TTIC 31290: Machine Learning for Algorithm Design (Fall 2025)
Avrim Blum and Dravyansh Sharma

Lecture 9: 10/28/25 Lecturer: Avrim Blum (Notes by Dravyansh Sharma)

Outline for today

- Piecewise Decomposable Function Classes and Generalization
- Examples

1 Piecewise Decomposable Function Classes and Generalization

Let $\mathcal{U} = \{ u_\rho : \mathcal{X} \to \mathbb{R} \mid \rho \in \mathcal{P}\}$ be a family of functions from $\mathcal{X}$ to $\mathbb{R}$. For each instance $x \in \mathcal{X}$ the *dual function* is defined as

$$u^*_x(\rho) := u_\rho(x),$$

and let

$$\mathcal{U}^* = \{ u^*_x : \mathcal{P} \to \mathbb{R} \mid x \in \mathcal{X} \}.$$

denote the dual function class.

We will typically analyze the structure of the dual utility functions. Often, we will show that the dual utility functions have a piecewise–decomposable structure defined below.

Definition 1 (Piecewise–Decomposable Function Class)**.** *A function class $\mathcal{H} \subseteq \mathbb{R}^{\mathcal{P}}$ that maps a domain $\mathcal{P}$ to $\mathbb{R}$ is* $(\mathcal{F}, \mathcal{G}, k)$-piecewise decomposable *for a class $\mathcal{G} \subseteq \{0,1\}^{\mathcal{P}}$ of boundary functions and a class $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{P}}$ of piece functions if the following holds: for every $h \in \mathcal{H}$, there are k boundary functions $g_1, \ldots, g_k \in \mathcal{G}$ and a piece function $f_{\mathbf{b}} \in \mathcal{F}$ for each bit vector $\mathbf{b} \in \{0,1\}^k$ such that for all $\rho \in \mathcal{P}$, $h(\rho) = f_{\mathbf{b}_\rho}(\rho)$ where $\mathbf{b}_\rho = (g_1(\rho), \ldots, g_k(\rho)) \in \{0,1\}^k$.*

For example, say the domain $\mathcal{P} = \mathbb{R}^d$. We can express functions that are piecewise constant with at most L linear boundaries as $(\mathcal{F}, \mathcal{G}, k)$-*piecewise decomposable* by setting $\mathcal{F}$ to be the set of constant functions, $\mathcal{G}$ to be the set of linear thresholds and setting $k = L$.

Theorem 1 (Pseudodimension Bound for Piecewise–Decomposable Dual Classes)**.** *Let $\mathcal{U}$ be as above and suppose that its dual class $\mathcal{U}^*$ is $(\mathcal{F}, \mathcal{G}, k)$–piecewise decomposable. Let $d_F = \text{Pdim}(\mathcal{F}^*)$ and $d_G = \text{VCdim}(\mathcal{G}^*)$. Then*

$$\text{Pdim}(\mathcal{U}) = O(d_F + d_G \log k)\,.$$

Proof. Let $x_1, \ldots, x_m \in \mathcal{X}$ be m arbitrary problem instances and $r_1, \ldots, r_m \in \mathbb{R}$ be m arbitrary thresholds. We seek to bound the number of sign patterns

$$|\{\big(\text{sign}(u_\rho(x_1) - r_1), \ldots, \text{sign}(u_\rho(x_m) - r_m)\big) : \rho \in \mathcal{P}\}|,$$

or equivalently,

$$|\{\big(\text{sign}(u^*_{x_1}(\rho) - r_1), \ldots, \text{sign}(u^*_{x_m}(\rho) - r_m)\big) : \rho \in \mathcal{P}\}|.$$

We first prove the following.

Lemma 1. *Given $x_1, ..., x_m$, there is a finite partition of $\mathcal{P}$ into at most $M < (ekm/d_G)^{d_G}$ subsets $P_1, \ldots, P_M$ such that for each subset P_j, there exist piece functions $f_1^{(j)}, \ldots, f_m^{(j)} \in \mathcal{F}$ such that $u^*_{x_i}(\rho) = f_i^{(j)}(\rho)$ for all parameters $\rho \in P_j$ and $i \in [m]$.*

Proof of Lemma 1. Consider all boundary functions $\{g_1, \ldots, g_T\}$ of the dual functions $u^*_{x_i}$ across the instances x_i for $i \in [m]$. Since each $u^*_{x_i}$ is $(\mathcal{F}, \mathcal{G}, k)$-piecewise decomposable, we have $T \leq km$. Consider the set of binary patterns

$$\mathcal{M} = \{(g_1(\rho), \ldots, g_T(\rho)) : \rho \in \mathcal{P}\}.$$

We want to bound

$$\begin{aligned} |\mathcal{M}| &\leq \max_{\{g'_1,...,g'_T\} \subseteq \mathcal{G}} |\{(g'_1(\rho), ..., g'_T(\rho) : \rho \in \mathcal{P}\}| \\ &= \max_{\{g'_1,...,g'_T\} \subseteq \mathcal{G}} |\{(\rho(g'_1), ..., \rho(g'_T) : \rho \in \mathcal{P}\}| \\ &= \Gamma_{\mathcal{G}^*}(T), \end{aligned}$$

where $\mathcal{G}^*$ is the dual function class of $\mathcal{G}$ and we define $\rho(g) = g(\rho)$.

By Sauer's Lemma applied to $\mathcal{G}^*$, we have $|\mathcal{M}| < (eT/d_G)^{d_G} \leq (ekm/d_G)^{d_G}$ which implies the Lemma by defining P_i to be the set of parameter values corresponding to each sign pattern in $\mathcal{M}$ and recalling that the piece function for each instance x_i are fixed once the sign pattern with respect to all its boundary functions is fixed (Definition 1). □

Now for any fixed $P_j \subseteq \mathcal{P}$, by definition of the partition we have:

$$\left| \left\{ \begin{pmatrix} \text{sign}(u^*_{x_1}(\rho) - r_1) \\ \vdots \\ \text{sign}(u^*_{x_m}(\rho) - r_m) \end{pmatrix} : \rho \in P_j \right\} \right| = \left| \left\{ \begin{pmatrix} \text{sign}(f_1^{(j)}(\rho) - r_1) \\ \vdots \\ \text{sign}(f_m^{(j)}(\rho) - r_m) \end{pmatrix} : \rho \in P_j \right\} \right|.$$

Now applying Sauer's Lemma to the dual function class $\mathcal{F}^*$ of the piece function class $\mathcal{F}$ gives that for each P_j, the above set has size at most $(em/d_F)^{d_F}$. Thus, across all of $\mathcal{P}$, the number of sign patterns is at most $(ekm/d_G)^{d_G}(em/d_F)^{d_F}$. For $x_1, \ldots, x_m$ to be shattered, we have

$$2^m \leq (ekm/d_G)^{d_G}(em/d_F)^{d_F}.$$

To simplify this, we recall the fact $\ln x \leq \alpha x - \ln \alpha - 1$ for all $\alpha, x > 0$ from a previous lecture.

$$
\begin{aligned}
m &\le d_G \log(ekm/d_G) + d_F \log(em/d_F) \\
&\le 2\left[d_G \ln(ekm/d_G) + d_F \ln(em/d_F)\right] \\
&= 2\left[d_G(1+\ln k) + d_F + d_G\left(\ln(m) - \ln(d_G)\right) + d_F\left(\ln(m) - \ln(d_F)\right)\right] \\
&\le 2\left[d_G(1+\ln k) + d_F + d_G\left(\frac{m}{8d_G} - \ln\frac{1}{8d_G} - 1 - \ln(d_G)\right) + d_F\left(\ln(m) - \ln(d_F)\right)\right] \\
&= 2\left[d_G(1+\ln k) + d_F + \frac{m}{8} + d_G \ln\frac{8}{e} + d_F\left(\ln(m) - \ln(d_F)\right)\right] \\
&\le 2\left[d_G(1+\ln k) + d_F + \frac{m}{8} + d_G \ln\frac{8}{e} + \frac{m}{8} + d_F \ln\frac{8}{e}\right] \\
&\le \frac{m}{2} + 2[d_G(3+\ln k) + 3d_F].
\end{aligned}
$$

This implies the claimed pseudo-dimension bound. □

2 Examples

We will now illustrate some applications of the above result to algorithm design for some classical problems.

2.1 Maximum weighted independent set

Let $G = (V, E)$ be an undirected graph with $|V| = n$ vertices and nonnegative vertex weights $w : V \to \mathbb{R}_{\ge 0}$. The *Maximum Weight Independent Set (MWIS)* problem is:

$$\max_{S \subseteq V \text{ independent}} w(S) := \sum_{v \in S} w(v).$$

This problem is NP-hard. A common greedy algorithm constructs an independent set incrementally by using a scoring rule that depends on the weight and degree of the vertices.

1. Initialize $S \leftarrow \emptyset$.
2. If there are no vertices in G, return S.
3. Let v' be the vertex with the largest score $\frac{w(v)}{(d(v)+1)^\alpha}$, where $d(v)$ is the degree of v in the current graph.
4. $S \leftarrow S \cup \{v'\}$.
5. Remove v' and all its neighbors. Go back to Step 2.

A natural choice for the utility function is the total weight of the independent set of vertices returned by the algorithm, $u_\alpha(G) = \sum_{v \in S_\alpha} w(v)$, where S_α is the set of vertices returned by the above greedy algorithm with parameter α.

Theorem 2. *Let* $\mathcal{U} = \{u_\alpha : \alpha \geq 0\}$. *Then* $\text{Pdim}(\mathcal{U}) = O(\log n)$.

Proof Sketch. We first show that for a fixed weighted graph G, the dual utility function $u^*_G(\alpha)$ is piecewise constant with $O(n^4)$ pieces. The argument is similar to that for the greedy algorithm for knapsack from the last lecture, except that now the degree of the vertex $d(v)$ may change when we update the graph in Step 5. A critical point satisfies $w(v_1)/(1+d_1)^\alpha = w(v_2)/(1+d_2)^\alpha$ for some vertices $v_1, v_2 \in V$ and for some $d_1, d_2 \in [n]$. This implies the claimed $O(n^4)$ bound.

To bound the pseudo-dimension of $\mathcal{U}$, we can use Theorem 1. The class of piece functions consists of constant functions ($d_F = 1$), the class of boundary functions consists of thresholds in one-dimension ($d_G = 1$), and $k = O(n^4)$. Using Theorem 1, $\text{Pdim}(\mathcal{U}) = O(\log n)$. □

2.2 Dynamic Programming for Sequence Alignment

Sequence alignment is a fundamental combinatorial problem. It has applications to computational biology. For example, to compare two DNA or RNA sequences the standard approach is to align the two sequences in order to detect similar regions. However, the optimal alignment depends on the relative costs or weights used for specific substitutions, insertions/deletions, etc. in the sequences. Given a set of weights, the optimal alignment computation is typically a simple dynamic program. Depending on the application, different weights can lead to the "best alignments". Our goal is to learn the weights, such that the alignment produced by the dynamic program has application-specific desirable properties.

Formally, given a pair of sequences $s_1 \in \Sigma^m, s_2 \in \Sigma^n$ over some alphabet Σ of lengths $m = |s_1|$ and $n = |s_2|$ (WLOG $m \leq n$), and a 'space' character $- \notin \Sigma$, a space-extension t of a sequence s over Σ is a sequence over $\Sigma \cup \{-\}$ such that removing all occurrences of $-$ in t gives s. A global alignment (or simply alignment) of s_1, s_2 is a pair of sequences t_1, t_2 such that $|t_1| = |t_2|$, t_1, t_2 are space-extensions of s_1, s_2 respectively, and for no $1 \leq i \leq |t_1|$ we have $t_1[i] = t_2[i] = -$.

Let $s[i]$ denote the i-th character of a sequence s and $s[:i]$ denote the first i characters of sequence s. Suppose we have two features, *mismatches* (or substitutions) with cost ρ_1 and *spaces* (or insertions/deletions) with cost ρ_2. The alignment that minimizes the cost for strings s_1, s_2 with $|s_1| = m, |s_2| = n$ may be obtained using a dynamic program in $O(mn)$ time. The dynamic program is given by the following recurrence relation for the cost function which holds for any $i, j > 0$, and for any cost parameters $\rho = (\rho_1, \rho_2)$,

$$C(s_1[:i], s_2[:j], \rho) = \begin{cases} C(s_1[:i-1], s_2[:j-1], \rho) & \text{if } s_1[i] = s_2[j], \\ \min\big\{\rho_1 + C(s_1[:i-1], s_2[:j-1], \rho), & \\ \qquad \rho_2 + C(s_1[:i-1], s_2[:j], \rho), & \text{if } s_1[i] \neq s_2[j]. \\ \qquad \rho_2 + C(s_1[:i], s_2[:j-1], \rho) \quad \big\} & \end{cases}$$

The base cases are $C(\phi, \phi, \rho) = 0, C(\phi, s_2[:j], \rho) = j\rho_2, = C(s_1[:i], \phi, \rho) = i\rho_2$ for $i, j \in [m] \times [n]$. Here ϕ denotes the empty sequence. We can write down a similar recurrence for computing the optimal alignment as well.

More generally, suppose there are d alignment features given by $l_1(s_1, s_2, t_1, t_2), \ldots, l_d(s_1, s_2, t_1, t_2)$, where each $l_i(\cdot)$ gives some notion of"badness" for aligning strings s_1, s_2 as t_1, t_2. The overall cost

may be given by a linear combination

$$c(s_1, s_2, t_1, t_2, \rho) = \sum_{k=1}^{d} \rho_k l_k(s_1, s_2, t_1, t_2),$$

where $\rho = (\rho_1, \ldots, \rho_d)$ are the parameters that govern the relative weight of the features. Assume we have an algorithm A_ρ that minimizes this objective (which might be a dynamic programming algorithm if the individual cost functions l_k are sums of local quantities).

Let utility function $u(s_1, s_2, t_1, t_2)$ denote the "ground truth" quality of the alignment t_1, t_2 for strings s_1, s_2. Define $u_\rho(s_1, s_2)$ as $u(s_1, s_2, t_1^{(\rho)}, t_2^{(\rho)})$, where $t_1^{(\rho)}, t_2^{(\rho)}$ is the alignment obtained using algorithm A_ρ.

Theorem 3. *Let $\mathcal{U}$ denote the utility function class $\{u_\rho \mid \rho \in \mathbb{R}^d\}$ that measures the quality of the alignment produced using different algorithm parameters ρ. We have* $\mathrm{Pdim}(\mathcal{U}) = O(dm \log(m+n))$.

Proof. We will show that the dual functions are $(\mathcal{F}, \mathcal{G}, m^2(m+n)^{2m})$-*piecewise decomposable*, where $\mathcal{F}$ consists of constant functions $\mathbb{R}^d \to \mathbb{R}$ and $\mathcal{G}$ consists of linear thresholds in $\mathbb{R}^d$. The above bound then follows using Theorem 1. We first establish the following useful lemma.

Lemma 2. *For a fixed pair of sequences $s_1, s_2 \in \Sigma^m \times \Sigma^n$, with $m \leq n$, there are at most $m(m+n)^m$ distinct alignments.*

Proof. For any alignment (t_1, t_2), by definition, we have $|t_1| = |t_2|$ and for all $i \in [|t_1|]$, if $t_1[i] = -$, then $t_2[i] \neq -$ and vice versa. This implies that t_1 has exactly $n - m$ more spaces than t_2. To prove the upper bound, we count the number of alignments (t_1, t_2) where t_2 has exactly i spaces for $i \in [m]$. There are $\binom{n+i}{i}$ choices for placing the space in t_2. Given a fixed t_2 with i spaces, there are $\binom{n}{n-m+i}$ choices for placing the space in t_1. Thus, there are at most $\binom{n+i}{i}\binom{n}{n-m+i} = \frac{(n+i)!}{i!(m-i)!(n-m+i)!} \leq (m+n)^m$ possibilities since $i \leq m$. Summing over all i, we have at most $m(m+n)^m$ alignments of s_1, s_2. □

Fix a pair of sequences s_1 and s_2. Let τ be the set of optimal alignments as we range over all parameter vectors $\rho \in \mathbb{R}^d$. By Lemma 2, we have $|\tau| \leq m(m+n)^m$. For any alignment $(t_1, t_2) \in \tau$, the algorithm A_ρ will return (t_1, t_2) only if

$$\sum_{i=1}^{d} \rho_i l_i(s_1, s_2, t_1, t_2) \geq \sum_{i=1}^{d} \rho_i l_i(s_1, s_2, t_1', t_2')$$

for all $(t_1', t_2') \in \tau \setminus \{(t_1, t_2)\}$. Therefore, there is a set H of at most $\binom{|\tau|}{2} \leq m^2(m+n)^{2m}$ hyperplanes such that across all parameter vectors ρ in a single connected component of $\mathbb{R}^d \setminus H$, the output of the algorithm A_ρ on (s_1, s_2) is fixed. This means that for any connected component R of $\mathbb{R}^d \setminus H$, there exists a real value c_R such that $u_\rho(s_1, s_2) = c_R$ for all $\rho \in \mathbb{R}^d$. By the definition of dual, $u^*_{s_1,s_2}(u_\rho) = u_\rho(s_1, s_2) = c_R$. For each hyperplane $h \in H$, let $g^{(h)} \in \mathcal{G}$ denote the corresponding halfspace. Order these $k \leq \binom{|\tau|}{2}$ functions arbitrarily as $g_1, \ldots, g_k$. For a given connected component R of $\mathbb{R}^d \setminus H$, let $\mathbf{b}_R \in \{0,1\}^k$ be the corresponding sign pattern with respect to $g_1, \ldots, g_k$. Define the piece function $f^{(\mathbf{b}_R)} = f_{c_R}$, the constant function $f_{c_R}(x) = c_R$ for all $x \in \mathbb{R}^d$. This gives us the claimed piecewise decomposition. To complete the proof, note that $d_F = 1$ and $d_G = d + 1$. □

Additional Resources:

- Maria-Florina Balcan, Dan Deblasio, Travis Dick, Carl Kingsford, Tuomas Sandholm, and Ellen Vitercik. "How much data is sufficient to learn high-performing algorithms? Generalization guarantees for data-driven algorithm design." In Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing, pp. 919-932. 2021.
- Maria-Florina Balcan, Dan Deblasio, Travis Dick, Carl Kingsford, Tuomas Sandholm, and Ellen Vitercik. "How much data is sufficient to learn high-performing algorithms?" Journal of the ACM 71, no. 5 (2024): 1-58.
- Maria-Florina Balcan, Christopher Seiler, and Dravyansh Sharma. "Accelerating ERM for data-driven algorithm design using output-sensitive techniques." Advances in Neural Information Processing Systems 37 (2024): 72648-72687.