2009-11-20

An Empirical Study of Instance Hardness

Michael Reed Smith
Brigham Young University - Provo

An Empirical Study of Instance Hardness

Michael R. Smith

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computer Science

Brigham Young University

April 2010

ABSTRACT

An Empirical Study of Instance Hardness

Michael R. Smith

Department of Computer Science

Master of Science

Most widely accepted measures of performance for learning algorithms, such as accuracy and area under the ROC curve, provide information about behavior at the data set level. They say nothing about which instances are misclassified, whether two learning algorithms with the same classification accuracy on a data set misclassify the same instances, or whether there are instances misclassified by all learning algorithms. These questions about behavior at the instance level motivate our empirical analysis of *instance hardness*, a measure of expected classification accuracy for an instance. We analyze the classification of 57 data sets using 9 learning algorithms. Of the over 175000 instances investigated, 5% are misclassified by all 9 of the considered learning algorithms, and 15% are misclassified by at least half. We find that the major cause of misclassification for hard instances is class overlap, manifested as outliers and border points which can be exacerbated by class skew. We analyze these causes and show to what extent each leads to misclassifications, both in isolation and jointly. 19.8% of all misclassified instances are outliers; 71.3% are border points; 21% belong to a minority class. We also find that 91.6% of all outliers and 38.3% of all border points are misclassified whereas only 3.5% of instances without class overlap are misclassified. We propose a set of heuristics to predict when an instance will be hard to correctly classify. Additionally, we analyze how different learning algorithms perform on tasks with varying degrees of outliers, border points and class skew.

ACKNOWLEDGMENTS

I would first like to acknowledge and thank Dr. Tony Martinez, my graduate advisor, for his support, guidance, and counsel through out this research. I am also indebted to my other graduate committee members, Dr. Christophe Giraud-Carrier and Dr. Jay McCarthy, for their input in the research as well as in the writing of this thesis.

Ultimately, I am grateful to my wife for supporting me through out my academic endeavors and for taking a chance on me.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

For classification tasks, generalization accuracy estimation is a common metric for evaluating the performance of a learning algorithm or for comparing learning algorithms on benchmark data sets. While accuracy is easy to compute, it provides only aggregate information about a learning algorithm and the task upon which it operates where a task is represented by a data set comprised of instances. Classification accuracy does not provide information about which instances were misclassified and why they were misclassified. Two learning algorithms are viewed to perform the same if they have the same classification accuracy, but are the same instances misclassified by both learning algorithms? Are some instances misclassified by all learning algorithms? Are some instances harder to classify than others and why?

Understanding why instances are misclassified could lead to the development of learning algorithms that address the causes of misclassification. The evaluation of learning algorithms would be more informative with an intuition of which instances are expected to be misclassified and which instances are expected to be correctly classified.

We are not the first to look at instances that are hard to classify. Previous studies [21, 18, 34, 25] have started from the premise that outliers, boundary points, or instances belonging to a minority class are hard to classify and have generally been carefully designed around one of these issues. For example, outlier detection often uses artificial data sets [1] or systematically injects noise into well-known data sets [20]. Class imbalance studies restrict their attention to binary classification tasks [10, 13]. Previous studies have also mostly considered only a limited number of data sets and algorithms.

In this paper, we argue that observing how each individual instance is classified provides insight into a learning algorithm's bias and coverage of the task space. We empirically analyze the classification of 57 data sets by 9 learning algorithms. We focus explicitly on instances that are hard to classify in the sense that most learning algorithms misclassify them, and seek to shed light on the reasons for such misclassifications. Contrasting previous work, we use a large number of data sets and algorithms. Furthermore, the selected data sets have not been artificially altered but have been selected to represent a variety of tasks along such dimensions as the number of attributes, the types of attributes, the number of classes, and the application domain. While the large number of learning algorithms and unaltered data sets makes it more challenging to identify the nature of instances, it does allow us to offer insight into why instances are misclassified independent of the learning algorithm.

Determining whether the misclassified instances can reasonably be expected to be misclassified is a difficult problem especially with high dimensional data sets. By "reasonably expected to 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. Most learning algorithms have some means of preventing overfit and removing subconcepts that are not believed to be meaningful to generalize on the training data. Hence, instances that are outliers and instances that lie near a classification boundary are expected to be misclassified more frequently. Similarly, instances belonging to a class with significantly fewer instances than another class (class skew) are also more frequently misclassified because high accuracy may be achieved by only learning the majority class(es) and ignoring the minority class(es). We propose a set of heuristics to identify instances that can reasonably be expected to be misclassified and compare them to the instances that are misclassified by all or most of the learning algorithms. The combination of the heuristics can be used to predict instance hardness on new data sets. We also expand these heuristics to identify instances that are outliers, border points, or belong to a minority class.

Of the over 175000 instances investigated, 5% are misclassified by all 9 of the learning algorithms we consider, and 15% are misclassified by at least half of the learning algorithms.

This is a significant number of misclassified instances and justifies our analysis of instance hardness. In examining the cause of instance hardness we find that 19.8% of all misclassified instances are outliers, 71.3% are border points, and 21% belong to a minority class. We find that class overlap, including outliers and border points, most directly affects classification accuracy. The considered learning algorithms misclassify 91.6% of all outliers and 38.3% of all border points whereas they only misclassify 3.5% of the instances without class overlap. Additionally, 39% of the instances misclassified by at least half of the learning algorithms are outliers and should have been misclassified. 59% of the instances misclassified by at least half of the learning algorithms are border points. Differing from previous studies, we find that class skew by itself does not affect classification accuracy but exacerbates the effects of class overlap. The considered learning algorithms misclassify 41.9% of the instances belonging to a minority class. However, 40% of the instances that belong to a minority class are also identified as outliers or border points.

The contributions of this paper include:

- We introduce the notion of instance hardness to measure how hard an instance is to classify.
- We present a set of heuristics that measures instance characteristics to determine what causes learning algorithms to misclassify instances.
- We present an extensive empirical analysis of instance hardness to characterize instances that are misclassified.
  - We show that a set of instances exists that is misclassified by all the considered learning algorithms.
  - We present a set of heuristics that identifies instances as outliers, border points, or belonging to a minority class.
  - We show that our heuristics predict instance hardness and data set hardness.
  - We identify class overlap as the cause of instance hardness and misclassifications.

– We show that class skew by itself does not cause an instance to be misclassified but exacerbates the effects of class overlap.

Section 2 provides more details on related work. Section 3 proposes the notion of *instances hardness*. We expand instance hardness to the data set level with data set hardness. Section 4 presents a set of heuristics (*hardness heuristics*) that measure instance characteristics to determine what affects instance hardness. Section 5 describes the experimental methodology and which data sets and learning algorithms are used in this study. The empirical analysis of the results is given in Sections 6-10. Section 6 analyzes the existence of hard instances. Section 7 evaluates the heuristics presented in Section 3 against instance hardness. From this evaluation, heuristics are developed that identify instances as outliers, border points, or belonging to a minority class. Section 8 examines what causes instances to be misclassified and shows that the hardness heuristics predict instance hardness. Section 9 examines hardness at the data set level using data set attributes employed in meta learning as well as the heuristics presented in this paper. Section 10 shows how the learning algorithms perform in the contexts of instance hardness and the heuristics presented in this paper. Section 11 concludes and gives directions for future work.

Chapter 2

Related Work

Significant research efforts have expanded to identify instances which learning algorithms are likely to misclassify. Outlier detection has received growing attention, especially from the data mining community where outliers may represent anomalies or points of focus [1, 21, 24, 27]. For example, in a credit card system a fraudulent transaction may appear as an outlier and should receive more attention than the normal transactions. 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. There are many outlier detection algorithms from a variety of fields using different approaches; a few techniques will be reviewed here. Khoshgoftaar et al. [20] suggest using 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. [23] 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) [7]. 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 [15].

Work has also been done to identify instances that lie on or near classification boundaries [8, 25], although to a lesser extent than that for outlier detection. Most of the attention for border instances has come from instance reduction techniques which try to avoid storing more instances than are necessary to generalize well on the data. Support vector machines

(SVM) [9] create a hyperplane that separates the classes and maximizes the distance of the hyperplane from the nearest instances of the opposing classes. In other words, only the instances closest to the classification boundary are used. Wilson and Martinez [39] present a survey of instance-based reduction techniques as well as propose their own. These and similar algorithms attempt to smooth the decision boundary by removing outliers and by only keeping enough boundary instances to maintain good classification accuracy. On the other hand, some instance-based reduction techniques only keep a central representation of the instances and discard the outliers and some border points [42].

Class skew has also received a lot of attention [4, 10, 13, 14, 34]. Class skew refers to a data set consisting of one or more classes heavily outnumbering the other class(es). Many learning algorithms have difficulties learning the concepts of the minority class(es) because learning algorithms maximize classification accuracy. Most previous work has used undersampling, oversampling, and cost-sensitive techniques and has been limited to binary classification tasks. In relationship to identifying instances which are hard to correctly classify, class imbalance also affects outliers and boundary instances. In some cases, instances are outliers because there is insufficient data and the learning algorithm excludes the instance through generalization. Weiss [36] addresses this issue from a data mining perspective and provides possible causes and solutions to working with under-represented classes and attributes. Class skew also affects the identification of boundary instances. Akbani et al. [2] utilize SMOTE [10] (an oversampling technique) in conjunction with SVMs to address the class imbalance problem. The resultant support vectors provide information about the class boundaries. Ertekin et al. [11] use active learning to provide a sampling of more balanced classes by focusing on instances that lie near the border.

Related to the idea of class skew is the notion of small disjuncts [16, 18, 35, 37, 38]. Some learning algorithms, such as decision trees and rule-based learning algorithms, can express the learned concept as a disjunctive description. While examining class skew, some have concluded that class skew is not the problem, rather the difficulty in classification is a result of the instances that belong to a small disjunct (small referring to the number of instances of the training set that are covered by a disjunct). Quinlan [30] examines the problem of predicting the error of small disjuncts and concludes that disjunct size alone is insufficient to predict accuracy and that classification accuracy is lowest for the small disjuncts of the minority class. Japkowicz [17] questions if class skew affects classification accuracy and concludes that class skew is a relative problem depending on the number of disjuncts that the classes are divided into and the overall size of the training set.

Previous work has focused on a single issue at a time. Our work differs from prior work in that we do not focus on a single issue as a cause for an instance being misclassified. We focus our analysis on discovering the underlying causes for instances being misclassified from a broad perspective. Our analysis is extensive in the number of learning algorithms and the number of data sets. Also, we do not alter the data sets.

In addition to outliers, border points, and class skew, our work also relates to metalearning. Metalearning studies have been conducted to predict which learning algorithm to use based on data set features [5, 32, 28, 22, 19, 6, 33]. While in metalearning the prediction is driven by accuracy and thus performance at the data set level, we focus here on the instance level. Using heuristics in conjunction with the classification of various learning algorithms we attempt to characterize instances which are commonly misclassified. We also examine features at the data set level but we are more interested in predicting the instances that will be misclassified for a particular task, rather than which learning algorithm performs the best. We also mention COD (Classifier Output Difference) [26], which is a distance metric to measure learning algorithm similarity. COD uses classification accuracy as well as how often the learning algorithms have the same output to determine how similar learning algorithms are. While our work is similar in that we are interested in more than just the classification of an instance, we focus on what causes instances to be misclassified rather than on how similar the learning algorithms are. Also, our focus is on each instance rather than the data set.

Chapter 3

Instance Hardness

Our first contribution is the notion of *instance hardness*. We define *instance hardness* as the ratio of the number of learning algorithms which incorrectly classify an instance to the total number of learning algorithms, as described in 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 a trained learning algorithm LA, and 0 otherwise. The hardest instances are those which no learning algorithm correctly classifies. Their hardness value is 1. Instance hardness is necessary to determine which instances are hard to classify.

It is possible to obtain an aggregate value of hardness for a complete data set. Indeed, we define *data set hardness* by averaging instance hardness over the instances in a data set, namely:

$$data\ set\ hardness(D) = \frac{\sum_{x \in D} instance\ hardness(x)}{\mid D \mid}$$

where D is the data set. Clearly, the hardest data sets have larger than average data set hardness values.

Chapter 4

Hardness Heuristics

Knowing which instances are hard to classify leads to the question of what characterizes them. To aid in this, we present our next contribution: a set of heuristics, which we term the *hardness heuristics*, to identify instances that may be expected to be misclassified more frequently.

Our first heuristic is inspired from Edited Nearest Neighbor (ENN), an extension of k-NN, which edits the training set by removing any instance that does not agree with the majority of its k-nearest neighbors [40]. ENN's procedure edits out the outliers *as well as* the close border points. We propose to use a similar idea that harder instances disagree with their k-nearest neighbors. We define the *k-Disagreeing Neighbors* (kDN) of an instance as the percentage of that instance's k neighbors who 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. kDN gives insight into the local neighborhood of an instance regarding the amount of overlap in the space where the instance is located. One would expect an instance that disagrees with its neighbors (i.e., high kDN value) to be more frequently misclassified than one that does agree with its neighbors.

Our second heuristic is inspired from work on small disjuncts. Recall that some learning algorithms (such as decision trees or rule-based learners) express the learned concept as a disjunction of conjunctions. A disjunct is a single conjunction in the disjunction. Weiss and Hirsh [38] showed that instances that are covered by small disjuncts have a higher misclassification rate than instances that are covered by large disjuncts . We define the *disjunct size* (DS) of an instance as the number of instances in a disjunct divided by the number of instances covered by the largest disjunct in a data set.

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

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 [31].[1] As the size of the disjunct created by C4.5 provides information on how tightly the learning algorithm has to divide the task space to correctly classify an instance and how clearly the decision boundary is defined, one would expect an instance with low DS value to be misclassified more frequently than an instance with a larger DS value.

Our third heuristic complements the DS heuristic. Without pruning (as done with DS), a decision tree creates small disjuncts because a single instance may lie in an area comprised of instances from a different class. When pruning is used, this single instance will be incorporated into a larger disjunct of another class. An instance whose class has a low percentage in the disjunct can be expected to be misclassified more frequently as opposed to an instance whose class comprises the majority of the disjunct. Using a pruned tree, we define the *disjunct class percentage* (DCP) of an instance as the number of instances 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 percentage of an instance's class in a disjunct gives information about the overlap in the space covered by the disjunct, and thus about its likelihood of being misclassified.

Our fourth heuristic builds on the idea of the naïve Bayes learning algorithm, where probability is reduced to likelihood by dropping the term for the prior. We define the *class*

[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.

likelihood (CL) of the attribute values for an instance belonging to a certain class by

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

where x_i is the value of instance x on its ith attribute and $t(x)$ is the class that x belongs to. CL gives insight into how likely the attribute values are for a particular class and a global measure of overlap. Obviously, the higher the likelihood is, the easier an instance should be to correctly classify. CL provides a more global view of the data set since all of the instances are used to compute it.

Our fifth heuristic complements the CL heuristic. CL gives the likelihood that an instance belongs to a certain class, but does not provide information about the likelihoods of the other classes. In an attempt to capture the difference in likelihoods, we define *class likelihood difference* (CLD) by subtracting the class likelihood of an instance from the maximum likelihood for all of the other classes.

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

where $t(x)$ is the class that the instance x belongs to and Y is the set of all classes. The higher the CLD value is, the more likely that the instance belongs to class $t(x)$. If the CLD value is negative then the likelihood for another class was greater than the likelihood for class $t(x)$. CLD thus gives perspective to CL. Suppose that there are two data sets, one with 2 classes and the other with 10. If an instance in the data set with 2 classes has a CL of 55, the CLD is 10. That same instance in the 10 class data set would have a CLD value varying from about 10 to 50.

Our sixth heuristic is intended to capture the skewness of the class an instance belongs to. For each instance, we define its *minority value* (MV) as the ratio of the number of instances sharing its target class value to the number of instances in the majority class.

$$MV(x) = \frac{\mid \{z : z \in D \wedge t(z) = t(x)\} \mid}{\max_{y \in Y} \mid \{z : z \in D \wedge t(z) = y\} \mid}$$

In this way, everything is taken with regard to the majority class. A smaller MV value signifies that the instance's class has fewer instances than the majority class in the data set, while larger MV values represent less class skew.

Our final heuristic offers an alternative to MV. If there is no class skew, then there is an equal number of instances for all classes. Hence, we define the *class balance* (CB) of an instance by:

$$CB(x) = \frac{\mid \{z : z \in D \wedge t(z) = t(x)\} \mid}{\mid D \mid} - \frac{1}{\mid Y \mid}.$$

CB captures how balanced the class distribution is. If the data set is completely balanced the class balance value will be 0. If the value is greater than 0, then the class has more instances than one or more of the other classes. If the CB value is less than 0 then the class has less instances than at least one of the other classes.

Chapter 5

Experimental Methodology

We computed instance hardness, data set hardness, and the hardness heuristics over a wide range of learning algorithms and data sets, and used them to analyze both the extent and the nature of hardness in typical machine learning tasks. All quantities were obtained through 10-fold cross-validation.

A collection of nine learning algorithms was used here. This set was not intended to be exhaustive. Rather, algorithms were selected to represent the diversity in the current state of machine learning, including representatives of each main model class (e.g., function-based, instance-based, tree-based). The learning algorithms that were used in this research are as follows. The names in curly braces denote what the algorithms will be referred to in the experiment tables and figures.

- Decision Tree (J48-Weka's implementation of C4.5 [31]) {C4.5}
- Naïve Bayes {NB}
- Multi-layer Perceptron trained with Back Propagation {MLP}
- Perceptron (A linear discriminator implemented as a multi-layer perceptron using back propagation with zero hidden layers) {Percep}
- Support Vector Machine (Implementation of John Platt's sequential minimal optimization algorithm for training an SVM [29]) {SVM}
- 1-NN (1-nearest neighbor) {IB1}
- 5-NN (5-nearest neighbors) {IB5}
- RIPPER (Repeated Incremental Pruning to Produce Error Reduction) {RIPPER}
- RBF Network (with normalized Gaussian radial basis function) {RBF}

No parameter optimization was performed on any of the algorithms. They were used as implemented in Weka with their default parameters [41]. Obviously by optimizing a learning algorithm for a data set some of the hard instances may be expected to be correctly classified more consistently. Also, many of the hard instances could be correctly classified more consistently with boosting or another cost-sensitive approach. The goal of this research, however, is not to try to eliminate hard instances, but rather to discover their existence and what general attributes characterize them.

Along with these learning algorithms, a collection of 57 data sets was selected from the UCI Machine Learning Repository [3]. They are described in Table 5.1. Again, while this set is clearly not exhaustive, it 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.

We wish to emphasize the extensiveness of our analysis. We examine 178109 instances individually. A total of 5130 models are produced from 9 learning algorithms evaluated against 57 data sets using 10-fold cross-validation. With such volume and diversity, study bias is reduced, and we feel confident that our results provide useful information about the extent to which hard instances exist and what contributes to instance hardness and data set hardness.

# Instances	# Attributes	Attribute Type		
		Categorical	Numerical	Mixed
$M < 100$	$k < 10$	Balloons Contact Lenses		Post- Operative
	$k < 100$	Lung Cancer		Labor Trains
$100 < M < 1000$	$k < 10$	Breast-w Breast Cancer	Iris Ecoli Pima Indians Glass Bupa Balance Scale	Badges 2 Teaching- Assistant
	$10 < k < 100$	Audiology Soybean(large) Lymphography Congressional- Voting Records Vowel Primary-Tumor Zoo	Ionosphere Wine Sonar Heart-Statlog	Annealing Dermatology Credit-A Credit-G Horse Colic Heart-c Hepatitis Autos Heart-h
	$k > 100$			Arrhythmia
$M > 1000$	$k < 10$	Car Evaluation Nursery Chess Titanic	Yeast MAGIC- Gamma- Telescope	Abalone
	$k < 100$	Mushroom Chess- (King-Rook vs. King-Pawn) Splice Letter	Waveform-5000 Segment Spambase Ozone level- Detection	Adult Census- Income (KDD) Thyroid- (sick & hypothyroid)
	$k > 100$		Musk (version 2)	

Table 5.1: Data sets used organized by number of instances, number of attributes, and attribute type.

Chapter 6

Hard Instances

This section shows that a significant amount of hard instances exist over all of the considered instances as well as in each considered data set. Figure 6.1 shows the percentage of instances per instance hardness value. Recall that we are using 9 algorithms. Hence, there are 10 possible levels of instance hardness, ranging from 0 (classified correctly by all algorithms) to 1 (misclassified by all algorithms). The first column shows the percentage over all instances and the second column shows the percentage of instances averaged per data set. As the percentages are very similar, we use the values which are averaged over all data sets so as not to be biased towards larger data sets.

About 53% of the instances have an instance hardness of 0 and 5% of the instances have an instance hardness of 1. 85% of the instances are correctly classified by at least half of the learning algorithms (i.e., instance hardness less than .5) and would be correctly classified by a voting ensemble.

These results show that a significant amount of instances are indeed hard: 5% are misclassified by all of the learning algorithms and 15% are misclassified by a majority of the learning algorithms. Seeking to improve our understanding of why these instances are

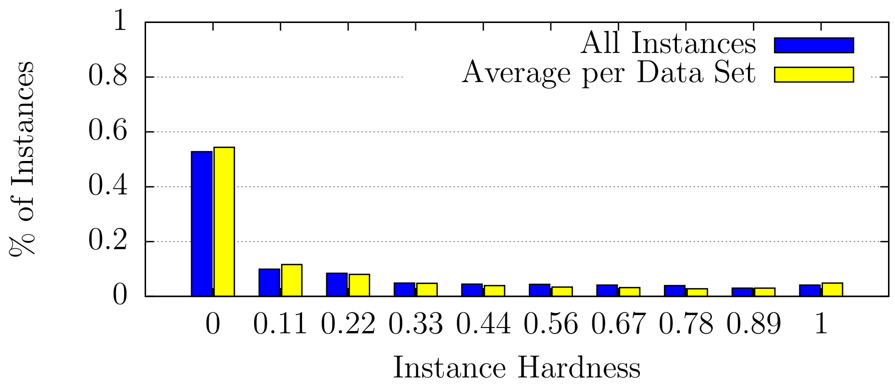


Figure 6.1: Overall instance hardness

misclassified becomes a justifiable quest. It may be particularly informative to examine those instances which have an instance hardness greater than 0 but less than 1. These instances are covered by at least one but not all of the learning algorithms, suggesting that it may be possible to classify them correctly. Such instances may be "close" outliers or instances which are not covered as fully due to the inductive bias of certain learning algorithms.

At the data set level, Figure 6.2 shows the distributions of instance hardness by data set, sorted in descending values of data set hardness (value in parenthesis next to the data set name). The axes for each graph are the same as in Figure 6.1, instance hardness vs. percentage of instances.

Nine data sets (data set name in bold in Figure 6.2) do not have any instances of instance hardness 1 (Trains, Labor, Car, Iris, Wine, Musk, Balloons, Mush, and Badges). Hence, for these data sets, each instance was correctly classified by at least one learning algorithm suggesting that it is possible to correctly classify all of the instances and that the task space is completely covered. Four data sets do not have any instances with an instance hardness greater than 0.5 (Musk, Balloons, Mush, and Badges) and thus could achieve 100% classification accuracy with a voting ensemble. Unsurprisingly, these four data sets also have the lowest data set hardness values.

For 43 of the data, at least 35% of the instances have instance hardness 0. The remaining 14 data sets (data set name in bold and italics in Figure 6.1), in which less than 35% of the instances have an instance hardness 0 (Abalone, Primary-tumor, Lung cancer, Chess, Teaching Assistant, Yeast, Trains, Post-operative patient, Arrhythmia, Glass, Bupa, Autos, Colon, and Vowel) (bold and italics) are generally those that have the highest data set hardness values. Interestingly, the classification accuracies of the learning algorithms on these 14 data sets differ significantly from each other, while for the other 43 data sets, the classification accuracies are very similar. This suggests that data set hardness may be an indicator of the classification accuracy of a learning algorithm on a data set as well as of how similar the classification accuracy is for the learning algorithms.

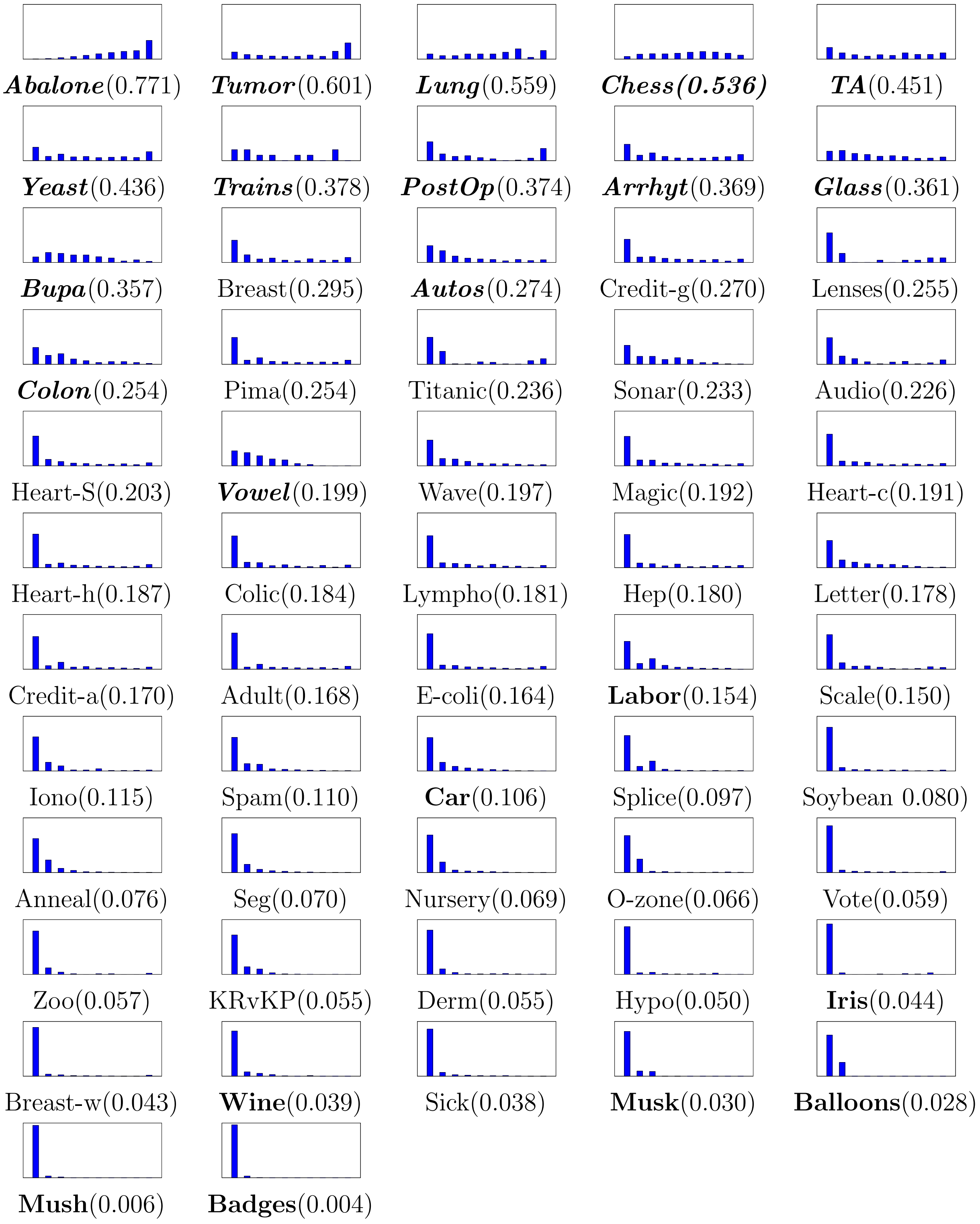


Figure 6.2: Instance hardness for each data set. Data set hardness is in parenthesis.

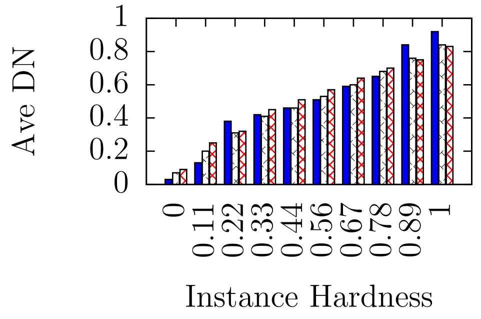
Chapter 7

Evaluation of the Hardness Heuristics

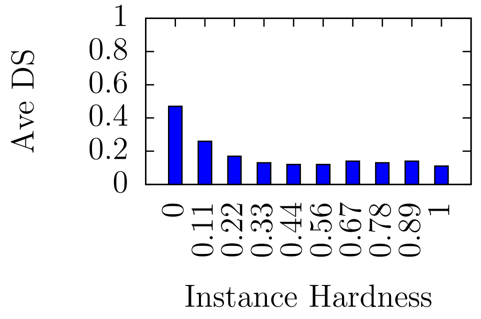
As shown in the previous section, hard instances do exist and to varying degrees. In this section we consider the relationships between instance hardness and the hardness heuristics defined earlier and present a set of heuristics for identifying instances as outliers, border points, or belonging to a minority class. Figure 7.1 depicts these relationships graphically.

These graphs suggest that there is some level of correlation between each of the heuristics and instance hardness, i.e., each seems able to identify instances which are hard to correctly classify. In particular, we observe the following.

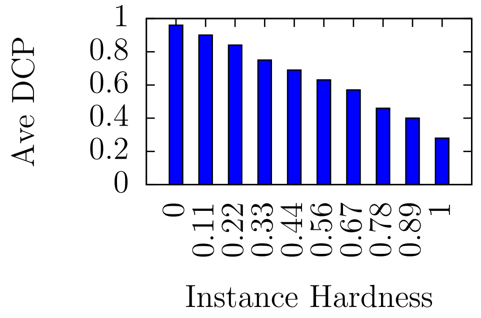
- The value of kDN (k-disagreeing neighbors) increases as k increases for instance hardness values less than 0.8. Conversely, the value of kDN decreases as k increases for instance hardness greater than 0.8. This effect is equivalent to the smoothing effect in image processing. The image becomes more blurred as the value of k is increased until each pixel is an average of all the pixels.
- There is a significant decrease in DS (disjunct size) between instance hardness 0 and 0.11 and again between instance hardness 0.11 and 0.22. From instance hardness 0.22 to 1, the value of DS remains relatively constant, suggesting independence from instance hardness past that point. While it is true that the harder instances belong to smaller disjuncts, the easier instances can also belong to smaller disjuncts. This is because an instance which is an outlier will split a larger disjunct of easier instances. For instance hardness 0, about 32% of the instances have a DS value less than one tenth of the largest disjunct and 31% of the instances belong to a disjunct which is greater than or equal to 90% of the largest disjunct in the data set. Therefore, the disjunct size alone is not a good indicator of instance hardness. If an instance belongs to the largest disjunct then the instance should be easier to correctly classify as 82%



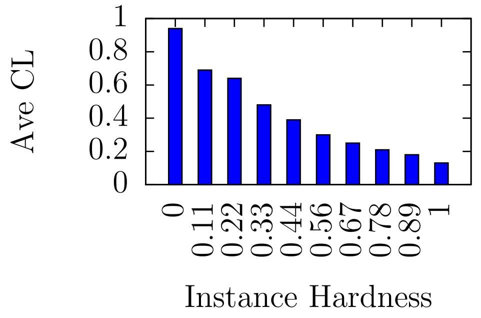
kDN (kDisagreeing Neighbors; k=1,5,17)



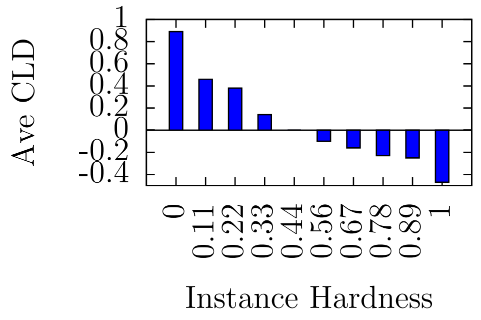
DS (Disjunct Size)



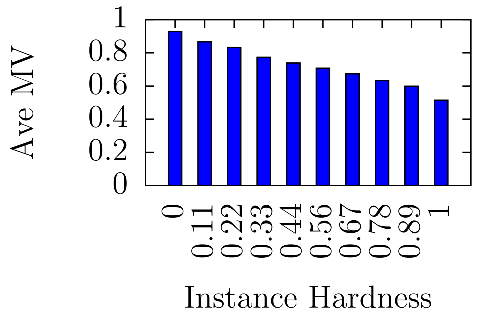
DCP (Disjunct Class Percentage)



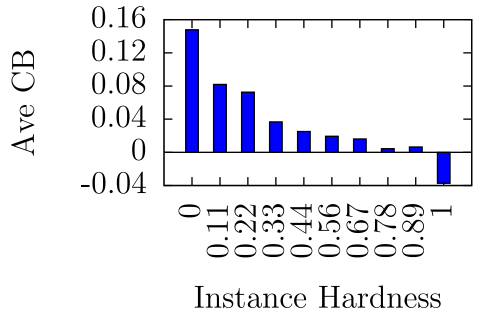
CL (Class Likelihood)



CLD (Class Likelihood Difference)



MV (Minority Value)



CB (Class Balance)

Figure 7.1: Relationship of heuristics to instance hardness

of the instances that belong to the largest disjunct in the data set have an instance hardness less than 0.33. The converse (an instance belonging to a smaller disjunct) is not necessarily true.

- The average value of DCP (disjunct class percentage) decreases as instance hardness increases. Interestingly, combining DCP and DS provides more information about instance hardness than they do individually. 99% of the instances with instance hardness value 1 and DS value 1 have a DCP value less than 0.5. This means that although the disjunct an instance belongs to may be the largest for the data set, the disjunct is often combined with one or more other disjuncts during pruning.

- The value of CL (class likelihood) decreases as instance hardness increases. The most significant decrease in CL is between instance hardness 0 (CL=0.94) and 0.11 (CL=0.69). For instance hardness 0, 83% of the instances have a CL value greater than or equal to 0.9. For instance hardness 0.11, 0.37% of the instances have a CL value greater than or equal to 0.9.

- The value of CLD (class likelihood difference) decreases as instance hardness increases. As with CL, the most notable decrease in CLD is between instance hardness 0 and 0.11, potentially providing a good discriminator between instances with an instance hardness 0 and instances with an instance hardness greater than 0.

- The change in MV (minority value) with respect to instance hardness is not as significant as it is with the other heuristics. The most significant difference between instance hardness values is with instances that have a MV value less than 0.1 (instances that belong to the majority class or a class that is at least 90% of the size of the majority class). There is a consistent decrease in the percentage of instances per instance hardness value that have a MV value less than 0.1.

- CB (class balance) also correlates with instance hardness. There is an obvious difference for instance hardness 1 where the average CB value is negative. There is also a significant decrease in the average CB values between instance hardness 0 and 0.11.

These observations make it possible for us to design composite heuristics to identify instances which may be outliers, border points or belong to a minority class. These heuristics were designed on the premise that outliers are likely to be misclassified and that border points and instances belonging to a minority class are more likely to be misclassified than other instances. The thresholds in the heuristics were chosen based on these assumptions and the evaluation of the hardness heuristics as well as examining some simple data sets (e.g. the iris and glass data sets).

An instance is identified as an outlier or a border point using the following equation.

$$instanceType(x) = \begin{cases} outlier & \text{if } (CLD(x,t(x)) < 0 \ \&\& \ ((DS(x) == 0 \\ & \&\& \ DCP(x) < 0.5) \ || DN(x) > 0.8)) \\ borderPoint & \text{else if } ((DS(x) == 0 \ \&\& \ DCP(x) < 1) \ || \cdot \\ & \ DN(x) > 0.2) \\ noOverlap & \text{otherwise} \end{cases}$$

That is, an instance is first identified as an outlier if the wrong class has the highest class likelihood for the instance ($CLD < 0$) and it meets one of two conditions: 1) the average DN value is greater than 0.8 ($DN(x) > 0.8$);[1] or 2) the instance is the only instance covered by the unpruned disjunct ($DS(x) = 0$) and the instance belongs to the minority class of the instances covered by a pruned disjunct ($DCP(x) < 0.5$). Therefore, an outlier 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 an outlier, it is evaluated

[1]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.

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 ($DN(x) > 0.2$); or 2) the instance is the only instance covered by an unpruned disjunct ($DS(x) = 0$) and all of the instances covered by the pruned disjunct do not belong to the same class ($DCP(x) < 1$). If an instance is neither an outlier nor a border point, it is identified as *noOverlap* signifying that there is no or little class overlap.

An instance is identified as belonging to a minority class independent of class overlap, as follows.

$$minorityClass(x) = \begin{cases} true & MV(x) \leq 0.5 \ \&\& \ CB(x) < 0 \\ false & otherwise \end{cases}.$$

That is, a class is a minority class based on two conditions: the number of instances in the class is less than or equal to half the number of instances belonging to the majority class ($MV(x) \leq 0.5$), and the number of instances in the class is less than the number of instances if all classes were balanced ($CB(x) < 0$).

Chapter 8

Instance-level Analysis

The heuristic values and instance hardness were compiled and linear regression was used to determine which heuristics best predict instance hardness. The resulting model is as follows.

$$\begin{aligned} instance\ hardness =& 0.9088 \\ &+0.5569 * DN \\ &-0.1984 * DCP \\ &-0.124 * CL \\ &+0.0752 * CB \\ &-0.072 * CLD \\ &+0.0365 * DS \\ &+0.0339 * MV \end{aligned}$$

with a correlation coefficient of 0.8856. The heuristics with the largest coefficients are DN, DCP, and CL. These heuristics give information about the amount of class overlap, primarily on a local level. There is no heuristic that relates to class skew in the equation, which coincides with Batista's conclusion that class skew by itself does not hinder learning algorithm performance, but rather other factors such as class overlap [4]. This result is intriguing because the hardness heuristics that measure class skew correlate with instance hardness and because class skew has been observed to cause misclassifications. The next section examines how class skew affects instance hardness. After the examination of class skew, we examine the heuristics at the instance level and further show that class overlap causes instance hardness and that class skew only exacerbates the problems of class overlap.

8.1 Effects of Class Skew

Class skew exacerbates existing contributors to instance hardness (namely class overlap). In Figure 8.1 the classification boundaries for the iris data set are projected onto the petal width (x-axis) and petal length (y-axis) attributes for the C4.5, MLP, NB, RIPPER, and SVM learning algorithms. Each color represents a class from the iris data set: red represents Iris-setosa, green represents Iris-versicolor, and blue represents Iris-virginica. Each pixel is assigned an RGB value based on the likelihood for the class. The likelihood is calculated using the values of the petal length attributes and petal width attributes and random values are assigned to the other attributes. Therefore, if a pixel has a likelihood of 1 for the Iris-versicolor class the RGB value for that pixel will be (0,1,0). If the likelihood for a class is not one, then the RGB values reflect that. For example if a pixel is on the border between the Iris-versicolor and Iris-virginica boundary the RGB value could be (0,0.65,0.35). Thus the transitional colors between red, green, and blue represent areas of uncertainty for the learning algorithm. For more details on this visualization technique see [12].

The first row shows the classification boundaries of the complete iris data set. The second row shows the iris data set with the majority of instances from the Iris-setosa (red) class removed; keeping 6 of the original 50 instances of Iris-setosa. The third row shows the iris data set with the majority of instances from the Iris-virginica (blue) class removed; keeping 16 of the original 50 instances. We preserved the instances with class overlap to maintain the class border. Note that the colors in the graphs for the SVM are between red, green, and blue implying that an instance does not have a likelihood of 1 for any class. This is because of the way SVMs handle multi-class output. For multi-class output, the SVM learning algorithm creates multiple SVMs; one SVM for every pair of classes. For the iris data set, three SVMs are used and maximum likelihood for an instance would be 2/3.

The Iris-setosa class does not overlap with the other two classes. Removing the majority of the instances belonging to the Iris-setosa class (second row in Figure 8.1) does not affect the classification boundary between the other two classes except for the RIPPER

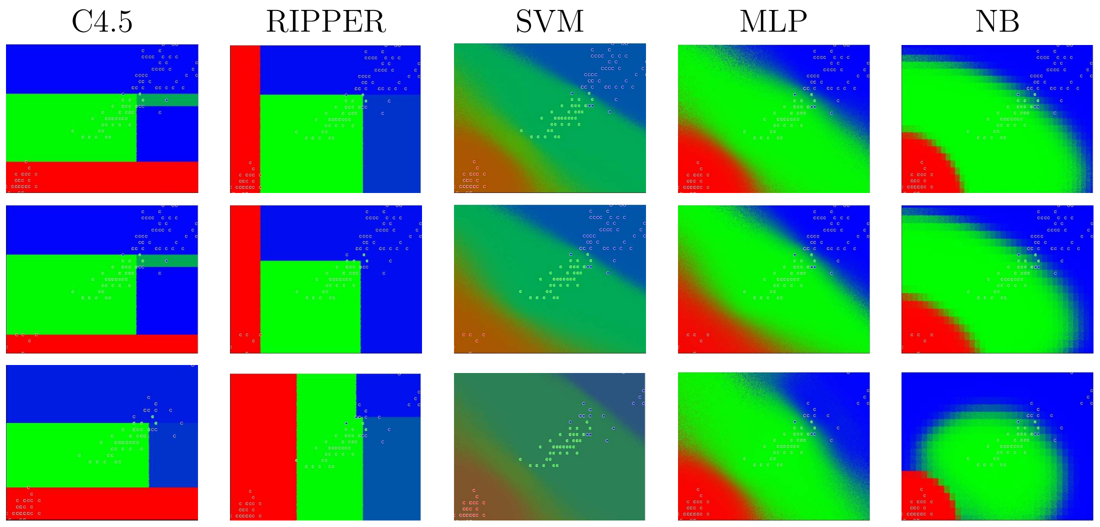
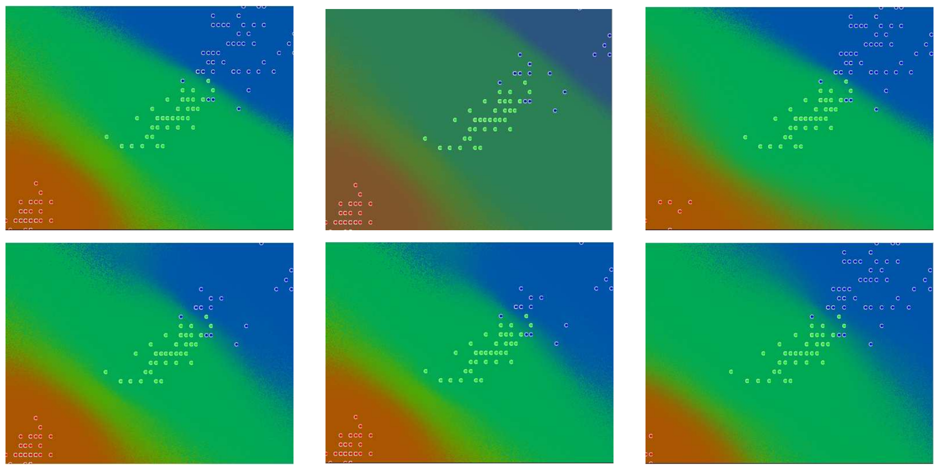


Figure 8.1: Visualization of the classification boundary for C4.5, MLP, NB, RIPPER, and SVM on the iris data set, projected onto the petal width (x-axis) and petal length (y-axis) attributes. The color represents the classes: red for Iris-setosa, green for Iris-versicolor, and blue for Iris-virginica. The colors in between red, green and blue represent uncertainty. The graphs in the first row were generated using all of the instances in the iris data set. The second row was generated by removing the majority of the instances from the Iris-setosa class. The third row was generated by removing the majority of the instances from the Iris-virginica class. Instances were removed from the center of the class to preserve the boundaries.

learning algorithm. RIPPER generates rules to define the classification boundary based on the instances that belong to the Iris-virginica class when all of the instances are present. With class skew, RIPPER generates the classification boundary based on the instances that belong to the Iris-versicolor class. Overall, the classification of the instances in the Iris-versicolor and Iris-virginica classes are unaffected when the Iris-setosa class is skewed.

The Iris-versicolor and Iris-virginica classes overlap. Removing the majority of the instances that belong to the Iris-virginica class (third row in Figure 8.1) does affect the classification boundary between the two classes. This is most notable with the SVM learning algorithm. With the skewed Iris-virginica class, the classification boundary shifts towards the Iris-virginica class (upper right corner). There is also notable changes in the classification boundary for the C4.5, RIPPER, and NB learning algorithms. The changes for the most part only affect the areas of the task space where there are no instances. The change in class skew affects those instances which overlap with the other class. If the instance belongs to the minority class then the instance hardness increases. If the instance belongs to the majority class then the instance hardness decreases.

The behavior of the SVM with a skewed overlapping class is interesting. The SVM learning algorithm is an optimization technique that maximizes the margin between two classes. One would expect the SVM learning algorithm to produce the same classification border for the original data set and for the data set with the center points removed from the Iris-vrginica class because the same set of support vectors would be expected in both cases. To handle noise well, a complexity parameter is provided as a means to trade-off between the complexity of the model and the proportion of overlapping instances. The parameter essentially penalizes overlapping instances. If the parameter is too low, then overlapping instances are not penalized enough resulting in underfit. This was the case for this example. Figure 8.2 shows the results after increasing the complexity parameter. The classification boundary between the Iris-versicolor and Iris-virginica classes is unchanged, as would be



All instances | Iris-virginica skewed | Iris-setosa skewed

Figure 8.2: Visualization of the classification boundary for SVM on the iris data set. The first row represents SVM with a low complexity parameter (c=1.0). The second row represents SVM with a high complexity parameter (c=200.0).

expected. This shows how sensitive SVMs are to parameter settings, especially if one class overlaps with another and has intra class separation.

8.2 Instance Types

Figure 8.3 shows the percentage of instances identified by the heuristics as outliers (first column), border points (second column), and class skew (third column) according to instance hardness. The percentage of instances identified as outliers increases with instance hardness. Less than 1% of the instances with an instance hardness 0 are identified as outliers while 75% of the instances with an instance hardness 1 are identified as outliers. The sharp increase in outliers as instance hardness increases shows that outliers are a main contributor to instance hardness and can be expected to be misclassified. 8% of the considered instances are identified as outliers.

The percentage of instances identified as border points increases very quickly with instances hardness up to instance hardness 0.33 and then stays relatively stable up to instance hardness 0.67. The percentage of instances identified as border points after instance hardness 0.67 decreases with instance hardness. Border points are more difficult to correctly classify

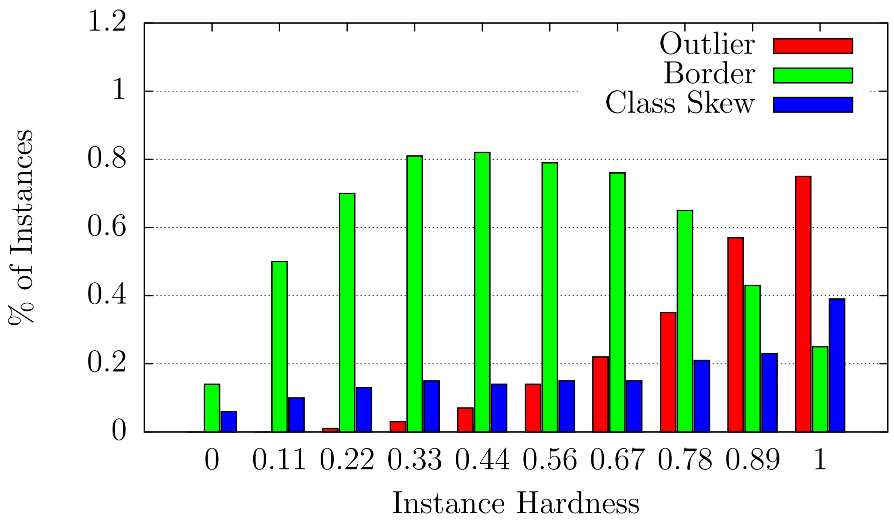


Figure 8.3: Instance type (outliers, border points, and class skew) according to instance hardness.

than the instances that do not have class overlap as learning algorithms are uncertain about which class the instances belong to. This is opposed to outliers where a learning algorithm will be confident, but wrong, about the classification of an instance. The high percentage of instances identified as border points for instances hardness values between 0.33 and 0.78 suggests that the learning algorithms are uncertain about the classes of the instances. 37% of all of the instances are identified as border points.

The percentage of instances belonging to a minority class according to instance hardness is fairly stable up to instance hardness 0.67. After instance hardness 0.67, the percentage of instances belonging to a minority class begins to increase. Class skew appears to be indicative of instance hardness. However, examining the instances that belong to a minority class, 40% of the instances that belong to a minority class also are outliers or border points. Figure 8.4 shows the percentage of instances that belong to a minority class and are also identified as outliers and border points according to instance hardness. Starting at instance hardness 0.67 the percentage of instances begins to increase with a significant increase at instance hardness 1. This indicates that outliers and border points are the hardest to correctly classify when they also belong to a minority class, reinforcing the point that class

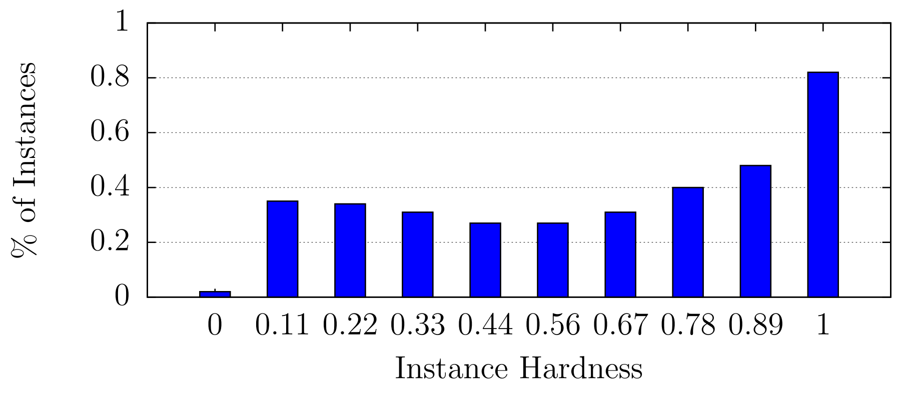


Figure 8.4: Percentage of instances identified as outliers or border points that belong to a minority class.

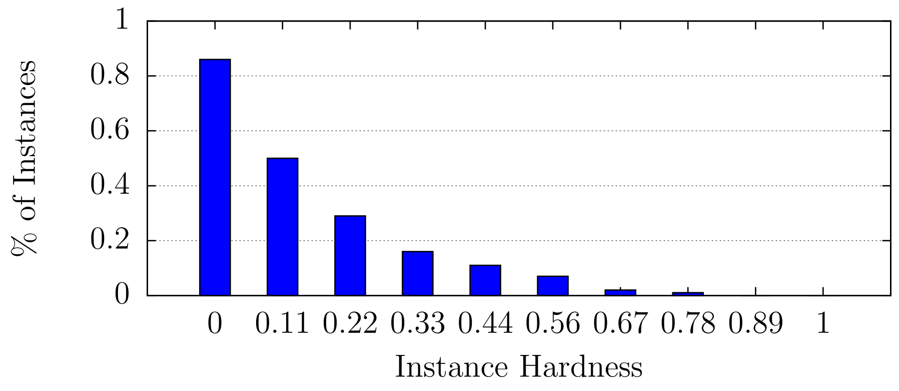


Figure 8.5: Percentage of instances not identified as outliers or border points.

skew exacerbates the problem of class overlap. 11% of all of the instances belong to minority class.

Figure 8.5 shows the percentage of instances not identified as outliers or border points. 86% of the instances with instance hardness 0 are not identified as outliers or border points. The remaining 14% are primarily border points. Only 7 instances with an instance hardness 0 are identified as outliers. At instance hardness 1 and 0.89, all of the instances are identified as outliers or border points, exhibiting some degree of class overlap. 55% of all instances are not identified as outliers or border points. This is similar to the 53% of the instances that that have an instance hardness 0.

For all instances with an instance hardness greater than 0.5, 39% of the instances are outliers and 59% of the instances are border points. Only 2% do not exhibit some degree of class overlap. This is in contrast to the instances with an instance hardness less than 0.5 for which 0.7% of the instances are identified as outliers, 32% as border points, and 67% do not exhibit class overlap. Class overlap plays a significant role in misclassifications.

22% of the instances with an instance hardness greater than 0.5 are identified as belonging to a minority class whereas only 8% of the instances with an instance hardness less than 0.5 belong to a minority class. Of the instances with an instance hardness greater than 0.5 that belong to a minority class, 99% are also identified as outliers or border points. In fact, 22% of all the instances with an instance hardness greater than 0.5 belong to a minority class and are also outliers or border points. Only 61% of the instances with an instance hardness less than 0.5 that belong to a minority class are also outliers or border points. About 5% of the instances with an instance hardness less than 0.5 belong to a minority class and are outliers or border points. This reiterates again the point that class overlap is exacerbated by class skew.

Table 8.1 shows the percentage of instances that were misclassified (second column) and correctly classified (third column) according to instance type. The percentages were calculated by averaging over all 9 learning algorithms. Note that the sets of instances overlap and that the percentages do not sum to one. 91% of the instances that were misclassified have class overlap and 19.8% are outliers. For the instances that were correctly classified 32.3% have class overlap and only 0.5% are outliers.

Table 8.2 shows the percentage of each instance type that was misclassified. The two highest percentages are for outliers. Note that for outliers and border points, the percentage of instances misclassified increases when the instance also belongs to a minority class. Also, only 38.3% of the border points are misclassified although the border points account for 71.3% of all misclassified instances. The learning algorithms vary on the classification of the border points.

Instance Type	Misclassified	Correctly Classified
Outliers	19.8	0.5
Border Points	71.3	31.7
Minority	21.0	8.0
Min and Out	6.6	0.1
Min and Bord	13.9	0.5
No Overlap	8.9	67.8

Table 8.1: Percentage of misclassified and correctly classified instances identified by the heuristics for instance type. Min and Out refer to instances that belong to a minority class and are also outliers. Min and Bord refers to instances that belong to a minority class and are also border points.

Outliers	Border Points	Minority	Min and Out	Min and Border	No Overlap
91.6	38.3	41.9	94.4	44.5	3.5

Table 8.2: Percentage of instances that were misclassified according to instance type.

Overall, class overlap is the principal contributor to instance hardness. The issue of class overlap is further magnified if there is class skew. Class skew in and of itself does not affect instance hardness. Thus, current learning algorithms can potentially be improved by focusing on the border points and increasing class separation.

Chapter 9

Data Set-level Analysis

As mentioned earlier, much research has been done to try to determine which learning algorithm should be used based on some data set features. Here, we are interested in how these features predict data set hardness. The following features from [6] were used:

- Number of Instances
- Number of Attributes
- Attribute Type (proportion of symbolic attributes)
- Proportion of Attributes with Outliers (an attribute has outliers if the ratio of the variances of the mean value and the α-trimmed mean is smaller than 0.7 setting $\alpha = 0.05$)
- Entropy of Classes (this is used as a measure of class skew using the equation:

  $$entropy = -\sum_{i=1}^{n} p(c_i) log_n p(c_i)$$

  where n is the number of classes and $p(c_i)$ is the probability of a class i (i.e., the number of instances belonging to class i divided by the total number of instances).

This list of features was augmented for this research to include the DN, CL, CLD, DS, DCP, MV, and CB values of the instances in the data set, the ratio of instances to attributes, the number of classes, and the percentage of outliers, border points, and instances belonging to a minority class as identified by our heuristics.

Table 9.1 shows the average data set hardness for the data sets that have certain data set features. The first row, in bold, is the average data set hardness for all data sets and is used as a baseline. The following rows are ordered by increasing average data set hardness values. The features in bold and italics are the hardness heuristics that measure class skew

Feature	DS hardness	Feature	DS hardness
All data sets	**0.202**	Ratio (inst:attr) Bottom 10	0.251
Entropy > 0.9	0.171	Ratio (inst:attr) Top 10	0.270
Minority Top 10	***0.179***	***Ave MV Top 10***	***0.313***
Ave CB Bottom 10	***0.181***	# Classes Top 10	0.329
All Real	0.191	**Border Top 10**	**0.341**
All Nominal	0.193	**Ave DS Bottom 10**	**0.363**
Has Outliers	0.204	**Ave DCP Bottom 10**	**0.414**
# Attributes Top 10	0.214	**No Overlap Bottom 10**	**0.440**
Mixed Attr	0.219	**Ave CL Bottom 10**	**0.454**
# Instances Top 10	0.226	**Ave CLD Bottom 10**	**0.454**
Entropy <= 0.9	0.236	**Outlier Top 10**	**0.462**
Mixed Attr w/ Out	0.241	**Ave DN TOP 10**	**0.473**

Table 9.1: Average data set hardness based on data set features.

and the features in bold are the hardness heuristics that measure class overlap. The features in plain text are the features used by [6]. The heuristics that measure class skew are not good indicators for data set hardness whereas the heuristics for class overlap are indicative of data set hardness. The results for each attribute are summarized below. The values in parenthesis refer to a feature in Table 9.1

- **Attribute Type.** The data sets with mixed attributes (Mixed Attr) have a higher data set hardness than the data sets that have all real (All Real) or all nominal (All Nominal) attributes. This is expected as many learning algorithms are designed to handle one case better than the other. The change in data set hardness for either case is minimal though.

- **Number of Instances and Attributes.** The number of instances or the number attributes do not appear to have a large impact on data set hardness. The data set with the most instances (# Instances Top 10) and the data sets with the most attributes (# Attributes Top 10) increase data set hardness. However, the data sets with the fewest instances and attributes also increase the data set hardness.

  The ratio of instances to attributes (Ratio) appears to affect data set hardness more than the number instances and attributes in isolation. The largest values (representing a large number of instances to an attribute) causes an increase in data set hardness. It would seem that having more instances per attribute would make the data set easier to classify. Examining the data sets with the ten largest ratios, six of the ten have all nominal attributes and only two have all real valued attributes. One of the data sets is the Titanic data set. This data set has 2201 instances and only 3 attributes; 1 has 4 nominal values and the other 2 have 2 nominal values. In such a data set saturation occurs as there are only 16 possible attribute combinations. This is not the case for all of the data sets. Other data sets, such as the chess and yeast data sets, approach saturation as well as have other complicating factors such as large number of classes (discussed below).

- **Class Skew and Number of Classes.** The data sets that have class skew measured using entropy show that class skew does affect data set hardness, although not drastically. Lower entropy values (Entropy <= 0.9) cause a small increase in data set hardness from the baseline (0.202 to 0.236) while the data sets that have more balanced classes (Entropy >0.9) achieve a decrease of about the same magnitude (0.202 to 0.171). Class skew is further shown to affect data set hardness with an increase in data set hardness to 0.313 for the 10 data sets that have the highest average minority values for their instances (Ave Minority Top 10). The minority value is a good indicator of data set hardness. The average class balance value per data set also attempts to measure class skew but it is not indicative of data set hardness and results in lowest data set hardness for the heuristics presented in this paper (Ave CB Bottom 10). Using the Top 10 values for CB results in an average data set hardness value of 0.147. Using our heuristic to identify minority classes, the data sets with top 10 highest percentage of instances belonging to a minority class have an average data set hardness of 0.179 (Minority Top 10). Class skew does not directly impact data set hardness.

  The data sets with a large number of classes also increase data set hardness (# Classes Top 10). This is not too surprising after examining the data sets that have the most classes. In 8 of the 10 data sets, they are the same as those that have the highest average minority values. There is a greater chance for class overlap by having more classes.

- **Border Points.** The data sets with the highest percentage of border points using our heuristic (Border Top 10) exhibit an increase in data set hardness to 0.341. This is about what is expected, a moderate increase in data set hardness, yet not as much of an increase as outliers.
- **Outliers.** When using the alpha-trimmed mean method used by [6] to identify outliers, outliers do not appear to impact data set hardness (Has Outliers). Data set hardness increases from 0.202 to 0.241 for data sets with mixed attributes that have outliers using the alpha-trimmed mean method (Mixed Attr w/ Out). This increase begins to show that outliers impact instance hardness. When using our heuristic to identify outliers, the average data set hardness for the data sets with the 10 highest percentages of outliers increases to 0.462 (Outliers Top 10).
- **Class Overlap.** The bottom average DS, DCP, CL and CLD values per data set have the greatest increase in data set hardness. These values all measure class overlap in a data set. Those data sets with the most amount of class overlap (as measured by our heuristics) are the ones with which it is the most difficult to achieve high classification accuracy. Looking at the 10 data sets with the lowest percentage of instances that do not overlap, the data set hardness increases to 0.440 (No Overlap Bottom 10). This further shows how class overlap directly impacts hardness.

Overall, class overlap as measured with average DN is the best predictor of data set hardness. However, the top 10 values of DN do not include all the 10 data sets with the 10 largest data set hardness values. If the data sets were ranked according to data set hardness, the top 10 values of DN includes data sets 1-7, 9, 11, and 13.

Applying linear regression to estimate data set hardness based on data set features also shows what contributes to data set hardness. The result is as follows.

$$\begin{aligned} data\ set\ Hardness =& 0.5547 * DN \\ & -0.3109 * CL \\ & -0.2064 * DCP \\ & +0.0442 * outliers \\ & +0.9613 \end{aligned}$$

with a correlation coefficient of 0.9526. This shows that the most highly weighted features are DN, CL and DCP. Interestingly the alpha-trimmed mean definition for an outlier is also included although it is minimally weighted. This further supports the claim that class overlap is the principal reason for instance and data set hardness. As with instance hardness, class skew is not included in the linear regression equation.

Chapter 10

Learning Algorithms-based Analysis

In this section, the performance of the learning algorithms is examined in the context of instance hardness and of instance types.

Figure 10.1 shows the accuracy for each learning algorithm at each instance hardness value. The x-axis represents the learning algorithms and the y-axis shows the accuracy for each learning algorithm. The perceptron learning algorithm does fairly well up to instance hardness 0.5, after which it is among the learning algorithms that achieve the lowest classification accuracy. The perceptron is a linear discriminator and suggests that the easier instances are linearly separable. NB is the opposite, being amongst the algorithms that have lower classification accuracy for instance hardness less than 0.5. For instance hardness greater than 0.5, NB then achieves a relatively higher classification accuracy. C4.5 achieves the highest accuracy for all instance hardness values greater than 0.5. Despite doing better on the harder instances, C4.5 is only average for instance hardnesses less than or equal to 0.33.

There is no single algorithm that achieves better classification accuracy based on instance hardness. As a generalization, instances with lower instance hardness values can be correctly classified with a simpler decision surface while those with high instance hardness values require a more complex decision surface. This is manifest through the learning algorithms that have a more complex decision process achieving a higher accuracy than the simpler learning algorithms on instances with higher instance hardness values. This also implies that some of the learning algorithms that have a higher classification accuracy on instances with higher instance hardness values overfit the harder instances. In this case the harder instances are correctly classified at the cost of generalization.

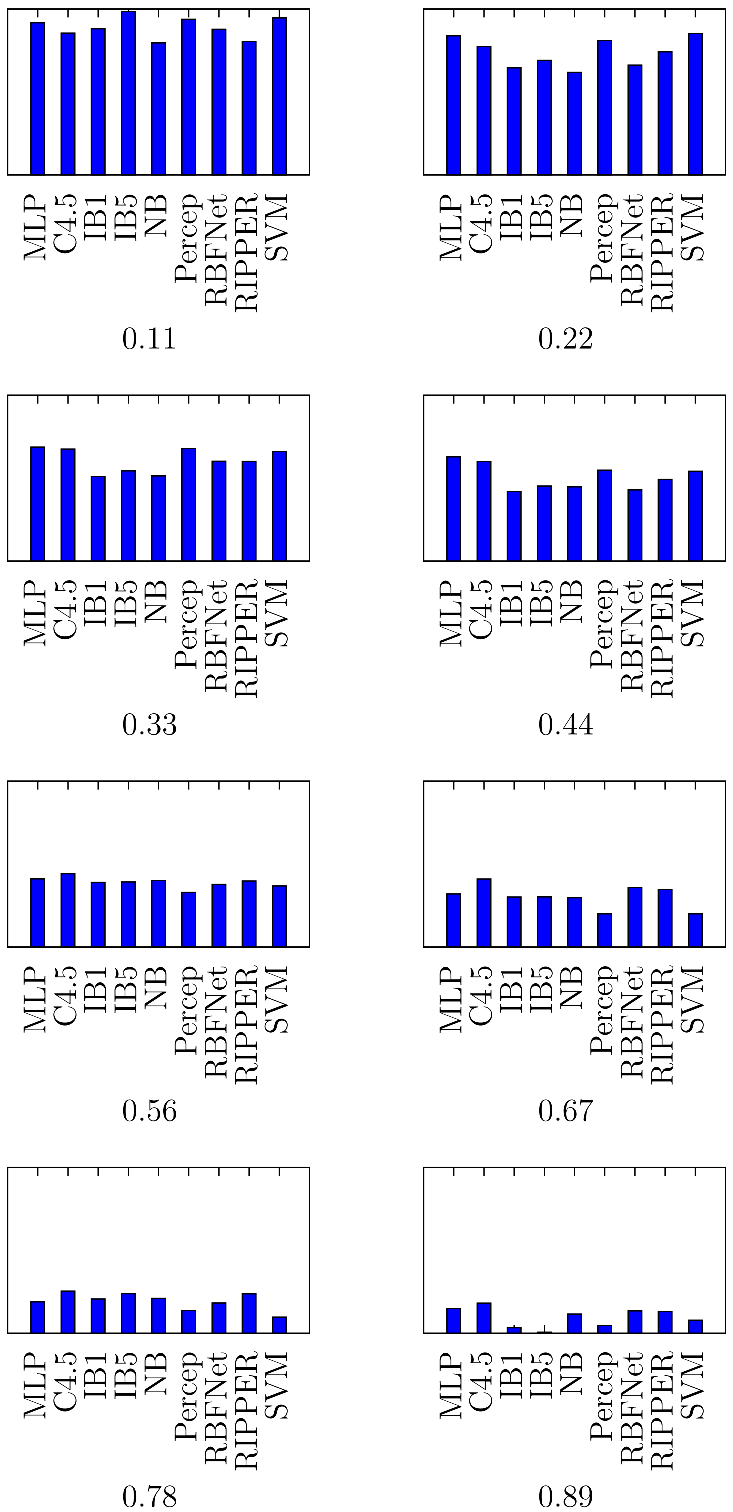


Figure 10.1: Learning algorithm accuracy over all instance hardness values

	DN	DS	DCP	CL	CDL	MV	CB
C4.5	0.180	0.375	0.923	0.789	0.644	0.879	0.114
IB1	0.146	0.381	0.909	0.780	0.655	0.888	0.114
IB5	0.177	**0.367**	**0.890**	**0.762**	**0.607**	**0.874**	**0.111**
MLP	**0.182**	0.372	0.899	0.789	0.646	0.876	0.113
NB	0.161	0.397	0.912	0.857	0.756	0.899	0.126
Perceptron	0.148	0.395	0.922	0.847	0.728	0.898	0.123
RBFNet	0.166	0.388	0.910	0.827	0.703	0.892	0.123
RIPPER	0.162	0.379	0.917	0.812	0.676	0.892	0.119
SVM	0.161	0.383	0.911	0.819	0.690	0.891	0.121

Table 10.1: Average hardness heuristic values for correctly classified instances for each learning algorithm.

Learning Algorithm	All	Class Overlap	Class Skew
IB5	1.29	1.4	1
MLP	2	2	2
C4.5	3.57	3.8	3
IB1	4.71	5	4
RIPPER	5.29	5.2	5.5
SVM	5.71	5.8	5.5
RBFNet	6.29	5.8	7.5
Perceptron	7.86	8	7.5
NB	8.29	8	9

Table 10.2: Learning algorithm ranking for the hardness heuristics.

Table 10.1 shows the average heuristic value for the instances that were correctly classified by a learning algorithm. The average values are similar between learning algorithms signifying that the learning algorithms perform similarly. The values in bold are significant because they exemplify more class overlap or class skew.

Ranking the algorithms based on their average heuristic value will show which learning algorithms are better able to handle class overlap and/or class skew. A rank of 1 means that a learning algorithm is most robust to that heuristic. The average ranking of each heuristic for the learning algorithms is shown in Table 10.2. The second column "All" refers to all of the heuristics, the third column "Class Overlap" refers to the heuristics that measure class overlap (DN, CL, CLD, DS, and DCP) and the fourth column "Class Skew" refers to the heuristics that measure class skew (MV and CB).

	Outlier	Border Point	Minority	No Overlap
C4.5	**0.147**	0.691	**0.628**	0.979
IB1	0.054	0.531	0.544	0.989
IB5	0.025	**0.709**	0.622	**0.998**
MLP	0.121	**0.707**	**0.628**	0.976
NB	0.059	0.558	0.524	0.896
Perceptron	0.075	0.535	0.567	0.955
RBFNet	0.093	0.593	0.547	0.941
RIPPER	0.100	0.616	0.590	0.977
SVM	0.082	0.611	0.578	0.974

Table 10.3: Learning algorithm accuracy for instance types.

IB5 is ranked 1 for all of the heuristics except DN (IB5 is ranked 3 for DN). The other learning algorithms are also consistently ranked the same for all of the heuristics. According to the rankings, class overlap and class skew are handled similarly by the learning algorithms. IB5 and MLP are the least affected by class overlap and class skew. NB is the most affected by class overlap and class skew. Class skew more heavily influences NB than the other learning algorithms. This is expected since NB weights all instances with a prior probability of the frequency of the class that an instance belongs to.

It is interesting that IB5 does so well especially since IB5 does not transform the task space and because the performance of the instance-based approaches has been shown to degrade as the number of dimensions increases. The classification accuracy for IB5 is within 2% of the other learning algorithms on 8 of the 10 data sets with the most features. The number of features in the set of data sets ranges from 40 to 6598.

Table 10.3 shows the accuracy of the learning algorithms over the instances identified as outliers, border points or belonging to a minority class using the heuristics for instance type. No Overlap refers to instances not identified as outliers or border points.

Overall, the learning algorithms achieved the lowest classification accuracy on the instances identified as outliers. This is expected as outliers should be misclassified and learning algorithms should be robust to outliers. C4.5 achieved the highest classification accuracy on the outliers which may be because of overfit. IB5 has the lowest accuracy on instances identified as outliers. This is desired as outliers should be misclassified. The maximum classification accuracy is only 14.7%. Although this is quite low, it shows that learning algorithms could do better at minimizing the effect of outliers. On the other hand, it could be that C4.5 and MLP are able to correctly classify instances that appear to be outliers by using more sophisticated techniques.

C4.5, IB5 and MLP clearly handle border points better than the other learning algorithms achieving an average accuracy 8% better than the other learning algorithms. It is interesting that IB5 is included in this list which does not transform the task space. On the other spectrum, there are learning algorithms that do transform the task space that do not perform well on border points. The maximum accuracy on border points is only 70.9% suggesting that improvement can still be made.

The learning algorithms have similar classification accuracy for instances that belong to a minority class. Again, C4.5, IB5 and MLP achieve the highest accuracy for instances that belong to a minority classy. The maximum classification accuracy is 62.8% (as compared to 14.7% for outliers), reinforcing the notion that class skew does contribute to instance and data set hardness.

Lastly, we note the "No Overlap" instances that were not identified as outliers or border points. For these instances, the classification accuracy is above 89.6% for all of the learning algorithms and the average classification accuracy is 96.5%. For all of the instances that are expected to be correctly classified, IB1 and IB5 have the highest classification accuracy (99.8% and 98.9% respectively). There are instances that should be correctly classified yet they are misclassified by some learning algorithms. Ideally, a learning algorithm should achieve 100% classification accuracy for the "No Overlap" instances. A learning algorithm may not achieve 100% classification accuracy on the instances with no overlap because the learning algorithm is affected by outliers, border points and/or class skew. We have discussed how the classification boundary is affected by class skew. This alteration of the border could lead non-overlapping instances to be misclassified. Also, if there is significant class skew, such as in the Titanic data set, then the instances with no overlap would still be misclassified. Additionally, the classification boundary is affected by the border points and outliers that effectively pull the classification boundary. Each learning algorithm will vary on how overlapping points are handled. Some learning algorithms, such as the MLP even vary based on presentation order which affects where the classification boundary is created.

Chapter 11

Conclusions and Future Work

We empirically analyzed to what extent instances are hard to correctly classify. Our analysis was extensive, examining 57 data sets, 178109 instances, and 9 learning algorithms. We generated over 5130 models and analyzed their results. We found that there is a set of instances that all learning algorithm misclassify comprising 5% of all the instances and that 15% of the instances used in this study are misclassified by at least half of the learning algorithms. We presented a set of hardness heuristics to identify instances that are hard to correctly classify. Using these heuristics, the results indicate that class overlap most directly affects instance hardness. This is expected as the outliers and border points, which are hard to correctly classify, have this characteristic of class overlap. For all misclassified instances, 19.8% are identified as outliers, 71.3% are identified as border points, and 21% belong to a minority class. We also find that 91.6% of all outliers are misclassified and 38.3% of all border points are misclassified whereas only 3.5% of the instances without class overlap are misclassified. Class skew also affects instance and data set hardness. By itself, class skew does not make an instance hard to correctly classify unless it is an issue of data underrepresentation. In the presence of class overlap, class skew exacerbates the difficulties of class overlap and affects the classification boundary.

The results indicate that IB5 is best able to correctly classify an instance in the presence of class overlap and class skew. This is surprising since IB5 does not transform the task space and because nearest neighbor algorithms are not supposed to perform well in high dimensions. MLP, which does transform the task space, also handles class overlap and class skew well. The similarity in the results for IB5 and MLP may suggest that the data sets used in this research can be learned comparably without transforming the task space. The similarity between IB5 and MLP may also suggest that the manner in which the task space is transformed does not result in a significant increase in accuracy. Finally, the data sets may not have a descriptive enough set of attributes.

Using the results presented in this study, it would be beneficial to examine other ways to transform the task space. The primary goal of learning algorithms should be to increase class separation. Knowing that class overlap causes misclassifications, some filtering algorithms could be used to remove instances that overlap with another class. The heuristics presented in this paper can be used to identify these instances.

The heuristics could also be used to ensemble learning algorithms together based on instance type (outlier, border point, no overlap, class skew). The learning algorithm that best handles certain instance types could be weighted more heavily in the classification of that instance.

Another potential use for the heuristics for instance types is to preprocess a data set. Using our hardness heuristics, hard instances could be removed before training. The removed instances would include instances identified as outliers and border points that are likely to be misclassified. By removing these instances, the learning algorithm could better determine the classification boundary and learning algorithms that are not robust to outliers and border points could be improved.

A comparison of our heuristics that identify instances as outliers or border points with previous work would be interesting. The heuristics in this work were developed on the notions of instance hardness rather than statistics, clustering techniques or other previously used techniques.

Since this is an empirical study, better generalizations might be made by analyzing more data sets and/or learning algorithms. Another future work will consider improved heuristics to better detect instances that should be hard to correctly classify. Our analysis of what contributes to instance hardness and data set hardness was limited to linear combinations. Future work for the contributions to instance hardness and data set hardness could be improved by using meta-learning and non-linear combinations of the heuristics for instance type. This could lead to further insight into what causes instance hardness.

Bibliography

[1] Naoki Abe, Bianca Zadrozny, and John Langford. Outlier detection by active learning. In *Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 504–509, New York, NY, USA, 2006. ACM.

[2] Rehan Akbani, Stephen Kwek, and Nathalie Japkowicz. Applying support vector machines to imbalanced datasets. In *Proceedings of the 15th European Conference on Machine Learning (ECML)*, pages 39–50, 2004.

[3] Arthur Asuncion and David J. Newman. UCI machine learning repository, 2007.

[4] Gustavo E. A. P. A. Batista, Ronaldo C. Prati, and Maria Carolina Monard. A study of the behavior of several methods for balancing machine learning training data. *SIGKDD Explorations Newsletter*, 6(1):20–29, 2004.

[5] Jonathan Baxter. A model of inductive bias learning. *Journal of Artificial Intelligence Research (JAIR)*, 12:149–198, 2000.

[6] Pavel B. Brazdil, Carlos Soares, and Joaquim Pinto Da Costa. Ranking learning algorithms: Using ibl and meta-learning on accuracy and time results. *Machine Learning*, 50(3):251–277, 2003.

[7] Markus M. Breunig, Hans-Peter Kriegel, Raymond T. Ng, and Jörg Sander. Lof: identifying density-based local outliers. *SIGMOD Record*, 29(2):93–104, June 2000.

[8] Henry Brighton and Chris Mellish. Advances in instance selection for instance-based learning algorithms. *Data Mining and Knowledge Discovery*, 6(2):153–172, 2002.

[9] Christopher J. C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[10] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research (JAIR)*, 16:321–357, 2002.

[11] Seyda Ertekin, Jian Huang, Lon Bottou, and C. Lee Giles. Learning on the border: active learning in imbalanced data classification. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management*, pages 127–136, New York, NY, USA, 2007. ACM.

[12] Eibe Frank and Mark Hall. Visualizing class probability estimators. In *In Lecture Notes in Artificial Intelligence 2838*, pages 168–179. Springer, 2003.

[13] Shohei Hido and Hisashi Kashima. Roughly balanced bagging for imbalanced data. In *Proceedings of the SIAM International Conference on Data Mining*, pages 143–152. SIAM, 2008.

[14] Tin K. Ho and Mitra Basu. Complexity measures of supervised classification problems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(3):289–300, 2002.

[15] Victoria Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[16] Robert C. Holte, Liane E. Acker, and Bruce W. Porter. Concept learning and the problem of small disjuncts. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 813–818. Morgan Kaufmann, 1989.

[17] Nathalie Japkowicz. Class imbalances: Are we focusing on the right issue? In *Proceedings of the ICML Workshop on Learning from Imbalanced Datasets II.*, pages 27–39, 2003.

[18] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent Data Analysis*, 6(5):429–449, 2002.

[19] Alexandros Kalousis and Melanie Hilario. Model selection via meta-learning: a comparative study. *International Journal of Artificial Intelligence Tools*, 10(4):525–554, 2001.

[20] Taghi M. Khoshgoftaar, Naeem Seliya, and Kehan Gao. Rule-based noise detection for software measurement data. In *Proceedings of the IEEE International Conference on Information Reuse and Integration*, pages 302–307. IEEE Systems, Man, and Cybernetics Society, 2004.

[21] Jeremy M. Kubica and Andrew Moore. Probabilistic noise identification and data cleaning. In *The Third IEEE International Conference on Data Mining*, pages 131–138. IEEE Computer Society, November 2003.

[22] Tjen-Sien Lim, Wei-Yin Loh, and Yu-Shan Shih. A comparison of prediction accuracy, complexity, and training time of thirty-three old and new classification algorithms. *Machine Learning*, 40(3):203–228, 2000.

[23] Xiao-Dong Liu, Churn-Yi Shi, and Xue-Doa Gu. A boosting method to detect noisy data. In *Proceedings of 2005 International Conference on Machine Learning and Cybernetics*, volume 4, pages 2015–2020, 2005.

[24] Jonathan I. Maletic and Andrian Marcus. Data cleansing: Beyond integrity analysis. In *Fifth Conference on Information Quality*, pages 200–209, 2000.

[25] Navneet Panda, Edward Y. Chang, and Gang Wu. Concept boundary detection for speeding up svms. In *Proceedings of the 23rd international conference on Machine learning*, pages 681–688, New York, NY, USA, 2006. ACM.

[26] Adam H. Peterson. COD: Measuring the similarity of classifiers. Master's thesis, Brigham Young University, January 2005.

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

[28] Bernhard Pfahringer, Hilan Bensusan, and Christophe G. Giraud-Carrier. Meta-learning by landmarking various learning algorithms. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 743–750, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[29] John C. Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods: support vector learning*, pages 185–208, Cambridge, MA, USA, 1999. MIT Press.

[30] J. Ross Quinlan. Improved estimates for the accuracy of small disjuncts. *Machine Learning*, 6(1):93–98, 1991.

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

[32] Ljupco Todorovski, Pavel B. Brazdil, and Carlos Soares. Report on the experiments with feature selection in meta-level learning. In *Proceedings of the ECML Practice of Knowledge Discovery in Databases Workshop on Data Mining, Decision Support, Meta-Learning and ILP: Forum for Practical Problem Presentation and Prospective Solutions*, pages 27–39, 2000.

[33] Christiaan van der Walt and Etienne Barnard. Data characteristics that determine classifier performance. In *Proceedings of the Sixteenth Annual Symposium of the Pattern Recognition Association of South Africa*, pages 160–165, Parys, South Africa, 2006.

[34] Jason van Hulse, Taghi M. Khoshgoftaar, and Amri Napolitano. Experimental perspectives on learning from imbalanced data. In *Proceedings of the 24th international conference on Machine learning*, pages 935–942, New York, NY, USA, 2007. ACM.

[35] Gary M. Weiss. Learning with rare cases and small disjuncts. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 558–565. Morgan Kaufmann, 1995.

[36] Gary M. Weiss. Mining with rarity: a unifying framework. *SIGKDD Explorations*, 6(1):7–19, 2004.

[37] Gary M. Weiss and Haym Hirsh. The problem with noise and small disjuncts. In *Proceedings of the Fifteenth International Conference on Machine Learning*, page 574, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[38] Gary M. Weiss and Haym Hirsh. A quantitative study of small disjuncts. In *Proceedings of the Seventeenth National Conference on Artificial Intelligence, 665-670. Menlo Park*, pages 665–670. AAAI Press, 2000.

[39] D. Randall Wilson and Tony R. Martinez. Reduction techniques for instance-based learning algorithms. *Machine Learning*, 38(3):257–286, 2000.

[40] Dennis L. Wilson. Asymptotic properties of nearest neighbor rules using edited data. *IEEE Transactions on Systems, Man, and Cybernetics*, (2-3):408–421, 1972.

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

[42] Jianping Zhang. Selecting typical instances in instance-based learning. In *Proceedings of the ninth international workshop on Machine learning*, pages 470–479, San Francisco, CA, USA, 1992. Morgan Kaufmann Publishers Inc.