# Classification by CUT: Clearance Under Threshold

Ryan McBride
School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, British Columbia
Email: rom2@sfu.ca

Ke Wang[1]
School of Computing Science
Simon Fraser University
8888 University Drive
Burnaby, British Columbia
Email: wangk@cs.sfu.ca

Wenyuan Li[2]
British Columbia Hydro
6911, Southpoint Dr
Burnaby, British Columbia
Email: Wenyuan.li@ieee.org

***Abstract*—Identifying bad objects hidden amidst many good objects is important for public safety and decision-making. These problems are complicated in that the cost of leaving a bad object unidentified may not be specified easily, making it difficult to apply existing cost-sensitive classification that depends on knowing a cost matrix or cost distribution. A compelling case for this "illusive cost" issue is presented in our project of identifying contaminated transformers with an industrial partner. To address this problem, we present an alternative formulation of cost-sensitive classification, Clearance Under Threshold (CUT) Classification. Given a training set, CUT classification is to partition the attribute space such that a partition is *cleared* if the probability of a future object in this partition being bad is less than a user-specified threshold. The goal is to clear many low-risk objects so that users can more effectively target high-risk objects. We present a solution to this problem and evaluate it on a case study for clearing contaminated transformers and on public benchmarks from UC Irvine's Machine Learning Repository. According to the experiments, our algorithms performed far better than the baselines derived from previous classification approaches.**

## I. Introduction

### A. Motivation

This work was motivated by a real industrial problem at BC Hydro, an electric power company in the province of British Columbia in Canada. Many pre-1986 manufactured high voltage transformers contain insulating polychlorinated biphenyls (PCBs for short) based oil mixtures and are still in service today. In later years, it became known that PCBs can cause harmful health effects and can build-up in the environment [1] [2]. For these reasons, Stockholm Convention parties, including Canada, intend to phase out the use of PCBs in equipment by 2025 [1]. However, immediately replacing all transformers is impractical because each transformer costs about 1-5 million dollars and the majority of transformers are non-PCB.

It is therefore important to accurately identify, remove, then replace PCB transformers. Unfortunately, identifying these contaminated transformers is not trivial: even transformers sold as PCB-free can potentially be contaminated because, before the harmful effects of PCBs were known, non-PCB oils could be inadvertently mixed with PCB oils. Also, many transformer bushings, the oil filled parts, are designed to be completely sealed with no drainage valves or other access facilities; only by expensively breaking the bushings, which incurs roughly 10% of the cost of a transformer, can we sample the PCB content of its oil mixture. Taking such samples from in-service transformers also results in power interruption and overloading other transformers, thus increasing risk in the power grid. A more realistic solution is to predict if a transformer is likely contaminated by building an analytical model using the known PCB status of already sampled transformers.

TABLE I: A Cost-Sensitive Problem's Cost Matrix.

| | | **Object Class j** | |
|---|---|---|---|
| | | Positive | Negative |
| **Predicted Class i** | Positive | $C_1$ | $C_2$ |
| | Negative | $C_3$ | $C_4$ |

An obvious approach to this problem would be to build a cost-sensitive classification model with a cost matrix like Table I. Let $C_2$ be the cost of a false positive error and $C_3$ be the cost of a false negative error then, typically, $C_1 = 0$, $C_4 = 0$, and $C_3 \gg C_2$. The goal of cost-sensitive classification is to build a model based on a training data set that makes decisions on future objects to minimize expected costs according to the cost matrix. The cost matrix can be object specific: for each object $x$, $C(i, j, x)$ is the cost of predicting class $i$ when the object $x$ is actually class $j$. If PCB contamination corresponds to the positive class and non-contamination corresponds to the negative class, a false negative error leaves a PCB contaminated transformer unidentified, with the cost $C_3$, while a false positive error needlessly tests an uncontaminated transformer, with the cost of $C_2$.

However, a closer study reveals that existing cost-sensitive methods are not applicable to PCB identification because the assumed cost information is not available. In particular, it is difficult to estimate $C_3$ (or $C(-, +, x)$), the cost of leaving a PCB contaminated transformer unidentified because the potential cost could vary from no damage to an environmental catastrophe. Even inexact costs, such as the worst case cost, are difficult to determine because the impact of leaking PCBs involves non-quantifiable costs, such as public panic, that cannot

[1]This work was done partially while the author was visiting Singapore Management University.

[2]This author is an adjunct professor with the School of Computing Science at Simon Fraser University, Canada, and a professor with the School of Electrical Engineering at Chongqing University, China.

be easily compared to the dollar amount of testing for PCBs. In fact, it is difficult to know when an unidentified contaminated transformer may leak and what impact the leak may have because the exact outcome depends on unpredictable factors such as future electrical demand, weather conditions, and the aging process of transformers. Even gathering this information from already removed PCB transformers, the training objects, is difficult since slightly different circumstances could adversely affect the PCB transformer's impact for the reasons discussed above. Ultimately, this complexity and uncertainty in defining the cost matrix prevent meaningful cost-sensitive solutions.

This "illusive" nature of cost is not unique to our PCB identification problem: in healthcare, quantifying the cost of failing to identify a diseased patient is very complicated. As discussed in a recent editorial by Vasant Dhar [3], setting the cost of a missed prediction of diabetes must represent possible health complications like "amputation or loss of vision" but how can any expert argue that lost eyesight costs $x$ dollars? Similarly, what is the cost $C_3$ of failing to identify a cancer patient, an unqualified applicant, a broken car brake, or contaminated food? $1,000 or $10,000,000? An over-estimate helps reduce such failures, but also expensively treats many non-cancer patients as cancer patients, qualified applicants as unqualified applicants, normal brakes as broken brakes, and usable food as contaminated food. For such problems, it is difficult to specify the cost $C_3$ or even a cost distribution (e.g. a minimum cost and a maximum cost), though it is clear that $C_3$ should be more costly than $C_2$.

### B. Our Approach

Compelled by our PCB identification problem, we propose *a general solution to cost-sensitive classification without a cost matrix*. Our solution was motivated by the following industrial insight: while it is difficult to specify an exact cost matrix, it is often possible for industries to specify the maximum allowed probability of a bad event and use this maximum risk measure to *clear* good objects. For example, in many applications, such as hazard disposal, power outage prediction, and quality assurance, regulatory guidelines exist and can be used to specify this maximum allowed probability. By clearing the objects that have no more than this probability of being a bad event, the user is able to prioritize resources, such as PCB tests, to the remaining high-risk objects. This maximum allowed probability of bad events for safely clearing an object, called the *clearance threshold*, can be more easily determined than other approaches because it captures risk as a probability, which we believe is a judicious way of factoring domain knowledge without over-committing to specific costs.

With this insight, we propose an alternative formulation for cost-sensitive classification problems, called *Clearance Under Threshold (CUT) Classification*. We assume that a labeled training set is available. For our purpose of clearing the majority class, we define the majority class as the positive class and the minority class as the negative class. In the PCB transformer case, for example, contaminated transformers are negative objects and non-PCB transformers are positive objects. Notice the difference from the cost-sensitive classification where the minority class is the positive class. We also assume that a clearance threshold $t$ is specified by the user, which is typically smaller than the proportion of the negative class in the training set since a cleared object is expected to have a smaller risk than the average risk in the training set. The CUT Classification seeks to partition the attribute space based on the training set to *clear* as many future objects as possible subject to the set clearance threshold. The clearance threshold ensures that each cleared object has a low-risk of being negative.

### C. Contributions

Our contributions can be summarized as follows:

- (Section III) We present the problem of Clearance Under Threshold Classification, or CUT Classification, as a classification problem defined by a user-specified clearance threshold. By switching from specifying the low level cost information to specifying the probability based clearance threshold, CUT Classification offers two benefits over traditional cost-sensitive classification: it is typically easier to obtain the clearance threshold than costs, and the clearance of an object conveys a more intuitive quality assurance through a maximum probability or risk in cleared groups; in contrast, the cost-based approach does not provide this risk guarantee.
- (Section IV) We present our solutions, CUT Classifiers, by addressing two issues. The first is formalizing the concept of risk on future objects; simply taking this risk probability to be the proportion of the negative class in a group of objects in the training set is unreliable because it ignores the statistical significance of the observation. The second issue is building a model from the training set to maximize the number of future objects cleared while satisfying the user-specified clearance threshold.
- (Section V) We validate our solutions on several public data sets and two PCB transformer identification sets. These results suggest that our methods better clear good objects while keeping bad objects under threshold in comparison to competitors from other classification problems.

## II. Related Work

Our work is most related to cost-sensitive classification, as reviewed in surveys such as [4] and [5]. The most common cost-sensitive solutions specify an exact cost matrix and minimize the aggregate cost of decisions by integrating cost directly into the algorithm. For example, Lomax and Vadera's 2013 survey describes many methods to guide a decision tree's separation of data with a cost matrix [6]. Other approaches may be "wrappers" that convert any cost-insensitive classifier into a cost-sensitive one. For example, MetaCost [7] first relabels the training set to minimize the expected cost in many standard classification models according to the given cost matrix then inputs the relabeled training set into a standard error-based classifier. Another approach is the *"Bayes minimum estimated cost"* relabeling strategy [4], that learns an error-based classifier from the training set as given, and then classifies an object $x$ with the label that minimizes the expected cost of a decision according to the cost matrix given and the probability estimates given by the classifier. For example, with two classes and with $C_1 = C_4 = 0$ the system would label an instance $x$ as a majority negative if the probability estimated by a classifier for $x$ to be in the minority positive class, $P(+ \mid x)$, is less than the maximum probability before the estimated cost of a false

negative ($C_3 \cdot P(+ \mid x)$) exceeds the estimated cost of a false positive ($C_2 \cdot P(- \mid x)$), the critical probability $p^*$ defined as:

$$p^* = \frac{C_2}{C_2 + C_3} \quad (1)$$

In all of these approaches, a cost matrix is necessary because the goal of the problem is to minimize the overall cost of the decisions on all objects. In contrast, our CUT Classification problem assumes no cost matrix and its objective is to clear as many objects as possible according to a given clearance threshold: these different objectives imply that threshold and cost problems require different algorithmic focuses.

Other cost based approaches assume partial or inexact cost information. For example, Zadrozny and Elkan develop a method to predict the donation amount of a customer [8], and therefore assume no cost matrix in Table I for future objects, but their method still requires the cost $C(i, j, x)$ for each training object $x$ to predict these future costs. As explained earlier, there are many problems where the training object costs may not be applicable to future objects as each object's exact cost can involve non-quantifiable factors such as reputational risk or environmental damage. Liu et al. [9] consider a cost distribution or minimum/maximum/average costs, which is useful when such parameters are easily found. For the applications that motivate our work, even estimating such parameters can be difficult, especially when the maximum and minimum costs can vary too much to represent the data.

Another related work is classification for imbalanced data where mistakes on a minority class need to be emphasized without defining explicit costs. For surveys, please read [10] and [11]. These approaches are either data-driven methods, which run standard classifiers on training data that is balanced by changing the frequency of classes through over-sampling and under-sampling, or are algorithm-driven and integrate minority bias directly into a classifier. Though our CUT Classification problem has an imbalanced class distribution, it has a clearance requirement for classifying an object as the majority class. None of these previous approaches address this requirement.

## III. CUT Classification

Let a training set $T$ consist of positive objects and negative objects where each object has a fixed set of attributes. Since our problem focuses on clearing the majority class, we treat the majority class as the positive class (good objects) and the minority class as the negative class (bad objects). We can partition $T$ into disjoint groups $\{G_1, \cdots, G_n\}$ in the attribute space, where $G_i$ contains the objects in $T$ sharing the same values over certain non-class attributes. $G_i$ can be considered as a random sample drawn from its *underlying population*, i.e., the set of all objects in the partitioned subspace for $G_i$. For a given *clearance threshold* $t$, we say that $G_i$ is *cleared* with respect to $t$ if for any object $o$ randomly sampled from the underlying population of $G_i$, the probability that $o$ is negative is no more than $t$ (strictly speaking, this statement holds for a given confidence level). Note that the clearance of $G_i$ is a statement about all the objects in the underlying population for $G_i$, not just the training objects in $G_i$. Therefore, if $G_i$ is cleared, we can clear any new object from the underlying population of $G_i$.

*Definition 1 (CUT Classification):* Given a training set $T$ and a clearance threshold $t$, *Clearance Under Threshold (CUT) Classification* is to partition $T$ into disjoint groups $\{G_1, \cdots, G_n\}$ in the attribute space and label each $G_i$ as cleared or non-cleared (with respect to $t$) such that as many future objects as possible are cleared with $\{G_1, \cdots, G_n\}$.

For an example, let $T$ be a set of observed transformers where 10% are contaminated (negative) and the remaining 90% are uncontaminated (positive). If no action is required when the risk of being contaminated is no more than 2%, then $t$ can be set to 2%. One partition of $T$ is $\{G_1, G_2\}$, where $G_1$ contains the transformers in $T$ manufactured by $M$ while $G_2$ contains the transformers in $T$ manufactured by any other manufacturer. Suppose that $G_1$ is cleared with respect to $t$ while $G_2$ is not then this model clears any transformer manufactured by $M$ because the risk of contamination is below 2%.

Since the training set is a random, probably small, sample of the underlying population, we cannot use the observed proportion of negatives in $G_i$ *as it is* to make a decision on clearing $G_i$. For example, if $G_i$ contains 1 negative object out of 5 objects, it does not mean that $G_i$ can be cleared for $t = 20\%$; however, if $G_i$ contains 20 negative objects out of 100, $G_i$'s risk is closer to the clearance threshold of 20% because a large sample size implies a more accurate observed risk. With this idea, we apply the standard statistical approach that estimates risk with *confidence interval*s and a *confidence level* [12] where $G_i$ is treated as a random sample of the underlying population. A confidence interval $[l, u]$ with a 95% confidence level says that if random samples were repeated multiple times, $[l, u]$ would encompass the true population's proportion 95% of the time. Therefore, if the upper limit $u$ of this confidence interval, denoted by $ub(G_i)$, is at most $t$, for any object in the underlying population of $G_i$, we have 97.5% confidence (the one-sided confidence level) that the probability of the object being negative is at most $t$.

Any partition-based algorithm, such as a C4.5 decision tree partitioning [13], could solve CUT Classification by labeling each $G_i$ for a leaf node as cleared if $ub(G_i) \leq t$, or non-cleared, otherwise. However, these methods are not ideal because they generate the set of $G_i$ regardless of the problem's clearance threshold $t$, and $t$ is factored in only at the "end of the game". Let us consider an example to explain this point.

*Example 1:* Let $T$ contain 1000 observed transformers, in Fig. 1, with two attributes, Region and Manufacturer, in which 150 (or 15%) contain PCBs and the other 850 do not. Let $t = 5\%$. In the following discussion, $ub(G_i)$ is computed from

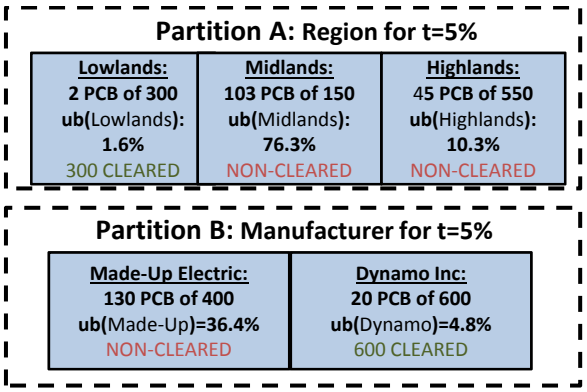
Fig. 1: A Hypothetical Training Set, $T$.

| Manufacturer | Region | Observed PCB | $\mathbf{ub(G_i)}$ |
|---|---|---|---|
| Made-up Electric | Lowlands | 2 in 150 (1.3%) | 4.2% |
| Made-up Electric | Midlands | 98 in 100 (98%) | 100% |
| Made-up Electric | Highlands | 30 in 150 (20%) | 25.8% |
| Dynamo Inc. | Lowlands | 0 in 150 (0%) | 1.8% |
| Dynamo Inc. | Midlands | 5 in 50 (10%) | 19.1% |
| Dynamo Inc. | Highlands | 15 in 400 (3.8%) | 5.6% |

Wilson's score interval [12] with a 90% confidence level.

Fig. 2 shows two initial partitions of $T$: *Partition A* is by Region and *Partition B* is by Manufacturer. A decision tree partition favors *Partition A* because of the larger information gain or gain ratio. *Partition A* clears the first group of 300 objects because $ub(Lowlands) \leq t$ while *Partition B* clears the last group of 600 objects because $ub(Dynamo) \leq t$. Thus *Partition A* clears only half the objects that *Partition B* clears. This problem is exacerbated with the decision tree's final partition shown in Fig. 3a, which still only clears 300 transformers in $T$, while *Partition B*'s further partition, in Fig. 3b, clears a far better 750. If future data is similar to $T$, the decision tree's purity-based partitioning would lead to significantly fewer cleared transformers.□

Fig. 2: Some $T$ partitions and their purity-based scores: *InfoGain(A)=0.223, InfoGain(B)=0.119; GainRatio(A)=0.162, GainRatio(B)=0.123.*




This example suggests that a maximum purity of classes (e.g. information gain) does not necessarily lead to a maximum clearance and the maximum clearance does not require a maximum purity of classes. The reason is that the purity based scores ignore the user's clearance threshold, which is also the reason why classification based on some cost information may not clear many future objects.

## IV. CUT Classifiers

Our CUT Classification solvers are motivated to find partitions, like Fig. 2's *Partition B*, that clear many training objects since such partitions are likely to clear many future objects. To clear large groups of a set of training data, $G$, $CUT^+(G,t)$ in Algorithm 1 searches for cleared groups in multiple rounds by calling the procedure $CUT\_Tree(G,t)$ (described later in Algorithm 2) in a loop. $CUT\_Tree(G,t)$ partitions the set of remaining training objects, $G$, by constructing a tree in which each leaf node is labeled as either "cleared" or "non-cleared". If any leaf node is labeled as cleared, all training objects in cleared leaf nodes are removed from $G$ and $CUT\_Tree(G,t)$ is called on the remaining non-cleared training objects $G$.

**Algorithm 1** $CUT^+(G,t)$

**Require:** A set of training objects, $G$, and a clearance threshold, $t$

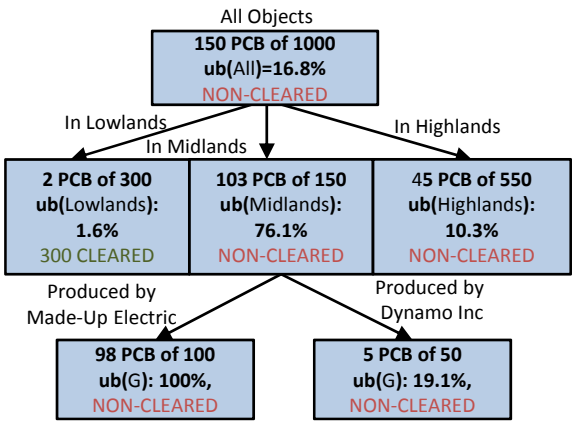
1: **repeat**
2: Call $CUT\_Tree(G,t)$
3: Remove the objects assigned to a cleared group from $G$
4: **until** no cleared group is found

**Clear New Objects using CUT Classifiers**. Suppose that $CUT^+(T,t)$ terminates after $k$ iterations and $CL_i$ denotes the set of cleared groups found in the $i$th iteration, $1 \leq i \leq k$. To classify a future object $o$ with an unknown class, we examine the list $CL_1, \cdots, CL_k$ from left to right and look for the first group that matches the attribute values of $o$. Groups within the same $CL_i$ are not ordered because at most one group can match the attribute values of $o$. If a matching group is found, $o$ is cleared; if no matching group is found, $o$ is non-cleared.

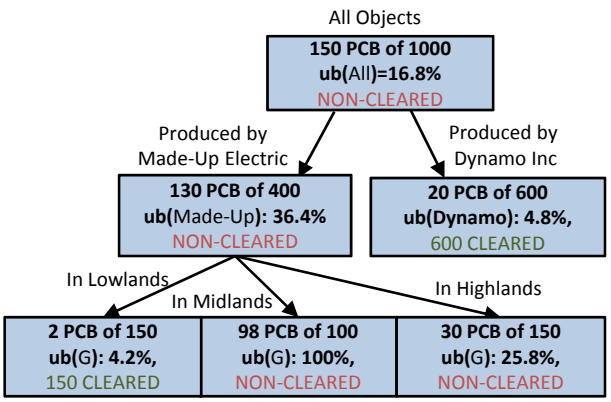
### A. CUT Tree Overview

Next, we discuss the $CUT\_Tree$ partition algorithm that searches for cleared groups using the pseudocode in Algorithm 2. At a high level, this partition algorithm is similar to decision tree construction in that it repeatedly partitions the set $G$ according to attribute values but, instead of maximizing the separation of class labels, our splitting criterion aims to

Fig. 3: Comparison of the partitions derived by a decision tree and our solutions for the data in Fig. 1, $t = 5\%$.




(a) Decision tree classifier clears 300 transformers.




(b) Our solution clears 750 transformers.

greedily clear as many objects as possible as per the goal of $CUT^+$. This partitioning process is repeated recursively on the partition's subgroups until a group is cleared.

**Algorithm 2** $CUT\_Tree(G, t)$

**Require:** A group of training objects, $G$, and a clearance threshold, $t$.

1: **if** $ub(G) \leq t$ **then**
2: label $G$ as *cleared* and return
3: **end if**
4: **if** $Candidate(G)$ is empty **then**
5: label $G$ as *non-cleared* and return
6: **end if**
7: **for all** partitions $P = \{G_1, \cdots, G_n\}$ in $Candidate(G)$ **do**
8: compute $Value(G, P, t) = \sum_{i=1}^{n} Score(G, G_i, t)$
9: **end for**
10: let $Best\_P = \{G_1, \cdots, G_n\}$ be the maximum valued partition $P$
11: **for all** $G_i$ in $Best\_P$ **do**
12: create a child node for $G_i$ and call $CUT\_Tree(G_i, t)$
13: **end for**
14: **return** all cleared and non-cleared groups

In this algorithm, Lines 1-3 check if $G$ can be cleared, i.e., $ub(G) \leq t$, where $ub(G)$ returns the confidence interval's upper limit for the probability of the negative class in the population of $G$, discussed shortly. If $ub(G) \leq t$, the node associated with $G$ is labeled *cleared* and the call returns. Otherwise, Lines 4-6 check if $G$ can be further partitioned, where $Candidate(G)$ contains all the candidate partitions of $G$. If not, the node associated with $G$ is labeled *non-cleared* and the call returns; otherwise, Lines 7-10 identify the candidate partition with the most "valued" ability to clear future objects. For each partition $P = \{G_1, \cdots, G_n\}$ in $Candidate(G)$, this value is determined by $Value(G, P, t)$, which is defined as the sum of $Score(G, G_i, t)$ for all non-empty $G_i$'s. This Score function values a group $G_i$ that clears many objects or will potentially clear many objects after further partitioning. The exact scoring formulas are discussed in Section IV-C. After this best partition $\{G_1, \cdots, G_n\}$ is chosen, Lines 11-13 calls $CUT\_Tree$ recursively on each $G_i$ to continue the search for cleared groups. When all recursive calls end, every leaf node is labeled either cleared or non-cleared. The last step is to return all cleared and non-cleared groups.

The rest of this section focuses on the remaining unexplained functions, $Candidate(G)$ and $Score(G, G_i, t)$. Please refer to Table II for notation.

TABLE II: NOTATION.

| Notation | Definition |
|---|---|
| $\mathbf{G}$ | A group of training set objects. |
| $\lvert\mathbf{G}\rvert$ | The number of objects in $G$ |
| $\lvert\mathbf{G}^-\rvert$ | The number of minority negative objects in $G$ |
| $\lvert\mathbf{G}^+\rvert$ | The number of majority positive objects in $G$ |
| $\mathbf{ub(G)}$ | The upper limit of the confidence interval estimated from $G$ |

### B. Candidate Partitions

For a categorical attribute with multiple values, we consider only binary partitions that are one-vs-the-rest. For example, to partition the training set by the attribute Region with three values Lowlands, Midlands, and Highlands, we consider three one-vs-the-rest candidate partitions in which one attribute value forms a subgroup and the remaining two values form the other subgroup. This restriction avoids the exponential number of candidate partitions and the resulting groups are easier to understand. For a numerical attribute, a possible split is tested at the middle point between every pair of adjacent values, similar to the decision tree approach [13].

### C. Scoring Functions

Recall that we use the upper limit $ub(G)$ of the confidence interval as the probability of the negative minority class in $G$'s region of attribute space. Therefore, if $ub(G) \leq t$, $G$ is cleared. We choose this $ub(G)$ to be the upper limit of Wilson's score interval because it has good properties for a small number of trials and near-zero proportions of negatives. See [12] for more details on Wilson's score interval.

Now we define $Score(G, G_i, t)$, the function that measures the "potential" that the future objects in $G_i$'s region of attribute space can be cleared. We consider three alternative scores based on the Wilson Interval's $ub(G_i)$. Refer to Table II for the notations used.

#### *Immediate Clearance Scoring*

Our first scoring policy, Immediate Clearance, values the immediate count of cleared objects in $G_i$ without further partitioning, defined as the number of elements in $G_i$ if $G_i$ can be cleared, or zero otherwise:

$$Score(G, G_i, t) = \begin{cases} |G_i| & \text{if } ub(G_i) \leq t \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

In short, this score is greedy in maximizing the number of cleared objects while ignoring all non-cleared objects.

#### *Risk Reduction Scoring*

Immediate Clearance is problematic if all candidate partitions have no cleared group because it will choose a partition arbitrarily. In this case, an alternative is to consider the "risk reduction" due to partitioning, defined as $|G_i|(ub(G) - ub(G_i))$ if $ub(G_i) < ub(G)$ and zero otherwise.

$$Score(G, G_i, t) = \begin{cases} |G_i| & \text{if } ub(G_i) \leq t \\ |G_i|(ub(G) - ub(G_i)) & \text{else if } ub(G) - ub(G_i) > 0 \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

The risk reduction comes from the reduced $ub(G_i)$ after partitioning. Note that, although $G_i$ is a smaller sample than $G$, if there is a higher proportion of positive objects in $G_i$, $G_i$ could have a smaller upper limit $ub(G_i)$ for the true proportion of negative objects than $G$.

*Pure Potential Scoring*

With either of the above scores, it was difficult to clear any objects at very low clearance thresholds because clearance requires a large amount of positive objects. To help clear objects in this case, our third strategy favors $G_i$'s with a higher proportion of positive objects than $G$, defined as

$$Score(G, G_i, t) = \begin{cases} |G_i| & \text{if } ub(G_i) \leq t \\ |G_i^+| & \text{else if } \frac{|G_i^+|}{|G_i|} - \frac{|G^+|}{|G|} > 0 \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

Both this scoring and the risk reduction scoring favor a higher proportion of positive objects, whereas the former also considers the impact of sample size.

With these definitions, we have described our three instantiations of $CUT^+$.

## V. Experimental Results

This section discusses our experiments[3] evaluating CUT Classifiers on a case study of identifying PCB contaminated transformers with BC Hydro and five public benchmark data sets from UCI's Machine Learning Repository [14].

### A. Evaluation Criteria

We evaluate a solution by how well it clears positive objects while leaving the negative objects non-cleared in a testing set. Using $k$-fold cross validation, we separate the labeled data into a training set and a testing set then build a partition with the training set to produce groups $\{G_1, \cdots, G_n\}$ that are cleared according to the condition $ub(G_i) \leq t$. We then use the partition $\{G_1, \cdots, G_n\}$ to classify each object in the testing set as cleared or non-cleared following the procedure in Section IV. This decision is compared with the actual class of each object $o$ in the testing set: if $o$ is cleared and has a positive class, it is a true positive; if $o$ is cleared and has a negative class, it is a false positive; if $o$ is non-cleared and has a positive class, it is a false negative; if $o$ is non-cleared and has a negative class, it is a true negative. Let $TP, FN, FP, TN$ denote the total counts of true positives, false negatives, false positives, and true negatives.

Our first evaluator is the standard true positive rate or TPR:

$$TPR = \frac{TP}{|Positives|}$$

where $|Positives|$ denotes the number of positive objects in testing set. TPR is the percentage of the positive objects in the testing set that are cleared by the algorithm. When this value is high, the partition correctly clears a large proportion of positive objects (e.g. non-PCB transformers).

However, we also need to ensure that cleared objects are reliably positive. Our initial idea was to apply the usual false positive rate but this evaluation ignores how the clearance threshold $t$ would allow some number of *acceptable* errors. For example, suppose $t$ is large and suppose a cleared group's testing data's proportion of negatives is $t/2$ then this cleared group is correctly cleared because the frequency of negatives is far below the user's clearance threshold. Nonetheless, the false positive rate could imply a poor result because this cleared group incurs a large number of false positive errors. We therefore develop a new evaluator that considers the threshold's tolerance.

By the definition of the clearance threshold, $t$ is the maximum acceptable probability of being a negative object. Therefore, if $|G_i'|$ is the number of objects in the testing set cleared by $G_i$, then up to $t \cdot |G_i'|$ of these objects are allowed to be negative; if $FP_i$ is the number of false positives cleared by $G_i$, $FP_i - t \cdot |G_i'|$ is the portion that exceeds this allowable maximum. This leads to the following "false positives above threshold" definition:

$$FP(t) = \sum_{i=1}^{n} max\{0, FP_i - t \cdot |G_i'|\} \quad (5)$$

We define the false positive rate with $t$, or FPR(t), as:

$$FPR(t) = \frac{FP(t)}{|Negatives|}$$

where $|Negatives|$ denotes the count of negative objects in the testing set. FPR(t) should be low because it is the percent of negative objects cleared over the limit allowed by $t$.

A larger TPR and smaller FPR(t) represent a stronger solution, so all algorithms will be evaluated by TPR and FPR(t). In general, a larger FPR(t) is expected for having a larger TPR. Notice that a larger FPR(t) means that more negative objects are cleared (incorrectly), but the risk of a cleared object is still under the specified clearance threshold $t$, as assured by the CUT problem statement.

### B. Tested Methods

Our $CUT^+$ algorithm can be instantiated by the three score functions in Section IV-C. So we consider three versions of the algorithm and name them by the corresponding scoring policy: Immediate Clearance (IC), Risk Reduction (RR), and Pure Potential (PP). For competitors, we are not aware of any existing algorithm that takes a user-specified clearance threshold $t$ and clears objects under the threshold. The best baseline solutions we can think of are classic cost-sensitive classifiers that use a cost matrix defined using the clearance threshold $t$. We focus on cost-sensitive classifiers based on attribute space partitioning like decision tree partitioning for two reasons. First, our method is based on attribute space partitioning, so baselines based on similar attribute space partitioning would eliminate performance differences originating from the modeling approach and thus focus on the differences from the novelty of our method. Secondly, attribute space partitioning produces readable structures to an industrial client in terms of the attributes that clear an object.

We consider three baseline algorithms. Baseline1 (BL1) is the standard decision tree classifier except its leafs are outputted as a partition of objects, where a leaf $G_i$ is cleared if $ub(G_i) \leq t$, or non-cleared otherwise. Baseline2 (BL2) is SMOTE from classification for imbalanced data [15] using decision trees. Baseline3 (BL3) is a version of MetaCost with C4.5 [7] that uses multiple decision trees on different

[3]The code for $CUT^+$, the baseline methods, and the UCI data sets used in this experiment are available for download at: http://www2.cs.sfu.ca/~wangk/software/CUT_classification/

random samples of the training data and relabeling of this data to guide a final decision tree to minimize certain costs. To derive MetaCost's cost matrix, we set the critical probability $p^* = C_2/(C_2 + C_3)$ [4], explained in Section II, to the clearance threshold $t$, which gives $C_1 = 0$, $C_4 = 0$, $C_2 = 1$, and $C_3 = 1/t - 1$; this system would label an instance $x$ as the majority positive class if the probability estimated for $x$ to be in the minority negative class is less than $t$ because the estimated cost of a false positive is less than the estimated cost of a false negative. Note that even with this cost matrix, the threshold $t$ on the critical probability does not enforce the clearance requirement for $t$ of CUT Classification. Our choices were motivated by a recent survey [16] that argued that SMOTE was one of the best imbalanced solutions while MetaCost with C4.5 was the best cost-sensitive solution. All baselines use the Weka library's default parameters [17].

### C. BC Hydro PCB Experiments

This data set consists of the records for 1050 sampled bushings (oil filled transformer parts) collected by our industrial partner BC Hydro since 1960, with the attributes described in Table III. To define an unsafe or negative bushing, we consider two different maximum levels for milligrams of PCBs per kilogram of oil: 50 $mg/kg$ is the maximum level before the United Nations requires a removal by 2025 [1] while 350 $mg/kg$ represents a more dangerous level that is close enough to the immediate removal level of 500 $mg/kg$, yet has enough proportion of negative objects to be a meaningful CUT Classification problem. We refer to these two data sets as *PCB50* and *PCB350*. Because of a non-disclosure agreement, we cannot divulge these contamination percentages but we will say that *PCB50* has less than 30% negative bushings and *PCB350* has less than 5% negative bushings. Because these problems use recognized unsafe PCB levels, these experiments are valuable to electrical companies.

TABLE III: PCB BUSHING ATTRIBUTES

| Attribute Type | List of Attributes |
|---|---|
| *Numeric* | Oil Volume, Current, and Voltage. |
| *Categorical* | Region, Subregion, Equipment Type, Equipment Model, Manufacturer, Substation, and Bushing Position. |
| *Contamination* | Milligrams of polychlorinated biphenyls per kilogram of oil. |

All experiments were conducted using three-fold cross-validation to ensure that there are sufficient negative objects in the testing set. Since *PCB350* has very few negative objects, stratified sampling is used to ensure that each fold gets roughly the same count of negative objects. The upper limit $ub(G)$ of the confidence interval is calculated by Wilson's Bernoulli estimation with the confidence level of 95% (equivalently, a one-sided confidence level of 97.5%). We believe that 97.5% is an acceptable confidence but, in other applications with more data, users may prefer a different confidence level.

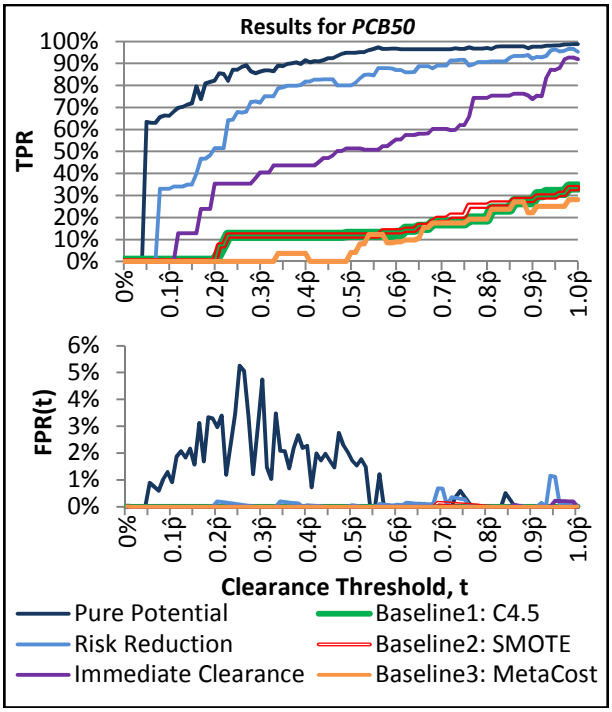
### PCB50 Results

Fig. 4 shows the TPR and FPR(t) of the algorithms as the threshold $t$ increases from 0% to the observed rate of PCB contamination in the data, $\hat{p}$, in the increment of $0.01\hat{p}$. For the TPR graph, Pure Potential (the dark blue line) is better than Risk Reduction (the light blue line), which is better than Immediate Clearance (the purple curve). The three baselines, however, clear far fewer positives: they clear only less than an eighth of the best TPR from a CUT Classifier in the main interesting problem range: low thresholds from $t = 0\%$ to $t = 0.5\hat{p}$. Between the baselines, Baseline1 (green) performs slightly worse than the imbalanced data competitor, Baseline2 (red), while Baseline3 (orange) performs the worse.

The baselines' poor TPR performance originates from ignoring the "clearance under threshold" goal of CUT problems: both Baseline1 and Baseline2 clear few positives because they underestimate acceptable rates of contamination, which results in the majority of groups being too contaminated to be cleared. In contrast, Baseline3 acts too conservatively: with the given cost matrix, a missed PCB bushing costs $1/t - 1$ more than a tested safe bushing, which can be very large for this domain. For example, $t = 5\%$ implies that missed PCB bushings are 19 times more expensive than unnecessarily testing a non-PCB bushing. This large cost causes the algorithm to partition into mostly very small and very pure *"Equipment Model"* groups because such groups have the lowest observed likelihood of contamination and therefore the lowest estimated cost; however, most of these pure groups only have tens of objects and are thus non-cleared because the risk of contamination is too high considering the small sample size. Overall, this shows how previous approaches, such as cost minimization, are poor CUT solutions in comparison to our methods.

Unlike the TPR graph, the FPR(t) graph has more mixed results with an increasing $t$. As $t$ increases, more objects are cleared, which increases the chance for false positives,

Fig. 4: *PCB50* Problem with 1050 Bushings



but since $FP(t)$ can be reduced by a larger $t$ (see Eqn (5)), FPR(t) can decrease, which matches the intuition that a larger tolerance threshold means less errors. This is one explanation for the fluctuation of FPR(t) as $t$ increases. The baselines have a smaller FPR(t) than the CUT Classifiers simply because they do very little in clearing positive objects in the testing set as shown by their small TPR. Pure Potential has up to $5.3\%$ FPR(t), Risk Reduction is up to $2\%$ FPR(t), while Immediate Clearance is capped to $0.5\%$. Pure Potential's very high TPR may be worth the $5\%$ FPR(t). However, Risk Reduction's lower FPR(t) but slightly lesser TPR may make it preferable. Ultimately, the industry client must weigh the benefit of clearing more positive objects with the risk of clearing negatives to make the final decision.

Good clearance of positives is not useful unless the non-cleared groups consist of mostly negatives, which is why we also evaluated the proportion of negative testing objects in the non-cleared groups, i.e., $TN/(FN + TN)$; because of limited space, we only consider the case for $t = 0.25\hat{p}$. For Pure Potential, Risk Reduction, and Immediate Clearance, the proportions of negative testing objects in non-cleared groups are $2.74\hat{p}$, $1.91\hat{p}$, and $1.33\hat{p}$, whereas with the same $t$ the proportions for Baseline1 and Baseline2 are both $1.09\hat{p}$ while Baseline3 is $\hat{p}$, where $\hat{p}$ is the observed proportion of negative objects in the data set. This shows that Pure Potential has the best quarantine of high-risk objects, i.e., the non-cleared groups contain negative objects 2.74 times more frequently than the observed rate.

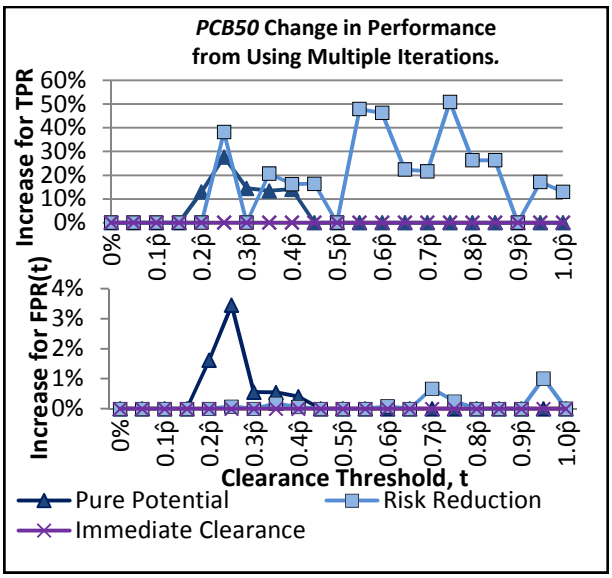
**Improvements from Multiple Iterations in CUT$^+$.** Another question to address is whether having more than one *CUT Tree* iteration in $CUT^+$ improves performance. First, let's explore this claim on a single threshold, $t = 0.35\hat{p}$. Table IV shows the TPR and FPR(t) after each iteration. All three algorithms terminate within three iterations, but each iteration improves TPR substantially without too much of an increase in FPR(t). This finding clearly supports the idea of clearance through multiple iterations.

TABLE IV: TPR AND FPR($t$) AFTER EACH ITERATION $i$, $t = 0.35\hat{p}$.

| | Risk Reduction | | Pure Potential | | Immediate Clearance | |
|---|---|---|---|---|---|---|
| **i** | **TPR** | **FPR(t)** | **TPR** | **FPR(t)** | **TPR** | **FPR(t)** |
| 1 | 58% | 0% | 75% | 1.5% | 44% | 0% |
| 2 | 65% | 0% | 89% | 2.1% | | |
| 3 | 79% | 0.2% | | | | |

Fig. 5 plots the increase in TPR and FPR(t) from a single iteration of $CUT^+$ to multi-iterations of $CUT^+$ for various settings of $t$. For example, TPR has increased by 40% at $t = 0.25\hat{p}$ for Risk Reduction when running the algorithm to its completion compared to running it only a single iteration. In general, Pure Potential's TPR increases by 10% to 25% while Risk Reduction's TPR can increase by 50%. The cases of 0% TPR improvement does show that additional iterations are not always successful, especially for Immediate Clearance, because either the threshold is too low, making it too difficult for the scoring system to extract any cleared groups, or because the first iteration clears the majority of the set therefore making additional attempts on the non-cleared set fruitless. For an example of this second case, Pure Potential extracts over 90% of uncontaminated bushings with only the first iteration when $t \geq 0.4\hat{p}$. Like the TPR, multiple iterations also increase FPR(t); for example, the largest increase is at $t = 0.25\hat{p}$ from Pure Potential's first iteration's FPR(t), 1.8%, to 5.3%. This increase in false positives is expected for a higher TPR.

Fig. 5: The increase of TPR and FPR(t) relative to a single iteration.



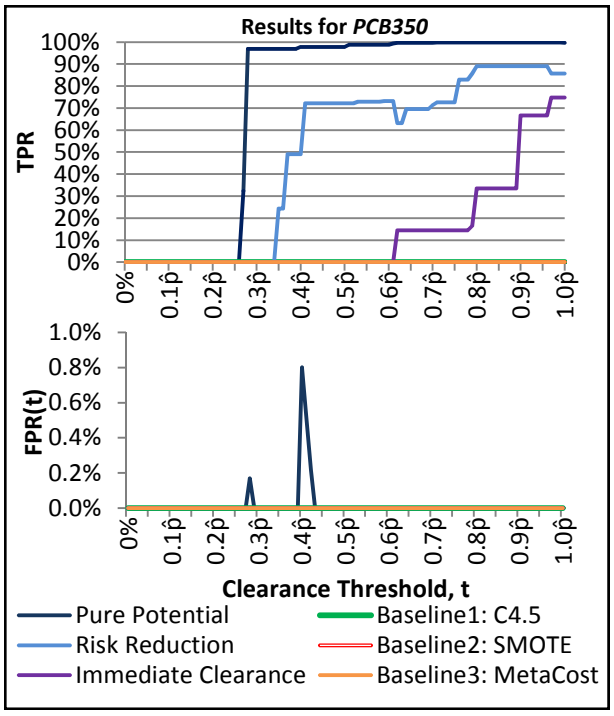

Further analysis showed that there are relatively few iterations, generally one or two and at most five, however, each additional iteration can largely improve performance as shown in these results. A similar performance improvement of $CUT^+$'s multi-iteration was also observed over the other data sets. In the rest of the section, we focus on the TPR and FPR(t) of multi-iteration $CUT^+$.

*PCB350 Results*

Fig. 6 shows TPR and FPR(t) for the *PCB350* data set, which has the proportion of negative bushings, $\hat{p}$, under 5%. With fewer negative objects, the CUT Classifiers' benefits are even more pronounced: no baseline clears any bushings until far after the clearance threshold exceeds the observed proportion $\hat{p}$ (beyond the graph's range in Fig. 6), which is not a useful solution for determining PCB bushings. This result occurs because the baselines either severely underestimated the cost of contamination and therefore trivially group together all bushings (Baseline1 and Baseline2) or severely overestimated cost so any groups of mostly safe objects are very small and therefore too difficult to clear with the threshold (Baseline3).

In contrast, the CUT Classifiers perform similarly to the *PCB50* case: Pure Potential's TPR is better than Risk Reduction's TPR, which is better than Immediate Clearance's. Also, FPR(t) for all algorithms other than Pure Potential is zero for every clearance threshold $t$ tested. Note that a zero FPR(t) does not mean that no PCB bushing is cleared at all; it only means that the proportion of PCB bushings cleared is less than the tolerable threshold $t$.

Fig. 6: *PCB350* Problem with 1050 Bushings.




Based on the above, Pure Potential's very high TPR and minimal FPR(t) make it the best method on this data. Additional results confirmed that the non-cleared groups are mostly made up of negative testing objects. At the threshold $t = 0.4\hat{p}$, the proportion of negatives in non-cleared groups are: Pure Potential $17.2\hat{p}$, Risk Reduction $2.6\hat{p}$, while all other methods clear nothing and thus have a proportion of $\hat{p}$. The high TPR and the extremely high proportion of negative testing objects in the non-cleared groups show that Pure Potential successfully quarantines contaminated objects.

These PCB results demonstrate that the baselines fail to adequately clear bushings because they ignore the CUT problem's risk requirement. In contrast, our methods are far better because they integrate a CUT problem's objective, i.e., finding cleared groups, directly into all modeling decisions. Considering the importance of safely removing PCBs, these results are very encouraging.

### D. Experiments on UCI Data Sets

This work was motivated by the PCB contamination problem to address a real industrial need but the PCB data sets are confidential, which is why we demonstrate the effectiveness of the proposed method on five publicly available data sets from UCI's machine learning repository [14]. These data sets are similar to our PCB contamination data set in that the negative class corresponds to bad objects hidden among good objects. The observed proportions of the negative class in these data sets, $\hat{p}$, are: *Cars* with 70% bad cars in 1728 records, *Mushroom* with 48% poisonous mushrooms in 8124 data, *Breast Cancer* with 35% malignant tumors of 699 data, *Surgery* [18] with 15% at-risk patients out of 420, and *Thyroid* for 2.75% hyperthyroid patients out of 2799. We replaced missing values with the average, if numeric, or the mode, if categorical, using Weka [17]. As before, we performed three-fold cross validation with stratified sampling to evenly distribute bad objects among each fold.

Given the observed proportion of negative objects, $\hat{p}$, we tested five clearance thresholds: $0.2\hat{p}$, $0.4\hat{p}$, $0.6\hat{p}$, $0.8\hat{p}$, and $\hat{p}$. These experiment results are presented in Table V where the **bold** numbers indicate the best performer, the algorithm that improves on competitors either by having better TPR and FPR(t) or a slightly higher FPR(t) but a sufficiently larger TPR. From these results, our new CUT Classifiers demonstrate the best overall performance for all data sets. Due to limited space, we omit the detailed proportion numbers of negative objects in the non-cleared testing set (i.e. $TN/(TN+FN)$), but we can confirm that the winning method has the highest proportion.

Furthermore, the CUT Classifiers perform better than the baselines in very different distributions from *Car*'s 70% negative class to *Thyroid*'s 2.75% negative class. The good performance on *Car* is surprising because more negatives than positives is not the typical scenario that CUT Classifiers are designed to deal with. Another benefit of CUT Classifiers is shown in the *Surgery* set where the model is built with only around 300 training instances. Considering how stringent the 95% confidence level is, Pure Potential (PP) still clears an impressive portion of positive objects with a small FPR(t). Contrariwise, the baselines do not clear any objects until very high thresholds. The baselines are more competitive for low thresholds in *Mushroom* and *Thyroid*, likely because these problems have thousands more objects than the other problems, making clearing groups easier. Nonetheless, the CUT Classifiers are still better in raw performance.

Among the three CUT Classifiers, Pure Potential (PP) is the most successful because it is first place in 22 of 25 cases. Even when it loses, it is not significantly off from the best competitor. These results therefore affirm the value of our CUT Classifiers on a variety of meaningful industrial problems.

## VI. Conclusion

Most existing cost-sensitive classifiers attempt to minimize expected costs according to a user-specified cost matrix. While there are cases where this cost information is available, there are many application scenarios where it is too difficult for a user to specify this cost information because of unpredictable or non-quantifiable costs. Even though this problem may be addressed in part by relaxing the exact costs to an approximate specification such as cost intervals or a cost distribution, these methods still require certain knowledge about the cost that might be hard or expensive to obtain. Our solution to this "illusive cost" problem is Clearance Under Threshold (CUT) Classification, a method that does not depend on the cost information and instead takes the user's input on a maximum acceptable probability for high-risk objects, called the clearance threshold, to clear low-risk groups. As shown by this project with our industrial partner, BC Hydro, such clearance thresholds, which captures the risk in terms of probability, often can be obtained from industry standards and reliability reports. As demonstrated by our study on both real industrial data and public benchmark data, this maximum risk based CUT classification approach better clears good objects and quarantines bad objects than existing methods.

TABLE V: TPR AND FPR($t$) FOR THE UCI DATA SETS.

| | | TPR | | | | | | FPR(t) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Baselines | | | CUT Classifiers | | | Baselines | | | CUT Classifiers | | |
| | t | BL1 | BL2 | BL3 | IC | RR | PP | BL1 | BL2 | BL3 | IC | RR | PP |
| **Cars** | 14.0% | 17.2% | 18.5% | 17.2% | 0% | **76.4%** | 63.5% | 0% | 0% | 0% | 0% | **0.0%** | 0.2% |
| | 28.0% | 39.8% | 37.8% | 42.1% | 2.9% | 94.4% | **96.7%** | 0.1% | 0.1% | 0.1% | 0% | 0% | **0%** |
| | 42.0% | 75.1% | 73.9% | 70.9% | 30.9% | **100%** | **100%** | 0% | 0% | 0.1% | 0.5% | **0%** | **0%** |
| | 56.0% | 83.4% | 84.9% | 83.2% | 87.5% | **100%** | **100%** | 0% | 0% | 0% | 0.1% | **0%** | **0%** |
| | 70% | 89.8% | 89.6% | 91.7% | 96.7% | **100%** | 99.2% | 0.1% | 0.1% | 0% | 0.5% | **0.6%** | 0.4% |
| **Mushroom** | 9.6% | 94.1% | 94.6% | 94.1% | 99.0% | **100%** | **100%** | 0% | 0% | 0% | 0% | **0%** | **0%** |
| | 19.3% | **100%** | 99.8% | **100%** | 99.7% | **100%** | **100%** | **0%** | 0% | **0%** | 0% | **0%** | **0%** |
| | 28.9% | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **0%** | **0%** | **0%** | **0%** | **0%** | **0%** |
| | 38.6% | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **0%** | **0%** | **0%** | **0%** | **0%** | **0%** |
| | 48.2% | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **0%** | **0%** | **0%** | **0%** | **0%** | **0%** |
| **Cancer** | 6.9% | 84.0% | 83.6% | 84.0% | 91.0% | 91.0% | **95.4%** | 0% | 0% | 0% | 0% | 0% | **0%** |
| | 13.8% | 84.0% | 83.6% | 83.4% | **95.2%** | **95.2%** | **95.2%** | 0% | 0% | 0% | **0%** | **0%** | **0%** |
| | 20.7% | 88.9% | 89.9% | 86.5% | **98.5%** | **98.5%** | **98.5%** | 0% | 0% | 0% | **0%** | **0%** | **0%** |
| | 27.6% | 88.9% | 89.9% | 88.6% | **98.9%** | **98.9%** | **98.9%** | 0% | 0% | 0.5% | **0.1%** | **0.1%** | **0.1%** |
| | 34.5% | 93.9% | 92.1% | 93.0% | **99.3%** | **99.3%** | **99.3%** | 0% | 0% | 0% | **0%** | **0%** | **0%** |
| **Surgery** | 3.0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% | 0% |
| | 6.0% | 0% | 0% | 0% | 0% | 11.5% | **8.0%** | 0% | 0% | 0% | 0% | 7.0% | **2.7%** |
| | 8.9% | 0% | 0% | 0% | 16.8% | 43.2% | **41.0%** | 0% | 0% | 0% | 5.8% | 9.2% | **2.7%** |
| | 11.9% | 0% | 0% | 36.5% | 26.8% | 55.8% | **55.3%** | 0% | 0% | 1.3% | 2.4% | 8.3% | **1.4%** |
| | 14.9% | 0% | 0% | 14.0% | 54.6% | 74.5% | **79.8%** | 0% | 0% | 3.4% | 6.7% | 11.7% | **0.1%** |
| **Thyroid** | 0.6% | 63.6% | 95.2% | 94.2% | 24.2% | 92.9% | **95.4%** | 0% | 0% | 0% | 0% | 0% | **0%** |
| | 1.1% | 95.4% | 95.2% | 92.9% | 92.9% | 92.9% | **97.3%** | 0% | 0% | 0% | 0% | 0% | **0%** |
| | 1.7% | 95.4% | 95.2% | 93.6% | **98.6%** | **98.6%** | 98.5% | 0% | 0% | 0% | **0%** | **0%** | 0% |
| | 2.2% | 95.4% | 95.2% | 93.4% | **99.1%** | **99.1%** | **99.1%** | 0% | 0% | 0% | **0%** | **0%** | **0%** |
| | 2.7% | 95.4% | 95.2% | 93.4% | **99.6%** | **99.6%** | **99.6%** | 0% | 0% | 0% | **0%** | **0%** | **0%** |

## VII. ACKNOWLEDGMENTS

Our greatest thanks to BC Hydro's R&D department for sponsorship and Canada's Natural Sciences and Engineering Research Council for a Canada Graduate Scholarships-Master's award and a Collaborative Research Development Grant (NSERC Project CRDPJ/402430-2010). This work was partially done when the second author visited SA Center for Big Data Research hosted in Renmin University of China. This Center is partially funded by a Chinese National 111 Project "Attracting International Talents in Data Engineering and Knowledge Engineering Research".

## REFERENCES

[1] United Nations Environment Programme Chemicals, "PCB transformers and capacitors from management to reclassification and disposal," 2002.

[2] United States Agency for Toxic Substances and Disease Registry and Research Triangle Institute, *Toxicological profile for polychlorinated biphenyls: draft*. Agency for Toxic Substances and Disease Registry, 1995.

[3] V. Dhar, "Big Data and Predictive Analytics in Health Care," in *Big Data*, Vol 2, No. 3, Sept. 2014, pp. 113–116,

[4] C. Elkan, "The foundations of cost-sensitive learning," in *IJCAI*, 2001, pp. 973–978.

[5] C. X. Ling and V. S. Sheng, "Cost-sensitive learning and the class imbalance problem," *Encyclopedia of Machine Learning*, 2008.

[6] S. Lomax and S. Vadera, "A survey of cost-sensitive decision tree induction algorithms," *ACM Comput. Surv.*, vol. 45, no. 2, pp. 16:1–16:35, Mar. 2013.

[7] P. Domingos, "MetaCost: A general method for making classifiers cost-sensitive," in *KDD*, 1999, pp. 155–164.

[8] B. Zadrozny and C. Elkan, "Learning and making decisions when costs and probabilities are both unknown," in *In Proceedings of the Seventh International Conference on Knowledge Discovery and Data Mining*. ACM Press, 2001, pp. 204–213.

[9] X.-Y. Liu and Z.-H. Zhou, "Learning with cost intervals," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ser. KDD '10. New York, NY, USA: ACM, 2010, pp. 403–412.

[10] "Data mining for imbalanced datasets: An overview," in *Data Mining and Knowledge Discovery Handbook*, O. Maimon and L. Rokach, Eds., 2005.

[11] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intell. Data Anal.*, vol. 6, no. 5, 2002.

[12] M. Kendall and D. Stuart, "Inference and relationship," *The Advanced Theory of Statistics*, vol. 2, 1973.

[13] J. R. Quinlan, *C4.5: Programs for Machine Learning*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.

[14] D. N. A. Asuncion, "UCI machine learning repository," 2007. [Online]. Available: http://www.ics.uci.edu/∼mlearn/MLRepository.html

[15] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: Synthetic minority over-sampling technique," *J. Artif. Int. Res.*, vol. 16, no. 1, pp. 321–357, Jun. 2002.

[16] V. López, A. Fernández, S. García, V. Palade, and F. Herrera, "An insight into classification with imbalanced data: Empirical results and current trends on using data intrinsic characteristics," *Inf. Sci.*, vol. 250, pp. 113–141, 2013.

[17] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten, "The weka data mining software: An update," *SIGKDD Explor. Newsl.*, vol. 11, no. 1, pp. 10–18.

[18] M. Zikeba, J. M. Tomczak, M. Lubicz, and J. 'Swikatek, "Boosted svm for extracting rules from imbalanced data in application to prediction of the post-operative life expectancy in the lung cancer patients," *Applied Soft Computing*, 2013.