Chapter 8

Generalization

This chapter discusses tools to analyze and understand the generalization of machine learning models, i.e, their performances on unseen test examples. Recall that for supervised learning problems, given a training dataset $\{(x^{(i)}, y^{(i)})\}_{i=1}^n$, we typically learn a model h_θ by minimizing a loss/cost function $J(\theta)$, which encourages h_θ to fit the data. E.g., when the loss function is the least square loss (aka mean squared error), we have $J(\theta) = \frac{1}{n}\sum_{i=1}^n (y^{(i)} - h_\theta(x^{(i)}))^2$. This loss function for training purposes is oftentimes referred to as the **training** loss/error/cost.

However, minimizing the training loss is **not** our ultimate goal—it is merely our approach towards the goal of learning a predictive model. The most important evaluation metric of a model is the loss on unseen test examples, which is oftentimes referred to as the test error. Formally, we sample a test example (x, y) from the so-called test distribution $\mathcal{D}$, and measure the model's error on it, by, e.g., the mean squared error, $(h_\theta(x) - y)^2$. The expected loss/error over the randomness of the test example is called the test loss/error,[1]

$$L(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[(y - h_\theta(x))^2] \tag{8.1}$$

Note that the measurement of the error involves computing the expectation, and in practice, it can be approximated by the average error on many sampled test examples, which are referred to as the test dataset. Note that the key difference here between training and test datasets is that the test examples

[1]In theoretical and statistical literature, we oftentimes call the uniform distribution over the training set $\{(x^{(i)}, y^{(i)})\}_{i=1}^n$, denoted by $\widehat{\mathcal{D}}$, an empirical distribution, and call $\mathcal{D}$ the population distribution. Partly because of this, the training loss is also referred to as the empirical loss/risk/error, and the test loss is also referred to as the population loss/risk/error.

are *unseen*, in the sense that the training procedure has not used the test examples. In classical statistical learning settings, the training examples are also drawn from the same distribution as the test distribution $\mathcal{D}$, but still the test examples are unseen by the learning procedure whereas the training examples are seen.[2]

Because of this key difference between training and test datasets, even if they are both drawn from the same distribution $\mathcal{D}$, the test error is not necessarily always close to the training error.[3] As a result, successfully minimizing the training error may not always lead to a small test error. We typically say the model **overfits** the data if the model predicts accurately on the training dataset but doesn't generalize well to other test examples, that is, if the training error is small but the test error is large. We say the model **underfits** the data if the training error is relatively large[4] (and in this case, typically the test error is also relatively large.)

This chapter studies how the test error is influenced by the learning procedure, especially the choice of model parameterizations. We will decompose the test error into "bias" and "variance" terms and study how each of them is affected by the choice of model parameterizations and their tradeoffs. Using the bias-variance tradeoff, we will discuss when overfitting and underfitting will occur and be avoided. We will also discuss the double descent phenomenon in Section 8.2 and some classical theoretical results in Section 8.3.

[2]These days, researchers have increasingly been more interested in the setting with "domain shift", that is, the training distribution and test distribution are different.

[3]the difference between test error and training error is often referred to as the generalization gap. The term *generalization error* in some literature means the test error, and in some other literature means the generalization gap.

[4]e.g., larger than the intrinsic noise level of the data in regression problems.

8.1 Bias-variance tradeoff

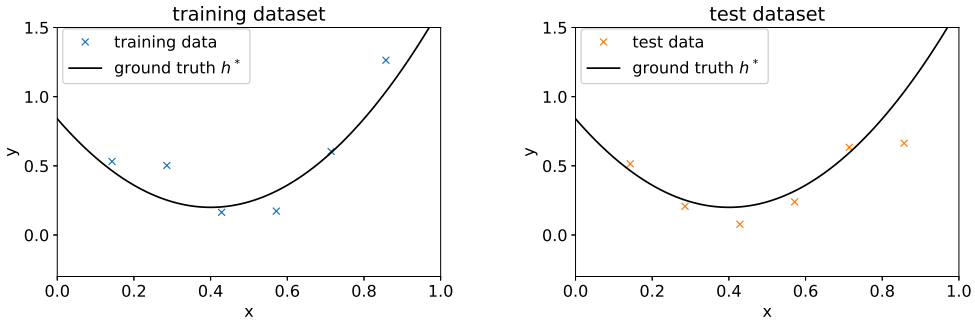

Figure 8.1: A running example of training and test dataset for this section.

As an illustrating example, we consider the following training dataset and test dataset, which are also shown in Figure 8.1. The training inputs $x^{(i)}$'s are randomly chosen and the outputs $y^{(i)}$ are generated by $y^{(i)} = h^\star(x^{(i)}) + \xi^{(i)}$ where the function $h^\star(\cdot)$ is a quadratic function and is shown in Figure 8.1 as the solid line, and $\xi^{(i)}$ is the a observation noise assumed to be generated from $\sim N(0, \sigma^2)$. A test example (x, y) also has the same input-output relationship $y = h^\star(x) + \xi$ where $\xi \sim N(0, \sigma^2)$. It's impossible to predict the noise ξ, and therefore essentially our goal is to recover the function $h^\star(\cdot)$.

We will consider the test error of learning various types of models. When talking about linear regression, we discussed the problem of whether to fit a "simple" model such as the linear "$y = \theta_0 + \theta_1 x$," or a more "complex" model such as the polynomial "$y = \theta_0 + \theta_1 x + \cdots \theta_5 x^5$."

We start with fitting a linear model, as shown in Figure 8.2. The best fitted linear model cannot predict y from x accurately even on the training dataset, let alone on the test dataset. This is because the true relationship between y and x is not linear—any linear model is far away from the true function $h^\star(\cdot)$. As a result, the training error is large and this is a typical situation of *underfitting*.

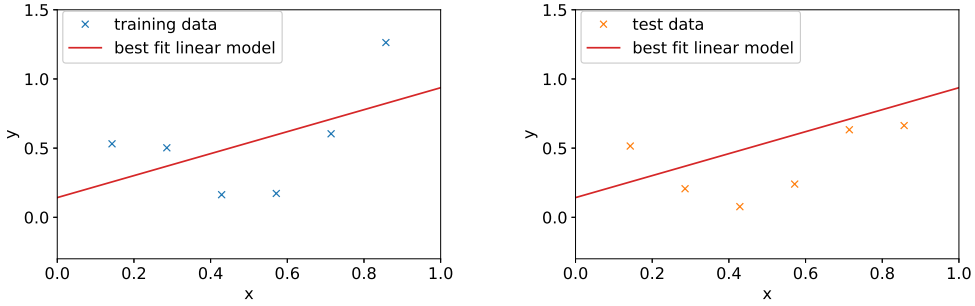

Figure 8.2: The best fit linear model has large training and test errors.

The issue cannot be mitigated with more training examples—even with a very large amount of, or even infinite training examples, the best fitted linear model is still inaccurate and fails to capture the structure of the data (Figure 8.3). Even if the noise is not present in the training data, the issue still occurs (Figure 8.4). Therefore, the fundamental bottleneck here is the linear model family's inability to capture the structure in the data—linear models cannot represent the true quadratic function $h^\star$—, but not the lack of the data. Informally, we define the **bias** of a model to be the test error even if we were to fit it to a very (say, infinitely) large training dataset. Thus, in this case, the linear model suffers from large bias, and underfits (i.e., fails to capture structure exhibited by) the data.

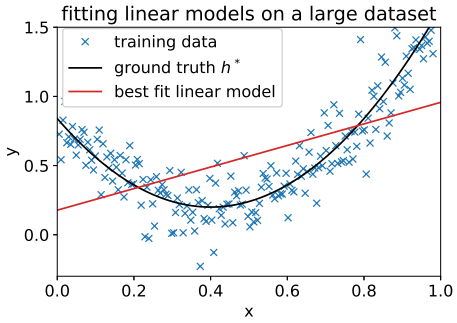

Figure 8.3: The best fit linear model on a much larger dataset still has a large training error.

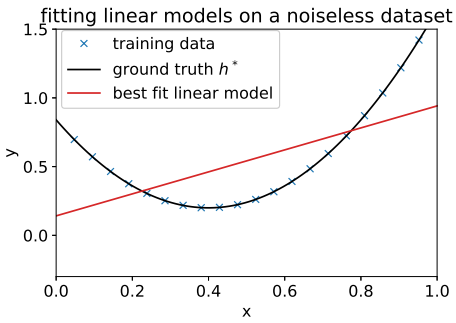

Figure 8.4: The best fit linear model on a noiseless dataset also has a large training/test error.

Next, we fit a 5th-degree polynomial to the data. Figure 8.5 shows that it fails to learn a good model either. However, the failure pattern is different from the linear model case. Specifically, even though the learnt 5th-degree

polynomial did a very good job predicting $y^{(i)}$'s from $x^{(i)}$'s for training examples, it does not work well on test examples (Figure 8.5). In other words, the model learnt from the training set does not *generalize* well to other test examples—the test error is high. Contrary to the behavior of linear models, the bias of the 5-th degree polynomials is small—if we were to fit a 5-th degree polynomial to an extremely large dataset, the resulting model would be close to a quadratic function and be accurate (Figure 8.6). This is because the family of 5-th degree polynomials contains all the quadratic functions (setting $\theta_5 = \theta_4 = \theta_3 = 0$ results in a quadratic function), and, therefore, 5-th degree polynomials are in principle capable of capturing the structure of the data.

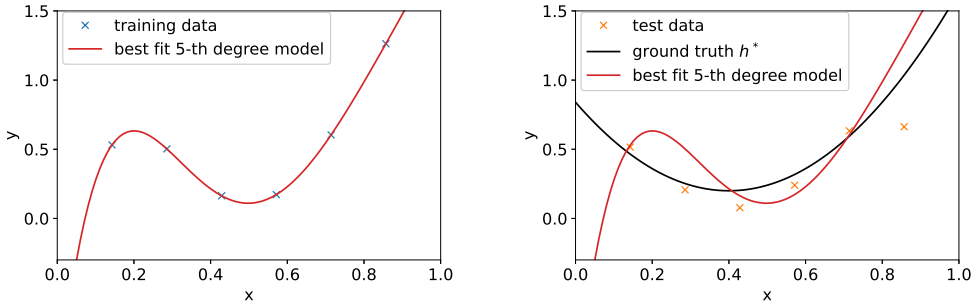

Figure 8.5: Best fit 5-th degree polynomial has zero training error, but still has a large test error and does not recover the the ground truth. This is a classic situation of overfitting.

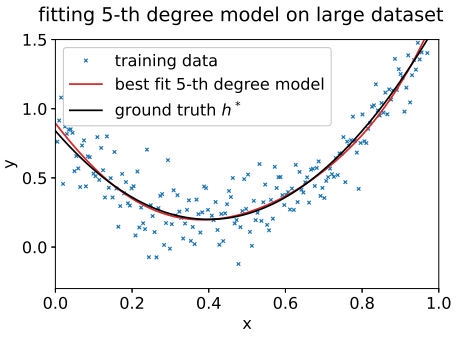

Figure 8.6: The best fit 5-th degree polynomial on a huge dataset nearly recovers the ground-truth—suggesting that the culprit in Figure 8.5 is the variance (or lack of data) but not bias.

The failure of fitting 5-th degree polynomials can be captured by another

component of the test error, called **variance** of a model fitting procedure. Specifically, when fitting a 5-th degree polynomial as in Figure 8.7, there is a large risk that we're fitting patterns in the data that happened to be present in our *small, finite* training set, but that do not reflect the wider pattern of the relationship between x and y. These "spurious" patterns in the training set are (mostly) due to the observation noise $\xi^{(i)}$, and fitting these spurious patters results in a model with large test error. In this case, we say the model has a large variance.

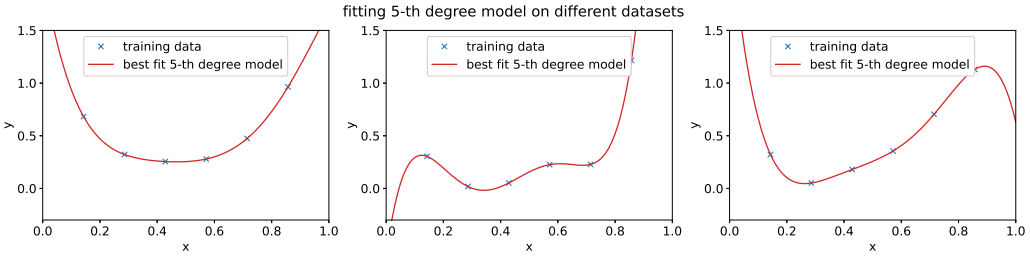

Figure 8.7: The best fit 5-th degree models on three different datasets generated from the same distribution behave quite differently, suggesting the existence of a large variance.

The variance can be intuitively (and mathematically, as shown in Section 8.1.1) characterized by the amount of variations across models learnt on multiple different training datasets (drawn from the same underlying distribution). The "spurious patterns" are specific to the randomness of the noise (and inputs) in a particular dataset, and thus are different across multiple training datasets. Therefore, overfitting to the "spurious patterns" of multiple datasets should result in very different models. Indeed, as shown in Figure 8.7, the models learned on the three different training datasets are quite different, overfitting to the "spurious patterns" of each datasets.

Often, there is a tradeoff between bias and variance. If our model is too "simple" and has very few parameters, then it may have large bias (but small variance), and it typically may suffer from underfittng. If it is too "complex" and has very many parameters, then it may suffer from large variance (but have smaller bias), and thus overfitting. See Figure 8.8 for a typical tradeoff between bias and variance.

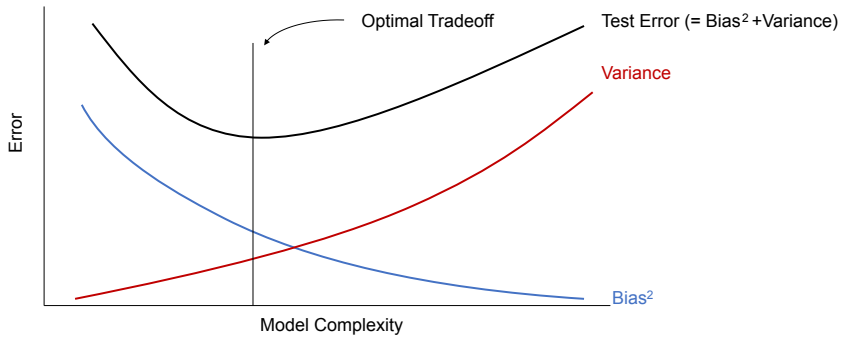

Figure 8.8: An illustration of the typical bias-variance tradeoff.

As we will see formally in Section 8.1.1, the test error can be decomposed as a summation of bias and variance. This means that the test error will have a convex curve as the model complexity increases, and in practice we should tune the model complexity to achieve the best tradeoff. For instance, in the example above, fitting a quadratic function does better than either of the extremes of a first or a 5-th degree polynomial, as shown in Figure 8.9.

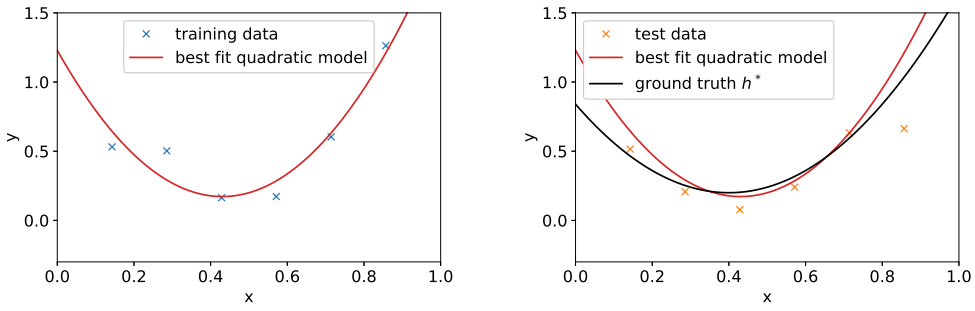

Figure 8.9: Best fit quadratic model has small training and test error because quadratic model achieves a better tradeoff.

Interestingly, the bias-variance tradeoff curves or the test error curves do not universally follow the shape in Figure 8.8, at least not universally when the model complexity is simply measured by the number of parameters. (We will discuss the so-called double descent phenomenon in Section 8.2.) Nevertheless, the principle of bias-variance tradeoff is perhaps still the first resort when analyzing and predicting the behavior of test errors.

8.1.1 A mathematical decomposition (for regression)

To formally state the bias-variance tradeoff for regression problems, we consider the following setup (which is an extension of the beginning paragraph of Section 8.1).

- Draw a training dataset $S = \{x^{(i)}, y^{(i)}\}_{i=1}^{n}$ such that $y^{(i)} = h^\star(x^{(i)}) + \xi^{(i)}$ where $\xi^{(i)} \in N(0, \sigma^2)$.
- Train a model on the dataset S, denoted by $\hat{h}_S$.
- Take a test example (x, y) such that $y = h^\star(x) + \xi$ where $\xi \sim N(0, \sigma^2)$, and measure the expected test error (averaged over the random draw of the training set S and the randomness of ξ)[5][6]

$$\mathrm{MSE}(x) = \mathbb{E}_{S,\xi}[(y - h_S(x))^2] \tag{8.2}$$

We will decompose the MSE into a bias and variance term. We start by stating a following simple mathematical tool that will be used twice below.

Claim 8.1.1: Suppose A and B are two independent real random variables and $\mathbb{E}[A] = 0$. Then, $\mathbb{E}[(A + B)^2] = \mathbb{E}[A^2] + \mathbb{E}[B^2]$.

As a corollary, because a random variable A is independent with a constant c, when $\mathbb{E}[A] = 0$, we have $\mathbb{E}[(A + c)^2] = \mathbb{E}[A^2] + c^2$.

The proof of the claim follows from expanding the square: $\mathbb{E}[(A+B)^2] = \mathbb{E}[A^2] + \mathbb{E}[B^2] + 2\mathbb{E}[AB] = \mathbb{E}[A^2] + \mathbb{E}[B^2]$. Here we used the independence to show that $\mathbb{E}[AB] = \mathbb{E}[A]\mathbb{E}[B] = 0$.

Using Claim 8.1.1 with $A = \xi$ and $B = h^\star(x) - \hat{h}_S(x)$, we have

$$\begin{aligned}
\mathrm{MSE}(x) = \mathbb{E}[(y - h_S(x))^2] &= \mathbb{E}[(\xi + (h^\star(x) - h_S(x)))^2] && (8.3)\\
&= \mathbb{E}[\xi^2] + \mathbb{E}[(h^\star(x) - h_S(x))^2] \quad \text{(by Claim 8.1.1)}\\
&= \sigma^2 + \mathbb{E}[(h^\star(x) - h_S(x))^2] && (8.4)
\end{aligned}$$

Then, let's define $h_{\mathrm{avg}}(x) = \mathbb{E}_S[h_S(x)]$ as the "average model"—the model obtained by drawing an infinite number of datasets, training on them, and averaging their predictions on x. Note that h_{avg} is a hypothetical model for analytical purposes that can not be obtained in reality (because we don't

[5] For simplicity, the test input x is considered to be fixed here, but the same conceptual message holds when we average over the choice of x's.

[6] The subscript under the expectation symbol is to emphasize the variables that are considered as random by the expectation operation.

have infinite number of datasets). It turns out that for many cases, h_{avg} is (approximately) equal to the the model obtained by training on a *single* dataset with infinite samples. Thus, we can also intuitively interpret h_{avg} this way, which is consistent with our intuitive definition of bias in the previous subsection.

We can further decompose $\text{MSE}(x)$ by letting $c = h^\star(x) - h_{\text{avg}}(x)$ (which is a constant that does not depend on the choice of S!) and $A = h_{\text{avg}}(x) - h_S(x)$ in the corollary part of Claim 8.1.1:

$$\begin{aligned}
\text{MSE}(x) &= \sigma^2 + \mathbb{E}[(h^\star(x) - h_S(x))^2] && (8.5)\\
&= \sigma^2 + (h^\star(x) - h_{\text{avg}}(x))^2 + \mathbb{E}[(h_{\text{avg}} - h_S(x))^2] && (8.6)\\
&= \underbrace{\sigma^2}_{\text{unavoidable}} + \underbrace{(h^\star(x) - h_{\text{avg}}(x))^2}_{\triangleq \text{ bias}^2} + \underbrace{\text{var}(h_S(x))}_{\triangleq \text{ variance}} && (8.7)
\end{aligned}$$

We call the second term the bias (square) and the third term the variance. As discussed before, the bias captures the part of the error that are introduced due to the lack of expressivity of the model. Recall that h_{avg} can be thought of as the best possible model learned even with infinite data. Thus, the bias is not due to the lack of data, but is rather caused by that the family of models fundamentally cannot approximate the $h^\star$. For example, in the illustrating example in Figure 8.2, because any linear model cannot approximate the true quadratic function $h^\star$, neither can h_{avg}, and thus the bias term has to be large.

The variance term captures how the random nature of the finite dataset introduces errors in the learned model. It measures the sensitivity of the learned model to the randomness in the dataset. It often decreases as the size of the dataset increases.

There is nothing we can do about the first term σ^2 as we can not predict the noise ξ by definition.

Finally, we note that the bias-variance decomposition for classification is much less clear than for regression problems. There have been several proposals, but there is as yet no agreement on what is the "right" and/or the most useful formalism.

8.2 The double descent phenomenon

Model-wise double descent. Recent works have demonstrated that the test error can present a "double descent" phenomenon in a range of machine