Lecture 16: Empirical risk minimization and model selection

CS 3780/5780, Sp25

Tushaar Gangavarapu (TG352@cornell.edu)

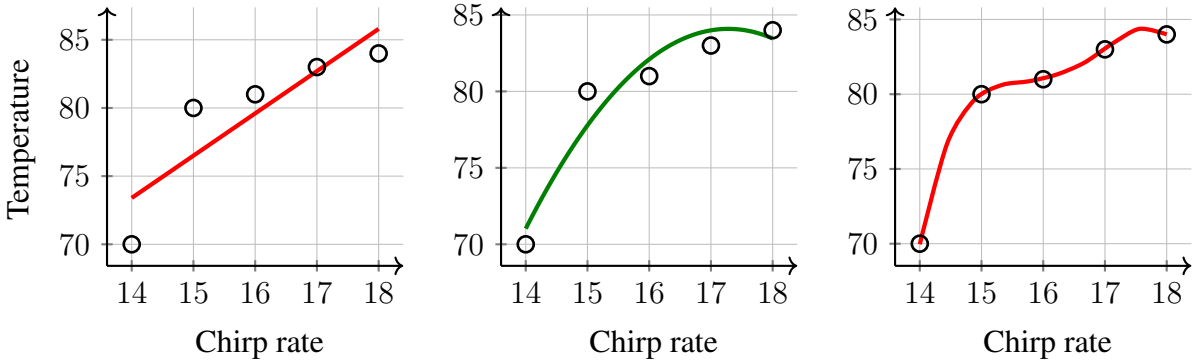
So far, we have seen several learning algorithms, under various settings: supervised vs. unsupervised, linearly- vs. non-linearly separable, classification vs. regression, discriminative vs. generative. In this lecture (and the next), we will move away from specific algorithms and instead, try to view the fundamental learning problem under the lens of *risk* minimization. More specifically, we wish to realize the settings under which a specific algorithm (or, a class of algorithms) is likely to succeed.

As a motivating example, recall our case of predicting the temperature based on the chirp rate. In modeling the temperature, we "assumed" a linear model, i.e., temperature $\propto$ chirp rate. We could have alternatively modeled a quadratic hypothesis:

$$\text{temperature} = \theta_1 \cdot \text{chirp rate} + \theta_2 \cdot \text{chirp rate}^2 + \theta_0,$$

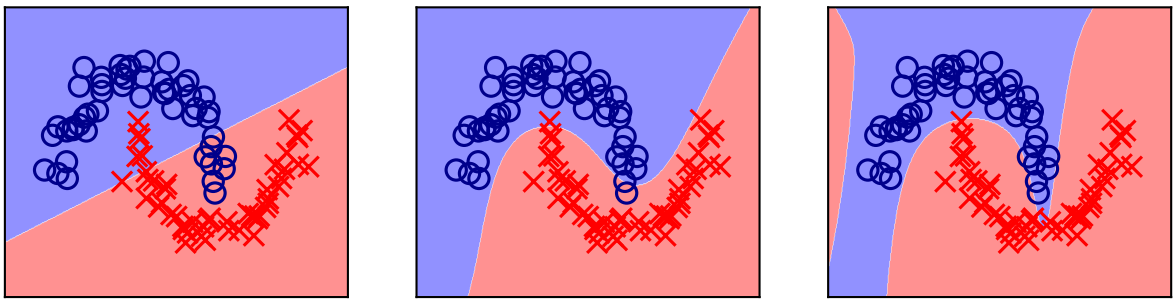
or, better(?) yet, an $n-1$-th order polynomial to exactly fit the n training points:[1]

$$\theta_1 \cdot \text{chirp rate} + \theta_2 \cdot \text{chirp rate}^2 + \cdots + \theta_{n-1} \cdot \text{chirp rate}^{n-1} + \theta_0.$$



From above, it is quite clear that a simple model (linear; left-most) is unable to fit the evident quadratic structure in the data. On the contrary, a 4-th order polynomial (right-most) resulted in a complex model that facilitates a zero training error, but one can easily realize that the model is not a good one for chirp rates not in the training set. In other words, neither the linear model, nor the 4-th order polynomial model *generalize* well.

For completeness, it is important to note that this notion of generalization, or lack thereof, is not emergent from the regression setup; we could as easily argue the same for classification:



[1] An $n-1$-th order polynomial $y^{(j)} = \theta_0 + \theta_1 x^{(j)} + \ldots + \theta_{n-1}(x^{(j)})^{n-1}$ has n unknowns: $\theta_0, \theta_1, \ldots, \theta_{n-1}$, and we have n equations ($y^{(j)}$s). Hence, we can fit an $n-1$-th order polynomial to go through all n data points.

Clearly, both a simple logistic regression model, $y^{(j)} = \sigma(\theta^T x^{(j)}) \geq 0.5$ (left-most) and a complex model with very high-order polynomial kernel, $y^{(j)} = \sigma(\theta_\phi^T \phi(x^{(j)})) \geq 0.5$ for large d (right-most) are both unable to generalize well.

To be more precise, there are two problems here: underfitting (left-most) and overfitting (right-most); we will formalize and discuss the tradeoffs between the two in the next lecture. In this lecture, we are mainly concerned with quantifying the notion of "generalizability."

1 Empirical risk minimization

For the remainder of this notes, we restrict our discussion to binary classification. That said, everything noted here holds for other settings, including multiclass classification and regression.

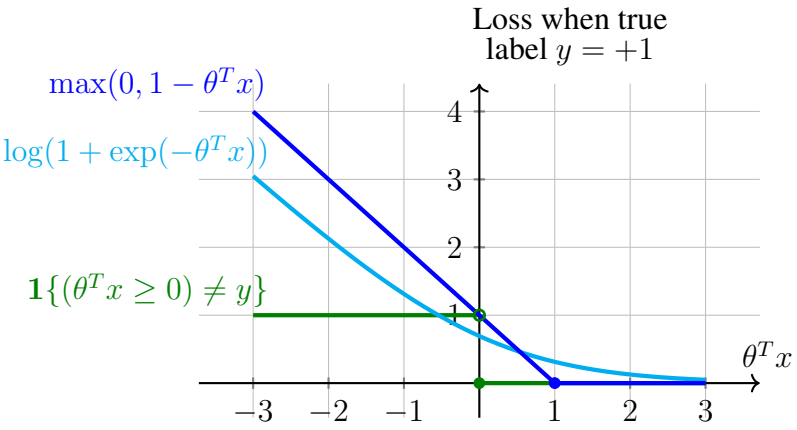
Now, given a training dataset of n samples drawn IID from some distribution $\mathcal{P}$ (unknown to us): $\mathcal{D} = \{(x^{(j)}, y^{(j)}) | 1 \leq j \leq n\}$, we can write the training error (or "*empirical* risk" in learning theory) of some hypothesis function, h_θ as

$$\hat{\varepsilon}(h_\theta) = \frac{1}{n} \sum_{j=1}^{n} \mathbf{1}\{h_\theta(x^{(j)}) \neq y^{(j)}\},$$

i.e., $\hat{\varepsilon}(h_\theta)$ indicates the fraction of training examples that h_θ misclassified.[2] As we've seen many times before, our goal is to learn θ such that

$$\hat{\theta} = \arg\min_\theta \hat{\varepsilon}(h_\theta).$$

(For obvious reasons,) the above framework is known as the *empirical risk minimization* (ERM). As it turns out, the above is an non-smooth and non-convex (gradient is zero everywhere) and in practice, we use convex approximations to this non-convex $0/1$ loss:



To this end, we can formally view logistic regression and support vector machines as convex approximations to the ERM problem.

Let us make this notion of finding optimal parameters to minimize the empirical risk more generic, in that, given a class $\mathcal{H}$ of hypothesis functions, we wish to find the optimal hypothesis such that

$$\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\varepsilon}(h).$$

For example, in linear regression, we have $\mathcal{H} = \{x \mapsto \theta^T x | \theta \in \mathbb{R}^{d+1}\}$, or in case of logistic regression, we have $\mathcal{H} = \{x \mapsto \sigma(\theta^T x) | \theta \in \mathbb{R}^{d+1}\}$. Practically speaking, we often restrict $\mathcal{H}$

[2] We can make the dependence on $\mathcal{D}$ more explicit by writing $\hat{\varepsilon}_\mathcal{D}(h_\theta)$ instead; however, for notational brevity we use $\hat{\varepsilon}(h_\theta)$ to implicitly mean $\hat{\varepsilon}_\mathcal{D}(h_\theta)$ in this notes.

even more by limiting the class of functions to those with specific desirable properties (e.g., solutions with small norms). Realize that this formulation of ERM is more generic than simply saying "find optimal θ"; $\mathcal{H}$ can be any class of functions.

While ERM seems reasonable enough, we don't necessarily (only) care about making accurate predicts on the training set; recall our motivating example of fitting a degree $n-1$ polynomial that passes through all n training points. We are more interested in how well $\hat{h}$ does on instances not seen at training. To this end, we define *generalization* error (or, *true* risk) of $\hat{h}$ as

$$\varepsilon(\hat{h}) = P_{(x,y)\sim\mathcal{P}}(\mathbf{1}\{\hat{h}(x) \neq y\}),$$

i.e., what is the probability that some (x, y) drawn from $\mathcal{P}$ is misclassified by $\hat{h}$.[3]

Two questions naturally arise from the above formulation: (1) what guarantees can we provide about the generalization error of some $\hat{h}$ obtained from ERM, and (2) can we somehow estimate the generalization error of a given hypothesis?

2 Bounds on ERM-chosen hypothesis

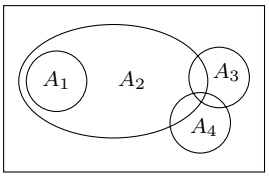
We will now attempt to show how well a hypothesis generalizes. Specifically, we wish to make statements of the form "if the training set is *at least* this large, the model class *at most* this complex, then $\varepsilon(h)$ will be within some γ of $\hat{\varepsilon}(h)$ with *at least* this probability." In other words, we wish to bound the $|\varepsilon(h) - \hat{\varepsilon}(h)|$ in terms of $|\mathcal{H}|$ and n.

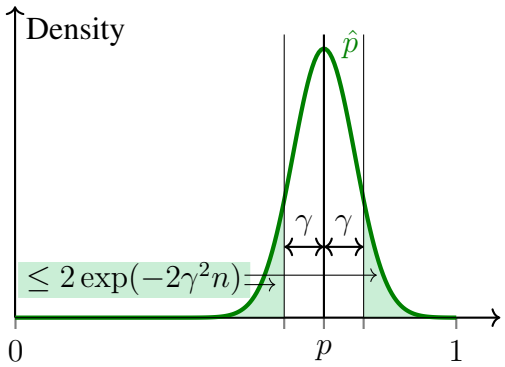
2.1 Preliminaries

To be able to bound $|\varepsilon(h) - \hat{\varepsilon}(h)|$, we will require two lemmas; the choice of these lemmas will be obvious in a moment.

Lemma (The union bound)**.** *Let $A_1, A_2, \ldots, A_k$ be k different events (not necessarily independent), then*

$$P(A_1 \cup \cdots \cup A_k) \leq P(A_1) + \cdots + P(A_k).$$





Lemma (The Chernoff bound)**.** *If we toss a biased coin with (true) probability of heads $P(H) = p$, n times, and let $\hat{p}$ be the fraction of times we observed heads in our coin tosses, i.e., $\hat{p}$ is our empirical estimate for p, then*

$$P(|p - \hat{p}| > \gamma) \leq 2\exp(-2\gamma^2 n),$$

where $\gamma > 0$ is some constant.

While will not prove this lemma, we wish to give an intuition of what the bound is saying and why it might be useful. While the above bound holds true for any k, let us assume a large k. Now, from the central limit theorem, we realize that the CDF of $\hat{p}$ can be approximated using a Gaussian. Now, all that the Chernoff bound is saying is that for a chosen γ, the total probability mass in the tails (as shown in the illustration above) is at most $2\exp(-2\gamma^2 n)$.

[3] It is important to draw attention to the notational convention that $\hat{*}$ (with hat) is estimating $*$ (without the hat). For instance, the training error $\hat{\varepsilon}$ is estimating the generalization error ε.

More importantly, the bound tells us that as n grows, the width of the Gaussian shrinks and the mass in the tails (or, equivalently, probability of misestimating p) exponentially decreases.

2.2 Relating training error to generalization error

In this notes, we will only consider the case of a finite $\mathcal{H}$ and regard the case of an infinite $\mathcal{H}$ out-of-scope.[4]

Let, $\mathcal{H} = \{h_1, \ldots, h_k\}$ be k hypotheses; ERM picks a hypothesis $\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\varepsilon}(h)$. First, we wish to relate $\hat{\varepsilon}(h_j)$ to $\varepsilon(h_j)$ for some $h_j \in \mathcal{H}$. It is easy to realize that $\mathbf{1}\{h_j(x) \neq y\} = Z$ is a Bernoulli random variable (similar to a coin toss) and $P(Z = 1)$ is exactly $\varepsilon(h_j)$. (This is equivalent to the true probability of heads in a coin toss.) Moreover, the training error of h_j,

$$\hat{\varepsilon}(h_j) = \frac{1}{n} \sum_{j=1}^{n} \mathbf{1}\{h_j(x^{(j)}) \neq y^{(j)}\} = \frac{1}{n} \sum_{j=1}^{n} Z^{(j)},$$

i.e., $\hat{\varepsilon}(h_j)$ is the fraction of n examples where $Z^{(j)} = 1$. (This is equivalent to tossing the coin n times to estimate $P(H)$ as the fraction of heads.) Hence, from the Chernoff bound, we have

$$P(|\varepsilon(h_j) - \hat{\varepsilon}(h_j)| > \gamma) \leq 2 \exp(-2\gamma^2 n).$$

Assuming n is large, then the probability that the generalization error is far from the training error is bounded (and small) for a specific h_j.

Now, let us extend this result to bound the training error of any $h_j \in \mathcal{H}$. To do this, we define A_j to be the event that $|\varepsilon(h_j) - \hat{\varepsilon}(h_j)| > \gamma$. We wish to bound the probability that one or more h_js result in $|\varepsilon(h_j) - \hat{\varepsilon}(h_j)| > \gamma$, i.e.,

$$P(A_1 \cup \ldots \cup A_k) \leq \sum_{j=1}^{k} P(A_j) \leq \sum_{j=1}^{k} 2 \exp(-2\gamma^2 n) = 2k \exp(-2\gamma^2 n).$$

This follows that the probability that *none* of the h_j result in $|\varepsilon(h_j) - \hat{\varepsilon}(h_j)| > \gamma$ is bounded as

$$1 - P(A_1 \cup \ldots \cup A_k) \geq 1 - 2k \exp(-2\gamma^2 n).$$

Hence, for *any* hypothesis in $\mathcal{H}$, the probability that the training error $\hat{\varepsilon}(h)$ is within some γ of the generalization error $\varepsilon(h)$ is *at least* $1 - 2k \exp(-2\gamma^2 n)$.

There are three quantities of interest here: dataset size n, the deviation γ, and the probability of error. The above bounds the probability of error in terms of n and γ. Alternatively, we could ask the following: Given some $\gamma > 0, 0 < \delta < 1$, how large a dataset is needed to guarantee that with probability $1 - \delta$, the training error is within γ of the generalization error? We can solve for n under $1 - \delta \geq 1 - 2k \exp(-2\gamma^2 n)$ to get:

$$n \geq \frac{1}{2\gamma^2} \log \frac{2k}{\delta}.$$

The key observation here is that we need (only) an order of $\log k$ examples to satisfy the above bound.

Alternatively, we could hold n and δ constant, and note that with probability $1 - \delta$, we have that for all $h_j \in \mathcal{H}$:

$$|\varepsilon(h_j) - \hat{\varepsilon}(h_j)| \leq \sqrt{\frac{1}{2n} \log \frac{2k}{\delta}}.$$

[4] While the arguments itself are not hard for the case of infinite $\mathcal{H}$, it requires discussing additional concepts such as Vapnik-Chervonenkis dimension, $\mathrm{VC}(\mathcal{H})$.

What can we say about $\hat{h}$? Now, the above results hold for any $h_j \in \mathcal{H}$. We wish to be more specific and understand what can be said about the ERM-chosen $\hat{h} = \arg\min_{h\in\mathcal{H}} \hat{\varepsilon}(h)$. To understand this, let us define $h^\star = \arg\max_{h\in\mathcal{H}} \varepsilon(h)$ to be the best possible hypothesis we could've chosen from $\mathcal{H}$, had we known $\mathcal{P}$. Now, it makes sense to compare the performance of $\hat{h}$ to that of $h^\star$. Under the guarantee of $|\varepsilon(h) - \hat{\varepsilon}(h)| \le \gamma$ for all $h \in \mathcal{H}$, we have

$$
\begin{aligned}
\varepsilon(\hat{h}) &\le \hat{\varepsilon}(\hat{h}) + \gamma \\
&\le \hat{\varepsilon}(h^\star) + \gamma && \text{by defn. } \hat{h} \text{ is the hypothesis w/ lowest training error in } \mathcal{H} \\
&= \hat{\varepsilon}(h^\star) - \varepsilon(h^\star) + \varepsilon(h^\star) + \gamma && \text{add and subtract } \varepsilon(h^\star) \\
&\le \gamma + \varepsilon(h^\star) + \gamma && |\varepsilon(h) - \hat{\varepsilon}(h)| \le \gamma \text{ for all } h \in \mathcal{H} \\
&= \varepsilon(h^\star) + 2\gamma.
\end{aligned}
$$

Thus, we have that $\hat{h}$ is *at most* 2γ worse than the best-in-$\mathcal{H}$ hypothesis!! With that, we can tie everything up into a theorem:

Theorem (Oracle inequality)**.** *Let $|\mathcal{H}| = k$ and let the dataset size n and some constant $0 < \delta < 1$ be fixed, then with probability $1 - \delta$, we have that*

$$
\varepsilon(\hat{h}) \le \varepsilon(h^\star) + 2\sqrt{\frac{1}{2n}\log\frac{2k}{\delta}} = \min_{h\in\mathcal{H}} \varepsilon(h) + 2\sqrt{\frac{1}{2n}\log\frac{2k}{\delta}}.
$$

A consequence of the above is quite evident: if we increase k, i.e., switch to a much larger hypothesis class (e.g., from a class of linear polynomials to degree $n-1$ polynomials), then $\min_{h\in\mathcal{H}} \varepsilon(h)$ can only decrease but the "$2\sqrt{\dots}$" term increases. Clearly, there is some tradeoff between model complexity and generalization, which we will explore in the next lecture. It is also worthwhile to think about how regularization affects the above oracle inequality.

2.3 On the "finiteness" of $\mathcal{H}$

Is it reasonable to assume a finite $\mathcal{H}$? In practice, every $\mathcal{H}$ is finite. For a model with d total parameters, each represented in fl32 format (a.k.a., single-precision format), we have a total of 2^{32d} hypotheses.

To put this into perspective, consider DeepSeek-R1,[5] a state-of-the-art reasoning model, better than OpenAI-o1 at certain tasks, which has 671B parameters. Now, $|\mathcal{H}| = 2^{32\times 671\text{B}} = 2^{21,472\text{B}} \approx \exp(1.5 \times 10^{13})$. For $0/1$ loss, this means that for our bound to show that ERM learns a 671B model even to within an extremely loose $\gamma = 20\%$ additive error with 50% probability would require

$$
n \ge \frac{1}{2\times 0.2^2}\log\frac{2\exp(1.5\times 10^{13})}{0.5} = 12.5(\log(4) + 1.5\times 10^{13}) \approx 1.9\times 10^{14} = 190\text{T}.^{6}
$$

The union bound above (also known as the *uniform convergence bound*) estimates that a whopping 190T examples are needed (at the very least) to bound the training error to be within 20% of the generalization error. (For comparison, DeepSeek was trained on $14.8\text{T} \ll 190\text{T}$ tokens.)

The above example puts into perspective that the union bound, while useful in analysis, isn't everything, this is especially true for large $\mathcal{H}$. This is mainly because the union bound ignores all structure in $\mathcal{H}$: a small change to one of the parameters, say, $\theta_{\text{old}} + 0.0000001$, would result

[5] https://github.com/deepseek-ai/DeepSeek-R1.

[6] 190T seconds is about 6M years; if you counted one parameter per second, you'd be at it for longer than human history has existed.

in an error that is treated totally separately from that with θ_{old}, when in reality those two errors are tightly correlated. For completeness, we simply note that there are other approaches that allow us to better bound the number of training examples needed to learn "well" using $\mathcal{H}$, which also handle an infinite $\mathcal{H}$.[7]

3 Model selection

The oracle inequality often depicts the worst-case scenario, which can be overly pessimistic, as illustrated in the DeepSeek example. As a result, it is often more meaningful to estimate the generalization error of $\hat{h}$, rather than merely bound it.

A straightforward choice to consider is to use the training error, $\hat{\varepsilon}(\hat{h})$, as an estimate for the generalization error $\varepsilon(\hat{h})$. It is easy to reason that a complex-enough model could achieve $\hat{\varepsilon}(\hat{h}) = 0$, while unable to generalize to any unseen examples. This clearly motivates the need for an "unseen-during-training" dataset, which is only used to estimate $\varepsilon(\hat{h})$. There are several strategies to achieve this.

3.1 Hold-out cross-validation

Given a training dataset $\mathcal{D}$ of n samples, we do the following:

(a) Randomly split $\mathcal{D}$ into train ($\mathcal{D}_{\text{train}}$; say, 70% of the data) and test ($\mathcal{D}_{\text{CV}}$; remaining 30% of the data) sets. Here, $\mathcal{D}_{\text{CV}}$ is the hold-out cross-validation set.

(b) Train each hypothesis, $h_j \in \mathcal{H}$, only on $\mathcal{D}_{\text{train}}$.

(c) Select the hypothesis h_j with the lowest $\hat{\varepsilon}_{\mathcal{D}_{\text{CV}}}(h_j)$. Here, we use the empirical error of h_j on $\mathcal{D}_{\text{CV}}$ as the estimate of $\varepsilon(h_j)$ as $\mathcal{D}_{\text{CV}}$ was never seen during training.

(With some exceptions,) it is often the case that the h_j obtained from step-(c) above is retrained on the entire dataset, $\mathcal{D}$, before being deployed.

The clear disadvantage of such (hold-out) cross-validation is that it wastes 30% of the data. Even if we finally train the model on $\mathcal{D}$, we are still only using 70% of the data to find the optimal hypothesis. This is a non-issue when dealing with significantly large datasets, but is problematic when, say, we have 20 total samples.

3.2 K-fold cross-validation

An alternate strategy to cross-validation that effectively utilizes the whole dataset is the k-fold cross-validation, which is as follows:

(a) Randomly split the dataset into k disjoint sets of n/k samples each: $\mathcal{D}_1, \ldots, \mathcal{D}_k$.

(b) For each hypothesis, $h_j \in \mathcal{H}$, we evaluate it as follows:
For $l = 1, \ldots, k$:
Train hypothesis h_j on $\mathcal{D}_1 \cup \cdots \cup \mathcal{D}_{l-1} \cup \mathcal{D}_{l+1} \cup \mathcal{D}_k$, i.e., train on all subsets except for $\mathcal{D}_l$.
Test the resultant hypothesis on $\mathcal{D}_l$ to obtain $\hat{\varepsilon}_{\mathcal{D}_l}(h_j)$.

[7] For further reading: Vapnik showed that the number of training examples needed to learn "well" in $\mathcal{H}$ is linear in $\text{VC}(\mathcal{H})$.

Estimate $\varepsilon(h_j)$ as the average of $\hat{\varepsilon}_{\mathcal{D}_l}(h_j)$ across all ls (a.k.a., folds).

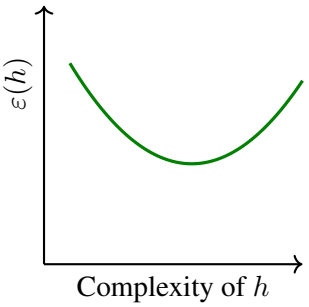
(c) Select the hypothesis h_j with the lowest estimated generalization error.

As before, you can retrain the output hypothesis from step-(c) above on all entire dataset, $\mathcal{D}$, before deployment. A typical choice for k would be 5 or 10. While we don't "waste" any data, k-fold cross-validation can be extremely expensive (think: training DeepSeek-R1 with 671B parameters $5\times$).

Leave-one-out cross-validation. In cases where data is extremely scarce (e.g., $n = 10$), we resort to setting $k = n$, i.e., we repeatedly train on $n - 1$ examples and test on the one held-out example. The resulting n errors (one per example) are averaged to obtain the estimate for the generalization error. Since we're leaving out one example at a time, this method is known as the *leave-one-out* cross-validation.[8]

4 Conclusion

In this lecture, we discussed the framework of empirical risk minimization and realized the guarantees that the training error offers about the generalization error of the underlying hypothesis. All in all, ERM allows us to make statements of the form: "if the dataset is at least this large and the hypothesis class at most this complex, then with at least this probability, the training error will be within some γ of generalization error." Additionally, we showed that in asymptotics, we need only $\mathcal{O}(d)$ training examples to learn well in a hypothesis class. Finally, we noted the oracle inequality which allowed us to bound the generalizability of ERM-obtained hypothesis to the best-in-$\mathcal{H}$ hypothesis. Following the oracle inequality, we realized a clear tradeoff between generalization and model complexity—an illustration to that effect is shown to the right—we will explore this in the next lecture.



We also discussed cross-validation (and variants) as a way of facilitating model selection to estimate the generalization error using a held-out test set.

[8] We have seen an example of this on the practice prelim (see Q1 on k-nearest neighbors).

A Notation

$\mathcal{D}$	The training dataset of n samples
n	The number of training samples in the dataset $\mathcal{D}$
d	The number of features; is also the number of model parameters
$x^{(j)} \in \mathbb{R}^d$	The d-dimensional feature vector associated with the j-th training sample
$y^{(j)}$	The target value associated with the j-th training sample (real-valued in regression; discrete in classification)
$\mathcal{P}$	The distribution from which the data samples are drawn; we write $(x, y) \sim \mathcal{P}$ to indicate that (x, y) was drawn from $\mathcal{P}$
$x_\ell^{(j)} \in \mathbb{R}$	The ℓ-th element of $x^{(j)}$
θ	The parameters of the model (a.k.a., hypothesis)
h_θ	The hypothesis function, parameterized by θ (e.g., $h_\theta(x) = \theta^T x$ for linear regression)
$\mathbf{1}\{a \neq b\}$	Indicator (or boolean) function that returns 1 if $a \neq b$ and 0 otherwise
$\hat{\varepsilon}_{\mathcal{D}}(h)$ or $\hat{\varepsilon}(h)$	The training error or *empirical risk* of the hypothesis function h, computed for the dataset $\mathcal{D}$
$\mathcal{H}$	A class of hypothesis functions (can be an infinite)
$\hat{h}$	The ERM-chosen hypothesis that has the lowest training error among all $h \in \mathcal{H}$
$\varepsilon(h)$	The generalization error or *true risk* of the hypothesis function h
$\gamma > 0$	Indicates how far away $\varepsilon(h)$ is from $\hat{\varepsilon}(h)$ for some hypothesis h
Z or $Z^{(j)}$	Shorthand notation for $\mathbf{1}\{h(x) \neq y\}$ (or $\mathbf{1}\{h(x^{(j)}) \neq y^{(j)}\}$
k	The number of hypothesis functions in $\mathcal{H}$, i.e., $\lvert\mathcal{H}\rvert$; also used for "k" in k-fold cross-validation
$0 < \delta < 1$	From the union bound, we have that with *at least* $1 - \delta$ probability, $P(\neg h_j \in \mathcal{H}$ s.t. $\lvert\varepsilon(h_j) - \hat{\varepsilon}(h_j)\rvert > \gamma)$; "$\neg$" indicates "logical not"
$h^\star$	The best-in-$\mathcal{H}$ possible hypothesis that achieves the lowest generalization error
$\mathcal{D}_{\text{train}}$ and $\mathcal{D}_{\text{CV}}$	The dataset $\mathcal{D}$ is split into train and cross-validation sets that are disjoint
$\mathcal{D}_1, \ldots, \mathcal{D}_k$	The dataset $\mathcal{D}$ is split into k disjoint sets, used in k-fold cross-validation

References

P. Bartlett. CS281B Lecture notes on Model Selection. *CS281B/Stat241B (Spring 2008) Statistical Learning Theory Lecture notes*, 1 (1):1–4, 2008. URL https://people.eecs.berkeley.edu/~bartlett/courses/281b-sp08/20.pdf. Scribed by: Christopher Hundt.

J. Hoogland. Empirical risk minimization is fundamentally confused, Mar 2023. URL https://www.lesswrong.com/posts/zuYRyC3zghzgXLpEW/empirical-risk-minimization-is-fundamentally-confused.

A. Ng. CS229 Lecture notes. *CS229 Lecture notes*, 1(1):139–142, 2000a. URL https://cs229.stanford.edu/main_notes.pdf. Version: June 11, 2023.

A. Ng. CS229 Lecture notes. *CS229 Lecture notes*, 1(1):126–131, 2000b. URL https://cs229.stanford.edu/main_notes.pdf. Version: June 11, 2023.

D. J. Sutherland. CPSC532D Lecture notes on Uniform convergence with finite classes. *CPSC 532D (Fall 2024) Modern Statistical Learning Theory Lecture notes*, 1(1):1–4, 2024. URL https://www.cs.ubc.ca/~dsuth/532D/24w1/notes/2-finite-classes.pdf.

(Last compiled: 4/8/2025, 11.29am ET.)