# Modifying Boosted Trees to Improve Performance on Task 1 of the 2006 KDD Challenge Cup

Robert M. Bell
AT&T Labs-Research
180 Park Ave.
Florham Park, NJ 07932
rbell@research.att.com

Patrick G. Haffner
AT&T Labs-Research
200 S Laurel Ave., Bldg A.
Middletown, NJ 07748
haffner@research.att.com

Chris Volinsky
AT&T Labs-Research
180 Park Ave.
Florham Park, NJ 07932
volinsky@research.att.com

## ABSTRACT

Task 1 of the 2006 KDD Challenge Cup required classification of pulmonary embolisms (PEs) using variables derived from computed tomography angiography. We present our approach to the challenge and justification for our choices. We used boosted trees to perform the main classification task, but modified the algorithm to address idiosyncrasies of the scoring criteria. The two main modifications were: 1) changing the dependent variable in the training set to account for multiple PEs per patient, and 2) incorporating neighborhood information through augmentation of the set of predictor variables. Both of these resulted in measurable predictive improvement. In addition, we discuss a statistically based method for setting the classification threshold.

## Categories and Subject Descriptors

I.5.1 [**Pattern Recognition**]: Models – *statistical*

## Keywords

classification, nonstandard loss function, boosted trees, Real AdaBoost, multiple instance learning.

## 1. THE BIG IDEA

Task 1 of the 2006 KDD Cup involved learning to classify candidate sites as pulmonary embolisms (PEs), or not, using 116 predictor variables measured by computed tomography angiography. A training sample included 3038 candidate sites over 46 patients. For 38 patients with at least one PE, there were 363 positive sites forming 142 PEs (a single PE could include one or more than one site).

In simplified terms, the goal was to identify the largest possible percentage of PEs (PE sensitivity) for a test sample of 21 patients, without exceeding any of the specified limits of 2, 4, and 10 false positives (FPs) per patient for three subtasks. For more details about the data and tasks, see the 2006 KDD Cup web site [5] or the article by Lane in this issue.

The largest challenge in this problem was to address the nonstandard scoring criteria. The criteria differed in two important ways from those for problems based on (weighted) classification error.

- The PE sensitivity criterion gave no extra credit for identifying more than one positive candidate from the same PE. Consequently, simply picking the candidates with the highest predicted probabilities of being positive was likely to waste many false positives (FPs) on candidates that would be unlikely to improve the PE sensitivity score even if they were positive.
- The hard limits for the numbers of FPs, above which a submission was disqualified, made it very important to obtain unbiased estimates for FP rates and good estimates for their uncertainty. Bootstrapping the test data for scoring purposes further complicated the problem.

We approached the problem in two stages. First, we tried to develop an algorithm that would automatically order candidates to optimize expected PE sensitivity for any given number of false positives. We used boosted trees—with a non standard dependent variable and an enhanced set of predictor variables derived from the results of prior boosting steps. We present our algorithm and its development in Sections 2 to 4. Second, given our ordering, we tried to determine how far down the list we could go while maintaining an acceptable risk of violating one or more of the subtask limits (see Section 5).

Task 2 had the same format except that the goal was to correctly classify at least one positive candidate site for each *patient* with one or more positive sites. We focus our presentation on Task 1, except for a brief discussion of Task 2 in Section 6.

## 2. THE BASE CLASSIFIER

Our basic tool for this entire problem was boosted trees, which we chose for three reasons. First, we expected boosted trees to be at least competitive among available classifiers for this type of problem—one with many predictor variables and little prior information about whether relationships were additive or monotonic, much less linear. According to Hastie et al. [4], in 1996, Breiman called AdaBoost the "best off-the-shelf classifier in the world" (see also [1]). Second, many of the available predictor variables are very skewed or have large outliers. Because trees use only the ranking of the predictor variables, this reduces the influence of outliers and the need for transformations. Third, like many other classifiers, boosting focuses on optimizing classification error rather than the desired criterion, PE sensitivity. However, because boosting involves fitting a series of models, it seemed to offer an ideal setting for modifications designed to improve performance on an alternative criterion.

This section describes a boosted trees model that we refer to as the *base classifier*. Section 3 illustrates the need to modify this classifier in order to improve performance on the Task 1 scoring criteria. In Section 4, we describe modifications to the base classifier used in our submission to the KDD Challenge Cup.

## 2.1 Boosted Trees

We created 6-node regression trees using a least squares error criterion, with no pruning. The dependent variable for the trees was set at +1 (for PEs) and -1 (for non PEs). We used the raw 116 predictor variables without any transformations, since trees are invariant to monotonic transformations of the input variables. Each tree produced real-valued predictions in the range [-1, +1].

Trees were aggregated using Real AdaBoost [3, 6], which extends the original AdaBoost algorithm to allow real-valued predictions at each boosting step. Consider a single candidate with outcome $y_i$ and vector of explanatory variables $x_i$. Suppose that at step $t$, the tree produces prediction $h_t(x_i)$. Real AdaBoost produces a series of scores

$$H_T(x_i) = \sum_1^T \alpha_t h_t(x_i) \ ,$$

which can be used for classification or, more generally, to order candidates.

The key to boosting is that the sample is reweighted at each step to focus estimation on the hard to classify cases. Specifically, the weight for case $i$ at step $T$ is proportional to $exp[-y_i H_{T-1}(x_i)]$. Small values of the "shrinkage" parameters $\{\alpha_t\}$ serve to keep the estimator from learning too much from any one tree. Although boosting generally does a surprisingly good job of not over fitting, that can still be a problem if the algorithm continues for too many steps. Sufficiently small values of the $\{\alpha_t\}$ in combination with early stopping provide a means to avoid over fitting. We used 60 boosting steps and set $\alpha_t = 0.15$ for all steps.

Although users of boosting often base binary classification simply on the sign of the final boosting score, $H_T(x_i)$, classification can be based on comparison of $H_T(x_i)$ with any real number. Because Task 1 calls for creating classifiers with a variety of false positive rates, we assess each boosting model using a variety of cut points $c$—with each candidate $i$ classified as positive if and only if $H_T(x_i) > c$.

## 2.2 Selection of Model Parameters and Predictor Variables

Because the goal of Task 1 is to maximize PE sensitivity within fixed FP limits, it was important to base model selection for the base classifier, as well as subsequent modifications, on those criteria. Consequently, we used graphs of estimated PE sensitivity as a function of FPs per patient (see Section 4 for example graphs) as our central model evaluation tool.

In analyses for model selection, both quantities in the graphs were estimated using five- or ten-fold cross validation (CV). For example, for ten-fold CV, patients were grouped into sets of 4 or 5 patients. Candidate models were fit ten times, leaving out a different set of patients each time. PE sensitivity and FP rates were computed for the patients omitted from the model fit and aggregated across the ten groups. Results presented in this paper are based on subsequent 46-fold CVs of the original training data, where patients were omitted one at a time.

Due to time constraints, we performed relatively little exploration of alternative forms for the base classifier. For example, we did not evaluate alternatives to six for the number of nodes in the trees.

Based on an arbitrary initial choice of $\alpha = 0.15$, we did evaluate a wide range of alternatives for the number of boosting steps. Performance improved up to a range of values and degraded beyond that range. Although it was difficult to pinpoint an optimum number of steps, a value of 60 seemed to provide sufficient learning without obvious signs of over fitting. In practice [2], it could be better to use a smaller value of $\alpha$, with correspondingly more steps. However, very limited testing of this idea did not demonstrate noticeable improvement, so we retained the original parameters.

At first glance the list of 116 predictor variables seemed ripe for some type of variable reduction, because large groups of variables were reported to measure related things (e.g., shape, intensity). However, we did not attempt any variable reduction due to the lack of information about what each variable actually measured or any insight about which variables were important for classification. We note that Real AdaBoost is a large-margin learning algorithm that generalizes well on a large number of predictor variables, particularly in combination with early stopping.

We did try a couple ideas for new predictors based on patterns in the data. First, because several predictor variables (most notably the "$z$" coordinate) seemed to be measured on different scales in one hospital (number 21) than in others, we created a dummy variable for that hospital. Second, the variable "$x$" seemed to measure position moving laterally across the chest, with a value of 269 somewhere near the center. Consequently, we created a new variable $|x - 269|$ as a measure of distance from the center. Neither predictor stood out as particularly important in early modeling explorations, so both were dropped.

We also tested using a large number of new predictor variables based on averages of values for sets of nearest neighbors and residuals from those averages. Consider nearest neighbor sets of size 10 (we also tried 3 and infinity). For a given variable $V$ and site $i$, we set $V_i^m$ equal to the mean value of $V$ at the ten sites nearest to $i$ and $V_i^r = V_i - V_i^m$. Although many of the variables created in this manner looked promising in terms of frequency of inclusion in the trees, all attempts to include additional variables like these degraded results from cross validation. Consequently, we abandoned these predictor variables as well.

Predictor variable labeled "Shape neighbor Feature N" (for N = 5, 6, 8, 11, and 33) dominated the resulting model. Those five variables accounted for 67 of the 300 splits across 60 trees, with each involved in at least ten splits (number 6 had 23 splits). Only two other predictor variables "simple intensity statistic 2" and "shape feature 4" reached the ten split level.

# 3. IMPLICATIONS OF THE NEED TO OPTIMIZE PE SENSITIVITY

As mentioned in the Task 1 problem statement "this problem is a multiple-instance problem, where each positive example has multiple instances" [5]. The scoring criteria place a premium on finding at least one candidate from as many PEs as possible. It is just as important to identify PEs of degree 1 (i.e., PEs with a single candidate) as is it to identify PEs of high degree. Indeed, slightly more than half the PEs in the training data had degree 1.

Table 1 shows 46-fold cross validated results from the base classifier calibrated to have exactly 4.0 FPs per patient (similar patterns occurred for other cut points). Overall, the base classifier correctly detected 52 percent of positive candidates and 75 percent of PEs at this cutoff. Notably, performance in terms of candidate sensitivity was best for PEs with degree one or two, although the difference was not statistically significant.

Even though the classifier performed best at identifying individual candidates from PEs with low degree, PE sensitivity was best for high-degree PEs because those PEs offered multiple chances for identification. Of 35 PEs not detected by the base classifier, 27 had degree one and another 6 had degree two; only 2 had degree three or greater. The final column of Table 1 shows that for high degree PEs, a large proportion of positive classifications were redundant.

Consequently, it appeared that it would be very valuable to modify the base classifier to do better on low degree PEs, even at the risk of sacrificing some power to identify high degree PEs. This would be especially valuable if mostly redundant positives were reclassified.

| PE Degree | Number of PEs | Number of Cands | Sensitivity | | Positives/ PE Found |
|---|---|---|---|---|---|
| | | | Cand | PE | |
| 1 | 64 | 64 | 57.8 | 57.8 | 1.00 |
| 2 | 40 | 80 | 58.8 | 85.0 | 1.38 |
| 3 | 12 | 36 | 47.2 | 91.7 | 1.55 |
| 4-5 | 13 | 58 | 41.4 | 92.3 | 2.00 |
| 6-15 | 13 | 125 | 50.4 | 100 | 4.85 |
| Total | 142 | 363 | 51.8 | 75.4 | 1.76 |

**Table 1. Classification results for the base classifier (at 4 FPs per patient), by PE degree**

## 4. MODIFICATIONS TO THE BOOSTING ALGORITHM

This section describes two modifications to boosted trees: modification of the dependent variable for positive candidates and enhancements to the set of predictor variables at selected boosting steps based on the results up to that point. We present cross validation results to illustrate the impact of these two innovations.

### 4.1 Focusing the Classifier on Low Degree PEs

To try to focus the classifier on detecting candidates from low degree PEs, we modified the standard dependent variable for the regression trees at each boosting step. For positive candidates only, we replaced $y_i = +1$ with $y_i = +1/(\text{PE degree})$. Note that this new dependent variable required no changes to our code that implements Real AdaBoost for regression trees.

One motivation for the form of this proposal is based on the characterization by Friedman et al. [3] of boosting as additive logistic regression with loss function $L(y, H(x)) = e^{-yH(x)}$. Rewriting $e^{-yH(x)}$ as $e^{-|y| \; sgn(y) \; H(x)}$ suggests that $|y_i|$ acts something like a weight in boosting. In that sense, the modified dependent variable defined above adjusts the data to give each PE the same weight. The hope was to improve candidate sensitivity for PEs of degree one or two, even at the risk of worse candidate sensitivity for high degree PEs. Of course, there was no guarantee that information in the predictor variables could discriminate candidates in low degree PEs from one in high degree PEs.

Figure 1 shows cross validated PE sensitivity versus the number of FPs per patient as the cutoffs are varied for the base classifier (solid line) and the classifier with the modified dependent variable (dashed line). This simple change to the dependent variable generally improved cross validated PE sensitivity by about 5 to 7 percentage points over much of the target range for Task 1.

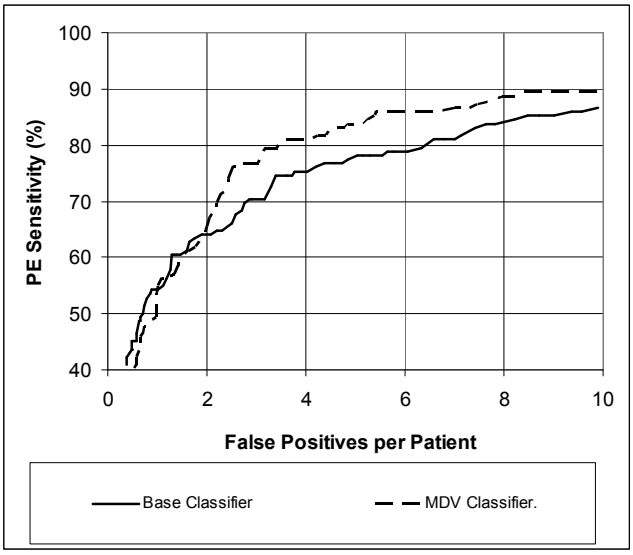


**Figure 1. Cross validated Task 1 performance for base classifier and MDV classifier**

Table 2 compares classification results, by PE degree, for the modified dependent variable (MDV) classifier with that for the base classifier (BC), with both constrained to have exactly 4.0 FPs

| PE Degree | Number of PEs | Cand Sensitivity | | PE Sensitivity | |
|---|---|---|---|---|---|
| | | BC | MDV | BC | MDV |
| 1 | 64 | 57.8 | 73.4 | 57.8 | 73.4 |
| 2 | 40 | 58.8 | 51.3 | 85.0 | 80.0 |
| 3 | 12 | 47.2 | 38.9 | 91.7 | 91.7 |
| 4-5 | 13 | 41.4 | 36.2 | 92.3 | 92.3 |
| 6-15 | 13 | 50.4 | 43.2 | 100.0 | 100.0 |
| Total | 142 | 51.8 | 48.8 | 75.4 | 81.0 |

**Table 2. Comparison of classification results for the base classifier (BC) and classifier with modified dependent variable (MDV), at 4 FPs per patient, by PE degree**

anticipated. Sensitivity improved dramatically for PEs of degree 1; the MDV classifier correctly detected 47 of 64 positives, compared with 37 of 64 for the base classifier. While candidate sensitivity declined noticeably for all higher PE degrees, the adverse impact on PE sensitivity was small, resulting in non detection of only two additional PEs of degree two. Overall, while candidate sensitivity declined by 3.0 percentage points, PE sensitivity rose by 5.6 points.

## 4.2 Enhancing the Set of Predictor Variables

The two classifiers described above treat the observations as if they are independent. We did not provide them with any information to indicate which candidates were from the same patient or which positive candidates shared the same PE. Consequently, the MDV classifier's ability to discriminate between low degree and high degree PEs appears to have derived purely from characteristics of individual candidates. We do note that many of the most important predictor variables associated with individual candidates bear names like "Shape neighbor feature N", so that they may actually contain information useful about nearby sites.

There is substantial structure in the make up of PEs that might be exploited. For example, among pairs of positive candidate sites from the same patient, at least 75 percent of those within 35 units of each other (based on Euclidean distance) came from the same PE, while fewer than 25 percent did for pairs separated by 75 or more units.

Given appropriate data about each candidate's neighbors, if any exist, a classifier might be better able to discriminate between candidates from low degree and high degree PEs. For example, the MDV classifier might learn that it is crucial to correctly classify positive candidates without any neighbors that appear to be positive, but less important to do so for candidates with one or more close neighbors that are already being classified as positive.

To improve the opportunity for the model to distinguish between high and low degree PEs, we augmented the list of explanatory variables at selected boosting steps ($T$ = 33, 36, 39, … , 60) with five new variables defined specifically at those steps. For neighborhoods of size 10, 20, 40, 60, and 100, we computed the difference between $H_{T\text{-}1}(x_i)$ for the candidate and the maximum value of the corresponding score for any other site within the neighborhood (this difference was set to a large positive constant for candidates with no neighbor). Negative values of these differences suggest candidates that are either negative or, if positive, are members of multiple degree PEs. In contrast, candidates from degree 1 PEs are likely to have positive differences scores.

When included in the list of predictor variables, one of these five new variables almost always defined the first split in the tree, and sometimes multiple splits. To avoid over fitting, we limited inclusion of these variables to later boosting steps and to every third tree.

Figure 2 compares cross validated results for this classifier (short dashed line) with those for the previous two classifiers. This enhancement appears to increase PE sensitivity by a couple percentage points on average for FP rates of 5 or higher. Because of this improvement, we used the MDV classifier with the enhanced set of predictor variables in our submission for Task 1.

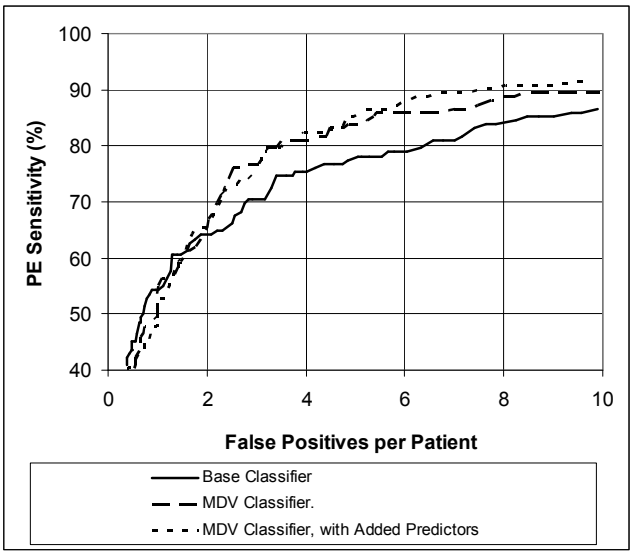


**Figure 2. Cross validated Task 1 performance for base classifier, MDV classifier, and MDV classifier with added predictors**

## 4.3 The Variable Modified Dependent Variable Classifier

In this subsection, we describe a promising alternative classifier, even though we ultimately did not use it as part of our KDD Cup submission.

Although the MDV classifier did very well at shifting the focus towards candidates from low degree PEs, there might still be gain available from trying to focus a classifier on one or two candidates from high degree PEs. Classifying four or five sites in close proximity is unlikely to increase PE sensitivity compared with picking the one or two "best" sites. Consequently, instead of treating all members of a multiple degree PE interchangeably, as the MDV classifier does, it might help to try to identify one candidate, or a few, for training a classifier. For multiple instance learning, Viola *et al.* [7] suggested that classification performance can be improved by focusing on a few critical examples in a "bag" (PE, for our purposes).

With this issue in mind, we tried to redistribute the +1 shared across values of the dependent variable for a PE, with more going at time $T$ to candidates with larger values of the current score, $H_{T\text{-}1}$ (e.g., a PE with degree 3 might have outcomes of 0.7, 0.2, and 0.1). Specifically, if cases 1 to $k$ were members of the same PE, we created a *variable* modified dependent variable (VMDV)

$$y_i = e^{aH_{T-1}(x_i)} \Big/ \sum_{j=1}^{k} e^{aH_{T-1}(x_j)}$$

for $a$ = ½. As $T$ grows, the values of $y_i$ for a single PE tend to diverge from $1/k$.

This classifier involves a tradeoff. While focusing weight on few cases from any particular PE may tend to reduce redundant positive classifications, it also risks decreasing the effective sample size available for learning how to discriminate positives from negatives.

Figure 3 compares results for the VMDV classifier (solid line) against those for the MDV classifier—with both using the same enhanced set of predictor variables. Overall, results were quite similar. However, there is mild evidence of a cross over near 5 FPs per patient, with VMDV doing better for low values of the FP rate and MDV doing better for high values. This suggests that VMDV may have the most promise when reducing FPs to a minimum is critically important. In contrast, VMDV may lose too much power when the limit on FPs is less stringent.

Because the VMDV classifier did not demonstrate substantial improvement at any point on the curve, we used the classifier described in Section 4.2 in our submission for Task 1.

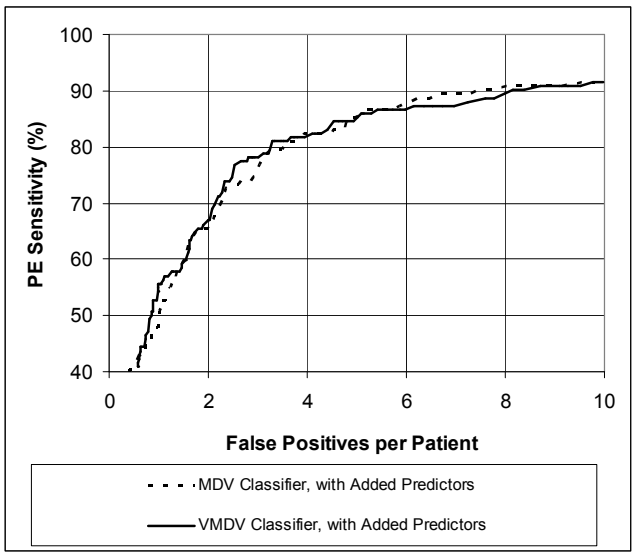


**Figure 3. Cross validated Task 1 performance for the MDV and VMDV classifiers, both with added predictors**

## 5. SETTING THRESHOLDS

To decide how many candidates to take from the ordered list created above, we used 46-fold cross validation on the training data to compute nearly unbiased estimates for FP probabilities as a function of the boosted scores (using the base classifier). We fit a logistic regression of PE status on a natural cubic spline with 4 knots [4] of the boosted score excluding the same patient.

To obtain FP probability estimates for the test data, we applied the base classifier to the test data and then used the logistic regression described above to compute an estimated FP probability, $p_i$, for each candidate in the test data. Summing these estimated probabilities $\{p_i\}$ yields estimates for the expected number of cumulative FPs at any point in the ordered list.

Table 3 illustrates this process for a subset of candidates from the test data set. The rows of the table are ordered by decreasing values of the boosted scores, using the classifier described in Subsection 4.2. Note that the values of $p_i$ do not increase monotonically, because those values are based on the base classifier. The column labeled "Cum. $p_i$" estimates the cumulative number of FPs through a particular candidate on the ordered list.

Similarly, summing estimated values of the nominal variances, $\{p_i(1-p_i)\}$, yields nominal estimates for the variances of these cumulative numbers of FPs under the assumption of independence. However, our analysis found intra patient correlation in PE status that was not explained by the base classifier. Near the threshold for Task 1, we estimated that the true variance for the cumulative number of FPs was about twice the nominal variance. The last two columns of Table 3 show "corrected" variances and standard deviations that account to the intra patient correlation.

For Task 1a, the disqualification limit was 2.0 FPs per patient, or 46 across the 23 patients in the test data. However, as the last column of Table 3 makes clear, we needed to aim well below 46 FPs in order to have a high probability of passing this task. We ended up setting a threshold at 40.00; that is, we classified candidates with scores of -0.56 or higher as positive, and all others as negative. This threshold corresponded to aiming about 1.0 standard deviations below the target number of FPs for Task 1. For Tasks 1b and 1c, we set cut points corresponding to 3.51 and 9.34 FPs per patient, both of which corresponded to approximately 1.5 estimated standard deviations below the subtask targets. We chose those thresholds based on a goal of qualifying on all three subtasks with probability about 0.75 (ignoring bootstrapping).

| Boosted Score | $p_i$ | Cum. $p_i$ | Cumulative Var | | Corrected SD |
|---|---|---|---|---|---|
| | | | Nom. | Corr. | |
| 1.29 | .09 | .09 | .08 | | |
| 1.19 | .12 | .22 | .19 | | |
| 1.12 | 20 | .41 | .35 | | |
| . | . | . | . | | |
| . | . | . | . | | |
| -.52 | .36 | 38.66 | 18.04 | 36.08 | 6.01 |
| -.55 | .69 | 39.36 | 18.25 | 36.50 | 6.04 |
| -.56 | .64 | 39.99 | 18.49 | 36.98 | 6.08 |
| -.60 | .50 | 40.49 | 18.74 | 37.48 | 6.12 |
| -.61 | .49 | 40.98 | 18.99 | 37.98 | 6.16 |
| -.67 | .58 | 41.56 | 19.23 | 38.46 | 6.20 |
| -.67 | .76 | 42.31 | 19.41 | 38.82 | 6.23 |
| -.68 | .73 | 43.04 | 19.61 | 39.22 | 6.26 |

**Table 3. Illustration of the process for setting the threshold for Task 1a**

With hindsight, it is clear that the use of bootstrap samples for scoring substantially increased the variance of the number of FPs at any cut point beyond that associated with the random sampling of 23 test patients. Although disqualification on any particular bootstrap sample was no longer fatal, the penalty was still quite severe. Consequently, qualifying for a very high proportion of bootstrap samples was needed to score well on Task 1, so even more conservative cut points might have been appropriate.

## 6. TASK 2

In theory, all the methods developed and tested for Task 1 transferred seamlessly to Task 2, where the goal was to correctly classify at least one positive site for each *patient* with one or more PEs. The modified dependent variable for positive PEs became +1 / (# of positive candidates for the patient). For the enhanced set of predictor variables, we dropped the neighborhood of size 10 and added one covering the whole patient.

Our results for Task 2 were substantially worse than for Task 1. One reason may be that we did not conduct any model evaluations specific to this task. In practice, discrimination between patients with few versus many positive sites may be a lot different that discrimination between PEs with low versus high degree. Also, because we had not tuned our models at all for Task 2, we set more aggressive thresholds of: 1.82, 3.67, and 9.65 FPs per patient. As noted above, this adjustment was probably directly opposite what we should have done.

## 7. ACKNOWLEDGMENTS

We thank R. Bharat Rao and Siemens Medical Solutions for providing the data and Terran Lane for orchestrating this fascinating and challenging problem.

## 8. REFERENCES

[1] Breiman, L. Arcing classifiers (with discussion and rejoinder). *Annals Statist.* 26 (1998), 801-849.

[2] Friedman, J. H. Greedy function approximation: a gradient boosting machine, *Annals Statist.* 29 (2001), 1189-1232.

[3] Friedman, J., Hastie, T., and Tibshirani, R. Additive logistic regression: a statistical view of boosting (with discussion and rejoinder). *Annals Statist.* 28 (2000), 337-407.

[4] Hastie, T., Tibshirani, R., and Friedman, J. *The Elements of Statistical Learning.* Springer, New York, 2001.

[5] KDD Cup 2006. Data description, task specification, rules. http://www.cs.unm.edu/files/kdd-cup-2006-task-spec-v1.pdf

[6] Schapire, R. E., and Singer, Y. Improved boosting algorithms using confidence-rated predictions. *Machine Learning* 37*,* (1999), 297-336.

[7] Viola, P., Platt, J, and Zhang, C. Multiple instance boosting for object detection. *NIPS* 18, 2006.