Pushing Novelty Criterion into Incremental Mining Algorithm

Ahmed Sultan Al-Hegami

Sana'a University, Sana'a, YEMEN

Summary

Classification is an important problem in data mining. Decision tree induction is one of the most common techniques that are applied to solve the classification problem. Many decision tree induction algorithms have been proposed based on different attribute selection and pruning strategies. Massively increasing volume of data in real life databases has motivated researchers to design novel and incremental algorithms for decision tree induction.

In this paper, we propose an incremental tree induction algorithm that integrates novelty criterion during tree induction. One of the main features of the proposed approach is to capture the user background knowledge, which is monotonically augmented. The incremental classifier that reflects the changing data and the user beliefs is attractive in order to make the over all KDD process more effective and efficient. We tested the proposed classifier and experiment with some public datasets and found the results quite promising.

Key words:
Knowledge discovery in databases, machine learning, data mining, incremental classifier, decision tree, pruning technique, domain knowledge, classification, novelty measure.

1. Introduction

Classification is an important data mining task that analyzes a given training set and develops a model for each class according to the features present in the data. The generated model is used to classify unseen data tuples. There are many approaches to develop the classification model including decision trees, neural networks, nearest neighbor methods and rough set-based methods [14, 29]. Decision tree induction methods are the most widely used to construct classification model [27,29]. These methods partition the data recursively until all tuples in every partition have the same class value. The resultant model is a tree that is used for predicting the class label. Decision tree induction algorithms operate in two phases, the C*onstruction phase* and *Pruning phase* [3,14,29]. The *construction phase* of decision tree results in a complex tree that often overfits the data. Overfitting not only reduces the accuracy, when applied to unseen data, but also increases the size of the tree, construction time, and prediction time. The *Pruning phase* of decision tree is the process of removing "overfitted" branches to improve the accuracy and performance of the decision tree. There are two approaches of tree pruning, pre-pruning and Post-pruning. In pre-pruning approach, a tree is pruned by stopping its construction by deciding not to further partition the subset of training data at a given node. As a consequence, a node becomes a leaf that holds a class value with the most frequent class among the subset of samples. Pre-pruning criteria are based on statistical significance, information gain, or error reduction. The post-pruning, removes branches from the completely grown tree, by traversing the constructed tree and uses the estimated errors to decide whether some undesired branches should be replaced by a leaf node or not. This replacement is the key issue of many pruning criteria that appear in the literature.

Many algorithms for inducing decision trees have been proposed in the literature (e.g., C4.5 [3], CART [1], SPRINT [5]), based on different attribute selection and pruning strategies. Most of these algorithms operate in a construction phase followed by pruning phase that make an impact on the time and efficiency of the algorithms. However, algorithms such as PUBLIC [4], BOAT [2] address such issues by constructing and pruning the tree in one stage. This process is established by pushing constraints such as accuracy and size into the decision tree in order to prune the tree dynamically and hence result in reduction the size and improvement the performance of decision tree. Although, these approaches require the user to provide constraints, the user background knowledge is not implicitly/explicitly stated. This lack of incorporating user domain knowledge (DK) and previously discovered knowledge (PDK) into the tree induction process results in a decision tree which may be optimal in size and accuracy but may generate branches that are similar to the earlier discovered tree and hence does not reflect the user interest. One of the main drawbacks with the classical decision tree induction algorithms is that they do not consider the time in which the data arrived. In practice, data is acquired in small batches over the time. In such scenario a combination of old and new data is used to build a new

Manuscript received December 5, 2007
Manuscript revised December 20, 2007

classifier from scratch. This results in losing of the previously discovered knowledge (PDK). Researchers therefore have been strongly motivated to propose techniques that update the classification model as new data arrives, rather than running the algorithms from scratch [14,16,17,18], resulting in incremental classifiers. Incremental algorithms build and refine the model as new data arrive at different points in time, in contrast to the traditional tree induction algorithms where they perform model building in batch manner [14,16,17,18]. The incremental classifiers that reflect the changing data trends and the user beliefs are attractive in order to make the over all KDD process more effective and efficient.

We propose an incremental algorithm based on the premise that unless the underlying data generation process has changed dramatically, it is expected that the rules discovered from one set are likely to be similar (in varying degrees) to those discovered from another set [7,15].

Novelty of a rule is the extent to which the rule adds to the prior knowledge of the user [7,11,15,22]. It can be used as an effective way to filter the rule set discovered from the target data set thereby, reducing the volume of the output. Our work extends the approaches presented in [7,15] and integrates them into tree induction algorithm in an incremental manner. The proposed approach is a self-upgrading filter that keeps *known knowledge* rule base updated as new novel rules discovered. The proposed filter quantifies *novelty* of the discovered knowledge on the basis of deviation of the newly discovered rules with respect to the known knowledge.

The proposed approach operates on the incremental training set and induces the decision tree. During induction, the algorithm omputes the accuracy at each branch (partial rule) to guarantee that the accuracy is not compromised. It dynamically prunes the decision tree by retaining only the branches with high *novelty* measure. The *novelty* measure of the partially constructed branches is computed with respect to known knowledge i.e. domain knowledge and previously discovered classification rules. The incremental nature of the approach makes it advantageous to discover new patterns at current time with respect to the previously discovered patterns (rules), rather than exhaustively discovering all patterns.

2. Related Works

We discuss the related work in three categories viz. i) incremental classifiers, ii) novelty measure, and iii) the work related to incorporating constraints into decision tree induction algorithms.

The problem of data set over evolving time has motivated development of many incremental classifiers including COBWEB [19], ID4 [20], ID5 [18], ID5R [18] and IDL [21]. The advantages of incremental techniques over traditional techniques are elaborated in [17]. Though, incremental tree induction takes the change of data over time, neither constraints nor user background knowledge is pushed into the induction algorithms which may result in inducing a decision tree that does not reflect the user interest.

Novelty has been considered as an important characteristic of the discovered knowledge, though difficult to measure because of user subjectivity. The work proposed in [22] detects the novelty of rules mined from text based on the lexical knowledge in WordNet. In [7,11,15], a framework has been proposed to quantify novelty in terms of deviation of currently discovered knowledge with respect to domain knowledge and previously discovered knowledge. The approach presented in [11] is intuitive in nature and lays more emphasis on user involvement in quantification process by way of parameter specification. In [7,15] the quantification of novelty is performed objectively and user involvement is sought for categorization of rules (as novel, unexpected, generalized, specialized, conformed) based on *novelty* measure. The present work integrates the work presented in [7,15] into a decision tree induction algorithm to form a constraint to discover novel rules.

Several algorithms have been proposed that push size and accuracy constraints into the tree induction algorithms [4,6,23,24,25]. Though not all of the specified constraints can be pushed into the data mining algorithm [26], yet recently constraints have been successfully used to restrict the search space [23,24,25].

The approaches presented in [4,6] push the accuracy and size constraints into the decision tree in order to prune the tree dynamically. In particular, the previously discovered knowledge (*PDK*) and the user domain knowledge (*DK*) have not been used as a constraint during incremental tree induction. This results into a decision tree with most of the branches that are similar to the earlier discovered tree.

3. Problem Statement

Given a dataset D collected over the time $[0,t_1,t_2,\ldots t_n]$. At each time instance t_i, an incremental dataset D_i , $i \in \{1,\ldots,n\}$, is collected and stored in D. Let T_{t_i} and $T_{t_{i+1}}$ be two decision trees induced at time instances t_i and t_{i+1} from incremental dataset D_i and D_{i+1} respectively. Let K_i and K_{i+1} be the set of extracted classification rules from T_{t_i} and $T_{t_{i+1}}$ respectively. Figure 1 shows the knowledge discovered at two time instances. Major volume of K_{i+1} would be the overlapping region that represents previously discovered classification rules, unless the data generation process has significantly changes. The shaded portion denotes the novel rules while the overlapping region

denoted discovered knowledge (with varying degree of sameness) at two points in time.
Intuitively, high degree of novelty is assigned to the rules falling in the shaded area compared to those in the overlapping regions.

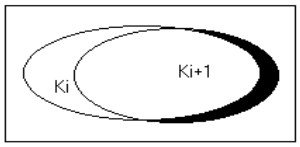

Figure 1.The Discovered Rules from two Decision Trees

The objective is to construct $T_{t_{i+1}}$ at t_{i+1} such that rules extracted from $T_{t_{i+1}}$ have desired degree of novelty with respect to the rules extracted from T_{t_i} at t_i (PDK), with acceptable degree of accuracy.
It is expected that the constructed decision tree $T_{t_{i+1}}$, using this approach, is smaller in size, generates novel rules with acceptable degree of accuracy. This is a useful feature as the volume of data keeps on escalating over the time and hence the user background knowledge is monotonically augmented.
In addition, the approach guarantees that a node pruned during constructing phase will certainly be pruned as the tree is built and then pruned using novelty criterion. This strategy saves time and effort required to build the tree.

4. Novelty Measure

The notion of *novelty* of the knowledge indicates the extent to which the discovered rules contribute to new knowledge for a user [7,11,15,22]. It is purely subjective and encompasses the unexpectedness of discovered model with respect to the *known knowledge* (domain knowledge (DK) and previously discovered knowledge (PDK)). Novelty of a rule can be used as an effective way to filter the rule set discovered from the target data set thereby, reducing the volume of the output. Though *novelty* is a subjective measure, we proposed a strategy to quantify objectively the *novelty* measure of each discovered rule, and facilitate categorization of rules based on the degree of novelty desired by the user [7,15].
To compute the *novelty* of a rule, the deviation is measured for the antecedent and the consequent at conjunct level and subsequently the conjunct level deviation is combined to compute rule level deviation.

4.1 Definitions and Notations

A rule R has the form: $A \rightarrow C$ where A denotes an antecedent and C denotes a consequent. Both A and C are in CNF ($c_1 \wedge c_2 \wedge \ldots\ldots \wedge c_k$). The conjunct c_j is of the form $<Å, O, V>$. Where $Å$ is an attribute, Dom ($Å$) is the domain of $Å$, and $V \in Dom$ ($Å$), $O \in \{=,<,>,\geq,\leq\}$. Without loss of generality, we consider A as a set of conjuncts and consider C consists of one conjunct.

4.2 Deviation at Conjunct Level

In order to quantify deviation between any two conjuncts, the attributes, operators, and attribute values of the two conjuncts in question need to be taken into account.

Definition 1: Two conjuncts c_i and c_j ($<Å_i O_i V_i>$ and $<Å_j O_j V_j>$ respectively) are compatible if and only if $Å_i = Å_j$. Otherwise, we consider c_i and c_j as non-compatibles.
Definition 2: Let c_i and c_j be two non-compatible conjuncts. The deviation $\delta(c_i,c_j)$ between them is defined to be 1.
We capture the following four types of deviations between two compatible conjuncts.
Definition 3: Let c_1 and c_2 be two compatible conjuncts ($A_1O_1V_1$ and $A_2O_2V_2$ respectively). The deviation of c_1 with respect to c_2 is defined as follows:

$$\delta(c_1,c_2) = \begin{cases} 0 & \text{if } O_1 = O_2 \text{ and } V_1 = V_2 . \\ \dfrac{|V_1 - V_2|}{Range(Dom(A_1))} & \text{if } O_1 = O_2 \text{ and } V_1 \neq V_2 . \\ opdist(O_1,O_2)/5 & \text{if } O_1 \neq O_2 \text{ and } V_1 = V_2 . \\ f(c,v) & \text{if } O_1 \neq O_2 \text{ and } V_1 \neq V_2 . \end{cases}$$

Notice that the computation of value deviation in the second term of Definition 3 is suitable for only numeric and ordinal attributes. In case of nominal attributes, the change in value can be quantified in terms of probabilities. Since ordinal domains generally have small and manageable cardinality, prior domain knowledge can be used to assign probabilities to domain values. In case it is not feasible to assign probabilities in the above-mentioned way (e.g. color of car), the dataset itself can be used to compute probabilities corresponding to each domain value. The condition deviation in the third term of Definition 3 takes into account the type of changes in the condition. The operators are formatted on a number line as shown in Figure 2. The deviation between the operators is quantified by the distance between the operators on the numberline. We define a function $opdist\ (O_1, O_2) \rightarrow \{0,1,2,3,4\}$, which denotes the distance between the two distinct operators (O_1, O_2) on the numberline. We define

five possible values of deviations (0, 1/5, 2/5, 3/5, 4/5) between any two operators, ranking the extent of deviation between condition operators in two conjuncts.

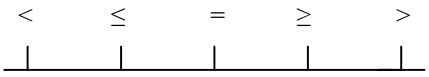

Figure 2. Operators on Numberline

The forth term of the Definition 3 captures the change in both conditions and attribute values in two conjuncts. We compute these deviations independently of each other in the two given conjuncts. The user defines a real valued function $f(c,v) \rightarrow [0,1]$ to combine the two types of deviations. Depending on the importance of the type of deviations for a specific application in a domain, different functions can be used for computing deviations on different attributes. Typically, $f(c,v)$ is of the form w_1c+w_2v, where w_1, w_2 are user numerical values and $w_1+w_2=1$.

Lemma 1: The conjunct level deviation lies between [0,1].
Proof 1: By Definition 2 and Definition 3.

4.3 Conjunct Set Deviation

In order to compute novelty of a rule, it is necessary to define the deviation Ψ between two conjunct sets, since both antecedents and consequents are considered to be sets of conjuncts. The deviation Ψ (S_1,S_2) between two conjuncts sets is quantified based on the analysis of the possible types of differences between two sets of conjuncts S_1 and S_2. Without loss of generalization, we assume that an attribute occurs at most once in a conjunct set S. Computation of deviation at this level is based on counting incompatible conjuncts among the two sets and quantifying total deviation among the compatible conjuncts. Intuitively, it is the number of incompatible conjuncts that contribute most towards the value of the deviation. While comparing two sets of conjuncts namely S_1 and S_2, three possibilities arise.

i) S_1 and S_2 are identical,
ii) S_1 is a generalization / specialization of S_2.
iii) S_1 and S_2 are different.

We compute the deviation between two conjunct sets as follows.

Definition 4: Let S_1 and S_2 be two conjunct sets with cardinalities $|S_1|$ and $|S_2|$ respectively. Let k be the pairs of compatible conjuncts between S_1 and S_2. The deviation between S_1 and S_2 is computed as:

$$\Psi(S_1,S_2)= \frac{\{|S_1|+|S_2|-2*k\}+\sum_{i=1}^{k}\delta(c_1^i,c_2^i)}{|S_1|+|S_2|}$$

where (c_1^i, c_2^i) is the i^{th} pair of compatible conjuncts.

4.4 Computing Novelty Measure

Novelty of a rule is defined with respect to a given rule set. The algorithm quantifies the deviation of the antecedents and consequents of the rule with those of the closest rule in the *known knowledge (KK)*. The user is encouraged to specify the threshold to sift novel rules based on the computation of $\Psi(S_1,S_2)$. The next definitions are the basis of computation of the degree of novelty of a rule.

Definition 5: Let R_1: $A_1 \rightarrow C_1$ and R_2: $A_2 \rightarrow C_2$ be two rules .We say that R_1 is compatible with respect to R_2, if $\Psi(C_1,C_2)=0$.

Definition 6: Let R_1: $A_1 \rightarrow C_1$ and R_2: $A_2 \rightarrow C_2$ be two rules and $S_1 \in A_1$ and $S_2 \in A_2$ be two sets of conjuncts ($\hat{A}_1O_1V_1$ and $\hat{A}_2O_2V_2$ respectively). The novelty measure of R_1 with respect to R_2 denoted by $\Omega(R_1, R_2)$ is computed as follows:

$$\Omega(R_1,R_2)=\begin{cases} \Psi(S_1,S_2) & \text{such that } \Psi(S_1,S_2) > \Phi \text{ and } \Psi(C_1,C_2)=0. \\ 1 & \text{iff } \Psi(C_1,C_2)=1. \\ 0 & \text{othetwise} \end{cases}$$

A rule R_1 is considered to be novel with respect to R_2 if either the deviation of both antecedents are greater than the user threshold value (Φ) or the rules are not compatible.

5. Pushing the Novelty Criterion into Incremental Decision Tree

One of the main features of the approach is to deal with time changing data and user beliefs. This is a useful functionality in situations when two datasets have arrived at different points of time or from different geographical locations. Certainly, it is attractive to update the discovered knowledge each time new data arrive.

The approach computes the accuracy at each branch (partial rule) to guarantee that the accuracy is not compromised. Once a branch found to be having an acceptable degree of accuracy, It utilizes the *novelty criterion* to construct novel branches. The novelty criterion forms a constraint to the tree induction algorithm in order to discover novel classification rules. The advantage of pushing such criterion is that the size of the tree built can be reduced and the discovered knowledge reflects the user's requirement on novelty.

5.1 Architecture of the Approach

Figure 3 shows the general architecture of the proposed classifier. At time t_i, database D_i is subjected to induction algorithm, which takes into account the knowledge that the user already has and the previously discovered classification rules i.e. *known knowledge (KK)*.
The *known knowledge* is available to the algorithm in form of rules of the form of $c_1 \wedge c_2 \wedge \ldots \wedge c_k \rightarrow k_i$, where c_j is a conjunct of the form$<\AA, O, V>$, where $\AA$ is an attribute, *Dom (Å)* is the domain of $\AA$, and
$V \in Dom(\AA)$, $O \in \{=,<,>,\geq ,\leq\}$ and k_i is a class value. *Domain knowledge* (DK) has the same knowledge representation of *Known Knowledge (KK).*
The proposed algorithm computes the accuracy at every branch to guarantee that the accuracy is not compromised and induces branches that correspond to n*ovelty* measure greater than the threshold value (Φ). The extracted novel rules are reported to the user, converted to rule format and subsequently are used to update the *known knowledge* rule base. The updated rule base is used by the prediction algorithm to predict new unseen instances.

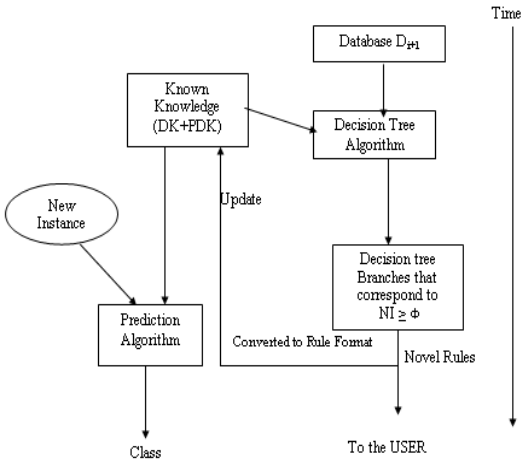

Figure 3. General Architecture of the Approach

The approach updates the classification model not only from the new training data, but also from the user expertise of the domain. Domain knowledge (DK) can be provided by the user, which is used for comparison during tree induction for assessing the novelty of the induced branches (rules).
In the following subsections, we discus the details of the proposed classifier.

5.2 Tree Construction

The approach makes use of information gain [3,27] to select an attribute that best split a portion of data. It dynamically decides whether to grow or not to grow a branch at a node. At each node, it takes into account the partial rule that is obtained by traversing the tree from the root to that node and performing Bayesian test for independence to guess the most frequent class value corresponding to the data used to create that branch. In this paper, we use the terms partial rule and branch interchangeably.
At each selected node, the approach performs the following tasks while constructing the tree:

1. Creation of a node using information gain
2. Computation of statistical significance in order to guess the best-correlated (frequent) class among different possible classes.
3. Computation of classification accuracy of the partial rule and the class value determined in task 2, with different possible combinations of attribute values of the node created in task 1.
4. Computation of the novelty of a partial rule with respect to the *known knowledge*.

We perform Bayesian test to determine the most probable (frequent) class value in the data corresponding to that partition. The classification accuracy for the combination of different attribute values in the partial rule is computed to ensure accuracy. Then, novelty of the partial rule is determined based on the result of comparing the partial rule against *known knowledge* to decide either further split is required or to stop growing the tree at that branch. The node is stopped grown if the novelty measure of the partial rule is not exceeding a user novelty threshold value. Such situation indicates that the partial rule is already discovered at earlier time.
In the following subsections, we explain these tasks in details.

5.2.1 Bayesian Independence Test

The approach utilizes Bayesian approach to guess the most correlated class value in the branch. This step aims at determining the most frequent class value that corresponds to the portion of data used to create the branch. While performing this task, it takes into account a node along with its ancestor.
According to *Baye's Theorem*, if $E_1, E_2, \ldots, E_n$ are mutually disjoint events with $P(E_i) \neq 0$ (i = 1,2, …,n) then for any arbitrary event A which is a subset of $\bigcup_{i=1}^{n} E_i$ such that $P(A) > 0$, we have

$$P(E_i/A) = \frac{P(E_i)\,P(A/E_i)}{\sum_{i=1}^{n} P(E_i)\,P(A/E_i)}$$

Where $P(E_i)$ (i = 1,2, …n) are *Apriori Probabilities*, the probabilities $P(A/E_i)$, i = 1,2, …,n are called *Likelihoods,* and the Probabilities $P(E_i/A)$, I = 1,2, …,n are called *Posterior Probability.*

5.2.2 Accuracy of a Partial Rule

It has been found that one can construct a sufficiently large tree to drive the misclassification error to zero [29]. However, the size of induced tree is a critical issue. The size is often measured as the number of leaves in the tree [4,6]. Therefore, the accuracy concern includes both size of the tree as well as the prediction error rate. The accuracy rate is the ratio of the number of cases for the most frequent class and the total number of cases. The misclassification (error) rate is the additive inverse of the accuracy rate.

In the context of a partially constructed tree, the approach computes the accuracy at each node as the probability that the partial rule classifies correctly given its ancestors with the combination of possible class values.

Given a partial rule $A \rightarrow B$, where A in the antecedent and B is the consequent. Assume that a subset of the training set corresponds to A called cover or Γ_A. Also assume that ς represents a subset of the training set with a particular class value. The classification accuracy of $A \rightarrow B$ [29] is given as follows:

$$Accuracy = \frac{|\Gamma_A \cap \varsigma|}{|\Gamma_A|}$$

where $|\Gamma_A \cap \varsigma|$ denotes the number of tuples that contain both the antecedent and the class value, and $|\Gamma_A|$ is the number of tuples that contain antecedent A.

To make certain that the accuracy is not compromised; the computed accuracy of a partial rule is compared against accuracy threshold value. A partial rule is accepted to be accurate if the computed probability of an attribute with associated value is greater than the user specified accuracy threshold value. This indicates that the data in that partition is homogeneous to an acceptable extent with respect to class value. Otherwise, it is an indication that, this portion of the data in the partition corresponding to that branch is not homogeneous with respect to a class value, which leads to higher classification error. In such case, the approach expands the branch to increase the degree of homogeneity in that partition.

5.2.3 Dynamic Pruning Based on Novelty Criterion

An important feature of the approach classifier is its ability to facilitate pre-pruning based on novelty of the branch given that the branch has acceptable degree of accuracy. The aim of the proposed approach is to reduce the size (complexity) of the decision tree with a guarantee that the resulting tree does not compromise in terms of *accuracy* and provides the user with *novel* classification rules**.**

The approach determines either a branch under construction leads to a novel rule or not. This is established by computing the novelty measure of that branch against *known knowledge*. The computation of novelty measure is given is section 4. A branch is expanded only if its novelty measure is higher than a user specified novelty threshold given that the branch has acceptable degree of accuracy. This is a useful feature in which the user may need to trade off some accuracy for novelty that may arise in some domains where the user wants a rough picture about the domain rather than an optimal classifier that contains a lot of details.

Upon expansion, branches are created for each attribute value and the process continues until either all data in a partition is homogeneous or no attribute remains to split the data.

We map the above concepts in the context of decision tree with the example below:

Example 1

Given a *known knowledge* (kk) rule base that represents rules discovered at time T_1. Consider $R_i \in$ kk has been discovered at time T_1. The aim is to construct a tree in such a way that each branch should have acceptable degree of accuracy and the *novelty measure* is greater than the threshold value (Φ) **.**

Let $R_i = (A=b) \wedge (A_2 = g) \wedge (A_{22} = h) \wedge (A_{221} = k) \longrightarrow Yes$

Assume that the dotted portion of the partial decision tree (Figure 4) is the current node under construction. We use Bayesian approach to test for the correlation of the partial rule for most likely class value. The most correlated (frequent) class value is attached to the partial rule and further computation is performed to compute the accuracy of the partial rule with all possible attribute values of the node A_{22} given its ancestor. The novelty measure of the dotted branch is quantified with respect to the closest rule R_i of *known knowledge* rule base to decide either to further grow the tree at that branch or not. A branch is expanded by creating branches correspond to each attribute value if the computed degree of novelty of the partial rule is greater than a user threshold value.

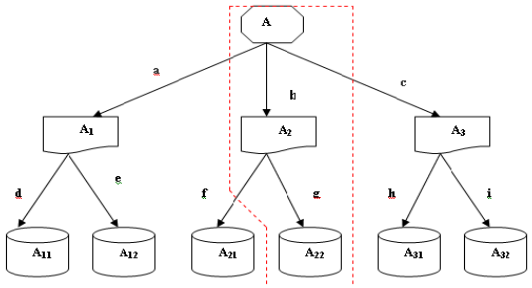

Figure 4. A Partial Decision Tree

6. Experimental Studies

The proposed approach is tested using public datasets available in [12]. We compared the approach with several classifiers for accuracy. The non-incremental classifiers were simulated as incremental algorithms. We found that the proposed approach compares well with respect to the minimum size and high accuracy. The following subsections give details of the experiments performed.

6.1 Experiment One

The first experiment was performed using 'Tic-Tac-Toe' dataset available at [12]. We considered this dataset as evolving with time, and partitioned it into 2 increments: D_1 and D_2 mined at times T_1 and T_2 respectively. We run C5.0 against D_1 and D_2. Figure 5 shows the resulting decision tree. In addition the approach is run against the same partitions. The approach generates PDK rule base using D_1 and subsequently, constructs an incremental decision tree is in light of PDK. The resulting tree is shown in Figure 6.

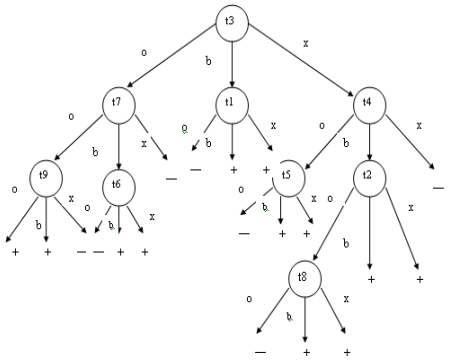

Figure 5. Decision Tree Constructed using C5.0

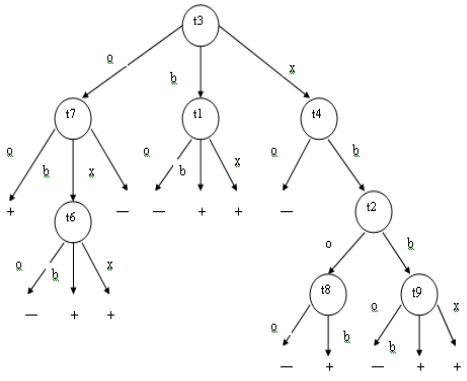

Figure 6. Decision Tree Constructed Using the Proposed Approach

Notice that both C5.0 and our approach construct same decision tree at time T_1 since the *known knowledge* rule base is set to be null at this time. At tome T_2, C5.0 combine D_1 and D_2 to construct a decision tree while the approach construct a tree from D_1 only in light of *known knowledge* discovered at time T_1. the proposed approach constructed the smallest in size and generated novel branches compared to the tree constructed using C5.0.

6.2 Experiment Two

The second experiment was performed using several datasets with different classifier algorithms to study the effectiveness of our proposed classifier. We run CBA [13], C5.0 and our classifiers on these datasets to analyze the size of constructed trees. Our approach has a minimum size compared to other classifiers as shown in Table 1. Figure 7 shows the graphical representation of Table 1.

Dataset	Instances	C5.0	CBA	Our approach
Breast	699	14	45	19
Diabetes	768	30	40	13
Iris	150	4	5	4
Sick	2800	32	50	17
Tic-tac-toe	958	39	28	22
Zoo	101	8	9	9
Heart	270	10	38	11
Hepati	155	6	38	14

Table 1. Comparison of our approach with Different Classifiers in terms of Size

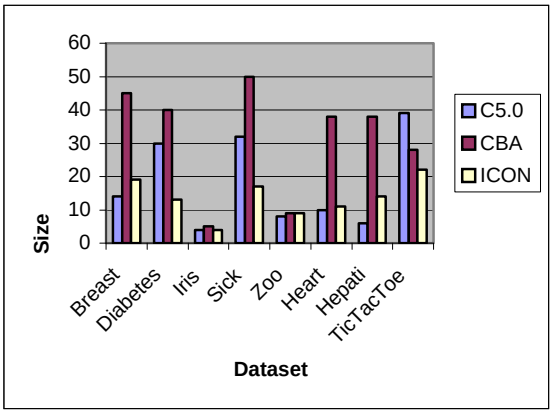

Figure 7. Comparison of our approach with different Classifiers in terms of Size

6.3 Experiment Three

The third experiment was performed using 'Sick' dataset available at [12]. We considered this dataset as

evolving with time, and partitioned it into 3 increments: D_1, D_2 and D_3 mined at times T_1, T_2 and T_3 respectively. We run the proposed approach against the three partitions to compute the accuracy. It has been shown that the accuracy is increased as the number of training instances increases. Figure 1 shows the change in the classification accuracy of the generated classifier.

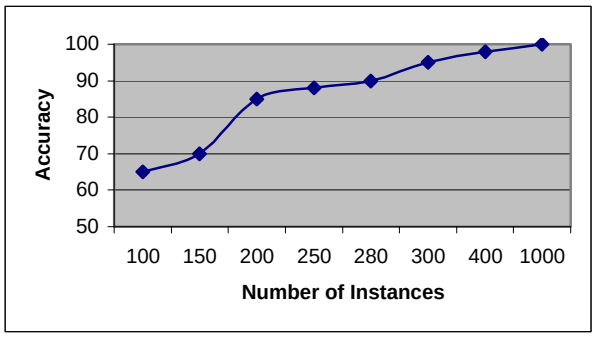

Figure 8. Change in Classification Accuracy on the Sick Dataset, for the proposed Algorithm.

7. Conclusion

In this work we proposed an algorithm for incremental decision tree induction that integrates building and pruning in one stage. The proposed approach is a self-upgrading classifier that utilizes novelty criterion to reflect the user subjectivity and extract patterns, incrementally, from data arrives at different points in time. The classifier makes use of novelty measure as the basis of extracting interesting patterns. This important feature of our approach is attractive and desirable in many real life applications as the volume of data keeps on growing and changing over the time and therefore the user background knowledge is monotonically augmented. This changing environment updates the user understandability and comprehensibility about the domain. The proposed algorithm keeps updating the user domain knowledge as well as discovering novel patterns. As evidence by the empirical evaluation on public data sets, the proposed classifier is able to provide significantly better results than conventional classification model on accuracy, size and reflecting user interesting. Our future work includes enhancing our classifier to create a classification system in which the training model can adapt to a data stream environment

References

1. Breiman L. J., Friedman, J. H., Olshen R. A., and Stone C. J., "Classifications and Regression trees", New York, Chapman and Hall, 1984.
2. Ghrke J., Ganti Vi, Ramakrishnan R. and Loh W-Y, "BOAT-optimistic Decision Tree Construction", In Proceedings of the ACM SIGMOD International Conference on Management of Data, 1999.
3. Quinlan J. R., "C4.5,"Programs for Machine Learning", San Mateo, CA: Morgan Kaufmanns, 1993.
4. Rastogi R., and Shim K., "PUBLIC: A Decision Tree Classifier that Integrates Building and Pruning", In Proceedings of the 24th International Conference on VLDB, 1998.
5. Shafer, J., Aggrawal, R. and Mehta, M., "SPRINT: A scalable parallel classifier for data mining", In 22nd VLDB conference., 1996.
6. Garofalakis M., Hyun D., Rastogi R. and Shim K., "Efficient Algorithms for Constructing Decision Trees with Constraints", Bell Laboratories Tech. Memorandum, 2000
7. Bhatnagar V., Al-Hegami A. S. and Kumar N., "A hybrid approach for Quantification of Novelty in Rule Discovery", In Proceedings of International Conference on Artificial Learning and Data Mining (ALDM'05), Turkey, Feb. 25-27, 2005, pp 39-42.
8. Quinlan J. R., "Simplifying of Decision Trees", International Journal of Machine learning studies, 27, 1987.
9. Clair, St. C., "A Usefulness Metric and its Application to Decision Tree Based Classification", Ph.D. thesis, School of Computer Science, Telecommunications and Information Systems, DePaul University, Chicago, USA, 1999.
10. Mingers, J., "An empirical comparison of pruning Methods for decision tree Induction", Machine learning, 4(2), 1987.
11. Al-Hegami A. S., Bhatnagar, V. and Kumar N., " Novelty Framework for Knowledge Discovery in Databases ", In Proceedings of 6th International Conference on Data Warehousing and Knowledge Discovery (DaWaK 2004), Spain, 2004, pp 48-57.
12. http://kdd.ics.uci.edu/
13. http://www.comp.nus.edu.sg/~dm2/p_download.html
14. Han, J., Kamber, M.:, "Data Mining: Concepts and Techniques", 1st Edition, Harcourt India Private Limited, 2001.

15. Bhatnagar, V., Al-Hegami, A. S. and Kumar, N., "Novelty as a Measure of Interestingness in Knowledge Discovery", In International Journal of Information Technology, Volume 2, Number 1, 2005.
16. Kalles, D. and Morris, T., "Efficient Incremental Induction of Decision Trees", In Machine Learning, 24, pp. 231-241, 1996.
17. Utgoff, P. E., "Incremental Induction of Decision Tress", Machine Learning, 4(2), pp.161-186, (1989).
18. Utgoff, P. E., "ID5: An Incremental ID3", in Proceedings of the 5th International Conference on Machine Learning, pp. 107-120, 1988.
19. Fisher, D., "Knowledge Acquisition via Incremental Conceptual Clustering", in Machine Learning, 2, pp. 139-172, 1987.
20. Schlimmer, J. C. and Fisher, D.," A Case Study of Incremental Concept Induction", In Proceedings of the 5th national Conference on Artificial Intelligence, pp.496-501, Philadelphia, PA, Morgan Kaufman, 1986.
21. van de Velde, W., "Incremental Induction of Topologically Minimal Decision Trees", In Proceedings of the 7th International Conference on Machine Learning, pp. 66-74, 1990.
22. Basu S., Mooney R. J., Pasupuleti K. V., and Ghosh J., "Using Lexical Knowledge to Evaluate the Novelty of Rules Mined from Text", In Proceedings of the NAACL workshop and other Lexical Resources: Applications, Extensions and Customizations, 2001.
23. Bronchi F., Giannotti F., Mazzanti A. and Pedreschi D., "Adaptive Constrain Pushing in Frequent Pattern Mining", In Proceedings of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD03), 2003.
24. Bonchi F., Giannotti F., Mazzanti A. and Pedreschi D., "ExAMiner: Optimized Level-wise Frequent Pattern Mining with Monotone Constrains", In Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM03), 2003.
25. Bonchi F., Giannotti F., Mazzanti A. and Pedreschi D., "Exante: Anticipated Data Reduction in constrained Pattern Mining", In Proceedings of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases (PKDD03), 2003.
26. Pei J. and Han J., "Can We Push More Constraints into Frequent Pattern Mining", In Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2000.
27. Quinlan, J.R., "Induction of Decision trees", In machine learning 1:1. Boston: Kluwer, Academic publishers, 1986.
28. Liu, W.Z. and White A.P., "The Importance of Attribute selection Measures in Decision Tree Induction", Machine Learning, 15, 1994.
29. Duda R. O., Hart P. E. and Stork D. G., "Pattern Classification", 2nd Edition, John Wiley & Sons (Asia) PV. Ltd., 2002.

Ahmed Sultan Al-Hegami received His B.Sc degree in Computer Science from King Abdul Aziz University, Saudi Arabia, MCA (Master of Computer Application) from Jawaharlal Nehru University, New Delhi, India; and Ph.D. degree from University of Delhi, Delhi, India. He is lecturer at the Department of Computer Science, Sana'a University, Yemen. Currently he is assistant professor at the Department of Computer Science, Sana'a University, Yemen. His research interest includes artificial intelligence, machine learning, temporal databases, real time systems, data mining, and knowledge discovery in databases.