07-380 Notes
Pre-reading: ML Theory

Carnegie Mellon University
School of Computer Science

Contents

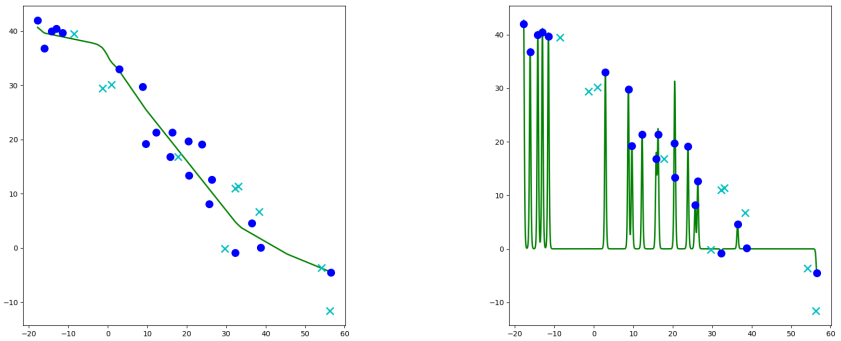
1 Learning Theory **1**
1.1 Training Error Is Not the Goal 1
1.2 Model for Supervised Learning 2
1.3 Bias and Variance 2
1.4 Risk 4
1.5 Three Hypotheses of Interest 4
1.6 Example: Three Points and a Line 5

1 Learning Theory

So far we have trained models by minimizing a loss on the training data. Learning theory asks what that tells us about data we have *not* seen. This pre-reading sets up the vocabulary and notation; in lecture we use it to put bounds on how far training error can be from the error we actually care about.

1.1 Training Error Is Not the Goal

Two neural networks trained on the same data (blue dots are training points, cyan crosses are validation points):



The network on the right has zero training error and the one on the left does not, yet the left one is clearly the better model. We want a model that does well on new data, that is, one that **generalizes**. To say this precisely we need a model of where data comes from.

1.2 Model for Supervised Learning

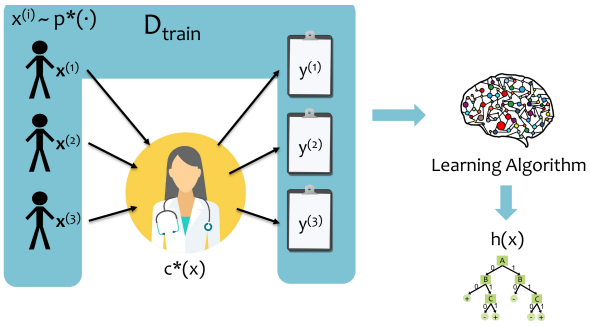


Figure: Matt Gormley, CMU MLD

- **Inputs** $\mathbf{x} \in \mathcal{X}$ (all possible patients) and **outputs** $y \in \mathcal{Y}$ (all possible diagnoses).
- Inputs are drawn from an *unknown* **distribution** $p^*(\mathbf{x})$.
- An *unknown* **target function** $c^* : \mathcal{X} \to \mathcal{Y}$ gives the true output for every input (the doctor).
- The **training data** is $\mathcal{D} = \left\{\left(\mathbf{x}^{(i)}, y^{(i)}\right)\right\}_{i=1}^{N}$, where $\mathbf{x}^{(i)} \sim p^*(\mathbf{x})$ and $y^{(i)} = c^*\left(\mathbf{x}^{(i)}\right)$.
- The **hypothesis space** $\mathcal{H}$ is the set of candidate functions $h : \mathcal{X} \to \mathcal{Y}$ that we are willing to consider (all decision trees, or all lines $h(x) = wx + b$).
- The **learning algorithm** takes $\mathcal{D}$ and returns one hypothesis $h \in \mathcal{H}$.

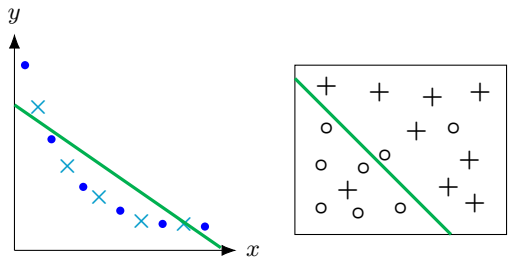
The training data is a *random sample*. A different draw from p^* gives a different $\mathcal{D}$, and so a different learned h.

1.3 Bias and Variance

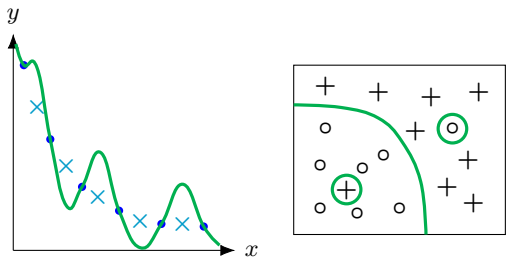
Overfitting and underfitting have more precise relatives. Picture training the same kind of model over and over, each time on a fresh training set drawn from p^*.

- **Variance**: how much the learned h changes as we swap out the training set.
- **Bias**: how far the *average* learned h is from the target function c^*.

The usual pictures, for regression and for classification. Dots, + and ∘ are training points; crosses are validation points.



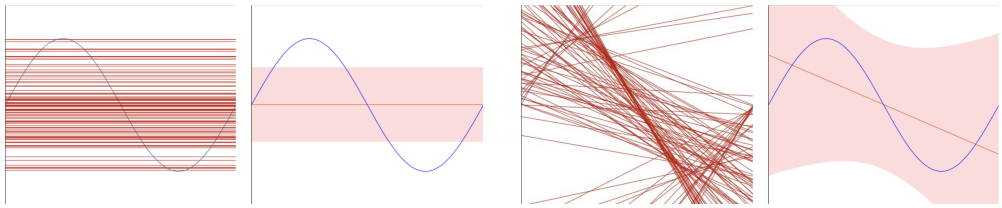
High bias, low variance (underfitting)



Low bias, high variance (overfitting)

- Left: a new training set would give almost the same line (low variance), but a line is the wrong shape however much data we get (high bias).
- Right: the model can follow any training set (low bias), but it chases every point, so a new training set would give a very different function (high variance).

Bias and variance as numbers: c^* is a sine wave (blue) and every training set is just $N = 2$ points on it. $\mathcal{H}_0$ is the set of constant functions $h(x) = b$ and $\mathcal{H}_1$ is the set of lines $h(x) = wx+b$. For each, the first plot shows the hypotheses learned from many different training sets and the second shows their average (red line) and their spread (shaded).



$\mathcal{H}_0$: constants $\mathcal{H}_1$: lines

Figures: Henry Chai, CMU MLD

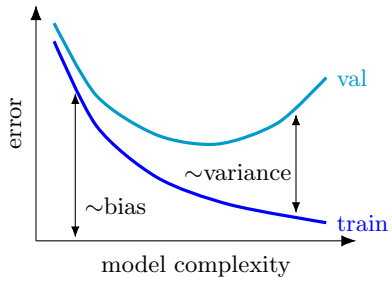
Because we know c^* here, bias and variance can actually be computed. With squared error, bias is the squared distance from the average hypothesis to c^*, variance is the squared distance from each learned hypothesis to the average hypothesis, and the two add up to the true error we expect from a freshly trained model:

	$\mathcal{H}_0$: constants	$\mathcal{H}_1$: lines
Bias	0.50	0.21
Variance	0.25	1.74
Expected true error = bias + variance	0.75	1.95

Lines can get closer to a sine wave than constants can (lower bias), but with only two training points the learned line swings wildly (much higher variance), so the simpler $\mathcal{H}_0$ is the better choice here.

In practice: computing bias and variance requires c^* and p^*, which we never have. Instead we read the symptoms off the training and validation error.

	h_A	h_B
Training error rate	0.25	0.10
Validation error rate	0.30	0.30
Symptom of	high bias low variance	lower bias? high variance



- High training error suggests high bias. A large gap between training and validation error suggests high variance. These are indicators, not measurements.
- Is h_B's training error of 0.10 low? Only compared to the best achievable error, which we do not know. Human performance on the same task is a useful stand-in.

Watch out: "bias" means several different things in machine learning.

- The bias in the bias-variance tradeoff (this section).
- The bias term of a linear model: the b in $wx + b$.
- Inductive bias: the assumptions a learning algorithm relies on to generalize (for nearest neighbor, that nearby points share a label).
- Societal and dataset bias: a model's predictions reflecting the biases of the people and data it learned from.

1.4 Risk

See the 07-280 Optimization and Linear Regression pre-reading (Section 3) for a reminder of loss, risk and empirical risk. In the notation of Section 1.2, with loss function $\ell(y, \hat{y})$:

Risk	$R(h) = \mathbb{E}_{\mathbf{x} \sim p^*}\left[\ell\big(c^*(\mathbf{x}),\, h(\mathbf{x})\big)\right]$	expected loss on a new input
Empirical risk	$\hat{R}(h) = \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} \ell\Big(y^{(i)},\, h\big(\mathbf{x}^{(i)}\big)\Big)$	average loss on the training data $\mathcal{D}$

The hat on $\hat{R}$ means "estimated from data". We can always compute $\hat{R}(h)$. We can never compute $R(h)$, because it needs p^* and c^*.

For classification with zero-one loss, $\ell(y, \hat{y}) = \mathbb{1}(y \neq \hat{y})$, both become error rates:

$R(h) = P_{\mathbf{x} \sim p^*}\big(c^*(\mathbf{x}) \neq h(\mathbf{x})\big)$	probability that h is wrong on a new input
$\hat{R}(h) = \dfrac{1}{N}\displaystyle\sum_{i=1}^{N} \mathbb{1}\Big(y^{(i)} \neq h\big(\mathbf{x}^{(i)}\big)\Big)$	fraction of the training points that h gets wrong

The same two quantities go by several names:

$R(h)$	risk, expected risk, **true error**, generalization error
$\hat{R}(h)$	empirical risk, **training error**

1.5 Three Hypotheses of Interest

True function	c^*	labeled the data: $y^{(i)} = c^*\big(\mathbf{x}^{(i)}\big)$
Expected risk minimizer	$h^* = \operatorname{argmin}_{h \in \mathcal{H}} R(h)$	lowest true error in $\mathcal{H}$
Empirical risk minimizer	$\hat{h} = \operatorname{argmin}_{h \in \mathcal{H}} \hat{R}(h)$	lowest training error in $\mathcal{H}$

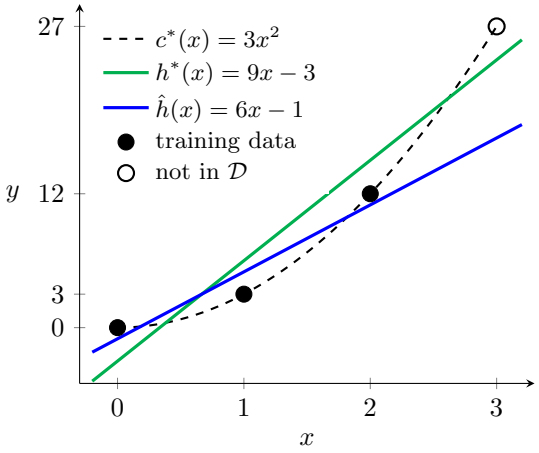
Empirical risk minimization (ERM) is the learning algorithm that returns $\hat{h}$. Of the three, $\hat{h}$ is the only one we can compute.

Notation warning: the 07-280 pre-reading wrote h^* for the empirical risk minimizer. From here on that is $\hat{h}$, and h^* is the best hypothesis in $\mathcal{H}$ for the true distribution.

1.6 Example: Three Points and a Line

To see all of these as numbers, we make up a tiny world in which we know p^* and c^*.

- Inputs: $\mathcal{X} = \{0, 1, 2, 3\}$, with p^* uniform: $p^*(x) = \frac{1}{4}$ for each x.
- Target function: $c^*(x) = 3x^2$.
- Hypothesis space: $\mathcal{H}$ = all lines $h(x) = wx + b$.
- Loss: squared error, $\ell(y, \hat{y}) = (y - \hat{y})^2$.
- Training data: we happened to draw $x = 0, 1, 2$, so $\mathcal{D} = \{(0, 0),\ (1, 3),\ (2, 12)\}$ and $N = 3$.



		$\hat{h}(x) = 6x - 1$		$h^*(x) = 9x - 3$	
x	$c^*(x)$	$\hat{h}(x)$	loss	$h^*(x)$	loss
0	0	-1	1	-3	9
1	3	5	4	6	9
2	12	11	1	15	9
3	27	17	100	24	9

The first three rows are the training data.

Empirical risk minimizer: least squares on the three training points gives $\hat{h}(x) = 6x - 1$. Its empirical risk averages the loss over the $N = 3$ training points, and its risk takes the expectation over all of p^*:

$$\hat{R}(\hat{h}) = \tfrac{1}{3}(1 + 4 + 1) = 2 \qquad R(\hat{h}) = \tfrac{1}{4}(1 + 4 + 1 + 100) = 26.5$$

Expected risk minimizer: least squares on all four inputs, weighted by p^*, gives $h^*(x) = 9x - 3$:

$$\hat{R}(h^*) = \tfrac{1}{3}(9 + 9 + 9) = 9 \qquad R(h^*) = \tfrac{1}{4}(9 + 9 + 9 + 9) = 9$$

True function: c^* has zero loss everywhere, but it is not a line, so $c^* \notin \mathcal{H}$.

	c^*	h^*	$\hat{h}$
Training error $\hat{R}$	0	9	2
True error R	0	9	26.5
In $\mathcal{H}$?	no	yes	yes

What to notice:

- $\hat{R}(\hat{h}) \leq \hat{R}(h^*)$ and $R(h^*) \leq R(\hat{h})$. Both always hold, by the definitions of $\hat{h}$ and h^*.
- The training error of $\hat{h}$, 2, badly underestimates its true error, 26.5.
- $\hat{h}$ depends on the sample. Had we drawn $x = 1, 2, 3$ instead, we would have learned a different line.
- Outside of a made-up world, the true error row of the table is hidden. All we see is $\hat{R}$.

In lecture: given $\hat{R}(\hat{h})$, the number of training points N and the size of $\mathcal{H}$, what can we say about $R(\hat{h})$?