Formalize Bias, Variance, and Noise

Given a data set $D = \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)\}$, with each sample drawn *independently and identically distributed* (i.i.d.) from some data distribution $P = \Pr(\mathbf{x}, y)$, and assuming a regression setting (i.e. $y \in \mathbb{R}$), let us decompose the test error of a classifier into three interpretable terms: bias, variance, and noise. We first define several terms then formally state the bias-variance decomposition.

> ☆ **Key Points**
>
> A classifier's error can be decomposed into three parts: bias, variance, and noise.
>
> Noise refers to intrinsic variation in the labels that even the best classifier cannot capture, either because the features are not enough to explain the labels, or due to incorrect labels.
>
> Variance tells us whether our model is overfitting to training sets, while bias tells us whether our model is making incorrect assumptions about the data distribution (underfitting).

Expected Label (given data $\mathbf{x} \in \mathbb{R}^d$)

First, let us consider that for any given input $\mathbf{x}$, there might not exist a unique label y. For example, if your vector $\mathbf{x}$ describes features of a house (e.g. number of bedrooms, square footage) and the label y its price, you could imagine two houses with identical descriptions selling for different prices. So, for any given feature vector $\mathbf{x}$, there is a distribution over possible labels. We therefore define the "Expected label", which will be useful later on:

$$\bar{y}(\mathbf{x}) = \mathbb{E}_{Y \sim \Pr(y|\mathbf{x})}[Y] = \int_y y \cdot \Pr(y|\mathbf{x}) \; \partial y$$

The expected label denotes the label you would *expect* to obtain, given a feature vector $\mathbf{x}$. In this equation, $\mathbb{E}_{Y \sim \Pr(y|\mathbf{x})}[Y]$ is the expectation of the random variable Y, which is drawn from the distribution $\Pr(y|\mathbf{x})$. The integral is just the definition of the expectation function. We will suppress the notation to $\mathbb{E}_{Y \sim y|\mathbf{x}}[Y]$ going forward for simplicity.

Generalization Error or Expected Test Error (given classifier h_D)

Now, imagine that we have a machine learning algorithm $\mathcal{A}$ trained on this data set to learn a hypothesis (a.k.a. classifier). Formally, we denote this process as $h_D = \mathcal{A}(D)$.

The generalization error $\epsilon(h)$ is defined as the expected loss of a hypothesis h on data sampled according to the real data distribution. That is,

$$\epsilon(h) = \mathbb{E}_{(\mathbf{x},y) \sim P}[\ell(h(\mathbf{x}), y)]$$

For a given h_D (learned on data set D with algorithm $\mathcal{A}$), we can compute the generalization error (as measured in squared loss) as follows:

$$\epsilon(h_D) = \mathbb{E}_{(\mathbf{x},y) \sim P}\left[(h_D(\mathbf{x}) - y)^2\right] = \int_{\mathbf{x}} \int_y (h_D(\mathbf{x}) - y)^2 \cdot \Pr(\mathbf{x}, y) \; \partial y \partial \mathbf{x}$$

We use squared loss here because it has nice mathematical properties and because it is the most common loss function.

Expected Hypothesis a.k.a. Classifier (given learning algorithm $\mathcal{A}$)

Now, remember that $\boldsymbol{D}$ itself is drawn from $\boldsymbol{P^n}$ and is therefore a random variable. Furthermore, $\boldsymbol{h_D}$ is a function of $\boldsymbol{D}$ and is therefore also a random variable, which means that we can compute its expectation:

$$\bar{h} = \mathbb{E}_{D \sim P^n}\left[h_D\right] = \int_D h_D \cdot \Pr(D)\, \partial D$$

where $\Pr(D)$ is the probability of drawing $\boldsymbol{D}$ from $\boldsymbol{P^n}$. You can think of $\bar{h}$ as a weighted average over all possible hypothesis $\boldsymbol{h_D}$, where the weights are $\Pr(D)$.

Generalization Error of the algorithm (given learning algorithm $\mathcal{A}$)

We previously computed the generalization error of a single hypothesis function h_D. We can go a step further, combine the equations in Expected Hypothesis and Generalization Error, and compute the generalization error $\epsilon\left(\bar{h}\right)$ of the expected hypothesis, thus only assuming a learning algorithm $\mathcal{A}$.

$$\begin{aligned}
\epsilon\left(\bar{h}\right) &= \mathbb{E}_{(\mathbf{x},y)\sim P}\left[\left(\bar{h}(\mathbf{x}) - y\right)^2\right] \\
&= \mathbb{E}_{(\mathbf{x},y)\sim P}\mathbb{E}_{D\sim P^n}\left[\left(h_D(\mathbf{x}) - y\right)^2\right] \\
&= \int_{\mathbf{x}}\int_{y}\int_{D} \left(h_D(\mathbf{x}) - y\right)^2 \cdot \Pr(\mathbf{x}, y) \cdot \Pr(D)\, \partial D \partial y \partial \mathbf{x}
\end{aligned}$$

To be clear, $\boldsymbol{D}$ represents our training points that the algorithm $\mathcal{A}$ trains on and the $(\mathbf{x}, y)$ pairs are the test points. We are interested in exactly this expression, because it evaluates the quality of a machine learning algorithm $\mathcal{A}$ on the data distribution $\Pr(\mathbf{x}, y)$.

Bias-Variance Decomposition

With a few standard operations, we can decompose the generalization error of the algorithm as defined above into three meaningful terms:

$$\begin{aligned}
\epsilon\left(\bar{h}\right) &= \underbrace{\mathbb{E}_{(\mathbf{x},y),D}\left[\left(h_D(\mathbf{x}) - y\right)^2\right]}_{\text{Generalization Error of } \mathcal{A}} \\
&= \underbrace{\mathbb{E}_{\mathbf{x},D}\left[\left(h_D(\mathbf{x}) - \bar{h}(\mathbf{x})\right)^2\right]}_{\text{Variance}} + \underbrace{\mathbb{E}_{\mathbf{x}}\left[\left(\bar{h}(\mathbf{x}) - \bar{y}(\mathbf{x})\right)^2\right]}_{\text{Bias}^2} + \underbrace{\mathbb{E}_{\mathbf{x},y}\left[\left(\bar{y}(\mathbf{x}) - y\right)^2\right]}_{\text{Noise}}
\end{aligned}$$

Variance: Captures how much your hypothesis function changes if you train on a different training set. How "overspecialized" is your hypothesis to a particular training set? If we have the best possible model for our training data, how far off are we from the average hypothesis?

Bias: What is the inherent error that you obtain from your hypothesis function even with infinite training data, i.e., from your average hypothesis? This is due to your hypothesis function being "biased" to a particular kind of solution (e.g. linear classifier). In other words, bias is inherent to your model.

Noise: How large is the data-intrinsic noise? This error measures ambiguity due to your data distribution and feature representation. You can never reduce it algorithmically; it is an inherent aspect of the data. You might, however, be able to add more features that capture this seemingly random variability.