Bias-Variance Tradeoff

Many slides adapted from cs4780 lectures in Cornell Univ.

Bias Variance Tradeoff

What is Bias Variance Tradeoff?

- If our model is too simple and has very few parameters then it may have high bias and low variance.
- On the other hand if our model has large number of parameters then it's going to have high variance and low bias.
- So we need to find the right/good balance without overfitting and underfitting the data.
- This tradeoff in complexity is why there is a tradeoff between bias and variance. An algorithm can't be more complex and less complex at the same time.

Total Error

- To build a good model, we need to find a good balance between bias and variance such that it minimizes the total error.
- Understanding bias and variance is critical for understanding the behavior of prediction models.

Decomposition of Test Error

- Suppose $D = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ drawn i.i.d. from some distribution P(X,Y). We will decompose the generalization error of a classifier into three interpretable terms

- For any given input x, there might not exist a unique label y. For example, if your vector x describes features of house (e.g. #bedrooms, square footage, ...) and the label y (its price), you could imagine two houses with identical description selling for different prices.

- So for any given feature vector x, there is a distribution over possible labels. We therefore define the expected label in the following.

$$\bar{y}(\mathbf{x}) = E_{y|\mathbf{x}}[Y] = \int_y y \Pr(y|\mathbf{x}) \partial y.$$

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

.

Decomposition of Test Error

- As a second step we typically call some machine learning algorithm A on this data set D to learn a hypothesis (aka classifier). Formally, we denote this process as $h_D = A(D)$

- For a given h_D, learned on data set D with algorithm A, we can compute the generalization error (as measured in squared loss) as follows.

$$E_{(\mathbf{x},y)\sim P}\left[(h_D(\mathbf{x}) - y)^2\right] = \int_x \int_y (h_D(\mathbf{x}) - y)^2 \Pr(\mathbf{x}, y) \partial y \partial \mathbf{x}.$$

Decomposition of Test Error

- Expected Test Error given A is as follows

$$E_{\mathbf{x},y,D}\left[\left[h_D(\mathbf{x})-y\right]^2\right] = E_{\mathbf{x},y,D}\left[\left[\left(h_D(\mathbf{x})-\bar{h}(\mathbf{x})\right)+\left(\bar{h}(\mathbf{x})-y\right)\right]^2\right]$$
$$= E_{\mathbf{x},D}\left[\left(h_D(\mathbf{x})-\bar{h}(\mathbf{x})\right)^2\right] + 2\,\underbrace{E_{\mathbf{x},y,D}\left[\left(h_D(\mathbf{x})-\bar{h}(\mathbf{x})\right)\left(\bar{h}(\mathbf{x})-y\right)\right]}_{=0} + E_{\mathbf{x},y}\left[\left(\bar{h}(\mathbf{x})-y\right)^2\right]$$

- Here, $\bar{h}$ is a weighted average over functions. (Weighted average of h values)

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

- The middle term of the above equation is 0 as we show below

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

Decomposition of Test Error

- Returning to the earlier expression, we're left with the variance and another term

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

- We can break down the second term in the above equation as follows

$$\begin{aligned} E_{\mathbf{x},y}\left[\left(\bar{h}(\mathbf{x})-y\right)^2\right] &= E_{\mathbf{x},y}\left[\left(\bar{h}(\mathbf{x})-\bar{y}(\mathbf{x})\right)+\left(\bar{y}(\mathbf{x})-y\right)^2\right] \\ &= \underbrace{E_{\mathbf{x},y}\left[\left(\bar{y}(\mathbf{x})-y\right)^2\right]}_{\text{Noise}} + \underbrace{E_{\mathbf{x}}\left[\left(\bar{h}(\mathbf{x})-\bar{y}(\mathbf{x})\right)^2\right]}_{\text{Bias}^2} + \underbrace{2\,E_{\mathbf{x},y}\left[\left(\bar{h}(\mathbf{x})-\bar{y}(\mathbf{x})\right)\left(\bar{y}(\mathbf{x})-y\right)\right]}_{=0} \end{aligned}$$

$$\bar{y}(\mathbf{x}) = E_{y|\mathbf{x}}\left[Y\right] = \int_y y\,\Pr(y|\mathbf{x})\partial y.$$

Decomposition of Test Error

- The third term in the equation above is 0 as we show below

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

- This gives us the decomposition of expected test error as follows

$$
\underbrace{E_{\mathbf{x},y,D}\left[\left(h_D(\mathbf{x})-y\right)^2\right]}_{\text{Expected Test Error}} = \underbrace{E_{\mathbf{x},D}\left[\left(h_D(\mathbf{x})-\bar{h}(\mathbf{x})\right)^2\right]}_{\text{Variance}} + \underbrace{E_{\mathbf{x},y}\left[\left(\bar{y}(\mathbf{x})-y\right)^2\right]}_{\text{Noise}} + \underbrace{E_{\mathbf{x}}\left[\left(\bar{h}(\mathbf{x})-\bar{y}(\mathbf{x})\right)^2\right]}_{\text{Bias}^2}
$$

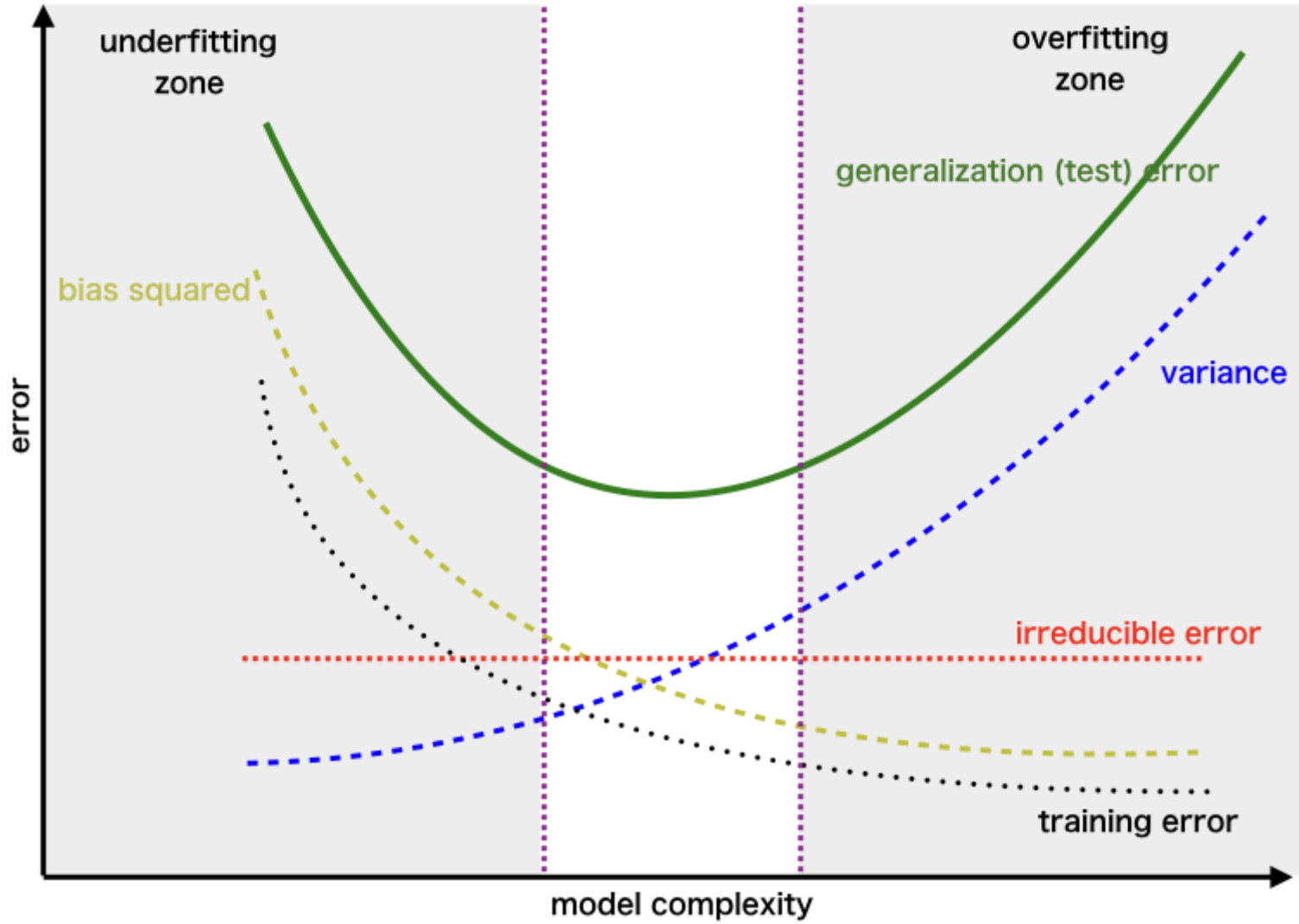
Decomposition of Test Error

$$\underbrace{E_{\mathbf{x},y,D}\left[\left(h_D(\mathbf{x})-y\right)^2\right]}_{\text{Expected Test Error}} = \underbrace{E_{\mathbf{x},D}\left[\left(h_D(\mathbf{x})-\bar{h}(\mathbf{x})\right)^2\right]}_{\text{Variance}} + \underbrace{E_{\mathbf{x},y}\left[\left(\bar{y}(\mathbf{x})-y\right)^2\right]}_{\text{Noise}} + \underbrace{E_{\mathbf{x}}\left[\left(\bar{h}(\mathbf{x})-\bar{y}(\mathbf{x})\right)^2\right]}_{\text{Bias}^2}$$

Variance: Describes how much $h_D(x)$ varies from training set D to another. Variance is the measure of the inconsistency of different predictions over varied datasets. Higher variance is an indication of overfitting in which the model loses the ability to generalize.
(heuristic: cross validation error – training error)

Bias: Describes the average error of $\bar{h}(x)$. Bias is the difference between the average prediction of classifier and the correct value. If the model is oversimplified, then the predicted value would be far from the ground truth resulting in more bias.
(heuristic: training error)

$$\text{Error} = \text{Bias}^2 + \text{Variance} + \text{IrreducibleError}$$

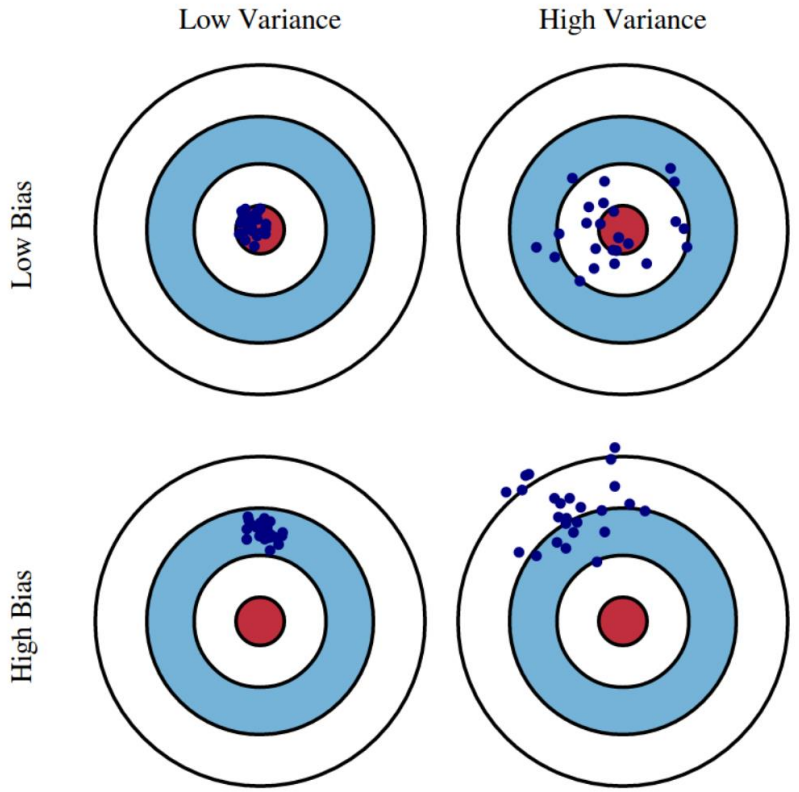


Bias and Variance

Variance:
- Captures how much your classifier changes if you train on a different training set.

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



Variance

- Captures how much your classifier changes if you train on a different training set.
- Variance is the variability of model prediction for a given data point or a value which tells us spread of our data.
- Model with high variance pays a lot of attention to training data and does not generalize on the data which it hasn’t seen before.

- As a result, such models perform very well on training data but has high error rates on test data.

Bias

- Bias is due to your classifier being "biased" to a particular kind of solution (e.g. linear classifier).
- Bias is inherent to your model
  - Low Bias: Suggests less assumptions about the form of the target function.
  - High-Bias: Suggests more assumptions about the form of the target function.

- Bias is the difference between the average prediction of classifier and the correct value.

- It always leads to high error on training and test data.

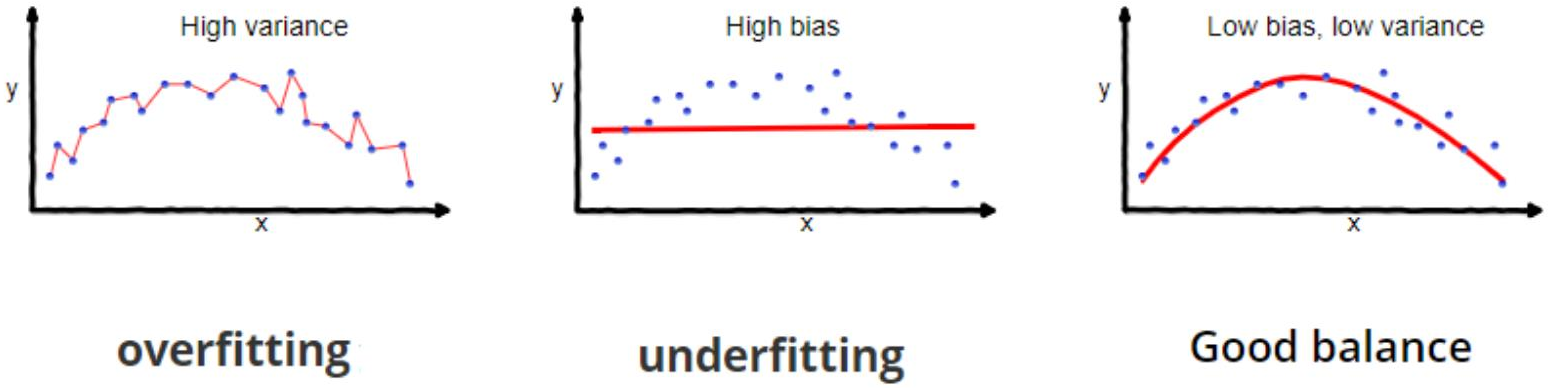
- Generally, linear algorithms have a high bias making them fast to learn but generally less flexible.
- In turn, they have lower predictive performance on complex problems that fail to meet the simplifying assumptions of the algorithms bias.

- Examples of low-bias machine learning algorithms include: Deep learning, Decision Trees, k-Nearest Neighbors and Support Vector Machines.
- Examples of high-bias machine learning algorithms include: Linear Regression and Logistic Regression.

Underfitting and Overfitting

- In supervised learning, **underfitting** happens when a model unable to capture the underlying pattern of the data.
- These models usually have high bias and low variance. It happens when we try to build a linear model (simple model) with a nonlinear data (complex data).

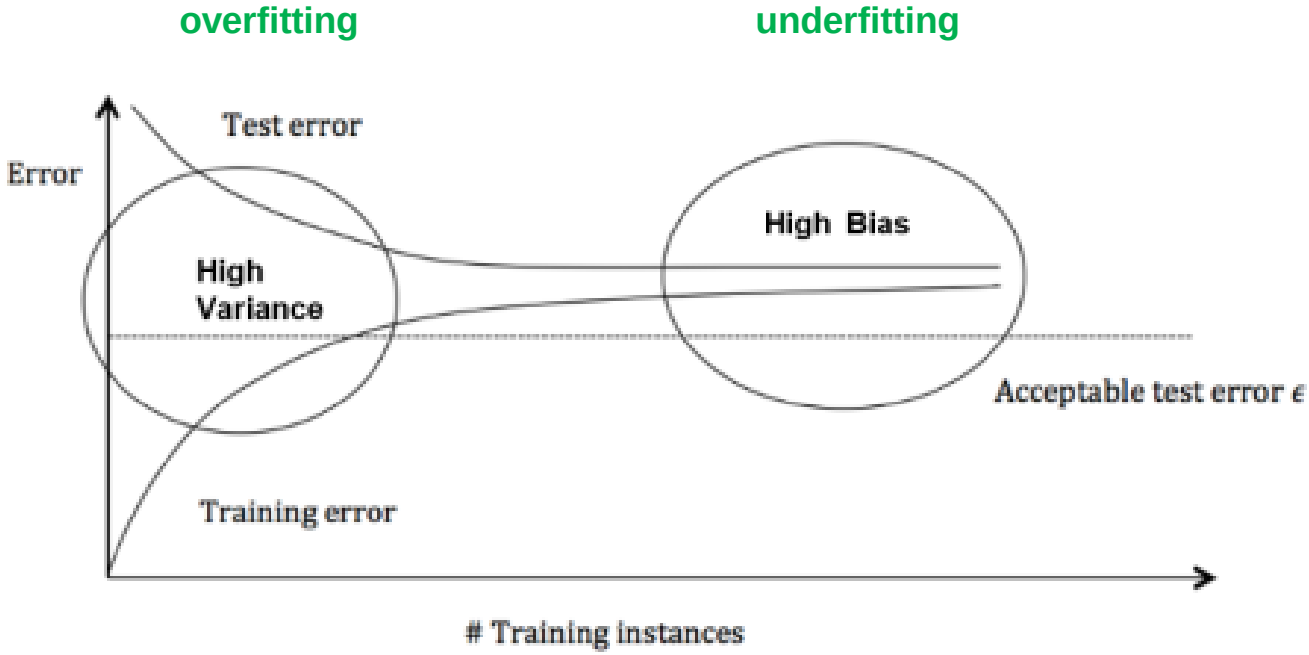
- In supervised learning, **overfitting** happens when our model captures the noise along with the underlying pattern in data.
- It happens when we train our model a lot over noisy dataset. These models have low bias and high variance.



Underfitting and Overfitting

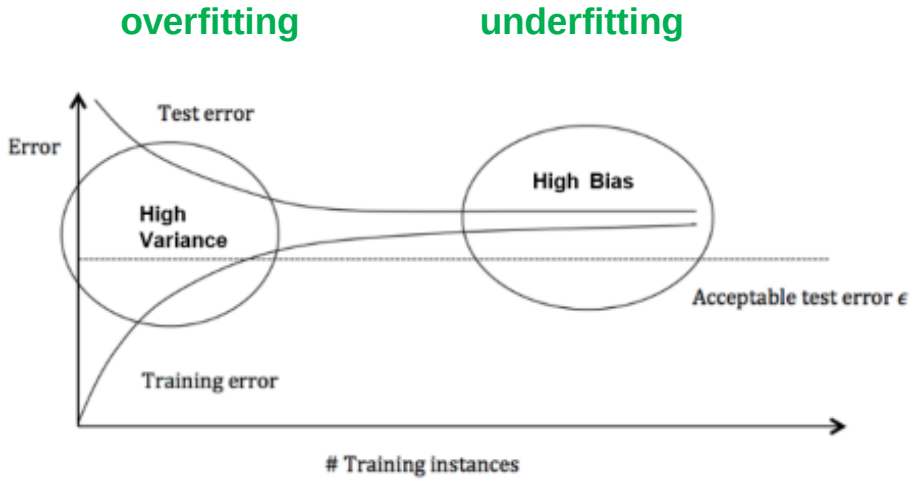
- To find out which of these many techniques is the right one for the situation, the first step is to determine the root of the problem.

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



- The graph above plots the training error and the test error and can be divided into two overarching regimes.
- In the first regime (on the left side of the graph), training error is below the desired error threshold (denoted by ϵ), but test error is significantly higher.
- In the second regime (on the right side of the graph), test error is remarkably close to training error, but both are above the desired tolerance of ϵ

Underfitting and Overfitting



Overfitting (High Variance)
The cause of the poor performance is high variance.

Symptoms:
Training error is much lower than test error
Training error is lower than ϵ and test error is above ϵ

Remedies:
Add more training data
Feature selection
Reduce model complexity -- complex models are prone to high variance
Bagging

Underfitting (High Bias)
The model is not robust enough to produce an accurate prediction.

Symptoms:
Training error is higher than ϵ

Remedies:
Use more complex model (e.g. kernelize, use non-linear models)
Add features
Boosting

Training error	Testing error	Indication
Low	Low	Good job!
Low	High	High variance
High	High	High bias
High	Low	You have a bug

Bagging and Boosting

Regime 1 (High Variance)

- Bagging: deep/large trees are generally employed as base learners. Large trees have high variance, but Low bias.
- Ensembling many large trees reduces the variance.

Regime 2 (High Bias)

- Boosting: Small trees have low variance, but high bias. Averaging over many trees (combined with updating the response variable after fitting each tree, which puts more weight on training observations not well predicted thus far) thus reduces the bias.
- We start with a data sample from the training data. Then we build a very simple underfit model (often tree based). Because the model is simple it means we are underfitting the data and also that means that bias is high and variance is low.

- What boosting does then is evaluating the model to identify wrong predictions, then for the next sample, it corrects the bias error that the previous model got wrong. In other words, the second model will focus on the areas where the previous model struggles to reduce the bias of the overall model.

- We will repeat the process n times where we are sequentially building models that are learning from the mistakes made by the prior models.

Bias and Variance in python – 1/2

```
# estimate the bias and variance for a regression model
from pandas import read_csv
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from mlxtend.evaluate import bias_variance_decomp

# load dataset
url = 'https://raw.githubusercontent.com/jbrownlee/Datasets/master/housing.csv'
dataframe = read_csv(url, header=None)

# separate into inputs and outputs
data = dataframe.values
X, y = data[:, :-1], data[:, -1]
```

Bias and Variance in python – 2/2

```
# split the data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.33, random_state=1)

# define the model
model = LinearRegression()

# estimate bias and variance
mse, bias, var = bias_variance_decomp(model, X_train, y_train, X_test, y_test,
loss='mse', num_rounds=100)

# summarize results
print('MSE: %.3f' % mse)
print('Bias: %.3f' % bias)
print('Variance: %.3f' % var)
```

MSE: 22.487
Bias: 20.726
Variance: 1.761