SENSITIVE ERROR CORRECTING OUTPUT CODES

Abstract. Sensitive error correcting output codes are a reduction from cost sensitive classification to binary classification. They are a modification of error correcting output codes [3] which satisfy an additional property: ϵ regret for binary classification implies at most 2ϵ l_2 regret for cost-estimation. This has a number of important implications:

1) Any 0/1 regret minimizing online algorithm is (via the reduction) a regret minimizing online cost sensitive algorithm. In particular, this means that online learning can be made to work for arbitrary (i.e. totally unstructured) loss functions.

2) The output of the reduction can be thresholded so ϵ regret for binary classification implies at most $4\sqrt{\epsilon}$ regret for cost sensitive classificaiton.

3) Using the canonical embeding of multiclass classifcation into cost sensitive classification, this reduction implies that ϵ binary regret implies at most 2ϵ l_2 error in the estimation of class probabilities. When thresholded, this implies at most $4\sqrt{\epsilon}$ multiclass regret.

1. Introduction

Background: The goal of classification is to predict labels on test examples given a set of labeled training examples. Binary classification, where the number of labels is two, is the most basic learning task as it involves predicting just a single bit for each example. Due to this simplicity, binary learning is (perhaps) better understood theoretically than any other prediction task, and several empirically good binary learning algorithms exist.

Practical machine learning, on the other hand, often requires predicting more than one bit. Furthermore (in the most general case) each prediction may have a different associated loss. For example, consider the commuting problem: Given information in the form of the current weather and radio reports, we must predict which of several routes to take. The goal in this task is to minimize the expected time of travel. We could think of this as a multiclass problem (predict which route is fastest), but this can make the problem more difficult[1] due to throwing away information.

General Warning: Attempts to understand the contents of this paper in terms of sample complexity results such as PAC learning [9] or uniform convergence [11] will fail. This is an orthogonal theory where we define leaning problems as measures over certain sets and analyze the transformation of losses under mappings between these measures.

Motivation: Reductions allow us to translate performance on well-studied binary problems into performance on the more general problems arising in practice.

[1]For example, it's easy to design a problem where there are several paths that are good except they are occasionally very slow. If there is another path which is never the best but never slow it may provide the best expected time of all choices. If this is altered into a multiclass problem of "predict the best path", the "never the best but never slow" option will never be the correct label.

We provide a reduction (SECOC) from cost sensitive classification to binary classification with the property that small regret on the created binary problem(s) implies small regret on the original cost sensitive classification problem. This is particularly compelling because *any* loss function on single examples can be expressed with cost sensitive classification. Therefore, this reduction can be used (at least theoretically) to solve a very broad set of learning problems. In addition, there is significant empirical evidence (see [7, 2, 12]) that analysis in this style results in algorithms providing good performance.

The basic SECOC reduction can be reused in several ways.

(1) General Online Learning: Any 0/1 regret minimizing online algorithm is (via the reduction) a regret minimizing online cost sensitive algorithm. In particular, this means that online learning can be made to work for arbitrary (i.e. totally unstructured) loss functions.
(2) General Regret Minimizing Boosting: Since the binary learning algorithm might be a version of boosting which can boost to the regret to 0 (as in [6]), the composition of boosting with SECOC can boost to zero regret on arbitrary (i.e. totally unstructured) loss functions.
(3) Cost Sensitive Classification: The output of the reduction can be thresholded so a small regret for binary classification implies a small regret for cost sensitive classification. This implies that any consistent binary classifier is a consistent cost sensitive classifier.
(4) Multiclass Probability Estimation: Using the canonical embeding of multiclass classifcation into cost sensitive classification, this reduction implies that small binary regret implies small l_2 error in the estimation of class probabilities.
(5) Multiclass classification: When the multiclass probability estimates are thresholded, this implies a small multiclass regret. Note that this implies *any* consistent binary classifier is (via the reduction) a consistent multiclass classifier. This is particularly important because generalization of SVMs to multiple classes [13] have been done wrong.

Context: SECOC is a variation of error-correcting output codes (ECOC) [3]. Later in section 5.1 we show that this variation is necessary in order to satisfy a regret transform. The ECOC reduction works by learning a binary classifier, which decides membership of a label amongst subsets of labels. Given a sequence of subsets, each label corresponds to a binary string (or a *codeword*) defined by the inclusion of this label in the sequence of subsets. A multiclass prediction is made by finding the codeword closest in Hamming distance to the sequence of binary predictions on the test example.

For the ECOC reduction, a basic statement [5] can be made: with a good code, for all training sets, the error rate of the multiclass classifier on the training set is at most 4 times the error rate of the individual binary classifiers. The proof of this statement is essentially the observation that there exist codes where the distance between any two codewords is at least $\frac{1}{2}$. Consequently, at least $\frac{1}{4}$ of the classifiers must err to induce a multiclass classification error, implying the theorem via the pigeonhole principle.

This theorem can be generalized [2] to quantify “for all measures” rather than “for all training sets”. This generalization is not as significant as it might at first seem because the measure implicit in a very large training set can approximate other measures (and it can do so arbitrarily well when the feature space is finite). Nevertheless, it is convenient to quantify over all measures so the statement holds for the process generating each individual example. Since there is always some process generating examples, the result is applicable even to adversarial processes.

The weighted all pairs algorithm of [2] intuitively guarantees that a small error rate on created classification problems implies a small cost sensitive loss. The core result here is similar except that small binary *regret* implies small cost sensitive *regret*. Regret is the error rate minus the minimum error rate. Consequently, the results here can have important implications even when (for example) the binary error induced from a multiclass problem is 0.25. SECOC does not supercede this earlier result because the regret bounds are weaker, roughly according to ϵ error rate $\rightarrow \sqrt{\epsilon}$ regret.

ECOC was modified [1] to consider margins of the binary classifiers – numbers internal to some classification algorithms that provide a measure of confidence in a binary prediction. Decoding proceeds in the same way as for ECOC except a "loss"-based[2] distance is used instead of the Hamming distance. Roughly speaking, SECOC uses the motivation behind this approach although not the approach itself. Instead of working with margins, we define binary classification problems for which the optimal solution computes the expected cost (rather than the margin) of choices. This approach allows us to accomplish several things:

(1) We can use arbitrary classifiers rather than margin-based classifiers.
(2) We remove the mismatch between the margin and the motivations. Optimizations of hinge loss (for SVMs) or exponential loss (for Adaboost) cause a distortion where the optimization increases the margin of small-margin examples at the expense of the margin on large-margin examples. The efficacy of Platt scaling [8] (i.e., fitting a sigmoid to a margin to get a probabilistic prediction) can be thought of as a strong empirical evidence of the deficiency of margins as probability estimates.
(3) We can generalize the approach to tackle all cost sensitive problems rather than just multiclass problems. This generalization comes at no cost in multiclass performance.

OUTLINE: The next sections of this paper do the following:

(1) We define the basic SECOC reduction.
(2) We define and then prove the main theorem.
(3) We prove corollaries of the main theorem for different applications.
(4) We prove some structural lower bounds: that ECOC can not achieve the regret transform we prove and that no tighter bound can be proved for SECOC.

2. The SECOC Reduction

We work in an assumption-free learning setting. To define the reduction, we first define the problems that we reduce to and from. We will actually reduce to importance weighted binary classification. Since importance weighted binary

[2]"Loss" is in quotes because the notion of loss is a specification of the optimization used by the binary learning algorithm rather than the loss given by the problem, as is used in the rest of the paper.

Algorithm 1 SECOC-train (Set of k-class cost sensitive examples S, $n \times k$ binary matrix M, importance weighted binary classifier learning algorithm B)

(1) For each subset s defined by the rows of M:
  (a) For each t in $\left[\min_D \frac{\sum_{y\in s} c_y}{\sum_y c_y}, \max_D \frac{\sum_{y\in s} c_y}{\sum_y c_y}\right]$
    (i) Let $b_{st} = \mathrm{B}(\{(x,\ I(\sum_{y\in s} c_y \geq t\sum_y c_y), |\sum_{y\in s} c_y - t\sum_y c_y|) : (x, \vec{c}) \in S\})$
(2) return $\{b_{st}\}$

classification can be reduced to binary classification [12], a reduction all the way to binary classification is achievable.

Definition 2.1. (Importance Weighted Binary Classification) An Importance weighted binary classification problem is defined by a measure D on a set $X \times \{0,1\} \times [0,\infty)$, where X is some arbitrary feature space,$\{0,1\}$ is the correct binary label and $[0,\infty)$ is the importance of correct classification. The goal in solving binary classification is to find a binary classifer $b : X \to \{0,1\}$ which minimizes the expected importance weighted loss, $E_{(x,y,i)\sim D} iI(b(x) \neq y)$ where $I()$ is 1 when the argument true and 0 otherwise.

SECOC reduces from cost sensitive classification.

Definition 2.2. (Cost Sensitive Classification) A cost sensitive problem is defined by a measure D on a set $X \times [0,\infty)^k$, where X is some arbitrary feature space and the additional information $[0,\infty)^k$ is the cost of each choice. . The goal in solving cost sensitive classification is to find a multiclass classifer $m : X \to \{1,...,k\}$ which minimizes the expected cost, $E_{(x,\vec{c})\sim D} c_{m(x)}$.

A cost sensitive learning algorithm typically takes as input a sequence of training examples in $(X \times [0,\infty)^k)^*$ as advice in constructing $h(x)$. The SECOC reduction is a cost sensitive learning algorithm given a binary learning algorithm.

The SECOC reduction use a code, just as the ECOC reduction. An $n \times k$ binary coding matrix M with columns corresponding to multiclass labels defines the code. Each of n rows defines a binary classification problem by specifying a subset of labels and asking whether the label is in the subset. The columns of M form a subset of any k codewords of a Hadamard code of length n, which has the property that any two distinct codewords differ in at least $n/2$ bit positions. Such codes exist and are easy to construct when k is a power of 2. Thus, for Hadamard codes, the number n of classification problems needed is less than $2k$.

To define a SECOC, we create a set of importance weighted classification problems for each subset s of the labels. Since importance weighted classification can be reduced to binary classification [12] this reduction can actually be done all the way to binary classification. The particular importance weighted classification problems are defined so that from their optimal solution we can compute the expected cost of each subset.

To make a prediction, for each label the SECOC reduction adds up the expected cost of the subsets containing the label and outputs the label with the largest probability.

The training and prediction algorithms are given in Figure 1 and Figure 2.

Algorithm 2 SECOC-predict (Binary classifiers $\{b_{st}\}$, $n \times k$ binary matrix M, test example $x \in X$)

(1) For each label y return $2\left(E_s E_t I(l \in s) b_{st}(x) + I(l \notin s)(1 - b_{st}(x)\right) - 1$

It might appear that SECOC-train requires several invocations of the oracle learning algorithm A, but a standard trick [2, 1] allows this to be collapsed to one classifier. The trick is to augment the feature space with the name of the call, $X \to X \times \{0,1\}^k \times [0,1]$, and then learn a classifier c on (a random subset of) the union of all training data. With this classifier, we can define $h_{st}(x) \equiv h(x, s, t)$, and all of our results hold for this single invocation classifier.

The implication of this observation is that we can regard SECOC-train as a machine which maps cost sensitive samples to importance weighted binary samples. Thus, we can regard SECOC-train as having type SECOC-train : $(X \times [0,\infty)^k)^* \to (X' \times \{0,1\} \times [0,\infty))^*$, and the learned binary classifier is simply $B(\text{SECOC-train}(S))$.

3. The Main Theorem

Before stating the theorem, we define exactly what loss and regret means. Given any distribution D on examples $X \times \{0,1\} \times [0,\infty)$, the importance weighted error rate of a binary classifier b is given by:

$$e(D, b) = E_{x,y,i \sim D} i I(b(x) \neq y)$$

Similarly, given any distribution D on examples $X \times [0,\infty)^k$, the cost sensitive loss of a multiclass classifier m is given by:

$$e(D, m) = E_{(x,\vec{c}) \sim D} c_{m(x)}$$

For each of these notions of loss, the regret is the difference between the achieved performance and best possible performance:

$$r(D, h) = e(D, h) - \min_{h'} e(D, h')$$

(Note that we mean the minimum over *all* h' not over some subset.)

We also must define exactly what the induced distribution on the binary classifier is. To do this, we simply define how to draw one sample from the induced distribution. To draw from the induced distribution, first draw a cost sensitive sample $(x, \vec{c})$ from D. Then apply SECOC-train to the singleton set $\{(x, y)\}$ to get a sequence of importance weighted binary examples $(x', y', i')^*$. Now, to draw from D', sample uniformly from this set. We overload and denote the induced distribution as SECOC-train(D).

Theorem 3.1. *(SECOC Regret Transform) For all distributions D over cost sensitive test examples $(x, \vec{c})$, for all importance weighted binary learning algorithms B and multiclass datasets S, let $b = B(\text{SECOC-train}(S))$. The* SECOC *reduction satisfies:*

$$\forall y \in \{1, ..., k\} : \; E_{(x,\vec{c}) \sim D}(\textit{SECOC-predict}(b, x, y) - \frac{E_{\vec{c}|x} c_y}{E_{\vec{c}|x} \sum_y c_y})^2 \leq 4 r(\textit{SECOC-train}(D), b)$$

For the proof, first note that the dependence on A and S can be removed by proving the theorem $\forall b$ (this is of equivalent generality).

Proof. We first analyze what happens when no regret is suffered, and then analyze the case with regret. For any choice of s and t optimal classifier is given by:

$$b^*_{st} = \arg\min_b E_{x,y,i\sim D_{st}} iI(b(x) \neq y)$$
$$= \arg\min_b E_{x,\vec{c}\sim D} |\sum_{y\in s} c_y - t\sum_y c_y| I(b(x) \neq I(\sum_{y\in s} c_y \geq t\sum_y c_y))$$

for any x, the optimal value of $b(x)$ is either 0 or 1. When it's 0, the expected cost is

$$= E_{\vec{c}\sim D|x}(\sum_{y\in s} c_y - t\sum_y c_y) I(\sum_{y\in s} c_y \geq t\sum_y c_y) \tag{3.1}$$

and when it's 1, the expected cost is

$$= E_{\vec{c}\sim D|x}(t\sum_y c_y - \sum_{y\in s} c_y) I(\sum_{y\in s} c_y < t\sum_y c_y) \tag{3.2}$$

. The first equation is continuous and decreases monotonically with t while the second increases monotonically, so we need only find the (single) equality point to describe the optimal behavior for all t. The equality point is given by:

$$E_{\vec{c}\sim D|x}(\sum_{y\in s} c_y - t\sum_y c_y) I(\sum_{y\in s} c_y \geq t\sum_y c_y) = E_{\vec{c}\sim D|x}(t\sum_y c_y - \sum_{y\in s} c_y) I(\sum_{y\in s} c_y < t\sum_y c_y)$$
$$\Rightarrow E_{\vec{c}\sim D|x}(\sum_{y\in s} c_y - t\sum_y c_y) = 0$$
$$\Rightarrow t = \frac{E_{\vec{c}\sim D|x}\sum_{y\in s} c_y}{E_{\vec{c}\sim D|x}\sum_y c_y}$$

so, $b^*_{st}(x) = I(E_{\vec{c}\sim D|x}\sum_{y\in s} c_y \geq tE_{\vec{c}\sim D|x}\sum_y c_y)$.

For any choice of s, the value of

$$E_t b^*_{st}(x) = E_t I\left(E_{\vec{c}\sim D|x}\sum_{y\in s} c_y \geq tE_{\vec{c}\sim D|x}\sum_y c_y\right)$$
$$= \frac{E_{\vec{c}\sim D|x}\sum_{y\in s} c_y}{E_{\vec{c}\sim D|x}\sum_y c_y}$$

since the $E_{t\in[0,1]} I(K \geq t) = K$ for all $K \in [0,1]$.

Since decoding is symmetric with respect to all labels, so we need analyze only one label y. Furthermore, since step 1 in Figure 2 is symmetric with respect to set inclusion or complement set inclusion, we can assume that y is in every subset. (In other words, for any subset s not containing y, we can create another code M' which is M except that s is replaced with the complement of s. M' decodes as M according to 1.) Consequently,

$$\hat{c}_l = E_s E_t I(l \in s) b_{st}(x) = E_s \frac{E_{\vec{c}\sim D|x}\sum_{y\in s} c_y}{E_{\vec{c}\sim D|x}\sum_y c_y} = \frac{E_{\vec{c}\sim D|x} E_s \sum_{y\in s} c_y}{E_{\vec{c}\sim D|x}\sum_y c_y}$$

For any other label y', half the time y' is in s and half the time it is not in s, implying:

$$= \frac{E_{\vec{c}\sim D|x} c_y + \frac{1}{2}\sum_{y'\neq y} c_{y'}}{E_{\vec{c}\sim D|x}\sum_y c_y} = \frac{1}{2}\left(\frac{E_{\vec{c}\sim D|x} c_y}{E_{\vec{c}\sim D|x}\sum_y c_y} + 1\right)$$

Consequently, step 1 outputs for each y, $\frac{E_{\vec{c}\sim D|x} c_y}{E_{\vec{c}\sim D|x} \sum_y c_y}$ when the classifiers are optimal.

Now we analyze the regret transformation properties. An observation simplifies this analysis. First, since the importance weighted loss scales with $\sum_y c_y$, we can normalize according to $\sum_y c_y = 1$. Next, examining equations 3.1 and 3.2, we see that (on one hand) the importance weighted loss grows linearly with the distance of t from $E_{\vec{c}\sim D|x} \sum_y c_y$, but on the other hand, each error has equal value in disturbing the expectation in line 1 of Figure 2. There are two consequences for an adversary attempting to disturb the expectation the most while paying the least importance weighted cost.

1) It is "cheapest" for an adversary to err on the t closest to $E_{\vec{c}\sim D|x} \sum_{y\in s} c_y$ first. (The cost an adversary pays in importance weighted loss can be monotonically decreased by swapping errors for large values of $|t - E_{\vec{c}\sim D|x} \sum_{y\in s} c_y|$ with small values.)

2) It is "cheapest" to have a small equal disturbance for each s rather than a large disturbance for a single s. (The cost an adversary pays for disturbing the overall expectation can be monotonically decreased by spreading errors uniformly over s.)

Consequently, the optimal strategy for an adversary wanting to disturb the expectation by Δ is to disturb the expectation for each s by Δ. The importance weighted regret of erring (with a "1") for $t = \Delta + \frac{E_{\vec{c}\sim D|x} \sum_{y\in s} c_y}{E_{\vec{c}\sim D|x} \sum_y c_y}$ can be found by subtracting equation 3.2 from equation equation 3.1.

$$E_{\vec{c}\sim D|x}(t \sum_y c_y - \sum_{y\in s} c_y) I(\sum_{y\in s} c_y < t \sum_y c_y) - E_{\vec{c}\sim D|x}(\sum_{y\in s} c_y - t \sum_y c_y) I(\sum_{y\in s} c_y \geq t \sum_y c_y)$$

$$= E_{\vec{c}\sim D|x} \left(\left(\Delta + \frac{E_{\vec{c}\sim D|x} \sum_{y\in s} c_y}{E_{\vec{c}\sim D|x} \sum_y c_y} \right) \sum_y c_y - \sum_{y\in s} c_y \right)$$

$$= \Delta E_{\vec{c}\sim D|x} \sum_y c_y$$

The same quantity holds for $t = -\Delta + \frac{E_{\vec{c}\sim D|x} \sum_{y\in s} c_y}{E_{\vec{c}\sim D|x} \sum_y c_y}$. The minimum expected importance weighted regret suffered by an adversary disturbing the quantity by Δ is therefore:

$$\int_{z=0}^{z=\Delta} z dz E_{\vec{c}\sim D|x} \sum_y c_y = \Delta^2 E_{\vec{c}\sim D|x} \sum_y c_y$$

using observation (2), this implies that the expectation over s is disturbed by at most Δ with the same (average) budget of importance weighted errors. Consequently, the predicted value for the relative cost is disturbed by at most 2Δ using the equation from line 1 of Figure 2. □

Since one of the rows can be made an all-one row (with membership probability 1), we actually have a slightly tighter bound of

$$\frac{n}{2}(p_l(x) + 1) + (n-1)\sqrt{2\epsilon}.$$

4. Corollaries

4.1. **Cost Sensitive Classification.**

4.2. **Multiclass Probability Estimation.**

4.3. **Multiclass Classification.** To prove the corollary, suppose the correct label has probability p_l and some other label has probability $p_{l'} < p_l$. In order to cause an error, the probability estimate $\hat{p}_l$ for the correct must be less than the probably estimate $\hat{p}_{l'}$. Again, due to the concavity of the square root, the most efficient way to create an error is for $\hat{p}_l = \frac{p_{l'}+p_l}{2} \simeq \hat{p}_{l'}$, implying that the adversary must pay the regret necessary to induce a probabilistic error of size $\frac{p_l - p_{l'}}{2}$. By theorem **??**, this is $\epsilon = \frac{1}{8}\left(\frac{p_l - p_{l'}}{2}\right)^2$. On the other hand, the amount of multiclass regret induced by choosing l' rather than l is just $p_l - p_{l'}$. Solving for $p_l - p_{l'}$ finishes the corollary.

4.4. **Use in Online Learning.**

5. Structural Lower Bounds

Is it possible to do the same with these other approaches? The answer is no, and the proofs are actually quite simple.

5.1. **Inconsistency of ECOC.** Intuitively, what we mean by inconsistency here is that even using an optimal binary classifier, the reduction fails to provide an optimal multiclass classifier.

Theorem 5.1. *(ECOC Inconsistency) For all $k > 2$, there exists a distributions D over multiclass test examples (x, y), such that for all codes M, with $c^* = \arg\min_c r(ECOC\text{-}train(D), c)$*

$$r(D, ECOC\text{-}test(c^*)) > \frac{1}{10}$$

Proof. The proof is constructive. We choose a D which places probability on 3 labels, "1", "2", and "3".

A few observations about symmetry greatly simplify the proof. The first observation is that since only 3 labels have probability we can rewrite any code M as a new *weighted* code M' over the 3 labels where each subset has a weight w_s=the number of times that subset of the 3 labels exists in M after projection. The second observation is that the symmetry with respect to complementarity implies each row (and each code word) has one '1' in it.

These observations imply that ECOC essentially uses the binary classifier to ask "is the probability of label $i > 0.5$?" for each $i \in \{1, 2, 3\}$. These answers are then combined with a weighted sum. If we let the probability of one label be $0.5 - \epsilon$ and the probability of the other labels be $0.25 + \frac{\epsilon}{2}$, the answer to every question will be "no" (i.e. 0).

Since we have a weighted sum, the exact weighting determines the outcome (possibly with randomization to break ties). The exact distribution therefore picks a label at random to have probability $0.5 - \epsilon$, encodes that choice in the x value, and then draws the label at random.

Under any code, the probability of predicting the label with greatest probability is at most $\frac{1}{3}$ implying a regret of $\frac{2}{3}[0.5 - \epsilon - (0.25 + \frac{\epsilon}{2})]$ can be made arbitrarily close to $\frac{1}{6}$. $\square$

MarginECOC also allows "don't cares" in the code matrix. The existence of "don't cares" allows questions of the form "is the probability of this label greater than that label?" which are (in general) sufficiently powerful to support regret transformation consistency. However, it is unclear how to get a good regret transform.

Take as an example, the all-pairs code, which is consistent. The all-pairs code creates a classifier for every pair which (for an optimal classifier) decides "Is class i more probable than class j?" The problem with this question is that the classifier is applied when class i and class j each have 0 probability. In this situation, an adversarial optimal classifier can freely choose to report either $p_i > p_j$ or $p_j > p_i$. Consequently, an adversarial optimal binary classifier could make some label with 0 probability and "beat" all labels except for one label with probability 1. Needless to say, this is not robust, because one error in one classifier can alter the result. This, is a counterexample to the main theorem of the paper.

5.2. SECOC can not be improved.

6. Discussion

6.1. Practical Matters.

The above algorithm provides nice theoretical guarantees, but has some obvious deficiencies. These deficiencies can be addressed without significantly weakening the theoretical guarantees.

1. When k is very large, we can instead use a code defined by picking a random subset of $O(\frac{1}{\epsilon^2}\log\frac{2k}{\delta})$ codewords. Hoeffding bounds imply that the estimates $\hat{p}[l]$ vary by at most ϵ with probability $1-\delta$.
2. Learning intractability. Intuitively, learning to distinguish a random subset may be significantly harder than learning to distinguish (say) one label from all other labels. This intuition is born out by experiments [1]. This observation suggest that PECOC should be applied on sparse codes (i.e. codes where each classifier learns to predict a small subset). The exact choice of code is a subtle affair not fully addressed here.

6.2. Optimal Codes.

References

[1] Erin Allwein, Robert Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *Journal of Machine Learning Research*, 1:113–141, 2000.

[2] Alina Beygelzimer, Varsha Dani, Tom Hayes, and John Langford. Reductions between classification tasks. *Electronic Colloquium on Computational Complexity*, TR04-077, 2004.

[3] Thomas Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[4] Yoav Freund and Robert Schapire. A decision-theoretic generalization of online learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1), 119–139, 1997.

[5] Venkat Guruswami and Amit Sahai. Multiclass learning, boosting, and error-correcting codes. In *Proceedings of the 12th Annual Conference on Computational Learning Theory (COLT)*, 145–155,1999.

[6] Kalai & Servidio

[7] John Langford and Bianca Zadrozny. Estimating class membership probabilities using classifier learners. In *Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, 2005.

[8] John Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, 61–74, 1999.

[9] Valiant
[10] Vladimir Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.
[11] VC
[12] Costing
[13] Wahba