Enhanced Decision Tree Algorithm for Discovering Intra and Inter Class Exceptions

Sunil Kumar, Saroj Ratnoo, Jyoti Vashishtha

Abstract*: Decision tree algorithms, being accurate and comprehensible classifiers, have been one of the most widely used classifiers in data mining and machine learning. However, like many other classification algorithms, decision tree algorithms focus on extracting patterns with high generality and in the process, these ignore some rare but useful and interesting patterns that may exist in small disjuncts of data. Such extraordinary patterns with low support and high confidence capture very specific but exceptional behavior present in data. This paper proposes a novel Enhanced Decision Tree Algorithm for Discovering Intra and Inter-class Exceptions (EDTADE). Intra-class exceptions cover objects of unique interest within a class whereas inter-class exceptions capture rare conditions due to which we are forced shift the class of few unusual objects. For instance, whales and bats are examples of intra-class exceptions since these have unique characteristics within the class of mammals. Further, most of the birds are flying creatures, but the rare birds, like penguin and ostrich fall in the category of no flying birds. Here, penguin and ostrich are inter-class exceptions. In fact, without knowing about such exceptional patterns, our knowledge about a domain is incomplete. We have enhanced the decision tree algorithm by defining a framework for capturing intra and inter-class exceptions at leaf nodes of a decision tree. The proposed algorithm (EDTADE) is applied to many datasets from UCI Machine Learning Repository. The results show that the EDTADE has been successful in discovering many intra and inter-class exceptions. The decision tree augmented with intra and inter-class exceptions are more accurate, comprehensible as well as interesting since these provide additional knowledge in the form of exceptional patterns that deviate from the general rules discovered for classification.

***Keywords*: Decision tree induction, Exception discovery, Intra and Inter Class exceptions**

I. INTRODUCTION

A significant amount of research has been carried out for developing efficient classification techniques for knowledge discovery in databases [1]. A decision tree induction algorithm is one of the most powerful algorithms for classification in data mining because of its high accuracy and comprehensibility. A decision tree model not only makes accurate predictions, but it is also very amiable for human insight and analysis for the process of decisions making.

Revised Manuscript Received on September 2, 2019.

Sunil Kumar*, Dept. of Computer Science & Engineering, Guru Jambheshwar University of Sci. & Tech., Hisar, Haryana, India. Email: skvermacse@gmail.com

Saroj Ratnoo, Dept. of Computer Science & Engineering, Guru Jambheshwar University of Sci. & Tech., Hisar, Haryana, India. Email: ratnoo.saroj@gmail.com

Jyoti Vashishtha, Dept. of Computer Science & Engineering, Guru Jambheshwar University of Sci. & Tech., Hisar, Haryana, India. Email: jyoti.vst@gmail.com

A decision tree algorithm is simple and straight forward. The instances in a dataset are used to design the decision tree from root node to leaf nodes. Constructing a decision tree is generally a divide and conquer recursive process followed in top-down manner. The building of the tree starts at the root node with entire training instances along with the associated class labels. Attributes in the dataset are ranked for the test splits based on heuristic measures like Entropy, Info-Gain, Gain-Ratio and Gini Index etc. The attribute that produces the purest partitions along its possible values in the data is preferred, i.e., the attribute that best separates the given instances into individual classes is selected for splitting data. This way dataset is partitioned into different subsets of data, which satisfies the splitting criteria. This process is repeated recursively for each subset of data along the branches of the decision tree until all instances of a subset belong to the same class or no attributes left for further partitioning or no instances are left. Readers can refer [1] for detailed steps for constructing decision trees.

Several decision tree models like CART, CHAID, ID3, C5.0 (J48 in WEKA) have been implemented for the task of classification in data mining [2]. The additional advantage of decision trees is that these are convertible to decision rules. The decision tree classifiers generate rules with high generalization power that cover a lot of examples in the data and the focus of decision tree induction process is on the rules with high support and high confidence. Further, decision trees are usually pruned to avoid over-fitting. The pruning phase may result in the loss of information which may be considered useless regarding classification. On the hindsight, some of these instances at the leaf nodes that get removed during pruning may be of great importance. In fact, some of the pruned instances may form valid exceptions to the generalized rules.

Exceptions are considered interesting nuggets of knowledge in data mining literature [3]. Knowing rules makes us wise but knowing exceptions to the rules makes us perfect. Since the tuples that may form exceptions are very few in number, the conventional decision tree algorithms ignore such rare but highly interesting examples as machine noise either during the induction process itself or later during the pruning step [4].

This paper proposes Enhanced Decision Tree Algorithm that not only discovers decision rules, but also the exceptional conditions that make the discovered rules more comprehensible, complete and interesting. The enhanced algorithm discovers two kinds of exceptions: Intra Class and inter-class. Intra Class exceptions capture the intriguing objects with unique and extraordinary characteristics within their class while Inter Class exception apprehend the exceptional condition(s) in the presence of which a classification rule does not work in a usual way. Ignoring an Inter Class exception leads to misclassifications [5]. To further clarify the distinction between intra and inter class exceptions, let us take some examples of animals. One of the categories of animals is of mammals. Within mammals, fruit bat is a unique mammal that flies. Here, the fruit bat is a unique mammal and is an example of Intra Class exception. Another category in the animals is of fish. Animals with fins are usually classified as fish. However, dolphins and seals are exceptional animals. These animals have fins but fall in the category of mammals because these give birth to babies and breastfeed them. A simple generalized rule- '***If*** fins ***Then*** fish'- would misclassify these animals in the fish category. Here, dolphins and seals are examples of Inter Class exceptions for the above rule. The decision tree algorithm proposed in this paper is an extension of J48 in WEKA and named as Enhanced Decision Tree for Exception Discovery (EDTADE).

In this paper, first, we have illustrated the suggested algorithm on a toy data set, specifically designed for the purpose. For further corroboration, it is also applied to ten datasets from the UCI Machine Learning repository. The results give you an idea about the proposed algorithm which captures a number of intra and Inter Class exceptions.

The rest of the paper is structured as below: Section 2 reviews the relevant literature to contextualize out research. Section 3 provides the problem statement. Section 4 presents the framework for discovering exceptions. It illustrates the finding of intra and Inter Class exceptions with the help of a small sample data set. Section 5 describes the design of the Enhanced Decision Tree Algorithm for Exception Discovery (EDTADE). Section 6 presents the experimental results. Finally, Section 7 concludes this paper.

II. LITERATURE REVIEW

Decision tree algorithms, one of the most popularly used and wide methods for classification, are accurate and easier to understand. Hence, several versions of decision tree algorithm (ID4, ID5, CART, CHAID, C4.5, C5.0) have been developed since its inception in the form of ID3 [6]-[9]. Decision tree algorithms have been compared to other learning algorithms in [10]. The comparative study shows that the recent versions of decision trees like C4.5 and C5.0 have less error rate and more speed. Further, many methods like fast tree-growing algorithm [11] data partitioning [12] and parallelization [13] have also been suggested for scaling-up the decision-tree building process. The rainforest methods have been proposed to develop fast and scalable algorithms to build decision trees [14] with high accuracy. Many approaches for improving decision trees have been enlisted in [15]-[17] and their focus has been on small modifications on the available decision tree models.

Discovering interesting rules has always been a priority of data mining community [18], [19]. Many approaches are found in data mining literature which either depend on filtering interestingness measures or on the user's specific domain knowledge to classify unexpected instances [20], [21]. Another framework for discovering exception has been suggested in the form of ripple down rules which capture exceptions at multiple levels [22]. Suzuki and colleagues in have extracted exceptions as rule pairs and rule triplets for dependence modeling, a data mining task that swaps numerous attributes for class label [23], [24]. Exceptions have been discovered in the form of Censored Production Rules (CPRs) using evolutionary approach in [25], [26]. Vashishtha et al. (2012) and Bala &Ratnoo (2014) have designed genetic algorithms to learn intra and Inter Class exceptions for simple and fuzzy decision rules [5], [27]. However, none of these researches, on the discovery of exceptions stated so far, use decision trees as the rule induction method.

We have only found two research works [28], [16] that deal with exception handling while performing classification using a decision tree algorithm. These works modify the rule generation algorithm to resolve ties between classes that emerge at leaf node during the design process. These ties are problematic in decision tree algorithm when majority of votes are not able to determine the predicted class at a leaf node. To deal with such type of exceptional cases, the authors calculate an influence factor for each attribute and subsequently apply an update procedure to make the final decision.

III. PROBLEM STATEMENT

Decision tree induction algorithm J48 in WEKA like many other decision tree algorithms focus on extracting generalized patterns that have high support and precision. However, the exceptional patterns always have low support and high confidence. Such patterns may be of great interest to us, but these get ignored as noise in the process of decision tree induction. There is an inadequate research for upgrading the learning process of decision trees for handling/capturing exceptions. This paper is an attempt to fill this gap. The emphasis of the work is on enhancing decision tree algorithm (J48 in WEKA) for discovering and appending exceptional clauses to decision rules, in the presence of which general rules either identify unique objects within their specified categories or stop working and misclassify certain rare objects. The first situation identifies Intra Class exceptions whereas the later captures the Inter Class exceptions. A rule with Inter Class exception(s) needs to suggest a corrective measure to avoid misclassifications

IV. INTRA AND INTER CLASS EXCEPTION DISCOVERY FRAMEWORK

An exception is a pattern that deviates from the general rules. A general rule is also referred as default rule in this paper. This section describes the framework for discovering intra and Inter Class exceptions. To make steps of discovering exceptions clear, we illustrate it with the help of a small sample data, a subset of the Zoo dataset available from UCI machine learning repository. The sample data is given in Table-I. It contains 19 objects with 8 mammals, 5 reptiles, 4 insects, 2 amphibian and 1 invertebrate.

Table- I: Sample data a subset of zoo dataset

feathers	Eggs	milk	airborne	backbone	fins	legs	tail	domestic	catsize	class	name of animal
F	T	F	F	T	F	4	F	F	F	amphibian	Tod
F	T	F	F	T	F	4	F	F	F	amphibian	Frog
F	T	F	F	F	F	6	F	F	F	Insect	Flea
F	T	F	T	F	F	6	F	F	F	Insect	Gnat
F	T	F	T	F	F	6	F	T	F	Insect	honey bee
F	T	F	T	F	F	6	F	F	F	Insect	house fly
F	T	F	F	F	F	0	F	F	F	invertebrate	clam
F	F	T	F	T	F	2	F	T	T	mammal	girl
F	F	T	F	T	F	4	T	F	T	mammal	bear
F	T	T	F	T	F	4	T	F	T	mammal	goat
F	F	T	F	T	F	4	T	T	T	mammal	cat
F	F	T	F	T	F	4	T	F	T	mammal	elephant
F	F	T	F	T	F	4	T	T	T	mammal	cow
F	F	T	F	T	F	4	T	F	T	mammal	deer
F	F	T	F	T	T	0	F	F	T	mammal	whale
F	T	F	F	T	F	0	T	F	F	reptile	pitviper
F	F	F	F	T	F	0	T	F	F	reptile	seasnake
F	T	F	F	T	F	0	T	F	F	reptile	slowworm
F	T	F	F	T	F	4	T	F	T	reptile	tortoise
F	T	F	F	T	F	4	T	F	F	reptile	tuatara

A. *Discovering Default Rules*

The knowledge discovered from a decision tree algorithm is represented in the form of a tree and as well in the form of '*If* P_i *Then* D_d' rules where P_i, known as the premise part of independent attributes, is made of conjunctions of a few conditional clauses, and D_d, known as the conclusion part of decision class label for the decision taken. A simple *If-Then* rule is symbolized as below.

$$\text{If P Then } D_k \quad (1)$$

$$\text{where } P = PC_1 \wedge PC_2 \wedge \ldots \wedge PC_n$$

A conditional clause (PC_i) in P is in the form [$A_j op\ V_{jk}$]. The value of i varies from 1 to n, where n indicates total number of conditional clauses present in the premise part, A_j is the j^{th} attribute, *op* is a relational operator, V_{jk} is the k^{th} value of the j^{th} attribute and D_k is the target class in "(1)".

The generality of default rules is usually estimated through two parameters- Precision (δ_1) and Recall (δ_2) as defined below in "(2)" and "(3)".

$$\delta_1 = \frac{|P \cap D_k|}{|P|} \quad (2)$$

$$\delta_2 = \frac{|P \cap D_k|}{|D_k|} \quad (3)$$

where $|P \cap D_k|$ is the cardinality of the set of instances covered by P and D_k;

$|P|$ is cardinality of the set of instances enclosed by P only;
$|D_k|$ is cardinality of the target class.

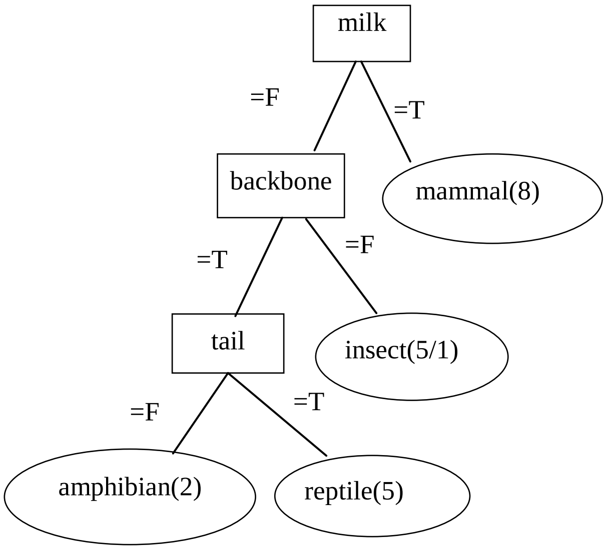

Fig. 1. Decision tree induced by J48 in WEKA on sample data.

General Rules.

R1: ***If*** (milk=true) ***Then*** class=mammal (δ_1=8/8=1, δ_2=8/8=1)

R2: ***If*** (milk=false) && (backbone=false) ***Then*** class=insect (δ_1=4/5=0.8, δ_2=4/4=1)

R3: ***If*** (milk = false) && (backbone=true) && (tail=false) ***Then*** class =amphibian (δ_1=2/2=1, δ_2=2/2=1)

R4: ***If*** (milk = false) && (backbone=true) && (tail=true) ***Then*** class= reptile) (δ_1=5/5=1, δ_2=5/5=1)

For any rule to have some amount of generalization power, values of δ_1 and δ_2 for the rule must meet the criteria of user defined thresholds for these parameters, i.e., $\delta_1 \geq \theta_{pr}$ and $\delta_2 \geq \theta_{rr}$.
Fig. 1 shows the decision tree and the corresponding rules generated by J48 in WEKA using the sample dataset. Assuming $\theta_{pr} = 0.8$ and $\theta_{rr} = 0.7$ as the thresholds for $\delta_1$and δ_2, all the rules qualify to be included in the list of default rules.

B. *Discovering Intra Class Exceptions.*

The Intra Class exceptions determine the special or unique features of an object within its class. A rule with Intra Class exception(s) is represented as below.

$$\boldsymbol{If} P\ \boldsymbol{Then}\ D_k With\ E_s: (\delta_3, \delta_4)$$

For adding Intra Class exceptions to the default rule, two parameters (δ_3 *and* δ_4) are given in "(4)" and "(5)".The constraints on these parameters are also defined below.

$$\delta_3 = \frac{|P \cap E_s \cap D_k|}{|P \cap D_k|} \quad (4)$$

$$\delta_4 = \frac{|P \cap E_s \cap D_k|}{|P \cap E_s|} \quad (5)$$

$$Where\ \ \delta_3 \leq 1 - \theta_{rr};\ \ \delta_4 = 1$$

The parameter δ_3 computes the support of an Intra Class exception correspond to the default rule. The value of δ_4 should always be equal to 1. This constraint ensures the uniqueness of the object, i.e., $P \wedge E_s$never holds true in any of the decision classes other than D_k.
Let us now find Intra Class exceptions to the rule R_1: ***If*** (milk = true ***Then*** mammal). According to the sample data, all mammals have four legs except one. This may form an Intra Class exception to the default rule R1 provided that it meets the constraints on δ_3 and δ_4 .

$$\delta_3 = \frac{|(milk = true) \cap (legs = 2) \cap mammal|}{|(milk = true) \cap mammal|} = \frac{1}{8} = 0.125$$

$$\delta_4 = \frac{|(milk = true) \cap (legs = 2) \cap mammal|}{|(milk = true) \cap (legs = 2)|} = \frac{1}{1} = 1$$

$$= \delta_3 \leq (1 - \theta_{rr}) \leq 0.3\ ; Where\ \theta_{rr} = 0.7$$

According to the above equations, the conditional clause (legs=2) qualifies to be an Intra Class exception for the general rule on mammals.

There is other similar Intra Class exception in the sample data which is given below.

R_{11}: ***If*** (milk = true ***Then*** mammal) ***With*** (legs=2):
$(\delta_3 = 0.125, \delta_4 = 1)$
R_{12}: ***If*** (milk = true ***Then*** mammal) ***With*** (fins=true):
$(\delta_3 = 0.125, \delta_4 = 1)$

The above two Intra Class exceptions are very interesting because these identify unique mammals-girl and whale. in the sample data. Intra Class exceptions are present for the default rules for which $\delta_2 = 1$.

C. *Discovering Inter Class Exceptions.*

The special features that alter the decision of a rule are referred as Inter Class exceptions. A notation of rule with Inter Class exception(s) is represented as below.

$$\boldsymbol{If} P\ \boldsymbol{Then}\ D_k\ \boldsymbol{Unless}\ E_s: (D_j)(\delta_5, \delta_6);$$

In the above rule, D_k is the class of default rule where as D_j is the modified class in the presence of Inter Class exception(s).For augmenting Inter Class exceptions, two additional parameters, δ_5 *and* δ_6, "(6)"and "(7)" along with the two constraints required are defined below.

$$\delta_5 = \frac{|P \cap E_s \cap D_j|}{|P|} \quad (6)$$

$$\delta_6 = \frac{|P \cap E_s \cap D_j|}{|P \cap E_s|} \quad (7)$$

$$Where\ \ \delta_5 \leq 1 - \theta_{pr};\ \delta_6 = 1$$

The parameter δ_5 is the ratio of the number of objects covered altogether by premise, exceptional condition and class D_j to the number of objects covered by the premise of the default rule. The value of δ_6 should always be equal to 1. This restriction ensures the uniqueness of the exceptional clause to class D_j.
Notice that decision tree in Fig. 1 misclassifies an object into insect category. According to the default rule- R_2 (***If*** (milk=false) && (backbone=false) ***Then*** insect)- animals that do not feed milk and have no backbone is classified as insect. An invertebrate in the data sample also have these characteristics and hence gets misclassified as an insect. Here appears a scope for Inter Class exception. Along with the properties given in the default rule, the invertebrate has zero legs. This leads to discovery of Inter Class exception for rule R_2.

R_{21}: (***If*** (milk=false) && (backbone=false) ***Then*** insect ***Unless*** (legs=0) (invertebrate) ($\delta_5 = 0.2,\ \delta_6 = 1$)

$$\delta_5 = \frac{|(\text{milk=false}) \cap (\text{backbone=false}) \cap (\text{legs=0}) \cap \text{invertebrate}|}{|(\text{milk=false}) \cap (\text{backbone=false})|} = \frac{1}{5} = 0.2$$

$$\delta_6 = \frac{|(\text{milk=false}) \cap (\text{backbone=false}) \cap (\text{legs=0}) \cap \text{invertebrate}|}{|(\text{milk=false}) \cap (\text{backbone=false}) \cap (\text{legs=0})|} = \frac{1}{1} = 1$$

$$\text{Where}\ \ \delta_5 \leq (1 - \theta_{pr}) \leq 0.2;\ \theta_{pr} = 0.8$$

Therefore, the clause (legs=0) in this case forms a valid Inter Class exception to the rule R2. The modified rule classifies an animal which does not feed milk, has no backbone and has no legs as invertebrate and not as an insect. The improved decision tree with inter and intra class exceptions is shown in Fig. 2.

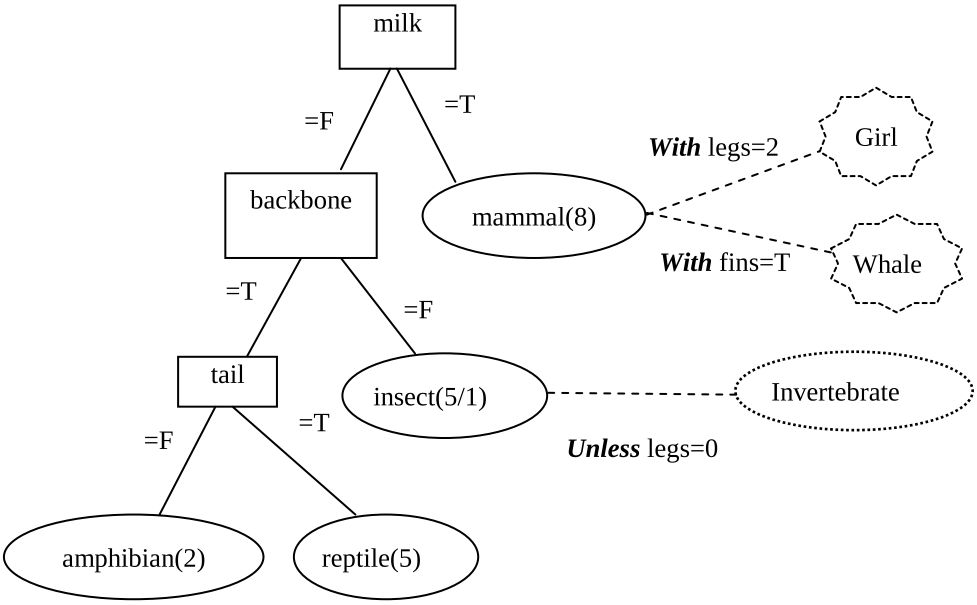

Fig. 2. Modified decision tree with intra and Inter Class exceptions.

V. ENHANCED DECISION TREE ALGORITHM FOR DISCOVERING EXCEPTIONS (EDTADE)

The proposed algorithm appends intra and inter class exceptions according to the framework described in the previous section at every leaf node created by the decision tree algorithm (J48). Whenever a leaf node is created the criteria for intra and Inter Class exceptions are tested and the clauses that meet the criteria are augmented to the tree accordingly.

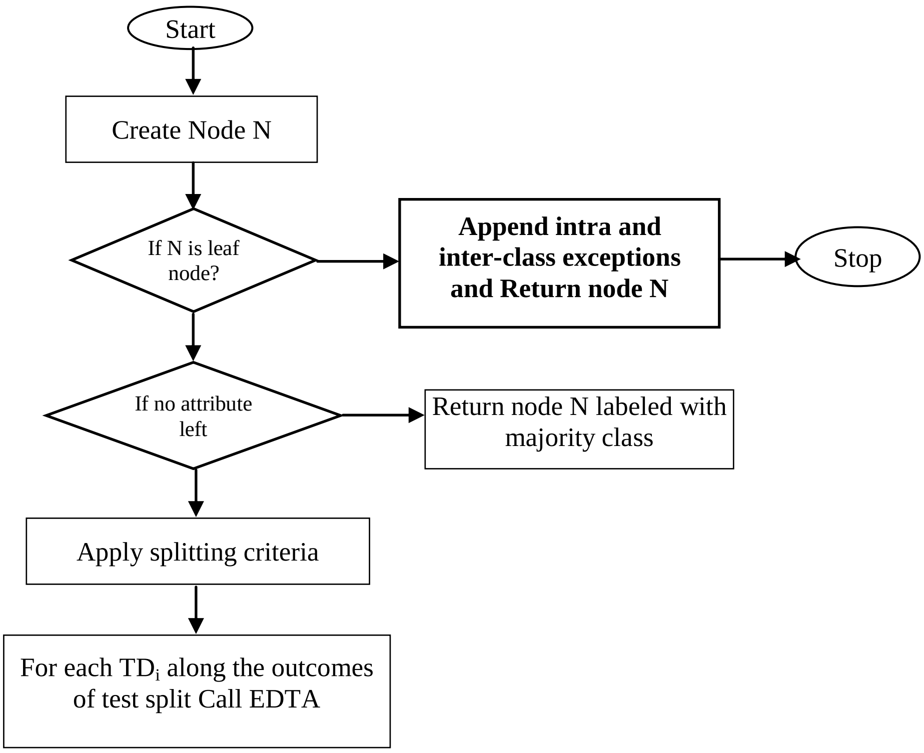

Fig 3. Block diagram of working of EDTADE.

The highlighted block in the overall block diagram in Fig. 3 shows the place where we have made enhancement in the decision tree algorithm (J48). The detailed algorithm is given in Fig.4.

.

Algorithm: Enhanced_Decision_Tree Algorithm.
Input: Training Dataset TD; Attribute List L; Splitting Criterion (Splitting Attribute SA, Split point SP);
Output: A decision tree T with Exceptions (if any) at each leaf node.
Begin
Create a node 'N'
If (N is a leaf node)
For every possible general rule R_i that can be created at the node 'N'
Compute
$\delta_1 = \frac{|P \wedge D_k|}{|P|}$; $\delta_2 = \frac{|P \wedge D_k|}{|D_k|}$
// If all instances belonging to any class D_k, (i.e., $\delta_2 == 1$), fall in node 'N' then look for intra-class exceptions
If($\delta_1 > \theta_P$&& $\delta_2 == 1$)
For each conditional clause *E* present in node 'N'
Compute
$\delta_3 = \frac{|P \wedge E \wedge D_k|}{|P|}$; $\delta_4 = \frac{|P \wedge E \wedge D_k|}{|P \wedge E|}$
If ($\delta_3 < \theta_E$) && ($\delta_4 == 1$) // validation for interclass exceptions
Add intra class exception to node N and to the rule R_i
end If
// If instances in node 'N' belong to different classes then look for inter-class exceptions
If (($\delta_2 > \theta_R$) &&($\delta_1 \geq \theta_P$ && $\delta_1 < 1$)) //conditions for inter class exception
For each conditional clause *E* present in node 'N'
Compute
$\delta_5 = \frac{|P \wedge E \wedge D_j|}{|P|}$; $\delta_6 = \frac{|P \wedge E \wedge D_j|}{|P \wedge E|}$
If ($\delta_5 < (1 - \delta_1)$ && ($\delta_6 == 1$) // fins inter class exception
Add inter class exception to node 'N' and to the rule R_i
end For
end If
end For
end If
Return node N labeled with the class D_k along with intra and/or inter-class exception(s)
If (attribute list L=Ø)
Return 'N' as a leaf node tagged with the majority class in TD // majority voting
Apply feature selection method (TD, L) to discover the "best" splitting criterion SC
Label N with SC
If (SA is discrete valued and multiway splits is allowed) // not restricted to binary trees
L = L – SA // remove splitting attribute
For each outcome i of SC
Let TD_i be the set of data instances in TD satisfying outcome i
If (TD_i=Ø)
Add a leaf node tagged with the majority class label in TD to node 'N'
Else
Add the node returned by Enhanced_Decision_Tree (TD_i, L) to node 'N'
End If
End For
Return N
End

Fig 4. Detailed EDTADE algorithm.

VI. EXPERIMENTAL RESULTS AND PERFORMANCE EVALUATION

We have evaluated the performance of EDTADE on 10 datasets, all taken from the UC Irvine ML Repository. The datasets are summarizes with the number of attributes and instances in Table-II. The algorithm is implemented in JAVA. The proposed enhanced decision tree algorithm is successful in discovering many intra and inter class exceptions along with the general rules from these datasets. Since it is not feasible to list the rules discovered from every datasets due to the space limitation, here, we give the sample rules discovered from Zoo dataset only (Table-III).

Retrieval number :K18160981119//2019©BEIESP
DOI: 10.35940/ijitee.xxxxx.xxxxxx
Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Table-II: Summary of datasets

Dataset Name	#Instances	#Attributes	#Classes
Wisconsin Breast Cancer (BCR)	699	10	2
Cleveland-14-heart-disease (CHD)	296	14	2
Contact-lenses (CLS)	24	5	3
Hepatitis (HPS)	80	20	2
Iris (IRS)	150	5	3
Kr-vs-kp (KVK)	3196	37	2
Lymph (LPH)	148	19	4
Mushroom (MSH)	5628	23	2
Vote (VOT)	232	17	2
Zoo (Zoo)	101	17	7

The second column of Table III shows the rules augmented through intra and inter class exceptions using '*With*' and '*Unless*' keywords respectively. The remaining columns capture the values for the relevant parameters for intra and

Table-III: Rules with Intra and Inter Class exceptions from Zoo dataset

#Rule	**Rules with intra and Inter Class exceptions**	**# Instances Covered**	**#Exceptions**	$\boldsymbol{\delta_3}$	$\boldsymbol{\delta_4}$	$\boldsymbol{\delta_5}$	$\boldsymbol{\delta_6}$
				Intra		**Inter**	
				Class Exceptions			
R1	***If***(feathers = false && milk = true) ***Then*** (decision = mammal) ***With*** (eggs = true) (platypus) ***If*** (feathers = false && milk = true) ***Then*** (decision = mammal) ***With*** (airborne = true) (fruit bat, vampire) ***If*** (feathers = false && milk = true) ***Then*** (decision = mammal) ***With*** (fins = true && tail = false) (seal, dolphin, whale)	41	3	0.024 0.049 0.024	1 1 1	-	-
R2	***If*** (feathers = false && milk = false && backbone = true && fins = false && tail = false) ***Then*** (decision = amphibian)	3	-	-	-	-	-
R3	***If*** (feathers = false && milk = false && backbone = true && fins = false && tail = true) ***Then*** (decision = reptile) ***Unless*** (aquatic = true && legs = 4) (decision = amphibian) (newt) ***If*** (feathers = false && milk = false && backbone = true && fins = false && tail = true)***Then*** (decision = reptile) ***With*** (breathes = false) (sea snake)	5	2	0.167	1	0.830	0.17
R4	***If*** (feathers = false && milk = false && backbone = true && fins = true)***Then*** (decision = fish) ***With*** (domestic = true) (carp)	13	1	0.077	1	-	-
R5	***If*** (feathers = false && milk = false && backbone = false && airborne = false && predator = true) ***Then*** (decision = invertebrate) ***With*** eggs = false) (scorpion)	8	1	0.125	1	-	-
R6	***If*** (feathers = false && milk = false && backbone = false && airborne = false && predator = false && legs <= 2) ***Then*** (decision = invertebrate)	2	-	-	-	-	-
R7	***If*** (feathers = false && milk = false && backbone = false && airborne = false && predator = false && legs > 2) ***Then*** (decision = insect)	2	-	-	-	-	-
R8	***If*** (feathers = false && milk = false && backbone = false && airborne = true) ***Then*** (decision = insect) ***With*** predator = true) (ladybird)	6	1	0.167	1	-	-
R9	***If*** (feathers = true) ***Then*** (decision = bird) ***With*** (domestic = true) (chicken, dove, parakeet)	20	3	0.15	1	-	-

Inter Class exceptions. The enhanced decision tree algorithm discovers 9 rules overall for Zoo dataset. These rules accommodate seven Intra Class exceptions and one Inter Class exception. Due to the discovery of Intra Class exceptions, EDTADE is able to underline mammals of special interest like platypus, fruit bat and vampire, whale, and seal etc. These mammals have one or the other unique characteristic(s). Platypus is a typical mammal because it lays eggs. Fruit bat and vampire are rare mammals that fly. Dolphin, whale and seal are distinctive mammals with fins. Discovery of such exceptions is certainly interesting and adds an extra dimension to the pool of our knowledge.

EDTADE has also discovered one Inter Class exception as well. Rule 5 in Table-III reveals that most of the animals that have backbone and tail but have no feathers and no fins, and do not breast feed, fall in the category of reptiles. However, the rule doesn't apply to '*newt*'. '*Newt*' is a small semi-aquatic amphibian with four legs which looks like a frog as well as a lizard. It is, in fact, an amphibian with all the properties stated in the rule for reptiles. In the absence of the Inter Class exception, '*newt*' would be misclassified as reptile. Discovery of Inter Class exceptions spares us against such misclassification of rare animals. The '*Unless*' clause present in the rule directs us to correct classification of '*newt*' into the amphibian class.

Predicted

Actual	a	b	c	d	e	f	g
a	41	0	0	0	0	0	0
b	0	13	0	0	0	0	0
c	0	0	20	0	0	0	0
d	0	0	0	10	0	0	0
e	0	0	0	0	8	0	0
f	0	0	0	0	0	3	**1**
g	0	0	0	0	0	0	**5**

(a)

Predicted

Actual	a	b	c	d	e	f	g
a	41	0	0	0	0	0	0
b	0	13	0	0	0	0	0
c	0	0	20	0	0	0	0
d	0	0	0	10	0	0	0
e	0	0	0	0	8	0	0
f	0	0	0	0	0	**4**	**0**
g	0	0	0	0	0	0	**5**

(b)

a: Mammal; b: Fish; c: Birds; d: Invertebrate; e: Insects; f: Amphibian; g: Reptiles

Fig. 5. (a) Confusion matrix produced by J48. (b) Confusion matrix produced by EDTADE.

The simple decision tree algorithm J48 provided in WEKA misclassifies '*newt*' as reptile. It is visible from the confusion matrix in Fig. 5(a) produced by this algorithm whereas EDTADE classifies it correctly as shown in Fig. 5 (b).

Hence, EDTADE is more accurate classifier. The increase in accuracy of any classifier produced by EDTADE will obviously depend on the number of Inter Class exceptions present in the data. The rules with Intra Class exceptions are interesting due to their unique characteristics while rules with inter class exceptions have the capability to adjust its decision in the existence of exceptional conditions. Table-IV summarizes results for all the experimental datasets in terms of total number of rules, number of Intra Class exceptions and Inter Class exceptions discovered. The table also compares the accuracy of J48 and EDTADE. Table-IV shows that EDTADE discovers intra and Inter Class exceptions to default rules and it achieves a slight increase inaccuracy for all the datasets. Table-V compares the accuracy of EDTADE with other rule-based classification algorithms. According to Table-V, EDTADE is a clear winner in accuracy. Moreover, the rules discovered by EDTADE are, complete and interesting since these rules capture unexpected information in the form of intra and inter class exceptions Further, these rules are comprehensible and semantically meaningful too.

Table-IV: Summary of experimental results for all datasets

Sr. No	Datasets	#Rules	#Intra-Class Exceptions	#Inter-Class Exceptions	Accuracy J48	Accuracy EDTADE
1	BCR	10	20	8	91.180	**98.975**
2	CHD	19	3	3	89.189	**90.878**
3	CLS	04	0	1	91.667	**95.833**
4	HPS	07	2	2	96.250	**98.750**
5	IRS	03	4	2	96.000	**98.000**
6	KVK	31	12	5	99.655	**100.00**
7	LPH	21	3	3	93.243	**95.270**
8	MSH	20	6	3	98.522	**100.00**
9	VOT	02	10	2	96.982	**100.00**
10	ZOO	09	1	1	99.010	**100.00**

Retrieval number :K18160981119//2019©BEIESP
DOI: 10.35940/ijitee.xxxxx.xxxxxx

Published By:
Blue Eyes Intelligence Engineering
& Sciences Publication

Table-V: Compares the accuracy of EDTADE and rule-based algorithms

Sr. no.	Datasets	Accuracy (%)			
		Decision Table	*JRip*	*PART*	*EDTADE*
1	BCR	92.97	97.80	95.90	**98.98**
2	CHD	79.05	79.39	78.72	**90.88**
3	CLS	87.50	87.50	91.67	**95.83**
4	HPS	83.75	83.75	83.75	**98.75**
5	IRS	97.41	**98.27**	**98.27**	**98.00**
6	KVK	97.22	99.19	99.06	**100.00**
7	LPH	81.76	87.84	95.27	**95.27**
8	MSH	**100**	**100**	**100**	**100.00**
9	VOT	97.41	98.28	98.28	**100.00**
10	ZOO	91.09	98.02	99.01	**100.00**

VII. CONCLUSION & FUTURE WORK

The existing decision tree algorithms capture default rules with high generalization power, and these do not deal with exceptions. Since exceptions have very less support, these get ignored in decision tree building process itself or get pruned at a later stage. In this paper, we have enhanced the decision tree algorithm (J48) available in WEKA to accommodate intra and Inter Class exceptions. Intra Class exceptions capture the unique instances within a category whereas Inter Class exceptions accommodate the unusual circumstances in the presence of which a default rule stops working in a usual way and has to take an alternative decision. The experimental results corroborate that the enhanced decision tree algorithm is able to capture many exceptions that make decision tree and the corresponding rules more accurate, comprehensible and interesting. The proposed algorithm has a vast scope in fault diagnosis, medical diagnosis and in identifying fraudulent financial transactions etc.

There are two issues with the EDTADE in its current form: 1) Some noisy samples can appear as exceptions. At times it may be difficult to differentiate between valid exceptions and what has appeared as an exception due to one or two noisy instances. Help from domain expert is required to filter the noise from the valid exceptions. 2) Discovering exceptions is guided by many parameters starting from δ_1 to δ_6. What makes an exception and what doesn't depend on the values of these parameters. Tuning these parameters is a difficult task. In future, we would like to find a way to set these parameters to appropriate values so that we can capture most of the exceptions. Moreover, the proposed algorithm needs to be further tested on real-world datasets from other domains.

REFERENCES

1. J. Han and M. Kamber, *Data mining: concepts and techniques*, 2nd ed. Amsterdam; Boston: San Francisco, CA: Elsevier; Morgan Kaufmann, 2006.
2. S. B. Kotsiantis, "Decision Trees: A Recent Overview," *Artif. Intell. Rev.*, vol. 39, no. 4, pp. 261–283, Apr. 2013.
3. E. Suzuki, "Discovering Interesting Exception Rules with Rule Pair," in *Proceedings of the ECML/PKDD workshop on advances in inductive rule learning*, vol. 20, pp. 163–178, 2004.
4. Yogita, Saroj, D. Kumar, and Vipin, "Rules + Exceptions: Automated Discovery of Comprehensible Decision Rules," in *2009 IEEE International Advance Computing Conference*, pp. 1479–1484, 2009.
5. J. Vashishtha, D. Kumar, and S. Ratnoo, "An Evolutionary Approach to Discover Intra– and Inter–Class Exceptions in Databases," *Int. J. Intell. Syst. Technol. Appl.*, vol. 12, no. 3–4, pp. 283–300, Jan. 2013.
6. J. R. Quinlan, "Induction of Decision Trees," *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, Mar. 1986.
7. P. E. Utgoff, "Incremental Induction of Decision Trees," *Mach. Learn.*, vol. 4, no. 2, pp. 161–186, Nov. 1989.
8. P. E. Utgoff, "ID5: An Incremental ID3," in *Machine Learning Proceedings 1988*, J. Laird, Ed. San Francisco (CA): Morgan Kaufmann, pp. 107–120, 1988.
9. P. E. Utgoff, "An Improved Algorithm for Incremental Induction of Decision Trees," in *Machine Learning Proceedings 1994*, W. W. Cohen and H. Hirsh, Eds. San Francisco (CA): Morgan Kaufmann, pp. 318–325, 1994.
10. T.-S. Lim, W.-Y. Loh, and Y.-S. Shih, "A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-Three Old and New Classification Algorithms," *Mach. Learn.*, vol. 40, no. 3, pp. 203–228, Sep. 2000.
11. J. Su and H. Zhang, "A Fast Decision Tree Learning Algorithm," in *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, Boston, Massachusetts, pp. 500–505, 2006.
12. A. A. Gill, G. D. Smith, and A. J. Bagnall, "Improving Decision Tree Performance Through Induction- and Cluster-Based Stratified Sampling," in *Intelligent Data Engineering and Automated Learning – IDEAL 2004*, Springer, Berlin, Heidelberg, pp. 339–344, 2004.
13. O. T. Yıldız and O. Dikmen, "Parallel Univariate Decision Trees," *Pattern Recognit. Lett.*, vol. 28, no. 7, pp. 825–832, May 2007.
14. J. Gehrke, R. Ramakrishnan, and V. Ganti, "RainForest—A Framework for Fast Decision Tree Construction of Large Datasets," *Data Min. Knowl. Discov.*, vol. 4, no. 2–3, pp. 127–162, Jul. 2000.
15. D. R. Carvalho and A. A. Freitas, "A Hybrid Decision Tree/Genetic Algorithm Method for Data Mining," *Inf. Sci.*, vol. 163, no. 1, pp. 13–35, Jun. 2004.
16. A. A. B. Subramanian, S. Pramala, B. Rajalakshmi, and R. Rajaram, "Improving Decision Tree Performance By Exception Handling," *Int. J. Autom. Comput.*, vol. 7, no. 3, pp. 372–380, Aug. 2010.
17. T. G. Dietterich, "Ensemble Methods in Machine Learning," in *Multiple Classifier Systems*, pp. 1–15, 2000.
18. L. Geng and H. J. Hamilton, "Interestingness Measures for Data Mining: A Survey," *ACM ComputSurv*, vol. 38, no. 3, Sep. 2006.
19. J. Vashishtha, D. Kumar, and S. Ratnoo, "Revisiting Interestingness Measures for Knowledge Discovery in Databases," in *2012 Second International Conference on Advanced Computing Communication Technologies (ACCT)*, pp. 72–78, 2012.

20. E. Suzuki and J. M. Żytkow, "Unified Algorithm for Undirected Discovery of Exception Rules," in *Principles of Data Mining and Knowledge Discovery*, Springer, Berlin, Heidelberg, pp. 169–180, 2000.
21. G. Garima and J. Vashishtha, "A Study of Interestingness Measures for Knowledge Discovery in Databases—A Genetic Approach," in *Computational Intelligence in Data Mining - Volume 2*, pp. 69–79, 2015.
22. P. Compton *et al.*, "Ripple Down Rules: Turning Knowledge Acquisition into Knowledge Maintenance," *ArtifIntell Med*, vol. 4, no. 6, pp. 463–475, Dec. 1992.
23. E. Suzuki, "Discovering Interesting Exception Rules with Rule Pair," p. 16,2004.
24. J. Vashishtha, D. Kumar, and S. Ratnoo, "Revisiting Interestingness Measures for Knowledge Discovery in Databases," in *Second International Conference on Advanced Computing Communication Technologies*, pp. 72–78, 2012.
25. K. K. Bharadwaj and B. M. Al-Maqaleh, "Evolutionary Approach for Automated Discovery of Censored Production Rules," vol. 1, no. 10, p. 6, 2007.
26. S. Saroj and K. K. Bharadwaj, "A Parallel Genetic Algorithm Approach for Automated Discovery of Censored Production Rules," in *Proceedings of the 25th Conference on Proceedings of the 25th IASTED International Multi-Conference: Artificial Intelligence and Applications*, Anaheim, CA, USA, pp. 435–441, 2007.
27. R. Bala and S. Ratnoo, "A Genetic Algorithm Approach for Discovering Tuned Fuzzy Classification Rules with Intra- and Inter Class Exceptions," *J. Intell. Syst.*, vol. 25, no. 2, pp. 263–282, 2016.
28. S. Appavu alias Balamurugan and R. Rajaram, "Effective solution for unhandled exception in decision tree induction algorithms," *Expert Syst. Appl.*, vol. 36, no. 10, pp. 12113–12119, Dec. 2009.

AUTHORS' PROFILE

Sunil Kumar is working as an Assistant Professor at the Department of Computer Science & Engineering (CSE) in Guru Jambheshwar University of Science and Technology (GJUS&T), Hisar, India. He is pursuing PhD in Computer Science and Engineering from the Department of CSE,GJUS&T, Hisar. His research interests include Data Mining and Soft Computing. He has published four papers.

Saroj Ratnoo received M.Sc. in Computing Science from Birkbeck College, University of London, UK in 1994. She joined the Department of Computer Science and Engineering, (GJUS&T), Hisar in 1996 as an Assistant Professor. She completed her Doctorate degree from School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, India in 2010. She continues to work in Department of Computer Science and Engineering, (GJUS&T), Hisar, India as Professor. Her research interests include Nature Inspired Algorithms, Data Mining and Machine Learning. She has published 30 papers.

Jyoti Vashishtha has done Master of Computer Applications from Himachal Pradesh University, Shimla, India in 1994. She joined the Department of Computer Science and Engineering, (GJUS&T), Hisar in 2004 as Assistant Professor. She completed her Doctorate degree from Department of Computer Science and Engineering, (GJUS&T), Hisar, India in 2014. She continues to work in Department of Computer Science and Engineering, (GJUS&T), Hisar, India as Associate Professor. Her research interests include Knowledge Discovery and Data Mining, Evolutionary Algorithms and Machine Learning. She has published 22 papers