Extreme Multi Class Classification

Anna Choromanska
Department of Electrical Engineering
Columbia University
aec2163@columbia.edu

Alekh Agarwal
Microsoft Research
New York, NY, USA
alekha@microsoft.com

John Langford
Microsoft Research
New York, NY, USA
jcl@microsoft.com

Abstract

We consider the multi class classification problem under the setting where the number of labels is very large and hence it is very desirable to efficiently achieve train and test running times which are logarithmic in the label complexity. Additionally the labels are feature dependent in our setting. We propose a reduction of this problem to a set of binary regression problems organized in a tree structure and we introduce a simple top-down criterion for purification of labels that allows for gradient descent style optimization. Furthermore we prove that maximizing the proposed objective function (splitting criterion) leads simultaneously to pure and balanced splits. We use the entropy of the tree leafs, a standard measure used in decision trees, to measure the quality of obtained tree and we show an upper-bound on the number of splits required to reduce this measure below threshold ϵ. Finally we empirically show that the splits recovered by our algorithm leads to significantly smaller error than random splits.

1 Introduction

The central problem of this paper is computational complexity in a setting where the number of classes k for multiclass prediction is very large. Almost all machine learning algorithms (with the notable exception of decision trees) have running times for multiclass classification which are $O(k)$ with a canonical example being one-against-all classifiers Rifkin and Klautau (2004). In this setting, the most efficient approach that could be imagined has a running time of $\mathcal{O}(\log k)$ for both training and testing, while effectively using online learning algorithms to minimizes passes over the data.

A number of prior works have addressed aspects of this problem, which we review next. The Filter Tree Beygelzimer et al. (2009b) addresses consistent (and robust) multiclass classification, showing that it is possible in the statistical limit. A critical problem not addressed there is the choice of *partition*. In effect, it can only succeed when the chosen partition happens to be easy. The partition finding problem is addressed in the conditional probability tree Beygelzimer et al. (2009a), but that paper addresses conditional probability estimation. Conditional probability estimation can be converted into multiclass prediction, but doing so is not a logarithmic time operation. Futher work Bengio et al. (2010) addresses the partitioning problem by recursively applying spectral clustering on a confusion graph. This approach is $O(k)$ or worse at training time, making it intractable by our standards. Empirically, this approach has been found to sometimes lead to badly imbalanced splits Deng et al. (2011). Also another work by Weston et al. (2013) make use of the k-means hierarchical clustering (other distortion-minimizing method could also be used) to recover the label sets for a given partition though this work primarily adresses the problem of ranking a very large set of labels rather than the multiclass classification problem.

Decision trees are naturally structured to allow logarithmic time prediction. Traditional decision trees often have difficulties with a large number of classes because their splitting criteria is not well-suited to the large class setting. However, newer approaches Agarwal et al. (2013) have addressed this effectively. The approach we discuss here can be thought of as a decision tree algorithm which differs in the objective optimized, and in how that optimization is done. The different objective allows us to prove certain useful properties while the optimization uses an online learning algorithm which is queried and trained simultaneously as we pass over the data. This offers a tradeoff: our nodes are more powerful, potentially exponentially reducing the state of the model, while being somewhat more expensive to compute since they depend on more than one feature.

2 Splitting criterion

2.1 Formulation of the objective function

In this section we will introduce the splitting criterion that we use in every node of the tree to decide whether an example should be sent to the left or right child node. For notation simplicity consider the split done in the root. Let $\mathcal{X}$ denotes the input dataset. The objective function that we aim to maximize is given as follows

$$\Omega(n_r, k_r) = \sum_{x \in \mathcal{X}} \left[\frac{n_l}{k_t(x)} \left| \frac{k_t(x)}{n_t} - \frac{k_l(x)}{n_l} \right| + \frac{n_r}{k_t(x)} \left| \frac{k_t(x)}{n_t} - \frac{k_r(x)}{n_r} \right| \right]$$

where n_t is the total number of examples, n_r (resp. n_l) is the number of examples going to the right (resp. left) child. Notice that $n_t = n_r + n_l$. $k_t(x)$ is the total number of examples labeled in the same way as x, $k_r(x)$ (resp. $k_l(x)$) is the number of examples labeled in the same way as x that are going to the right (resp. left) child. Notice that $\forall_{x \in \mathcal{X}} k_t(x) = k_r(x) + k_l(x)$. Thus the objective function can be rewritten as

$$\Omega(n_r, k_r) = 2 \sum_{x \in \mathcal{X}} \frac{n_r}{k_t(x)} \left| \frac{k_t(x)}{n_t} - \frac{k_r(x)}{n_r} \right|$$

By the maximally balanced split we will understand the one for which $n_r = n_t - n_r$ and thus the same number of examples were directed to the left and right child nodes. By the maximally pure split we will understand the one for which $k_r(x) = 0$ or $k_r(x) = k_t(x)$ and thus there exists no two distinct data points with the same label that are in different child nodes. The proposed objective function has certain desirable properties which are captured in Lemma 1 and Lemma 2 and in the entire Section 3.

Lemma 1. *If there exists a maximally pure and balanced split, this split maximizes the objective.*

Proof. Let $\mathcal{X}_R$ be the set of data points in the right child node such that no data point in the left child node has the same label as any data point in $\mathcal{X}_R$ ($\mathcal{X}_L$ is defined in analogy). Let $\mathcal{X}_B$ be the set of data points such that their labels appear in both child nodes.

$$\Omega(n_r, k_r) = 2 \sum_{x \in \mathcal{X}} \frac{n_r}{k_t(x)} \left| \frac{k_t(x)}{n_t} - \frac{k_r(x)}{n_r} \right|$$

$$= 2 \sum_{x \in \mathcal{X}_R} \frac{n_r}{k_t(x)} \left| \frac{k_t(x)}{n_t} - \frac{k_r(x)}{n_r} \right| + 2 \sum_{x \in \mathcal{X}_L} \frac{n_r}{k_t(x)} \left| \frac{k_t(x)}{n_t} - \frac{k_r(x)}{n_r} \right| + 2 \sum_{x \in \mathcal{X}_B} \frac{n_r}{k_t(x)} \left| \frac{k_t(x)}{n_t} - \frac{k_r(x)}{n_r} \right|$$

$$= 2 \sum_{x \in \mathcal{X}_R} \frac{n_t - n_r}{n_t} + 2 \sum_{x \in \mathcal{X}_L} \frac{n_r}{n_t}$$

The last step comes from the fact that we consider maximally pure split thus $\forall_{x \in \mathcal{X}_R} k_r(x) = k_t(x)$, $\forall_{x \in \mathcal{X}_L} k_r(x) = 0$ and thus furthermore $\mathcal{X}_B$ must be an empty set (that eliminates the third term). We can further simplify as follows

$$2 \frac{n_r(n_t - n_r)}{n_t} + 2 \frac{n_r(n_t - n_r)}{n_t} = 4 \frac{n_r(n_t - n_r)}{n_t}$$

Since we are maximizing the objective, we set n_r to $n_r = n_l = \frac{1}{2}n_t$. That shows the split is also balanced. $\square$

Lemma 2. *In isolation, i.e. when n_r is fixed and under the condition that a maximally pure and balanced split exists, maximizing the objective recovers this split.*

Proof. Objective function is

$$\Omega(n_r, k_r) = 2\sum_{x\in\mathcal{X}} \frac{n_r}{k_t(x)}\left|\frac{k_t(x)}{n_t} - \frac{k_r(x)}{n_r}\right| = 2\sum_{x\in\mathcal{X}} \frac{1}{k_t(x)}\left|\frac{n_r k_t(x) - n_t k_r(x)}{n_t}\right|$$

$$= 2\sum_{x\in\mathcal{X}} \frac{1}{k_t(x)}\left|\frac{n_r k_t(x)}{n_t} - k_r(x)\right|$$

If n_r is fixed and one optimizes for k_r (notice $\forall_{x\in\mathcal{X}} k_r(x) \in< 0, k_t(x) >$) then either one has to set $\forall_{x\in\mathcal{X}} k_r(x) = 0$ or $k_r(x) = k_t(x)$. $\square$

3 Purity and balancing factors

In order to show some more interesting properties of the objective function we need to introduce more formal notation. Let k be the number of labels. Let $\mathcal{H}$ be the hypothesis class. Let π_i be the probability that randomly chosen data point from the dataset has label i. Consider hypothesis $h \in \mathcal{H}$ and denote $Pr(h(x) > 0|i)$ t be the probability that $h(x) > 0$ given that x has label i. We can then define pure and balanced splits as follows

Definition 1. *The hypothesis $h \in \mathcal{H}$ induces a pure split if*

$$\sum_{i=1}^{k} \pi_i \min(Pr(h(x) > 0|i), Pr(h(x) < 0|i)) \leq \delta.$$

Definition 2. *The hypothesis $h \in \mathcal{H}$ induces a balanced split if*

$$\exists_{c_1\in(0,1)} c \leq Pr(h(x) > 0) \leq 1 - c.$$

We will refer to $\sum_{i=1}^{k} \pi_i \min(Pr(h(x) > 0|i), Pr(h(x) < 0|i))$ as the purity factor as it determines how pure the split is and we will refer to $Pr(h(x) > 0$ as the balancing factor as it determines how balanced the split is. One can then express the objective function in the equivalent form as

$$\Omega(h) = 2\sum_{i=1}^{k} \pi_i \left[|P(h(x) > 0) - P(h(x) > 0|x \in \mathcal{X}_i))|\right]$$

We will now show that increasing the value of objective function leads to recovering the hypothesis that induces more balanced splits and also more pure splits.

3.1 Balancing factor

We want to show the relation between the balancing factor and the value of the objective function. In order to do that we will start from deriving an upper-bound on $\Omega(h)$, where $h \in \mathcal{H}$. For the ease of notation let $P_i = Pr(h(x) > 0|x \in \mathcal{X}_i)$. Thus

$$\Omega(h) = 2\sum_{i=1}^{k} \pi_i \left|P(h(x) > 0|x \in \mathcal{X}_i) - P(h(x) > 0)\right| = 2\sum_{i=1}^{k} \pi_i \left|P_i - \sum_{j=1}^{k} \pi_j P_j\right|,$$

where $\forall_{i=\{1,2,\dots,k\}} 0 \leq P_i \leq 1$. The objective $\Omega(h)$ is definitely maximized on the extremes of the $[0, 1]$ interval. The upper-bound on $\Omega(h)$ can be thus obtained by setting some of the P_i's to 1's and remaining ones to 0's. To be more precise, let

$$L_1 = \{i : i \in \{1, 2, \dots, k\}, P_i = 1\}$$

and

$$L_2 = \{i : i \in \{1,2,\dots,k\}, P_i = 0\}$$

We can then write that

$$\Omega(h) \leq 2\left[\sum_{i\in L_1}\pi_i(1-\sum_{j\in L_1}\pi_j) + \sum_{i\in L_2}\pi_i\sum_{j\in L_1}\pi_j\right] = 2\left[\sum_{i\in L_1}\pi_i - (\sum_{i\in L_1}\pi_i)^2 + (1-\sum_{i\in L_1}\pi_i)\sum_{i\in L_1}\pi_i\right]$$

$$= 4\left[\sum_{i\in L_1}\pi_i - (\sum_{i\in L_1}\pi_i)^2\right]$$

For the ease of notation let $p = Pr(h(x) > 0)$. Notice that $p = \sum_{i\in I_1}\pi_i$ thus

$$\Omega(h) \leq 4p(1-p) \Leftrightarrow 4p^2 - 4p + \Omega(h) \leq 0$$

Thus

$$p \in \left[\frac{1-\sqrt{1-\Omega(h)}}{2}, \frac{1+\sqrt{1-\Omega(h)}}{2}\right].$$

That yields the balancing factor c to be $c = \frac{1-\sqrt{1-\Omega(h)}}{2}$. Maximizing $\Omega(h)$ leads to narrowing the $[c, 1-c]$ interval around value $\frac{1}{2}$ which corresponds to the maximally balanced split. In particular for the objective-maximizing hypothesis h^* (then $\Omega(h^*) = 1$) we obtain that $c = \frac{1}{2}$ and the split is maximally balanced then.

3.2 Purity factor

In analogy to what was shown before now we want to show the relation between the purity factor and the value of the objective function. As before in order to do that we will start from deriving an upper-bound on $\Omega(h)$, where $h \in \mathcal{H}$. For the ease of notation let $P_i = Pr(h(x) > 0 | x \in \mathcal{X}_i)$. Thus

$$\Omega(h) = 2\sum_{i=1}^{k}\pi_i\left|P(h(x)>0|x\in\mathcal{X}_i) - P(h(x)>0)\right| = 2\sum_{i=1}^{k}\pi_i\left|P_i - \sum_{j=1}^{k}\pi_jP_j\right|,$$

where $\forall_{i=\{1,2,\dots,k\}} 0 \leq P_i \leq 1$. Let $\epsilon_i = \min(P_i, 1-P_i)$ and $\epsilon = \sum_{i=1}^{k}\pi_i\epsilon_i$. Let $p = P(h(x) > 0)$. Without loss of generality let $p \leq \frac{1}{2}$. Let

$$L_1 = \{i : i \in \{1,2,\dots,k\}, P_i \geq \frac{1}{2}\},$$

$$L_2 = \{i : i \in \{1,2,\dots,k\}, P_i \in [p, \frac{1}{2})\}$$

and

$$L_3 = \{i : i \in \{1,2,\dots,k\}, P_i < p\}.$$

First notice that:

$$p = \sum_{i=1}^{k}\pi_iP_i = \sum_{i\in L_1}\pi_i(1-\epsilon_i) + \sum_{i\in L_2\cup L_3}\pi_i\epsilon_i = \sum_{i\in L_1}\pi_i - 2\sum_{i\in L_1}\pi_i\epsilon_i + \epsilon$$

We can then write that

$$\frac{\Omega(h)}{2} = \sum_{i=1}^{k}\pi_i|P_i - p| = \sum_{i\in L_1}\pi_i(1-\epsilon_i-p) + \sum_{i\in L_2}\pi_i(\epsilon_i - p) + \sum_{i\in L_3}\pi_i(p-\epsilon_i)$$

$$= \sum_{i\in L_1}\pi_i(1-p) - \sum_{i\in L_1}\pi_i\epsilon_i + \sum_{i\in L_2}\pi_i\epsilon_i - \sum_{i\in L_2}\pi_ip + \sum_{i\in L_3}\pi_ip - \sum_{i\in L_3}\pi_i\epsilon_i$$

$$= \sum_{i\in L_1}\pi_i(1-p) - \sum_{i\in L_1}\pi_i\epsilon_i + \sum_{i\in L_2}\pi_i\epsilon_i - \sum_{i\in L_2}\pi_ip + p(1-\sum_{i\in L_1}\pi_i - \sum_{i\in L_2}\pi_i) - \sum_{i\in L_3}\pi_i\epsilon_i$$

$$= \sum_{i\in L_1}\pi_i(1-2p) - \sum_{i\in L_1}\pi_i\epsilon_i + \sum_{i\in L_2}\pi_i\epsilon_i + p(1-2\sum_{i\in L_2}\pi_i) - \sum_{i\in L_3}\pi_i\epsilon_i$$
$$= \sum_{i\in L_1}\pi_i(1-2p) + p(1-2\sum_{i\in L_2}\pi_i) - \epsilon + 2\sum_{i\in L_2}\pi_i\epsilon_i$$
$$= (1-2p)(p + 2\sum_{i\in L_1}\pi_i\epsilon_i - \epsilon) + p(1-2\sum_{i\in L_2}\pi_i) - \epsilon + 2\sum_{i\in L_2}\pi_i\epsilon_i$$
$$= 2(1-p)(p-\epsilon) + 2(1-2p)\sum_{i\in L_1}\pi_i\epsilon_i - 2p\sum_{i\in L_2}\pi_i + 2\sum_{i\in L_2}\pi_i\epsilon_i$$
$$= 2(1-p)(p-\epsilon) + 2(1-2p)\sum_{i\in L_1}\pi_i\epsilon_i + 2\sum_{i\in L_2}\pi_i(\epsilon_i - p)$$
$$\leq 2(1-p)(p-\epsilon) + 2(1-2p)\sum_{i\in L_1}\pi_i\epsilon_i + 2\sum_{i\in L_2}\pi_i(\frac{1}{2} - p)$$
$$\leq 2(1-p)(p-\epsilon)+2(1-2p)\sum_{i\in L_1}\pi_i\epsilon_i+2\sum_{i\in L_2}\pi_i(\frac{1}{2}-p) \leq 2(1-p)(p-\epsilon)+2(1-2p)\sum_{i\in L_1}\pi_i\epsilon_i+1-2p$$
$$\leq 2(1-p)(p-\epsilon) + 2(1-2p)\epsilon + 1 - 2p = 2p(1-p) - 2\epsilon(1-p) + 2\epsilon(1-2p) + 1 - 2p$$
$$= 2p(1-p) - 2\epsilon(1-p-1+2p) + 1 - 2p = 2p(1-p) - 2p\epsilon + 1 - 2p$$
$$= 1 - 2p^2 - 2p\epsilon$$

Thus:

$$\epsilon \leq \frac{2-\Omega(h)}{4p} - p$$

That yields the balancing factor δ to be $\delta = \frac{2-\Omega(h)}{4p} - p$. We already know that maximizing $\Omega(h)$ leads to narrowing the $[c, 1-c]$ interval around value $\frac{1}{2}$ which corresponds to the maximally balanced split. Thue p will be then pushed closer to value $\frac{1}{2}$ and that will result in the decrease of δ. In particular for the objective-maximizing hypothesis h^* (then $\Omega(h^*) = 1$) we obtain the maximally balanced split and then $p = \frac{1}{2}$ and simultaneously we obtain that $\delta = 0$ and thus this split is also maximally pure then.

4 Boosting statement

We will now use the entropy of the tree leafs, a standard measure used in decision trees, to measure the quality of obtained tree and show the upper-bound on the number of splits required to reduce this measure below threshold ϵ. We borrow from the theoretical analysis of decision tree algorithms in Kearns and Mansour (1995) originally developed to show the boosting properties of the decision trees for binary classification problems. Our analysis generalizes the analysis there to the multi class classification setting. Consider the tree T, where every node except for leafs (we will refer to the set of the tree leafs as $\mathcal{L}$) is 'characterized' by the splitting hypothesis $h \in \mathcal{H}$ recovered by maximizing the objective function introduced before. We will consider the entropy function G as the measure of the quality of tree T:

$$G(T) = \sum_{n\in\mathcal{L}} w(n)\sum_{i=1}^{k} -\pi_{ni}\ln(\pi_{ni})$$

where π_{ni}'s are the probabilities that randomly chosen x drawn from the underlying target distribution $\mathcal{P}$ has label i given that x reaches node n and $w(n)$ is the weight of leaf n defined as the probability of randomly chosen x drawn from $\mathcal{P}$ to reach leaf n (note that $\sum_{n\in\mathcal{L}} w(n) = 1$).

Lets fix a leaf node n. For the ease of notation let $w = w_n$. We will consider splitting the leaf to two children n_0 and n_1. For the ease of notation let $w_0 = w_{n_0}$ and $w_1 = w_{n_1}$. Also for the ease of notation let $p = P(h_n(x) > 0)$ and $P_i = P(h_n(x) > 0|i)$. Let π_i be the probability that randomly chosen x drawn from $\mathcal{P}$ has label i given that x reaches node n. Recall that $p = \sum_{i=1}^{k}\pi_i P_i$ and $\sum_{i=1}^{k}\pi_i = 1$. Also notice that $w_0 = w(1-p)$ and $w_1 = wp$. Let $\boldsymbol{\pi}$ be the k-element vector with i^{th} entrance equal to π_i. Furthermore let $G(\boldsymbol{\pi}) = \sum_{i=1}^{k} -\pi_i\ln(\pi_i)$. Before the split the contribution of node n to the total loss of the tree T, that we will refer to as G_t (t is the index of the current iteration), was $wG(\pi_1, \pi_2, \dots, \pi_k)$. Let $\pi_i(n_0) = \frac{\pi_i(1-P_i)}{1-p}$ and $\pi_i(n_1) = \frac{\pi_i P_i}{p}$ be the probabilities that randomly chosen x drawn from $\mathcal{P}$ has label i given that x reaches node respectively n_0 or n_1. Furthermore let $\boldsymbol{\pi}(n_0)$ be the k-element vector with i^{th} entrance equal to $\pi_i(n_0)$ and let $\boldsymbol{\pi}(n_1)$ be the k-element vector with i^{th} entrance equal to $\pi_i(n_1)$. Notice that $\boldsymbol{\pi} = (1-p)\boldsymbol{\pi}(n_0) + p\boldsymbol{\pi}(n_1)$. After the split the contribution of the same, now internal, node n changes to $w((1-p)G(\pi(n_0)) + pG(\pi(n_1))$. We will denote the difference between them as Δ_t and thus

$$\Delta_t = w\left[G(\boldsymbol{\pi}_1) - (1-p)G(\boldsymbol{\pi}(n_0)) - pG(\boldsymbol{\pi}(n_1)\right]$$

We will aim to lower-bound Δ_t, but before that we still need to present two more useful results. Without loss of generality assume that $P_1 \leq P_2 \leq \dots \leq P_k$. For the ease of notation let $\Omega = \Omega(h_n)$. Recall that

$$\frac{\Omega}{2} = \sum_{i=1}^{k} \pi_i |P_i - p|$$

From strong concativity we know that

$$\Delta_t \geq wp(1-p)\|\boldsymbol{\pi}(n_0) - \boldsymbol{\pi}(n_1)\|_1^2 = wp(1-p)\left(\sum_{i=1}^{k} |\pi_i(n_0) - \pi_i(n_1)|\right)^2$$

$$= wp(1-p)\left(\sum_{i=1}^{k} \pi_i \left|\frac{P_i}{p} - \frac{1-P_i}{1-p}\right|\right)^2 = wp(1-p)\left(\sum_{i=1}^{k} \pi_i \left|\frac{P_i - p}{p(1-p)}\right|\right)^2$$

$$= \frac{w}{p(1-p)}\left(\sum_{i=1}^{k} |\pi_i(P_i - p)|\right)^2 = \frac{w\Omega^2}{4p(1-p)}$$

Furthermore notice that at round t there must be a leaf n such that $w(n) \geq \frac{G_t}{2t\ln k}$ (we assume we selected this leaf to the currently considered split), where $G_t = \sum_{n \in \mathcal{L}} w(n) \sum_{i=1}^{k} -\pi_{ni} \ln(\pi_{ni})$. That is because
$G_t = \sum_{n \in \mathcal{L}} w(n) \sum_{i=1}^{k} -\pi_{ni} \ln(\pi_{ni}) \leq \sum_{n \in \mathcal{L}} w(n) \ln k \leq 2tw_{max} \ln k$ where $w_{max} = \max_n w(n)$. Thus $w_{max} \geq \frac{G_t}{2t\ln k}$. Thus

$$\Delta_t \geq \frac{\Omega^2 G_t}{8p(1-p)t\ln k}$$

Definition 3. *(Weak Hypothesis Assumption)*
Let $\gamma \in (0, \min(p_n, 1-p_n)]$. Let for any distribution $\mathcal{P}$ over $\mathcal{X}$ at each non-leaf node n of the tree T there exists a hypothesis $h \in \mathcal{H}$ such that $\forall_{i \in \{1,2,\dots,k\}} |P_{ni} - p_n| \geq \gamma$.

Note that the Weak Hypothesis Assumption in fact requires that each non-leaf node of the tree T have a hypothesis h in its hypothesis class $\mathcal{H}$ which guarantees certain 'weak' purity of the split on any distribution $\mathcal{P}$ over $\mathcal{X}$. Also note that the condition $\gamma \in (0, \min(p_n, 1-p_n)]$ implies that $\gamma \leq \frac{1}{2}$. From the Weak Hypothesis Assumption it follows that for any n, p_n cannot be too near 0 or 1 since $1-\gamma \geq p_n \geq \gamma$. We will now proceed to further lower-bounding Δ_t. Note that

$$\frac{\Omega}{2} = \sum_{i=1}^{k} \pi_i(|p - P_i|) \geq \sum_{i=1}^{k} \pi_1 \gamma = \gamma$$

Thus

$$\Omega \geq 2\gamma$$

and finally

$$\Delta_t \geq \frac{\gamma^2 G_t}{2(1-\gamma)^2 t \ln k}.$$

Let $\eta = \sqrt{\frac{8}{(1-\gamma)^2 \ln k}}\gamma$. Then

$$\Delta_t > \frac{\eta^2 G_t}{16t}$$

Thus we obtain the recurrence inequality

$$G_{t+1} \leq G_t - \Delta_t < G_t - \frac{\eta^2 G_t}{16t} = G_t \left[1 - \frac{\eta^2}{16t}\right]$$

We can now compute the minimum number of splits required to reduce G_t below ϵ, where $\epsilon \in [0, 1]$. We use the result from Kearns and Mansour (1995) (see the proof of Theorem 10) and obtain the following theorem.

Theorem 1. *Under the Weak Hypothesis Assumption, for any $\epsilon \in [0, 1]$, to obtain $G_t \leq \epsilon$ it suffices to make*

$$t \geq \left(\frac{1}{\epsilon}\right)^{\frac{4(1-\gamma)^2 \ln k}{\gamma^2}}$$

splits.

We will now show one more interesting result for the special case when having $k = 2$ classes. We will show that the worst case value of p w.r.t. the change in the potential is a balanced p, in particular we will prove the Lemma which states that in the worst case setting when Δ_t is minimized, the value of p has to lie in the interval $[0.4, 0.6]$. We will start from showing a useful result. Recall that $\pi(n_0)(1-p) + \pi(n_1)p = \pi_1$ and thus $\pi(n_0) = \frac{\pi_1 - \pi(n_1)p}{1-p}$. Therefore we can write that:

$$\pi(n_0) - \pi(n_1) = \left[\frac{\pi_1 - \pi(n_1)p}{1-p} - \pi(n_1)\right] = \frac{\pi_1}{1-p} - \frac{\pi(n_1)}{1-p} = \frac{\pi_1}{1-p} - \frac{\pi_1 P_1}{p(1-p)},$$

where the last equality comes from the fact that $\pi(n_1) = \frac{\pi_1 P_1}{p}$. Let $\delta = \pi(n_0) - \pi(n_1)$ and thus

$$p(1-p)\delta = p\pi_1 - P_1\pi_1 = \pi_1(p - P_1) \geq \gamma\pi_1 \geq \gamma\pi_1(1-\pi_1),$$

where the first inequality comes from the Weak Learning Assumption. Thus we obtained that $p(1-p)\delta \geq \gamma\pi_1(1-\pi_1)$. One can compare this result with Lemma 2 from Kearns and Mansour (1995) and observe that as expected we obtained similar condition. Now recall that

$$\Delta_t = G(\pi_1) - (1-p)G(\pi(n_0)) - pG(\pi(n_1)),$$

where without loss of generality weight w was assumed to be $w = 1$. Notice that $\pi(n_0) = \pi_1 - p\delta$ and $\pi(n_1) = \pi_1 + (1-p)\delta$ (it can be easily verified by substituting $\pi_1 = \pi(n_0)(1-p) + \pi(n_1)p$). We can further express Δ_t as a function of π_1, p, δ as follows

$$\Delta_t(\pi_1, p, \delta) = G(\pi_1) - (1-p)G(\pi_1 - p\delta) - pG(\pi_1 + (1-p)\delta).$$

For any fixed values $\pi_1, p \in [0, 1]$, $\Delta_t(\pi_1, p, \delta)$ is minimized by choosing δ as small as possible. Thus let us set $\delta = \frac{\gamma\pi_1(1-\pi_1)}{p(1-p)}$ and define

$$\begin{aligned}\Delta_t(\pi_1, p) &= \Delta_t(\pi_1, p, \frac{\gamma\pi_1(1-\pi_1)}{p(1-p)}) \\ &= G(\pi_1) - (1-p)G\left(\pi_1 - \frac{\gamma\pi_1(1-\pi_1)}{(1-p)}\right) - pG\left(\pi_1 + \frac{\gamma\pi_1(1-\pi_1)}{p}\right). \qquad (1)\end{aligned}$$

The next Lemma is a direct application of Lemma 4 from Kearns and Mansour (1995).

Lemma 3. *Let $\gamma \in [0, 1]$ be any fixed value, and let $\Delta_t(\pi_1, p)$ be as defined in Equation 1. Then for any fixed $\pi_1 \in [0, 1]$, $\Delta_t(\pi_1, p)$ is minimized by a value of p falling in the interval $[0.4, 0.6]$.*

That implies that in the worst case setting when Δ_t is minimized the split has to be close to balanced.

5 Algorithm

Based on the previous formulation of the objective function one can show one more equivalent form of this objective function which will yield a simple online algorithm for tree construction and training. Notice that

$$\Omega(h) = 2|\mathbb{E}[\mathbb{1}(h(x) > 0)] - \mathbb{E}[\mathbb{1}(h(x) > 0|i)]|$$

This is a discrete optimization problem that we relax and instead consider the following objective function to maximize

$$\Omega(h) = 2|\mathbb{E}[h(x) > 0] - \mathbb{E}[h(x) > 0|i]|$$

We can store the empirical estimate of the expectations and very easily update them online. The sign of the difference decides whether the currently seen example should be send to the left or right child node.

Algorithm 1 Online tree training (regression algorithm R, threshold p_{thr})

```
create the root node $r$
initialize $\boldsymbol{q}_1^r = \text{zeros}(k,1)$, $q_2^r = 0$, $n_t^r = 0$, $\forall_{i \in \{1,2,\ldots,k\}} n_i^r = 0$
foreach example $s(x,y)$ do
        Set $j = r$
        while $j$ is not a leaf do
            update
                $q_1^j(y) *= n_y^j/(n_y^j + 1)$;  $q_1^j(y) += f^j(s(x,y))$;  $q_1^j(y) /= n_y^j + 1$
                $q_2^j *= n_t^j/(n_t^j + 1)$;  $q_2^j += f^j(s(x,y))$;  $q_2^j /= n_t^j + 1$
            if $(q_1^j(y) > q_2^j)$
                $Y_R^j = Y_R^j \cup y(s(x,y))$, $c = 1$
            else
                $Y_L^j = Y_L^j \cup y(s)$, $c = 0$
            Train $f^j$ with example $(s(x,y), c)$ (use absolute value loss and take step in the
                                                    subgradient direction)
            Set $j$ to the child of $j$ corresponding to $c$
            $n_t^j += 1$;  $n_y^j += 1$
        create children of leaf $j$
            left with a copy of $j$ (including $f^j$)
            right with label $y$ trained on $(s(x,y), 0)$
        Train $f^j$ with $(s(x,y), 1)$
```

6 Experiments

We empirically compare the quality of the split at the root obtained by our algorithm with twenty random partitions. In order to do so we build only a single layer of the tree (root and two children) and train the root by sending the examples to the children nodes following the sign of the difference of the expectations. The label is then assigned to the child node that has seen more data points with this particular label. We report the error rate of this approach and compare against the error rate of random partitions. We performed experiments on two publicly available datasets, MNIST (Lecun and Cortes (2009), 10 labels) and Reuters RCV1 (Lewis et al. (2004), 99 labels). Additionally we also performed experiments on the artificially generated dataset (we call it data100), mixture of 100 well-separated spherical Gaussians with means placed on the 10×10 grid. Each dataset was split into a training and test set. All methods were implemented in the open source system Vowpal Wabbit (Langford et al. (2007)). For the MNIST dataset, our algorithm recovered a balanced partition, i.e. half of the labels were assigned to each child node. For the data100 dataset the recovered partition was almost balanced, i.e. 51 labels were assigned to one child and 49 to the other. For the RCV1 dataset, our algorithm recovered a partition where 33 labels were assigned to one child node and the remaining ones were assigned to the other. Thus in this case we compare our algorithm with the random partitions that also divide the labels in proportion 33/66 and also with random partitions that are balanced (proportion 49/50). The results are captured in Table 1 where clearly our algorithm is able to recover significantly better partitions.

Dataset	Algorithm	Training error	Testing error
data100	Our algorithm	**0.0449**	**0.0077**
	Random partitions	0.4953 ± 0.0387	0.4954 ± 0.0384
MNIST	Our algorithm	**0.1263**	**0.1253**
	Random partitions	0.1702 ± 0.0340	0.1744 ± 0.0365
RCV1	Our algorithm	**0.0942**	**0.1185**
	Random partitions (33/66)	0.1666 ± 0.0209	0.2176 ± 0.0276
	Random partitions (49/50)	0.1906 ± 0.0192	0.2443 ± 0.0261

Table 1: Error rates at the root for data100 dataset for $49/51$ partitions, for MNIST dataset for $50/50$ partitions and for RCV1 dataset for $33/66$ and $49/50$ partitions.

Acknowledgments

We would like to thank Dean Foster, Robert Schapire and Matus Telgarsky for valuable discussions.

References

Agarwal, R., Gupta, A., Prabhu, Y., and Varma, M. (2013). Multi-label learning with millions of labels: Recommending advertiser bid phrases for web pages. In *WWW*.

Bengio, S., Weston, J., and Grangier, D. (2010). Label embedding trees for large multi-class tasks. In *NIPS*.

Beygelzimer, A., Langford, J., Lifshits, Y., Sorkin, G. B., and Strehl, A. L. (2009a). Conditional probability tree estimation analysis and algorithms. In *UAI*.

Beygelzimer, A., Langford, J., and Ravikumar, P. D. (2009b). Error-correcting tournaments. In *ALT*.

Deng, J., Satheesh, S., Berg, A. C., and Li, F.-F. (2011). Fast and balanced: Efficient label tree learning for large scale object recognition. In *NIPS*.

Kearns, M. and Mansour, Y. (1995). On the boosting ability of top-down decision tree learning algorithms. In *In Proceedings of the Twenty-Eighth Annual ACM Symposium on the Theory of Computing*, pages 459–468. ACM Press.

Langford, J., Li, L., and Srehl, A. (2007). Vowpal wabbit program, http://hunch.net/ vw.

Lecun, Y. and Cortes, C. (2009). The mnist database of handwritten digits.

Lewis, D. D., Yang, Y., Rose, T. G., and Li, F. (2004). Rcv1: A new benchmark collection for text categorization research. *J. Mach. Learn. Res.*, 5:361–397.

Rifkin, R. and Klautau, A. (2004). In defense of one-vs-all classification. *J. Mach. Learn. Res.*, 5:101–141.

Weston, J., Makadia, A., and Yee, H. (2013). Label partitioning for sublinear ranking. In *ICML*.