Electronic Colloquium on Computational Complexity, Report No. 137 (2019)

Decision list compression by mild random restrictions

Shachar Lovett*
Computer Science Department
University of California, San Diego
shachar.lovett@gmail.com

Kewen Wu
School of EECS
Peking University, Beijing
shlw_kevin@pku.edu.cn

Jiapeng Zhang†
School of Engineering and Applied Science
Harvard University
jpeng.zhang@gmail.com

September 24, 2019

Abstract

A decision list is an ordered list of rules. Each rule is specified by a term, which is a conjunction of literals, and a value. Given an input, the output of a decision list is the value corresponding to the first rule whole term is satisfied by the input. Decision lists generalize both CNFs and DNFs, and have been studied both in complexity theory and in learning theory.

The size of a decision list is the number of rules, and its width is the maximal number of variables in a term. We prove that decision lists of small width can always be approximated by decision lists of small size, where we obtain sharp bounds (up to constants). This in particular resolves a conjecture of Gopalan, Meka and Reingold (Computational Complexity, 2013) on DNF sparsification.

An ingredient in our proof is a new random restriction lemma, which allows to analyze how DNFs (and more generally, decision lists) simplify if a small fraction of the variables are fixed. This is in contrast to the more commonly used switching lemma, which requires most of the variables to be fixed.

1 Introduction

Decision lists are a model to represent boolean functions, first introduced by Rivest [23]. A decision list is given by a list of rules $(C_1, v_1), \ldots, (C_m, v_m)$. A rule is composed of a condition, given by a term C_i, which is a conjunction of literals (variables or their negations); and an output value v_i in some set V. A decision list computes a function $f : \{0,1\}^n \to V$ as follows:

If $C_1(x) =$ `True` **then** output v_1,
else if $C_2(x) =$ `True` **then** output v_2,
…,
else if $C_m(x) =$ `True` **then** output v_m.

*Research supported by NSF award 1614023.
†Research supported by NSF award 1614023.

ISSN 1433-8092

The last rule is the *default value*, where we assume that $C_m \equiv \texttt{True}$.

Decision lists generalize both CNFs and DNFs. For example, a DNF is a decision list with $v_1 = \cdots = v_{m-1} = 1$ and $v_m = 0$, and a CNF is a decision list with $v_1 = \cdots = v_{m-1} = 0$ and $v_m = 1$. It can be shown that decision lists are a strict generalization of both DNFs and CNFs [16, 23]. Following Rivest's original work, decision lists have been studied both in complexity theory [2, 5, 6, 8, 11, 17, 25] as well as learning theory [3, 7, 12, 15, 19, 26, 27].

Complexity measures of decision lists. There are two natural complexity measures of decision lists: *size* and *width*. Let $L = ((C_i, v_i))_{i \in [m]}$ be a decision list. Its *size* is the number of rules in it (namely m), and its *width* is the maximal number of variables in a term C_i.

Decision list approximation. A decision list L ε-approximates another decision list L' if the two agree on a $(1-\varepsilon)$ fraction of the inputs. It is straightforward to see that small-size decision lists can be approximated by small-width decision lists, by removing rules of large width. Concretely, a decision list of size m can be ε-approximated by a decision list of width $w = \log(m/\varepsilon)$, simply by removing all rules with terms of width more than w. The reverse direction is the main focus on this work. We prove the following result, which provides sharp bounds on approximating small-width decision lists by small-size decision lists.

Theorem 1.1 (Main result)**.** *Let $w \geq 1, \varepsilon > 0$. Any width-$w$ decision list L can be ε-approximated by a decision list L' of width w and size $s = \left(2 + \frac{1}{w} \log \frac{1}{\varepsilon}\right)^{O(w)}$. Moreover, L' is sub-decision list of L, obtained by keeping s rules in L and removing the rest. The bound on s is optimal, up to the unspecified constant in the $O(w)$ term.*

The proof of Theorem 1.1 appears in Section 2. We note that the size bound can be simplified, depending on whether the required error ε is below or above 2^{-w}:

$$\left(2 + \frac{1}{w} \log \frac{1}{\varepsilon}\right)^{O(w)} = \begin{cases} 2^{O(w)} & \varepsilon \geq 2^{-w} \\ \left(\frac{2}{w} \log \frac{1}{\varepsilon}\right)^{O(w)} & \varepsilon \leq 2^{-w}. \end{cases}$$

In both cases, the bound we obtain is sharp, up to the unspecified constant in the $O(w)$ term. We give examples demonstrating this in Section 3.

1.1 Random restrictions

Random restrictions are an essential ingredient of the proof of Theorem 1.1. Håstad's switching lemma [4, 13, 21] is based on the fact that small-width DNFs simplify under random restrictions. More concretely, a random restriction that fixes a $1 - O(1/w)$ fraction of the inputs, simplifies a width-w DNF to a small-depth decision tree. In this work, we study random restrictions where a small constant fraction of the variables is fixed.

A good example to keep in mind is the TRIBES function: a read-once DNF with 2^w terms of width w on disjoint variables. The TRIBES function does not simplify significantly under a random restriction, unless one really fixes a $1 - O(1/w)$ fraction of the inputs. For example, if we randomly fix 50% of the inputs, say, then the TRIBES function simplifies to what is essentially to a smaller TRIBES function (more formally, it simplifies with high probability to a read-once DNF of width $\Omega(w)$). However, we show that this is in essence the worst possible example.

The following lemma is a special case of Lemma 2.12 applied to DNFs (the full lemma deals with decision lists). Given a DNF $f : \{0,1\}^n \to \{0,1\}$, let $\rho \in \{0,1,*\}^n$ be a restriction, and let $f \restriction_\rho$ be the restricted DNF. Clearly, some terms in f might become redundant in $f \restriction_\rho$. For example, they could be false, or they could be implied by other terms. A term that is not redundant is called *useful*. We show that after fixing even a small fraction of the variables (say, 1%), a width-w DNF simplifies to have at most $2^{O(w)}$ useful terms, and hence can not be "too complicated".

Lemma 1.2 (DNFs simplify after mild random restrictions)**.** *Let f be a width-w DNF, and let $f \restriction_\rho$ be a restriction of f obtained by restricting each variable with probability α. Then the expected number of useful terms in $f \restriction_\rho$ is at most $(4/\alpha)^w$.*

1.2 Applications

We discuss some applications of Theorem 1.1 below.

1.2.1 DNF sparsification

This decision list compression problem is a natural generalization of the *DNF sparsification* problem, introduced by Gopalan, Meka and Reingold [10] as a means to obtain pseudorandom generator fooling small-width DNFs. Their main structural result can be summarized as follows.

Theorem 1.3 ([10])**.** *Any width-w DNF can be ε-approximated by a DNF of width w and size $(w \log(1/\varepsilon))^{O(w)}$.*

They conjectured that a better bound is possible.

Conjecture 1.4 ([10])**.** *Any width-w DNF can be ε-approximated by a DNF of width w and size $s(w,\varepsilon)$, where:*

- Weak version: *$s(w,\varepsilon) = c(\varepsilon)^w$ for some function c.*
- Strong version: *$s(w,\varepsilon) = (\log(1/\varepsilon))^{O(w)}$.*

The weak version was resolved by Lovett and Zhang [18], where they showed that $c(\varepsilon) = (1/\varepsilon)^{O(1)}$ suffices. Our main result, Theorem 1.1, verifies the strong version of their conjecture (and in fact, proves a sharper bound than the one conjectured).

Corollary 1.5 (This work)**.** *Any width-w DNF can be ε-approximated by a DNF of width w and size $\left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}$.*

Corollary 1.5 is tight, up to the unspecified constant in the $O(w)$ term. We give examples demonstrating this for the more general model of decision lists in Section 3. If we restrict our attention to DNFs, then the following explicit functions demonstrate this as well:

- For $2^{-2w} \le \varepsilon \le 1/3$, consider approximating the Majority function on $2w$ variables with error $1/3$. This requires a width-w DNF of size $2^{\Omega(w)}$ (this is attributed to Rocco Servedio in [10]). Note that $2^{\Theta(w)} = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{\Omega(w)}$ in this regime.
- For $\varepsilon \le 2^{-2w}$, consider exactly computing the Threshold-w function on $\log(1/\varepsilon)$ variables, which amounts to approximation with any error $< \varepsilon$. This requires a width-w DNF of size $\binom{\log(1/\varepsilon)}{w} = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{\Omega(w)}$.

1.2.2 Junta theorem

A k-junta is a function depending on at most k variables. Friedgut's junta theorem [9] shows that boolean functions of small influence can be approximated by juntas. For the relevant definitions see for example [20].

Theorem 1.6 (Friedgut's junta theorem [9])**.** *Let $f : \{0,1\}^n \to \{0,1\}$ be a boolean function with total influence I. Then for any $\varepsilon > 0$, f can be ε-approximated by a k-junta for $k = 2^{O(I/\varepsilon)}$.*

It is well known that width-w DNFs have total influence $I = O(w)$, which implies by Theorem 1.6 that width-w DNFs can be ε-approximated by $2^{O(w/\varepsilon)}$-juntas. As a corollary of Theorem 1.1, we improve the bound, and generalize it to decision lists.

Corollary 1.7 (This work)**.** *Any width-w decision list can be ε-approximated by a k-junta for $k = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}$.*

This improves previous bounds, even when restricted to DNFs or CNFs. By combining the results in [10, 18] one gets the bound $k = \min\{w\log(1/\varepsilon), 1/\varepsilon\}^{O(w)}$ for width-w DNFs or CNFs. It can be verified that our new result is indeed better; for example for $\varepsilon = w^{-w}$ we obtain $(\log w)^{O(w)}$ instead of $w^{O(w)}$. It is also worthwhile noting that the result of [18], which obtained the bound $(1/\varepsilon)^{O(w)}$, can be extended to decision lists with minimal changes.

1.2.3 Learning small-width DNF

A class of boolean functions is said to be (ε, δ)-PAC learnable using q queries if there exists a learning algorithm that, given query access to an unknown function in the class, returns with probability $(1 - \delta)$ a function which ε-approximates the unknown function, while making at most q queries. In our context we consider membership queries, where the learning algorithm can query the value of the unknown function on any chosen input.

A celebrated result of Jacskson [14] shows that polynomial-size DNFs can be PAC learned under the uniform distribution using membership queries.

Theorem 1.8 (Jackson's harmonic sieve [14])**.** *The class of n-variate DNFs of size s is (ε, δ)-PAC learnable under the uniform distribution with $q = \mathrm{poly}(s, n, 1/\varepsilon, \log(1/\delta))$ membership queries.*

Using Theorem 1.1, we can extend Jackson's result to small-width DNFs.

Corollary 1.9 (This work)**.** *The class of n-variate DNFs of width w is (ε, δ)-PAC learnable under the uniform distribution with $q = \mathrm{poly}(s, n, 1/\varepsilon, \log(1/\delta))$ membership queries, where $s = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}$.*

Proof Sketch. Jackson's algorithm combines a weak learner based on Fourier analysis and a boosting algorithm that converts this weak learner to a strong learner. Let $f(x)$ be the target DNF that we are trying to learn. The weak learner solves the following problem: given a distribution D on $\{0,1\}^n$, output a set S such that the parity $\chi_S(x) = \bigoplus_{i \in S} x_i$ is correlated with f under the distribution D. Initially D is the uniform distribution, but the boosting algorithm keeps adapting D to focus on inputs where it made many mistakes.

In Jackson's algorithm, the existence of such S is shown by observing that for a size-s DNF, at least one of the terms must be $1/s$ correlated to the function; and each term's contribution can be

attributed to the parities supported on it. For width w terms, this leads to at most a 2^{-w} decrease in the correlation.

Assume now that $f(x)$ is a width-w DNF with too many terms, so we cannot apply the previous argument directly. Apply Theorem 1.1 with error γ (to be determined soon), to obtain an approximate width-w DNF $g(x)$ which γ-approximate $f(x)$, where g has at most $s = \left(2 + \frac{1}{w}\log\frac{1}{\gamma}\right)^{O(w)}$ terms. Crucially, we obtain $g(x)$ by removing some of the terms in $f(x)$, and hence $g(x) \leq f(x)$ for all inputs x. In particular, $\Pr_{x\sim D}[f(x) = 1] \geq \Pr_{x\sim D}[g(x) = 1]$.

Assume that we know that the distribution D is not too far from uniform. Concretely, that $D(x) \leq K2^{-n}$ for some parameter K. This implies that

$$\Pr_{x\sim D}[f(x) = 1] \leq \Pr_{x\sim D}[g(x) = 1] + \gamma K.$$

We will choose $\gamma = 1/12K$. We may assume that $\Pr_{x\sim D}[f(x) = 1] \in [1/3, 2/3]$, otherwise the constant 1 function correlates with f under D. Thus $\Pr_{x\sim D}[g(x) = 1] \in [1/4, 3/4]$. This implies, by the same argument as in the original paper of Jackson, there there is a term C of g which is $\Omega(1/s)$-correlated with g. One can verify that as $g(x) \leq f(x)$ then C is also $\Omega(1/s)$-correlated with f.

Finally, we need to bound K. It is known that boosting algorithms can be restricted to have $K = \varepsilon^{-O(1)}$, which completes the proof. □

1.3 Proof overview

We give a high-level overview of the proof of Theorem 1.1. Let $L = ((C_i, v_i))$ be a decision list of width w and size m.

General Framework. Given a subset $J \subset [m]$, we denote by $L|_J$ the decision list restricted to the rules in J, where we delete the rest. Our goal is to find a small subset $J \subset [m]$ such that $L|_J$ approximates L. We say that a rule (C_i, v_i) of L is *hit* by an input x if $C_i(x) = 1$ and $C_j(x) = 0$ for $j < i$; in this case, $L(x) = v_i$. The main intuition underlying our approach is:

If a rule is rarely hit by random inputs, then we can safely remove it.

Armed with this intuition, our approach is to choose J to be the set of rules with the highest probability of being hit. We show that in order to get an ε-approximation, it suffices to keep the top $\left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}$ rules.

Our general approach follows that of Lovett and Zhang [18]. They combined two central results in the analysis of boolean functions: *random restrictions* and *noise stability*. The main innovation in the current work is that we apply random restrictions that fix only a small fraction of the inputs; this is in contrast to the common use of random restrictions, such as in the proof of Håstad's switching lemma [13], where most variables are fixed. The ability to handle random restrictions which fix only a small fraction is what allows us to obtain improved bounds.

Mild random restrictions. An index $i \in [m]$ is said to be *useful* if there exists an assignment x such that the evaluation of $L(x)$ hits the i-th rule (and hence outputs v_i). We denote the number of useful indices in L by $\#\text{useful}\,(L)$. This notion is natural, as we can always discard rules if no assignment hits it. The main point is that restrictions can render some rules in a decision list

non-useful. Let ρ be a random restriction that keeps each variable alive with probability α. We show that on average, the restricted decision list $L \restriction_\rho$ has a small number of useful indices:

$$\mathop{\mathbb{E}}_{\rho}\left[\#\text{useful}\left(L \restriction_\rho\right)\right] \leq \left(\frac{4}{1-\alpha}\right)^w.$$

The proof is based on an encoding argument. Let ρ be a restriction for which $L \restriction_\rho$ has T useful indices. Let $t \in [T]$ be uniformly chosen. We construct a new restriction ρ' by further restricting the variables in the t-th useful rule so that it is satisfied. Then from ρ' and some small additional information a, we can recover both ρ and t. This shows that if T is too large, it can only happen with low probability, as the entropy of (ρ', a) is much lower than that of (ρ, t).

Noise Stability. Since there is no guarantee about the value on each rule of the decision list, it is convenient to consider the following index function. Let $L = ((C_i, v_i))_{i\in[m]}$ be a decision list on n variables. The index function of L outputs for an input x the index i of the first term in L satisfied by x. Equivalently, $\mathrm{Ind}L$ is given by the decision list $\mathrm{Ind}L = ((C_i, i))_{i\in[m]}$.

We make two important definitions. What we *want* to analyze are the quantities

$$p_L(i) := \Pr\left[\mathrm{Ind}L(x) = i\right].$$

In particular, we want to show that there is a small set of indices J such that $\sum_{i\in J} p_L(i) \geq 1-\varepsilon$. What we *can* analyze using random restrictions are the quantities

$$q_L(\alpha, i) = \Pr_{\rho}\left[\text{index } i \text{ is useful in } L \restriction_\rho\right],$$

since it holds that

$$\sum_i q_L(\alpha, i) = \mathop{\mathbb{E}}_{\rho}\left[\#\text{useful}\left(L \restriction_\rho\right)\right] \leq \left(\frac{4}{1-\alpha}\right)^w.$$

We use noise stability to bridge between the two.

Let $\beta = 1-\alpha$. For any $x \in \{0,1\}^n$, the noise distribution $y \sim \mathcal{N}_\beta(x)$ is sampled by taking $\Pr[y_i = x_i] = \frac{1+\beta}{2}$ independently for $i \in [n]$. Consider sampling $x \in \{0,1\}^n$ uniformly and $y \sim \mathcal{N}_\beta(x)$. We can equivalently sample the pair (x, y) by first sampling a common restriction ρ, where each variables stays alive with probability α, and then sample its completion for x and y independently. Let

$$\mathrm{Stab}_L(\beta, i) := \Pr_{x,y}\left[\mathrm{Ind}L(x) = \mathrm{Ind}L(y) = i\right].$$

We show that $p_L(i)$ and $q_L(\alpha, i)$ are both polynomially related, by relating them to $\mathrm{Stab}_L(\beta, i)$:

$$\frac{p_L(i)^2}{q_L(1-\beta, i)} \leq \mathrm{Stab}_L(\beta, i) \leq p_L(i)^{\frac{2}{1+\beta}}.$$

The upper bound is proven by hypercontrativity, and the lower bound by a somewhat delicate Cauchy-Schwarz inequality. This allows us to obtain that

$$p_L(i) \leq q_L(1-\beta, i)^{\frac{1+\beta}{2\beta}}.$$

Finally, we put everything together by optimizing the value of β.

Related works. We already discussed the works of Gopalan, Meka and Reingold [10] and Lovett and Zhang [18] which gave weaker bounds for DNF sparsification, compared to Theorem 1.1.

There has been previous works studying how small-width DNFs simplify under mild random restrictions, that fix a small fraction of the variables (say, 1%). Segerlind, Buss and Impagliazzo [24], improved by Razborov [22], show that width-w DNFs simplify to a decision tree of depth $2^{O(w)}$. Compared to Theorem 1.1, we obtain bounds on size (namely, number of useful terms), which are better than bounds on depth. However, we only bound the first moment (that is, expected number of useful terms), while [22] bounds higher moments as well. So to some extent, the results are incomparable. We believe that with some further work, one can improve our techniques to obtain bounds on higher moments as well (this was unnecessary for the current work). Finally, It is also worthwhile to mention the work by the authors and Alweiss [1] , where mild random restrictions (of a somewhat different flavor) were used to obtain improved bounds for the sunflower lemma in combinatorics.

Paper Organization. In Section 2, we prove the upper bound of decision list compression. In Section 3, we give the lower bounds to show the tightness of our result.

Acknowledgements. We thank Ben Rossman for invaluable discussions.

2 Upper bounds

We start by make some definitions formal. We denote $[n] = \{1, 2, \ldots, n\}$, variables are $x_1, \ldots, x_n$, and literals are $x_1, \neg x_1, \ldots, x_n, \neg x_n$. A *term* is a conjunction of literals.

Definition 2.1 (Decision list)**.** *A width-w size-m decision list is a list $L = ((C_i, v_i))_{i\in[m]}$ of rules. A rule is a pair (C_i, v_i), where C_i is a term containing at most w literals, each v_i is a value in some finite set V. We assume $C_m \equiv 1$, and (C_m, v_m) is the final default rule.*

For any $J \subseteq [m]$ with $m \in J$, we denote by $L|_J = ((C_j, v_j))_{j\in J}$ the restriction of L to the rules in J, where elements of J are taken in ascending order.

The evaluation of L given assignment x is to find the first index i such that $C_i(x) = 1$ and then output $L(x) = v_i$. We make additional remarks for decision list to avoid potential pitfalls.

- If $m \notin J$, we will consider $L|_J$ invalid, as it does not have a default rule at the end.
- No variable appears in any single term more than once, which rules out $x_1 \wedge x_1$ and $x_1 \wedge \neg x_1$.

Our goal in this section is to prove the following theorem, which is the upper bound part in Theorem 1.1.

Theorem 2.2. *Let $L = ((C_i, v_i))_{i\in[m]}$ be a width-w decision list. Then for every $\varepsilon > 0$, there exists $J \subseteq [m], m \in J$ of size $|J| = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}$ such that $\Pr[L(x) \neq L|_J(x)] \leq \varepsilon$.*

2.1 Useful indices

Since there is no guarantee about the value on each rule of the decision list, it is convenient to consider the index function. Let $L = ((C_i, v_i))_{i\in[m]}$ be a decision list on n variables. The index function of L is a function $\mathrm{Ind}L : \{0,1\}^n \to [m]$, given by

$$\mathrm{Ind}L(x) = \min\{i \in [m] \mid C_i(x) = 1\}.$$

Equivalently, $\mathrm{Ind}L$ is given by the decision list $\mathrm{Ind}L = ((C_i, i))_{i\in[m]}$. Using the index function, it suffices to discard some rules of L and show it still approximates the index function.

Claim 2.3. *Let $L = ((C_i, v_i))_{i\in[m]}$ be a decision list. Then for any $J \subseteq [m], m \in J$, we have*

$$\Pr[L(x) \neq L|_J(x)] \leq \Pr[\mathit{Ind}L(x) \notin J].$$

Proof. This follows as if $\mathrm{Ind}L(x) = j \in J$, then $L(x) = L|_J(x) = v_j$. □

Obviously, if a rule of a decision list is covered by some previous rules, then we can safely remove it. For example, in $(x_1, 1), (x_1 \wedge x_2, 2)$ the second rule is useless. To make this more formal, we introduce the following notion of a *useful index*.

Definition 2.4 (Useful index)**.** *Given size-m decision list L, an index $i \in [m]$ is said to be* useful *if there exists an assignment x such that $\mathit{Ind}L(x) = i$. We denote by $\#\mathit{useful}(L)$ the number of useful indices in L.*

Example 2.5. *Assume $L = ((x_1, a), (x_1 \wedge \neg x_2, b), (1, c), (x_1, d), (1, e))$. Then indices $1, 3$ are useful, but indices $2, 4, 5$ are not. So $\#\mathit{useful}(L) = 2$.*

The main intuition underlying our approach is that rules that are hardly hit by random inputs can be removed. Motivated by this, we define

$$p_L(i) := \Pr[\mathrm{Ind}L(x) = i].$$

Claim 2.6. *For any size-m decision list L, we have $\sum_{i=1}^{m} p_L(i) = 1$.*

Proof. This follows as the events $[\mathrm{Ind}L(x) = i]$ are a partition of the probability space. □

The following is our main technical lemma.

Lemma 2.7. *Let $L = ((C_i, v_i))_{i\in[m]}$ be a width-w decision list. Sort $[m] = \{j_1, \ldots, j_m\}$ such that $p_L(j_1) \geq p_L(j_2) \geq \cdots \geq p_L(j_m)$. For any $\varepsilon > 0$, let*

$$t = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}.$$

Then for $J = \{j_1, \ldots, j_t, m\}$ it holds that $\Pr[\mathit{Ind}L(x) \notin J] \leq \varepsilon$.

The proof of Theorem 2.2 follows immediately, by combining Lemma 2.7 and Claim 2.3.

2.2 Random restrictions and encoding

A *restriction* on n variables is $\rho \in \{0,1,*\}^n$. An (n,k)-random restriction is the uniform distribution over restrictions $\rho \in \{0,1,*\}^n$ with exactly k stars, which we denote by $\mathscr{R}(n,k)$. An (n,α)-random restriction assigns independently each bit of the restriction ρ as $0,1,*$ with probability $\frac{1-\alpha}{2}, \frac{1-\alpha}{2}, \alpha$ respectively, which we denote by $\mathscr{U}(n,\alpha)$. Given a decision list $L : \{0,1\}^n \to V$, its restriction under ρ is $L \restriction_\rho : \{0,1\}^{\rho^{-1}(*)} \to V$.

Definition 2.8 (Useful probability)**.** *Given size-m decision list L and $\alpha \in (0,1)$, the useful probability of an index $i \in [m]$ is*

$$q_L(\alpha, i) := \Pr_{\rho \sim \mathscr{U}(n,\alpha)} [\textit{index } i \textit{ is useful in } L \restriction_\rho].$$

Note that we assume L initially does not contain useless rules, so for any α and i, we always have $q_L(\alpha, i) > 0$. We also have the following simple fact regarding useful probability.

Claim 2.9. *For any size-m decision list L, we have $\sum_{i=1}^m q_L(\alpha, i) = \mathbb{E}_{\rho \sim \mathscr{U}(n,\alpha)} [\#\textit{useful}(L \restriction_\rho)]$.*

Proof. Let $1_{\rho,i}$ be the indicator of index i being useful in $L \restriction_\rho$. Then

$$\mathop{\mathbb{E}}_{\rho \sim \mathscr{U}(n,\alpha)} [\#\text{useful}\,(L \restriction_\rho)] = \mathop{\mathbb{E}}_{\rho} \left[\sum_{i=1}^m 1_{\rho,i} \right] = \sum_{i=1}^m \mathop{\mathbb{E}}_{\rho} [1_{\rho,i}] = \sum_{i=1}^m q_L(\alpha, i).$$

□

Now we present an encoding/decoding scheme for random restriction and analyze the expectation in Claim 2.9 explicitly. Let $\alpha \in (0,1)$ be such that αn is an integer. Define:

$$\mathcal{U} := \left\{ (\rho, s) \middle| \rho \in \mathscr{R}(n, \alpha n), s \in \{1, \ldots, \#\text{useful}\,(L \restriction_\rho)\} \right\}$$
$$\mathcal{V} := \left\{ (\rho', a) \middle| \rho' \in \bigcup_{k=0}^{w} \mathscr{R}(n, \alpha n - k), a \in \{\textsc{Old}, \textsc{New}\}^w \right\}.$$

We define two deterministic algorithms $\mathrm{Enc} : \mathcal{U} \to \mathcal{V}$ and $\mathrm{Dec} : \mathrm{Enc}(\mathcal{U}) \subseteq \mathcal{V} \to \mathcal{U}$ such that $\mathrm{Dec}(\mathrm{Enc}(\rho, s)) = (\rho, s)$ holds for any $(\rho, s) \in \mathcal{U}$.

Algorithm 1: Encoding algorithm $\mathrm{Enc}(\rho, s)$

Input: restriction and index $(\rho, s) \in \mathcal{U}$
Output: restriction and string $(\rho', a) \in \mathcal{V}$

1 $I \leftarrow \{i \mid i \text{ is a useful index in } L \restriction_\rho\}$
2 $j \leftarrow$ the s-th element in I
3 $\rho' \leftarrow \rho, a \leftarrow \varnothing$
/* Assume $C_j = \bigwedge_{k=1}^{c} y_{j_k}, y_{j_k} \in \{x_{j_k}, \neg x_{j_k}\}, c \leq w$ */
4 **for** $k = 1$ **to** c **do**
5 **if** $\rho(x_{j_k}) \in \{0, 1\}$ **then**
6 Append a with OLD /* x_{j_k} is already set by ρ */
7 **else**
8 Append a with NEW /* x_{j_k} is newly set to satisfy this term */
9 **if** $y_{j_k} = x_{j_k}$ **then** Update $\rho'(x_{j_k}) \leftarrow 1$ **else** Update $\rho'(x_{j_k}) \leftarrow 0$
10 **end**
11 Append a arbitrarily to length w
12 **end**
13 **return** (ρ', a)

Algorithm 2: Decoding algorithm $\mathrm{Dec}(\rho', a)$

Input: restriction and string $(\rho', a) \in \mathrm{Enc}(\mathcal{U}) \subseteq \mathcal{V}$
Output: restriction and index $(\rho, s) \in \mathcal{U}$

1 $j \leftarrow$ index of the first satisfied term in $L \restriction_{\rho'}$
2 $\rho \leftarrow \rho'$
/* Assume $C_j = \bigwedge_{k=1}^{c} y_{j_k}, y_{j_k} \in \{x_{j_k}, \neg x_{j_k}\}, c \leq w$ */
3 **for** $k = 1$ **to** c **do**
4 **if** $a_k =$ NEW **then** /* x_{j_k} was not set by ρ */
5 Update $\rho(x_{j_k}) \leftarrow *$
6 **end**
7 **end**
8 $I \leftarrow \{i \mid i \text{ is a useful index in } L \restriction_\rho\}$
9 $s \leftarrow$ rank of j in I
10 **return** (ρ, s)

The following claim proves the correctness of the encoding and decoding algorithms.

Claim 2.10. *$\mathrm{Dec}(\mathrm{Enc}(\rho, s)) = (\rho, s)$ holds for any $(\rho, s) \in \mathcal{U}$.*

Proof. Sort literals in each term of $L = ((C_i, v_i))_{i \in [m]}$ arbitrarily. To justify the correctness, let $(\rho', a) = \mathrm{Enc}(\rho, s)$, then we need to ensure:

- $\mathrm{Dec}(\rho', a)$ obtains the same j in line 1 as $\mathrm{Enc}(\rho, s)$ does in line 2:

  During $\mathrm{Enc}(\rho, s)$, index j is useful in $L \restriction_\rho$, thus setting unfixed variables to satisfy C_j will not make any term C_i for $i < j$ satisfied. Hence the first satisfied term in $L \restriction_{\rho'}$ is C_j.

- $\mathrm{Dec}(\rho', a)$ in line 8 obtains the correct ρ:

Since each term is sorted in advance, and a encodes which variable in C_j is set by $\mathrm{Enc}(\rho, s)$ rather than ρ, the loop in $\mathrm{Dec}(\rho', a)$ will set these variables back to $*$ and recover ρ.

□

Corollary 2.11. $|\mathcal{U}| \leq |\mathcal{V}|$.

Proof. Enc is an injection from $\mathcal{U}$ to $\mathrm{Enc}(\mathcal{U}) \subset \mathcal{V}$. □

Lemma 2.12. *Let L be a width-w decision on n variables and let $\alpha \in (0,1)$. Then*

$$\mathop{\mathbb{E}}_{\rho\sim\mathscr{U}(n,\alpha)}\left[\#\mathit{useful}\left(L\restriction_\rho\right)\right] \leq \left(\frac{4}{1-\alpha}\right)^w.$$

Proof. We first prove the bound for $\rho \sim \mathscr{R}(n, \alpha n)$ and then increase the number of variables to infinity, by adding dummy variables. This proves the desired bound as for $n' \to \infty$, the restriction of $\mathscr{R}(n', \alpha n')$ to the first n variables converges to $\mathscr{U}(n, \alpha)$. We have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\rho\sim\mathscr{R}(n,\alpha n)}\left[\#\mathrm{useful}\left(L\restriction_\rho\right)\right] = \frac{1}{|\mathscr{R}(n,\alpha n)|}\sum_{\rho\in\mathscr{R}(n,\alpha n)}\#\mathrm{useful}\left(L\restriction_\rho\right)\\
=&\frac{|\mathcal{U}|}{|\mathscr{R}(n,\alpha n)|} \leq \frac{|\mathcal{V}|}{|\mathscr{R}(n,\alpha n)|} \leq \frac{\left(\sum_{k=0}^{w}\binom{n}{\alpha n-k}2^{(1-\alpha)n+k}\right)\times 2^w}{\binom{n}{\alpha n}2^{(1-\alpha)n}}\\
\leq&\frac{\left(\sum_{k=0}^{w}\binom{n}{\alpha n-k}\right)\times 4^w}{\binom{n}{\alpha n}} \leq \frac{\binom{n+w}{\alpha n}\times 4^w}{\binom{n}{\alpha n}} \leq \left(\frac{4}{1-\alpha}\right)^w.
\end{aligned}$$

□

2.3 Noise stability

We use noise stability as a bridge to connect $p_L(i)$ and $q_L(\alpha, i)$.

Definition 2.13 (Noisy distribution)**.** *Given $x \in \{0,1\}^n$ and a noise parameter $\beta \in (0,1)$, we denote by $\mathcal{N}_\beta(x)$ the distribution over $y \in \{0,1\}^n$, where $\Pr[y_i = x_i] = \frac{1+\beta}{2}, \Pr[y_i \neq x_i] = \frac{1-\beta}{2}$ independently for all $i \in [n]$.*

Definition 2.14 (Stability)**.** *Let $g : \{0,1\}^n \to \{0,1\}$ be a boolean function. The β-stability of g is*

$$\mathit{Stab}_\beta(g) = \Pr_{x\in\{0,1\}^n, y\sim\mathcal{N}_\beta(x)}\left[g(x) = g(y) = 1\right].$$

The hypercontractive inequality (see for example [20], page 259) allows us to bound the stability of a boolean function by its acceptance rate.

Fact 2.15. *Let $g : \{0,1\}^n \to \{0,1\}$ and $\beta \in (0,1)$. Then $\mathit{Stab}_\beta(g) \leq \left(\Pr[g(x) = 1]\right)^{\frac{2}{1+\beta}}$.*

Next, we define index stability and relate it to useful probability and hit probability.

Definition 2.16 (Index stability)**.** *Given size-m decision list L on n variables, the β-stability of index $i \in [m]$ is*

$$\mathit{Stab}_L(\beta, i) := \Pr_{x\in\{0,1\}^n, y\sim\mathcal{N}_\beta(x)}\left[\mathit{IndL}(x) = \mathit{IndL}(y) = i\right].$$

Lemma 2.17 (Bridging lemma)**.** *Let L be a size-m width-w decision list on n variables. Then for any index $i \in [m]$ and $\beta \in (0,1)$, we have*

$$\frac{p_L(i)^2}{q_L(1-\beta,i)} \leq \mathit{Stab}_L(\beta,i) \leq p_L(i)^{\frac{2}{1+\beta}}.$$

Proof. We first prove the upper bound. Let $g : \{0,1\}^n \to \{0,1\}$ be an indicator boolean function for $\mathrm{Ind}L(x) = i$. Then using Fact 2.15, we have

$$\mathrm{Stab}_L(\beta,i) = \mathrm{Stab}_\beta(g) \leq (\Pr[g(x)=1])^{\frac{2}{1+\beta}} = (\Pr[\mathrm{Ind}L(x)=i])^{\frac{2}{1+\beta}} = p_L(i)^{\frac{2}{1+\beta}}.$$

We now turn to prove the lower bound. Let $\alpha = 1-\beta$. Observe that we can sample (x,y) where $x \in \{0,1\}^n, y \sim \mathcal{N}_\beta(x)$ as follows:

- Sample restriction $\rho \sim \mathscr{U}(n,\alpha)$;
- Sample uniform $x' \in \{0,1\}^{\rho^{-1}(*)}$ and complete stars in ρ with it as x;
- Sample uniform $y' \in \{0,1\}^{\rho^{-1}(*)}$ and complete stars in ρ with it as y.

We thus have

$$\mathrm{Stab}_L(\beta,i) = \Pr_{\rho,x',y'}\left[\mathrm{Ind}L\restriction_\rho(x') = \mathrm{Ind}L\restriction_\rho(y') = i\right].$$

We now make a seemingly redundant, but surprisingly useful, conditioning. Let $\mathcal{E}(\rho,i)$ denote the event

$$\mathcal{E}(\rho,i) := [i \text{ is useful in } L\restriction_\rho].$$

Then we can equivalently write

$$\mathrm{Stab}_L(\beta,i) = \Pr_{\rho,x',y'}\left[\mathrm{Ind}L\restriction_\rho(x') = \mathrm{Ind}L\restriction_\rho(y') = i \wedge \mathcal{E}(\rho,i)\right].$$

For any fixed ρ, define

$$r_\rho(i) := \Pr_{x'}\left[\mathrm{Ind}L\restriction_\rho(x') = i\right].$$

Since x', y' are independent for any fixed restriction, we have

$$\begin{aligned}
\mathrm{Stab}_L(\beta,i) =& \Pr_\rho[\mathcal{E}(\rho,i)] \cdot \Pr_{\rho,x',y'}\left[\mathrm{Ind}L\restriction_\rho(x') = \mathrm{Ind}L\restriction_\rho(y') = i \middle| \mathcal{E}(\rho,i)\right] \\
=& q_L(\alpha,i) \cdot \mathop{\mathbb{E}}_\rho\left[r_\rho(i)^2 \middle| \mathcal{E}(\rho,i)\right] \\
\geq& q_L(\alpha,i) \cdot \left(\mathop{\mathbb{E}}_\rho\left[r_\rho(i) \middle| \mathcal{E}(\rho,i)\right]\right)^2 && \text{(Cauchy-Shwarz inequality)} \\
=& \frac{1}{q_L(\alpha,i)}\left(q_L(\alpha,i) \cdot \mathop{\mathbb{E}}_\rho\left[r_\rho(i) \middle| \mathcal{E}(\rho,i)\right]\right)^2 \\
=& \frac{1}{q_L(\alpha,i)}\left(\Pr_{\rho,x'}\left[\mathrm{Ind}L\restriction_\rho(x') = i \wedge \mathcal{E}(\rho,i)\right]\right)^2 \\
=& \frac{1}{q_L(\alpha,i)}\left(\Pr_{\rho,x'}\left[\mathrm{Ind}L\restriction_\rho(x') = i\right]\right)^2 \\
=& \frac{1}{q_L(\alpha,i)}\left(\Pr_x[\mathrm{Ind}L(x) = i]\right)^2 = \frac{p_L(i)^2}{q_L(\alpha,i)}.
\end{aligned}$$

□

Corollary 2.18. *Let L be a size-m width-w decision list. Then for any index $i \in [m]$ and $\beta \in (0,1)$, we have*

$$p_L(i) \le q_L(1-\beta, i)^{\frac{1+\beta}{2\beta}}.$$

2.4 Putting everything together

Now we put everything together and give the proof of Lemma 2.7.

Proof of Lemma 2.7. Recall that we sorted $[m] = \{j_1, \dots, j_m\}$ such that $p_L(j_1) \ge p_L(j_2) \ge \cdots \ge p_L(j_m)$. Let $J = \{j_1, \dots, j_t, m\}$ for t to be optimized later.

Next, let $\beta \in (0,1)$ to be optimized later and set $\alpha = 1-\beta$. Sort $[m] = \{i_1, \dots, i_m\}$ such that $q_L(\alpha, i_1) \ge q_L(\alpha, i_2) \ge \cdots \ge q_L(\alpha, i_m)$. By Claim 2.9 and Lemma 2.12, we have

$$\sum_{k=1}^{m} q_L(\alpha, i_k) = \mathop{\mathbb{E}}_{\rho \sim \mathscr{R}(n,\alpha)} [\#\text{useful}\,(L \restriction \rho)] \le \left(\frac{4}{1-\alpha}\right)^w = \left(\frac{4}{\beta}\right)^w.$$

Note that we have sorted q_L in decreasing order, so

$$q_L(\alpha, i_k) \le \frac{1}{k}\left(\frac{4}{\beta}\right)^w.$$

Observe that $j_1, \dots, j_t$ have the largest hit probability, and apply Corollary 2.18, then

$$\begin{aligned}
\sum_{j \notin J} p_L(j) &\le \sum_{k=t+1}^{m} p_L(j_k) \le \sum_{k=t+1}^{m} p_L(i_k) \le \sum_{k=t+1}^{m} q_L(\alpha, i_k)^{\frac{1+\beta}{2\beta}} \\
&\le \left(\frac{4}{\beta}\right)^{w \times \frac{1+\beta}{2\beta}} \sum_{k \ge t+1} \left(\frac{1}{k}\right)^{\frac{1+\beta}{2\beta}} \\
&\le \left(\frac{4}{\beta}\right)^{w \times \frac{1+\beta}{2\beta}} \times \frac{2\beta}{1-\beta} \times t^{-\frac{1-\beta}{2\beta}}.
\end{aligned}$$

If we restrict $\beta \le 1/2$ and choose

$$t = \left(\frac{1}{\varepsilon}\right)^{\frac{2\beta}{1-\beta}} \left(\frac{4}{\beta}\right)^{w \times \frac{1+\beta}{1-\beta}} \left(\frac{2\beta}{1-\beta}\right)^{\frac{2\beta}{1-\beta}} \le 4\left(\frac{1}{\varepsilon}\right)^{4\beta} \left(\frac{4}{\beta}\right)^{3w},$$

then

$$\Pr\left[\mathrm{Ind}L(x) \notin J\right] = \sum_{j \notin J} p_L(j) \le \varepsilon.$$

Now we divide ε into two cases. Assume $\varepsilon = 2^{-\ell w}$. Then:

- If $\ell \le 2$ we set $\beta = 1/2$ and get $t = 2^{O(w)}$.
- If $\ell \ge 2$ we set $\beta = 1/\ell$ and get $t = \ell^{O(w)}$.

One can verify that in either case we get

$$t = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}.$$

□

3 Lower bounds

In this section, we prove two lower bounds for decision list compression, which show that the bounds in Theorem 1.1 are tight up to constants.

Claim 3.1. *For any w, there is a width-w decision list $L : \{0,1\}^w \to \{0,1\}$ such that*

$$\Pr\left[L(x) \neq L'(x)\right] > 1/3$$

for any width-w decision list L' of size at most $2^w/100w$.

Proof. Since any boolean function on w variables can be expressed as some width-w decision list, there are 2^{2^w} possible L. On the other hand, for any fixed L', it can approximate at most

$$\binom{2^w}{2^w/3} \times 2^{2^w/3} \leq 2^{0.97\times 2^w}$$

different boolean functions within distance $1/3$; and for fixed size m, there are at most $(3^w \times 2)^m$ distinct size-m width-w decision lists. As small-size decision list can be embedded in larger ones, when restricted to size at most $2^w/100w$, it only approximates at most

$$(3^w \times 2)^{\frac{2^w}{100w}} \times 2^{0.97\times 2^w} < 2^{2^w}$$

different boolean functions on w variables. □

Claim 3.2. *For any w and $n > 2w$, there is a width-w decision list $L : \{0,1\}^n \to \{0,1\}$ which is not equivalent to any width-w decision list L' of size smaller than $\binom{n}{w}/n^2$.*

Proof. Let $m = \binom{n}{w}$ and sort all $\binom{n}{w}$ subsets of $[n]$ with size w as $\{S_1, \ldots, S_m\}$ arbitrarily. For any $i \in [m]$, define $C_i = \bigwedge_{j\in S_i} x_j$. For any $v \in \{0,1\}^m$, let $L_v = ((C_1, v_1), \ldots, (C_m, v_m), (1, 0))$ be a size-$(m+1)$ width-w decision list.

As small-size decision list can be embedded in larger ones, assume towards a contradiction that any L_v is equivalent to some size-(m/n^2) width-w decision list L'_v. The underlying mapping $L_v \mapsto L'_v$ is a bijection, since all rules in L_v are useful and, given L'_v, we can recover L_v by enumerating all those assignments. However, the number of possible L'_v is upper bounded by

$$\left(2 \times \sum_{k=0}^{w} 2^k \binom{n}{k}\right)^{\binom{n}{w}/n^2} \leq \binom{n}{w}^{2m/n^2} < 2^m.$$

□

Now the general lower bound follows immediately.

Corollary 3.3. *For any w and $\varepsilon \leq 1/3$, there is a width-w decision list L such that*

$$\Pr\left[L(x) \neq L'(x)\right] > \varepsilon$$

holds for any width-w decision list L' of size at most

$$\left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}.$$

Proof. For $\varepsilon \geq 2^{-2w}$, let L be the decision list in Claim 3.1, then it can not be approximated within $\varepsilon < 1/3$ of size at most

$$\frac{2^w}{100w} = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}.$$

For $\varepsilon < 2^{-2w}$, let L be the decision list in Claim 3.2 with $n = \log(1/\varepsilon)$. Since now $\varepsilon = 2^{-n}$, the desired L' must be equivalent to L, which can not be realized with size at most

$$\frac{\binom{n}{w}}{n^2} = \binom{\log\frac{1}{\varepsilon}}{w}^{O(1)} = \left(2 + \frac{1}{w}\log\frac{1}{\varepsilon}\right)^{O(w)}.$$

□

References

[1] R. Alweiss, S. Lovett, K. Wu, and J. Zhang. Improved bounds for the sunflower lemma. *arXiv preprint arXiv:1908.08483*, 2019.

[2] V. Arvind, J. Köbler, S. Kuhnert, G. Rattan, and Y. Vasudev. On the isomorphism problem for decision trees and decision lists. *Theoretical Computer Science*, 590:38–54, 2015.

[3] G. Bagallo and D. Haussler. Boolean feature discovery in empirical learning. *Machine learning*, 5(1):71–99, 1990.

[4] P. Beame. A switching lemma primer. Technical report, Technical Report UW-CSE-95-07-01, Department of Computer Science, 1994.

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

[6] A. Chattopadhyay, M. Mahajan, N. S. Mande, and N. Saurabh. Lower bounds for linear decision lists. *CoRR*, abs/1901.05911, 2019.

[7] A. Ehrenfeucht, D. Haussler, M. Kearns, and L. Valiant. A general lower bound on the number of examples needed for learning. *Information and Computation*, 82(3):247–261, 1989.

[8] T. Eiter, T. Ibaraki, and K. Makino. Decision lists and related boolean functions. *Theoretical Computer Science*, 270(1-2):493–524, 2002.

[9] E. Friedgut. Boolean functions with low average sensitivity depend on few coordinates. *Combinatorica*, 18(1):27–35, 1998.

[10] P. Gopalan, R. Meka, and O. Reingold. DNF sparsification and a faster deterministic counting algorithm. *Computational Complexity*, 22(2):275–310, 2013.

[11] D. Guijarro, V. Lavin, and V. Raghavan. Monotone term decision lists. *Theoretical Computer Science*, 259(1-2):549–575, 2001.

[12] T. Hancock, T. Jiang, M. Li, and J. Tromp. Lower bounds on learning decision lists and trees. *Information and Computation*, 126(2):114–122, 1996.

[13] J. Håstad. *Computational Limitations of Small-depth Circuits*. MIT Press, Cambridge, MA, USA, 1987.

[14] J. C. Jackson. An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences*, 55(3):414–440, 1997.

[15] M. Kearns, M. Li, L. Pitt, and L. Valiant. On the learnability of boolean formulae. In *Annual ACM Symposium on Theory of Computing: Proceedings of the nineteenth annual ACM conference on Theory of computing*, volume 1987, pages 285–295. Citeseer, 1987.

[16] R. Kohavi and S. Benson. Research note on decision lists. *Machine Learning*, 13(1):131–134, 1993.

[17] M. Krause. On the computational power of boolean decision lists. *computational complexity*, 14(4):362–375, 2006.

[18] S. Lovett and J. Zhang. DNF sparsification beyond sunflowers. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019, Phoenix, AZ, USA, June 23-26, 2019.*, pages 454–460, 2019.

[19] Z. Nevo and R. El-Yaniv. On online learning of decision lists. *Journal of Machine Learning Research*, 3(Oct):271–301, 2002.

[20] R. O'Donnell. *Analysis of boolean functions*. Cambridge University Press, 2014.

[21] A. A. Razborov. Bounded arithmetic and lower bounds in boolean complexity. In *Feasible Mathematics II*, pages 344–386. Springer, 1995.

[22] A. A. Razborov. Pseudorandom generators hard for k-dnf resolution and polynomial calculus resolution. *Annals of Mathematics*, pages 415–472, 2015.

[23] R. L. Rivest. Learning decision lists. *Machine learning*, 2(3):229–246, 1987.

[24] N. Segerlind, S. Buss, and R. Impagliazzo. A switching lemma for small restrictions and lower bounds for k-dnf resolution. *SIAM Journal on Computing*, 33(5):1171–1200, 2004.

[25] G. Turán and F. Vatan. Linear decision lists and partitioning algorithms for the construction of neural networks. In *Foundations of Computational Mathematics*, pages 414–423. Springer, 1997.

[26] F. Wang and C. Rudin. Falling rule lists. In *Artificial Intelligence and Statistics*, pages 1013–1022, 2015.

[27] I. H. Witten, E. Frank, M. A. Hall, and C. J. Pal. *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann, 2016.

ECCC ISSN 1433-8092

https://eccc.weizmann.ac.il