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

Lecture 8: 10/23/25 Lecturer: Dravyansh Sharma

## Outline for today

- Sample complexity for infinite concept classes (agnostic case; real-valued case)
- Primal and dual function classes
- Piecewise-constant case

# 1 Sample complexity for infinite concept classes

Last time we showed a bound on the sample complexity of realizable PAC learning. We will now look at extensions of the result to agnostic PAC learning and beyond the binary case to real-valued functions.

## 1.1 Agnostic PAC Learning

We have the following sample complexity bound for agnostic PAC learning.

**Theorem 1.** *Let $C$ be an arbitrary concept space with VC dimension $d$. Then $C$ is agnostic PAC learnable with sample complexity*

$$m_C(\epsilon, \delta) = O\left(\frac{1}{\epsilon^2}\left(d \ln\left(\frac{1}{\epsilon}\right) + \ln\frac{1}{\delta}\right)\right).$$

*For any $D$, and any $\epsilon, \delta \in (0,1)$, given an i.i.d. sample $S$ of size at least $m_C(\epsilon, \delta)$ above, with probability at least $1-\delta$ over the draw of $S$, all hypotheses $h \in C$ satisfy $|\mathrm{err}_S(h) - \mathrm{err}_D(h)| \le \epsilon$.*

A proof can be given using the same overall strategy as in the realizable case with a few modifications. We will describe below the main modifications.

*Proof Sketch.* We set the bad event $B_1$ as: there exists $h \in C$ with $|\mathrm{err}_S(h) - \mathrm{err}_D(h)| > \epsilon$. Similarly we modify $B_2$ to be the event that for some $h$, the sample error on $S$ and the "ghost sample" $S'$ differs by at least $\epsilon/2$.

As before, using Chernoff bounds, $\Pr[B_1] \le 2\Pr[B_2]$. To bound $\Pr[B_2]$, we again consider the double sample $U$ of size $2m$ to construct $S, S'$ and apply Hoeffding bounds to show that

$$\Pr[|\mathrm{err}_S(h) - \mathrm{err}_{S'}(h)| > \epsilon/2] \le e^{-\epsilon^2 m/8},$$

and apply a union bound over $\Gamma_C(2m)$ hypotheses $h$ as before. □

A sharper analysis gets rid of the $\log \frac{1}{\epsilon}$ term (Anthony and Bartlett, 2001). It can be further shown that the $O\left(\frac{1}{\epsilon^2}\left(d + \ln \frac{1}{\delta}\right)\right)$ bound on the sample complexity of agnostic PAC learning is optimal up to constants.

### 1.2 Extension to real-valued functions

We have the following analogous result for real-valued functions (Anthony and Bartlett, 1999).

**Theorem 2.** *Let $C$ denote a class of functions with domain $X$ and range $[0, U]$, with pseudo-dimension* $\text{Pdim}(C)$*. For every distribution $D$ over $X$, every $\epsilon > 0$, and every $\delta \in (0, 1]$, if*

$$m \geq c\frac{U^2}{\epsilon^2}\left(\text{Pdim}(C) + \ln \frac{1}{\delta}\right),$$

*for some absolute constant c, then with probability at least $1 - \delta$ over $S \sim D^m$,*

$$\left|\frac{1}{m}\sum_{x_i \in S} h(x_i) - \mathbb{E}_{x \sim D}[h(x)]\right| < \epsilon,$$

*for every $h \in C$.*

## 2 Primal and dual function classes

Recall our formulation for algorithm design as PAC Learning.

We have a set of problem instances of interest $\Pi$ and a (potentially infinite) set $\mathcal{A}$ of algorithms. We will typically have a parameterized family of algorithms given by a set of parameters $\mathcal{P} \subseteq \mathbb{R}^d$. That is, each $\rho \in \mathcal{P}$ corresponds to an algorithm $A_\rho \in \mathcal{A}$. We also fix a utility function $u : \Pi \times \mathcal{P} \to [0, U]$, where $u(x, \rho)$ measures the performance of the algorithm with parameter setting $\rho$ on problem instance $x \in \Pi$. For example, $u$ could denote the algorithm's running time and $H$ could be the time-out deadline.

To apply the above theorem, we consider the class of functions $\mathcal{U} = \{u_\rho : \Pi \to [0, U] \mid \rho \in \mathcal{P}\}$, where $u_\rho(x) = u(x, \rho)$ for any $x, \rho$. We will call this the "primal" class of functions. We have the sample complexity of uniform convergence for $\mathcal{U}$ is $O\left(\frac{U^2}{\epsilon^2}(\text{Pdim}(\mathcal{U}) + \ln \frac{1}{\delta})\right)$.

However, characterizing the behavior of functions $u_\rho$ (e.g. all behaviors of a clustering algorithm as the instances are varied) is challenging. A useful analytical tool will be to consider the "dual" functions $\mathcal{U}^* = \{u_x^* : \mathcal{P} \to [0, U] \mid x \in \Pi\}$, where $u_x^*(\rho) = u_\rho(x) = u(x, \rho)$ for any $x, \rho$. The advantage is that it will often be simpler to analyze the functions $u_x^*$, which give the variation of the algorithm performance as the parameter is varied for a *fixed* problem instance $x$.

## 3 Piecewise-constant case

A simple but widely occurring case is where $\rho$ is a single real parameter. We have the following useful lemma (Balcan, 2020).

**Lemma 1.** *Suppose that for every instance $x \in \Pi$, the function $u_x^*(\rho) : \mathbb{R} \to \mathbb{R}$ is piecewise constant with at most $N$ pieces. Then the family $\mathcal{U} = \{u_\rho(x)\}$ has pseudo-dimension $O(\log N)$.*

*Proof.* Consider a fixed problem instance $x \in \Pi$. Since the function $u_x^*(\rho)$ is piecewise constant with at most $N$ pieces, this means there are at most $N-1$ critical points such that between any two consecutive critical points, the function $u_x^*(\rho)$ is constant.

Consider $m$ problem instances $x_1, \ldots, x_m \in \Pi$. Taking the union of their critical points, between any two consecutive of these critical points we have that all of the functions $u_{x_i}^*(\rho)$ are constant. These critical points break up the real line into at most $(N-1)m + 1 \leq Nm$ intervals, and all $u_{x_i}^*(\rho)$ are constant in each interval. Thus, overall there are at most $Nm$ different $m$-tuples of values produced over all $\rho$. Equivalently, the functions $u_\rho(x)$ produce at most $Nm$ different $m$-tuples of function values on the m inputs $x_1, \ldots, x_m$. However, in order to shatter the $m$ instances, we must have $2^m$ different $m$-tuples of values to get all the $2^m$ distinct above-below patterns. Solving $Nm \geq 2^m$ shows that only sets of instances of size $m = O(\log N)$ can be shattered. □

We will now show an example application of the above lemma.

### 3.1 Greedy algorithms for Knapsack

As an example, a canonical problem we consider in this chapter is the *knapsack problem.* A knapsack instance $x$ consists of $n$ items given by values $v_1, \ldots, v_n \in \mathbb{R}_{\geq 0}$ and sizes $s_1, \ldots, s_n \in \mathbb{R}_+$, and an overall knapsack capacity $C \in \mathbb{R}_+$. The goal is to find the most valuable subset of items for which the total size is at most $C$.

We analyze a family of greedy algorithms parametrized by a one dimensional set of parameters, $\mathcal{P} = \mathbb{R}$. For $\rho \in \mathcal{P}$, the algorithm $A_\rho$ is the following greedy procedure: Set the score of item $i$ to $v_i/s_i^\rho$; then, in decreasing order of score, add each item to the knapsack if there is enough capacity left (breaking ties by selecting the item of smallest index).

The utility function $u_\rho(x) = u(x, \rho)$ is defined as the total value of the items chosen by the greedy algorithm $A_\rho$ with parameter $\rho$ on input $x$.

**Theorem 3.** *The family of utility functions $\mathcal{U}_{Knapsack} = u_\rho(x)$ defined above has pseudo-dimension $O(\log n)$, where $n$ is the maximum number of items in an instance.*

*Proof.* We will show that each dual function $u_x^*(\rho)$ is piecewise constant with at most $\binom{n}{2} + 1$ pieces. Then the above lemma gives the pseudo-dimension bound.

Fix a knapsack instance $x$. If two algorithms $A_\rho$ and $A_{\rho'}$ differ ($\rho < \rho'$), then we must have some first point where $A_\rho$ picks some item $i$ and $A_{\rho'}$ picks $i' \neq i$. This implies $v_i/s_i^\rho - v_{i'}/s_{i'}^\rho \geq 0$ but $v_i/s_i^{\rho'} - v_{i'}/s_{i'}^{\rho'} \leq 0$. Since $f(y) = v_i/s_i^y - v_{i'}/s_{i'}^y$ is a continuous function of $y$, $v_i/s_i^y - v_{i'}/s_{i'}^y = 0$ for some $y$ in $[\rho, \rho']$. Thus, $u_x^*(\rho)$ must be a constant function over any $[\rho, \rho']$ if there is no point with $v_i/s_i^y - v_{i'}/s_{i'}^y = 0$ for some pair of items $i, i'$. That is, it is piecewise constant with number of critical points at most the number of distinct choices for $i, i'$. □

If we scale (divide) the utility function above by the optimal value for the instance $x$, we have $u(x, \rho) \leq 1$ for all $x, \rho$. Now, $O\left(\frac{\log n/\delta}{\epsilon^2}\right)$ problem instances are sufficient to learn a near-optimal value of the greedy algorithm parameter $\rho$.

**Additional Resources:**


- Martin Anthony and Peter Bartlett, *Neural Network Learning: Theoretical Foundations*, Cambridge University Press, 1999.
- Maria-Florina Balcan, "Data-Driven Algorithm Design" (book chapter). In *Beyond Worst Case Analysis of Algorithms*, Tim Roughgarden (Ed). Cambridge University Press, 2020.