Research Article

Asma' Amro, Mousa Al-Akhras*, Khalil El Hindi, Mohamed Habib, and Bayan Abu Shawar

Instance Reduction for Avoiding Overfitting in Decision Trees

<https://doi.org/10.1515/jisys-2020-0061>

Received Jun 16, 2020; accepted Oct 04, 2020

Abstract: Decision trees learning is one of the most practical classification methods in machine learning, which is used for approximating discrete-valued target functions. However, they may overfit the training data, which limits their ability to generalize to unseen instances. In this study, we investigated the use of instance reduction techniques to smooth the decision boundaries before training the decision trees. Noise filters such as ENN, RENN, and ALLKNN remove noisy instances while DROP3 and DROP5 may remove genuine instances. Extensive empirical experiments were conducted on 13 benchmark datasets from UCI machine learning repository with and without intentionally introduced noise. Empirical results show that eliminating border instances improves the classification accuracy of decision trees and reduces the tree size, which reduces the training and classification times. In datasets without intentionally added noise, applying noise filters without the use of the built-in Reduced Error Pruning gave the best classification accuracy. ENN, RENN, and ALLKNN outperformed decision trees learning without pruning in 9, 9, and 8 out of 13 datasets, respectively. The datasets reduced using ENN and RENN without built-in pruning were more effective when noise was intentionally introduced in different ratios.

Keywords: Decision Trees, Overfitting, Pruning, Instance Reduction, Noise filtering

1 Introduction

Decision trees (DT) learning is used for approximating discrete-valued target functions. The learned function is represented as a tree. New instances are classified using DT by taking decisions and a path down the tree from the root to a leaf node representing a class value (label).

Overfitting is a common problem in machine learning algorithms. It means that while a classifier has a high classification accuracy on the training data, it fails to generalize well over unseen instances. It is the ability of a classifier to deal with unseen (new) instances is what matters in ML [1].

Overfitting is caused by three main reasons that are common in real-life applications: 1) the existence of noisy instances that are not labeled correctly, 2) having a too small number of training examples, which make them not enough to produce a representative sample of the true target function, and 3) over-learning [2].

***Corresponding Author: Mousa Al-Akhras:** Computer Information Systems Department, King Abdullah II School for Information Technology, The University of Jordan, Amman 11942, Jordan; Computer Science Department, College of Computing and Informatics, Saudi Electronic University, Riyadh 11673, Saudi Arabia; Email: m.akhras@seu.edu.sa, mousa.akhras@ju.edu.jo

Asma' Amro: Computer Information Systems Department, King Abdullah II School for Information Technology, The University of Jordan, Amman 11942, Jordan

Khalil El Hindi: Department of Computer Science, College of Computer and Information Sciences, King Saud University, Riyadh 11543, Saudi Arabia

Mohamed Habib: Faculty of Engineering, Port Said University, Port Said, 42526 Egypt; Computer Science Department, College of Computing and Informatics, Saudi Electronic University, Riyadh 11673, Saudi Arabia

Bayan Abu Shawar: Cybersecurity Department, College of Engineering, Al Ain University, Al Ain, UAE

Different overfitting mitigation methods have been proposed in the literature for different machine learning algorithms. This study proposes a solution to the overfitting problem in decision trees learning. The proposed solution is based on pre-pruning the DT using instance reduction techniques to reduce the effect of overfitting problem without degrading the classification accuracy of the DT.

Li [3] proposed a system for recognizing the person's physical activity based on generating a DT from data collected from wearable devices. Since the size of the data is very big, a pruning process was applied to cut unnecessary branches of the tree and to improve the accuracy. Another study by Metting *et al.* [4] was based on implementing a DT from huge real-life primary care population data. A pruning step was used in order to have a simplified version of the generated tree that can help in daily clinical practice for doctors.

An optimized algorithm for building DT was introduced for natural language processing (NLP) tasks [5]. Since big data was required in NLP, the proposed algorithm applied pruning to delete nodes from overfitted models, which speeded up the classification process and increased the accuracy. Thus, the pruning process is very important to build a simpler DT and to overcome the overfitting problem.

Instance reduction algorithms have been extensively used in Instance-Based Learning (IBL) algorithms. During training, IBL stores the training data, and when a query instance is to be classified, the memory is searched for the most similar instance(s). Then these instances are used to classify the query instance. A distance function is used to determine the similarity between the query instance and the training instances. The nearest neighbors are then used to classify the query instance. In the k Nearest Neighbors (kNN) algorithm, for example, the most common class in the k nearest neighbors is assigned to the query instance. The accuracy of IBLs usually increases with the size of the training dataset at the expense of a large memory requirement and long classification time. Instance reduction algorithms were developed to identify and retain the most relevant instances and thus reduce the memory requirement while maintaining good classification accuracy [6].

In this study, seven instance reduction algorithms are utilized to filter out noise as pre-pruning techniques before a DT tree learning algorithm is applied. Empirical experiments were conducted using 13 benchmarked datasets from UCI machine learning repository to study the effectiveness of instance reduction in keeping the DT from overfitting the training data. The results were compared to DT built-in Reduced Error Pruning.

Our empirical comparison involves experiments with and without adding noise. We performed experiments to cover the following 4 main cases:

- 1) Using the full dataset without applying the built-in pruning and without applying any instance reduction algorithm.
- 2) Using the full dataset with pruning using the reduced error pruning and without applying any instance reduction algorithm.
- 3) Using reduced datasets, obtained using an instance reduction, to construct a DT and without using reduced error pruning.
- 4) Both instance reduction algorithms and pruning were applied before and after constructing the DTs.

Results show that when there is no intentionally added noise, the use of noise filters, namely Edited Nearest Neighbor Algorithm (ENN) and Repeated ENN (RENN), without the use of Reduced Error pruning gave the best results in terms of classification accuracy. The resulting DT outperformed the corresponding DT obtained using the full datasets in 9 out of 13 datasets. While using ALLKNN to filter out the noise outperformed the corresponding DT obtained using the full datasets in 8 datasets. Similar performance gains were achieved using 5%, 10%, and 20% noise ratios; noise filters without built-in pruning proved to give better results than the full datasets.

Noise filters gave the best performance along with DROP3 and DROP5 in datasets without intentionally added noise. With 5%, 10%, and 20% noise ratios, the use of DROP3 and DROP5 noise filters to filter out the noise gave significantly better results compared to Reduced Error Pruning.

The rest of this paper is organized as follows: Section 2 presents background information that is necessary to understand the proposed solutions. Section 3 overviews the related work. Section 4 presents the followed research methodology of the proposed solutions. Section 5 discusses the results of the conducted empirical experiments. Section 6 concludes the paper by summarizing the findings and presenting possible avenues for future work.

2 Background

This section presents an overview of decision tree learning focusing on the overfitting and the classical techniques used to mitigate its effect. In this section, we also review some of the most widely used instance reduction algorithms.

2.1 Decision tree learning algorithms

Decision trees learning is used for approximating discrete-valued target functions. The learned function is represented by a tree. Instances are classified using decision trees by taking a path down the tree from the root to a leaf node representing a class value. In a decision tree, attributes are represented by internal nodes, while their values are used to label the branches descending below them. Leaf nodes represent the target function values (class values).

Iterative Dichotomiser 3 (ID3) [7] and C4.5 [8] are the most common decision trees learning algorithms. ID3 goal is to determine which attribute best classifies instances by calculating the information gain for each attribute. Information gain of an attribute is the reduction in entropy when instances are split based on that attribute. It measures how well a specific attribute separates the training examples based on their target classification [2]. While entropy specifies the minimum number of bits needed to encode the classification accuracy of an instance [9]. C4.5 is an extension of ID3 that deals with unavailable values, continuous attribute value ranges, post-pruning of decision trees, and rule derivation. Missing attributes can be classified in the tree by estimating the probability of all possible results [8].

Decision trees have a powerful representation capability and can represent any Boolean function. However, they may overfit the training data and fail to generalize well as a result [10, 11]. The primary cause of overfitting is noise and atypical instances that may not represent the distribution of instances well. However, having noise in the training data in real-life applications is unavoidable [12].

There is a positive correlation between the size of a decision tree and the likelihood that it overfits the training data [2]. Therefore, decision tree pruning techniques were designed to mitigate this effect by reducing the size of a decision tree [2].

Overfitting avoidance techniques aim at producing smaller decision trees, which can be done by either 1) Stopping growing the tree when doing so is not based on sufficient data 2) or by growing the tree then post-pruning it. Examples of pruning techniques include 1) Reduced Error Pruning (REP) Technique, and 2) Rule Post Pruning [8]. In REP technique, nodes are pruned iteratively, starting with removing the nodes if doing so does not harm the classification accuracy. Alternatively, Rule Post Pruning technique starts by converting the decision tree into an equivalent set of rules, then the conditions (antecedents) of the rules are eliminated if doing so does not harm the classification accuracy.

What we propose in this paper is a third method to avoid the overfitting problem by eliminating outliers using some instance reduction techniques. The technique can be viewed as a pruning method of decision trees. This method proved to be very successful in improving the classification accuracy and reducing the training time of Artificial Neural Networks (ANN) [1, 13].

2.2 Instance reduction algorithms

Instance reduction algorithms have been extensively used in Instance-Based Learning (IBL). During training, IBL stores the training data, and when a new instance is to be classified, they search their memory for the most similar instance(s) using a distance function and use the class with the majority of votes as the predicted class for the new instance.

Instance reduction algorithms attempt to identify and retain only the most relevant instances to reduce memory requirement and maintain the classification accuracy.

Instance reduction algorithms aim to identify which instances to retain and which instances to remove from a dataset to reduce memory requirement and maintain the classification accuracy.

Instance reduction can produce a learning model with enhanced capabilities such as: shorter learning process and scaling up to large data sources [14]. Examples of these techniques include [6]:

- 1) Noise Filters: They tend to remove the instances that are close to the edges of a decision boundary, as these instances tend to be noisy instances. They retain the internal instances, and thus the amount of reduction they achieve is usually quite limited.
 - The Edited Nearest Neighbor Algorithm (ENN): the algorithm removes an instance if it has a different class than the class of the majority of its k nearest neighbors. The rationale behind this is such an instance is probably a noisy instance or a near border instance, and using it to classify new instances may lead to incorrect classification.
 - The Repeated ENN (RENN): the algorithm repeats ENN several rounds until no more instances can be removed. In other words, until all retained instances are consistent with their k neighbors (have the same class).
 - All k -NN: this algorithm is also an extension of the ENN. In iteration i , it flags all instances that are not correctly classified by their i th nearest neighbors. This process is repeated k times after that all bad instances are removed.
- 2) Instance Reducers:
 - Encoding Length (ELGrow): The algorithm uses an encoding length heuristic to determine how well the reduced set represents the whole training set. It simply adds an instance to the reduced set if that results in a lower cost than not adding it. This growing phase is followed by a pruning phase where instances are removed from the reduced set if that lowers the cost.
 - Encoding Length (Explore): It performs the ELGrow algorithm first, and then it performs 1000 mutations in an attempt to increase the classifier's efficiency. In each mutation, the reduced set is modified by deleting, inserting, or replacing an instance. The change is only retained if it does not increase the cost of the classifier.
- 3) Instance Reducers with Noise Filtering phase:
 - DROP3: the algorithm uses a noise-filtering pass, such as ENN, to remove noisy instances first. The instances are sorted based on the distance to their nearest enemy (nearest neighbor with a different class). Then instances that are far away from their nearest enemy are removed first. Therefore, this algorithm tends to remove center points and retain border points; this why starting by removing the noisy instances using ENN is crucial.
 - DROP5: this algorithm considers removing first the instances that are close to their nearest enemy, and proceeds outward. This causes most internal instances to be removed. After this pass, instances are then checked for removal, beginning at the instance furthest from its nearest enemy. An instance is removed if at least as many of the instances, that have it among their k nearest neighbors would be classified correctly without it. This is performed repeatedly until no further improvement can be made.

2.3 Heterogeneous value difference metric (HVDM)

Various distance functions can be used to calculate the distance between two instances. Euclidean distance function and Value Difference Metric (VDM) can be used for continuous and nominal attributes, respectively, but not for both types of attributes.

Heterogeneous Value Difference Metric (HVDM) is used in our experiments as it is appropriate for heterogeneous applications that have both types of attributes. It is also capable of calculating distance when there

are missing values. HVDM basically calculates the distance between two input vectors x and y as follows:

$$HVDM(x, y) = \sqrt{\sum_{a=1}^m d_a^2(x_a, y_a)} \quad (1)$$

Where: m : is the number of attributes. The function $d_a(x_a, y_a)$ returns a distance between the two values x_a and y_a of vectors x and y for attribute a . Depending on the type of attribute, it uses the Euclidean distance function (equation 2), if the attribute is numeric or the VDM distance metric (equation 3), if the attribute is nominal.

$$d(x_a, y_a) = (x_a - y_a) / (x_{max} - y_{min}) \quad (2)$$

$$d(x_a, y_a) = vdm(x_a, y_a) = \sqrt{\sum_{c=1}^C (P(c | x_a) - P(c | y_a))^2} \quad (3)$$

Where:

- x and y are two vectors (documents); typically, one is a training instance, and the other is a vector that needs to be classified.
- x_a and y_a are the values of attribute a in the vectors x and y , respectively.
- m is the number of attributes.
- C is the number of classes (document categories).

3 Related Work

El Hindi & Al-Akhras [1] proposed a method to reduce the probability of overfitting in neural networks training. The proposed method smooths the decision boundary by eliminating the training set's idiosyncrasies by filtering near-border (noisy) instances. Before they trained an Artificial Neural Network, the authors applied instance reduction algorithms on the training set to eliminate near-border examples. They performed two sets of experiments; in the first set they used the original datasets. In the second set, they introduced classification noise with various ratios to these datasets. The used instance reduction algorithms are: 1) Noise Filters (ENN, RENN, ALLKNN), 2) Instance Reducers (ELGrow, Explore), and 3) Instance Reducers with Noise Filtering (DROP3, DROP5). Researchers showed that eliminating near-border, and noisy instances improves learning accuracy of a neural network [1]. This also reduces the number of epochs needed by ANN to learn.

An empirical multi-strategy learning algorithm that depends on the unification of instance-based learning and rule induction was proposed by Domingos [15]. This approach has the following two features: 1) no distinction between rules and instances and 2) this strategy consists of a single algorithm that can be viewed (depending on its behavior) as either instance-based learning or rule induction. Thirty datasets were used, the researcher chose the datasets to be representative for different types of attributes, sizes, and domains. Domingos presented the results of the proposed strategy in [15] that are only related to the comparison with decision trees. The proposed multi-strategy learning algorithm produced higher accuracy than using C4.5 alone; because it helps to avoid overfitting. C4.5 needed smaller space (i.e., produced smaller sized outputs), while the proposed strategy is faster. Domingos showed that the new strategy gives better accuracies in most used datasets.

Czarnowski and Jędrzejowicz [16] proposed a new approach for instance reduction in a dataset based on grouping of instances into clusters. Grouping is performed by calculating the similarity coefficient for each instance in the dataset. Only a limited number of instances are then used to be part of the new reduced dataset. Experiments were held on multi-database and also on mono-database mining. Researchers used C4.5 algorithm with 10-folds cross-validation approach. The performance criterion is the generalization accuracy. Results showed that the proposed approach results are independent of the dataset location. When moving instances into their corresponding clusters, multi-database produced larger reduced datasets than those

produced by the mono-database. However, multi-database produced better results even when the number of instances retained was the same for the mono-database. The results of this study confirm that the reduction of the original dataset will lead to not only an increase in the classification accuracy of the decision tree, but also will produce smaller trees.

Jensen and Cornelis [17] proposed three methods for instance selection based on fuzzy-rough sets. Rough-set theory proved its success in addressing problems related to computationally efficient techniques such as instance selection. Fuzzy-rough sets theory is an improvement on the set theory by enabling more effective modeling of uncertainty. The common base between the three proposed Fuzzy-Rough Instance Selection (FRIS) approaches is that removing or leaving instances in the reduced dataset is determined by using the information in the positive region. The three proposed approaches are FRIS-I, FRIS-II, and FRIS-III. Results reported by the researchers show that the proposed algorithms can reduce the number of instances effectively while maintaining high classification accuracy. FRIS-I, the simplest approach which produced the smallest reduced dataset without affecting the classification accuracy.

Hansen and Olsson [18] proposed a pruning algorithm as a substitute for the error based pruning (part of C4.5 algorithm for decision tree learning). The proposed algorithm takes advantage of the Automatic Design of Algorithms through Evolution (ADATE). ADATE is used for the improvement of machine learning algorithms. Researchers use the ADATE to rewrite the pruning code for C4.5. The ADATE system specification consists of three main parts: The f Function, Available Functions, and ADATE Training Inputs and Evaluation Function. According to the researches, the proposed pruning algorithm has a similar structure as the original pruning algorithm, and the ADATE system did not change the tree traversal. The main advantage of using the ADATE system is that it contains an error estimation function, which is useful for pruning.

A post pruning methodology for decision trees called NSGA-II is implemented by Brunello *et al.* using a multi-objective evolutionary algorithm [19]. A comparison is presented between the ordinary pruning method in C4.5 with the newly presented methodology. Results show that evolutionary algorithms give better results with decision tree classical problems as the presented methodology generates more variegated solution set than C4.5 with a smaller decision tree that has the same and, in some cases, better accuracy.

Xiang and Ma [20] proposed a priority heuristic correlated information method for pruning decision trees. The researchers used a behavior prediction and reasoning approach with the probability correlation analysis framework. The results show that the size of the generated decision tree is reduced, and the predictive accuracy is not affected; it was even improved for real-life decision tree problems.

A boosted adaptive Apriori post pruned decision tree algorithm was developed by Sim *et al.* [21]. The proposed method approximates the re-substitution, cross validation and generalization error rates before and after post pruning. The method was enhanced by adaptive Apriori characteristics, and by applying AdapBoost collaborative technique. The results indicated a stepwise improvement in comparison to ordinary and boosted decision trees.

Ahmed *et al.* proposed a decision tree pruning technique using Bayes minimum risk [22]. The algorithm starts bottom-up by converting the parent node in a subtree to a leaf node, based on the estimated risk of the parent node. Many parameters were considered in building the algorithm, such as accuracy, precision, recall, attribute selection, and required time for the pruning process. Results showed that this new algorithm produces higher accuracy and satisfactory performance for different parameters.

A Parallel Shared Decision Tree (PSDT) algorithm was proposed by She *et al.* [23] in order to solve the problem of building and pruning a decision tree from big data, where regular memory classification algorithms cannot work with this huge size of data. The algorithm is based on Hadoop system for parallel processing. It was shown that the algorithm improved efficiency and accuracy, but still needs more optimization in noisy domains.

4 Methodology

In this section, the datasets used in our experiments are presented in addition to the followed research methodology.

4.1 Datasets

Thirteen benchmarked datasets from the University of California at Irvine (UCI) Machine Learning Repository [24] are used in the conducted experiments and to compare different proposed scenarios. Table 1 shows a description of the datasets used in this study.

Table 1: UCI Benchmarked Datasets used in our experiments

Name	Instances	Input Attributes	Output Attributes	Classes
Australian Credit Approval	690	14	1	2
Breast Cancer Wisconsin (Original)	699	10	1	2
Glass Identification	214	10	1	7
SPECT Heart	267	22	1	2
Image Segmentation	210	19	1	7
Ionosphere	351	34	1	2
Iris	150	4	1	3
Liver Disorders	345	7	1	2
Pima Indians Diabetes	768	8	1	2
Connectionist Bench (Sonar)	208	60	1	2
Statlog (Vehicle Silhouettes)	946	18	1	4
Congressional Voting Records	435	16	1	2
Zoo	101	17	1	7

The chosen datasets cover a wide range of domains with different numbers of features and different numbers of instances. Datasets are split into three categories depending on a certain threshold. The category is determined for each dataset by multiplying its number of instances by the number of input attributes it has. The three dataset categories are: 1) small (<3000), 2) medium (<7000 and >3000) and 3) large (>7000).

4.2 Experiments Description

UCI datasets are usually available in (.data) format. Before applying instance reduction techniques, (.data) files had to be preprocessed into (.NUM) files that are compatible with the C++ code of the instance reduction algorithms.

Noise was also inserted into full datasets before any instance reduction algorithm was applied. Mainly, adding noise is performed by deliberately changing the original class of some instances in a dataset. Inserting noise can be performed manually or automatically, in our experiments, the noise was added automatically. The number of instances affected is determined by noise ratio: (*i.e.*, 5%, 10%, or 20% of instances). Instances affected by this process are randomly chosen.

For each combination of a dataset and an instance reduction algorithm, the experiments were conducted over 10-fold cross-validation then averaged. Almost equal number of instances were produced for each fold. For ELGrow and Explore two folds are used. When pre-processing steps using instance reduction algorithm were used for each fold, they were applied to the training data while the test dataset for this fold has not been

Table 2: Description of the Conducted experiments

#	Experiment Name	Description
1	no reduction, no pruning (NR-NP)	<ol style="list-style-type: none"> 1. No instance reduction algorithm was applied. 2. Noise is inserted with different ratios: 0%, 5%, 10% and 20%. 3. No built-in reduced error pruning was applied. 4. Decision trees were built with 10-folds cross validation and accuracy, tree size and number of attributes are reported.
2	no reduction, pruning (NR-P)	<ol style="list-style-type: none"> 1. No instance reduction algorithm was applied. 2. Noise is inserted with different ratios: 0%, 5%, 10% and 20%. 3. Built-in reduced error pruning was applied. 4. Decision trees were built with 10-folds cross validation and accuracy, tree size and number of attributes are reported.
3	reduction, no pruning (R-N)	<ol style="list-style-type: none"> 1. Noise is inserted with different ratios: 0%, 5%, 10% and 20%. 2. 2-folds cross validation are formed for (ELGrow or Explore), 10 folds are formed for all other instance reduction algorithms. 3. The used instance reduction algorithm is applied to the training set. 4. No built-in reduced error pruning was applied. 5. Decision trees were built and accuracy, tree size and number of attributes are reported.
4	reduction with pruning (R-P)	<ol style="list-style-type: none"> 1. Noise is inserted with different ratios: 0%, 5%, 10% and 20%. 2. 2-folds cross validation are formed for (ELGrow or Explore), 10 folds are formed for all other instance reduction algorithms. 3. The used instance reduction algorithm is applied to the training set. 4. Built-in reduced error pruning was applied. 5. Decision trees were built and accuracy, tree size and number of attributes are reported.

pre-processed to make sure that the results do not get biased. After that we applied a decision tree classification algorithm with and without pruning.

After applying instance reduction techniques, datasets were converted into a comma separated values (.CSV) files to be processed by WEKA data mining tool [25] to build both the pruned and unpruned decision trees. WEKA uses many algorithms for building decision trees. We used J48, which is an open-source Java implementation of C4.5 algorithm for building our trees.

In the experiments where no instance reduction technique was used, WEKA was also responsible for inserting different ratios of noise to the full datasets before building decision trees.

All possible combinations of instance reduction and built-in pruning were utilized in building decision trees with different noise ratios. The number of experiments exceeded 7690 experiments in total. Table 2 shows a description of the experiments conducted in this study.

5 Results

In this section the obtained results are discussed in both cases when no noise was injected and when noise was injected

5.1 Results of using datasets without added noise

This section discusses the results of using full datasets and reduced datasets without deliberately inserting any noisy examples.

5.1.1 Without pruning

Reduced Error Pruning was not applied in this set of experiments. A comparison between the decision tree built using the full dataset and the tree built using RENN noise filter for Zoo dataset is shown in Figure 1(a) and 1(b), respectively. It can be noticed that the decision tree built using the reduced dataset is a compact tree.

Figure 1: Decision tree for Zoo dataset with 0% noise built using (a) Full dataset, (b) RENN Reduced dataset.

Table 3 shows the results in terms of classification accuracy for each benchmarked dataset using seven different instance reduction algorithms and when the full dataset is used. Figure 2 shows the average classification accuracy over the 13 datasets.

Noise filters (ENN, RENN, and ALLKNN) gave significantly better results using the t-test at a 95% significance level in (9, 9, and 8) datasets than the full datasets, respectively. Datasets with better performance than the original full dataset are marked in bold-face in Table 3. This is an expected result since noise filters remove border instances (hard to learn instances or noisy instances). Border instances make the learning process a hard one. From the results, it can be seen that by eliminating those instances, learning accuracy has improved.

The average accuracy of using instance reducers (ELGrow and Explore) is worse than the full datasets' average accuracy. This is expected since instance reducers remove too many instances, some of which may be good representatives of the dataset. The left few instances are not enough to build a tree with good classification accuracy compared to the full dataset.

Table 3: Classification Accuracy for Unpruned Trees With 0% Noise Ratio

Dataset	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ELGrow	Explore	RENN
Australian	82.61%	88.57%	64.07%	59%	94.36%	0%	0%	94.90%
Breast Cancer	93.85%	96.91%	64.30%	59.11%	96.84%	0%	0%	96.91%
Glass Identification	45.79%	39.77%	2%	11%	39.74%	1%	0%	39.74%
Heart	80.52%	90.64%	71.12%	59.61%	88.92%	0%	0%	92.25%
Image	89.05%	81.95%	22.35%	22%	80.24%	8%	5%	42.29%
Ionosphere	91.45%	85.44%	74.11%	60.71%	84.11%	-	0%	85.07%
Iris	96%	97.63%	79%	70.82%	97.36%	0%	0%	97.43%
Liver	57.39%	65.12%	63.65%	63.14%	66.92%	-	-	70.92%
Pima	59.90%	74.93%	54.23%	54.81%	71.30%	-	-	78.08%
Sonar	69.71%	59.06%	53.49%	55.32%	58.53%	0%	0%	58.40%
Vehicle	59.57%	50.88%	59.93%	37.36%	84.57%	-	-	89.96%
Voting	96.32%	99.43%	99%	88.59%	99.12%	0%	0%	99.12%
Zoo	92.08%	97.14%	86%	76.07%	96.87%	0%	0%	97.09%
Average	78.02%	79.04%	61.02%	55%	81.45%	1%	1%	80.17%
++, unknown, -		8, 0, 5	2, 1, 10	1, 0, 12	9, 0, 4	-	-	9, 0, 4

Figure 2: Average Classification Accuracy for Unpruned Trees With 0% Noise Ratio

DROP3 and DROP5 results in terms of average classification accuracy were 61.02% and 55%, respectively. They performed better than instance reducers because they retain more instances. On the other hand, they performed worse than noise filters because DROP3 and DROP5 remove some center instances which affect the classification accuracy.

The next issue to discuss is the size of the produced decision trees in terms of the number of nodes. Table 4 shows the size of the built trees for each dataset. Having (zero) nodes as a tree size means that this algorithm (for example ELGrow) uses one attribute to classify instances with fixed values. Figure 3 shows the average tree size when instance-reduction algorithms are applied and when the full datasets are used.

Table 4: Tree Size (Number of Nodes) for unpruned trees With 0% noise ratio

Dataset	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ELGrow	Explore	RENN
Australian	11	5.1	2.2	2.3	4.1	0	0	3.9
Breast Cancer	15	6.5	0.7	1.1	6.1	0	0	6.1
Glass Identification	2	2	0	0.6	2	0	0	2
Heart	20	6.8	6.3	6.9	10.8	0	0	8.5
Image	14	7.6	0.2	0.5	7.2	0	0	2
Ionosphere	17	1	1	1	1	-	0	1
Iris	4	1	1	1	1.1	0	0	1.1
Liver	20	5.4	2.1	1.8	7.7	-	-	6.3
Pima	2	4.5	1	1	1.2	-	-	5.8
Sonar	17	1	1	1	1	0	0	1
Vehicle	13	2.2	1.3	1.9	3	-	-	2.9
Voting	18	2.3	1	2.4	2.1	0	0	2.1
Zoo	8	5.6	5.4	5.4	5.6	1.6	2	5.5
Average	12.38	3.92	1.78	2.07	4.07	0.18	0.2	3.71
++, unknown, -		10, 2, 1	7, 6, 0	9, 4, 0	10, 3, 0	-	-	9, 3, 1

Figure 3: Average Tree Size (Number of Nodes) for Unpruned Trees With 0% Noise Ratio

On average, instance reducers produced the smallest trees, ELGrow (0.17) and Explore (0.2). DROP3 and DROP5 also produced significantly small trees 1.78, and 2.07, respectively. These four techniques are expected to give such results because they reduce the number of instances significantly; however, this affects the classification accuracy.

Noise filtering algorithms (ENN, RENN, and ALLKNN) also produced small-sized decision trees compared to the full datasets; (4.07, 3.71, and 3.92), respectively. This indicates that removing border instances improves classification accuracy and produces smaller trees, which means easier and faster learning process.

Table 5 shows the number of attributes used to build each decision tree for the full and reduced datasets. Figure 4 shows the average number of the used attributes. On average ELGrow and Explore produced trees with

Table 5: Number of Attributes for unpruned trees with 0% noise ratio

Dataset	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ELGrow	Explore	RENN
Australian	8	5.1	2.2	2.3	4.1	0	0	3.9
Breast Cancer	6	4.5	0.7	1.1	5.4	0	0	5
Glass Identification	2	2	0	0.6	2	0	0	2
Heart	14	6.4	6.2	6.4	9.3	0	0	7.5
Image	9	6.3	0.2	0.5	5.4	0	0	1.9
Ionosphere	10	1	1	1	1		0	1
Iris	2	1	1	1	1.1	0	0	1.1
Liver	5	3.6	2	1.8	3.9	-	-	3.5
Pima	2	2.5	1	1	1.2	-	-	2.9
Sonar	13	1	1	1	1	0	0	1
Vehicle	8	1.8	1.3	1.7	2.1	-	-	2.1
Voting	13	2.3	1	2.4	2.1	0	0	2.1
Zoo	8	5.5	5	5.4	5	1.6	2	5.1
Average	7.69	3.31	1.74	2.02	3.35	0.18	0.2	3.01

Figure 4: Average Number of Attributes for Unpruned Trees With 0% Noise Ratio

0.18 and 0.2 number of attributes, respectively which is much less than the full datasets that have on average of 7.69 attributes. It is an expected result; because instance reducers produce a very small set of instances, therefore, it is logical to choose a very small number of attributes to build the decision trees. DROP3 and DROP5 remove border instances and some center instances and thus use fewer attributes than the full datasets.

On the other hand, noise filters (ENN, RENN, and ALLKNN) removed border instances and used a reduced number of attributes compared to the full datasets. They use (3.35, 3.01, and 3.31) attributes, respectively. This corresponds to 43.56%, 39.14%, and 43.04%, respectively of the number of attributes used by the full datasets. Removing border instances increases the classification accuracy and makes the learning easier as selecting which attributes to use is more effective.

5.1.2 With pruning

This section introduces the results of applying built-in Reduced Error Pruning on datasets with the full and the reduced datasets. Noise filters outperformed the full datasets. ENN, RENN, and ALLKNN achieved an average accuracy of 79.79%, 77.87%, and 77.86%, while the average accuracy of the full datasets was 76.66%. Table 6 and Figure 5 show no significant difference between these results and the results achieved with no pruning, as shown in Table 3 and Figure 2.

This leads us to conclude that noise filters have more impact on the classification accuracy of decision trees than using built-in pruning. Again, better results for noise filters on datasets are bold-faced. RENN also significantly outperformed the full datasets in 9 out of 13 datasets.

Table 6: Classification Accuracy for pruned Trees With 0% Noise Ratio

Dataset	Instance Reduction Algorithms						
	None	ALLKNN	DROP3	DROP5	ENN	ELGrow	Explore
Australian	85.22%	87.59%	60%	56%	93.46%	instances<folds	instances<folds
Breast Cancer	92.99%	96.53%	65.06%	60.70%	96.45%	instances<folds	instances<folds
Glass Identification	40.65%	35.80%	11.16%	6.10%	36.08%	0%	0%
Heart	80.90%	91.16%	64.22%	58.63%	87.58%	instances<folds	instances<folds
Image	83.33%	78.13%	21.02%	21%	77.31%	8.29%	6%
Ionosphere	87.46%	85.44%	74.58%	62.42%	84.11%	-	instances<folds
Iris	94.67%	95.59%	70%	48.26%	95.17%	instances<folds	instances<folds
Liver	56.23%	64.99%	62.65%	60.75%	66.57%	-	-
Pima	65.10%	76.77%	54.59%	54.50%	74.14%	-	-
Sonar	69.23%	58.88%	53.68%	52.26%	59.75%	instances<folds	instances<folds
Vehicle	52.13%	48.58%	55.90%	39.54%	73.50%	-	-
Voting	95.63%	99.68%	99%	83.68%	99.49%	instances<folds	instances<folds
Zoo	93.07%	92.99%	31.23%	27.99%	93.62%	0%	0%
Average	76.66%	77.86%	56%	49%	79.79%	2.76%	2.00%
++, unknwn, -		7, 3, 3	3, 0, 10	1, 0, 12	7, 3, 3	-	-

Figure 5: Average Classification Accuracy for Pruned Trees With 0% Noise Ratio

DROP3 and DROP5 gave worse results than full datasets; (56% and 49%), respectively. As mentioned earlier, these two instance reduction algorithms remove border instances and some center instances, which means they remove some representative instances that impact classification accuracy.

Instance reducers produce the worst results. ELGrow and Explore have an average classification accuracy of 2.76% and 2%, respectively as they remove huge number of instances, leaving few instances that are almost always not enough to build a decision tree with high classification accuracy. Using reduced error pruning improved the classification accuracy when using either; ELGrow or Explore as reported in Table 3.

Table 7 and Figure 6 report the tree size. ELGrow, Explore, DROP3 and DROP5 produce the smallest trees with an average of (0, 0, 0.97, 0.97), respectively. These algorithms extremely reduce the number of instances used to build the decision trees. As the number of instances greatly reduced, learning ability drops, which is shown by the very small tree size.

Table 7: Tree Size (Number of Nodes) for pruned trees With 0% noise ratio

Dataset	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ELGrow	Explore	RENN
Australian	4	2.9	0.8	2.1	2.2	instances<folds	instances<folds	2.8
Breast Cancer	8	2.9	0.2	0.4	3.2	instances<folds	instances<folds	3.5
Glass Identification	2	0.7	0	0	0.7	0	0	0.7
Heart	6	2.3	2.5	1.8	4.5	instances<folds	instances<folds	4.9
Image	8	5.5	0.1	0.1	5.5	0	0	1
Ionosphere	4	1	0	0	1	-	instances<folds	1
Iris	4	1	1	0.7	1	instances<folds	instances<folds	1
Liver	0	0.6	0.7	0.5	1.1	-	-	0.8
Pima	0	0.4	0.4	0.3	0.1	-	-	0.7
Sonar	3	0.5	0.2	0.4	0.3	instances<folds	instances<folds	0
Vehicle	4	1.1	0.8	0.4	1.9	-	-	1.8
Voting	4	1	1	1.2	1	instances<folds	instances<folds	1
Zoo	6	5.2	4.9	4.7	5.5	0	0	5.5
Average	4.08	1.93	0.97	0.97	2.15	0	0	1.9
++, unknown, -		11, 0, 2	7, 4, 2	9, 2, 2	9, 3, 1	-	-	7, 4, 2

Figure 6: Average Tree Size (Number of Nodes) for Pruned Trees With 0% Noise Ratio

Noise filters gave trees with average size almost equal half of full datasets average tree size. ENN average tree size was (2.15), RENN average tree size was (1.9), and ALLKNN average tree size was (1.93) while full datasets average tree size was (4.08). Border instances are removed using those noise filters, which usually results in eliminating a small number of instances, and thus, the size of trees is not greatly reduced. ALLKNN significantly outperformed the full datasets in 11 out of 13 datasets.

Reduced Error Pruning significantly affected tree size in the cases of: 1) using noise filters and 2) using full datasets. Since Reduced Error Pruning aims to reduce the size of a decision tree without negatively affecting its classification ability, this result was expected.

Table 8 shows the number of attributes for the full and the reduced datasets. Figure 7 shows the average number of attributes over the datasets. The results regarding the number of attributes are very close to results in terms of the tree size. Instance reducers, along with DROP3 and DROP5 used the smallest number of attributes because of the small instance sets they produce. The number of attributes was not significantly affected by

Table 8: Average Number of Attributes for pruned trees With 0% noise ratio

Dataset	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ELGrow	Explore	RENN
Australian	4	2.9	0.8	2.1	2.2	instances<folds	instances<folds	2.8
Breast Cancer	5	2.6	0.2	0.4	3.1	instances<folds	instances<folds	3.2
Glass Identification	2	0.7	0	0	0.7	0	0	0.7
Heart	6	2.2	2.5	1.7	4.3	instances<folds	instances<folds	4.5
Image	6	4.6	0.1	0.1	4.4	0	0	1
Ionosphere	4	1	0	0	1	-	instances<folds	1
Iris	3	1	1	0.7	1	instances<folds	instances<folds	1
Liver	-	0.6	0.7	0.5	1.1	-	-	0.8
Pima	-	0.4	0.4	0.3	0.1	-	-	0.7
Sonar	3	0.5	0.2	0.4	0.3	instances<folds	instances<folds	0
Vehicle	4	1.1	0.8	0.4	1.9	-	-	1.8
Voting	4	1	1	1.2	1	instances<folds	instances<folds	1
Zoo	6	5	4.6	4.7	5.1	0	0	5.3
Average	4.27	1.82	0.95	0.96	2.02	0	0	1.83

Figure 7: Average Number of Attributes for Pruned Trees With 0% Noise Ratio

Reduced Error Pruning. On the other hand, noise filters and full datasets used more attributes, but still, the number of attributes was significantly affected by applying Reduced Error Pruning when compared with the case when only noise filters were used.

5.2 Results of using noisy datasets

This section discusses the results of using the full and the reduced datasets when noise was deliberately inserted by changing the class of randomly chosen instances. Different noise ratios, 5%, 10%, and 20% were inserted to each dataset. Each dataset was reduced using the seven different instance reduction algorithms, and the resulting datasets, along with the full noisy datasets, were used to build the decision trees both without and with pruning.

5.2.1 Without pruning

Table 9 shows the average classification accuracy of unpruned decision trees over the 13 used benchmarked datasets with noisy datasets. In general, the classification accuracy degrades with the increase in noise ratio. Noise filters removed noisy instances and consequently improved the classification accuracy. ENN outperformed other noise filtering techniques in 0%, 5%, and 10% noise ratios. On 20% noise ratio, RENN and ENN gave almost the same result. ALLKNN gave classification accuracy that is very close to ENN with 0%, 5%, and 10% noise ratios. However, ENN gave higher average classification accuracy with 20% noise ratio.

Table 9: Average classification accuracy for unpruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ELGrow	Explore	RENN
0%	78.02%	79.04%	61.02%	55%	81.45%	1%	1%	80.17%
5%	73.06%	80.56%	61.12%	54.98%	80.83%	1%	1%	79.23%
10%	68.61%	79.14%	60.80%	53.63%	80.18%	1%	1%	79.52%
20%	61.10%	72.01%	57%	50.56%	76.56%	1%	3%	76.99%

It can be noticed that noise filters effectiveness becomes more obvious in higher noise ratios, because the results indicate that the difference in the classification accuracy between the filtered datasets and the full datasets increases as the noise ratio increases. DROP3, DROP5, ELGrow, and Explore performed much worse than the full datasets in terms of classification accuracy.

Table 10 shows the number of datasets where the accuracy of the reduced dataset was significantly better (++) than the original (full) dataset at 95% confidence level using t-test, significantly worse (–) or the difference was not statistically significant (unknown). RENN noise filter achieved the maximum advantage over the full

Table 10: Total Number of datasets with affected average accuracy (++, unknown, –) for unpruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms				
	ALLKNN	DROP3	DROP5	ENN	RENN
0%	8, 0, 5	2, 1, 10	1, 0, 12	9, 0, 4	9, 0, 4
5%	9, 1, 3	1, 3, 9	0, 1, 12	9, 1, 3	9, 0, 4
10%	10, 1, 2	2, 5, 6	0, 2, 11	10, 1, 2	10, 0, 3
20%	8, 2, 3	4, 2, 7	0, 4, 9	9, 3, 1	11, 0, 2

dataset. Its performance was significantly better than the full dataset in 11 out of the 13 used datasets at 20% noise ratio.

Table 11 compares decision tree for 0%, 5%, 10% and 20% noise ratios in terms of the tree size which tends to increase as noise ratio increases. Although DROP3, DROP5 and instance reducers produce the smallest decision trees sizes, but this reduction influences the classification accuracy of those algorithms. Noise filters produced trees with balanced size and classification accuracy, as they remove noisy- untypical border instances. ALLKNN produces the smallest trees among noise filters followed by RENN then ENN.

Table 11: Average Tree Size for unpruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ElGrow	Explore	RENN
0%	12.38	3.92	1.78	2.07	4.07	0.18	0.2	3.71
5%	12.62	4.56	1.82	2	4.63	0.14	0.18	3.87
10%	16.15	4.18	1.87	2.5	5.24	0.14	0.18	4.69
20%	18.15	3.93	2.17	2.94	4.91	0.14	0.16	4.09

Table 12 shows the number of datasets where the tree size of the reduced dataset was smaller, *i.e.* statistically better (++) than the original dataset, statistically worse (–) or the difference was not statistically significant (unknown). ALLKNN noise filter achieved the maximum advantage over the full dataset. Its performance was significantly better than the full dataset in 10 out of the 13 used datasets at all noise ratios.

Table 12: Total Number of datasets with affected average Tree Size (++, unknown, –) for unpruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms				
	ALLKNN	DROP3	DROP5	ENN	RENN
0%	10, 2, 1	7, 6, 0	9, 4, 0	10, 3, 0	9, 3, 1
5%	10, 1, 2	7, 6, 0	10, 3, 0	8, 4, 1	8, 3, 2
10%	10, 2, 1	7, 6, 0	10, 3, 0	8, 4, 1	8, 3, 2
20%	10, 1, 2	6, 7, 0	9, 3, 1	8, 4, 1	8, 3, 2

Table 13 shows the average number of attributes among various instance reduction techniques at different noise ratios. ElGrow and Explore gave the smallest number of attributes compared to the full datasets. DROP3 and DROP5 remove too many instances, this affects their classification accuracy. On the other hand, noise filters balance the number of attributes and size with classification accuracy. Also, when using full datasets, decision trees tend to have many repeatedly used attributes. This is clear from the difference between second column in Table 11 and second column in Table 13.

Table 13: Average Number of Attributes for unpruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ElGrow	Explore	RENN
0%	7.69	3.31	1.74	2.02	3.35	0.18	0.2	3.01
5%	8.54	3.51	1.78	1.94	3.72	0.14	0.18	3.11
10%	8.54	3.39	1.85	2.32	3.67	0.14	0.18	3.35
20%	8.69	3.25	2.05	2.61	3.75	0.14	0.16	3.16

Figure 8 shows the difference in classification accuracy between the reduced datasets and the full dataset with different noise ratios. It can be noticed that noise filters outperformed the full dataset, the difference in performance between noise filtered reduced datasets and the full dataset is more apparent with higher noise ratios. Positive values indicate the reduced dataset achieved better performance than the full dataset while negative values indicate better performance for the full dataset.

Figure 8: The Difference in accuracy between the full datasets and the reduced datasets without pruning

5.2.2 With pruning

Table 14 shows the average classification accuracies of different instance reduction algorithms on noisy datasets when pruning is applied to the decision trees.

In general, there is no significant effect for noise on the classification accuracy with 0%, 5%, and 10% noise ratios. However, accuracies drop on 20% noisy datasets. In all noise ratios, noise filters outperformed the full datasets. The gap between noise filters average classification accuracy and its correspondence of full datasets increases as the noise ratio increases, this means that noise filters are more resistant to noise.

DROP3 and DROP5 gave worse classification accuracies than the full datasets. This is expected since those instance reduction algorithms remove number of useful center instances. ELGrow and Explore gave the worst accuracies; because they reduce instances to the minimum by eliminating many center instances.

Considering the pruning effect when Table 14 is compared to Table 9. Results are slightly better when Reduced Error Pruning is not used for building the trees.

Table 15 shows the number of datasets where the accuracy of the reduced dataset was statistically better (++) or statistically worse (–) than the original dataset or the difference was not statistically significant (unknown). RENN noise filter achieved the maximum advantage over the full dataset. Its performance was statistically better than the full dataset in 10 out of the 13 used datasets in 10% and 20% noise ratios.

Table 14: Average classification accuracy for pruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ElGrow	Explore	RENN
0%	76.66%	77.86%	56%	49%	79.79%	2.76%	2.00%	77.87%
5%	73.35%	78.35%	55.59%	49.89%	79.49%	2.88%	1.14%	77.89%
10%	68.82%	77.63%	56.07%	48.30%	79.22%	3.58%	2.18%	77.48%
20%	63.04%	71.96%	52%	46.75%	75.21%	0.56%	2.07%	75.28%

Table 15: Total Number of datasets with affected average accuracy (++, unknown, –) for pruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms				
	ALLKNN	DROP3	DROP5	ENN	RENN
0%	7, 3, 3	3, 0, 10	1, 0, 12	7, 3, 3	9, 0, 4
5%	8, 2, 4	1, 2, 10	1, 0, 12	9, 1, 3	9, 1, 3
10%	10, 2, 1	3, 0, 10	0, 2, 11	10, 2, 1	10, 2, 1
20%	9, 1, 3	3, 3, 7	0, 2, 11	10, 1, 2	10, 1, 2

Table 16: Average Tree Size for pruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ElGrow	Explore	RENN
0%	4.08	1.93	0.97	0.97	2.15	0	0	1.9
5%	4.08	2.03	0.92	0.79	2.08	0	0	1.59
10%	4.23	1.67	0.92	0.97	2.18	0	0	1.85
20%	3.08	1.46	1.07	0.88	2.12	0	0	1.79

Table 17: Total Number of datasets with affected average Tree Size (++, unknown, –) for pruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms				
	ALLKNN	DROP3	DROP5	ENN	RENN
0%	11, 0, 2	7, 4, 2	9, 2, 2	9, 3, 1	7, 4, 2
5%	10, 0, 3	7, 5, 1	7, 5, 1	7, 3, 3	7, 3, 3
10%	8, 3, 2	6, 5, 2	9, 2, 2	7, 3, 3	4, 4, 5
20%	8, 1, 4	4, 5, 4	4, 5, 4	5, 3, 5	5, 3, 5

Table 18: Average Number of Attributes for pruned trees at different noise ratios

Noise Ratio	Instance Reduction Algorithms							
	None	ALLKNN	DROP3	DROP5	ENN	ElGrow	Explore	RENN
0%	4.27	1.82	0.95	0.96	2.02	0	0	1.83
5%	3.54	1.84	0.92	0.78	1.98	0	0	1.59
10%	5	1.56	0.92	0.94	1.99	0	0	1.75
20%	4.11	1.4	1.02	0.88	1.98	0	0	1.75

Reduced Error Pruning has a more obvious effect on the average tree size, as shown in Table 16 and Table 17 when compared to the unpruned case as shown in Table 11 and Table 12, respectively. It is clear that the use of built-in pruning reduced the tree size and the number of attributes.

In terms of tree size and number of attributes as shown in Table 18, ELGrow, Explore, DROP3 and DROP5, as expected gave the smallest trees and smallest number of attributes at the expense of their classification accuracy. Noise filters gave larger trees with larger number of attributes than other techniques, but almost half of the full datasets. This is evidence that noise filters not only give higher classification accuracy, but they also can produce smaller trees than the full dataset.

Figure 9 shows the difference in classification accuracy between the reduced datasets and the full dataset with different noise ratios when pruning is applied. Noise filters outperformed the full dataset; the difference in performance between noise filtered datasets and the full dataset is more apparent with higher noise ratios. However, the gap between noise filters and the full dataset is smaller than the case when pruning was not applied as was shown in Figure 8.

Figure 9: The Difference in accuracy between the full datasets and the reduced datasets with pruning

5.3 Results considering dataset size

The obtained results are discussed in this section according to the size of datasets. Datasets are split into three categories depending on a certain threshold. The category is determined for each dataset by multiplying its number of instances by the number of input attributes it has. The three dataset categories are: 1) small (<3000), 2) medium (<7000 and >3000) and 3) large (>7000).

Four datasets are in the first category: (Glass, Iris, Liver, and Zoo), the second category contains 5 datasets: (Breast Cancer, Image, Voting, Pima, and Heart) and the last category has the remaining 4 datasets as its members: (Australian, Ionosphere, Sonar, and Vehicle).

Noise filters give their best performance when used on the medium datasets. The classification accuracy obtained using noise filters is the lowest when using the small datasets. The previous findings are true when applying the built-in reduced error pruning and without applying it.

Without using reduced error pruning, tree size tends to be large when using medium datasets while the smallest trees are produced with large datasets. When using built-in reduced error pruning, the large datasets produce decision trees with the smallest size.

Results in terms of the number of attributes used to build decision trees are similar to the tree size results. Large datasets gave the smallest number of attributes with and without using built-in pruning.

6 Conclusions and future work

Classification aims to categorize instances into their different classes, decision boundary can be drawn halfway between two nearest instances of different classes. Therefore, border instances are seen as noisy instances or instances that do not agree with their neighbors. In this study, the use of many instance reduction algorithms as pre-pruning techniques to smooth decision trees' decision boundaries is investigated. Although this is a general approach that can be applied to any machine learning algorithm, conducted experiments reported in this work are limited to decision trees.

Conducted experiments prove that eliminating border instances increases the classification accuracy of decision trees. Removing border instances improves classification accuracy and produces smaller decision trees, which means faster learning and classification. Noise filters produced trees with balanced tree size and classification accuracy as well as number of attributes. Although instance reducers, DROP3 and DROP5 gave the smallest trees, but they have low classification accuracy.

Reduced Error Pruning affected the size and the number of attributes significantly; unfortunately, this is not true considering the classification accuracy. Noise filters reduced the tree size and number of attributes and improved classification accuracy. Using built-in Reduced Error Pruning with noise filter optimized the learning process but did not improve the classification accuracy.

In general, using noise filters reduction algorithms without pruning outperformed other techniques in terms of classification accuracy. On the other hand, pruning with and without prior use of reduction algorithms produce the smallest trees with the smallest number of attributes.

According to the dataset size, results show that medium datasets outperformed large and small datasets in terms of classification accuracy. However, large datasets outperformed medium and small datasets in terms of the tree size and the number of attributes.

All of the above conclusions are valid without and with deliberately inserted noise, although the advantage of using noise filters was more apparent with the increase in noise ratio as they show more resistance to noise.

Other instance reduction algorithms may be tested in the future; also other machine learning algorithms could be tested. This noise tested in this research is classification noise; future research can evaluate the effect of inserting attribute noise and the impact of missing values. Our experiments rely heavily on the use of the Heterogeneous Value Distance Metric (HVDm) distance function, and other distance functions can be used, such as the ISCDM [11, 26].

Other datasets [27, 28] can be used in future studies with either decision trees or other machine learning techniques.

Feature selection based on correlation with the class but not with other features was proposed by Hall [29]. This technique's effectiveness can be tested with the proposed use of instance reduction techniques as pre-pruning, with or without intentionally added noise.

References

- [1] K. El Hindi, & Al-Akhras, M., "Smoothing Decision Boundaries to Avoid Overfitting in Neural Network Training.," *Neural Network World*, vol. 21, pp. 311-325, 2011.
- [2] T. M. Mitchell, *Machine Learning*. New York, NY, USA: McGraw-Hill, 1997.
- [3] C. Li, "Wearable Computing: Accelerometer-Based Human Activity Classification Using Decision Tree," Utah State University, 2017.
- [4] E. I. Metting, J. C. C. M. in 't Veen, P. N. R. Dekhuijzen, E. van Heijst, J. W. H. Kocks, J. B. Mulwijk-Kroes, *et al.*, "Development of a diagnostic decision tree for obstructive pulmonary diseases based on real-life data," *ERJ Open Research*, vol. 2, pp. 00077-2015, 01/2016 2016.

- [5] T. Boros, S. D. Dumitrescu, and S. Pipa, "Fast and Accurate Decision Trees for Natural Language Processing Tasks," in *RANLP 2017 – Recent Advances in Natural Language Processing Meet Deep Learning*, 2017, pp. 103-110.
- [6] D. R. Wilson, & Martinez, T. R., "Reduction Techniques for Exemplar-Based Learning Algorithms.," *Machine Learning*, vol. 38, pp. 257-268, 2000.
- [7] J. R. Quinlan, "Induction of decision trees," *Machine Learning*, vol. 1, pp. 81-106, 1986/03/01 1986.
- [8] J. R. Quinlan, *C4.5: programs for machine learning*: Morgan Kaufmann Publishers Inc., 1993.
- [9] E. Alpaydin, "Introduction to Machine Learning.," *the MIT press.*, 2004.
- [10] R. Kohavi and D. Sommerfield, "Feature subset selection using the wrapper method: overfitting and dynamic search space topology," presented at the Proceedings of the First International Conference on Knowledge Discovery and Data Mining, Montréal, Québec, Canada, 1995.
- [11] K. El Hindi, "Specific-class distance measures for nominal attributes," *AI Communications*, vol. 26, pp. 261–279, 2013.
- [12] C. A. Brunk and M. J. Pazzani, "An Investigation of Noise-Tolerant Relational Concept Learning Algorithms," in *Machine Learning Proceedings 1991*, L. A. Birnbaum and G. C. Collins, Eds., ed San Francisco (CA): Morgan Kaufmann, 1991, pp. 389-393.
- [13] K. El Hindi and M. AL-Akhras, "Eliminating Border Instance to Avoid Overfitting," presented at the IADIS International Conference Intelligent Systems and Agents 2009 (ISA 2009), 2009.
- [14] I. Czarnowski, "Cluster-based instance selection for machine classification," *Knowledge and Information Systems*, vol. 30, pp. 113-133, 2012/01/01 2012.
- [15] P. Domingos, "Unifying instance-based and rule-based induction," *Machine Learning*, vol. 24, pp. 141-168, 1996/08/01 1996.
- [16] I. Czarnowski and P. Jędrzejowicz, "Instance reduction approach to machine learning and multi-database mining," in *Annales UMCS Informatica AI*, 2006, pp. 60-71.
- [17] R. Jensen and C. Cornelis, "Fuzzy-rough instance selection," in *International Conference on Fuzzy Systems*, 2010, pp. 1-7.
- [18] S. Hansen and R. Olsson, "Improving Decision Tree Pruning through automatic Programming," presented at the In Proceedings of the Norwegian Informatics Conference., 2007.
- [19] A. Brunello, E. Marzano, A. Montanari, and G. Sciacicco, "Decision Tree Pruning via Multi-Objective Evolutionary Computation," *International Journal of Machine Learning and Computing*, vol. 7, pp. 167-175, 12/2017 2017.
- [20] Y. Xiang and L. Ma, "A Priority Heuristic Correlation Technique for Decision Tree Pruning," in *Advanced Multimedia and Ubiquitous Engineering*. vol. 590, J. J. Park, L. T. Yang, Y.-S. Jeong, and F. Hao, Eds., ed Singapore: Springer Singapore, 2020, pp. 176-182.
- [21] D. Y. Y. Sim, C. S. Teh, and A. I. Ismail, "Improved Boosted Decision Tree Algorithms by Adaptive Apriori and Post-Pruning for Predicting Obstructive Sleep Apnea," *Advanced Science Letters*, vol. 24, pp. 1680-1684, 2018-03-01 2018.
- [22] A. M. Ahmed, A. Rizaner, and A. H. Ulusoy, "A novel decision tree classification based on post-pruning with Bayes minimum risk," *PLOS ONE*, vol. 13, p. e0194168, 2018-4-4 2018.
- [23] X. She, T. Lv, and X. Liu, "The Pruning Algorithm of Parallel Shared Decision Tree Based on Hadoop," in *2017 10th International Symposium on Computational Intelligence and Design (ISCID)*, 2017, pp. 480-483.
- [24] C. Blake and C. Merz. UCI Repository of Machine Learning Databases [Online]. Available: <https://archive.ics.uci.edu/ml/index.php>
- [25] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The WEKA data mining software," *ACM SIGKDD Explorations Newsletter*, vol. 11, p. 10, 2009.
- [26] M. Jamjoom and K. El Hindi, "Partial instance reduction for noise elimination," *Pattern Recogn. Lett.*, vol. 74, pp. 30–37, 2016.
- [27] L. Jiang, L. Zhang, L. Yu, and D. Wang, "Class-specific attribute weighted naive Bayes," *Pattern Recognition*, vol. 88, pp. 321-330, 2019/04/01/ 2019.
- [28] H. Zhang, L. Jiang, and L. Yu, "Class-specific attribute value weighting for Naive Bayes," *Information Sciences*, vol. 508, pp. 260-274, 2020/01/01/ 2020.
- [29] M. Hall, "Correlation-Based Feature Selection for Machine Learning," *Department of Computer Science*, vol. 19, 06/17 2000.