ID3 Learns Juntas for Smoothed Product Distributions

Alon Brtuzkus   ALONBRUTZKUS@MAIL.TAU.AC.IL
The Blavatnik School of Computer Science, Tel Aviv University, Israel.

Amit Daniely   AMIT.DANIELY@MAIL.HUJI.AC.IL
School of Computer Science, The Hebrew University, Israel, and Google Research Tel Aviv.

Eran Malach   ERAN.MALACH@MAIL.HUJI.AC.IL
School of Computer Science, The Hebrew University, Israel.

Abstract

In recent years, there are many attempts to understand popular heuristics. An example of such heuristic algorithm is the ID3 algorithm for learning decision trees. This algorithm is commonly used in practice, but there are very few theoretical works studying its behavior. In this paper, we analyze the ID3 algorithm, when the target function is a k-Junta, a function that depends on k out of n variables of the input. We prove that when $k = \log n$, the ID3 algorithm learns in polynomial time k-Juntas, in the smoothed analysis model of Kalai and Teng (2008). That is, we show a learnability result when the observed distribution is a "noisy" variant of the original distribution.

Keywords: Decision Trees, ID3, Parity, Junta

1. Introduction

In recent years there has been a growing interest in analyzing machine learning algorithms that are commonly used in practice. A primary example is the gradient-descent algorithm for learning neural-networks, which achieves remarkable performance in practice but has very little formal guarantees. A main approach in studying such algorithms is proving that they are able to learn models that are known to be learnable. For examples, it has been shown that SGD can learn neural-networks when the target function is linear, or belongs to a certain kernel space (Brutzkus et al., 2017; Xie et al., 2016; Daniely, 2017; Du et al., 2018; Oymak and Soltanolkotabi, 2018; Allen-Zhu et al., 2018a,b; Arora et al., 2019; Oymak and Soltanolkotabi, 2019; Ma et al., 2019; Lee et al., 2019).

In this paper we take a similar approach aiming to give theoretical guarantees for the ID3 algorithm (Quinlan, 1986) - a popular algorithm for learning decision trees. We analyze the behavior of this algorithm when the target function is a k-Junta, a function that depends only on k bits from the input, and the underlying distribution is a product distribution, where the bits in the input examples are independent. While we cannot guarantee that the ID3 algorithm learns Juntas under any such distribution, as there are distributions which fail the algorithm, we show that the algorithm can learn "most" product distributions. That is, we show that for any product distribution and a k-Junta, the ID3 algorithm learns the junta over a "noisy" variant of the original distribution. Such a result is in the spirit of smoothed analysis (Spielman and Teng, 2004), which is often used to give results when a worst-case analysis is not satisfactory.

Related Work There are a number of works studying the learnability of decision trees (Rivest, 1987; Kushilevitz and Mansour, 1993; Blum, 1992; Ehrenfeucht and Haussler, 1989; Bshouty and Feldman, 2002; Bshouty and Burroughs, 2003; Bshouty et al., 2005). We next elaborate on papers that analyze decision trees under product distributions, as we do. The work of (Kalai and Teng, 2008) gives learnability results of decision trees for product distributions with smoothed analysis, in a problem setting similar to ours. Their work analyzes an algorithm that estimates the Fourier coefficients of the target function in order to learn the decision tree. Another work (O'Donnell and Servedio, 2007) proves learnability of decision trees implementing monotone Boolean functions under the uniform distribution. Other algorithms for learning decision trees under the uniform distribution are given in Jackson and Servedio (2003) and Hazan et al. (2017), again relying on Fourier analysis of the target function. Another work by Chen and Moitra (2018) gives an algorithm for learning stochastic decision trees under the uniform distribution. The work of Blum et al. (1994) gives negative results on learning polynomial size decision trees under the uniform distribution in the statistical query setting.

While the above works study learnability of decision trees under various distributional assumptions, they all consider algorithms that are very different from algorithms used in practice. Our work, on the contrary, gives guarantees for algorithms that enjoy empirical success. In the current literature there are very few works that analyze such algorithms. Notably, the work of Fiat and Pechyony (2004) studies the class of impurity-based algorithms, which contains the ID3 algorithm. This work shows that unate functions, like linear threshold function or read-once DNF, are learnable under the uniform distribution, using impurity-based algorithms. Our work, on the other hand, considers a different choice of target functions (Juntas), and shows learnability under "most" distributions, and not only for a fixed distribution. Another work that studies an algorithm used in practice (Kearns and Mansour, 1999) shows that the CART and C4.5 algorithms can leverage weak approximation of the target function, and thus can perform boosting. However, it is not clear whether such weak approximation typically happens, and in what cases this result can be applied. In contrast, our results apply for a concrete family of functions and distributions.

1.1. Problem Setting

The ID3 Algorithm Let $\mathcal{X} = \{0, 1\}^n$ be the domain set and let $\mathcal{Y} = \{0, 1\}$ be the label set. We next describe the ID3 algorithm, following the presentation in Shalev-Shwartz and Ben-David (2014). Define an impurity function C to be any concave function $C : [0, 1] \to \mathbb{R}$, satisfying that $C(x) = C(1 - x)$ and $C(0) = C(1) = 0$. Given an impurity function C, a sample $S \subset \mathcal{X} \times \mathcal{Y}$ and an index $i \in [n]$, we define the gain measure to be as follows:

$$\text{Gain}(S, i) = C(\mathbb{P}_S [y = 1])$$
$$- (\mathbb{P}_S [x_i = 1] \, C(\mathbb{P}_S [y = 1 | x_i = 1]) + \mathbb{P}_S [x_i = 0] \, C(\mathbb{P}_S [y = 1 | x_i = 0]))$$

Given a sample $S \subseteq \mathcal{X} \times \mathcal{Y}$, the ID3 algorithm generates a decision tree in a recursive manner. At each step of the recursion, the algorithm chooses the feature x_j to be assigned to a current node. The algorithm iterates over all the unused features, and calculates the gain measure with respect to the examples that reach the current node. Then, it chooses the feature that maximizes the gain. This algorithm is described formally in algorithm 1. The output of the algorithm is given by the initial call to ID3$(S, [n])$.

Algorithm 1 ID3(S, A)

---
input:
  Training set $S \subset \mathcal{X} \times \mathcal{Y}$
  Feature subset $A \subseteq [n]$
if all examples in S have the same label $y \in \mathcal{Y}$ **then**
  return a leaf with label y
else
  Let $j = \arg\max_{i \in A} \text{Gain}(S, i)$
  Let T_1 be the tree returned by ID3$(\{(\boldsymbol{x}, y) \in S \; : \; x_j = 1\}, A \setminus \{j\})$
  Let T_2 be the tree returned by ID3$(\{(\boldsymbol{x}, y) \in S \; : \; x_j = 0\}, A \setminus \{j\})$
  Return the a tree with root x_j, whose left and right sub-trees are T_2 and T_1
end if

Learning Juntas A k-Junta is a function $f : \{0, 1\}^n \to \{0, 1\}$ that depends on k coordinates. Namely, there is a set $J = \{i_1 < i_2 < \ldots < i_k\} \subset [n]$ and a function $\tilde{f} : \{0, 1\}^k \to \{0, 1\}$ such that $f(\boldsymbol{x}) = \tilde{f}(x_{i_1}, \ldots, x_{i_k})$. In this case, we will say that f is supported in J. Throughout the paper, we assume that the examples are sampled from a **product** distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, that is realizable by a $\log(n)$-Junta. Namely, we assume that for $(\boldsymbol{x}, y) \sim \mathcal{D}$, $\boldsymbol{x} \sim \prod_{i=1}^{n} \text{Bernoulli}(p_i)$ for some $p_1, \ldots, p_n \in [0, 1]$, and $y = f(\boldsymbol{x})$ for some $\log(n)$-Junta f. The main goal of this paper is to show that for "most" product distributions, the ID3 algorithm succeeds to learn $\log(n)$-Juntas in polynomial time. Namely, it will return a tree T whose generalization error, $\mathcal{L}_{\mathcal{D}}(T) := \Pr_{(\boldsymbol{x}, y) \sim \mathcal{D}}(T(\boldsymbol{x}) \neq y)$, is small (in fact, zero). We note that the sample complexity of learning $\omega(\log(n))$-Juntas is super polynomial, hence, $\log(n)$-Juntas is the best that we can hope to learn in polynomial time.

1.2. Results

We will show two positive results for learning $\log(n)$-Juntas. The first establishes learnability of parities, while the second is about learnability of general Juntas. Throughout, we assume that the impurity function C is strongly concave and Lipschitz.

Learning Parities A k-*parity* is a function of the form $\chi_J(\boldsymbol{x}) = \begin{cases} 1 & \sum_{i \in J} x_i \text{ is odd} \\ 0 & \sum_{i \in J} x_i \text{ is even} \end{cases}$, where $J \subset [n]$ is a set of k indices. Note that any k-parity is a k-Junta. We first consider learnability of $\log(n)$-parities by the ID3 algorithm[1].

  Learning parity functions is a classical problem in machine learning, for which there exists an efficient algorithm (Feldman et al., 2006, 2009). Still, parities often serve as a hard benchmark, as many common algorithms cannot learn these functions (Blum et al., 2003; Shalev-Shwartz et al., 2017). In the case of the ID3 algorithm, when the underlying distribution is uniform (i.e, when $p_i = \frac{1}{2}$ for all $i \in [n]$), the algorithm fails to learn parity functions (Kearns, 1996). We show that the case of the uniform distribution is in some sense exceptional. That is, we show that for every distribution that is not "too close" to the uniform distribution, the ID3 algorithm succeeds to

---

1. As opposed to general k-Juntas, k-parities with any k are learnable in polynomial time. Yet, in the context of decision tree algorithms, we cannot hope to learn k-parities with $k = \omega(\log(n))$. Indeed, such parities cannot be computed, or even approximated, by a poly-sized tree.

learn any such parity function. To this end, we say that $\mathcal{D}$ is (α, c)-distribution if $\left| p_i - \frac{1}{2} \right| > c$ and $p_i \in (\alpha, 1 - \alpha)$ for all $i \in [n]$.

Theorem 1 *Fix $\alpha, c > 0$. There is a polynomial[2] p for which the following holds. Suppose that the ID3 algorithm runs on $p\left(n, \log\left(\frac{1}{\delta}\right)\right)$ examples from an (α, c)-distribution $\mathcal{D}$ that is realized by a $\log(n)$-parity. Then, w.p. $\geq 1 - \delta$, ID3 will output a tree T with $\mathcal{L}_{\mathcal{D}}(T) = 0$.*

Smoothed Analysis of Learning General Juntas   In the case of parities, we show that there is a single "bad" distribution, namely the uniform distribution, which fails the ID3 algorithm. For general Juntas, however, there might be various "bad" distributions, which depend on the specific choice of Junta. Still, we can show that the algorithm learns "most" product distributions, following the *smoothed analysis* framework. Namely, for every fixed distribution, we show that the algorithm succeeds to learn, with high probability, a "noisy" version of this distribution. Formally, a *smoothened (α, c)-distribution* $\mathcal{D}$ is a *random* distribution where $p_i = \hat{p}_i + \Delta_i$ for some fixed $\hat{p}_i \in (\alpha + c, 1 - \alpha - c)$ and $\Delta_i \sim_{i.i.d} Uni([-c, c])$.

Theorem 2 *Fix $\alpha, c > 0$. There is a polynomial[3] p for which the following holds. Suppose that the ID3 algorithm runs on $p\left(n, \frac{1}{\delta}\right)$ examples from a smoothened (α, c)-distribution $\mathcal{D}$ that is realized by a $\log(n)$-junta. Then, w.p. $\geq 1 - \delta$, ID3 will output a tree T with $\mathcal{L}_{\mathcal{D}}(T) = 0$.*

1.3. Open Question

We now turn to discussing possible open questions and future directions arising from this work. Our main result applies for the case where the target function is a k-Junta, which can be implemented by a tree of depth $k = \log n$. An immediate open question is whether a similar learnability result can be shown for general trees of depth $\log n$. We conjecture that this is indeed the case.

Conjecture 3 *Fix $\alpha, c > 0$. There is a polynomial p for which the following holds. Suppose that the ID3 algorithm runs on $p\left(n, \frac{1}{\delta}, \frac{1}{\epsilon}\right)$ examples from a smoothened (α, c)-distribution $\mathcal{D}$ that is realized by a $\log(n)$-depth-tree. Then, w.p. $\geq 1 - \delta$, ID3 will output a tree T with $\mathcal{L}_{\mathcal{D}}(T) \leq \epsilon$.*

As we previously mentioned, our work could be viewed in a broader context of understanding heuristic learning algorithms that enjoy empirical success. In this field of research, a main challenge of the machine learning community is to understand the behavior of neural-networks learned with gradient-based algorithms. While our analysis is focused on proving results for the ID3 algorithm, we believe that similar techniques could be used to show similar results for learning neural-networks with gradient-descent. Specifically, we raise the following interesting question:

Open Question 1 *Can gradient-descent learn neural-networks when the target function is a k-Junta, in the smoothed analysis setting?*

---

2. The polynomial p depends on α, c and the impurity function C. See theorem 10 for a detailed dependency.
3. The polynomial p again depends on α, c and the impurity function C. See theorem 12 for a detailed dependency.

2. Proofs

2.1. General Approach

Throughout, we assume that $\mathcal{D}$ is a distribution that is realized by a Junta f, supported in $J \subset [n]$, with $|J| = k$. We assume w.l.o.g. that $J = [k]$.

   To prove our result, we will show that w.h.p., the algorithm chooses only variables from $[k]$, and furthermore, any root-to-leaf path will contain all the variables from $[k]$. In this case, the resulting tree will have zero generalization error. To formalize this, we will use the following notation. We define the support of a vector $w \in \{*, 0, 1\}^n$ as

$$\text{supp}(w) = \{i \in [n] : w_i \neq *\}$$

and let

$$\mathcal{X}_w = \{x \in \mathcal{X} : x_i = w_i \text{ for any } i \in \text{supp}(w)\}$$

For a sample $S \subseteq \mathcal{X} \times \mathcal{Y}$, we denote

$$S_w = \{(x, y) \in S \; : \; x \in \mathcal{X}_w\}$$

Finally, for a distribution $\mathcal{D}$ we denote $\mathcal{D}_w = \mathcal{D}|_{x \in \mathcal{X}_w}$

Lemma 4 *Suppose that the sample S is realized by a Junta f. Assume that for any $w \in \{0, 1, *\}^n$ with $supp(w) \subset J$ we have $S_w \neq \emptyset$ and one of the following holds:*

- *All examples in S_w have the same label.*

- *There exists $i \in J \setminus supp(w)$ s.t. for all $j \in [n] \setminus J$ we have $Gain(S_w, i) > Gain(S_w, j)$.*

Then, the ID3 algorithm will build a tree with zero loss on $\mathcal{D}$.

   Not surprisingly, the gain of coordinates outside of J is always small. This is formalized in the following lemma.

Lemma 5 *Assume that C is γ-Lipschitz. Fix $w \in \{0, 1, *\}^n$, with $|supp(w)| \leq k$, $j \in [n] \setminus J$ and $\epsilon, \delta \in (0, 1)$. Assume we sample $S \sim \mathcal{D}^m$ with $m \gtrsim \epsilon^{-2}\alpha^{-2k} \log(\frac{1}{\delta})$. Then with probability at least $1 - \delta$ we have $S_w \neq \emptyset$ and:*

$$Gain(S_w, j) < 2\gamma\epsilon$$

Given lemma 5, in order to apply lemma 4, it remains to show that the gain of the coordinates in J is large. To this end, we will use a measure of dependence between a coordinate x_i and the label y, which we define next. For a sample $S \subset \mathcal{X} \times \mathcal{Y}$ and an index $i \in [n]$, we denote the empirical covariance between x_i and the label y by:

$$\mathcal{I}(S, i) := \mathbb{E}_S[yx_i] - \mathbb{E}_S[y]\mathbb{E}_S[x_i]$$

Similarly, for a distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$ we denote the covariance between x_i and y over the distribution $\mathcal{D}$ by:

$$\mathcal{I}(\mathcal{D}, i) = \mathbb{E}_\mathcal{D}[yx_i] - \mathbb{E}_\mathcal{D}[y]\mathbb{E}_\mathcal{D}[x_i]$$

Note that x_j and y are independent if and only if $\mathcal{I}(\mathcal{D}, i) = 0$. The following lemma connects $Gain(S_w, i)$ to $\mathcal{I}(\mathcal{D}_w, i)$.

Lemma 6 *Assume C is β strongly concave (i.e, $-C$ is β strongly convex). Assume for some $w \in \{0, 1, *\}^n$, with $supp(w) \leq k$ and index $i \in [n]$ we have $|\mathcal{I}(\mathcal{D}_w, i)| \geq \epsilon > 0$. Fix $\delta > 0$. Then, if we sample $S \sim \mathcal{D}^m$ for $m \gtrsim \epsilon^{-2}\alpha^{-2k}\log(\frac{1}{\delta})$, then with probability at least $1 - \delta$ we have $S_w \neq \emptyset$ and:*

$$Gain(S_w, i) \geq \frac{\beta\epsilon^2}{8}$$

Combining lemmas 4, 5 and 6, we get the following theorem:

Theorem 7 *Assume C is β strongly concave and γ-Lipschitz. Assume for any $w \in \{0, 1, *\}^n$, with $supp(w) \subset J$ we have $S_w \neq \emptyset$ and either of the following holds:*

- *All examples in $\mathcal{D}_w$ have the same label.*

- *There exists some index $i \in J \setminus supp(w)$ with $|\mathcal{I}(\mathcal{D}_w, i)| \geq \epsilon > 0$.*

Fix $\delta > 0$. Then, if we sample $S \sim \mathcal{D}^m$ for $m \gtrsim \beta^{-2}\gamma^2\epsilon^{-4}\alpha^{-2k}k\log(\frac{n}{\delta})$, then with probability at least $1 - \delta$ the ID3 algorithm will build a tree with zero loss on $\mathcal{D}$

By the above theorem, in order to show that the ID3 algorithm succeeds in learning, it is enough to lower bound $|\mathcal{I}(\mathcal{D}_w, i)|$. This is done in the remaining sections, together with the proofs of lemmas 4, 5 and 6.

2.2. Proof of the basic lemmas

Proof (of lemma 4) At every iteration, the ID3 algorithm assigns a splitting variable for a given node, or otherwise returns a leaf for this node. We will show that for every node that the algorithm iterates on, if the path from the root to this node contains only variables from J, then either the algorithm adds a splitting variable from J, or the algorithm returns a leaf. Indeed, assume that the path from the root to this node contains only variables from J. We can decode the root-to-node path by a vector $w \in \{*, 0, 1\}^n$, where $w_i = 1$ if the node $x_i = 1$ is in the path, $w_i = 0$ if the node $x_i = 0$ is in the path, and $w_i = *$ otherwise. Therefore, by our assumption we have $supp(w) \subseteq J$. Note that in this case, the algorithm observes the sample S_w, so if all examples in S_w have the same label, then the algorithm returns a leaf. Otherwise, by the assumption we get $\arg\max_{i \in A} Gain(S_w, i) \in J$, so the algorithm chooses a splitting variable from J.

From the above, the algorithm adds only splitting variables from J, so it can build a tree of size at most 2^k before stopping. This tree has zero loss on the distribution. Indeed, for any $x' \in \{0, 1\}^k$, denote $w(x') \in \{0, 1\}^n$ such that $w(x')_i = x'_i$ for every $i \in [k]$ and $w(x')_i = *$ for every $i \notin [k]$. Then, since we assume $S_{w(x')} \neq \emptyset$, there exists a sample $(x, y) \in S$ such that $x_i = x'_i$ for every $i \in [k]$. By definition, the algorithm returns a tree that correctly labels the example x, therefore it returns a function that agrees with $f(x) = \tilde{f}(x')$. Since this is true for every choice of $x' \in \{0, 1\}^k$, the function returned by the tree agrees with the Junta defined by $\tilde{f}$, so it gets zero loss. ∎

We next relate the empirical measure $\mathcal{I}(S, i)$ to $\mathcal{I}(\mathcal{D}, i)$.

Lemma 8 *Fix $w \in \{0, 1, *\}^n$, with $supp(w) \leq k$, $i \in [n]$, $\epsilon, \delta \in (0, 1)$. Let $S \sim \mathcal{D}^m$ with $m \gtrsim \alpha^{-2k}\epsilon^{-2}\log(\frac{1}{\delta})$. Then with probability at least $1 - \delta$ we have $S_w \neq \emptyset$ and:*

$$|\mathcal{I}(S_w, i) - \mathcal{I}(\mathcal{D}_w, i)| < \epsilon$$

Proof Denote $S = \{(\boldsymbol{x}_1, y), \ldots, (\boldsymbol{x}_m, y)\}$. Let $\bar{p}_i = \mathbb{E}_{S_w}[x_i]$ and $p_w = \Pr_{x \sim \mathcal{D}}(x \in \mathcal{X}_w)$. We have

$$\mathbb{E}_{S_w}[y] = \frac{\sum_{j=1}^m 1[\boldsymbol{x}_j \in \mathcal{X}_w] y_j}{\sum_{j=1}^m 1[\boldsymbol{x}_j \in \mathcal{X}_w]} = \frac{\frac{\sum_{j=1}^m 1[\boldsymbol{x}_j \in \mathcal{X}_w] y_j}{p_w m}}{\frac{\sum_{j=1}^m 1[\boldsymbol{x}_j \in \mathcal{X}_w]}{p_w m}}$$

By Hoeffding's bound, with probability $\geq 1 - \frac{\delta}{3}$, we have

$$\left| p_w \mathbb{E}_{\mathcal{D}_w} y - \frac{\sum_{j=1}^m 1[\boldsymbol{x}_j \in \mathcal{X}_w] y_j}{m} \right| \lesssim \epsilon \alpha^k \text{ and } \left| p_w - \frac{\sum_{j=1}^m 1[\boldsymbol{x}_j \in \mathcal{X}_w]}{m} \right| \lesssim \epsilon \alpha^k$$

dividing by p_w we get

$$\left| \mathbb{E}_{\mathcal{D}_w} y - \frac{\sum_{j=1}^m 1[\boldsymbol{x}_j \in \mathcal{X}_w] y_j}{p_w m} \right| \lesssim \frac{\epsilon \alpha^k}{p_w} \lesssim \epsilon \text{ and } \left| 1 - \frac{\sum_{j=1}^m 1[\boldsymbol{x}_j \in \mathcal{X}_w]}{p_w m} \right| \lesssim \frac{\epsilon \alpha^k}{p_w} \lesssim \epsilon$$

Notice that from the above we get that $S_w \neq \emptyset$. It follows that

$$|\mathbb{E}_{S_w}[y] - \mathbb{E}_{\mathcal{D}_w} y| \lesssim \epsilon$$

Similarly,

$$|\mathbb{E}_{S_w}[x_i] - \mathbb{E}_{\mathcal{D}_w} x_i| \lesssim \epsilon \text{ and } |\mathbb{E}_{S_w}[yx_i] - \mathbb{E}_{\mathcal{D}_w} yx_i| \lesssim \epsilon$$

In this case, we have $|\mathcal{I}(S_w, i) - \mathcal{I}(\mathcal{D}_w, i)| < \epsilon$ ∎

We next prove lemmas 5 and 6

Proof (of lemma 5) Notice that since x_i and y are independent, we have $\mathcal{I}(\mathcal{D}_w, i) = 0$. By the choice of m, from Lemma 8 we get that with probability $1 - \delta$:

$$|\mathcal{I}(S_w, i)| = |\mathcal{I}(S_w, i) - \mathcal{I}(\mathcal{D}_w, i)| < \epsilon$$

Denote $\bar{p}_i = \mathbb{P}_{S_w}[x_i = 1]$. Notice that if $\bar{p}_i = 0$ or $\bar{p}_i = 1$ then $\mathcal{I}(S_w, i) = 0$, and the result trivially holds. We can therefore assume $\bar{p}_i \in (0, 1)$. Now, we have the following:

$$|\mathbb{P}_{S_w}[y = 1|x_i = 1] - \mathbb{P}_{S_w}[y = 1]| = \left| \frac{\mathbb{P}_{S_w}[y = 1 \wedge x_i = 1] - \mathbb{P}_{S_w}[y = 1]\mathbb{P}_{S_w}[x_i = 1]}{\mathbb{P}_{S_w}[x_i = 1]} \right|$$

$$= \left| \frac{\mathcal{I}(S_w, i)}{\bar{p}_i} \right| < \frac{\epsilon}{\bar{p}_i}$$

Similarly, we get:

$$|\mathbb{P}_{S_w}[y = 1|x_i = 0] - \mathbb{P}_{S_w}[y = 1]| = \left| \frac{\mathbb{P}_{S_w}[y = 1 \wedge x_i = 0] - \mathbb{P}_{S_w}[y = 1]\mathbb{P}_{S_w}[x_i = 0]}{\mathbb{P}_{S_w}[x_i = 0]} \right|$$

$$= \left| \frac{\mathcal{I}(S_w, i)}{1 - \bar{p}_i} \right| < \frac{\epsilon}{1 - \bar{p}_i}$$

Using the γ-Lipschitz property, we get:

$$|C(\mathbb{P}_{S_w}[y = 1|x_i = 1]) - C(\mathbb{P}_{S_w}[y = 1])| \leq \gamma |\mathbb{P}_{S_w}[y = 1|x_i = 0] - \mathbb{P}_{S_w}[y = 1]| < \frac{\gamma \epsilon}{\bar{p}_i}$$

And similarly:
$$|C(\mathbb{P}_{S_w}[y=1|x_i=0]) - C(\mathbb{P}_{S_w}[y=1])| < \frac{\gamma\epsilon}{1-\bar{p}_i}$$

Now plugging into the gain definition:

$$
\begin{aligned}
|Gain(S_w,i)| = & |C(\mathbb{P}_{S_w}[y=1]) - (\mathbb{P}_{S_w}[x_i=1]\,C(\mathbb{P}_{S_w}[y=1|x_i=1]) \\
& + \mathbb{P}_{S_w}[x_i=0]\,C(\mathbb{P}_{S_w}[y=1|x_i=0]))| \\
\leq & \mathbb{P}_{S_w}[x_i=1]\,|C(\mathbb{P}_{S_w}[y=1|x_i=1]) - C(\mathbb{P}_{S_w}[y=1])| \\
& + \mathbb{P}_{S_w}[x_i=0]\,|C(\mathbb{P}_{S_w}[y=1|x_i=0]) - C(\mathbb{P}_{S_w}[y=1])| < 2\gamma\epsilon
\end{aligned}
$$

∎

Proof (of lemma 6) By the choice of m, from Lemma 8 we get that with probability $1-\delta$:

$$|\mathcal{I}(S_w,i) - \mathcal{I}(\mathcal{D}_w,i)| \leq \frac{\epsilon}{2}$$

Since we assume $|\mathcal{I}(\mathcal{D}_w,i)| \geq \epsilon$, we get that $|\mathcal{I}(S_w,i)| \geq \frac{\epsilon}{2}$. Therefore, we have that $\bar{p}_i \in (0,1)$ (again denoting $\bar{p}_i = \mathbb{P}_{S_w}[x_i=1]$). Observe that we have the following:

$$
\begin{aligned}
& \mathbb{P}_{S_w}[x_i=1]\,\mathbb{P}_{S_w}[x_i=0]\,(\mathbb{P}_{S_w}[y=1|x_i=0] - \mathbb{P}_{S_w}[y=1|x_i=0]) \\
= & \mathbb{P}_{S_w}[x_i=0]\,\mathbb{P}_{S_w}[x_i=1 \wedge y=1] - \mathbb{P}_{S_w}[x_i=1]\,\mathbb{P}_{S_w}[x_i=0 \wedge y=1] \\
= & \mathbb{P}_{S_w}[x_i=0]\,\mathbb{P}_{S_w}[x_i=1 \wedge y=1] - \mathbb{P}_{S_w}[x_i=1]\,(\mathbb{P}_{S_w}[y=1] - \mathbb{P}_{S_w}[x_i=1 \wedge y=1]) \\
= & \mathbb{P}_{S_w}[x_i=1 \wedge y=1] - \mathbb{P}_{S_w}[x_i=1]\,\mathbb{P}_{S_w}[y=1] = \mathcal{I}(S_w,i)
\end{aligned}
$$

Therefore, we have:

$$\mathbb{P}_{S_w}[y=1|x_i=1] - \mathbb{P}_{S_w}[y=1|x_i=0] = \frac{\mathcal{I}(S_w,i)}{\bar{p}_i(1-\bar{p}_i)}$$

Since C is β strongly concave we get that for all $a,b,t \in [0,1]$ we have:

$$C(ta + (1-t)b) \geq tC(a) + (1-t)C(b) + \frac{\beta}{2}t(1-t)(a-b)^2$$

Using this property we get that:

$$
\begin{aligned}
& \bar{p}_i C(\mathbb{P}_{S_w}[y=1|x_i=1]) + (1-\bar{p}_i)C(\mathbb{P}_{S_w}[y=1|x_i=0]) \\
\leq & \; C(\mathbb{P}_{S_w}[y=1]) - \frac{\beta}{2}\bar{p}_i(1-\bar{p}_i)(\mathbb{P}_{S_w}[y=1|x_i=1] - \mathbb{P}_{S_w}[y=1|x_i=0])^2 \\
= & \; C(\mathbb{P}_{S_w}[y=1]) - \frac{\beta}{2} \cdot \frac{\mathcal{I}(S_w,i)^2}{\bar{p}_i(1-\bar{p}_i)}
\end{aligned}
$$

Plugging this to the gain equation we get:

$$Gain(S_w,i) \geq \frac{\beta}{2} \cdot \frac{\mathcal{I}(S_w,i)^2}{\bar{p}_i(1-\bar{p}_i)} \geq \frac{\beta}{2}\mathcal{I}(S_w,i)^2$$

Since $|\mathcal{I}(S_w,i)| \geq \frac{\epsilon}{2}$, we get $Gain(S_w,i) \geq \frac{\beta\epsilon^2}{8}$.

∎

2.3. Parities

Lemma 9 *Let $\mathcal{D}$ be a distribution on $\mathcal{X} \times \mathcal{Y}$ labelled by χ_J with $|J| \leq k$. Assume that for every $j \in J$ we have $p_j \in (\alpha, 1 - \alpha)$ and $|p_j - \frac{1}{2}| \geq c$, for some $c, \alpha > 0$. Fix some $w \in \{0, 1, *\}^k$. Then for every $j \in J \setminus supp(w)$ we have:*

$$|\mathcal{I}(\mathcal{D}_w, j)| > \alpha^2 (2c)^{k-1}$$

By theorem 7 we have

Theorem 10 *Let $\mathcal{D}$ be a distribution on $\mathcal{X} \times \mathcal{Y}$ labelled by χ_J with $|J| \leq k$. Assume that for every $j \in J$ we have $p_j \in (\alpha, 1 - \alpha)$ and $|p_j - \frac{1}{2}| \geq c$, for some $c, \alpha > 0$. Assume furthermore that C is β strongly concave and γ-Lipschitz.*

Then, if we sample $S \sim \mathcal{D}^m$ for $m \gtrsim \beta^{-2} \gamma^2 (2c)^{-4k-4} \alpha^{-2k-8} k \log(\frac{n}{\delta})$, then with probability at least $1 - \delta$ the ID3 algorithm will build a tree with zero loss on $\mathcal{D}$

Note that since we assume $k \leq \log n$, the runtime and sample complexity in the above theorem are polynomial in n. We give the proof of this theorem in the rest of this section.

Proof Denote $\epsilon_i := p_i - \frac{1}{2}$, and $k' := |A|$. For simplicity of notation, assume w.l.o.g that $A = [k']$ and $j = k'$. Observe the following:

$$\mathbb{P}_{\mathcal{D}_w}[y = 1] = \mathbb{P}_{\mathcal{D}_w} \left[\prod_{i=1}^{k'} (2x_i - 1) = 1 \right] = \sum_{\prod z_i = 1} \prod_{i=1}^{k'} \mathbb{P}[2x_i - 1 = z_i] = \frac{1}{2} + 2^{k'-1} \prod_{i=1}^{k'} \epsilon_i$$

Similarly, we get that:

$$\mathbb{P}_{\mathcal{D}_w}[y = 1 | x_{k'} = 1] = \sum_{\prod z_i = 1} \prod_{i=1}^{k'-1} \left(\frac{1}{2} + z_i \epsilon_i \right) = \frac{1}{2} + 2^{k'-2} \prod_{i=1}^{k'-1} \epsilon_i$$

Therefore, we get that:

$$|\mathcal{I}(\mathcal{D}_w, j)| = p_j \left| \mathbb{P}_{\mathcal{D}_w}[y = 1] - \mathbb{P}_{\mathcal{D}_w}[y = 1 | x_i = 1] \right|$$

$$= p_j \left| \epsilon_{k'} - \frac{1}{2} \right| \cdot \left(2^{k'-1} \prod_{i=1}^{k'-1} |\epsilon_i| \right) \geq \alpha^2 (2c)^{k'-1} \geq \alpha^2 (2c)^{k-1}$$

$\blacksquare$

2.4. Juntas

Lemma 11 *Fix some $w \in \{*, 0, 1\}^k$, and assume not all examples in $\mathcal{D}_w$ have the same label. Let $A = \{i \in [k] : w_i = *\}$. Assume $p_i \in (\alpha, 1 - \alpha)$ for $\alpha > 0$ for every i, and fix $\delta > 0$. Then there exists $i \in A \cap [k]$ such that with probability $1 - \delta$ over the choice of Δ:*

$$|\mathcal{I}(\mathcal{D}_w, i)| > 2\alpha^2 \delta^2 \left(\frac{c}{2} \right)^{2k}$$

Using a union bound over all choices of w, together with theorem 7 we get:

Theorem 12 *Assume C is β strongly concave and γ-Lipschitz. Fix $\delta_1, \delta_2 > 0$. Then, there exists a universal constant B such that if we sample $S \sim \mathcal{D}^m$ for $m \gtrsim B^k \beta^{-2} \gamma^2 c^{-8k} \delta_1^{-8} \alpha^{-2k-8} k \log(\frac{n}{\delta_2})$, then with probability at least $1 - \delta_1 - \delta_2$ the ID3 algorithm will build a tree with zero loss on $\mathcal{D}$.*

Proof of Lemma 11. For simplicity of notation, we assume w.l.o.g. that $A = [k']$ for some $k' \leq k$. Denote $f_w : \{0,1\}^{k'} \to \{0,1\}$, such that $f_w(x_1, \ldots, x_{k'}) = f(x_1, \ldots, x_{k'}, w_{k'+1}, \ldots, w_k)$. Observe the Fourier coefficients of f_w:

$$f_w(\boldsymbol{x}) = \sum_{I \subset [k']} \alpha_I \chi_I(\boldsymbol{x})$$

Where $\chi_I = \prod_{i \in I}(2x_i - 1)$, and note that χ_I is a Fourier basis (with respect to the uniform distribution). Notice that $|\alpha_I| \geq \frac{1}{2^k}$ for every $\alpha_I \neq 0$. Indeed, we have:

$$\alpha_I = \mathbb{E}_{\boldsymbol{x} \sim U(\{0,1\}^k)}\left[\chi_I(\boldsymbol{x})f(\boldsymbol{x})\right] = \frac{1}{2^k} \sum_{\boldsymbol{x} \in \{0,1\}^k} \chi_I(\boldsymbol{x})f(\boldsymbol{x})$$

where $\chi_I(\boldsymbol{x})f(\boldsymbol{x}) \in \{-1, 0, 1\}$, and this gives the required. Since not all examples in $\mathcal{D}_w$ have the same label, we know that f_w is not a constant function. Therefore, there exists $\emptyset \neq I_0 \subseteq [k']$ such that $\alpha_{I_0} \neq 0$. Fix some $i \in I_0$, and we assume w.l.o.g. that $i = 1$ (so $1 \in I_0$). Now, we can write:

$$f_w(x_1, \ldots, x_{k'}) = (2x_1 - 1)g_w(x_2, \ldots, x_{k'}) + h_w(x_2, \ldots, x_{k'})$$

Where: $g(x_2, \ldots, x_{k'}) = \sum_{I \subset [k'], 1 \in I} \alpha_I \chi_{I \setminus \{1\}}(\boldsymbol{x})$.
  and since $\alpha_{I_0} \neq 0$ and $1 \in I_0$ we get $g \neq 0$. Now, notice that since $\boldsymbol{x} \in \{0,1\}^n$ we get:

$$\mathbb{E}_{\mathcal{D}_w}[f(\boldsymbol{x})] = \mathbb{E}_{\mathcal{D}}[f(\boldsymbol{x})|x_{k'+1} = w_{k'+1}, \ldots, x_k = w_k] = \mathbb{E}_{\mathcal{D}}[f_w(x_1, \ldots, x_{k'})] = f_w(p_1, \ldots, p_{k'})$$

And similarly: $\mathbb{E}_{\mathcal{D}_w}[f(\boldsymbol{x})|x_1 = 1] = f_w(1, p_2, \ldots, p_{k'})$.
  Therefore we get:

$$
\begin{aligned}
\left|\mathbb{E}_{\mathcal{D}_w}[f_w(\boldsymbol{x})x_1] - \mathbb{E}_{\mathcal{D}_w}[f_w(\boldsymbol{x})]\mathbb{E}_{\mathcal{D}_w}[x_1]\right| &= |p_1 f_w(1, p_2, \ldots, p_{k'}) - p_1 f_w(p_1, \ldots, p_{k'})| \\
&= p_1|g_w(p_2, \ldots, p_{k'}) - (2p_1 - 1)g_w(p_2, \ldots, p_{k'})| \\
&= 2p_1(1 - p_1)|g_w(p_2, \ldots, p_{k'})| \\
&= 2p_1(1 - p_1)|g_w(\hat{p}_2 + \Delta_2, \ldots, \hat{p}_{k'} + \Delta_{k'})| \\
&= 2p_1(1 - p_1)|g_0(\Delta_2, \ldots, \Delta_{k'})|
\end{aligned}
$$

Where g_0 is given by:

$$
\begin{aligned}
g_0(\Delta_2, \ldots, \Delta_{k'}) &= g_w(\hat{p}_2 + \Delta_2, \ldots, \hat{p}_{k'} + \Delta_{k'}) \\
&= \sum_{I \subset [k'], 1 \in I} \alpha_I \prod_{i \in I \setminus \{1\}} (2\hat{p}_i + 2\Delta_i - 1) \\
&= \sum_{I \subseteq [k'], 1 \in I} \alpha_I \sum_{I' \subset I} \prod_{i \in I'}(2\hat{p}_i - 1) \prod_{j \notin I'}(2\Delta_j) := \sum_{I \subseteq [k'], 1 \in I} \beta_I \prod_{i \in I \setminus \{1\}} \Delta_i
\end{aligned}
$$

Denote $k_0 = \deg(p_0)$ and note that $k_0 \leq k' - 1$. For some choice of β_I-s. Notice that for some maximal $I \subset [k']$ with $1 \in I$ and $\alpha_I \neq 0$ (so $|I| = k_0$), we have $\beta_I = 2^{|I|}\alpha_I$, so $|\beta_I| \geq \frac{2^{k_0}}{2^{k'}}$.

Now, denote $\xi_i = \frac{1}{c}\Delta_i$, so we have $\xi_i \sim Uni([-1, 1])$, and observe the polynomial:

$$G_0(\xi_2, \ldots, \xi_{k'}) = \frac{2^{k'}}{2^{k_0}c^{k_0-1}}g_0(c\xi_2, \ldots, c\xi_k)$$

$$= \sum_{I \subseteq [k'], 1 \in I} \frac{2^{k'}}{2^{k_0}c^{k_0-1}}\beta_I c^{|I|-1} \prod_{i \in I \setminus \{1\}} \xi_i := \sum_{I \subseteq [k'], 1 \in I} \gamma_I \prod_{i \in I \setminus \{1\}} \xi_i$$

And from what we have shown, G_0 is a polynomial of degree k_0, and there exists I with $|I| = k_0$ such that $|\gamma_I| \geq 1$. Therefore, we can use Lemma 3 from Kalai and Teng (2008) to get that:

$$\mathbb{P}_{\xi \sim Uni([-1,1]^{k'})}\left[|G_0(\xi)| \leq \epsilon\right] \leq 2^{k_0}\sqrt{\epsilon}$$

And therefore:

$$\mathbb{P}_{\Delta \sim Uni([-c,c]^{k'})}\left[|g_0(\Delta)| \leq \epsilon\right] = \mathbb{P}_{\xi \sim Uni([-1,1]^{k'})}\left[|G_0(\xi)| \leq \frac{2^{k'}}{2^{k_0}c^{k_0-1}}\epsilon\right]$$

$$\leq 2^{k_0}\frac{2^{k'/2}}{2^{k_0/2}c^{k'/2-1/2}}\sqrt{\epsilon} \leq \left(\frac{2}{c}\right)^{k'}\sqrt{\epsilon} \leq \left(\frac{2}{c}\right)^{k}\sqrt{\epsilon}$$

So if we take $\epsilon = \delta^2\left(\frac{c}{2}\right)^{2k}$ we get that $\mathbb{P}\left[|g_0| \leq \epsilon\right] \leq \delta$, which completes the proof. $\blacksquare$

References

Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. *arXiv preprint arXiv:1811.04918*, 2018a.

Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via overparameterization. *arXiv preprint arXiv:1811.03962*, 2018b.

Sanjeev Arora, Simon S Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. *arXiv preprint arXiv:1901.08584*, 2019.

Avrim Blum. Rank-r decision trees are a subclass of r-decision lists. *Information Processing Letters*, 42(4):183–185, 1992.

Avrim Blum, Merrick Furst, Jeffrey Jackson, Michael Kearns, Yishay Mansour, and Steven Rudich. Weakly learning dnf and characterizing statistical query learning using fourier analysis. In *STOC*, volume 94, pages 253–262, 1994.

Avrim Blum, Adam Kalai, and Hal Wasserman. Noise-tolerant learning, the parity problem, and the statistical query model. *Journal of the ACM (JACM)*, 50(4):506–519, 2003.

Alon Brutzkus, Amir Globerson, Eran Malach, and Shai Shalev-Shwartz. Sgd learns overparameterized networks that provably generalize on linearly separable data. *arXiv preprint arXiv:1710.10174*, 2017.

Nader H Bshouty and Lynn Burroughs. On the proper learning of axis-parallel concepts. *Journal of Machine Learning Research*, 4(Jun):157–176, 2003.

Nader H Bshouty and Vitaly Feldman. On using extended statistical queries to avoid membership queries. *Journal of Machine Learning Research*, 2(Feb):359–395, 2002.

Nader H Bshouty, Elchanan Mossel, Ryan O'Donnell, and Rocco A Servedio. Learning dnf from random walks. *Journal of Computer and System Sciences*, 71(3):250–265, 2005.

Sitan Chen and Ankur Moitra. Beyond the low-degree algorithm: Mixtures of subcubes and their applications. *arXiv preprint arXiv:1803.06521*, 2018.

Amit Daniely. Sgd learns the conjugate kernel class of the network. In *Advances in Neural Information Processing Systems*, pages 2422–2430, 2017.

Simon S Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. *arXiv preprint arXiv:1810.02054*, 2018.

Andrzej Ehrenfeucht and David Haussler. Learning decision trees from random examples. *Information and Computation*, 82(3):231–246, 1989.

Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. New results for learning noisy parities and halfspaces. In *2006 47th Annual IEEE Symposium on Foundations of Computer Science (FOCS'06)*, pages 563–574. IEEE, 2006.

Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM Journal on Computing*, 39(2):606–645, 2009.

Amos Fiat and Dmitry Pechyony. Decision trees: More theoretical justification for practical algorithms. In *International Conference on Algorithmic Learning Theory*, pages 156–170. Springer, 2004.

Elad Hazan, Adam Klivans, and Yang Yuan. Hyperparameter optimization: A spectral approach. *arXiv preprint arXiv:1706.00764*, 2017.

Jeffrey C Jackson and Rocco A Servedio. Learning random log-depth decision trees under the uniform distribution. In *Learning Theory and Kernel Machines*, pages 610–624. Springer, 2003.

Adam Tauman Kalai and Shang-Hua Teng. Decision trees are pac-learnable from most product distributions: a smoothed analysis. *arXiv preprint arXiv:0812.0933*, 2008.

Michael Kearns. Boosting theory towards practice: Recent developments in decision tree induction and the weak learning framework. In *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 1337–1339, 1996.

Michael Kearns and Yishay Mansour. On the boosting ability of top–down decision tree learning algorithms. *Journal of Computer and System Sciences*, 58(1):109–128, 1999.

Eyal Kushilevitz and Yishay Mansour. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

Jaehoon Lee, Lechao Xiao, Samuel S Schoenholz, Yasaman Bahri, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *arXiv preprint arXiv:1902.06720*, 2019.

Chao Ma, Lei Wu, et al. A comparative analysis of the optimization and generalization property of two-layer neural network and random feature models under gradient descent dynamics. *arXiv preprint arXiv:1904.04326*, 2019.

Ryan O'Donnell and Rocco A Servedio. Learning monotone decision trees in polynomial time. *SIAM Journal on Computing*, 37(3):827–844, 2007.

Samet Oymak and Mahdi Soltanolkotabi. Overparameterized nonlinear learning: Gradient descent takes the shortest path? *arXiv preprint arXiv:1812.10004*, 2018.

Samet Oymak and Mahdi Soltanolkotabi. Towards moderate overparameterization: global convergence guarantees for training shallow neural networks. *arXiv:1902.04674 [cs, math, stat]*, February 2019. URL http://arxiv.org/abs/1902.04674. arXiv: 1902.04674.

J. Ross Quinlan. Induction of decision trees. *Machine learning*, 1(1):81–106, 1986.

Ronald L Rivest. Learning decision lists. *Machine learning*, 2(3):229–246, 1987.

Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

Shai Shalev-Shwartz, Ohad Shamir, and Shaked Shammah. Failures of gradient-based deep learning. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3067–3075. JMLR. org, 2017.

Daniel A Spielman and Shang-Hua Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM (JACM)*, 51(3):385–463, 2004.

Bo Xie, Yingyu Liang, and Le Song. Diverse neural network learns true target functions. *arXiv preprint arXiv:1611.03131*, 2016.