Improving Classification Accuracy by Identifying and Removing Instances that Should Be Misclassified

Michael R. Smith and Tony Martinez

Abstract— **Appropriately handling noise and outliers is an important issue in data mining. In this paper we examine how noise and outliers are handled by learning algorithms. We introduce a filtering method called PRISM that identifies and removes instances that *should* be misclassified. We refer to the set of removed instances as *ISMs (instances that should be misclassified)*. We examine PRISM and compare it against 3 existing outlier detection methods and 1 noise reduction technique on 48 data sets using 9 learning algorithms. Using PRISM, the classification accuracy increases from 78.5% to 79.8% on a set of 53 data sets and is statistically significant. In addition, the accuracy on the non-outlier instances increases from 82.8% to 84.7%. PRISM achieves a higher classification accuracy than the outlier detection methods and compares favorably with the noise reduction method.**

I. Introduction

It is common that noise and outliers exist in real world data sets due to errors such as typographical errors or measurement errors. When the data is modeled using machine learning algorithms, the presence of noise and outliers can affect the model that is generated. Improving how learning algorithms handle noise and outliers can produce better models.

Handling noise and outliers has been addressed in a number of different ways, beginning with preventing overfit. A common approach to prevent overfit is adhering to Occam's razor which states that the simplest hypothesis that fits the data tends to be the best one. Using Occam's razor, a trade-off is made between accuracy on the training set and the complexity of the model, preferring a simpler model that will not overfit the training set. Another technique to prevent overfit is to use a validation set during training to ensure that noise and outliers are not learned. Some learning algorithms have a built in method to remove suspected outliers, such as C4.5 which prunes leaves that are thought to be insignificant [16].

Other approaches explicitly address noise and outliers by trying to identify them. One difficulty, however, is that there is no agreed upon definition of what constitutes an outlier or how to distinguish noise from an exception. Outlier detection aims a finding anomalies in the data and has been done using a variety of approaches such as statistical methods [11], rule creation [10], and clustering techniques [3]. Noise reduction methods attempt to identify and remove mislabeled instances such as repeated edited nearest neighbor that removes any instance that is misclassified by a 3-nearest neighbor classifier [24] or removing instances misclassified by a voting ensemble [4].

Michael R. Smith and Tony Martinez are with the Department of Computer Science, Brigham Young University, Provo, Utah, USA (email: msmith@axon.cs.byu.edu, martinez@cs.byu.edu).

In this paper, rather than attempting to identify anomalies or mislabeled instances, we identify instances that *should* be misclassified and examine how they affect classification accuracy. We introduce PRISM (*PReprocessing Instances that Should be Misclassified*), a novel filtering method that removes instances that *should* be misclassified using heuristics that predict how likely it is that an instance will be misclassified [21]. By "instances that should be misclassified," we mean that in the absence of additional information other than what the data set provides, the label assigned by the learning algorithm to the instance is the most appropriate one, even if it happens to be different from the instance's target value.

We call the instances that are removed *Instances that Should be Misclassified* or ISMs to distinguish them from traditional outliers and class noise. ISMs exhibit a high degree of class overlap where class overlap refers to how similar an instance is to other instances of different classes in an area of the task space. This idea is portrayed in Figure 1 that shows a hypothetical 2-dimensional dataset with three outliers (instances with striped fill). In this paper, outlier instances 2 and 3 are removed by PRISM since they should be misclassified. Traditional outlier approaches would deem instances 1 and 3 as outliers but would not consider instance 2 as an outlier since class is not taken into account. Noise reduction techniques may not remove instance 3 since it is sufficiently different from the square instances.

We also classify each instance as a ISM, border point, or other based on a set of heuristics that predict how likely an instance is to be misclassified [21]. The presence of noise and outliers affects the classification border, effectively pulling the classification border as the the learning algorithm optimizes the squared error, or some other objective function. Removing the noise and outliers during training allows the learning algorithm to learn a more accurate classification boundary. This idea is also portrayed in Figure 1. The instances with a solid fill represent border points. The solid line represents the true classification border and the dashed line represents the classification from a learning algorithm that is affected by the outlier.

Removing the ISMs prior to training increases the overall average accuracy from 78.5% to 79.8%. In addition, the accuracy for instances that are not ISMs increases from 82.8% to 84.7% while the accuracy on ISMs decreases. In this manner, the learning algorithm models the data more precisely. Removing the ISMs during training improves the classification accuracy on the border points from 69.0% to

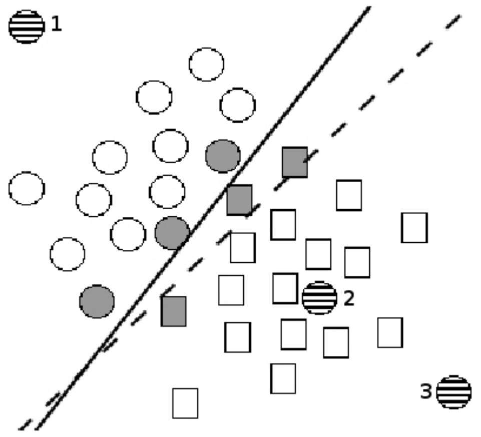

Fig. 1. A hypothetical 2-dimensional data set with that illustrates how outliers (circles with striped fill) affect the model generated by a learning algorithm. The solid line represents the true classification border and the dashed line represents the classification from a learning algorithm that is affected by the outlier. The filled in instances represent border points.

71.6% on all of the data sets and from 63.6% to 69.0% on data sets that had more than 10% of the instances identified as ISMs.

We also broadly investigate how outliers (as defined using different outlier detection methods) and noise affect the classification accuracy. In addition to PRISM, we use three outlier detection methods and 1 noise reduction algorithm; a distance-based approach [17], local outlier factor (LOF) [3], the enhanced class outlier distance based algorithm (ECODB) [18], and repeated edited nearest neighbor (RENN) [24]. The distance approach and LOF identify the traditional outliers and slightly decrease the classification accuracy. The other methods take the class label into account and improve classification accuracy.

Section II describes the experimental methodology. Section III describes how PRISM detects ISMs and, as an extension, how we identify instances as border points. The results are then presented in Section IV. We review related works in Section V and conclude in Section VI.

II. Experimental Methodology

We examine how filtering affects the classification accuracy of 53 data sets and 9 learning algorithms trained with and without filtering. The learning algorithms were chosen with the intent of being representative of a diverse set of learning algorithms commonly used in practice. The algorithms that were used are shown in Table I. No parameter optimization was performed on any of the algorithms. They were used as implemented in Weka with their default parameters since we are interested in the effect that filtering has rather than parameter tuning [25]. The set of 53 data sets was selected from the UCI Machine Learning Repository [2]. This set was built to include data sets that vary significantly along important dimensions such as the number of attributes, the types of the attributes, the number of instances and the application domain. All of the data sets have a percentage of instances that are identified as outliers ranging from 0.1% to 52%. Each data set and algorithm is evaluated using 10-fold cross-validation on the filtered data set. The instances that are filtered (outliers/noise) are evaluated on a model trained using all of the non-filtered instances.

TABLE I
List of learning algorithms.

Learning Algorithms
Decision Tree (C4.5 [16]) {C4.5}
Naïve Bayes {NB}
Multi-layer Perceptron trained with Back Propagation {MLP}
Perceptron {Percep}
Support Vector Machine {SVM}
1-NN (1-nearest neighbor) {IB1}
5-NN (5-nearest neighbors) {IB5}
Repeated Incremental Pruning to Produce Error Reduction) {RIPPER}
RBF Network {RBF}

To examine the effect of filtering on classification accuracy, we first determine which instances to filter and remove them from the training set. We use the following outlier detection and noise reduction methods to filter instances:

- A distance-based approach as implemented by Ramaswamy et al [17] that ranks each instance based on its distance to its k nearest neighbors. The instances with the top n rankings are identified as outliers. This method partitions the data set to accommodate large data sets.
- LOF (Local Outlier Factor) [3] is an approach loosely related to density-based clustering that assigns each instance a value representing its potential of being an outlier with respect to the instances in its neighborhood. In this work, we identified the instances with the top n LOF values as outliers.
- ECODB (Enhanced Class Outlier Distance Based) [18] is an outlier detection approach that takes the class label into account. ECODB chooses as outliers the top n instances that have the smallest distance to their k-nearest neighbors, the greatest deviation, and a different class label from its k-nearest neighbors.
- RENN (Repeated Edited Nearest Neighbor) [24] repeatedly removes instances that are misclassified by a 3-NN classified until no instances are misclassified.
- PRISM (P Reprocessing Instances that Should be Misclassified) removes instances that should be misclassified by a learning algorithm. PRISM is described in Section III.

We used RapidMiner [13] to implement the first three outlier detection methods. Once the datasets are filtered, we evaluate each learning algorithm using 10-fold cross-validation using the filtered dataset for training and the whole data set for testing. We then compare these results to those obtained by training the learning algorithm using all of the instances. In addition to filtering instances, we also identify instances as ISMs, border points or others as described in Section III.

III. PRISM and Instance Types

PReprocessing Instances that Should be Misclassified or PRISM filters instances that *should* be misclassified. By "should be misclassified," we mean that based on the information in the dataset, the label assigned by the learning algorithm is the most appropriate even though it is incorrect. PRISM uses heuristics from Smith et al [21] to identify instances that should be misclassified. They conducted a comprehensive empirical analysis of what causes instances to be misclassified and found that class overlap is the primary contributor to misclassification. The work is summarized here to provide context for the heuristics that are used to identify ISMs. First, each instance is assigned an instance hardness value to determine which instances are intrinsically hard to correctly classify. The instance hardness values for each instance in a set of 57 data sets was collected on 9 learning algorithms using the following equation:

$$instance\ hardness(x) = \frac{\sum_i^N incorrect(LA_i, x)}{N}$$

where x is the data instance, N is the number of learning algorithms, and $incorrect(LA, x)$ is a function returning 1 if an instance x was misclassified by the learning algorithm LA, and 0 otherwise. The hardest instances are those which no learning algorithm correctly classifies and are what PRISM attempts to filter. Their hardness value is 1. To avoid having to run all nine learning algorithms over each novel instance and to generalize the instances that are hard to correctly classify, we use five heuristics to predict instance hardness and to filter the instances.

The first heuristic, *k-Disagreeing Neighbors* (*k*DN), measures the local overlap of an instance in the original task space. The *k*DN of an instance is the percentage of that instance's k nearest neighbors (using Euclidean distance) that do not share its target class value.

$$kDN(x) = \frac{\mid \{y : y \in kNN(x) \wedge t(y) \neq t(x)\} \mid}{k}$$

where $kNN(x)$ is the set of k nearest neighbors of x and $t(x)$ is the target class value associated with x.

The next heuristic examines the disjunct size and measures how tightly the learning algorithm has to divide the task space to correctly classify an instance and the complexity of the decision boundary. The *Disjunct Size* (DS) of an instance is the number of instances in a disjunct divided by the number of instances covered by the largest disjunct in a data set.

$$DS(x) = \frac{\mid disjunct(x) \mid -1}{\max_{y \in D} \mid disjunct(y) \mid -1}$$

where the function $disjunct(x)$ returns the disjunct that covers instance x, and D is the data set that contains instance x. The disjuncts are formed using a C4.5 decision tree, created without pruning and setting the minimum number of instances per leaf node to 1 [16].[1] In a decision tree, the disjuncts are the leaf nodes.

The third heuristic measures an instance's overlap on a subset of the features. C4.5 forms disjuncts but uses pruning. Using a pruned tree, the *Disjunct Class Percentage* (DCP) of an instance is the number of instances in a disjunct belonging to its class divided by the total number of instances in the disjunct.

$$DCP(x) = \frac{\mid \{z : z \in disjunct(x) \wedge t(z) = t(x)\} \mid}{\mid disjunct(x) \mid}$$

The fourth heuristic provides a global measure of overlap using all of the instances and attributes and a measure of the likelihood of an instance belonging to a class. The *Class Likelihood* (CL) of the attribute values for an instance belonging to a certain class is defined as

$$CL(x, t(x)) = \prod_i^{|x|} P(x_i | t(x))$$

where x_i is the value of instance x on its ith attribute. Continuous variables are assigned a probability using a kernel density estimation [9].

The fifth heuristic captures the difference in likelihoods and global overlap. The *Class Likelihood Difference* (CLD) is the difference between the class likelihood of an instance and the maximum likelihood for all of the other classes.

$$CLD(x, t(x)) = CL(x, t(x)) - \operatorname*{argmax}_{y \in Y - t(x)} CL(x, y)$$

Using these heuristics we can identify instances that should be misclassified (ISMs) by assuming that ISMs have high instance hardness values. We can also identify instances as border points and other by assuming that the instance hardness values for border points range between 0.11 and 1, and that other instances have a low instance hardness values. Based on these observations and the correlation between instance hardness and the heuristics, we identify an instance as a ISM or a border point using the following equation.

$$type(x) = \begin{cases} ISM & \text{if} \quad CLD(x, t(x)) < 0\ \&\& \\ & \quad ((DS(x) == 0\ \&\& \\ & \quad DCP(x) < 0.5) || \\ & \quad DN(x) > 0.8) \\ border & \text{else if} \quad (DS(x) == 0\ \&\& \\ & \quad DCP(x) < 1)\ || \\ & \quad DN(x) > 0.2 \\ other & \text{otherwise} \end{cases}$$

That is, an instance is first identified as a ISM if the wrong class has the highest class likelihood for the instance and it meets one of two conditions: 1) the average DN value[2]

[1] Note that C4.5 will create fractional instances in a disjunct for instances with unknown attribute values, possibly leading to DS values less than 1. Such cases are treated as though the disjunct covered a single instance.

[2] To factor out the effect of neighborhood size, we use $DN(x)$ rather than $kDN(x)$, where $DN(x)$ is the average of $kDN(x)$ over all values of k between 1 and 17. Setting DN above 0.8 implies that on average, for every 5 instances in the neighborhood, at least 4 disagree with the instance under consideration.

TABLE II

The average accuracy for the nine considered learning algorithms and voting ensemble on the data sets and the p-values using the Wilcoxon Signed-Ranks test comparing training with the original dataset and training without removing ISMs. $>=$ 10% averages the data sets with at least 10% of the instances being ISMs according to instance type (ISMs, border points, and other). $<$ 10% averages the data sets with less than 10% of the instances being ISMs and "Overall" averages all of the data sets. "Orig" uses all of the data to train the learning algorithms and "NoISM" refers to training the learning algorithms without ISMs.

	Classifier	>= 10%		p-value	<10%		p-value	Overall		p-value
		Orig	NoISM		Orig	NoISM		Orig	NoISM	
All	C4.5	0.601	0.617	**0.0681**	0.860	0.864	**0.1170**	0.802	0.808	0.0244
	IB1	0.529	0.587	0.0032	0.834	0.851	<0.0001	0.765	0.792	<0.0001
	IB5	0.577	0.617	0.0436	0.852	0.869	<0.0001	0.790	0.812	<0.0001
	MLP	0.595	0.626	**0.0571**	0.871	0.883	0.0495	0.808	0.825	0.0113
	NB	0.578	0.581	**0.2451**	0.804	0.815	0.0015	0.753	0.762	0.0017
	Perceptron	0.574	0.597	0.0217	0.851	0.857	**0.2358**	0.788	0.798	**0.0548**
	RBFNet	0.538	0.566	0.0392	0.850	0.856	0.0066	0.779	0.791	0.0021
	RIPPER	0.545	0.584	0.0087	0.846	0.851	0.0436	0.778	0.790	0.0027
	SVM	0.604	0.619	**0.0571**	0.856	0.862	0.0351	0.799	0.807	0.0089
	Average	0.571	0.599	0.0040	0.847	0.856	<0.0001	0.785	0.798	<0.0001
	Voting	0.573	0.599	0.0089	0.885	0.891	**0.0778**	0.814	0.825	0.0041
ISMs	C4.5	0.073	0.061	>**0.05**	0.186	0.158	**0.4090**	0.160	0.136	**0.2843**
	IB1	0.104	0.086	>**0.05**	0.174	0.149	0.0003	0.158	0.135	0.0006
	IB5	0.117	0.078	>**0.05**	0.150	0.133	0.0008	0.143	0.121	0.0033
	MLP	0.184	0.082	0.0197	0.203	0.193	**0.0582**	0.198	0.168	0.0069
	NB	0.051	0.044	>**0.05**	0.133	0.125	**0.2005**	0.115	0.107	**0.1492**
	Perceptron	0.133	0.076	0.0037	0.141	0.128	**0.2912**	0.139	0.117	0.0329
	RBFNet	0.103	0.066	0.0089	0.233	0.174	0.0011	0.204	0.149	0.0001
	RIPPER	0.135	0.108	<0.01	0.216	0.157	**0.0618**	0.198	0.146	0.0179
	SVM	0.192	0.149	0.0392	0.128	0.118	0.0287	0.143	0.125	0.0044
	Average	0.121	0.083	0.0052	0.174	0.148	0.0016	0.162	0.134	0.0001
	Voting	0.090	0.064	>**0.05**	0.183	0.128	0.0146	0.162	0.114	0.0045
Border points	C4.5	0.672	0.709	0.0485	0.755	0.751	**0.4920**	0.737	0.741	**0.0853**
	IB1	0.567	0.664	0.0024	0.632	0.679	0.0001	0.617	0.675	<0.0001
	IB5	0.641	0.719	0.0032	0.660	0.708	0.0001	0.656	0.710	<0.0001
	MLP	0.659	0.729	0.0052	0.752	0.769	**0.2005**	0.731	0.760	0.0060
	NB	0.676	0.683	**0.3707**	0.680	0.696	0.0197	0.679	0.693	0.0036
	Perceptron	0.639	0.682	0.0068	0.720	0.724	**0.3409**	0.702	0.714	**0.1038**
	RBFNet	0.603	0.654	0.0087	0.716	0.731	0.0485	0.691	0.713	0.0009
	RIPPER	0.592	0.663	0.0051	0.722	0.735	**0.2266**	0.693	0.719	0.0071
	SVM	0.676	0.708	0.0150	0.713	0.719	**0.2451**	0.705	0.717	0.0239
	Average	0.636	0.690	0.0015	0.706	0.723	0.0008	0.690	0.716	<0.0001
	Voting	0.662	0.711	<0.01	0.774	0.784	**0.1190**	0.749	0.767	0.0060
Other	C4.5	0.878	0.991	0.0500	0.962	0.969	0.0028	0.945	0.973	0.0003
	IB1	0.892	0.985	0.0050	0.988	0.993	0.0150	0.969	0.992	0.0002
	IB5	0.943	1.000	**NEI**	0.997	0.999	0.0018	0.986	0.999	0.0002
	MLP	0.922	0.982	0.0500	0.977	0.987	0.0006	0.966	0.986	0.0001
	NB	0.882	0.990	**NEI**	0.912	0.918	0.0222	0.906	0.932	0.0052
	Perceptron	0.959	0.876	>**0.05**	0.971	0.975	0.0307	0.969	0.955	0.0170
	RBFNet	0.939	0.973	>**0.05**	0.961	0.968	0.0080	0.957	0.969	0.0024
	RIPPER	0.914	0.928	**NEI**	0.956	0.958	**0.1711**	0.948	0.952	**0.0668**
	SVM	0.990	0.992	**NEI**	0.977	0.981	0.0060	0.979	0.983	0.0044
	Average	0.924	0.968	0.0027	0.967	0.972	0.0001	0.958	0.971	<0.0001
	Voting	0.995	0.998	**NEI**	0.995	0.996	**0.2420**	0.995	0.997	**0.1131**

is greater than 0.8; or 2) the instance is the only instance covered by the unpruned disjunct and the instance belongs to the minority class of the instances covered by a pruned disjunct. Therefore, a ISM disagrees with at least 80% of its neighbors or it is the only instance belonging to a disjunct and after pruning it is a minority class in the disjunct. If an instance is not a ISM, it is evaluated to determine if it is a border point. An instance is a border point if it satisfies one of the following two conditions: 1) the average DN value is greater than 0.2; or 2) the instance is the only instance covered by an unpruned disjunct and all of the instances covered by the pruned disjunct do not belong to the same class. If an instance is neither a ISM nor a border point, it is identified as *other* signifying that there is no or little class overlap. The values chosen in the heuristics were chosen based on empirical data to correlate with instance hardness [21]. In the following experiments, the other outlier detection methods are set to identify the same number of outliers as PRISM detected.

IV. Results

This section presents the results of the experiments. Removing ISMs by PRISM for training increases the classification accuracy significantly. Only accuracy for the perceptron learning algorithm is not significantly changed by training without the ISMs as shown in the top part of Table II. This may be due to the perceptron's inability to overfit the data and causing it to naturally ignore ISMs and hard instances. The change is most significant for the nearest neighbor learning algorithms.

To further determine the effectiveness of training without the ISMs, we examine the accuracy of the learning algorithms in the context of instance types. Table II shows the average accuracy for each of the nine considered learning algorithms and for a voting ensemble according to instance type on the datasets, where "$>= 10\%$" averages the data sets with at least 10% of the instances being ISMs, "$< 10\%$" averages those data sets with less than 10% of the instances being ISMs and "Overall" averages all of the data sets. "Orig" uses all of the data to train the learning algorithms and "NoISM" refers to training the learning algorithms without ISMs. The *p*-values are from the Wilcoxon signed-rank test and values in bold represent those that are not statistically significant with alpha equal to 0.05. The average classification accuracy and the voting ensemble are included to provide insight at a higher level than an individual learning algorithm. The change in classification accuracy by removing the ISMs for training is statistically significant in the majority of the cases.

The classification accuracy on the ISMs decreases for all learning algorithms when filtering with PRISM, which is expected since ISMs should not be learned by the learning algorithms and ISMs should be misclassified. The change is not significant for C4.5 and naïve Bayes. When 10% or more of the instances are ISMs, the change in accuracy is not significant for C4.5, IB1, IB5, and naïve Bayes while C4.5, MLP, naïve Bayes, perceptron, and RIPPER are not statistically significant when less than 10% of the instances are ISMs. The non-significance is important as it shows that the learning algorithms correctly misclassify ISMs. RBFNet and SVM appear to be most affected by ISMs as the change in classification accuracy is always significant.

By removing the ISMs for training, there is significant improvement for all of the learning algorithms on the border points except for C4.5 and perceptron. The increase is by as much as 10% for the data sets where 10% or more of the instances are ISMs. Only the naïve Bayes learning algorithm does not have a significant increase in classification on the border points on datasets with 10% or more ISMs. The increase is less or not significant on the border points on datasets with less than 10% ISMs and actually decreases slightly for C4.5.

For the other instances, there was not enough information (NEI in the table) to determine if the change in classification accuracy was significant for some of the learning algorithms. This is due in part to both methods being equivalent on some data sets. The change is significant for all of the learning algorithms except for RIPPER and the voting ensemble, although removing the outliers does not decrease the classification either. For all of the other learning algorithms, there is a significant increase in classification accuracy. Thus, the presence of ISMs does affect non-ISM instances.

We also compared PRISM with other filtering techniques. When comparing the other filtering methods, only 48 of the 53 data sets are used. Five of the data sets were omitted because the distance approach, LOF, and/or ECODB ran out of memory (15 GB) when running on them. To determine statistical significance, we used the Friedman test and post-hoc tests as well as the Wilcoxon Signed-Ranks test as suggested by Demšar [5].

TABLE III
THE AVERAGE CLASSIFICATION ACCURACY FOR EACH LEARNING ALGORITHM TRAINED WITH AND WITHOUT FILTERING.

	Orig	Dist	LOF	ECODB	RENN	PRISM
C4.5	0.803	0.794	0.802	**0.807**	**0.805**	**0.809**
IB1	0.771	0.773	0.773	0.784	**0.809**	0.797
IB5	0.791	0.789	0.793	0.802	**0.822**	0.814
MLP	0.813	0.814	0.814	0.822	**0.829**	**0.831**
NB	0.765	**0.773**	0.767	**0.772**	**0.774**	**0.776**
Percept	0.801	0.803	0.798	**0.808**	**0.811**	**0.812**
RBFNet	0.796	0.791	0.792	0.797	**0.807**	**0.806**
RIPPER	0.787	0.787	0.788	0.792	0.790	**0.798**
SVM	0.805	0.803	0.801	**0.810**	0.808	**0.814**
Overall	0.792	0.792	0.792	0.799	**0.806**	**0.806**

Table III shows the average classification accuracy on the test sets with and without filtering for each learning algorithm trained on the different subsets of data used for training. "Orig" refers to using the whole data set for training. The "Dist," "LOF," "ECODB," "RENN," and "PRISM" columns remove instances that were identified as outliers for training using the distance approach, LOF, ECODB, RENN, and PRISM methods respectively. The values in bold are the highest classification accuracy or within 0.5% of the highest classification accuracy for each learning algorithm. RENN and PRISM achieve the highest overall classification accuracy and a higher classification accuracy than Orig for all of the learning algorithms.

TABLE IV
THE AVERAGE RANK FOR EACH LEARNING ALGORITHM ON 48 DATA SETS TRAINED WITH AND WITHOUT FILTERING.

Algorithm	Orig	Dist	LOF	ECODB	RENN	PRISM
C4.5	3.38	4.33	3.58	3.44	3.46	**2.81**
IB1	4.30	4.29	4.10	3.38	**2.24**	2.69
IB5	4.08	4.48	4.08	3.51	2.43	**2.42**
MLP	3.47	4.10	3.88	3.65	3.08	**2.82**
NB	4.06	3.53	3.67	3.5	3.33	**2.91**
Percept	3.44	3.75	3.79	3.76	3.30	**2.96**
RBFNet	3.63	3.94	3.84	3.74	3.01	**2.84**
RIPPER	3.98	3.83	3.77	3.02	3.72	**2.68**
SVM	3.54	3.73	3.70	3.5	3.64	**2.90**
Overall	3.76	4.00	3.82	3.50	3.13	**2.78**

The effectiveness of each method is further shown by ranking the classification accuracy for each filtering method as shown in Table IV. The lowest rank is desired. The values in bold indicate the best average rank. Removing outliers using PRISM has the lowest rank overall and for

TABLE V

THE AVERAGE CLASSIFICATION ACCURACY FOR EACH LEARNING ALGORITHM TRAINED WITH VARIOUS SUBSETS OF THE DATA SET.

Data Set	% ISMs	Orig	Dist	*p*-value	LOF	*p*-value	ECODB	*p*-value	RENN	*p*-value	PRISM	*p*-value
Breast Uterus	0.009	0.961	0.959	>0.05	0.962	>0.05	0.961	>0.05	0.966	>0.05	**0.972**	**<0.025**
ar1	0.05	0.901	0.917	**0.005**	0.881	**0.025**	0.907	>0.05	**0.926**	**<0.05**	0.915	**0.05**
cm1 req	0.056	0.725	0.734	>0.05	0.742	>0.05	0.74	>0.05	**0.775**	**0.01**	0.774	**0.025**
desharnais	0.148	0.641	0.664	>0.05	0.636	>0.05	0.641	>0.05	**0.665**	>0.05	0.661	>0.05
eucalyptus	0.179	0.578	0.56	>0.05	0.553	**<0.025**	0.576	>0.05	**0.612**	>0.05	0.600	**<0.05**
pasture	0.056	0.713	0.698	>0.05	0.688	>0.05	0.71	>0.05	0.772	**0.01**	**0.744**	**<0.05**
Average	0.083	0.753	0.755	>0.05	0.744	>0.05	0.756	>0.05	**0.786**	**0.05**	0.778	**0.01**

each learning algorithm except for IB1. PRISM is the only outlier detection method that is ranked lower than using the original dataset for every learning algorithm. PRISM's overall ranking is the furthest away from 3.5 and only PRISM and RENN have a ranking lower than 3.5. The Friedman test rejects the null hypothesis that each subset of the original dataset used for training is equivalent with a *p*-value less than 0.01. Thus, classification accuracy is affected by removing outliers. The Friedman test only rejects the null hypothesis that every method should have the same rank (3.5 in this case). To compare the different methods for identifying outliers, we perform post-hoc tests as described by Demšar [5].

The Bonferroni-Dunn significance test was used to compare training the learning algorithms trained with filtered data sets against using the original data set for training. The test indicated that the distance approach (decrease in classification accuracy) and RENN and PRISM (increase in classification accuracy) are statistically different from training with the original data set with an alpha value of 0.05. We also compared PRISM against the distance approach, LOF and ECODB. The difference in classification accuracy between using PRISM to remove outliers and the other approaches is statistically significant with an alpha value of 0.05. These results were confirmed using Holm's and Hochenberg's procedures.

This shows that preprocessing a dataset before training affects classification accuracy. Filtering by PRISM and RENN do the best at increasing the classification accuracy overall. To a lesser extent, ECODB in general also increases classification accuracy and is statistically significant using the Wilcoxon signed-ranks test. Also, the increase in classification accuracy by PRISM and RENN is statistically significant and the decrease in accuracy by the distance approach and LOF are statistically insignificant with alpha equal to 0.05.

We also examined removing outliers during training on a test set of six non-UCI datasets drawn from a number of different fields: bioinformatics [22], agriculture [23], and software engineering [19] to demonstrate that PRISM is effective on novel data sets. Because the heuristics were discovered on UCI datasets, this set of data sets was used as a test set to ensure that the heuristics generalize well. The percentage of ISMs in the data sets range from 0.9% to 17.9%. The results are summarized in Table V. Each row gives the percentage of ISMs that each data set contains, the average accuracy for each training set and the *p*-value for the change in accuracy using the Wilcoxon signed-rank test. The highest classification accuracies and the *p*-values that are statistically significant are in bold with alpha equal to 0.05. PRISM and RENN achieve the highest classification accuracy on each data set and provide an average increase of 2.5%. The other methods often result in lower classification accuracy. The change in accuracy by PRISM is also statistically significant for all of the data sets except for desharnais where as RENN is statistically significant for only three of the datasets. Table VI shows how each learning algorithm performed on the additional set of data sets. PRISM and RENN provide the highest classification accuracy for all of the learning algorithms.

TABLE VI

THE AVERAGE CLASSIFICATION ACCURACY FOR EACH LEARNING ALGORITHM TRAINED WITH VARIOUS SUBSETS OF THE ADDITIONAL DATA SETS.

	Orig	Dist	LOF	ECODB	RENN	PRISM
C4.5	0.780	0.769	0.751	0.775	0.781	**0.808**
IB1	0.733	0.734	0.718	0.736	**0.792**	0.769
IB5	0.737	0.753	0.729	0.748	**0.792**	0.771
MLP	0.745	0.758	0.771	0.766	**0.806**	0.796
NB	0.727	0.721	0.720	0.744	0.760	**0.763**
Percept	0.746	0.761	0.751	0.752	**0.787**	0.769
RBFNet	0.751	0.733	0.725	0.736	**0.767**	0.747
RIPPER	0.785	0.789	0.749	0.772	0.787	**0.805**
SVM	0.774	0.780	0.779	0.775	**0.801**	0.772
Overall	0.753	0.755	0.744	0.756	0.786	0.778

Finally, we examine the effect of traditional outliers. We have shown that ISMs affect the classification accuracy of a data set by pulling the classification boundary toward the wrong class. Outliers could also pull the classification boundary in the other direction. Could the accuracy be increased even more by handling the outliers as well as the ISMs? To determine the effect of removing both ISM and outliers for training, we combine the set of ISMs from PRISM with the outliers using the distance method, LOF, and ECODB. 32.7%, 36.7%, and 37.8% of the outliers identified by the distance approach, LOF, and ECODB overlap with the set of ISMs found by PRISM.

Combining the instances identified by PRISM and an outlier detection method results in a loss of accuracy compared to using PRISM by itself. The loss, however, is only 0.8% on average and is *not* statistically significant with an alpha of 0.05 using the Wilcoxon signed-rank test. By definition, outliers are instances that are different than other instances and/or are in an underrepresented area of the task space. By removing the outliers, there is no information for the learning algorithm to generalize well on them. Thus, removing them can be detrimental to the classification accuracy. However, removing the combination of outliers identified by PRISM and another outlier detection method resulted in a higher classification accuracy than just using the other outlier detection method. For each outlier detection method, the increase in accuracy is statistically significant with a *p*-value less than or equal to 0.0001. Thus, ISMs most directly affect the classification boundaries produced by the learning algorithms.

V. Related Work

Outlier detection has received growing attention, especially from the data mining community where outliers may represent anomalies or points of focus [1, 11, 15]. One difficulty in outlier detection is that there is no agreed upon definition of what constitutes an outlier. As such, outlier detection methods have used synthetic data sets or have injected noisy instances into a data set to establish which instances are outliers, thereby making assumptions about the characteristics of outliers. Also, there are many outlier detection algorithms from a variety of fields using different approaches; a few techniques are reviewed here. Khoshgoftaar et al [10] use a rule-based outlier detection method to remove outliers. They analyzed their approach by artificially injecting noise into clean data from software measurement data of a NASA software project. Liu et al [12] present an ensemble method for detecting outliers similar to boosting. In boosting, each training instance is assigned a weight. This method is augmented by adding a weight to each attribute (information gain) and outliers are detected by comparing the weights of the training instances. Finally, rules are generated that result in the largest attribute weight information gain. An approach loosely related to density-based clustering is Local Outlier Factor (LOF) [3]. LOF assigns each instance a value representing its potential of being an outlier with respect to the instances in its neighborhood. A thorough survey of outlier detection methodologies is provided by Hodge and Austin [8].

The concept of class outlier mining has also been examined [7, 14]. The goal of class outlier mining is to detect outliers taking into account the class label. For example, Semantic Outlier Factor (SOF) [6] is a class outlier mining approach based on applying a clustering technique that takes the class label into account. ECODB [18], used in this work, is another example of class outlier mining. PRISM is similar to these approaches in that it takes the class label into account. PRISM differs from the other methods in that it also takes into account the expected classification of the instance.

Closely related to class outlier mining is noise reduction [24, 20] that attempts to identify and remove mislabeled instances. For example, Brodley and Friedl [4] attempt to identify mislabeled instances using an ensemble of classifiers. Rather than determining if an instance is mislabeled, PRISM filters instances that should be misclassified. The sets of removed instances from the PRISM and other noise reduction techniques will expectedly be similar. A different approach by Zeng and Martinez [26] uses multi-layer perceptrons that changes the class label on suspected outliers assuming that the wrong label was assigned to that instance. Our work does not focus on a single learning algorithm, but rather examines the effects of instances that should be misclassified in a broader context.

VI. Conclusions

In this paper we introduced PRISM, a novel filtering method that identifies instances that should be misclassified (ISMs). We used a composite heuristic to identify ISMs that combines ideas from multiple learning algorithms. We have shown that noise and outliers do affect how learning algorithms model the data. However, noise and outlier detection and removal is difficult because there is no universal definition of what an outlier actually is or if an instance is noisy. In addition to PRISM, we used 3 outlier detection approaches and 1 noise reduction method to train 9 learning algorithms with filtering and compared the results to those from the learning algorithms trained using the original data set. RENN and PRISM both resulted in higher classification accuracy and consistently ranked better than the other approaches. PRISM consistently ranked the best among all of the filtering approaches. The distance-based approach and LOF did not show an improvement in classification accuracy but did allow a speed up in training by having less instances to train. The distance-based approach and LOF both ranked worse than training with the original data set.

Removing instances identified by RENN and PRISM for training achieved the highest overall classification accuracy compared with the learning algorithms trained on the original data sets as well as with outliers removed by the other methods. With PRISM, we were able to achieve improvements in classification accuracy regardless of the learning algorithm being evaluated. On average, the increase in accuracy was about 1.3%. However, on data sets where more than 10% of the instances are ISMs, the increase on average is 2.8% compared to 1.2% for data sets with less than 10% ISMs. Rather than focusing on correctly classifying the instances that should be misclassified and arbitrarily adjusting the classification boundary, removing the ISMs for training allows the learning algorithms to focus on the instances that can be correctly classified. Removing the ISMs allows a more appropriate decision surface to be discovered since the ISMs do not arbitrarily pull the decision surface from its more optimal position. This leads to higher classification accuracy.

The presence of noise and outliers affects the learned model as the accuracy on border points and other instances increases when the model is trained on filtered data. Learning algorithms such as C4.5 and multi-layer perceptrons are more robust to outliers in the training data than other models but removing the outliers for training improved their classification accuracy as well as the less robust learning algorithms. Examining each instance type, the accuracy for the ISMs decreased as would be expected, but the accuracy of the border points and other instances increased sufficiently to provide an overall increase in accuracy despite the decrease on the ISMs.

Another advantage to identifying ISMs is for evaluation. The instances that should be misclassified can be handled differently. For example, all of the outliers can be ignored when calculating classification accuracy since the outliers should be misclassified. The accuracy would then give more insight as to how closely the learning algorithms models the data. Using this approach for evaluation, ignoring the ISMs during training increases the classification accuracy on average by 1.94% as compared to when training using all of the instances. Ignoring outliers during training is most effective with a high percentage of instances being outliers. When 10% or more of the instances are outliers, the average increase in classification accuracy is 5.0% compared to 1.1% for data sets with less than 10% outliers.

Outliers and noise affect how learning algorithms model a data set. By filtering noise and outliers for training, the classification accuracy can be improved and the model will more effectively model the data.

References

[1] N. Abe, B. Zadrozny, and J. Langford. Outlier detection by active learning. In *Proc. of the 12th ACM SIGKDD int. conf. on knowl. discov. and data min.*, pages 504–509, New York, NY, USA, 2006. ACM.

[2] A. Asuncion and D. J. Newman. UCI machine learning repository, 2007.

[3] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander. Lof: identifying density-based local outliers. *SIGMOD Record*, 29(2):93–104, June 2000.

[4] C. E. Brodley and M. A. Friedl. Identifying mislabeled training data. *Journal of Artificial Intelligence Research*, 11:131–167, 1999.

[5] J. Demšar. Statistical comparisons of classifiers over multiple data sets. *J. Mach. Learn. Res.*, 7:1–30, 2006.

[6] Z. He, S. Deng, and X. Xu. Outlier detection integrating semantic knowledge. In *WAIM '02: Proc. of the Third Int. Conf. on Advances in Web-Age Inf. Management*, pages 126–131, 2002.

[7] Z. He, J. Z. Huang, X. Xu, and S. Deng. Mining class outliers: Concepts, algorithms and applications. In *WAIM '04: Proc. of the Fifth Int. Conf. on Advances in Web-Age Inf. Management*, pages 589–599, 2004.

[8] V. Hodge and J. Austin. A survey of outlier detection methodologies. *artif. intell. Review*, 22(2):85–126, 2004.

[9] G. H. John. Robust decision trees: Removing outliers from databases. In *knowl. discov. and Data min.*, pages 174–179. AAAI Press, 1995.

[10] T. M. Khoshgoftaar, N. Seliya, and K. Gao. Rule-based noise detection for software measurement data. In *Proc. of the IEEE int. conf. on inf. Reuse and Integration*, pages 302–307. IEEE Syst., Man, and Cybern. Society, 2004.

[11] J. M. Kubica and A. Moore. Probabilistic noise identification and data cleaning. In *The Third IEEE int. conf. on Data min.*, pages 131–138. IEEE Comput. Society, November 2003.

[12] X.-D. Liu, C.-Y. Shi, and X.-D. Gu. A boosting method to detect noisy data. In *Proc. of 2005 int. conf. on Machine Learning and Cybern.*, volume 4, pages 2015–2020, 2005.

[13] I. Mierswa, M. Wurst, R. Klinkenberg, M. Scholz, and T. Euler. Yale: Rapid prototyping for complex data mining tasks. In *KDD '06: Proc. of the 12th ACM SIGKDD int. conf. on knowl. discov. and data min.*, pages 935–940, New York, NY, USA, 2006. ACM.

[14] S. Papadimitriou and C. Faloutsos. Cross-outlier detection. In *SSTD '03: Proc. of the Eighth Int. Symposium on Advances in Spatial and Temporal Databases*, pages 199–213, 2003.

[15] M. I. Petrovskiy. Outlier detection algorithms in data mining systems. *Programming and Comput. Software*, 29(4):228–237, 2003.

[16] J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA, USA, 1993.

[17] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. *Int. Conf. on Management of Data (SIGMOD)*, 29(2):427–438, 2000.

[18] M. K. Saad and N. M. Hewahi. A comparative study of outlier mining and class outlier mining. *CS Letters*, 1(1), 2009.

[19] J. Sayyad Shirabad and T. Menzies. The PROMISE Repository of Software Engineering Databases. School of Information Technology and Engineering, University of Ottawa, Canada, 2005.

[20] N. Segata and E. Blanzieri. Fast and scalable local kernel machines. *Journal of Machine Learning Research*, 11:1883–1926, August 2010.

[21] M. R. Smith, T. Martinez, and C. Giraud-Carrier. An empirical study of instance hardness. Submitted to ICML 2011.

[22] G. Stiglic and P. Kokol. GEMLer: Gene expression machine learning repository [http://gemler.fzv.uni-mb.si/]. University of Maribor, Faculty of Health Sciences, 2009.

[23] K. Thomson and R. J. McQueen. Machine learning applied to fourteen agricultural datasets. Technical Report 96/18, The University of Waikato, September 1996.

[24] I. Tomek. An experiment with the edited nearest-neighbor rule. *IEEE Transactions on Systems, Man, and Cybernetics*, 6:448–452, 1976.

[25] I. H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, San Fransisco, 2nd edition, 2005.

[26] X. Zeng and T. R. Martinez. An algorithm for correcting mislabeled data. *intell. Data Analysis*, 5:491–502, 2001.