多分岐の決定木型 ブースティングアルゴリズム

畑埜晃平
東京工業大学 数理・計算科学専攻
〒152-8552 東京都目黒区大岡山 2-12-1
hatano@is.titech.ac.jp

概要

本研究では，Mansour と McAlleter が提案した多分岐の決定木学習アルゴリズムに対してより単純な解析を行う．ここで多分岐の決定木とは各内部頂点の分岐数が 2 以上の決定木で，内部頂点ごとに分岐数が異なってもよいものとする．彼らは，2 分岐の決定木構成アルゴリズムを多分岐の場合に拡張し，葉の個数を固定した時，拡張されたアルゴリズムの出力する多分岐の決定木の誤差が拡張前のアルゴリズムの出力する 2 分岐の決定木の誤差の上限を越えない事を示した．我々は，より緩やかで単純な条件の下でも彼らの結果が成り立つ事を示す．

Simplified analysis of a decision tree boosting algorithm with multi-way branching

Kohei Hatano
Department of Mathematicala and Computing Sciences,
Tokyo Institute of Technology
Ookayama 2-12-1,Meguro-ku,Tokyo,152-8552,Japan
hatano@is.titech.ac.jp

Abstract

We show an simpler analysis of a decision tree learning algorithm proposed by Mansour and McAlleter. They extend a binary decision tree learning algorithm to a multi-way branching one and showed that the error of the multi-way branching tree produced by the extended algorithm is bounded with an upperbound of the error of the binary tree of the same size produced by non-extended one. We show that their result holds under a simpler and weaker condition.

1 Introduction

Decision tree learning is one of the most successful areas in machine learning literature. It is simple and experimentally effective[5][2]. But theoretical guarantee hadn't been given for many years. First theoretical guarantee was given by Kearns and Mansour[4]. The key is "boosting." Boosting is an technique to construct a "strong" hypothesis based on "weak" hypotheses. AdaBoost[3] is a representative boosting algorithm. For binary classification problem,Kearns and Mansour assumed weak hypotheses and showed that decision tree learning algorithm performed boosting. There are some extensions of their results. Takimoto and Maruoka generalized Kearns and Mansour's result to multi-class function learning.[6]. Here we analyze Mansour and McAllester 's work:They extended Kearns and Mansour's algorithm that produces a binary branching decision tree to that produces a multi-way branching decision tree.

Most decision tree learning algorithm use a top-down growth process. Given a current tree, the algorithm chooses a leaf and extend it to an internal node labeled with "branching function"and adding a leaf to each possible output value of the branching function.

Decision tree can over-fit the data. It is easy to construct the tree that divides all the data,whose training error is zero. But such tree might have unnecessary rules dependent with the data then, the generalization error might be large. Occam Razor Principle[1] says that small training error and small size of the tree imply small generalization error. Thus major decision tree learning algorithms have pruning phase:After constructing the tree,the algorithm prunes away some of its nodes. Here we take an another approach. We fix the target size s in advance and consider the problem of constructing a tree T with $|T| = s$ and the training error small as possible, where $|T|$ is the number of leaves. If we choose sufficiently small target size, then the generalization error is small.

We consider the situation that the algorithm is given multi-way branching functions,for example four-way branching function, as well as binary ones. We would like the algorithm using multi-way branching functions performs better than the algorithm using only binary branching functions. Mansour and McAllester extend a binary decision tree learning algorithm to a multi-way branching one. They showed that the extended algorithm remain effective. We show a simpler proof for their result.

2 Preliminaries

We introduce our learning model briefly. Our model is based on PAC learning model proposed by Valiant[7]. Let X denote an instance space. We assume the unknown target function $f : X \rightarrow \{0, 1\}$. Learner is given a *sample* $S = (\langle x_1, f(x_1)\rangle, \ldots, \langle x_1, f(x_m)\rangle)$,where each x_i is drawn independently randomly with respect to an unknown distribution P over X. The goal of the learner is to output an hypothesis $h_f : X \rightarrow \{0, 1\}$ such that its *generalization error*

$$\epsilon(h_f) \overset{\text{def}}{=} \Pr_P[f(x) \neq h_f(x)]$$

is smaller than any given constant ε $(0 < \varepsilon \leq 1)$. In order to accomplish the goal, it is sufficient to design learning algorithms based on "Occam Razor."[1] Namely,

1. *training error*

$$\hat{\epsilon}(h_f) \overset{\text{def}}{=} \Pr_D[f(x) \neq h_f(x)]$$

   is small,where D is the uniform distribution over S.

2. $size(h_f) = o(m)$,where $size(\cdot)$ is the length of the bit string for h under some fixed encoding scheme.

When a hypothesis h_f is represented as a decision tree, the second condition can be interpreted that the number of leaves of the tree is sufficiently small with respect to the number of data.

We introduce the notion of boosting. In the classical definition,boosting is to construct a hypothesis h_f such that $\Pr_D[f(x) \neq h_f(x)] \leq \varepsilon$ for any given ε combining hypotheses h satisfying $\Pr_D[f(x) \neq h_f(x)] \leq 1/2 - \gamma$.

But ,in this paper,we measure the goodness of hypotheses from an information-theoretic view point. Here we use an extended entropy proposed by Takimoto and Maruoka[6].

Definition 2.1. A function $G : [0,1]^2 \to [0,1]$ is *pseudo-entropy function* if,for any $q_0, q_1 \in [0,1] : q_0 + q_1 = 1$,

1. $\min(q_0, q_1) \leq G(q_0, q_1)$,
2. $G(q_0, q_1) = 0 \iff q_0 = 1$ or $q_1 = 1$,
3. G is concave.

For example, Shannon entropy function $H_2(q_0, q_1) = q_0 \log q_0 + q_1 \log q_1$ is a pseudo entropy function. We define the entropy of function f using a pseudo entropy function G.

Definition 2.2. *G-entropy* of f with respect to D,denoted $H_G^D(f)$, is defined as

$$H_D^G(f) \stackrel{\text{def}}{=} G(q_0, q_1),$$

where $q_i = \Pr_D[f(x) = i]$ $(i = 0, 1)$.

We can interpret G-entropy as "impurity" of value of f under the distribution D. For example, if f takes only one value, G-entropy becomes the minimum. If the value of f is random, G-entropy becomes the maximum.

Then we define the conditional G-entropy given a hypothesis $h : X \to Z$, where Z is a finite set but possibly different from $\{0, 1\}$.

Definition 2.3. *Conditional G-entropy* of f given h with respect to D, denoted as $H_D^G(f|h)$, is defined as

$$H_D^G(f|h) \stackrel{\text{def}}{=} \sum_{z \in Z} \Pr_D[h(x) = z] G(q_{0|z}, q_{1|z}),$$

where $q_{i|z} = \Pr_D[f(x) = 1|h(x) = z]$ $(i = 0, 1)$.

Now we show the relationship between the error and G-entropy. Since the range of h may be different from that of f, we have to give a way to interpret h's values.More precisely,We define a mapping $M : Z \to \{0, 1\}$:

$$M(z) \stackrel{\text{def}}{=} \arg_{i \in \{0,1\}} \max q_{i|z}$$

Then, the following fact holds.

Proposition 2.4.

$$\Pr_D[f(x) \neq M(h(x))] \leq H_D^G(f|h)$$

Proof.

$$\begin{aligned}
&\Pr_D[f(x) \neq M(h(x))] \\
=&\sum_{z \in Z} w_z \Pr_D[f(x) \neq M(h(x))|h(x) = z] \\
=&\sum_{z \in Z} w_z \{1 - \max(q_{0|z}, q_{1|z})\} \\
=&\sum_{z \in Z} w_z \min(q_{0|z}, q_{1|z}) \\
\leq&\sum_{z \in Z} w_z G(q_{0|z}, q_{1|z}) = H_D^G(f|h)
\end{aligned}$$

□

Now we state our assumption. We assume a set of "weak hypotheses" of the target function f.

Definition 2.5. H is a γ*-weak-hypothesis-set* for f if for any sample S of f and any distribution D over S, there exists a hypothesis $h \in H$ satisfying

$$H_D^G(f) - H_D^G(f|h) \geq \gamma H_D^G(f).$$

We call this constant γ *advantage.*

Later, we sometimes refer to $H_D^G(f) - H_D^G(f|h)$ as *gain.*

3 Algorithm(binary case)

We show an simple decision tree learning algorithm proposed by Kearns and Mansour[4].

We assume some pseudo-entropy function G and a set of binary branching functions H_2. The algorithm is given as input a sample S and target size s $(s \geq 2)$,then outputs a decision tree. Let T be any decision tree for output. Internal nodes of T are labeled with functions in H_2. Let $L(T)$ denote the set of leaves of T. Then T can be interpreted as a mapping any instance $x \in X$ to a leaf in $L(T)$. Let D be the uniform distribution over S. For each $\ell \in L(T)$, $w_\ell \overset{\text{def}}{=} \Pr_D[T(x) = \ell]$ and $q_{i|\ell} \overset{\text{def}}{=} \Pr_D[f(x) = i|T(x) = \ell]$ $(i = 0, 1)$.

Now we show the way to put label on the leaves of the decision tree. When given a sample S, let S_ℓ be the subset of S where each labeled example $\langle x, f(x)\rangle$ reaches the leaf ℓ. Then, for each leaf ℓ, we choose the most major label in S_ℓ. Formally, the way to label is defined as the following function that maps $L(T)$ to $\{0,1\}$:

$$M(\ell) \overset{\text{def}}{=} \arg\max(q_{0|\ell}, q_{1|\ell})$$

Then,we define the training error of T $\hat{\epsilon}(T)$ is defined as

$$\hat{\epsilon}(T) \overset{\text{def}}{=} \Pr_D[f(x) \neq M(T(x))].$$

We denote the conditional G-entropy of f given T as

$$H_D^G(f|T) \overset{\text{def}}{=} \sum_{\ell \in L(T)} w_\ell G(q_{0|\ell}, q_{1|\ell}).$$

Then,by from proposition 2.4, the following holds.

$$\hat{\epsilon}(T) \leq H_D^G(f|T) = \sum_{\ell \in L(T)} w_\ell G(q_{1|\ell}, \ldots, q_{N|\ell}).$$

Thus, if the conditional G entropy of f given T is sufficiently small,so is the training error of T.

Now we are ready to describe the algorithm **BIN-TOPDOWN**$_{\mathbf{G,H_2}}$. Given a sample S and target size s, the algorithm repeat the following sequence of procedures.

1. Choose the leaf ℓ that maximizes $w_\ell G(q_{0|\ell}, q_{1|\ell})$.
2. Calculates the sample S_ell and the uniform distribution D_ℓ over S_ℓ.
3. Choose the binary hypothesis $h \in H_2$ that maximizes the gain $(H_{D_l}^G(f) - H_{D_l}^G(f|h))$.
4. Expand the leaf ℓ with h : replace ℓ with an internal node labeled with h and create two new child leaves. (We denote the expanded tree as $T_{\ell,h}$.)

Finally the algorithm outputs the tree T with $|T| = s$. We give the detail in Figure 1.

```
BIN-TOPDOWN_{G,H_2}(S, s)
begin
  T ← the single-leaf tree;
  While |T| < s do
    $\ell \leftarrow \arg\max_{\ell \in L(T)} w_\ell \cdot G(q_{1|\ell}, q_{2|\ell}, \ldots, q_{N|\ell})$;
    $S_\ell \leftarrow \{\langle x, f(x)\rangle \in S | T(x) = \ell\}$;
    $D_\ell$ ← the uniform distribution over $S_\ell$;
    $h \leftarrow \arg\max_{h \in H_2}(H_{D_\ell}^G(f) - H_{D_\ell}^G(f|h))$;
    $T \leftarrow T_{\ell,h}$;
    end-while
  Output T ;
end.
```

Figure 1: Algorithm **BIN-TOPDOWN**$_{\mathbf{G,H_2}}$

We note that if H_2 is a γ-weak-hypothesis-set, **BIN-TOPDOWN**$_{\mathbf{G,H_2}}$ reduce the conditional G entropy at each expansion. By definition, there exists a hypothesis $h \in H$ whose advantage is greater than some constant γ. Because h maximizes the gain,

$$H_{D_\ell}^G(f) - H_{D_\ell}^G(f|h) \geq \gamma H_{D_\ell}^G(f).$$

Let C_ℓ be the set of new leaves created at the expansion of ℓ. Then, following relation holds:

$$\begin{aligned} &H_D^G(f|T) - H_D^G(f|T_{\ell,h}) \\ =&w_\ell(H_{D_\ell}^G(f) - H_{D_\ell}^G(f|h)) \\ \geq&w_\ell \gamma H_{D_\ell}^G(f). \end{aligned}$$

This implies each expansion reduce the conditional G-entropy.

Kearns and Mansour proved the following theorem.

Theorem 3.1 (Kearns and Mansour[4]). *Assume that H_2 is a γ-weak-hypothesis-set for f.Then* **BIN-TOPDOWN**$_{\mathbf{G,H_2}}(S, s)$ *outputs T with $\hat{\epsilon}(T) \leq H_G^D(f|T) \leq s^{-\gamma}$.* ∎

We ommit the proof here. Later, we show a generalized theorem and a proof for the theorem.

```
MULTI-TOPDOWN_{G,H}(S, s)
begin
  $T \leftarrow$ the single-leaf tree;
  While $(|T| < s)$ do
    $\ell \leftarrow \arg\max_{\ell \in L(T)} w_\ell G(q_{1|\ell}, q_{2|\ell})$;
    $S_\ell \leftarrow \{\langle x, f(x)\rangle \in S | T(x) = \ell\}$;
    $D_\ell \leftarrow$ the uniform distribution over $S_\ell$ ;
    $h \leftarrow \arg\max_{h \in H, \text{acceptable}} \frac{H^G_{D_\ell}(f) - H^G_{D_\ell}(f|h)}{\lceil \log K_h \rceil}$
    $T \leftarrow T_{\ell,h}$;
    end-while
  Output $T$ ;
end.
```

Figure 2: Algorithm **MULTI-TOPDON$_{\mathbf{G,H}}$**

4 Algorithm(multi-way case)

We now show Mansour and McAllester's decision tree learning algorithm that uses multiway branching functions as well as binary ones.

Let H be a set of branching functions where each $h \in H$ is a function from X to $\{1, \ldots, K_h\}$ $(K_h \geq 2)$ .We allow different functions in H have different ranges. We assume H contains a set of binary branching functions H_2.

We say that a branching function h is *acceptable* for tree T and target size s if either $|K_h| = 2$ or $|T| \leq s/|T|$.

We show the detail of the algorithm is in Figure 2.

We remark that if $H_2 \subset H$ is a γ-weak-hypothesis-set and K-way branching function h is selected for ℓ, then,

$$H^G_{D_\ell}(f) - H^G_{D_\ell}(f|h) \geq \gamma\lceil \log K \rceil H^G_{D_\ell}(f).$$

5 Analysis

We give an analysis for the algorithm **MULTI-TOPDOWN$_{\mathbf{G,H}}$**. For any leaf ℓ, we define *weight* of ℓ W_ℓ as

$$W_\ell \overset{\text{def}}{=} w_\ell G(q_{0|\ell}, q_{1|\ell}).$$

Then following facts holds.

Fact 5.1. 1. $\hat{\epsilon}(T) \leq \sum_{\ell \in L(T)} W_\ell$.

2. When $H_2 \subset H$ is a weak hypothesis set, for parent leaf ℓ and its children leaves $1, \ldots, K$,

$$W_1 + \ldots, W_K \leq (1 - \gamma\lceil \log K \rceil) W_\ell.$$

We review the algorithm focusing on weights. In each iteration, algorithm chooses the leaf that maximize the weight,and expand it.Then it holds that sum of weights of children leaves is bounded by $(1-\gamma)$ times the weight of their parent leaf. Now we consider the worst case. Then our problem can be formalized as follows.

Weight Distribution Problem

- The player is given a single-leaf tree where weight of the leaf is W, $(0 \leq W \leq 1)$,and target size s $(s \geq 2)$.
- While $|T| < s$, the player repeat the following procedures:
  1. Choose the leaf ℓ with the maximum weight
  2. Choose any integer K (≥ 2) with $K = 2$ or $K \leq s/|T|$.
  3. Expand the leaf ℓ: replace the leaf with an internal node with K children leaves. For weight of parent leaf ℓ W_ℓ and weights of children leaves $W_1, \ldots, W_K$, the following equation holds:

     $$W_1 + \cdots + W_K = (1 - \gamma\lceil \log K \rceil) W_\ell.$$
- The player distribute weights of children leaves freely.

Player's goal: To maximize the sum of weights of the tree.
Problem: What is the best strategy for the player?

Lemma 5.2. *If the player always chooses the number of children leaves $K = 2$ and distribute weights of children leaves uniformly, the sum of weight of the tree is maximized.*

From lemma 5.2, we can show our main result.

Theorem 5.3. *Assume that $H_2 \subset H$ is a γ-weak-hypothesis-set for f. Then,* **MULTI-TOPDOWN$_{\mathbf{G,H}}$**(S, s) *outputs T with $\hat{\epsilon}(T) \leq H^D_G(f|T) \leq s^{-\gamma}$.*

5.1 Analysis of the weight distribution problem

We say that a K-dimensional vector $d_K = (a_1, \ldots, a_K) \in [0,1]^K$ is a *distribution* of size K if $\sum_{i=1}^{K} a_i = 1$. Let D_K denote the set of all possible distributions of size K. Especially, let d^*_K be the uniform one,i.e.,$d^*_K = (1/K, \ldots, 1/K)$. We also define $D = \cup_{K \geq 2} D_K$.

Then we consider the sequence of distributions correspond to the sequence of branching functions that are acceptable for s. For notational convention,we say that such sequences are acceptable. Formally, a sequence of distributions $d_{K_1}, \ldots, d_{K_t}$ $(t \geq 1)$ is *acceptable* for s if $s = (K_1 - 1) + \cdots + (K_t - 1) + 1$ and $K_i \leq s/\{(K_1 - 1) + \cdots + (K_{i-1}) + 1\}$ $(1 \leq i \leq s)$. For any $W \in [0,1]$, any integer $t \geq 1$, and any sequence of distribution $d^{(1)}_{K_1}, \ldots, d^{(t)}_{K_t} \in D$, We denote $\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, d^{(1)}_{K_1}, \ldots, d^{(t)}_{K_t})$ as the sum of weghts of leaves such that

1. The initial weight is W.
2. The ℓ th way to distriute is specified with $d^{(\ell)}_{K_\ell}$ $(1 \leq \ell \leq t)$.

We can represent lemma 5.2 as follows.

Lemma 5.4. *For any weight W,any integer $s \geq 2$, any sequence of distributions $d_{K_1}, \ldots, d_{K_t}$ that is acceptable for s, and the sequence of distributions with length $s-1$ $d^*_2, \ldots, d^*_2$,*

$$\begin{aligned}&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, d_{K_1}, \ldots, d_{K_t})\\ \leq&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d^*_2, \ldots, d^*_2}_{s-1})\\ \leq& s^{-\gamma}.\end{aligned}$$

To prove our lemma 5.4, we need two more lemmas.

Lemma 5.5. *For any weight $W \in [0,1]$,any integer $s \geq 2$, any sequence of distributions $d_{K_1}, \ldots, d_{K_t}$ that is acceptable for s, and the sequence of distributions with length $s-1$ $d^*_2, \ldots, d^*_2$,*

$$\begin{aligned}&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d_K, d^*_2, \ldots, d^*_2}_{t})\\ \leq&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d^*_2, \ldots, d^*_2}_{s-1}).\end{aligned}$$

Proof. We consider the situation that the player does not choose the leaf whose weight is the maximum,but select the leaf in the oldest generation that has the the maximum weight among the generation. We denote the sum under such a situation as $\mathrm{sum}'_{\gamma\lceil\log k\rceil}$. Then,

$$\begin{aligned}&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d_K, d^*_2, \ldots, d^*_2}_{t})\\ \leq&\mathrm{sum}'_{\gamma\lceil\log k\rceil}(W, \underbrace{d_K, d^*_2, \ldots, d^*_2}_{t}).\end{aligned}$$

We show that the following inequality.

$$\begin{aligned}&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d^*_K, d^*_2, \ldots, d^*_2}_{t})\\ \leq&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d^*_2, \ldots, d^*_2}_{s-1}).\end{aligned}$$

Let $s = 2^i + s'$ $(2^i \leq s \leq 2^{i+1},\ s' \geq 0)$. Then,

$$\begin{aligned}&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d^*_2, \ldots, d^*_2}_{s-1})\\ =&(1-\gamma)^i W - s'\gamma\frac{(1-\gamma)^i}{2^i}W\\ =&(1-\gamma)^i\left(1 - \frac{s'\gamma}{2^i}\right)W\\ \stackrel{\mathrm{def}}{=}&\phi_\gamma(s)W.\end{aligned}$$

On the other hand, let $s = K2^{i'} + s''$ $(K2^{i'} \leq s \leq K2^{i'+1},\ s'' \geq 0)$.

$$\begin{aligned}&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d^*_K, d^*_2, \ldots, d^*_2}_{t})\\ =&(1-\gamma\lceil\log K\rceil)(1-\gamma)^{i'}W\\ &- s''\gamma\frac{(1-\gamma\lceil\log K\rceil)(1-\gamma)^i}{K2^{i'}}W\\ =&(1-\gamma\lceil\log K\rceil)(1-\gamma)^{i'}\left(1 - \frac{s''\gamma}{K2^{i'}}\right)W\\ =&(1-\gamma\lceil\log K\rceil)\phi_\gamma(s/K)W\end{aligned}$$

$\phi_\gamma(2s) = (1-\gamma)\phi_\gamma(s)$ and ϕ_γ is monotone non-increasing function.Thus,

$$\frac{\phi_\gamma(s)}{(1-\gamma)^{\lfloor \log K \rfloor}} \le \phi_\gamma\left(\frac{s}{K}\right) \le \frac{\phi_\gamma(s)}{(1-\gamma)^{\lceil \log K \rceil}}.$$

Now the following inequality holds.

$$\begin{aligned}
&\mathrm{sum}_{\gamma\lceil \log k\rceil}(W, \underbrace{d_K^*, d_2^*, \ldots, d_2^*}_{t}) \\
=&(1-\gamma\lceil \log K\rceil)\phi_\gamma(s/K)W \\
\le&(1-\gamma\lceil \log K\rceil)\frac{\phi_\gamma(s)}{(1-\gamma)^{\lceil \log K\rceil}} \\
\le&\phi_\gamma(s) \\
=&\mathrm{sum}_{\gamma\lceil \log k\rceil}(W, \underbrace{d_2^*, \ldots, d_2^*}_{s-1}).
\end{aligned}$$

From $i + s'/2^i \ge \ln(2^i + s')$,finally,

$$\begin{aligned}
&\mathrm{sum}_{\gamma\lceil \log k\rceil}(W, \underbrace{d_2^*, \ldots, d_2^*}_{s-1}) \\
=&(1-\gamma)^i\left(1-\frac{s'\gamma}{2^i}\right)W \\
\le&\exp[-\gamma(i+s'/2^i)] \\
\le&\exp\left[-\gamma\ln(2^i+s')\right] = s^{-\gamma}.
\end{aligned}$$

□

Lemma 5.6. *For any weight $W \in [0,1]$,any integer $s \ge 2$, any sequence of distributions $d_{K_1}, \ldots, d_{K_u}, d_2^*, \ldots, d_2^*$ that is acceptable for s with length t, and the sequence of distributions with length $s-1$ $d_{K_1}, \ldots, d_{K_{u-1}}, d_2^*, \ldots, d_2^*$,*

$$\begin{aligned}
&\mathrm{sum}_{\gamma\lceil \log k\rceil}(W, \underbrace{d_{K_1}, \ldots, d_{K_u}, d_2^*, \ldots, d_2^*}_{t}) \\
\le&\mathrm{sum}_{\gamma\lceil \log k\rceil}(W, \underbrace{d_{K_1}, \ldots, d_{K_{u-1}}, d_2^*, \ldots, d_2^*}_{t})
\end{aligned}$$

Proof. We assume the sequence of length t $d_{K_1}, \ldots, d_{K_u}, d_2^*, \ldots, d_2^*$ is acceptable for s. Then,the sequence of length $t + K_u - 1$ $d_{K_1}, \ldots, d_{K_{u-1}}, d_2^*, \ldots, d_2^*$ is also acceptable for s.

Let T be the tree corresponds to $\mathrm{sum}_{\gamma\lceil \log k\rceil}(W, \underbrace{d_{K_1}, \ldots, d_{K_{u-1}}, d_2^*, \ldots, d_2^*}_{t})$.

Let C be the set of leaves of T after the $u-1$ th expansion. We denote W_ℓ as the weight of each leaf $\ell \in C$. Then,

$$\begin{aligned}
&\mathrm{sum}_{\gamma\lceil \log k\rceil}(W, \underbrace{d_{K_1}, \ldots, d_{K_{u-1}}, d_2^*, \ldots, d_2^*}_{t}) \\
=&\sum_{\ell\in C}\mathrm{sum}_{\gamma\lceil \log k\rceil}(W_\ell, \underbrace{d_2^*, \ldots, d_2^*}_{t_\ell}),
\end{aligned}$$

where t_ℓ is the number of expansions for leaf ℓ and its descendants leaves. Note that $t = \sum_{\ell\in C} t_\ell$. Let ℓ^* be the leaf in C that has the maximum weight. (So ℓ^* is the u th leaf to be expanded.) Let T_{ℓ^*} be the subtree of T rooted from ℓ^*. We replace T_{ℓ^*} with $\widehat{T}_{\ell^*}$ that has the following properties:

1. $|T_{\ell^*}| = |\widehat{T}_{\ell^*}|$
2. $\widehat{T}_{\ell^*}$ is generated by $d_{K_u}, \underbrace{d_2^*, \ldots, d_2^*}_{t_{\ell^*}-K_u+1}$, whereas T_{ℓ^*} is generated by $\underbrace{d_2^*, \ldots, d_2^*}_{t_{\ell^*}}$.

To replace T_{ℓ^*} with $\widehat{T}_{\ell^*}$, we guarantee that $K_u \le |T_{\ell^*}|$. By assumption, the sequence $d_{K_1}, \ldots, d_{K_u}, d_2^*, \ldots, d_2^*$ is acceptable for s ,thus $K_u \le s/|C|$. On the other hand, T_{ℓ^*} is the biggest subtree among all subtrees rooted from leaves in C. Thus $s/|C| \le |T_{\ell^*}|$. Thus $K_u \le |T_{\ell^*}|$.

From lemma 5.5,

$$\begin{aligned}
&\mathrm{sum}_{\gamma\lceil \log k\rceil}(W_\ell^*, \underbrace{d_2^*, \ldots, d_2^*}_{t_\ell}) \\
\ge&\mathrm{sum}_{\gamma\lceil \log k\rceil}(W_\ell^*, d_{K_u} \underbrace{d_2^*, \ldots, d_2^*}_{t_\ell-K_u+1}).
\end{aligned}$$

Thus we conclude that by replacing subtree T_{ℓ^*} with $\widehat{T}_{\ell^*}$, the sum of weights becomes small. We denote the replaced tree as $\widehat{T}$, and its sum of weights of leaves as $\widehat{\mathrm{sum}}_{\gamma\lceil \log k\rceil}$. Then,

$$\begin{aligned}
&\sum_{\ell\in C}\mathrm{sum}_{\gamma\lceil \log k\rceil}(W_\ell, \underbrace{d_2^*, \ldots, d_2^*}_{t_\ell}) \\
\ge&\widehat{\mathrm{sum}}_{\gamma\lceil \log k\rceil}(W, \underbrace{d_{K_1}, \ldots, d_{K_u}, d_2^*, \ldots, d_2^*}_{t}).
\end{aligned}$$

Finally,

$$\begin{aligned}&\widehat{\mathrm{sum}}_{\gamma\lceil\log k\rceil}(W, \underbrace{d_{K_1}, \ldots, d_{K_u}, d_2^*, \ldots, d_2^*}_{t})\\ \geq&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d_{K_1}, \ldots, d_{K_u}, d_2^*, \ldots, d_2^*}_{t}).\end{aligned}$$

□

Proof for Lemma 5.2

From lemma 5.5 and lemma 5.6, for any sequence of distribution $d_{K_1}, \ldots, d_{K_t}$ that is acceptable for s,

$$\begin{aligned}&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, d_{K_1}, \ldots, d_{K_t})\\ \leq&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, d_{K_1}, \ldots, d_{K_t-1}, \underbrace{d_2^*, \ldots, d_2^*}_{K_t-1})\\ \leq&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, d_{K_1}, \ldots, d_{K_t-2}, \underbrace{d_2^*, \ldots, d_2^*}_{K_{t-1}+K_t-1})\\ \leq&\ldots\\ \leq&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d_2^*, \ldots, d_2^*}_{s-1})\\ =&\mathrm{sum}_{\gamma\lceil\log k\rceil}(W, \underbrace{d_2^*, \ldots, d_2^*}_{s-1}) \leq s^{-\gamma}\end{aligned}$$

, where $s-1 = (K_1 - 1) + (K_2 - 1) + \cdots + (K_t - 1)$. □

6 Acknowledgement

I am grateful to Prof. Osamu Watanabe and Tadashi Yamazaki for important hints and comments. I also thank members of Watanabe laboratory for helpful discussions.

References

[1] A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth. Occam's razor. *Information Processing Letters*, 24:377–380, April 1987.

[2] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Wadsworth International Group, 1984.

[3] Yoav Freund and Robert E. Schapire:. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[4] M. Kearns and Y. Mansour. On the boosting ability of top-down decision tree learning algorithms. *Journal of Computer and System Sciences*, 58(1):109–128, 1999.

[5] J. Ross Quinlan. *C4.5: Programs for Machine Learning.* Morgan Kaufmann, 1993.

[6] Eiji Takimoto and Akira Maruoka. On the boosting algorithm for multiclass functions based on information-theoretic criterion for approximation. In *Proceedings of the 1st International Conference on Discovery Science*, volume 1532 of *Lecture Notes in Artificial Intelligence*, pages 256–267. Springer-Verlag, 1998.

[7] L. G. Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, November 1984.