Optimal Minimal Margin Maximization with Boosting

Allan Grønlund [*1] **Kasper Green Larsen** [*1] **Alexander Mathiasen** [*1]

Abstract

Boosting algorithms iteratively produce linear combinations of more and more base hypotheses and it has been observed experimentally that the generalization error keeps improving even after achieving zero training error. One popular explanation attributes this to improvements in margins. A common goal in a long line of research, is to maximize the smallest margin using as few base hypotheses as possible, culminating with the AdaBoostV algorithm by (Rätsch & Warmuth, 2005). The AdaBoostV algorithm was later conjectured to yield an optimal trade-off between number of hypotheses trained and the minimal margin over all training points (Nie et al., 2013). Our main contribution is a new algorithm refuting this conjecture. Furthermore, we prove a lower bound which implies that our new algorithm is optimal.

1. Introduction

Boosting is one of the most famous and succesful ideas in learning. Boosting algorithms are meta algorithms that produce highly accurate predictors by combining already existing less accurate predictors. Probably the most famous boosting algorithm is AdaBoost by Freund and Schapire (Freund & Schapire, 1995), who won the 2003 Gödel Prize for their work.

AdaBoost was designed for binary classification and works by combining base hypotheses learned by a given base learning algorithm into a weighted sum that represents the final classifier. This weighed set of base hypotheses is constructed iteratively in rounds, each round constructing a new base hypothesis that focuses on the training data misclassified by the previous base hypotheses constructed. More precisely, AdaBoost takes training data $D = \{(x_i, y_i) \mid x_i \in \mathbb{R}^d, y_i \in \{-1, +1\}\}_{i=1}^n$ and constructs a linear combination classifier $\text{sign}(\sum_{t=1}^T \alpha_t h_t(x))$, where h_t is the base hypothesis learned in the t'th iteration and α_t is the corresponding weight.

It has been proven that AdaBoost decreases the training error exponentially fast if each base hypothesis is slightly better than random guessing on the weighed data set it is trained on (Freund et al., 1999). Concretely, if ϵ_t is the error of h_t on the weighed data set used to learn h_t then the linear combination has training error at most $\exp(-2\sum_{t=1}^T (1/2 - \epsilon_t)^2)$. If each ϵ_t is at most a half minus a fixed constant, then the training error is less than $1/n$ after $O(\lg n)$ rounds which means the all training points are classified correctly. Quite surprisingly, experiments show that continuing the AdaBoost algorithm even after the training data is perfectly classified, making the model more and more complex, continues to improve generalization (Schapire et al., 1998).

The most prominent approach to explaining this generalization phenomenon considers *margins* (Schapire et al., 1998). The margin of a point x_i is

$$\text{margin}(x_i) = \frac{y_i \sum_{t=1}^T \alpha_t h_t(x_i)}{\sum_{t=1}^T |\alpha_t|}.$$

For binary classification, if each $h_t(x) \in [-1, +1]$, then the margin of a point is a number between -1 and +1. Notice that a point has positive margin if it is classified correctly and negative margin if it is classified incorrectly. It has been observed experimentally that the margins of the training points usually increase when training, even after perfectly classifying the training data. This has inspired several bounds on generalization error that depend on the distribution of margins (Schapire et al., 1998; Breiman, 1999; Koltchinskii et al., 2001; Wang et al., 2008; Gao & Zhou, 2013). The conceptually simplest of these bounds depend only on the minimal margin, which is the margin of the point x_i with minimal margin. The point x_i with minimal margin can be interpreted as the point the classifier struggles the most with. This has inspired a series of algorithms with guarantees on the minimal margin (Breiman, 1999; Grove & Schuurmans, 1998; Bennett et al., 2000; Rätsch & Warmuth, 2002; 2005).

These algorithms have the following goal: Let $\mathcal{H}$ be the (possibly infinite) set of all base hypotheses that may be returned by the base learning algorithm. Suppose the best

[*]Equal contribution [1]Department of Computer Science, University of Aarhus, Denmark. Correspondence to: Kasper Green Larsen <larsen@cs.au.dk>.

Proceedings of the 36th International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019.

possible minimal margin on some training data for any linear combination of $h \in \mathcal{H}$ is ρ^*, i.e.

$$\rho^* = \max_{\alpha \neq 0} \left(\min_i \sum_{h \in \mathcal{H}} \frac{y_i \sum_{h \in \mathcal{H}} \alpha_h h(x_i)}{\sum_{h \in \mathcal{H}} |\alpha_h|} \right).$$

Given some precision parameter v, the goal is to construct a linear combination with minimal margin at least $\rho = \rho^* - v$ using as few hypotheses as possible. In this case we say that the linear combination has a gap of v. The current state of the art is AdaBoostV (Rätsch & Warmuth, 2005). It guarantees a gap of v using $O(\lg(n)/v^2)$ hypotheses. It was later conjectured that there exists data sets D and a corresponding set of base hypotheses $\mathcal{H}$, such that any linear combination of base hypotheses from $\mathcal{H}$ must use at least $\Omega(\lg(n)/v^2)$ hypotheses to achieve a gap of v for any $\sqrt{\lg n/n} \leq v \leq a_1$ for some constant $a_1 > 0$. This would imply optimality of AdaBoostV. This conjecture was published as an open problem in the Journal of Machine Learning Research (Nie et al., 2013).

Our main contribution is a refutal of this conjecture. We refute the conjecture by introducing a new algorithm called SparsiBoost, which guarantees a gap of v with just $T = O(\lg(nv^2)/v^2)$ hypotheses. When $v \leq n^{o(1)}/\sqrt{n}$, SparsiBoost has $T = O(\lg(n^{o(1)})/v^2) = o(\lg(n)/v^2)$, which is asymptotically better than AdaBoostV's $T = O(\lg(n)/v^2)$ guarantee. This refutes the conjectured lower bound. Our algorithm involves a surprising excursion to the field of combinatorial discrepancy minimization. We also show that our algorithm is the best possible. That is, there exists data sets D and corresponding set of base hypotheses $\mathcal{H}$, such that any linear combination of base hypotheses from $\mathcal{H}$ with a gap of v, must use at least $T = \Omega(\lg(nv^2)/v^2)$ hypotheses.

This work thus provides the final answer to over a decade's research into understanding the trade-off between minimal margin and the number of hypotheses: Given a gap v, the optimal number of hypotheses is $T = \Theta(\lg(nv^2)/v^2)$ for any $\sqrt{1/n} \leq v \leq a_1$ where $a_1 > 0$ is a constant. Notice that smaller values for v are irrelevant since it is always possible to achieve a gap of zero using $n+1$ base hypotheses. This follows from Carathéodory's Theorem.

1.1. Previous Work on Minimal Margin

Upper Bounds. (Breiman, 1999) introduced Arc-GV, which was the first algorithm that guaranteed to find a finite number of hypotheses $T < \infty$ with gap zero ($v = 0$). As pointed out by (Rätsch & Warmuth, 2005), one can think of Arc-GV as a subtle variant of AdaBoost where the weights α_t of the hypotheses are slightly changed. If AdaBoost has hypothesis weight α_t, then Arc-GV chooses the hypothesis weight $\alpha'_t = \alpha_t + x$ for some x that depends on the minimal margin of $h_1, \ldots, h_t$. A few years later, (Grove & Schuurmans, 1998) and (Bennett et al., 2000) introduced DualLPBoost and LPBoost with similar guarantees.

(Rätsch & Warmuth, 2002) introduced AdaBoost$_\rho$, which was the first algorithm to give a guarantee on the gap achieved in terms of the number of hypotheses used. Their algorithm takes a parameter $\rho \leq \rho^*$ that serves as the target margin one would like to achieve. It then guarantees a minimal margin of $\rho - \mu$ using $T = O(\lg(n)/\mu^2)$ hypotheses. One would thus like to choose $\rho = \rho^*$. If ρ^* is unknown, it can be found up to an additive approximation of v by binary searching using AdaBoost$_\rho$. This requires an additional $O(\lg 1/v)$ calls to AdaBoost$_\rho$, resulting in $O(\lg(n)/v^2)\lg(1/v)$ iterations of training a base hypothesis to find the desired linear combination of $T = O(\lg(n)/v^2)$ base hypotheses. Similar to Arc-GV, AdaBoost$_\rho$ differs from AdaBoost only in choosing the weights α_t. Instead of having the additional term depend on the minimal margin of $h_1, \ldots, h_t$, it depends only on the estimation parameter ρ.

A few years later, (Rätsch & Warmuth, 2005) introduced AdaBoostV. It is a clever extension of AdaBoost$_\rho$ that uses an adaptive estimate of ρ^* to remove the need to binary search for it. It achieves a gap of v using $T = O(\lg(n)/v^2)$ base hypotheses and no extra iterations of training.

Lower Bounds. (Klein & Young, 1999) showed a lower bound for a seemingly unrelated game theoretic problem. It was later pointed out by (Nie et al., 2013) that their result implies the following lower bound for boosting: there exists a data set of n points and a corresponding set of base hypotheses $\mathcal{H}$, such that any linear combination of $T \in [\lg n; \sqrt{n}]$ base hypotheses must have a gap of $v = \Omega(\sqrt{\lg(n)/T})$. Rewriting in terms of T we get $T = \Omega\left(\lg(n)/v^2\right)$ for $\sqrt{\lg(n)}/n^{1/4} \leq v \leq a_1$ for some constant $a_1 > 0$.

(Nie et al., 2013) conjectured that Klein and Young's lower bound of $v = \Omega(\sqrt{\lg(n)/T})$ holds for all $T \leq a_1 \cdot n$ for some constant $a_1 > 0$. Rewriting in terms of T, they conjecture that $T = \Omega(\lg(n)/v^2)$ holds for $\sqrt{a_2/n} \leq v \leq a_3$ where $a_2, a_3 > 0$ are some constants.

1.2. Our Results on Minimal Margin

Our main result is a novel algorithm, called SparsiBoost, which refutes the conjectured lower bound in (Nie et al., 2013). Concretely, SparsiBoost guarantees a gap of v with just $T = O(\lg(nv^2)/v^2)$ hypotheses. At a first glance it might seem SparsiBoost violates the lower bound of Klein and Young. Rewriting in terms of v, our upper bound becomes $v = O(\sqrt{\lg(n/T)/T})$ (see appendix in the full version of this paper (Grønlund et al., 2019) for how to perform the rewriting). When $T \leq \sqrt{n}$ (the range of parameters where their lower bound applies), this becomes $v = O(\sqrt{\lg(n)/T})$ which does not violate Klein and Young's lower bound. Moreover, our upper bound ex-

plains why both (Klein & Young, 1999) and (Nie et al., 2013) were not able to generalize the lower bound to all $T = O(n)$: When $T = n^{1-o(1)}$, our algorithm achieves a gap of $v = O(\sqrt{\lg(n^{o(1)})/T}) = o(\sqrt{\lg(n)/T})$.

The high level idea of SparsiBoost is as follows: Given a desired gap v, we use AdaBoostV to find $m = O(\lg(n)/v^2)$ hypotheses $h_1, \ldots, h_m$ and weights $w_1, \ldots, w_m$ such that $\sum_i w_i h_i$ achieves a gap of $v/2$. We then carefully "sparsify" the vector $w = (w_1, \ldots, w_m)$ to obtain another vector w' that has at most $T = O(\lg(nv^2)/v^2)$ non-zeroes. Our sparsification is done such that the margin of every single data point changes by at most $v/2$ when replacing $\sum_i w_i h_i$ by $\sum_i w'_i h_i$. In particular, this implies that the minimum margin, and hence gap, changes by at most $v/2$. We can now safely ignore all hypotheses h_i where $w'_i = 0$ and we have obtained the claimed gap of at most $v/2 + v/2 = v$ using $T = O(\lg(nv^2)/v^2)$ hypotheses.

Our algorithm for sparsifying w gives a general method for sparsifying a vector while approximately preserving a matrix-vector product. We believe this result may be of independent interest and describe it in more detail here: The algorithm is given as input a matrix $U \in [-1, +1]^{n \times m}$ and a vector $w \in \mathbb{R}^m$ where $\|w\|_1 = 1$. It then finds a vector w' such that $\|Uw - Uw'\|_\infty = O(\lg(n/T)/T), \|w'\|_0 \leq T$ and $\|w'\|_1 = 1$. Here $\|x\|_1 = \sum_i |x_i|$, $\|x\|_\infty = \max_i |x_i|$ and $\|x\|_0$ denotes the number of non-zero entries of x. When we use this result in SparsiBoost, we will define the matrix U as the "margin matrix" that has $u_{ij} = y_i h_j(x_i)$. Then $(Uw)_i = \text{margin}(x_i)$ and the guarantee $\|Uw - Uw'\|_\infty = O(\lg(n/T)/T)$ will ensure that the margin of every single point changes by at most $O(\lg(n/T)/T)$ if we replace the weights w by w'. Our algorithm for finding w' is based on a novel connection to the celebrated but seemingly unrelated "six standard devitations suffice" result by (Spencer, 1985) from the field of combinatorial discrepancy minimization.

When used in SparsiBoost, the matrix U is defined from the output of AdaBoostV, but the vector sparsification algorithm could just as well be applied to the hypotheses output by any boosting algorithm. Thus our results give a general method for sparsifying a boosting classifier while approximately preserving the margins of all points.

We complement our new upper bound with a matching lower bound. More concretely, we prove that there exists data sets D of n points and a corresponding set of base hypotheses $\mathcal{H}$, such that any linear combination of T base hypotheses must have a gap of at least $v = \Omega(\sqrt{\lg(n/T)/T})$ for any $\lg n \leq T \leq a_1 n$ where $a_1 > 0$ is a constant. Rewriting in terms of T, one must use $T = \Omega(\lg(nv^2)/v^2)$ hypotheses from $\mathcal{H}$ to achieve a gap of v. This holds for any v satisfying $\sqrt{a_2/n} < v \leq a_3$ for constants $a_2, a_3 > 0$ (see appendix in full version (Grønlund et al., 2019) for how to perform the rewriting). Interestingly, our lower bound proof also uses the discrepancy minimization upper bound by (Spencer, 1985) in a highly non-trivial way. Our lower bound also shows that our vector sparsification algorithm is optimal for any $T \leq n/C$ for some universal constant $C > 0$.

1.3. Margin Bounds and Doubts on Margin Theory

The first margin bound on boosted classifiers was introduced by (Schapire et al., 1998). Shortly after, (Breiman, 1999) introduced a sharper minimal margin bound alongside Arc-GV. Experimentally Breimann found that Arc-GV produced better margins than AdaBoost on 98 percent of the training data, however, AdaBoost still obtained a better test error. This seemed to contradict margin theory: according to margin theory, better margins should imply better generalization. This caused Breimann to doubt margin theory. It was later discovered by (Reyzin & Schapire, 2006) that the comparison was unfair due to a difference in the complexity of the base hypotheses used by AdaBoost and Arc-GV. (Reyzin & Schapire, 2006) performed a variant of Breimann's experiments with decision stumps to control the hypotheses complexity. They found that even though Arc-GV produced a better minimal margin, AdaBoost produced a larger margin on almost all other points (Reyzin & Schapire, 2006) and that AdaBoost generalized better.

A few years later, (Wang et al., 2008) introduced a sharper margin bound than Breimann's minimal margin bound. The generalization bound depends on a term called the *Equilibrium Margin*, which itself depends on the margin distribution in a highly non-trivial way. This was followed by the k-margin bound by (Gao & Zhou, 2013) that provide generalization bounds based on the k'th smallest margin for any k. The bound gets weaker with increasing k, but stronger with increasing margin. In essence, this means that we get stronger generalization bounds if the margins are large for small values of k.

Recall from the discussion in Section 1.2 that our sparsification algorithm preserves all margins to within $O(\sqrt{\lg(n/T)/T})$ additive error. We combined this result with AdaBoostV to get our algorithm SparsiBoost which obtained an optimal trade-off between minimal margin and number of hypotheses. While minimal margin might be insufficient for predicting generalization performance, our sparsification algorithm actually preserves the full distribution of margins. Thus according to margin theory, the sparsified classifier should approximately preserve the generalization performance of the full unsparsified classifier. To demonstrate this experimentally, we sparsified a classifier trained with LightGBM (Ke et al., 2017), a highly efficient open-source implementation of Gradient Boosting (Mason et al., 2000; Friedman, 2001). We compared the margin distribution and the test error of the sparsified classifier against a LightGBM classifier trained directly to have the same

number of hypotheses. Our results (see Section 4) show that the sparsified classifier has a better margin distribution and indeed generalize better than the LightGBM classifier.

1.4. Previous Work on Sparsification

Maintaining or improving the performance of a linear combination of base learners, while minimizing the number of base learners used, is known as ensemble pruning. The idea was introduced in (Margineantu & Dietterich, 1997) and many new methods and heuristics have been proposed since then, most focusing on optimizing both objectives in different ways. We refer to (Guo et al., 2018; Qian et al., 2015) for a longer explanation of the history of ensemble pruning, including a method that considers margins. We note that none of the previous approaches provide provable guarantees on the margin distribution.

2. SparsiBoost

In this section we introduce SparsiBoost. The algorithm takes the following inputs: training data D, a target number of hypotheses T and a base learning algorithm A that returns hypotheses from a class $\mathcal{H}$ of possible base hypotheses. SparsiBoost initially trains $c \cdot T$ hypotheses for some appropriate c, by running AdaBoostV with the base learning algorithm A. It then removes the extra $c \cdot T - T$ hypotheses while attempting to preserve the margins on all training examples. See Algorithm 1 for pseudocode.

In more detail, let $h_1, ..., h_{cT} \in \mathcal{H}$ be the hypotheses returned by AdaBoostV with weights $w_1, ..., w_{cT}$. Construct a margin matrix U that contains the margin of every hypothesis h_j on every point x_i such that $u_{ij} = y_i h_j(x_i)$. Let w be the vector of hypothesis weights, meaning that the j'th coordinate of w has the weight w_j of hypothesis h_j. Normalize $w = w/\|w\|_1$ such that $\|w\|_1 = 1$. The product Uw is then a vector that contains the margins of the linear combination on all points: $(Uw)_i = y_i \sum_{j=1}^{cT} w_j h_j(x_i) = \text{margin}(x_i)$. Removing hypotheses while preserving the margins can be formulated as sparsifying w to w' while minimizing $\|Uw - Uw'\|_\infty$ subject to $\|w'\|_0 \leq T$ and $\|w'\|_1 = 1$.

We still haven't described how to find w' with the guarantees shown in Algorithm 1 step 4., i.e. a w' with $\|Uw - Uw'\|_\infty = O(\sqrt{\lg(2 + n/T)/T})$. It is not even clear that such w' exists, much less so that it can be found efficiently. Before we dive into the details of how to find w', we briefly demonstrate that indeed such a w' would be sufficient to establish our main theorem:

Theorem 2.1. *SparsiBoost is guaranteed to find a linear combination w' of at most T base hypotheses with gap $v = O(\sqrt{\lg(2 + n/T)/T})$.*

Proof. We assume throughout the proof that a w' with the guarantees claimed in Algorithm 1 can be found. Suppose we run AdaBoostV to get cT base hypotheses $h_1, ..., h_{cT}$ with weights $w_1, ..., w_{cT}$. Let ρ_{cT} be the minimal margin of the linear combination $\sum_i w_i h_i$ on the training data D, and let ρ^* be the optimal minimal margin over all linear combinations of base hypotheses from $\mathcal{H}$. As proved in (Rätsch & Warmuth, 2005), AdaBoostV guarantees that the gap is bounded by $\rho^* - \rho_{cT} = O(\sqrt{\lg(n)/(cT)})$. Normalize $w = w/\|w\|_1$ and let U be the margin matrix $u_{ij} = y_i h_j(x_i)$ as in Algorithm 1. Then $\rho_{cT} = \min_i(Uw)_i$. From our assumption, we can efficiently find w' such that $\|w'\|_1 = 1$, $\|w'\|_0 \leq T$ and $\|Uw - Uw'\|_\infty = O(\sqrt{\lg(2 + n/T)/T})$. Consider the hypotheses that correspond to the non-zero entries of w'. There are at most T. Let ρ_T be their minimal margin when using the corresponding weights from w'. Since w' has unit ℓ_1-norm, it follows that $\rho_T = \min_i(Uw')_i$ and thus $|\rho_T - \rho_{cT}| \leq \max_i |(Uw)_i - (Uw')_i|$, i.e. $|\rho_{cT} - \rho_T| \leq \|Uw - Uw'\|_\infty = O(\sqrt{\lg(2 + n/T)/T})$. We therefore have:

$$\rho^* - \rho_T = (\rho^* - \rho_{cT}) + (\rho_{cT} - \rho_T) \leq$$
$$O(\sqrt{\lg(n)/(cT)}) + O(\sqrt{\lg(2 + n/T)/T}).$$

By choosing $c = \lg(n)/\lg(2 + n/T)$ (as in Algorithm 1) we get that $\rho^* - \rho_T = O(\sqrt{\lg(2 + n/T)/T})$. □

The core difficulty in our algorithm is thus finding an appropriate w' (step 4 in Algorithm 1) and this is the focus of the remainder of this section. Our algorithm for finding w' gives a general method for sparsifying a vector w while

Algorithm 1 SparsiBoost

Input: Training data $D = \{(x_i, y_i)\}_{i=1}^n$ where $x_i \in X$ for some input space X and $y_i \in \{-1, +1\}$. Target number of hypotheses T and base learning algorithm A.
Output: Hypotheses $h_1, \ldots, h_k$ and weights $w_1, \ldots, w_k$ with $k \leq T$, such that $\sum_i w_i h_i$ has gap $O(\sqrt{\lg(2 + n/T)/T})$ on D.

1. Run AdaBoostV with base learning algorithm A on training data D to get cT hypotheses $h_1, \ldots, h_{cT}$ and weights $w_1, \ldots, w_{cT}$ for the integer $c = \lceil \lg(n)/\lg(2 + n/T) \rceil$.
2. Construct margin matrix $U \in [-1, +1]^{n \times cT}$ where $u_{ij} = y_i h_j(x_i)$.
3. Form the vector w with i'th coordinate w_i and normalize $w \leftarrow w/\|w\|_1$ so $\|w\|_1 = 1$.
4. Find w' such that $\|w'\|_0 \leq T$, $\|w'\|_1 = 1$ and $\|Uw - Uw'\|_\infty = O(\sqrt{\lg(2 + n/T)/T})$.
5. Let $\pi(j)$ denote the index of the j'th non-zero entry of w'.
6. **Return** hypotheses $h_{\pi(1)}, \ldots, h_{\pi(\|w'\|_0)}$ with weights $w'_{\pi(1)}, \ldots, w'_{\pi(\|w'\|_0)}$.

approximately preserving every coordinate of the matrix-vector product Uw for some input matrix U. The guarantees we give are stated in the following theorem:

Theorem 2.2. *(Sparsification Theorem) For all matrices $U \in [-1,+1]^{n\times m}$, all $w \in \mathbb{R}^m$ with $\|w\|_1 = 1$ and all $T \leq m$, there exists a vector w' where $\|w'\|_1 = 1$ and $\|w'\|_0 \leq T$, such that $\|Uw - Uw'\|_\infty = O(\sqrt{\lg(2+n/T)/T})$.*

Theorem 2.2 is exactly what was needed in the proof of Theorem 2.1. Our proof of Theorem 2.2 will be constructive in that it gives an algorithm for finding w'. To keep the proof simple, we will argue about running time at the end of the section.

The first idea in our algorithm and proof of Theorem 2.2, is to reduce the problem to a simpler task, where instead of reducing the number of hypotheses directly to T, we only halve the number of hypotheses:

Lemma 2.1. *For all matrices $U \in [-1,+1]^{n\times m}$ and $w \in \mathbb{R}^m$ with $\|w\|_1 = 1$, there exists w' where $\|w'\|_0 \leq \|w\|_0/2$ and $\|w'\|_1 = 1$, such that $\|Uw - Uw'\|_\infty = O(\sqrt{\lg(2+n/\|w\|_0)/\|w\|_0})$.*

To prove Theorem 2.2 from Lemma 2.1, we can repeatedly apply Lemma 2.1 until we are left with a vector with at most T non-zeroes. Since the loss $O(\sqrt{\lg(2+n/\|w\|_0)/\|w\|_0})$ has a $\sqrt{1/\|w\|_0}$ factor, we can use the triangle inequality to conclude that the total loss is a geometric sum that is asymptotically dominated by the very last invocation of the halving procedure. Since the last invocation has $\|w\|_0 > T$ (otherwise we would have stopped earlier), we get a total loss of $O(\sqrt{\lg(2+n/T)/T})$ as desired. The proof is in the full version (Grønlund et al., 2019).

The key idea in implementing the halving procedure Lemma 2.1 is as follows: Let $\pi(j)$ denote the index of the j'th non-zero in w and let $\pi^{-1}(j)$ denote the index i such that w_i is the j'th non-zero entry of w. First we construct a matrix A where the j'th column of A is equal to the $\pi(j)$'th column of U scaled by the weight $w_{\pi(j)}$. The sum of the entries in the i'th row of A is then equal to the i'th entry of Uw (since $\sum_j a_{ij} = \sum_j w_{\pi^{-1}(j)}u_{i\pi^{-1}(j)} = \sum_{j:w_j\neq 0} w_j u_{ij} = \sum_j w_j u_{ij} = (Uw)_i$). Minimizing $\|Uw - Uw'\|_\infty$ can then be done by finding a subset of columns of A that approximately preserves the row sums. This is formally expressed in Lemma 2.2 below. For ease of notation we define $a \pm [x]$ to be the interval $[a-x, a+x]$ an $\pm[x]$ to be the interval $[-x,x]$.

Lemma 2.2. *For all matrices $A \in [-1,1]^{n\times T}$ there exists a submatrix $\hat{A} \in [-1,1]^{n\times k}$ consisting of $k \leq T/2$ distinct columns from A, such that for all i, it holds that $\sum_{j=1}^k \hat{a}_{ij} \in \frac{1}{2}\sum_{j=1}^T a_{ij} \pm \left[O(\sqrt{T\lg(2+n/T)})\right]$.*

Intuitively we can now use Lemma 2.2 to select a subset S of at most $T/2$ columns in A. We can then replace the vector w with w' such that $w'_i = 2w_i$ if $i = \pi^{-1}(j)$ for some $j \in S$ and $w'_i = 0$ otherwise. In this way, the i'th coordinate $(Uw')_i$ equals the i'th row sum in $\hat{A}$, scaled by a factor two. By Lemma 2.2, this in turn approximates the i'th row sum in A (and thus $(Uw)_i$) up to additively $O(\sqrt{T\lg(2+n/T)})$.

Unfortunately our procedure is not quite that straightforward since $O(\sqrt{T\lg(2+n/T)})$ is way too large compared to $O(\sqrt{\lg(2+n/\|w\|_0)/\|w\|_0}) = O(\sqrt{\lg(2+n/T)/T})$. Fortunately Lemma 2.2 only needs the coordinates of A to be in $[-1,1]$. We can thus scale A by $1/\max_i |w_i|$ and still satisfy the constraints. This in turn means that the loss is scaled down by a factor $\max_i |w_i|$. However, $\max_i |w_i|$ may be as large as 1 for highly unbalanced vectors. Therefore, we start by copying the largest $T/3$ entries of w to w' and invoke Lemma 2.2 twice on the remaining $2T/3$ entries. This ensures that the $O(\sqrt{T\lg(2+n/T)})$ loss in Lemma 2.2 gets scaled by a factor at most $3/T$ (since $\|w\|_1 = 1$, all remaining coordinates are less than or equal to $3/T$), while leaving us with at most $T/3 + (2T/3)/4 = T/3 + T/6 = T/2$ non-zero entries as required. Since we normalize by at most $3/T$, the error becomes $\|Uw - Uw'\|_\infty = O(\sqrt{T\lg(2+n/T)/T}) = O(\sqrt{\lg(2+n/T)/T})$ as desired. As a last technical detail, we also need to ensure that w' satisfies $\|w'\|_1 = 1$. We do this by adding an extra row to A such that $a_{(n+1)j} = w_j$. In this way, preserving the last row sum also (roughly) preserves the ℓ_1-norm of w and we can safely normalize w' as $w' \leftarrow w'/\|w\|_1$. A formal proof is in the full version (Grønlund et al., 2019).

The final step of our algorithm is thus to select a subset of at most half of the columns from a matrix $A \in [-1,1]^{n\times T}$, while approximately preserving all row sums. Our idea for doing so builds on the following seminal result by Spencer:

Theorem 2.3. *(Spencer's Theorem (Spencer, 1985)) For all matrices $A \in [-1,+1]^{n\times T}$ with $T \leq n$, there exists $x \in \{-1,+1\}^T$ such that $\|Ax\|_\infty = O(\sqrt{T\ln(en/T)})$. For all matrices $A \in [-1,+1]^{n\times T}$ with $T > n$, there exists $x \in \{-1,+1\}^T$ such that $\|Ax\|_\infty = O(\sqrt{n})$.*

We use Spencer's Theorem as follows: We find a vector $x \in \{-1,+1\}^T$ with $\|Ax\|_\infty = O(\sqrt{T\ln(en/T)})$ if $T \leq n$ and with $\|Ax\|_\infty = O(\sqrt{n}) = O(\sqrt{T})$ if $T > n$. Thus we always have $\|Ax\|_\infty = O(\sqrt{T\lg(2+n/T)})$. Consider now the i'th row of A and notice that $|\sum_{j:x_j=1} a_{ij} - \sum_{j:x_j=-1} a_{ij}| \leq \|Ax\|_\infty$. That is, for every single row, the sum of the entries corresponding to columns where x is 1, is almost equal (up to $\pm\|Ax\|_\infty$) to the sum over the columns where x is -1. Since the two together sum to the full row sum, it follows that the subset of columns with $x_i = 1$ *and* the subset of columns with $x_i = -1$ *both* preserve the row sum as required by Lemma 2.2. Since x has at most $T/2$

of either $+1$ or -1, it follows that we can find the desired subset of columns. The proof of Lemma 2.2 using Spencer's Theorem is in the full version (Grønlund et al., 2019).

Algorithm 2 Sparsification

Input: Matrix $U \in [-1,1]^{n\times m}$, vector $w \in \mathbb{R}^m$ with $\|w\|_1 = 1$ and target $T \leq m$.
Output: A vector $w' \in \mathbb{R}^m$ with $\|w'\|_1 = 1, \|w'\|_0 \leq T$ and $\|Uw - Uw'\|_\infty = O(\sqrt{\lg(2+n/T)/T})$.

1. Let $w' \leftarrow w$.
2. **While** $\|w'\|_0 > T$:
3. Let R be the indices of the $\|w'\|_0/3$ entries in w' with largest absolute value.
4. Let $\omega := \max_{i\notin R} |w_i|$ be the largest value of an entry outside R.
5. **Do Twice**:
6. Let $\pi(1), \pi(2), \ldots, ..., \pi(k)$ be the indices of the non-zero entries in w' that are not in R.
7. Let $A \in [-1,1]^{(n+1)\times k}$ have $a_{ij} = u_{i\pi(j)} w'_{\pi(j)}/\omega$ for $i \leq n$ and $a_{(n+1)j} = |w'_{\pi(j)}|/\omega$.
8. Invoke Spencer's Theorem to find $x \in \{-1,1\}^k$ such that $\|Ax\|_\infty = O(\sqrt{k\lg(2+n/k)})$.
9. Let $\sigma \in \{-1,1\}$ denote the sign such that $x_i = \sigma$ for at most $k/2$ indices i.
10. Update w'_i as follows:
11. If there is a j such that $i = \pi^{-1}(j)$ and $x_j = \sigma$: set $w'_i \leftarrow 2w'_i$.
12. If there is a j such that $i = \pi^{-1}(j)$ and $x_j \neq \sigma$: set $w'_i \leftarrow 0$.
13. Otherwise ($i \in R$ or $w'_i = 0$): set $w'_i \leftarrow w'_i$.
14. Update $w' \leftarrow w'/\|w'\|_1$.
15. **Return** w'.

We have summarized the entire sparsification algorithm in Algorithm 2 and end with a few added remarks.

Running Time. While Spencer's original result (Theorem 2.3) is purely existential, recent follow up work (Lovett & Meka, 2015) show how to find the vector $x \in \{-1,+1\}^n$ in expected $\tilde{O}((n+T)^3)$ time, where $\tilde{O}$ hides polylogarithmic factors. A small modification to the algorithm was suggested in (Larsen, 2017). This modification reduces the running time of Lovett and Meka's algorithm to expected $\tilde{O}(nT + T^3)$. This is far more desirable as T tends to be much smaller than n in boosting. Moreover, the nT term is already paid by running AdaBoostV. Using this in Step 7. of Algorithm 2, we get a total expected running time of $\tilde{O}(nT + T^3)$. We remark that these algorithms are randomized and lead to different vectors x on different executions.

Non-Negativity. Examining Algorithm 2, we observe that the weights of the input vector are only ever copied, set to zero, or scaled by a factor two. Hence if the input vector w has non-negative entries, then so has the final output vector w'. This may be quite important if one interprets the linear combination over hypotheses as a probability distribution.

Importance Sampling. Another natural approach one might attempt in order to prove our sparsification result, Theorem 2.2, is to apply importance sampling. Importance sampling samples T entries from w with replacement, such that each entry i is sampled with probability $|w_i|$. It then returns the vector w' where coordinate i is equal to $\text{sign}(w_i)n_i/T$ where n_i denotes the number of times i was sampled and $\text{sign}(w_i) \in \{-1,1\}$ gives the sign of w_i. Analysing this method gives a w' with $\|Uw - Uw'\|_\infty = \Theta(\sqrt{\lg(n)/T})$ (with high probability), i.e. slightly worse than our approach based on discrepancy minimization. The loss in the lg is enough that if we use importance sampling in SparsiBoost, then we get no improvement over simply stopping AdaBoostV after T iterations.

3. Lower Bound

In this section, we explain our lower bound stating that there exist a data set and corresponding set of base hypotheses $\mathcal{H}$, such that if one uses only T of the base hypotheses in $\mathcal{H}$, then one cannot obtain a gap smaller than $\Omega(\sqrt{\lg(n/T)/T})$. Similar to the approach taken in (Nie et al., 2013), we model a data set $D = \{(x_i, y_i)\}_{i=1}^n$ of n data points and a corresponding set of k base hypotheses $\mathcal{H} = \{h_1, \ldots, h_k\}$ as an $n \times k$ matrix A. The entry $a_{i,j}$ is equal to $y_i h_j(x_i)$. We prove our lower bound for binary classification where the hypotheses take values only amongst $\{-1,+1\}$, meaning that $A \in \{-1,+1\}^{n\times k}$. Thus an entry $a_{i,j}$ is $+1$ if hypothesis h_j is correct on point x_i and it is -1 otherwise. We remark that proving the lower bound under the restriction that $h_j(x_i)$ is among $\{-1,+1\}$ instead of $[-1,+1]$ only strengthens the lower bound.

Notice that if $w \in \mathbb{R}^k$ is a vector with $\|w\|_1 = 1$, then $(Aw)_i$ gives exactly the margin on data point (x_i, y_i) when using the linear combination $\sum_j w_j h_j$ of base hypotheses. The optimal minimum margin ρ^* for a matrix A is thus equal to $\rho^* := \max_{w\in\mathbb{R}^k:\|w\|_1=1} \min_i (Aw)_i$. We now seek a matrix A for which ρ^* is at least $\Omega(\sqrt{\lg(n/T)/T})$ larger than $\min_i (Aw)_i$ for all w with $\|w\|_0 \leq T$ and $\|w\|_1 = 1$. If we can find such a matrix, it implies the existence of a data set (rows) and a set of base hypotheses (columns) for which any linear combination of up to T base hypotheses has a gap of $\Omega(\sqrt{\lg(n/T)/T})$. The lower bound thus holds regardless of how an algorithm would try to determine which linear combination to construct.

When showing the existence of a matrix A with a large

gap, we fix $k = n$, i.e. the set of base hypotheses $\mathcal{H}$ has cardinality equal to the number of data points. The following theorem shows the existence of the desired matrix A:

Theorem 3.1. *There exists a universal constant $C > 0$ such that for all sufficiently large n and all T with $\ln n \leq T \leq n/C$, there exists a matrix $A \in \{-1,+1\}^{n\times n}$ such that: 1) Let $v \in \mathbb{R}^n$ be the vector with all coordinates equal to $1/n$. Then all coordinates of Av are greater than or equal to $-O(1/\sqrt{n})$. 2) For every vector $w \in \mathbb{R}^n$ with $\|w\|_0 \leq T$ and $\|w\|_1 = 1$, it holds that: $\min_i(Aw)_i \leq -\Omega\left(\sqrt{\lg(n/T)/T}\right)$.*

Quite surprisingly, Theorem 3.1 shows that for any T with $\ln n \leq T \leq n/C$, there is a matrix $A \in \{-1,+1\}^{n\times n}$ for which the *uniform* combination of base hypotheses $\sum_{j=1}^n h_j/n$ has a minimum margin that is much higher than what is possible using only T base hypotheses. Specifically, let A be a matrix satisfying the properties of Theorem 3.1 and let $v \in \mathbb{R}^n$ be the vector with all coordinates $1/n$. Then $\rho^* := \max_{w\in\mathbb{R}^k:\|w\|_1=1} \min_i(Aw)_i \geq \min_i(Av)_i = -O(1/\sqrt{n})$. By the second property in Theorem 3.1, it follows that any linear combination of at most T base hypotheses must have a gap of $-O(1/\sqrt{n}) - \left(-\Omega\left(\sqrt{\lg(n/T)/T}\right)\right) = \Omega\left(\sqrt{\lg(n/T)/T}\right)$. This is precisely the claimed lower bound. This also shows that our vector sparsification algorithm from Theorem 2.2 is optimal for any $T \leq n/C$. To prove Theorem 3.1, we first show the existence of a matrix $B \in \{-1,+1\}^{n\times n}$ having the second property. We then apply Spencer's Theorem (Theorem 2.3) to "transform" B into a matrix A having both properties. We find it surprising that Spencer's result finds applications in both our upper and lower bound.

That a matrix satisfying the second property in Theorem 3.1 exists is expressed in the following lemma:

Lemma 3.1. *There exists a universal constant $C > 0$ such that for all sufficiently large n and all T with $\ln n \leq T \leq n/C$, there exists a matrix $A \in \{-1,+1\}^{n\times n}$ such that for every vector $w \in \mathbb{R}^n$ with $\|w\|_0 \leq T$ and $\|w\|_1 = 1$ it holds that: $\min_i(Aw)_i \leq -\Omega\left(\sqrt{\ln(n/T)/T}\right)$.*

We prove Lemma 3.1 in the full version (Grønlund et al., 2019), and move on to show how we use it in combination with Spencer's Theorem to prove Theorem 3.1:

Proof. Let B be a matrix satisfying the statement in Lemma 3.1. Using Spencer's Theorem (Theorem 2.3), we get that there exists a vector $x \in \{-1,+1\}^n$ such that $\|Bx\|_\infty = O(\sqrt{n\ln(en/n)}) = O(\sqrt{n})$. Now form the matrix A which is equal to B, except that the i'th column is scaled by x_i. Then $A\mathbf{1} = Bx$ where $\mathbf{1}$ is the all-ones vectors. Normalizing the all-ones vector by a factor $1/n$ yields the vector v with all coordinates equal to $1/n$. Moreover, it holds that $\|Av\|_\infty = \|Bx\|_\infty/n = O(1/\sqrt{n})$, which in turn implies that $\min_i(Av)_i \geq -O(1/\sqrt{n})$.

Now consider any vector $w \in \mathbb{R}^n$ with $\|w\|_0 \leq T$ and $\|w\|_1 = 1$. Let $\tilde{w}$ be the vector obtained from w by multiplying w_i by x_i. Then $Aw = B\tilde{w}$. Furthermore $\|\tilde{w}\|_1 = \|w\|_1 = 1$ and $\|\tilde{w}\|_0 = \|w\|_0 \leq T$. It follows from Lemma 3.1 and our choice of B that $\min_i(Aw)_i = \min_i(B\tilde{w})_i \leq -\Omega\left(\sqrt{\ln(n/T)/T}\right)$. □

We sketch the proof of Lemma 3.1 here: At a high level, our proof goes as follows: First argue that for a fixed vector $w \in \mathbb{R}^n$ with $\|w\|_0 \leq T$ and $\|w\|_1 = 1$ and a random matrix $A \in \{-1,+1\}^{n\times n}$ with each coordinate chosen uniformly and independently, it holds with very high probability that Aw has many coordinates that are less than $-a_1(\sqrt{\ln(n/T)/T})$ for a constant a_1. Now intuitively we would like to union bound over all possible vectors w and argue that with non-zero probability, all of them satisfies this simulatenously. This is not directly possible as there are infinitely many w. Instead, we create a *net* W consisting of a collection of carefully chosen vectors. The net has the property that any w with $\|w\|_1 = 1$ and $\|w\|_0 \leq T$ is close to a vector $\tilde{w} \in W$. Since the net is not too large, we can union bound over all vectors in W and find a matrix A with the above property for all vectors in W simultaneously.

For an arbitrary vector w with $\|w\|_1 = 1$ and $\|w\|_0 \leq T$, we can then write $Aw = A\tilde{w} + A(w - \tilde{w})$ where $\tilde{w} \in W$ is close to w. Since $\tilde{w} \in W$ we get that $A\tilde{w}$ has many coordinates that are less than $-a_1(\sqrt{\ln(n/T)/T})$. The problem is that $A(w - \tilde{w})$ might cancel out these negative coordinates. However, $w - \tilde{w}$ is a very short vector so this seems unlikely. To prove it formally, we show that for every vector $v \in W$, there are also few coordinates in Av that are greater than $a_2(\sqrt{\ln(n/T)/T})$ in absolute value for some constant $a_2 > a_1$. We can then round $(w-\tilde{w})/\|w-\tilde{w}\|_1$ to a vector in the net, apply this reasoning and recurse on the difference between $(w - \tilde{w})/\|w - \tilde{w}\|_1$ and the net vector. See the full version (Grønlund et al., 2019) for details.

4. Experiments

Gradient Boosting (Mason et al., 2000; Friedman, 2001) is probably the most popular boosting algorithm in practice. It has several highly efficient open-source implementations (Chen & Guestrin, 2016; Ke et al., 2017; Prokhorenkova et al., 2018) and obtain state-of-the-art performance in many machine learning tasks (Ke et al., 2017). In this section we demonstrate how our sparsification algorithm can be combined with Gradient Boosting. For simplicity we consider a single dataset in this section, the Flight Delay dataset (air), see the full version (Grønlund et al., 2019) for similar results on other dataset.

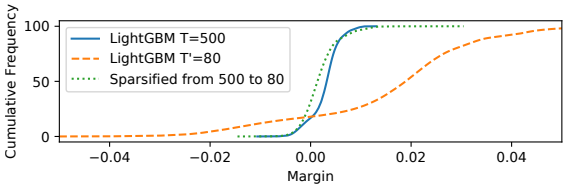

Figure 1. The plot depicts the cumulative margins of three classifiers: (1) a LightGBM classifier with 500 hypotheses (2) a classifier sparsified from 500 to 80 hypotheses and (3) a LightGBM classifier with 80 hypotheses.

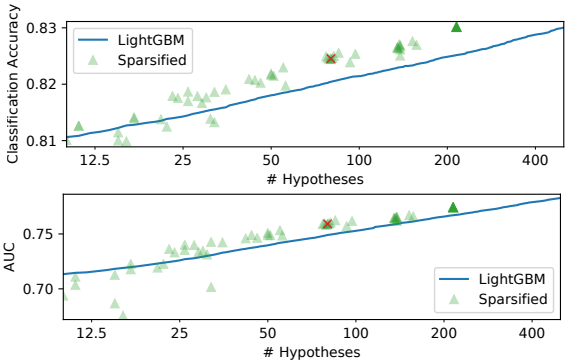

Figure 2. The plot depicts test AUC and test classification accuracy of a LightGBM classifier during training as the number of hypotheses increase (in blue). Notice the x-axis is logarithmically scaled. The final classifier with 500 hypotheses was sparsified with Theorem 2.2 multiple times to have between $T/2$ to $T/16$ hypotheses. The green triangles show test AUC and test accuracy of the resulting sparsified classifiers. The red cross represents the sparsified classifier used to plot the cumulative margins in Figure 1.

We train a classifier with $T = 500$ hypotheses using LightGBM (Ke et al., 2017) which we sparsify using Theorem 2.2 to have $T' = 80$ hypotheses. The sparsified classifier is guaranteed to preserve all margins of the original classifier to an additive $O(\sqrt{\lg(n/T')/T'})$. The cumulative margins of the sparsified classifier and the original classifier are depicted in Figure 1. Furthermore, we also depict the cumulative margins of a LightGBM classifier trained to have $T' = 80$ hypotheses. First observe the difference between the LightGBM classifiers with $T = 500$ and $T' = 80$ hypotheses (blue and orange in Figure 1). The margins of the classifier with $T = 500$ hypotheses vary less. It has fewer points with a large margin, but also fewer points with a small margin. The margin distribution of the sparsified classifier with $T' = 80$ approximates the margin distribution of the LightGBM classifier with $T = 500$ hypotheses. Inspired by margin theory one might suspect this leads to better generalization. To investigate this, we performed additional experiments computing AUC and classification accuracy of several sparsified classifiers and LightGBM classifiers on a test set (we show the results for multiple sparsified classifiers due to the randomization in the discrepancy minimization algorithms). The experiments indeed show that the sparsified classifiers outperform the LightGBM classifiers with the same number of hypotheses. See Figure 2 for test AUC and test classification accuracy.

Further Experiments and Importance Sampling. Inspired by the experiments in (Wang et al., 2008; Ke et al., 2017; Chen & Guestrin, 2016) we also performed the above experiments on the Higgs (Whiteson, 2014) and Letter (Dheeru & Karra Taniskidou, 2017) datasets. See the full version (Grønlund et al., 2019) for (1) further experimental details and (2) for cumulative margin, AUC and test accuracy plots on all dataset for different values of n and T and (3) a comparison that shows SparsiBoost obtain equal or better minimal margin compared to AdaBoost and AdaBoostV in the experimental setting of (Reyzin & Schapire, 2006).

As mentioned in Section 2, one could use importance sampling for sparsification. It has a slightly worse theoretical guarantee, but might work better in practice. The full version (Grønlund et al., 2019) also contains test AUC and test accuracy of the classifiers that result from using importance sampling instead of our algorithm based on discrepancy minimization. Our algorithm and importance sampling are both random so the experiments were repeated several times. On average over the experiments, our algorithm outperforms importance sampling.

A Python implementation of Theorem 2.2 can be found at: `https://github.com/AlgoAU/DiscMin`

5. Conclusion

A long line of research into obtaining a large minimal margin using few hypotheses (Breiman, 1999; Grove & Schuurmans, 1998; Bennett et al., 2000; Rätsch & Warmuth, 2002) culminated with the AdaBoostV (Rätsch & Warmuth, 2005) algorithm. AdaBoostV was later conjectured by (Nie et al., 2013) to provide an optimal trade-off between minimal margin and number of hypotheses. In this article, we introduced SparsiBoost which refutes the conjecture of (Nie et al., 2013). Furthermore, we show a matching lower bound, which implies that SparsiBoost is optimal.

The key idea behind SparsiBoost, is a sparsification algorithm that reduces the number of hypotheses while approximately preserving the entire margin distribution. Experimentally, we combine our sparsification algorithm with LightGBM. We find that the sparsified classifiers obtains a better margin distribution, which typically yields a better test AUC and test classification error when compared to a classifier trained directly to the same number of hypotheses.

References

Flight delay data. https://github.com/szilard/benchm-ml#data.

Bennett, K. P., Demiriz, A., and Shawe-Taylor, J. A column generation algorithm for boosting. In *ICML*, pp. 65–72, 2000.

Breiman, L. Prediction games and arcing algorithms. *Neural computation*, 11(7):1493–1517, 1999.

Chen, T. and Guestrin, C. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pp. 785–794. ACM, 2016.

Dheeru, D. and Karra Taniskidou, E. UCI machine learning repository. http://archive.ics.uci.edu/ml, 2017.

Freund, Y. and Schapire, R. E. A desicion-theoretic generalization of on-line learning and an application to boosting. In *European conference on computational learning theory*, pp. 23–37. Springer, 1995.

Freund, Y., Schapire, R., and Abe, N. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

Friedman, J. H. Greedy function approximation: a gradient boosting machine. *Annals of statistics*, pp. 1189–1232, 2001.

Gao, W. and Zhou, Z.-H. On the doubt about margin explanation of boosting. *Artificial Intelligence*, 203:1–18, 2013.

Grønlund, A., Larsen, K. G., and Mathiasen, A. Optimal minimal margin maximization with boosting. *arXiv preprint arXiv:1901.10789*, 2019.

Grove, A. J. and Schuurmans, D. Boosting in the limit: Maximizing the margin of learned ensembles. In *AAAI/IAAI*, pp. 692–699, 1998.

Guo, H., Liu, H., Li, R., Wu, C., Guo, Y., and Xu, M. Margin & diversity based ordering ensemble pruning. *Neurocomputing*, 275:237 – 246, 2018. ISSN 0925-2312. doi: https://doi.org/10.1016/j.neucom.2017.06.052. URL http://www.sciencedirect.com/science/article/pii/S0925231217311803.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., and Liu, T.-Y. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems*, pp. 3146–3154, 2017.

Klein, P. and Young, N. On the number of iterations for dantzig-wolfe optimization and packing-covering approximation algorithms. *Lecture Notes in Computer Science*, 1610:320–327, 1999.

Koltchinskii, V., Panchenko, D., and Lozano, F. Some new bounds on the generalization error of combined classifiers. In *Advances in neural information processing systems*, pp. 245–251, 2001.

Larsen, K. G. Constructive discrepancy minimization with hereditary L2 guarantees. *CoRR*, abs/1711.02860, 2017. URL http://arxiv.org/abs/1711.02860.

Lovett, S. and Meka, R. Constructive discrepancy minimization by walking on the edges. *SIAM Journal on Computing*, 44(5):1573–1582, 2015.

Margineantu, D. D. and Dietterich, T. G. Pruning adaptive boosting. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pp. 211–218, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc. ISBN 1-55860-486-3. URL http://dl.acm.org/citation.cfm?id=645526.757762.

Mason, L., Baxter, J., Bartlett, P. L., and Frean, M. R. Boosting algorithms as gradient descent. In Solla, S. A., Leen, T. K., and Müller, K. (eds.), *Advances in Neural Information Processing Systems 12*, pp. 512–518. MIT Press, 2000.

Nie, J., Warmuth, M., Vishwanathan, S., and Zhang, X. Open problem: Lower bounds for boosting with hadamard matrices. *Journal of Machine Learning Research*, 30:1076–1079, 01 2013.

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. Catboost: unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems*, pp. 6637–6647, 2018.

Qian, C., Yu, Y., and Zhou, Z.-H. Pareto ensemble pruning. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, AAAI'15, pp. 2935–2941. AAAI Press, 2015. ISBN 0-262-51129-0. URL http://dl.acm.org/citation.cfm?id=2888116.2888125.

Rätsch, G. and Warmuth, M. K. Maximizing the margin with boosting. In *COLT*, volume 2375, pp. 334–350. Springer, 2002.

Rätsch, G. and Warmuth, M. K. Efficient margin maximizing with boosting. *Journal of Machine Learning Research*, 6(Dec):2131–2152, 2005.

Reyzin, L. and Schapire, R. E. How boosting the margin can also boost classifier complexity. In *Proceedings of the 23rd international conference on Machine learning*, pp. 753–760. ACM, 2006.

Schapire, R. E., Freund, Y., Bartlett, P., Lee, W. S., et al. Boosting the margin: A new explanation for the effectiveness of voting methods. *The annals of statistics*, 26(5): 1651–1686, 1998.

Spencer, J. Six standard deviations suffice. *Transactions of the American mathematical society*, 289(2):679–706, 1985.

Wang, L., Sugiyama, M., Yang, C., Zhou, Z.-H., and Feng, J. On the margin explanation of boosting algorithms. In *COLT*, pp. 479–490. Citeseer, 2008.

Whiteson, D. Higgs data set. https://archive.ics.uci.edu/ml/datasets/HIGGS, 2014.