Applied Machine Learning

Bias-Variance Decomposition

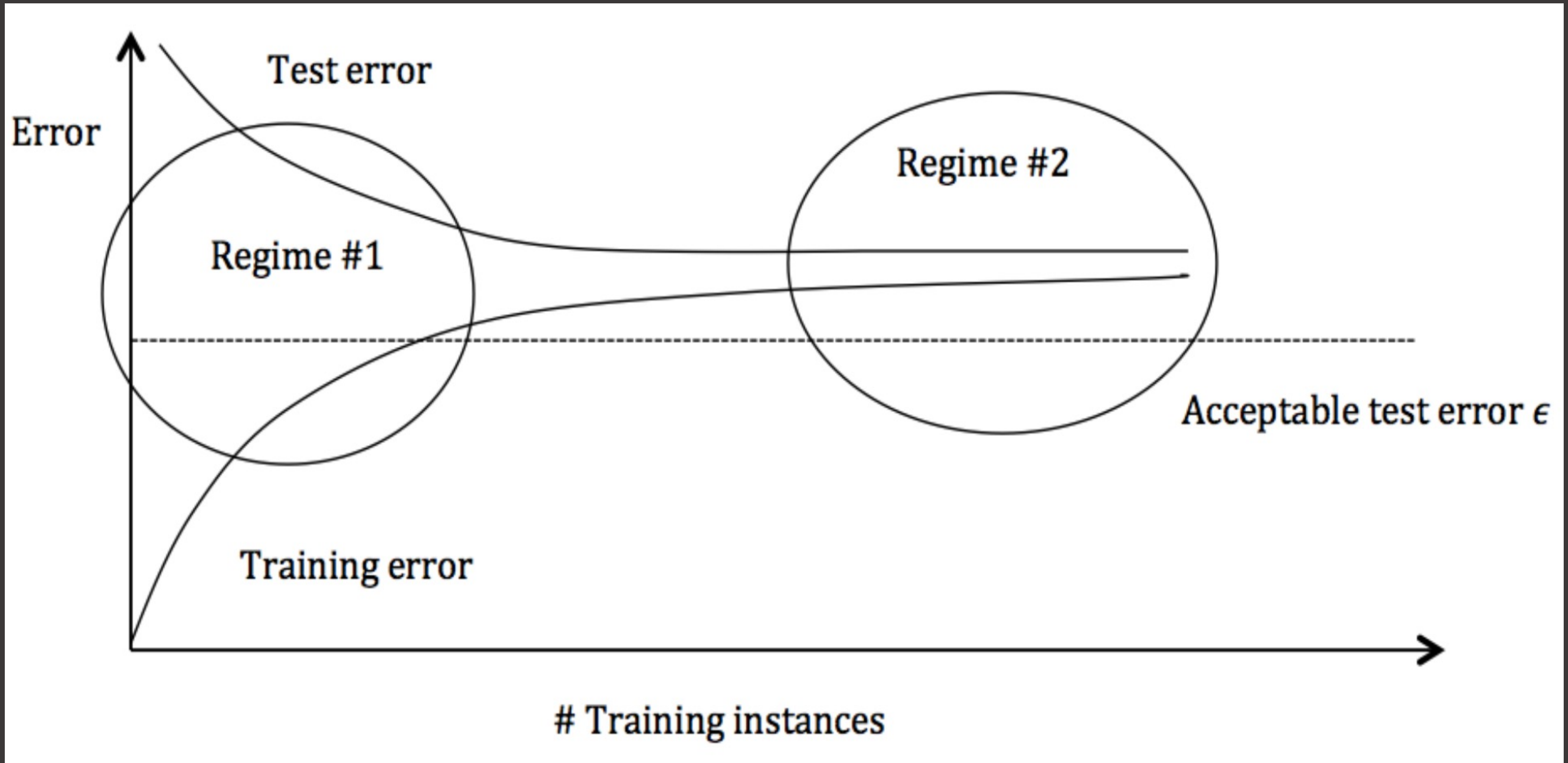
Computer Science, Fall 2022
Instructor: Xuhong Zhang

Bias and Variance

- When we talk about model performance, we might say that “overfitting” or “underfitting”

- Why we have “overfitting” and why we have “underfitting”?
  - Overfitting --- high variance
  - Underfitting --- high bias

Detecting High Bias and High Variance



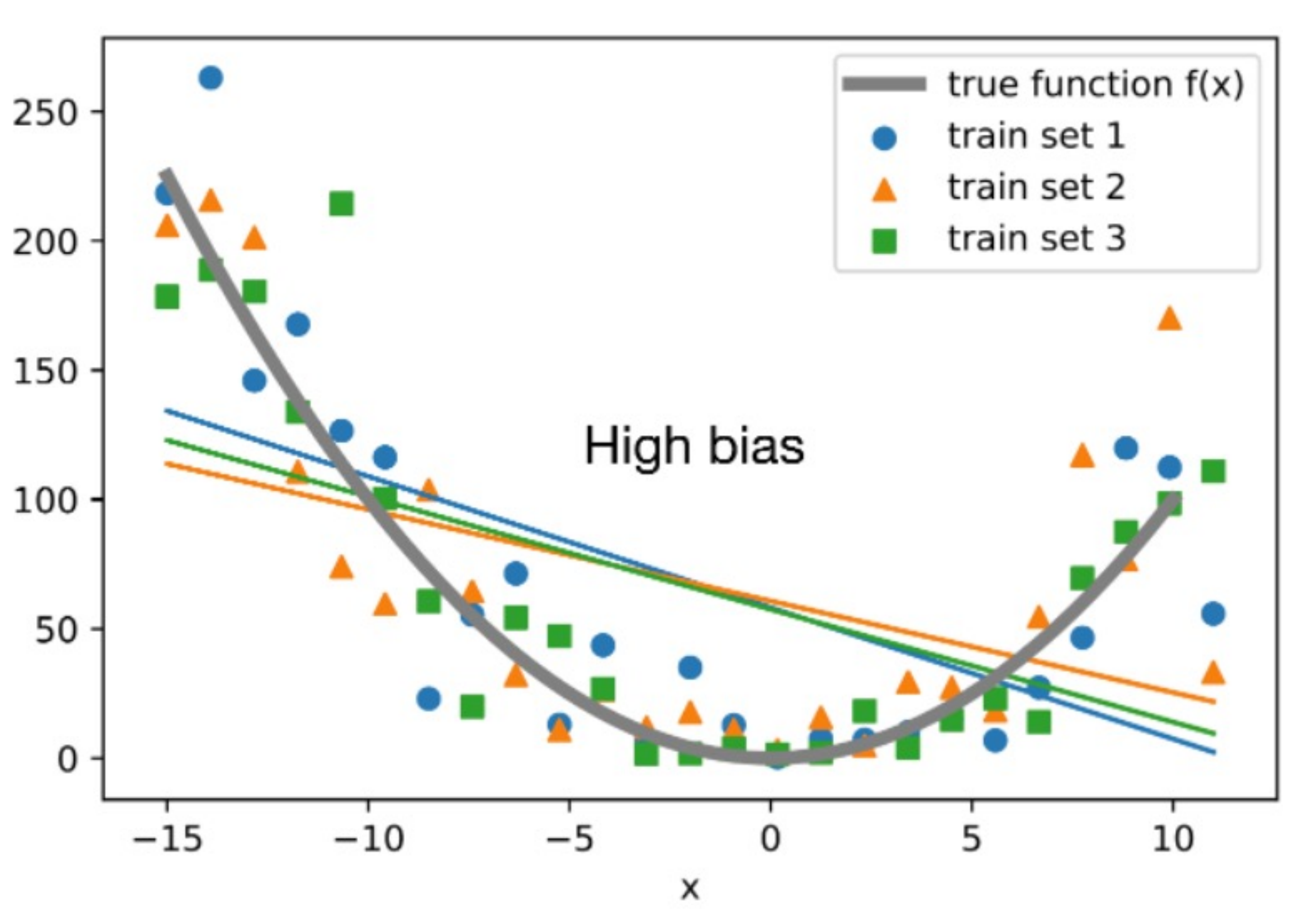
Test and training error as the number of training instances increases.

Detecting High Bias and High Variance

- Let's assume we have a point estimator $\hat{\theta}$ for some function.
- The bias is commonly defined as the difference between the expected value of the estimator and the true parameter:

$$Bias = E\left[\hat{\theta}\right] - \theta$$

Detecting High Bias and High Variance



Detecting High Bias and High Variance

- If the bias is larger than zero, we say that the estimator is positively biased.
- If the bias is smaller than zero, the estimator is negatively biased.
- If the bias is exactly zero, the estimator is unbiased.

Detecting High Bias and High Variance

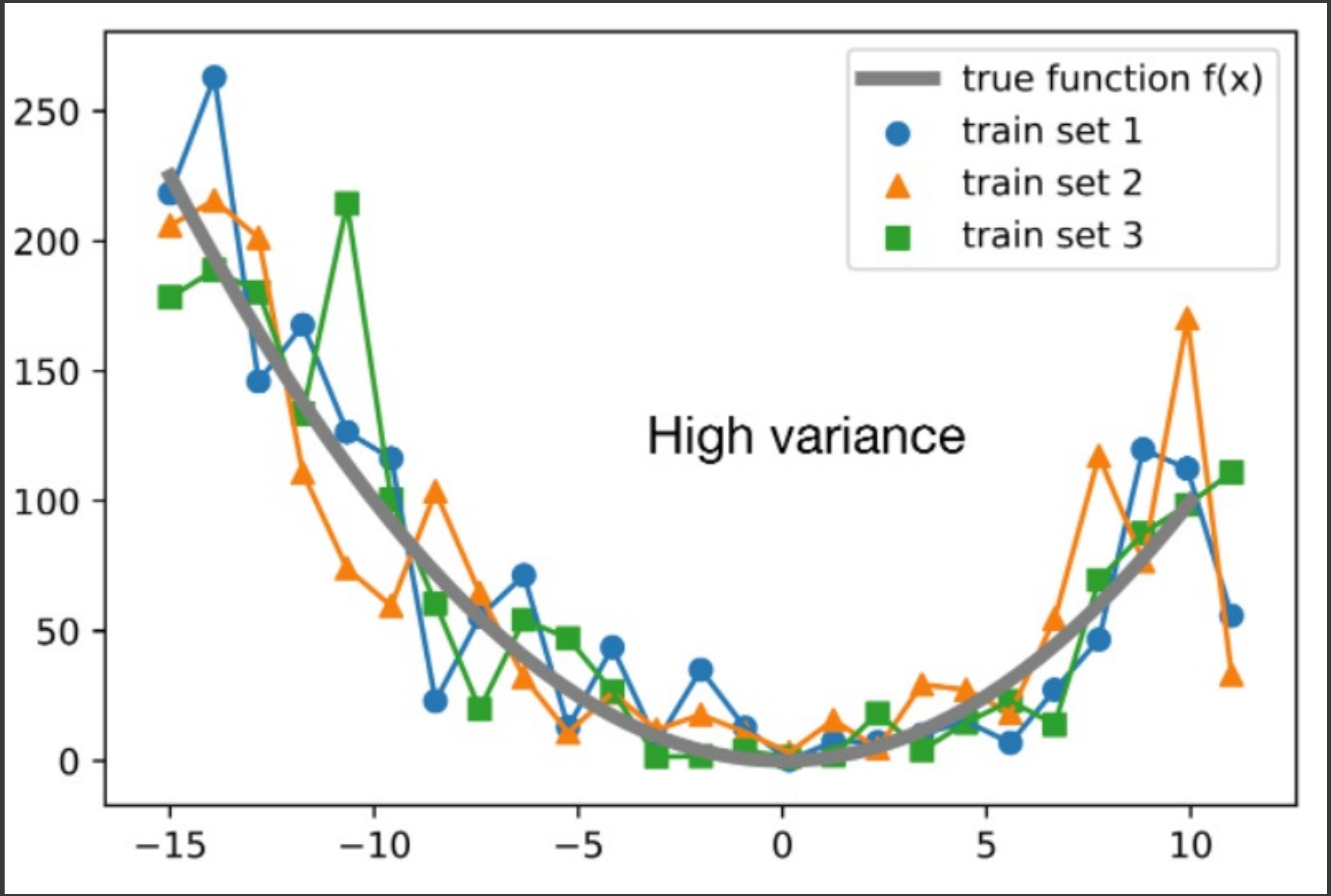
- The variance is defined as the difference between the expected value of the squared estimator minus the squared expectation of the estimator.

$$Var(\hat{\theta}) = E[\hat{\theta}^2] - (E[\hat{\theta}])^2$$

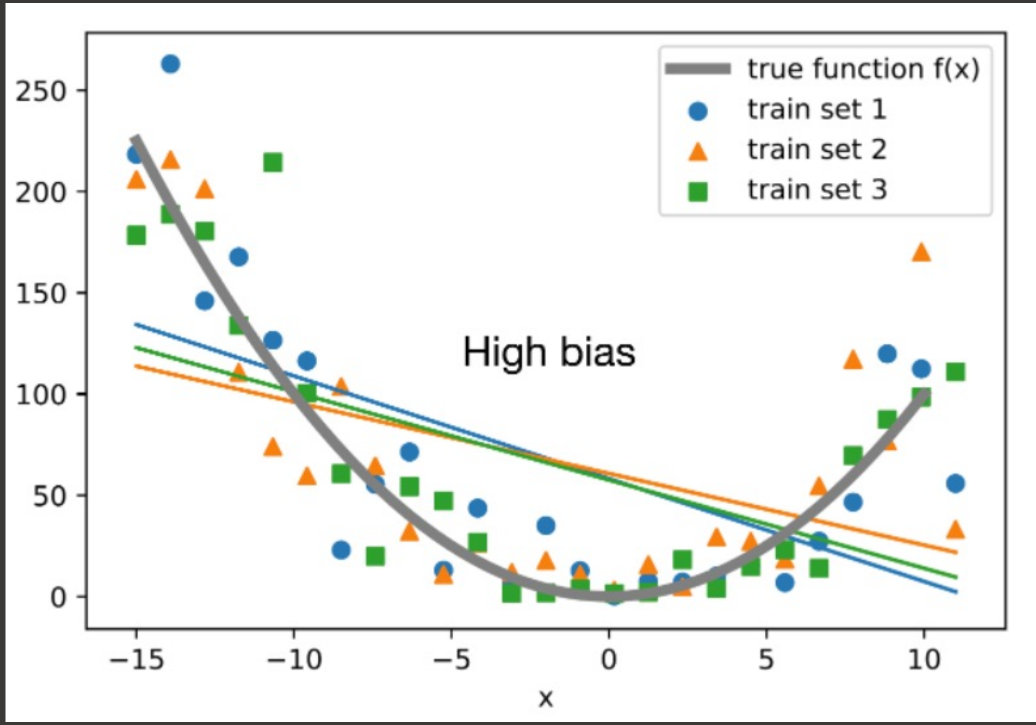
- Alternatively, it is more convenient to use another form:

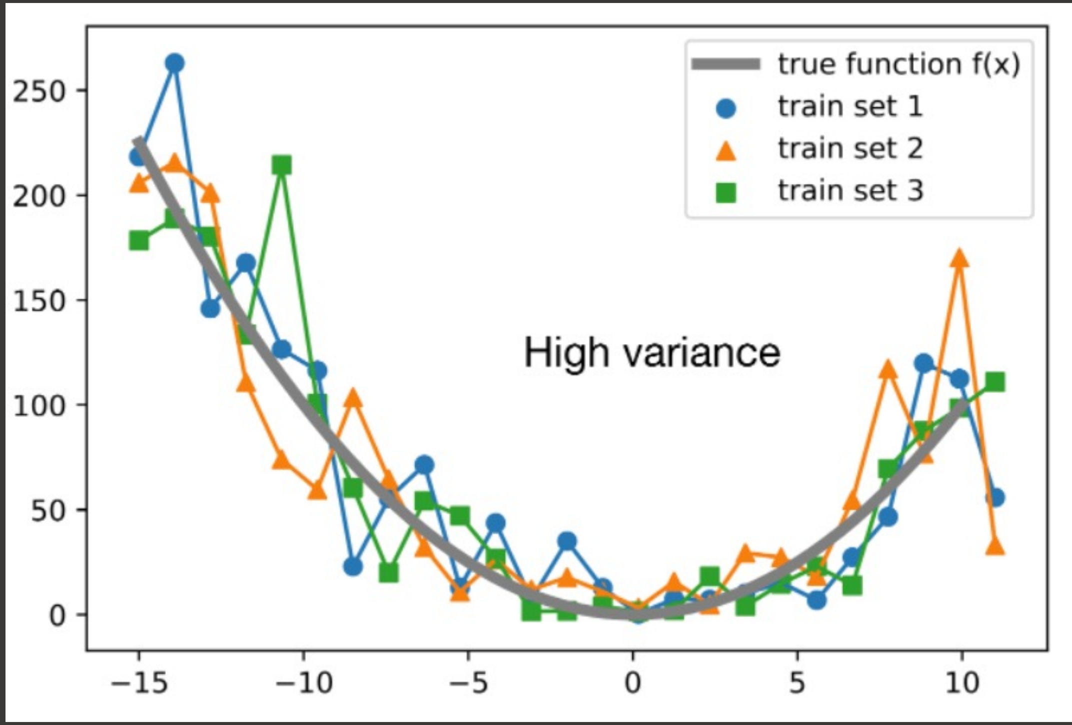
$$Var(\hat{\theta}) = E[(E[\hat{\theta}] - \hat{\theta})^2]$$

Detecting High Bias and High Variance



Detecting High Bias and High Variance





Detecting High Bias and High Variance

- Before we can decompose the error function, we need to do some simple algebraic manipulation

  - Adding and subtracting
  - Expanding an expression using the quadratic formula

$$(a+b)^2 = a^2 + b^2 + 2ab$$

- And we also need to get some sense that everything can be thought as a random variable

Setting

- Suppose we have dataset, as usual, $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$, drawn i.i.d. from some unknown distribution $P(X, Y)$. Let's assume a regression setting, i.e. $y \in \mathbb{R}$.

- What we care the most for machine learning, is the generalization error of a classifier.

- Here, we will decompose the generalization error of a classifier into three interpretable terms.

Setting

- Additional setting: for any given input x there might not exist a unique label y.
  - For example, if your vector x represents some features of house (e.g. # bedrooms, square footage,…) and the label y is the price.
  - You can think of identical description selling for different prices.
  - Thus, for any given feature vector x, there is a distribution over possible labels.

Setting

- Given the previous setting, we then can define the following:
  - **<u>Expected label</u>** (given $x \in \mathbb{R}^d$): $\bar{y}(x) = E_{y|x}[Y] = \int_y yP(y|x)\partial y$

The expected label denotes the label you would expect to obtain, given a feature vector x.

Setting

- So we draw our training set D, consisting of n inputs, i.i.d. from the distribution P.

- After we draw our training set D, we typically call some machine learning algorithm $\mathcal{A}$ on this data set to learn a hypothesis (aka classifier). Let's label the classifier as $h_D = \mathcal{A}(D)$.

Setting

- 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:
  - <u>**Expected Test Error**</u> (given h_D):

$$E_{(\mathbf{x},y)\sim P}[(h_D(\mathbf{x}) - y)^2] = \iint_{x\,y} (h_D(\mathbf{x}) - y)^2 P(\mathbf{x}, y)\, \partial y \partial x$$

Note that we can use other loss functions. We use squared loss because it has nice mathematical properties, and it is also the most common loss function.

Setting

- Previously, we are talking about a given training set D.
- Now, we consider that D itself is drawn from P^n, and therefore D is a random variable.
- Further, since h_D is a function of D, and is therefore also a ***random variable***.

Setting

- Thus, we can compute the expectation of h_D:
  - **Expected Classifier** (given $\mathcal{A}$):

$$\bar{h} = E_{D\sim P^n}[h_D] = \int_D h_D P(D) \partial D$$

  Where $P(D)$ is the probability of drawing D from P^n. Here $\bar{h}$ is a weighted average over functions.

Setting

- We can also use the fact that h_D is a random variable to compute the expected test error only given $\mathcal{A}$, taking the expectation also over D.

  - **<u>Expected Test Error</u>** (given $\mathcal{A}$):

$$E_{\substack{(\mathbf{x},y)\sim P \\ D\sim P^n}}\left[(h_D(x)-y)^2\right] = \int_D\int_{\mathbf{x}}\int_y (h_D(\mathbf{x})-y)^2 P(x,y)P(D)\partial\mathbf{x}\partial y\partial D$$

To be clear, D is our training points and the $(\mathbf{x}, y)$ pairs are the test points.

So far, we get...

- **Expected label** (given $x \in \mathbb{R}^d$): $\bar{y}(x) = E_{y|x}[Y] = \int yP(y|x)\partial y$
- **Expected Test Error** (given h_D):

$$E_{(\mathbf{x},y)\sim P}[(h_D(\mathbf{x}) - y)^2] = \iint (h_D(\mathbf{x}) - y)^2 P(\mathbf{x}, y)\, \partial y \partial x$$

- **Expected Classifier** (given $\mathcal{A}$):

$$\bar{h} = E_{D\sim P^n}[h_D] = \int h_D P(D) \partial D$$

- **Expected Test Error** (given $\mathcal{A}$):

$$E_{\substack{(\mathbf{x},y)\sim P \\ D\sim P^n}}[(h_D(x) - y)^2] = \iiint (h_D(\mathbf{x}) - y)^2 P(x,y) P(D) \partial\mathbf{x} \partial y \partial D$$

We are interested in exactly this expression, because it evaluates the quality of a machine learning algorithm $\mathcal{A}$ with respect to a data distribution $P(X, Y)$.

Decomposition of Expected Test Error

$$E_{\mathbf{x},y,D}[[h_D(x) - y]^2]$$

$$= E_{\mathbf{x},y,D}\left[\left[\left(h_D(x) - \bar{h}(x)\right) + (\bar{h}(x) - y)\right]^2\right]$$

$$= \underbrace{E_{\mathbf{x},D}\left[\left(h_D(x) - \bar{h}(x)\right)^2\right]}_{1} + \underbrace{2E_{\mathbf{x},y,D}\left[\left(h_D(x) - \bar{h}(x)\right)\left(\bar{h}(x) - y\right)\right]}_{2} + \underbrace{E_{\mathbf{x},y}[(\bar{h}(x) - y)^2]}_{3}$$

The middle term:

$$\begin{aligned} E_{\mathbf{x},y,D}\left[\left(h_D(x) - \bar{h}(x)\right)\left(\bar{h}(x) - y\right)\right] &= E_{x,y}\left[E_D\left[h_D(x) - \bar{h}(x)\right]\left(\bar{h}(x) - y\right)\right] \\ &= E_{x,y}\left[\left(E_D\left[h_D(x)\right] - \bar{h}(x)\right)\left(\bar{h}(x) - y\right)\right] \\ &= E_{x,y}\left[\left(\bar{h}(x) - \bar{h}(x)\right)\left(\bar{h}(x) - y\right)\right] \\ &= 0 \end{aligned}$$

Decomposition of Expected Test Error

Now returning to the earlier expression, only 1 and 3 are left.

$$E_{\mathbf{x},y,D}[[h_D(x) - y]^2] = E_{\mathbf{x},D}\left[\left(h_D(x) - \bar{h}(x)\right)^2\right] + E_{\mathbf{x},y}[(\bar{h}(x) - y)^2]$$

1 3

Take a look at the 3rd term:

$$E_{\mathbf{x},y}\left[\left(\bar{h}(x) - y\right)^2\right] = E_{\mathbf{x},y}[(\bar{h}(x) - \bar{y}(x) + \bar{y}(x) - y)^2]$$

$$= E_{\mathbf{x}}\left[\left(\bar{h}(x) - \bar{y}(x)\right)^2\right] + E_{\mathbf{x},y}[(\bar{y}(x) - y)^2] + 2E_{\mathbf{x},y}[(\bar{h}(x) - \bar{y}(x))(\bar{y}(x) - y)]$$

4 5

$$\begin{aligned} E_{\mathbf{x},y}\left[\left(\bar{h}(x) - \bar{y}(x)\right)(\bar{y}(x) - y)\right] &= E_x\left[E_{y|x}[\bar{y}(x) - y](\bar{h}(x) - \bar{y}(x))\right] \\ &= E_x\left[(\bar{y}(x) - E_{y|x}[y])(\bar{h}(x) - \bar{y}(x))\right] \\ &= E_x\left[(\bar{y}(x) - \bar{y}(x))(\bar{h}(x) - \bar{y}(x))\right] \\ &= 0 \end{aligned}$$

Finally

$$E_{\mathbf{x},y,D}[(h_D(x)-y)^2] = E_{\mathbf{x},D}\left[\left(h_D(x) \overset{1}{-} \bar{h}(x)\right)^2\right] + E_{\mathbf{x}}\left[\left(\bar{h}(x) \overset{4}{-} \bar{y}(x)\right)^2\right] + E_{\mathbf{x},y}[(\bar{y}(x) \overset{5}{-} y)^2]$$

Expected Test Error

Variance

Bias^2

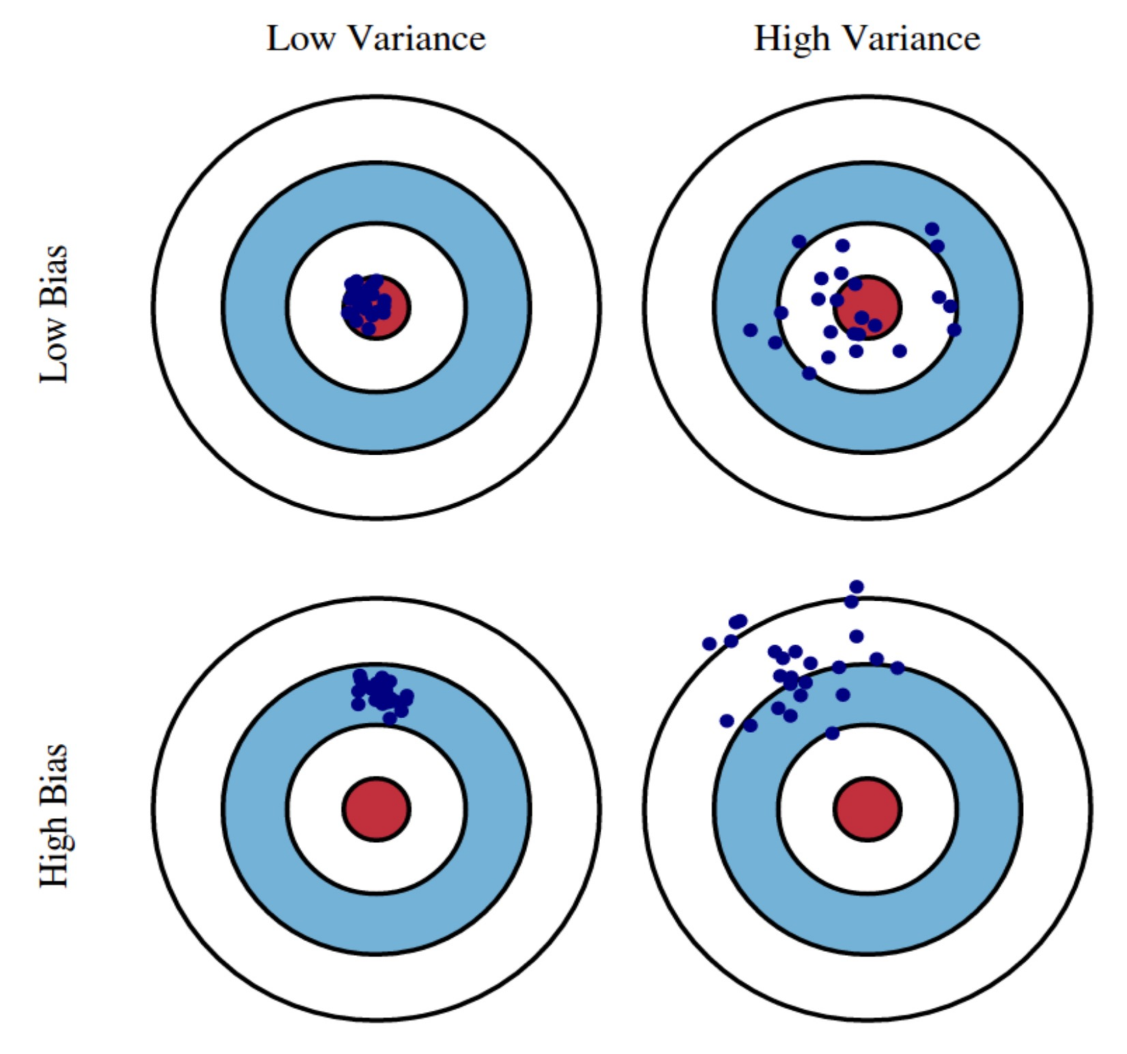
Noise

Variance: Captures how much your classifier changes if you train on a different training set. How "over-specialized" is your classifier to a particular training set (overfitting)? If we have the best possible model for our training data, how far off are we from the average classifier?

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

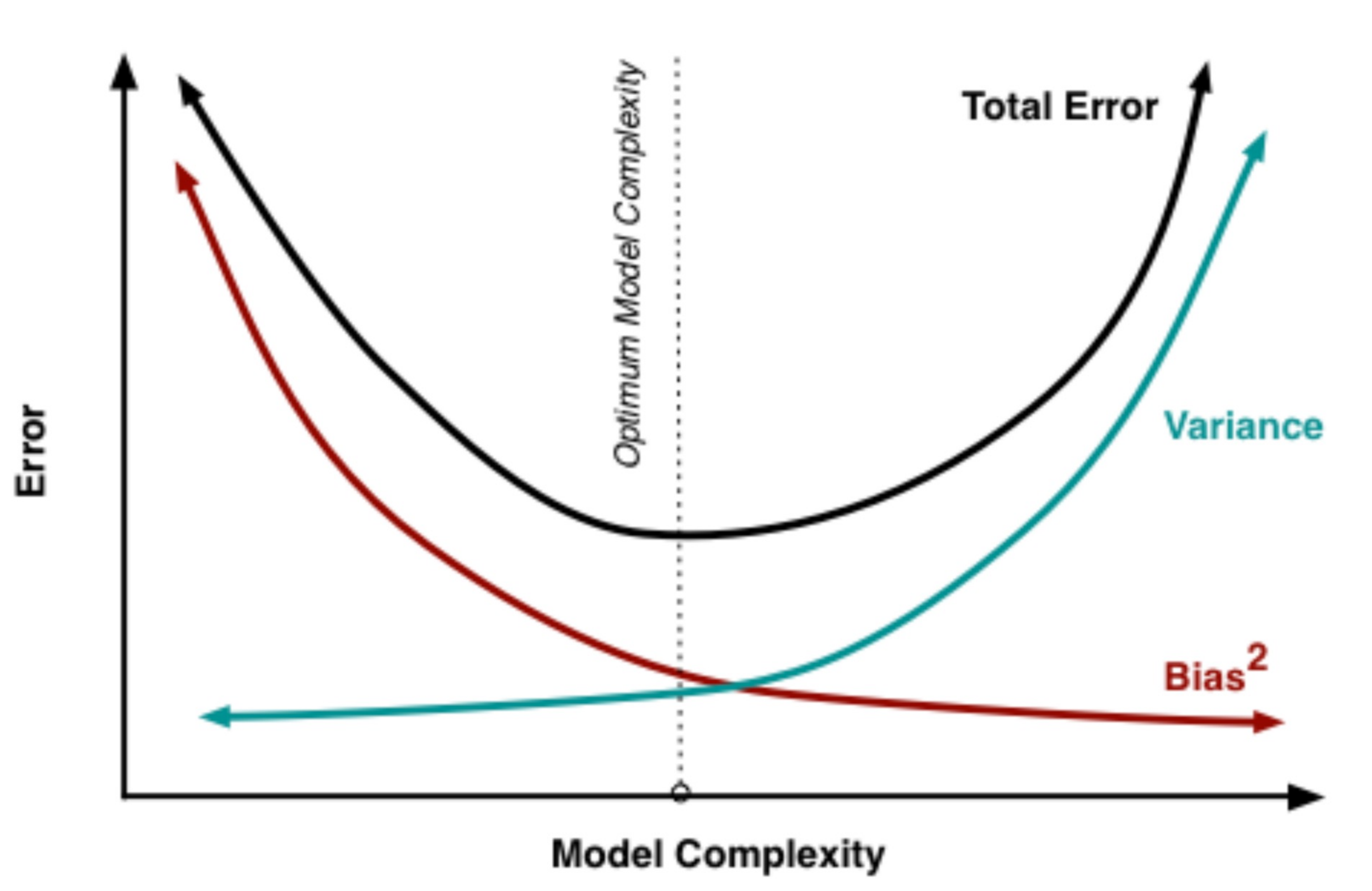
Noise: How big is the data-intrinsic noise? This error measures ambiguity due to your data distribution and feature representation. You can never beat this, it is an aspect of the data.

Graphical illustration of bias and variance



Source: http://scott.fortmann-roe.com/docs/BiasVariance.html

Model Complexity vs. Bias & Variance



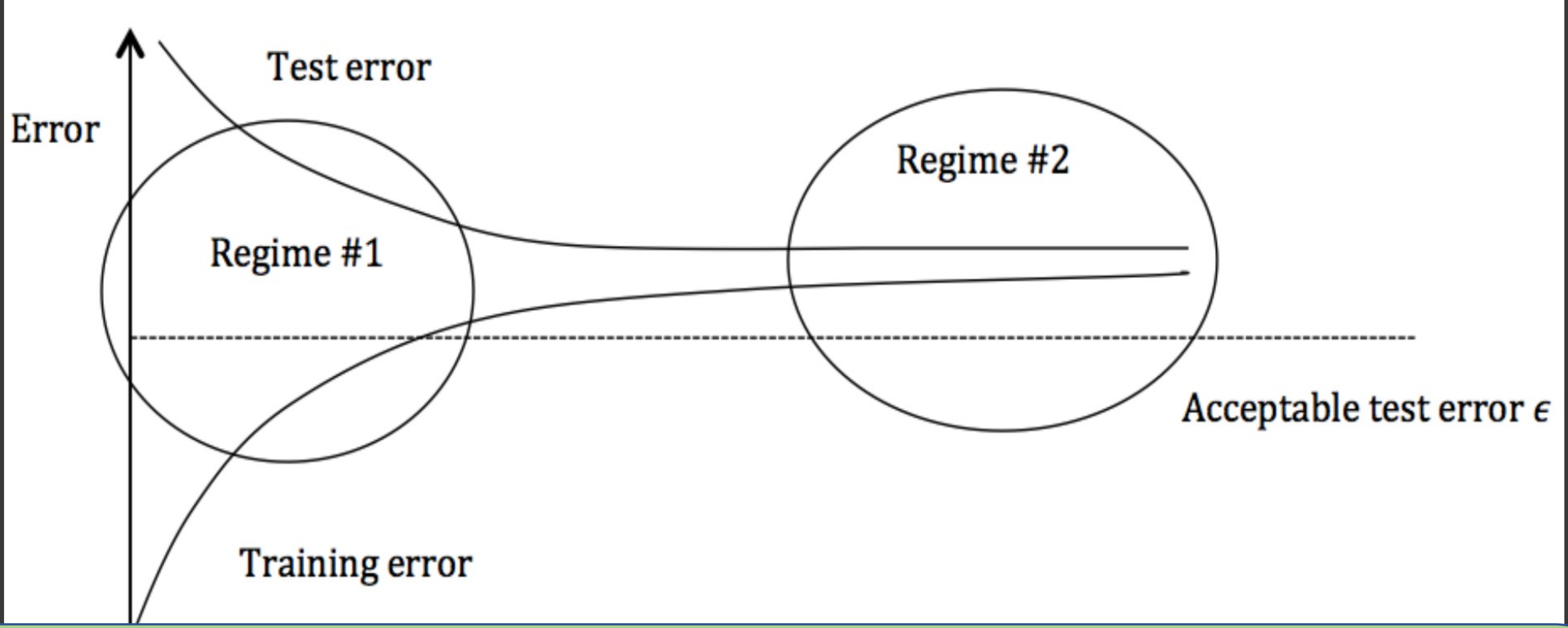
The variation of Bias and Variance with the model complexity. This is similar to the concept of overfitting and underfitting. More complex models overfit while the simplest models underfit.

Source: http://scott.fortmann-roe.com/docs/BiasVariance.html

Detecting High Bias and High Variance

- If a classifier is under-performing (e.g. if the test or training error is too high), there are several ways to improve performance.

- To find out which of these many techniques is the right one for the situation, the first step is to determine the root of the problem.



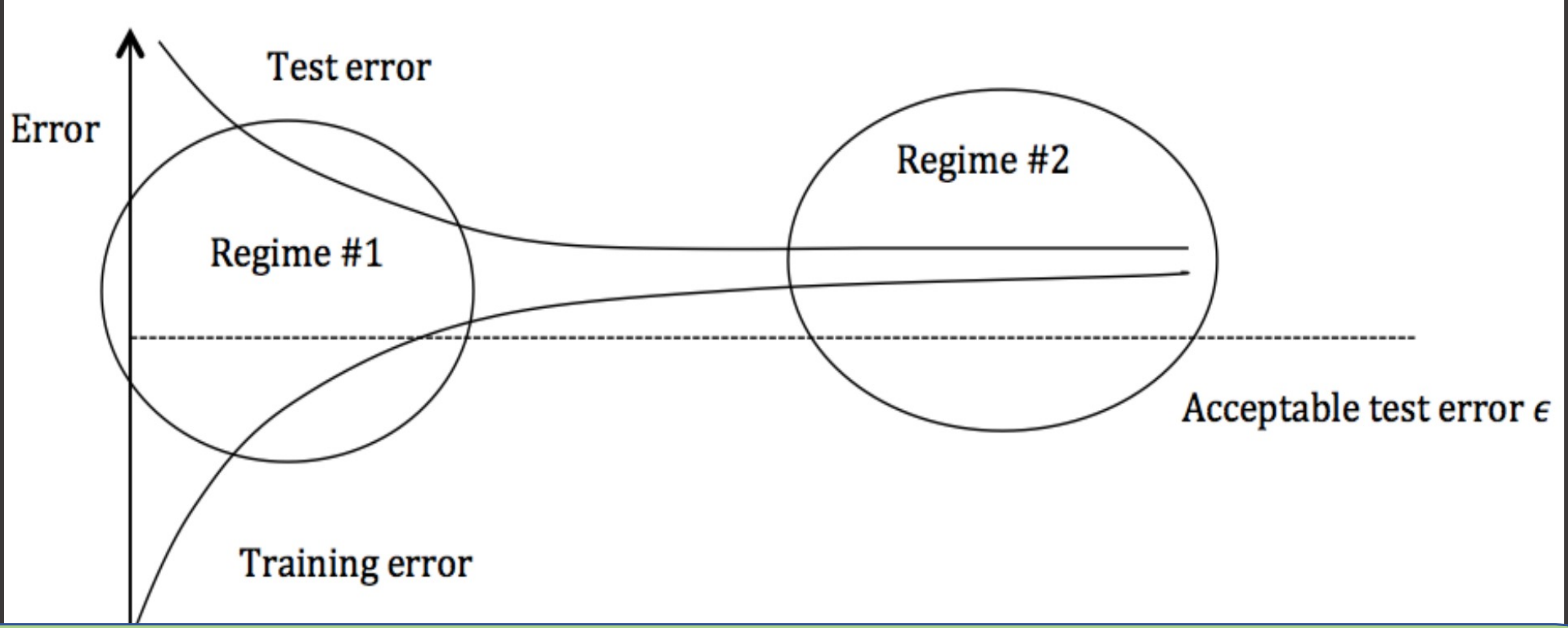
Regime 1 (High Variance): In the first regime, the cause of the poor performance is high variance.

Symptoms:

1. Training error is much lower than test error
2. Training error is lower than ϵ
3. Test error is above ϵ

Remedies:

1. Add more training data
2. Reduce model complexity – complex models are prone to high variance
3. Bagging (will be covered later in the course)



Regime 2 (High Bias): The second regime indicates high bias: the model being used is not robust enough to produce an accurate prediction.

Symptoms:

1. Training error is higher than ϵ

Remedies:

1. Use more complex model (e.g. Kernelize, use non-linear models)
2. Add features
3. Boosting (will be covered later in the course)