• 1400191306
copia 1

A Note on Learning Decision Lists

Jorge Castro

Report LSI–95–2–R

A note on learning decision lists

Jorge Castro
Dept. Llenguatges i Sistemes Informàtics
Universitat Politècnica de Catalunya
Email: castro@lsi.upc.es

January 13, 1995

Abstract
We show an algorithm that learns decision lists via equivalence queries, provided that a set G including all terms of the target list is given. The algorithm runs in time polynomial in the cardinality of G. From this learning algorithm, we prove that $\log n$-decision lists –the class of decision lists such that all their terms have low Kolmogorov complexity– are simple pac-learnable.

1 Introduction

In Valiant's original model of learning, "pac-learning" [7], one has to learn a target concept with high probability, in polynomial time, from a polynomial number of examples, within a certain error, under all probability distributions on the examples. This last requirement, to learn under all distributions, is a strong one. Many concept classes are not known to be polynomially learnable or known not to be polynomially learnable if RP $\neq$ NP, although some such concept classes are polynomially learnable under some fixed distributions. However, learning under one fixed distribution may be too restrictive to be useful.

Li and Vitányi proposed in [3] the simple pac-learning model with the aim to get a compromise between practicability –in the sense that more concept classes are learnable– and usefulness, understood as the property that guarantees that the concepts are learnable under a wide and interesting class of distributions. Roughly speaking, simple pac-learning model replaces the condition of learning under all distributions of Valiant's original model by the request of learning under all simple distributions, provided the sample is given according to the "universal" distribution (see next paragraph).

Simple distributions form a wide class that properly include all enumerable ones. Specifically, they are those multiplicatively dominated by the universal enumerable distribution $\mathbf{m}$. This distribution assigns high probabilities to low Kolmogorov complexity examples. As we will see later, these properties of $\mathbf{m}$ have nice consequences with regard to pac-learning under simple distributions.

With the new model, Li and Vitányi [3] developed a theory of learning for simple concepts –concepts with low Kolmogorov complexity– that intuitively should be polynomially learnable. In fact, they showed several examples that strengthen this intuition.

A decision list f over n Boolean variables is a sequence of terms $f_1 \dots f_t$, such that each term f_i, for $i = 1, \dots, t-1$, is a pair $\langle fm_i, fb_i \rangle$ where fm_i is a monomial and fb_i is either 0 or 1. The last term f_t is always $\langle 1, fb_t \rangle$. The value of a decision list f on a setting of the n Boolean variables is defined to be fb_i, where i is the least number such that fm_i is satisfied by the assignment. The class of k-decision lists consists of all decision lists f for which each monomial fm_i has at most k literals. Rivest gave in [5] an algorithm for learning k-decision lists, for each constant k.

We will show here an algorithm for exact learning, via equivalence queries, of decision lists. It works under the assumption that a set G, including all terms of the target list, is known. This algorithm runs in time polynomial in the cardinality of G. From this learning algorithm, we can solve the simple pac-learning problem for $\log n$-decision lists, cited as open in [3] (see also page 308 of [4]). Log n-decision lists is the class of all decisions lists f over n Boolean variables such that each term f_i of f has Kolmogorov complexity $O(\log n)$.

2 Preliminaries

We follow sections 5.5 and 5.6 of [4] and section 2 of [3]. In this paper we work with a discrete sample space S. The elements of S are called examples. A concept c is a subset of S. Abusing notation, we use c and the characteristic function $f : S \longrightarrow \{0,1\}$ of c interchangeably for the concept $c \subseteq S$ and for the syntactic representation of c. For a concept c, we denote by $l(c)$ the minimum length of the representations of c. A concept class C is a set of concepts. Fixed f as the target concept, a learning algorithm draws examples from the sample space S according to a fixed but unknown probability distribution P. Each example e comes with a label that shows the value of $f(e)$.

Definition 2.1 A concept class C is pac-learnable iff there exists a learning algorithm A such that, for each $f \in C$ and ϵ $(0 < \epsilon < 1)$, algorithm A always halts within time and number of examples $p(l(f), 1/\epsilon)$, for some polynomial p, and outputs a concept $h \in C$ which satisfies

$$\mathrm{Prob}(P(h \neq f) < \epsilon) > 1 - \epsilon.$$

Note that the pac-learning model requires that the algorithm learns under all distributions (the distribution P in the definition is unknown). We can also define, in a natural way, pac-learning under a class Δ of distributions: simply requiring definition 2.1 only on distributions from Δ.

It can be proved that there exists an universal enumerable distribution, denoted by $\mathbf{m}$, that multiplicatively dominates each enumerable distribution. It can be shown that

$$\mathbf{m}(x) = 2^{-K(x)+O(1)},$$

where $K(x)$ denotes the Kolmogorov complexity of x. The universal distribution has many important properties. Under $\mathbf{m}$, easily describable objects have high probability, and complex objects have low probability.

Definition 2.2 A distribution P is simple iff it is multiplicatively dominated by the universal distribution. That is, there exists a constant c, such that for all x,

$$c\mathbf{m}(x) \geq P(x).$$

It can be shown that simple distributions properly include enumerable ones and that there is a distribution which is not simple. The following theorem relates learning under simple distributions with learning under the universal distribution $\mathbf{m}$.

Theorem 2.3 ([3]) A concept class C is pac-learnable under the universal distribution $\mathbf{m}$, iff it is pac-learnable under simple distributions, provided that in the learning phase the set of examples is drawn according to $\mathbf{m}$.

This theorem says roughly that a concept class is polynomially learnable under $\mathbf{m}$ if and only if it is polynomially learnable under each simple distribution. In [3] it is shown how to exploit this completeness theorem to obtain new learning algorithms. Now, we define

Definition 2.4 A concept class C is simple pac-learnable iff it is pac-learnable under $\mathbf{m}$.

We assume the reader knows the exact learning or learning via queries models (see [1] and [2]). In particular, we will work with the learning via equivalence queries model. Roughly speaking, in this model the learning algorithm has to produce in polynomial time a representation of the target concept, doing a polynomial number of equivalence queries. An equivalence query asks whether a representation given by the algorithm is a representation of the target concept. Equivalence queries are answered by a teacher affirmatively –when the asked representation is a representation of the target concept– or with a counterexample that shows the difference.

A pac-learning algorithm can be obtained from a learning via equivalence queries algorithm by means of a now standard transformation (see [1] and [2]):

Theorem 2.5 Let C be a concept class. If C is learnable via equivalence queries then C is pac-learnable.

In the next section we will use this theorem to show a simple pac-learning algorithm for $\log n$-decision lists. We will proceed as follows. First we will give an algorithm that, assuming a set G including all terms of the target list is known, learns the target list doing equivalence queries in time polynomial in the cardinality of G. After that, we will show that if the target list is a $\log n$-decision list and we draw a polynomial number of examples according to the universal distribution $\mathbf{m}$, with high probability we can obtain a set G of polynomial size that includes all terms of the target list. From these facts we may conclude that $\log n$-decision lists are simple pac-learnable.

3 Learning decision lists

Let $f = f_1 \dots f_t$ be a decision list with $f_i = \langle fm_i, fb_i \rangle$, for $i = 1, \dots, t-1$, and $f_t = \langle 1, fb_t \rangle$. Let $G = \{g_1, \dots, g_s\}$ be a set of terms with $g_i = \langle gm_i, gb_i \rangle$ for $i = 1, \dots, s$, and let us assume that $\{f_1, \dots, f_t\} \subseteq G$.

We consider the following algorithm *Learn_Dlist* that knowing the set G, tries to learn f doing equivalence queries. Here E denotes a set of labeled examples according to f, obtained from previous queries.

```
function Learn_Dlist ($G$) ret $g$ : decision_list
$E := \emptyset$
$g$ :=Group_list($G, E$)
Ask the equivalence query $g = f$?
while the teacher does not reply "yes" do
    Let $e$ be the new counterexample
    Let $g_l = \langle gm_l, gb_l \rangle$ be the first term of $g$ such that $gm_l(e) = 1$
    $E := E \cup \langle e, \overline{gb_l} \rangle$
    $g$ :=Group_list($G, E$)
    Ask the equivalence query $g = f$?
endwhile
return $g$
endfunction
```

The core of the algorithm is the function *Group_list*. Intuitively, this function returns a decision list g by grouping all terms of G according to their behaviors on E. Given a labeled example $\langle e, b \rangle$ of E and a term g_i of G, we say that $\langle e, b \rangle$ is a consistency (resp. an inconsistency) of g_i iff $gm_i(e) = 1$ and $gb_i = b$ ($gb_i = \overline{b}$). Each term g_i of G appears in g placed in the first group of terms with the following property: each inconsistency of g_i is a consistency of some term in a previous group. The function *Group_list* is formally defined as follows (here ++ denotes list concatenation):

```
function Group_list ($G', E$) ret $g$ : decision_list
for all $i = 1, \dots, s$ do
    Let $C_i$ be the set of labeled examples $\langle e, b \rangle \in E$
    such that $gm_i(e) = 1$ and $gb_i = b$
    (* let us call $C_i$ the consistent set of the term $g_i$ *)
    Let $I_i$ be the set of labeled examples $\langle e, b \rangle \in E$
    such that $gm_i(e) = 1$ and $gb_i = \overline{b}$
    (* let us call $I_i$ the inconsistent set of the term $g_i$ *)
endfor
$g$ := nul_list
while $G' \neq \emptyset$ do
    Let $Newgroup$ be the list of terms of $G'$ such that their inconsistent
    sets are empty. It does not matter the order in the list
    $g := g$++$Newgroup$
    $G' := G' - \{$terms of $Newgroup\}$
    for all $g_i \in G'$ do
        for all $g_j \in Newgroup$ do
            $I_i := I_i - (I_i \cap C_j)$
        endfor
    endfor
    (* consistent examples of terms in $Newgroup$ have been erased
       from the inconsistent sets *)
endwhile
return $g$
endfunction
```

Assuming that *Group_list* halts, each time that it is called it generates a new list g, in such a way that $g = gr(g,1)$++$\cdots$++$gr(g,k)$, where $gr(g,1)$ –the first group of g– is a list of terms having no inconsistencies in E, and in general, $gr(g, i+1)$ is a list of terms whose inconsistencies in E are included in the union of the consistent sets of terms in previous groups. Now, we are ready to prove the following lemma

Lemma 3.1 Assuming that G includes all terms of f, the algorithm *Learn_Dlist* halts and learns f in time polynomial in s, where s is the cardinality of G.

Proof We have just to prove that *Learn_Dlist* runs in time polynomial in s. We proceed by steps.

Fact 1. The function *Group_list* halts in time polynomial in s and the cardinality of E. We show that *Newgroup* is always non empty. At the beginning of the process of *Group_list*, the set G' is G and includes all terms $f_1, \ldots, f_t$ of the target list f. The first group $gr(g,1)$ is always nonempty because f_1 does not have inconsistencies. Let us suppose that $gr(g,i)$ is the last group constructed and $G' \neq \emptyset$. Either, all terms of f are already in a group, or there exists a term of f in G'. In the first case, the term $f_t = \langle 1, fb_t \rangle$ of f is in a group. As each example of E is either consistent or inconsistent for f_t, we may conclude that all inconsistent sets of terms in G' have already been erased. So, $Newgroup$ is G'.

In the second case, let f_{σ_i} the first term of f that belongs to G'. As E is a set of examples of f, the inconsistent set of f_{σ_i} is empty because all its inconsistencies in E are included in the union of the consistent sets of $f_1, \ldots, f_{\sigma_i - 1}$ that, by hypothesis, belong to previous groups. So, f_{σ_i} belongs to *Newgroup*. □

Fact 2. The decision list g returned by *Group_list* has at most $t+1$ groups of terms, where t is the number of terms of f.

Using ideas from above it can be shown the following: $f_1 \in gr(g,1)$, so $f_2 \in gr(g,1) \cup gr(g,2)$ and, in general, $f_i \in gr(g,j)$ and $j \leq i$. So, $f_t = \langle 1, fb_t \rangle \in gr(g,j)$ and $j \leq t$. Now the result is clear. □

Fact 3. Let g_i be a term of G and let $gr(g, \sigma_i)$ be the group of g_i before processing *Group_list*. Let $gr(g, \mu_i)$ be the group of g_i after processing *Group_list*. It holds that $\sigma_i \leq \mu_i$.
When a new counterexample is put in E the inconsistent set of g_i cannot decrease. A contradiction appears if we suppose $\sigma_i > \mu_i$ for some $1 \leq i \leq s$. □

Fact 4. The term g_l has to change of group by processing *Group_list*.
Let us suppose that $g_l \in gr(g, i)$ before processing *Group_List*. Note that the last counterexample e will be put in the inconsistent set of g_l and will not be put in the consistent sets of terms in $gr(g, j)$ with $j < i$. Now, by fact 3, it is clear that g_l has to change of group by processing *Group_List*. □

Fact 5. Each term g_i changes of group at most t times.
It is a consequence of facts 2 and 3. □

Fact 6. The function *Learn_Dlist* halts in time polynomial in s.
By facts 4 and 5, the *while* instruction of *Learn_Dlist* is processed at most st times. Now, by fact 1 we get the result. □

Now, let us suppose that f is a $\log n$-decision list. The following lemma says that, drawing examples under the universal distribution, with high probability a set G including all terms of f can be obtained in polynomial time.

Lemma 3.2 Let f be a $\log n$-decision list and let $f_i = \langle fm_i, fb_i \rangle$ for $i = 1, \ldots, t$ be the terms of f. Let c be a constant such that for all $i = 1, \ldots, t$, it holds $K(fm_i) < c \log n$. Let ϵ be a real number such that $0 < \epsilon < 1$. There exists an integer n_c depending only on c such that for all $n \geq n_c$, if we draw n^{c+2}/ϵ examples according to $\mathbf{m}$, with probability greater than $1 - \epsilon$ a set G including all terms of f can be obtained in time $O(n^{2(c+2)}/\epsilon^2)$.

Proof The idea is from [3] (see also [4]). If we draw n^{c+2}/ϵ examples according to $\mathbf{m}$, it holds the following,

Fact. For all $n \geq n_c$, where n_c is an integer depending only of c, with probability greater than $1 - \epsilon$ all examples of the following form will be drawn.

> For each monomial m over n variables with $K(m) < c \log n$: the example vectors 0_m, defined as the vectors that satisfy m and have 0 entries for all variables not in m; the example vectors 1_m, defined as the vectors that satisfy m and have 1 entries for all variables not in m.

This fact is shown in the proof of theorem 3 of [3]. Now, we obtain the set G with the following procedure:

```
Draw $n^{c+2}/\epsilon$ examples according to $\mathbf{m}$. Let $E$ be this set of examples.
$G := \emptyset$
For each pair of examples in $E$ do
    Let $m$ be the monomial which contains $x_i$ if both examples have '1' in
    position $i$, contains $\overline{x}_i$ if both examples have '0' in position $i$, and does
```

```
        not contain variable $x_i$ otherwise $(1 \leq i \leq n)$
        $G := G \cup \{\langle m, 0\rangle, \langle m, 1\rangle\}$
    endfor
```

Let us call this procedure *Get_set*. Using the fact above, it is clear that *Get_set* satisfies the lemma. □

Finally, we are ready to show simple pac-learning for $\log n$-decision lists. Let f be as in lemma 3.2 and let t be the time of *Learn_Dlist* running on a set that includes all terms of f. With probability at least $1 - \epsilon$, the algorithm

```
Get_set($G$)
Let $s \in O(n^{2(c+2)}/\epsilon^2)$ be the cardinality of $G$
Simulate Learn_Dlist for $t(s)$ steps
```

learns f exactly. Removing the equivalence queries in the standard way (as in the proof of theorem 2.5, see [1] and [2]) we obtain a pac-learning algorithm, say A, with the following property,

$$\begin{aligned}
&\mathrm{Prob}\,(\mathbf{m}(g_A \neq f) < \epsilon) \geq \\
&\mathrm{Prob}\,(\mathbf{m}(g_A \neq f) < \epsilon | \{f_1, \ldots, f_t\} \subseteq G) \cdot \mathrm{Prob}\,(\{f_1, \ldots, f_t\} \subseteq G) > \\
&(1-\epsilon)(1-\epsilon) \geq 1 - 2\epsilon
\end{aligned}$$

where g_A denotes the output list of A. Therefore, we have shown

Theorem 3.3 Let $\mathcal{D}_c$ be the class of $\log n$-decision lists having all their terms Kolmogorov complexity bounded by $c \log n$. The $\mathcal{D}_c$ classes are simple pac-learnable.

4 Conclusions

Li and Vitányi gave in [3] several examples of simple pac-learning. The outline of their proofs is the following. First a certain property P –that depends on the class $\mathcal{C}$ they try to learn– is shown to happen with high probability drawing examples under $\mathbf{m}$. Second, assuming that property P happens, a pac-learning distribution-free algorithm –frequently an Occam algorithm– for $\mathcal{C}$ is given. This work shows that, in some cases, a learning via equivalence queries algorithm may be easier to get than a pure pac-learning distribution-free or an Occam one.

We have some evidences of the fact that the similar learning scheme used in this paper, namely:

1. With high probability get property P.

2. Assuming P, try a learning via equivalence queries algorithm for $\mathcal{C}$.

can be improved in such a way that more general classes of decision lists would be simple pac-learnable. If we confirm these results, they will be written in a separate paper.

In a recent paper [6], an algorithm for exact learning, via equivalence queries, of decision lists has been independently obtained by Simon. Given a set of Boolean functions F it can be defined, in a natural way, the class $\mathcal{D}_F$ of decision lists having all their terms functions in F. Assuming that F is known, Simon's algorithm, denoted by *Declist*, learns $\mathcal{D}_F$ in time polynomial in the cardinality of F. Algorithms *Declist* and *Learn_Dlist* are similar. Note that if F is known, the set G including all terms of the target list needed by *Learn_Dlist* can be obtained in linear time, just consider:

$$G = \bigcup_{f \in F} \{\langle f, 0\rangle, \langle f, 1\rangle\}.$$

Acknowledgements

I discussed ideas of this work with José L. Balcázar who introduced me topics of learning theory. Ricard Gavaldà showed me the proof of theorem 2.5.

References

1. D. Angluin. Learning regular sets from queries and counterexamples. *Information and Computation*, 75:87–106, 1987.

2. D. Angluin. Queries and concept learning. *Machine Learning*, 2:319–342, 1988.

3. M. Li and P. Vitányi. Learning simple concepts under simple distributions. *SIAM Journal of Computing*, 20(5):911–935, 1991.

4. M. Li and P. Vitányi. *An introduction to Kolmogorov complexity and its applications.* Springer-Verlag, 1993.

5. R. Rivest. Learning decision lists. *Machine Learning*, 2:229–246, 1987.

6. H. Simon. Learning decision lists an trees with equivalence queries. To appear in EUROCOLT 95.

7. L. Valiant. A theory of the learnable. *Comm. ACM*, 27:1134–1142, 1984.