ML Theory Lecture 5

Matus Telgarsky

1 Polynomial fit

We'll use a tool rooted in polynomial approximation to show that RBF SVM and 2 layer networks can fit continuous functions.

Remark 1.1. As discussed in lecture, this tool is the standard (and shortest) way to prove such results, but it leaves no intuition about the quality of approximation, how one can construct approximations, etc. ◇

Theorem 1.2 (Stone-Weierstrass; cf. (Folland, 1999, Theorem 4.45))**.** *Let a class of functions $\mathcal{F}$ be given, satisfying the following properties.*

(1.2.a) *Each $f \in \mathcal{F}$ is continuous.*

(1.2.b) *For every x, there exists $f \in \mathcal{F}$ with $f(x) \neq 0$.*

(1.2.c) *$\mathcal{F}$* separates points*, meaning for every x, y there exists $f \in \mathcal{F}$ with $f(x) \neq f(y)$. (Note that if this property failed, then we could pick some continuous function g which differs on x and y, and $\mathcal{F}$ couldn't hope to approximate it well.)*

(1.2.d) *$\mathcal{F}$ is an* algebra*, meaning it is closed under multiplication and under vector space operations.*

Then for every continuous $g : \mathbb{R}^d \to \mathbb{R}$ and $\epsilon > 0$, there exists $f \in \mathcal{F}$ with $\|f - g\|_{\mathrm{u}} \leq \epsilon$.

Remark 1.3 (Sanity checks.)**.** Note that the conditions of the theorem are satisfied for polynomials:

$$\operatorname{span}(\operatorname{prod}(\{x \mapsto x_i : i \in \{1, \ldots d\}\})).$$

Note moreover that the conditions are *not* satisfied for polynomials of *fixed* degree; the above display has all polynomials of all degrees. This is consistent with the Minsky-Papert construction from earlier lectures, which provided limitations of linear classifiers (degree 1 polynomials).

This theorem says that classes of functions behaving somewhat similarly to polynomials also have polynomial approximation propeties. ◇

Remark 1.4 (Notation.)**.** It is tiresome to always state "for every continuous function f and $\epsilon > 0 \ldots$". Another way to state the theorems is: the closure of the function class under $\|\cdot\|_t u$ is equal to all continuous functions; this turns out to be the right topology. For more information, see any standard text, for instance (Folland, 1999). ◇

Remark 1.5 (Proofs.)**.** We won't prove this theorem, but here are some comments on how to do so.

- The original Weierstrass theorem only dealt with polynomial approximation; it was extended by Stone to the more general form above.

  The proof of the Weirstrass theorem which is popular in computer science circles is due to Bernstein: some n and consider the *Bernstein polynomial*

$$x \mapsto \sum_{i_1=0}^{n} \cdots \sum_{i_d=0}^{n} g\left(\sum_{j=1}^{d} \frac{i_j \mathbf{e}_j}{n}\right) \prod_{j=1}^{d} \Pr\left[\operatorname{Bin}(n, x_j) = i_j\right],$$

which is a polynomial since $\Pr\left[\mathrm{Bin}(n, x_j) = i_j\right] = \binom{n}{i_j} x_j^{i_j}(1-x_j)^{n-i_j}$. The proof then uses Chebyshev's inequality (or the law of large numbers) to control the polynomials between the n^d interpolation points.

- Weierstrass's original proof considers the *mollification* $x \mapsto \mathbb{E}(f(x+\xi))$, where $\xi \sim \mathcal{N}(0, \sigma^2)$. This mollified mapping approaches f as $\sigma \downarrow 0$, and moreover it is analytic so it has a well-behaved Taylor expansion (a polynomial).

◇

Remark 1.6 (Why this set of conditions?).

- It makes sense that the third property (separation) is necessary: if there is a pair (x, x') which can not be distinguished by $\mathcal{F}$ (there does not exist $f \in \mathcal{F}$ with $f(x) \neq f(x')$), then there is no hope of approximation of even a simple continuous function with different behavior on x and x' (e.g., $z \mapsto \langle z - x, x' - x\rangle$, which is 0 at x and $\|x - x'\|^2$ at x').
- To see how multiplication is useful, recall that for 3 layer networks, we first approximated the function $x \mapsto \mathbb{1}[x \in \times_{i=1}^d [a_i, b_i]]$. Since $\mathbb{1}[x \in \times_{i=1}^d [a_i, b_i]] = \prod_{i=1}^d \mathbb{1}[x_i \in [a_i, b_i]]$, instead of using an extra layer as last time, we could simply form this product.

◇

1.1 A convenient lemma

Define a class of single neural network nodes:

$$\mathcal{H}_\sigma := \left\{x \mapsto \sigma(a^\top x + b) \ :\ a \in R^d, b \in \mathbb{R}\right\}.$$

Lemma 1.7. *For every continuous $f : \mathbb{R}^d \to \mathbb{R}$ and $\epsilon > 0$, there exists $g \in \mathcal{H}_{\exp}$ with $\|f - g\|_{\mathrm{u}} \leq \epsilon$.*

[I got the idea for this proof from (Steinwart and Christmann, 2008, Chapter 4).]

Proof. (1.2.a) is direct, and (1.2.b) holds by picking $x \mapsto \exp(\langle 0, x\rangle)$ (indeed the elements of $\mathcal{H}_{\exp}$ are strictly positive, we can pack any one).

Now consider (1.2.c). We want to show $\mathcal{H}_{\exp}$ can separate points; intuitively this is easy, we just put an exp along the line between a given $x \neq y$. That is, define $h(z) := \exp(\langle x - y, z\rangle)$, and note

$$\frac{h(y)}{h(x)} = \frac{\exp(\langle x, y\rangle)\exp(-\langle y, y\rangle)}{\exp(\langle x, x\rangle)\exp(-\langle x, y\rangle)} = \exp(-\|x - y\|^2),$$

which is not 1 since $\|x - y\| \neq 0$, and thus $h(y) \neq h(x)$.

For (1.2.d), vector space operations are satisfied for free since we are working with $\mathrm{span}(\mathcal{H}_{\exp})$ and not $\mathcal{H}_{\exp}$. For multiplication, note

$$\left(\sum_{i=1}^N \alpha_i \exp(\langle x_i, z\rangle)\right)\left(\sum_{j=1}^M \beta_j \exp(\langle y_i, z\rangle)\right) = \sum_{i=1}^N \sum_{j=1}^M \alpha_i \beta_j \exp(\langle x_i + y_i, z\rangle) \in \mathrm{span}(\mathcal{H}_{\exp}).$$

□

For the rest of the lecture, we'll use Lemma 1.7 and not Theorem 1.2.

Remark 1.8. When invoking Lemma 1.7, we'll leave out the offset term "$+b$" within each neural network node; this is because we can pull $\exp(b)$ into the linear combination coefficient by properties of exp. This is not the case for general activations. . . ◇

1.2 2 layer networks

Theorem 1.9 ((Hornik et al., 1989) and others)**.** *Let any continuous nondecreasing $\sigma : \mathbb{R} \to \mathbb{R}$ be given with*

$$\lim_{z\to-\infty} \sigma(z) = 0, \qquad \lim_{z\to+\infty} \sigma(z) = 1.$$

For any continuous g and $\epsilon > 0$, there exists $f \in \operatorname{span}(\mathcal{H}_\sigma)$ with $\|f - g\|_u \le \epsilon$.

Proof sketch. There are many ways to prove it. One, based on (Hornik et al., 1989), is as follows.

1. Pick some univariate ϕ so that $\operatorname{span}(\mathcal{H}_\phi)$ does what we want (e.g., by Lemma 1.7, we could use $\phi = \exp$; the proof by Hornik et al. (1989) used $\phi = \cos$).
2. Now argue that ϕ can be approximated by $\mathcal{H}_\sigma$, and that this completes the proof.

The homework will fill in the gaps of this proof scheme. □

Remark 1.10. Now that we've seen both the 2- and 3-layer proofs, note that the 2-layer proof gives no sense of how to construct an approximating networks, whereas the 3-layer proof does. . . ◇

Remark 1.11. The proof doesn't apply to the ReLU as stated, but there are a few ways to extend it to handle the ReLU (e.g., this is done in the homework). ◇

1.3 RBF SVM

Lastly, consider the functions constructed by RBF SVMs (radial basis function support vector machines). First define (for $\gamma > 0$)

$$R_\gamma := \left\{ x \mapsto \exp(-\gamma\|x - \beta\| \ : \ \beta \in \mathbb{R}^d \right\};$$

RBF SVMs work with elements of $\operatorname{span}(R_\gamma)$.

Theorem 1.12. *Given any continuous f and $\epsilon > 0$ and $\gamma > 0$, there exists $g \in \operatorname{span}(R_\gamma)$ with $\|f - g\|_u \le \epsilon$.*

Proof. [Proof again using ideas from (Steinwart and Christmann, 2008, Chapter 4).]

By Lemma 1.7, there exists

$$g = \sum_i \alpha_i \exp(\langle \cdot, \beta_i \rangle) \qquad \text{with} \qquad \|g - f e^{\gamma\|\cdot\|^2}\|_u \le \epsilon.$$

Define

$$\tilde\beta_i := \beta/(2\gamma), \qquad \tilde\alpha_i := \alpha_i \exp(\gamma\|\tilde\beta_i\|^2), \qquad \tilde g = \sum_i \tilde\alpha_i \exp\left(-\gamma\|\cdot - \tilde\beta_i\|^2\right) \in \operatorname{span}(S_\gamma).$$

For any $x \in [0,1]^d$,

$$\begin{aligned}
\tilde g(x) &= \sum_i \tilde\alpha_i \exp\left(-\gamma\|x - \tilde\beta_i\|^2\right) \\
&= \sum_i \alpha_i e^{\gamma\|\tilde\beta_i\|^2} \exp\left(-\gamma\|x\|^2 + 2\gamma\left\langle x, \tilde\beta_i\right\rangle - \gamma\|\tilde\beta_i\|^2\right) \\
&= e^{-\gamma\|x\|^2} \sum_i \alpha_i \exp\left(2\gamma\left\langle x, \tilde\beta_i\right\rangle\right) \\
&= e^{-\gamma\|x\|^2} g(x),
\end{aligned}$$

thus

$$|\tilde{g}(x) - f(x)| = e^{-\gamma\|x\|^2}\left|\tilde{g}(x)e^{\gamma\|x\|^2} - f(x)e^{\gamma\|x\|^2}\right| = e^{-\gamma\|x\|^2}\left|g(x) - f(x)e^{\gamma\|x\|^2}\right| \le \epsilon.$$

□

Remark 1.13. As you can see, it is fairly common within machine learning that our predictors can approximate arbitrary continuous functions, indeed even using the same proof (that is, Lemma 1.7). ◇

2 Succinct

To close, we briefly discussed succinct representations: here the goal is to argue that certain interesting functions can be efficiently approximated by some classes of predictors.

As an initial example (further explored in homework), consider *branching programs*: decision trees where nodes in the same layer can share children.

Consider the function $x \mapsto \frac{1}{d}\sum_{i=1}^{d} x_i$ over $\{0,1\}^d$. A branching program can be constructed with the following form:

- The first layer has a single node with predicate $\mathbb{1}[x_1 \ge 1/2]$.
- Thereafter, layer $i+1$ has $i+1$ nodes, each with predicate $\mathbb{1}[x_i \ge 1/2$, and connected to their left and right parents (drawing the tree in two dimensions).

The branching program operates by tracking the number of bits seen so far: if node $j \le i$ of layer i is reached, that means j coordinates are equal to 1 so far. (Counting from 0.)

This branching program has size $O(d^2)$. In the homework, we'll verify that every decision tree for this problem has size at least 2^d.

References

Gerald B. Folland. *Real analysis: modern techniques and their applications*. Wiley Interscience, 2 edition, 1999.

K. Hornik, M. Stinchcombe, and H. White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, july 1989.

Ingo Steinwart and Andreas Christmann. *Support Vector Machines*. Springer, 1 edition, 2008.