics to benchmark optimized techniques: (1) the time taken to find the optimum Δ; and (2) the optimized value of Δ that models the quality of the solution. The output of the optimization techniques – utilizing our formal model – is given as an input to the above-mentioned watermarking scheme. It is important to emphasize here that the proposed usability constraints

[5]We omitted Protein dataset from this experiment because it does not have any numeric feature.

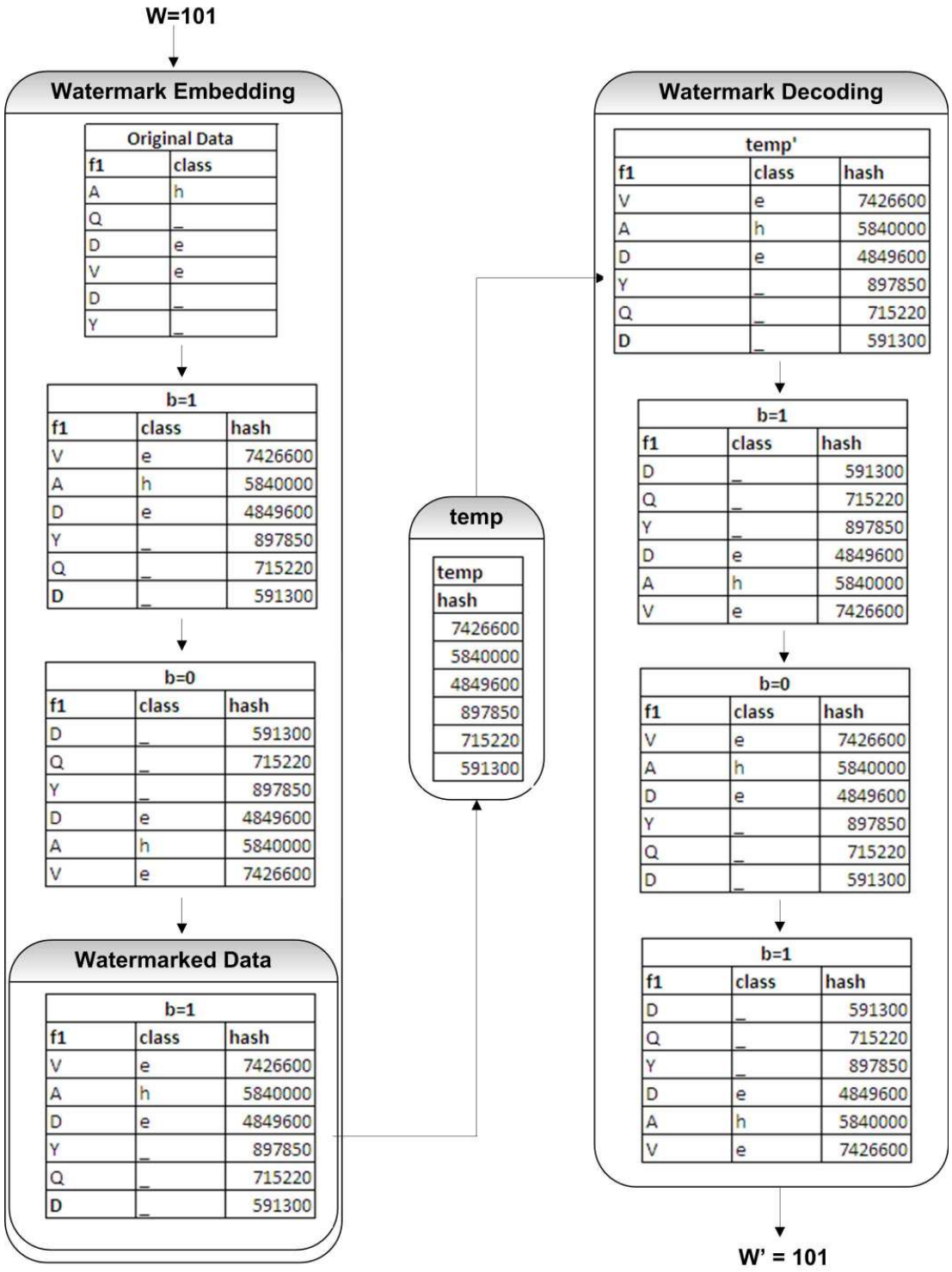

Fig. 7. Watermark encoding and decoding process for non-numeric features. Note that in the watermark embedding process, if a watermark bit b is 1 then the rows of the dataset are sorted in the descending order of hash values and if b is 0 then the rows are sorted in the ascending order. In comparison, during watermark decoding process, if rows are sorted in the descending order then the bit b is decoded as 1 and the order of rows is updated in the ascending order of hash values; otherwise, b is decoded as 0 and the rows are sorted in the descending order of hash values.

model and watermarking scheme has enhanced watermark security of [10] by using the secret parameter γ for computing C_{P_T} and upper and lower bounds of Δs. Furthermore, embedding the watermark in high ranking features further increases the watermark security because now an attacker cannot target only low ranked features to remove the watermark from the dataset. Moreover, an attacker cannot delete the high ranking features because doing so will make the data useless for the purpose of extracting knowledge by mining datasets.

The time taken (in seconds) by each optimization technique is tabulated in Table I. All three techniques took relatively

more time for larger datasets or for datasets with more features. The reason is that feature selection schemes, used during the optimization process, take longer time to identify relevant features. As expected the classic technique finds the optima in less time compared with bio-inspired techniques – PSO and GA. However, MINLP is unable to find (in most cases) the global optimum (maximum allowable information-preserving alterations), (see Table I).

TABLE I
TIME TAKEN (IN SECONDS) BY DIFFERENT OPTIMIZATION SCHEMES FOR FINDING THE OPTIMUM VALUE OF Δ, AND WHETHER OR NOT THE GLOBAL OPTIMUM WAS FOUND IN THAT PARTICULAR TIME.

Dataset	**Optimization Techniques**			**Global optimum found?**		
	PSO	**GA**	**MINLP**	**PSO**	**GA**	**MINLP**
Ann-Thyroid	0.18	0.532	0.144	Yes	Yes	No
BreastCancer	0.096	0.3304	0.0768	Yes	Yes	No
BreastCancerDiagnostic	0.101	0.3424	0.0808	No	No	No
BreastCancerPrognostic	0.101	0.3424	0.0808	Yes	Yes	No
Cleveland-Heart	0.098	0.3352	0.0784	Yes	Yes	No
ContraceptiveMethod	0.099	0.3376	0.0792	Yes	Yes	No
Dermatology	0.096	0.3304	0.0768	No	Yes	No
Echocardiogram	0.098	0.3352	0.0784	Yes	Yes	No
E-Coli	0.098	0.3352	0.0784	Yes	Yes	No
HabermansSurvival	0.096	0.3304	0.0768	Yes	Yes	Yes
Hepatitis	0.098	0.3352	0.0784	Yes	Yes	No
HorseColic	0.098	0.3352	0.0784	Yes	Yes	No
HungarianHeart	0.098	0.3352	0.0784	Yes	Yes	No
HyperThyroid	0.101	0.3424	0.0808	Yes	Yes	No
Hypo-Thyroid	0.1	0.34	0.08	Yes	Yes	No
LiverDisorders	0.098	0.3352	0.0784	Yes	No	No
LymphNodes	0.096	0.3304	0.0768	No	Yes	Yes
MammographicMasses	0.096	0.3304	0.0768	Yes	Yes	No
NewThyroid	0.097	0.3328	0.0776	Yes	Yes	No
PimaIndiansDiabetes	0.098	0.3352	0.0784	Yes	Yes	No
Sick	0.099	0.3376	0.0792	Yes	Yes	No
StatlogHeart	0.098	0.3352	0.0784	Yes	Yes	No
SwitzerlandHeart	0.098	0.3352	0.0784	Yes	Yes	No
Thyroid0387	0.102	0.3448	0.0816	Yes	Yes	No
VA-Heart	0.096	0.3304	0.0768	Yes	Yes	No
Mean	0.101	0.3435	0.081	Yes	Yes	No

We have conducted experiments on all the 24 datasets listed in Table I. For brevity, we have chosen a relatively small dataset – PimaIndiansDiabetes – with 8 features and a class label to build insights about the proposed technique. We report the results of data grouping on only one dataset $pima-diabetes$ with an average mutual information, $Avg = 0.1338$, grouping parameter =0.3, and threshold $C_{P_T} = 0.04014$ in Table II.

TABLE II
ASSIGNMENT OF VARIOUS FEATURES (F1 TO F8) TO DATA GROUPS USING EQUATION (29) WITH THE PARAMETERS' VALUES $Avg = 0.1338$, $\gamma = 0.3$ AND THRESHOLD $C_{P_T} = 0.04014$.

S	**f1**	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**	**f8**
I	0.0618	0.3042	0.0593	0.0817	0.2771	0.1257	0.02	0.1409
Group	g_1	g_5	g_1	g_2	g_5	g_3	g_4	g_5

The optimum value of Δ for each feature in the selected dataset is depicted in Figure 8. It is evident from the figure that the optimum value of Δ for features that have relatively high classification potentials is smaller and vice versa. Moreover, it is clear from the Figure 8 that MINLP stuck at the local optima in most of the cases, whereas PSO and GA have been relatively more successful in moving closer to the global optimum for almost every feature. We have chosen PSO because it gives nearly optimum solution but in less time.

For brevity, we report a test run of watermark embedding on numeric features in only four rows of the dataset shown in Table III.

TABLE III
ORIGINAL DATA.

f1	f2	f3	f4	f5	f6	f7	f8	class
6.000	148.000	72.000	35.000	0.000	33.600	0.627	50.000	1
1.000	85.000	66.000	29.000	0.000	26.600	0.351	31.000	0
8.000	183.000	64.000	0.000	0.000	23.300	0.672	32.000	1
1.000	89.000	66.000	23.000	94.000	28.100	0.167	21.000	0

The steps of watermark embedding (with $W = 101$) process for numeric features are tabulated in Table IV. In this example, $\Delta_{f1} = 1.3\%$, $\Delta_{f2}0.0000054\%$, $\Delta_{f3} = 1.5\%$, $\Delta_{f4} = 0.06\%$, $\Delta_{f5} = 0.000001\%$, $\Delta_{f6} = 0.1\%$, $\Delta_{f7} = 0.00049\%$, $\Delta_{f8}0.001\%$ as determined by our implementation of PSO algorithm.

TABLE IV

WATERMARK EMBEDDING PROCESS FOR NUMERIC FEATURES WITH A WATERMARK $W = 101$. THE NUMERIC FEATURES IN A GROUP g_i ARE ADDED WITH POSITIVE Δ_i, IF A WATERMARK BIT b IS 1; AND WITH NEGATIVE Δ_i IF A WATERMARK BIT b IS 0.

W	f1	f2	f3	f4	f5	f6	f7	f8	class	T_1^*	T_2^*	T_3^*	T_4^*	T_5^*	T_6^*	T_7^*	T_8^*
b=1	6.078	148	73.08	35.021	0	33.634	0.627	50.001	1	0.078	0.00000799	1.08	0.021	0	0.0336	0.00000307	0.0005
	1.013	85	66.99	29.017	0	26.627	0.351	31	0	0.013	0.00000459	0.99	0.0174	0	0.0266	0.00000172	0.00031
	8.104	183	64.96	0	0	23.323	0.672	32	1	0.104	0.00000988	0.96	0	0	0.0233	0.00000329	0.00032
	1.013	89	66.99	23.014	94	28.128	0.167	21	0	0.013	0.00000481	0.99	0.0138	0.00000094	0.0281	0.000000818	0.00021
b=0	5.999	148	71.984	35	0	33.6	0.627	50	1	0.079014	0.00000799	1.0962	0.021013	0	0.033634	0.00000307	0.0005
	1	85	65.985	29	0	26.6	0.351	31	0	0.013169	0.00000459	1.00485	0.01741	0	0.026627	0.00000172	0.00031
	7.999	183	63.986	0	0	23.3	0.672	32	1	0.105352	0.00000988	0.9744	0	0	0.023323	0.00000329	0.00032
	1	89	65.985	23	94	28.1	0.167	21	0	0.013169	0.00000481	1.00485	0.013808	0.00000094	0.028128	0.000000818	0.00021
b=1	6.077	148	73.064	35.021	0	33.634	0.627	50.001	1	0.077987	0.00000799	1.07976	0.021	0	0.0336	0.00000307	0.0005
	1.013	85	66.975	29.017	0	26.627	0.351	31	0	0.013	0.00000459	0.989775	0.0174	0	0.0266	0.00000172	0.00031
	8.103	183	64.946	0	0	23.323	0.672	32	1	0.103987	0.00000988	0.95979	0	0	0.0233	0.00000329	0.00032
	1.013	89	66.975	23.014	94	28.128	0.167	21	0	0.013	0.00000481	0.989775	0.0138	0.00000094	0.0281	0.000000818	0.00021

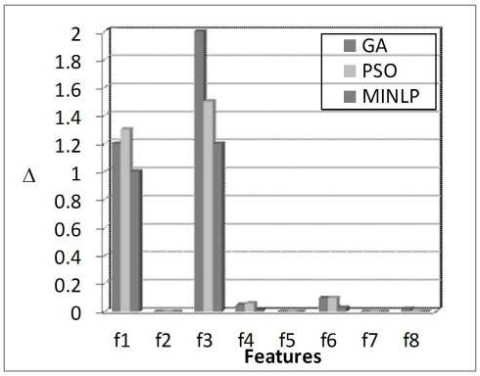

Fig. 8. Performance of optimization schemes used to find an optimum value of Δ. It is clear from this figure that higher (except feature $f7$) the value of mutual information I, smaller the optimized value of Δ and vice versa.

It is important to mention here that the any constraints on the data type are strictly enforced during the watermark embedding process. For instance, if a feature in the original dataset does not contain a floating point number then the watermarked data should also possess the same properties of that feature. We have reported our results in Tables IV, V, and VI for an illustrative purpose only. Moreover, the rows having zero values are also left unmarked. After embedding all the watermark bits, the final data takes the form as shown in Table V.

TABLE V
WATERMARKED DATA.

f1	f2	f3	f4	f5	f6	f7	f8	class
6.077	148.000	73.064	35.021	0.000	33.634	0.627	50.001	1
1.013	85.000	66.975	29.017	0.000	26.627	0.351	31.000	0
8.103	183.000	64.946	0.000	0.000	23.323	0.672	32.000	1
1.013	89.000	66.975	23.014	94.000	28.128	0.167	21.000	0

The steps of watermark decoding from the watermarked data are reported in Table VI. Note that the watermark is not decoded from the features f2, f5, f7, and f8 because watermark embedding did not modify their values. The reason is that either have very high classification potential or the value of Δ for them was approximately zero.

TABLE VI
WATERMARK DECODING PROCESS FOR NUMERIC FEATURES. THE VALUE OF THE PARAMETER val FOR A FEATURE IS COMPARED WITH THE DECODING THRESHOLD T^* (SEE TABLE IV) FOR THAT FEATURE AND IF val IS FOUND TO BE GREATER THAN T^*, THE WATERMARK BIT b IS DECODED AS 1; OTHERWISE, IT IS DECODED AS 0. THE WATERMARK BITS SHOWN IN BOLD FACE ARE THE RESULT OF MAJORITY VOTING THAT IS USED AS AN ERROR CORRECTION MECHANISM AFTER DECODING A WATERMARK BIT b FROM ALL THE ROWS OF THE DATASET. THE FINAL ROW IN THIS TABLE SHOWS THE DECODED WATERMARK FROM EACH OF THE MARKED FEATURES.

f1	f3	f4	f6	class	val_1	val_3	val_4	val_6	Decoded bits			
6.077	73.089	35.021	33.634	1	0.079	1.09634	0.02101	0.03363	1	1	1	1
1.013	66.998	29.017	26.627	0	0.01317	1.00497	0.01741	0.02663	1	1	1	1
8.103	64.968	0.000	23.323	1	0.10534	0.97452	0	0.02332	1	1	×	1
1.013	66.998	23.014	28.128	0	0.01317	1.00497	0.01381	0.02813	1	1	1	1
Majority voting									**1**	**1**	**1**	**1**
6.078	73.064	35.021	33.634	1	0.07901	1.09596	0.02101	0.03363	0	0	0	0
1.013	66.975	29.017	26.627	0	0.01317	1.00463	0.01741	0.02663	0	0	0	0
8.105	64.946	0.000	23.323	1	0.10537	0.97419	0	0.02332	1	0	×	0
1.013	66.975	23.014	28.128	0	0.01317	1.00463	0.01381	0.02813	0	0	0	0
Majority voting									**0**	**0**	**0**	**0**
6.077	73.089	35.021	33.634	1	0.079	1.09634	0.02101	0.03363	1	1	1	1
1.013	66.998	29.017	26.627	0	0.01317	1.00497	0.01741	0.02663	1	1	1	1
8.107	64.924	0.000	23.323	1	0.10539	0.97386	0	0.02332	1	1	×	1
1.013	66.998	23.014	28.128	0	0.01317	1.00497	0.01381	0.02813	1	1	1	1
Majority voting									**1**	**1**	**1**	**1**
Decoded Watermarks									101	101	101	101

After applying the majority voting step on all the decoded bits, we get the bit sequence "101" which is same as the embedded watermark.

C. Knowledge-preserving characteristic of proposed model

We give the optimized "usability constraints", derived from our formal model, to the enhanced watermark embedding algorithm. We report the effect of watermarking, with the proposed usability constraints model, on various feature selection schemes in Table VII (for brevity we report our result on only one dataset $pima - diabetes$). Recall that in Theorem 1, it is

proven that mutual information between a feature and the class attribute can only remain same for original and watermarked datasets, if the features belong to the same class in both datasets. To see the impact of this theorem, we report mutual information of original and watermarked features in Table VII to prove that the features' values of the watermarked dataset have the same relation with the class labels as they had before watermark embedding. It is evident from Table VII that learning statistics of all feature selection schemes have also been preserved by enforcing "usability constraints" of our formal model.

TABLE VII

EFFECT OF WATERMARKING, WITH PROPOSED USABILITY CONSTRAINTS MODEL, ON VARIOUS FEATURE SELECTION SCHEMES. IN THIS TABLE F1 TO F8 ARE THE DATASET FEATURES AND S DENOTES FEATURE SELECTION SCHEMES. IN THE COLUMNS FOR CFS AND CBF THE ENTRY *Yes* INDICATES THAT FEATURE WAS SELECTED BY S AND *No* MEANS THAT THE FEATURE WAS NOT SELECTED BY S.

Original data								
S	**Features**							
	f1	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**	**f8**
I	0.0618	0.3042	0.0593	0.0817	0.2771	0.1257	0.02	0.1409
IG	0.0392	0.1901	0.014	0.0443	0.0595	0.0749	0.0208	0.0725
IG_r	0.0515	0.0986	0.0144	0.0224	0.0394	0.0863	0.0226	0.0726
CFS	No	Yes	No	No	No	Yes	Yes	Yes
CBF	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PC	0.7382	0.5218	0.3931	0.2837	0.1884	0.103	0.0506	0
Watermarked data								
S	**Features**							
	f1	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**	**f8**
I	0.0618	0.3042	0.0593	0.0817	0.2771	0.1257	0.02	0.1409
IG	0.0392	0.1901	0.014	0.0443	0.0595	0.0749	0.0208	0.0725
IG_r	0.0515	0.0986	0.0144	0.0224	0.0394	0.0863	0.0226	0.0726
CFS	No	Yes	No	No	No	Yes	Yes	Yes
CBF	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
PC	0.7382	0.5218	0.3931	0.2837	0.1884	0.103	0.0506	0

Once the watermark is inserted in a dataset, we classify – using five well known machine learning algorithms – all 25 original datasets and their corresponding 24 watermarked datasets. We compare the change in two learning statistics: TP_{Rate} and FP_{Rate}. The learning statistics (TP_{Rate} and FP_{Rate}) are preserved for all learning algorithms with the exception of JRip – its TP_{Rate} and FP_{Rate} for some watermarked datasets vary in between ± 0.001 which is negligible. But we can safely generalize that the overall learning statistics – TP_{Rate} and FP_{Rate} – are preserved by enforcing the usability constraints of our formal model. Table VIII shows the learning statistics of original and watermarked $pima - diabetes$ dataset (the results of other datasets are skipped for brevity but they also show the same pattern.)

TABLE VIII

EFFECT OF WATERMARKING, WITH PROPOSED USABILITY CONSTRAINTS MODEL, ON TP_{Rate} AND FP_{Rate} OBTAINED BY RUNNING THE LEARNING ALGORITHM FOR CLASSIFYING ORIGINAL DATASET AND WATERMARKED DATASET. IN THIS TABLE, Δ_{TP} AND Δ_{FP} ARE THE DIFFERENCE IN TP_{Rate} AND FP_{Rate} FOR D_O (ORIGINAL DATA) AND D_W (WATERMARKED DATA) RESPECTIVELY.

TP_{Rate}															
S	**Learning algorithms**														
	J48			**SMO**			**NB**			**IBk**			**JRip**		
	D_O	D_W	Δ_{TP}	D_O	D_W	Δ_{TP}	D_O	D_W	Δ_{TP}	D_O	D_W	Δ_{TP}	D_O	D_W	Δ_{TP}
I	0.738	0.738	0	0.773	0.773	0	0.763	0.763	0	0.732	0.732	0	0.751	0.752	-0.001
IG	0.738	0.738	0	0.773	0.773	0	0.763	0.763	0	0.732	0.732	0	0.754	0.753	0.001
IG_r	0.738	0.738	0	0.773	0.773	0	0.763	0.763	0	0.732	0.732	0	0.751	0.751	0
CFS	0.749	0.749	0	0.77	0.77	0	0.775	0.775	0	0.738	0.738	0	0.747	0.748	-0.001
CBF	0.738	0.738	0	0.773	0.773	0	0.763	0.763	0	0.732	0.732	0	0.751	0.752	-0.001
PC	0.719	0.719	0	0.775	0.775	0	0.74	0.74	0	0.741	0.741	0	0.714	0.714	0
FP_{Rate}															
S	**Learning algorithms**														
	J48			**SMO**			**NB**			**IBk**			**JRip**		
	D_O	D_W	Δ_{FP}	D_O	D_W	Δ_{FP}	D_O	D_W	Δ_{FP}	D_O	D_W	Δ_{FP}	D_O	D_W	Δ_{FP}
I	0.327	0.327	0	0.334	0.334	0	0.307	0.307	0	0.358	0.358	0	0.317	0.318	-0.001
IG	0.327	0.327	0	0.334	0.334	0	0.307	0.307	0	0.358	0.358	0	0.348	0.348	0
IG_r	0.327	0.327	0	0.334	0.334	0	0.307	0.307	0	0.358	0.358	0	0.305	0.305	0
CFS	0.341	0.341	0	0.343	0.343	0	0.313	0.313	0	0.343	0.343	0	0.34	0.341	-0.001
CBF	0.327	0.327	0	0.334	0.334	0	0.307	0.307	0	0.358	0.358	0	0.317	0.318	-0.001
PC	0.369	0.369	0	0.339	0.339	0	0.349	0.349	0	0.354	0.354	0	0.379	0.379	0

We show in Figure 9 that the rule boundaries $\Re_b$ – the third element of learning statistic – are also preserved. For brevity, we show in Figure 9 the classification rules, extracted by J48 only, for the original and watermarked $pima - diabetes$ datasets. It is evident that the rules to predict the class labels have remained unchanged.

To conclude, we have empirically proven that by enforcing our model of Definition 15 the watermarking scheme is knowledge-preserving & lossless (Theorem 2).

VII. CONCLUSION

In this paper, a novel knowledge-preserving and lossless usability constraints model has been proposed for watermarking data mining datasets. The proposed model has been given as an input to a new generic watermarking scheme for data mining

f2 <= 127 and f6 <= 26.4: 0 (132.0/3.0)

f2 <= 127 and f6 > 26.4 and f8 <= 28: 0 (180.0/22.0)

f2 <= 127 and f6 > 26.4 and f8 >28 and f2 <= 99: 0 (55.0/10.0)

f2 <= 127 and f6 > 26.4 and f8 >28 and f2 > 99 and f7 > 0.561 and f1 <= 6 and f8 <= 30: 1 (4.0)

f2 <= 127 and f6 > 26.4 and f2 > 99 and f7 > 0.561 and f1 <= 6 and f8 > 34 and f1 > 6: 1 (13.0)

f2 > 127 and f6 <= 29.9 and f2 <= 145: 0 (41.0/6.0)

f6 <= 29.9 and f2 > 145 and f8 <= 25: 0 (4.0)

f6 <= 29.9 and f2 > 145 and f8 > 25 and f8 <= 61 and f6 >27.1 and f3 > 82: 0 (4.0)

f6 <= 29.9 and f2 > 145 and f8 > 61: 0 (4.0)

f2 > 127 and f6 > 29.9 and f2 <= 157 and f3 <= 61: 1 (15.0/1.0)

f2 > 127 and f6 > 29.9 and f2 <= 157 and f3 > 61 and f8 <= 30: 0 (40.0/13.0)

f2 > 127 and f6 > 29.9 and f2 <= 157 and f3 > 61 and f8 > 30: 1 (60.0/17.0)

(a) Original data

f2 <= 127 and f6 <= 26.4: 0 (132.0/3.0)

f2 <= 127 and f6 > 26.4 and f8 <= 28: 0 (180.0/22.0)

f2 <= 127 and f6 > 26.4 and f8 >28 and f2 <= 99: 0 (55.0/10.0)

f2 <= 127 and f6 > 26.4 and f8 >28 and f2 > 99 and f7 > 0.561 and f1 <= 6 and f8 <= 30: 1 (4.0)

f2 <= 127 and f6 > 26.4 and f2 > 99 and f7 > 0.561 and f1 <= 6 and f8 > 34 and f1 > 6: 1 (13.0)

f2 > 127 and f6 <= 29.9 and f2 <= 145: 0 (41.0/6.0)

f6 <= 29.9 and f2 > 145 and f8 <= 25: 0 (4.0)

f6 <= 29.9 and f2 > 145 and f8 > 25 and f8 <= 61 and f6 >27.1 and f3 > 82: 0 (4.0)

f6 <= 29.9 and f2 > 145 and f8 > 61: 0 (4.0)

f2 > 127 and f6 > 29.9 and f2 <= 157 and f3 <= 61: 1 (15.0/1.0)

f2 > 127 and f6 > 29.9 and f2 <= 157 and f3 > 61 and f8 <= 30: 0 (40.0/13.0)

f2 > 127 and f6 > 29.9 and f2 <= 157 and f3 > 61 and f8 > 30: 1 (60.0/17.0)

(b) Watermarked data

Fig. 9. Effect of watermarking scheme, using proposed usability constraints model, on formation of Decision rule boundaries $\Re_b$.

datasets. The benefits of this technique are: (1) identifying the vital characteristics of a dataset which need to be preserved during watermarking; (2) ranking the features on the basis of their classification potentials; (3) logically grouping the data into different groups (clusters) based on this ranking for defining local usability constraints for each group; (4) defining global usability constraints for the complete dataset; (5) modeling the local and global usability constraints in such a manner so that the learning statistics of a classifiers are preserved; (6) optimizing the watermark embedding such that all usability constraints remain intact. To the best of our knowledge, no technique in the literature exists that automatically computes "usability constraints" for a dataset that once enforced would preserve the knowledge contained in it. Moreover, the enhanced watermarking scheme can work with any type of data: numeric, non-numeric and strings. The proposed technique can be easily employed by the customers of companies like Kaggle to share datasets with data-mining experts by safeguarding and protecting their ownership. The technique, in a future work, could be extended to images that contain sensitive information (medical and surveillance) images.

REFERENCES

[1] "Kaggle's contests: Crunching numbers for fame and glory," http://www.businessweek.com/magazine/kaggles-contests-crunching-numbers-for-fame-and-glory-01042012.html, last accessed February, 27 2012.

[2] "Patients sue walgreens for making money on their data," http://www.healthcareitnews.com/news/patients-sue- walgreens-making-money-their-data, last accessed: February 21, 2012.

[3] R. Agrawal, P. Haas, and J. Kiernan, "Watermarking relational data: framework, algorithms and analysis," *The VLDB journal*, vol. 12, no. 2, pp. 157–169, 2003.

[4] J. Palsberg, S. Krishnaswamy, M. Kwon, D. Ma, Q. Shao, and Y. Zhang, "Experience with software watermarking," in *acsac*. Published by the IEEE Computer Society, 2000, pp. 308–316.

[5] M. Atallah, V. Raskin, M. Crogan, C. Hempelmann, F. Kerschbaum, D. Mohamed, and S. Naik, "Natural language watermarking: Design, analysis, and a proof-of-concept implementation," in *Information Hiding*. Springer, 2001, pp. 185–200.

[6] R. Agrawal and J. Kiernan, "Watermarking relational databases," in *28th International Conference on Very Large Data Bases*. Morgan Kaufmann Pub, 2002, pp. 155–166.
[7] R. Sion, M. Atallah, and S. Prabhakar, "Rights protection for relational data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 6, pp. 1509–1525, 2004.
[8] M. Shehab, E. Bertino, and A. Ghafoor, "Watermarking relational databases using optimization-based techniques," *IEEE Transactions on Knowledge and Data Engineering*, vol. 20, no. 1, pp. 116–129, 2008.
[9] R. Lewis, V. Torczon, I. for Computer Applications in Science, and Engineering, *Pattern search methods for linearly constrained minimization*. Citeseer, 1998.
[10] M. Kamran and M. Farooq, "An information-preserving watermarking scheme for right protection of emr systems," *IEEE Transactions on Knowledge and Data Engineering*, In press.
[11] M. Pinsker, "Information and information stability of random variables and processes," 1960.
[12] J. Kennedy, R. Eberhart *et al.*, "Particle swarm optimization," in *Proceedings of IEEE international conference on neural networks*, vol. 4. Piscataway, NJ: IEEE, 1995, pp. 1942–1948.
[13] E. Elbeltagi, T. Hegazy, and D. Grierson, "Comparison among five evolutionary-based optimization algorithms," *Advanced Engineering Informatics*, vol. 19, no. 1, pp. 43–53, 2005.
[14] J. Holland, *Adaptation in natural and artificial systems*. MIT press Cambridge, MA, 1992.
[15] M. Bussieck and A. Pruessner, "Mixed-integer nonlinear programming," *SIAG/OPT Newsletter: Views & News*, vol. 14, no. 1, pp. 19–22, 2003.
[16] M. Duran and I. Grossmann, "An outer-approximation algorithm for a class of mixed-integer nonlinear programs," *Mathematical Programming*, vol. 36, no. 3, pp. 307–339, 1986.
[17] I. Grossmann and Z. Kravanja, "Mixed-integer nonlinear programming: A survey of algorithms and applications," *Large-Scale Optimization with Applications, Part II*, pp. 73–100.
[18] L. Schrage, *Lindo: An optimization modeling system*. Scientific Press, South San Francisco, CA, 1991.
[19] M. Kamran and M. Farooq, "A formal usability constraints model for watermarking of outsourced data mining datasets," *Technical Report: TR-59-Kamran*, 2012.