MACHINE LEARNING BASICS

MACHINE LEARNING BASICS

AGENDA

Overfitting — what is it, how do we combat it

Maximization bias — why we need test data sets to evaluate our model

Stopping criteria for gradient descent

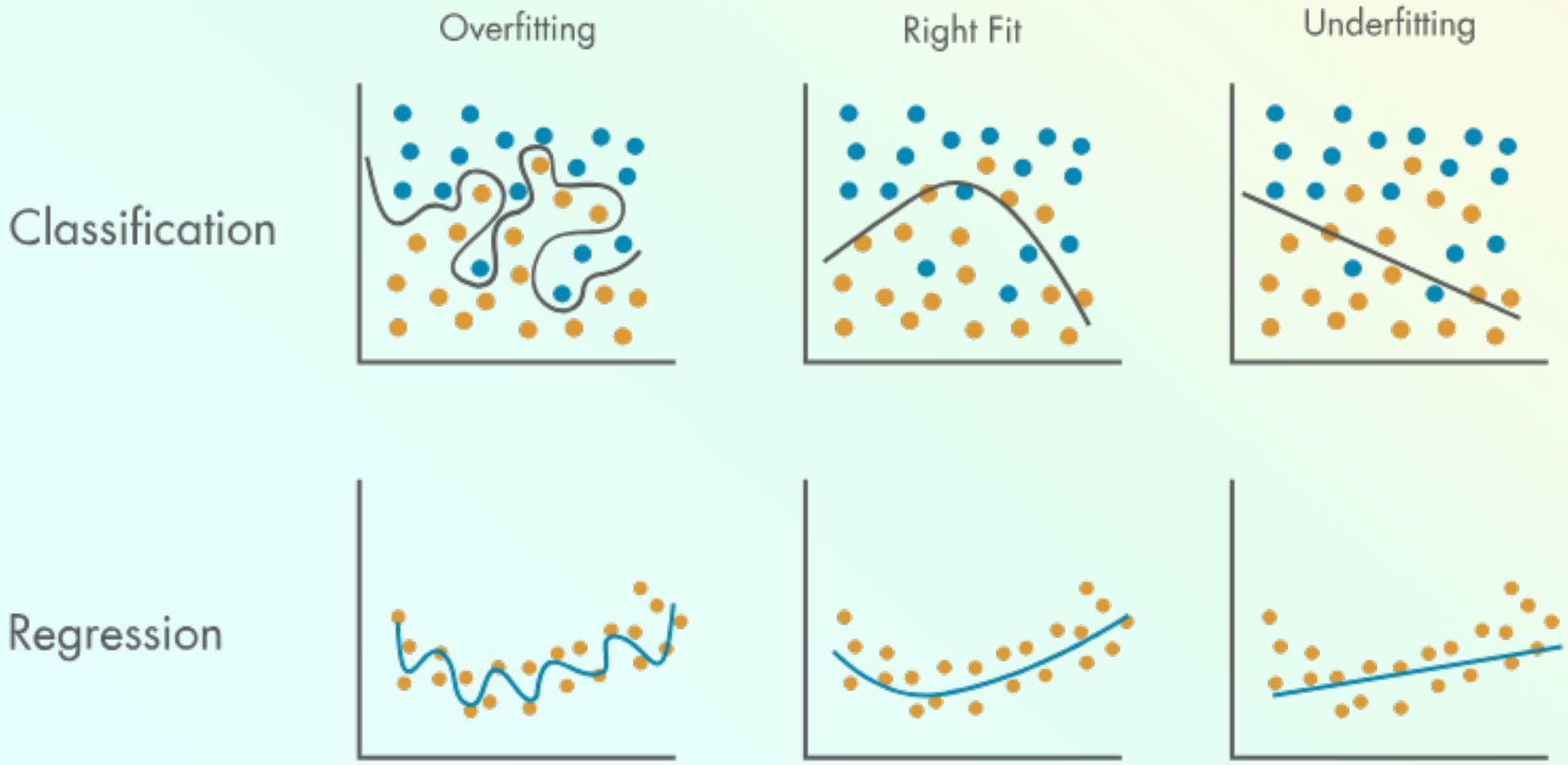
- Early stopping method — reduces overfitting

Cross validation:

- Select model and hyperparameters without overfitting to the data

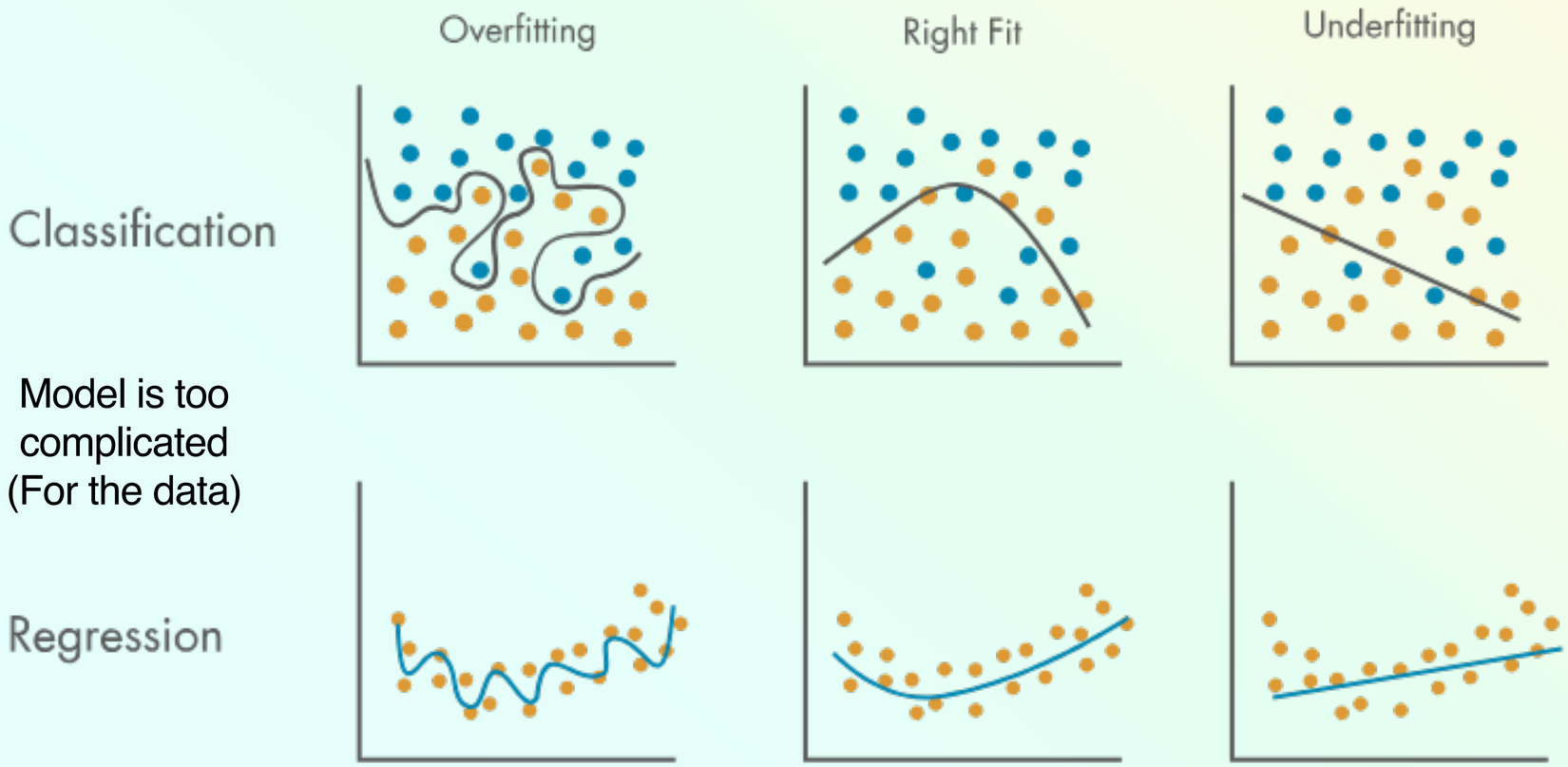
MACHINE LEARNING BASICS

OVERFITTING



MACHINE LEARNING BASICS

OVERFITTING



Model is too complicated (For the data)

Model is too simple. Cannot represent the function well.

MACHINE LEARNING BASICS

OVERFITTING

Reasons for overfitting:

1. Model is too complicated
2. Model is being fit to noise in the data
3. Insufficient data to know what is noise vs not noise

MACHINE LEARNING BASICS

OVERFITTING

Reasons for overfitting:

1. Model is too complicated
2. Model is being fit to noise in the data
3. **Insufficient data to know what is noise vs not noise**

MACHINE LEARNING BASICS

OVERFITTING

$$l(w) = \mathbf{E}\left[\left(f(X, w) - Y\right)^2\right]$$

MACHINE LEARNING BASICS

OVERFITTING

$l(w) = \mathbf{E}\left[\left(f(X, w) - Y\right)^2\right]$ — we do not know this, nor can we compute it

What we minimize is:

$$l_D(w) = \frac{1}{m}\sum_{i=1}^{m}\left(f(x_i, w) - y_i\right)^2$$

MACHINE LEARNING BASICS

OVERFITTING - WHY IS IT BAD

If $l_D(w) \approx l(w)$ then

$$\arg\min_w l_D(w) \approx \arg\min_w l(w)$$

(We hope, but it depends on f and l)

MACHINE LEARNING BASICS

OVERFITTING - WHY IS IT BAD

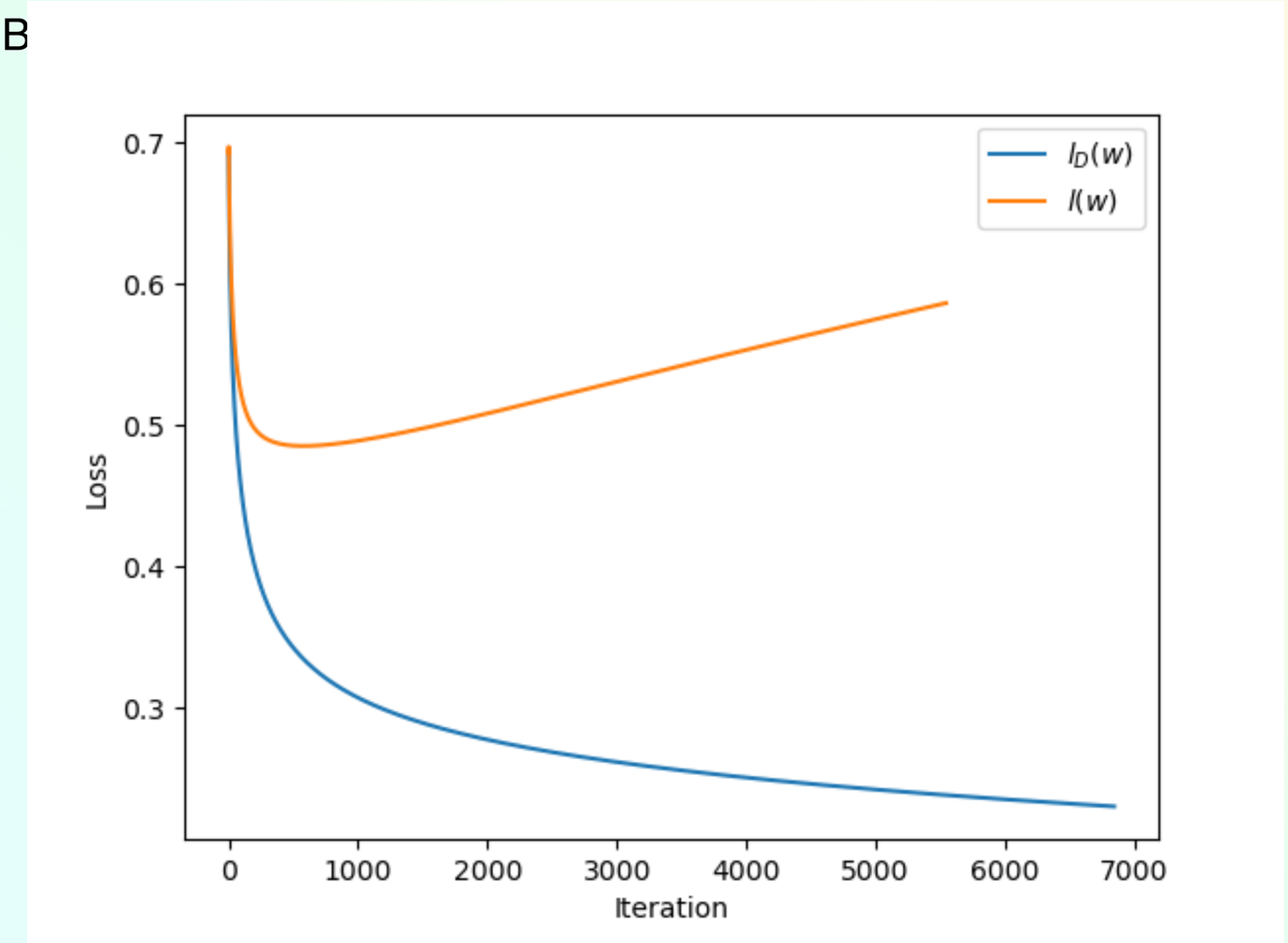
The optimal weights $w*$ are different for $l(w)$ and $l_D(w)$

Doing well on $l_D(w)$ usually means doing well on $l(w)$

But being optimal on $l_D(w)$ often means being worse on $l(w)$

MACHINE LEARNING BASICS

OVERFITTING - WHY IS IT B



Class question

BIAS

DEFINITION

Bias is the difference in the expectation of an estimator and the true expected value we want to estimate

$\mu = \mathbf{E}[X]$ — we want to estimate μ

$$\hat{\mu}_1 = \frac{1}{n}\sum_{i=1}^{n} X_i$$

Bias of $\hat{\mu}_1$: $\mathbf{E}[\hat{\mu}_1] - \mu = \mathbf{E}\left[\frac{1}{n}\sum_{i=1} X_i\right] - \mu = \left(\frac{1}{n}\sum_{i=1}^{n} \mathbf{E}[X_i]\right) - \mu = \left(\frac{1}{n}\sum_{i=1}^{n} \mu\right) - \mu = \mu - \mu = 0$

$\hat{\mu}_1$ is an unbiased estimator of μ

BIAS

BIASED ESTIMATOR

$$\hat{\mu}_2 = \frac{1}{n}\sum_{i=1}^{n}(X_i + C_i)$$

Bias of $\hat{\mu}_2$:

$$\mathbf{E}[\hat{\mu}_2] - \mu = \mathbf{E}\left[\frac{1}{n}\sum_{i=1}^{n}(X_i + C_i)\right] - \mu = \mu + \frac{1}{n}\sum_{i=1}^{n}\mathbf{E}[C_i] - \mu = \frac{1}{n}\sum_{i=1}^{n}\mathbf{E}[C_i]$$

$\hat{\mu}_2$ is a biased estimator if $\mathbf{E}[C_i] \neq 0$

MAXIMIZATION BIAS

BIAS DEFINITION

For stochastic gradient descent we want unbaised estimates of $l(w)$ and $\nabla l(w)$

$$\mathbf{E}[l_i(w)] - l(w) = 0$$

$$\mathbf{E}[\nabla l_i(w)] - \nabla l(w) = 0$$

MAXIMIZATION BIAS

BIAS DEFINITION

For stochastic gradient descent we want unbaised estimates of $l(w)$ and $\nabla l(w)$

$$\mathbf{E}[l_i(w)] - l(w) = 0$$

$$\mathbf{E}[\nabla l_i(w)] - \nabla l(w) = 0$$

Also want to be able to select the best weights.

We need to be able to estimate $l(w)$ without bias or we might choose the wrong weights

MAXIMIZATION BIAS

EXAMPLE

Consider playing a game where you get to roll dice and want to have the highest roll on average. You own two old dice that have been beat up and may not be unbiased any more. To optimize your advantage in the game you want to roll with your best dice.

You roll each dice m times and compute the average roll score for each one.

$$\bar{X}_m = \frac{1}{m}\sum_{i=1}^{m} X_i \qquad \bar{Y}_m = \frac{1}{m}\sum_{i=1}^{m} Y_i$$

You choose $\arg\max\{\bar{X}_m, \bar{Y}_m\}$ and decare that the advantage is $\max\{\bar{X}_m - \bar{Y}_m, \bar{Y}_m - \bar{X}_m\}$

MAXIMIZATION BIAS

EXAMPLE

Consider playing a game where you get to roll dice and want to have the highest roll on average. You own two old dice that have been beat up and may not be unbiased any more. To optimize your advantage in the game you want to roll with your best dice.

You roll each dice m times and compute the average roll score for each one.

$$\bar{X}_m = \frac{1}{m}\sum_{i=1}^{m} X_i \qquad\qquad \bar{Y}_m = \frac{1}{m}\sum_{i=1}^{m} Y_i$$

You choose $\arg\max\{\bar{X}_m, \bar{Y}_m\}$ and decare that the roll average is $\max\{\bar{X}_m, \bar{Y}_m\}$

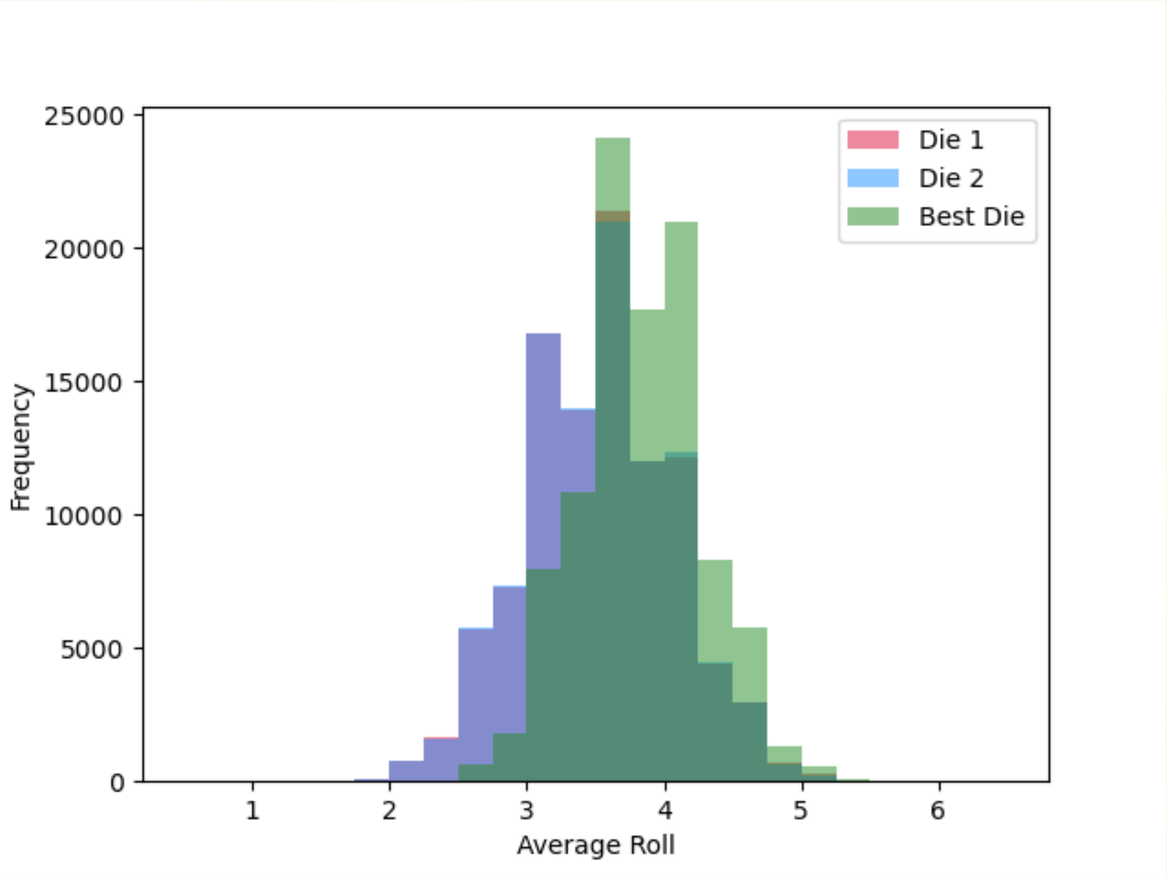
Do we choose the better dice? Is our assement of the average roll of the chosen dice fair?

MAXIMIZATION BIAS

EXAMPLE

Roll each die 10 times

Run this experiment with two identical dies 100,000 times.



MAXIMIZATION BIAS

EXAMPLE

Roll each die 10 times

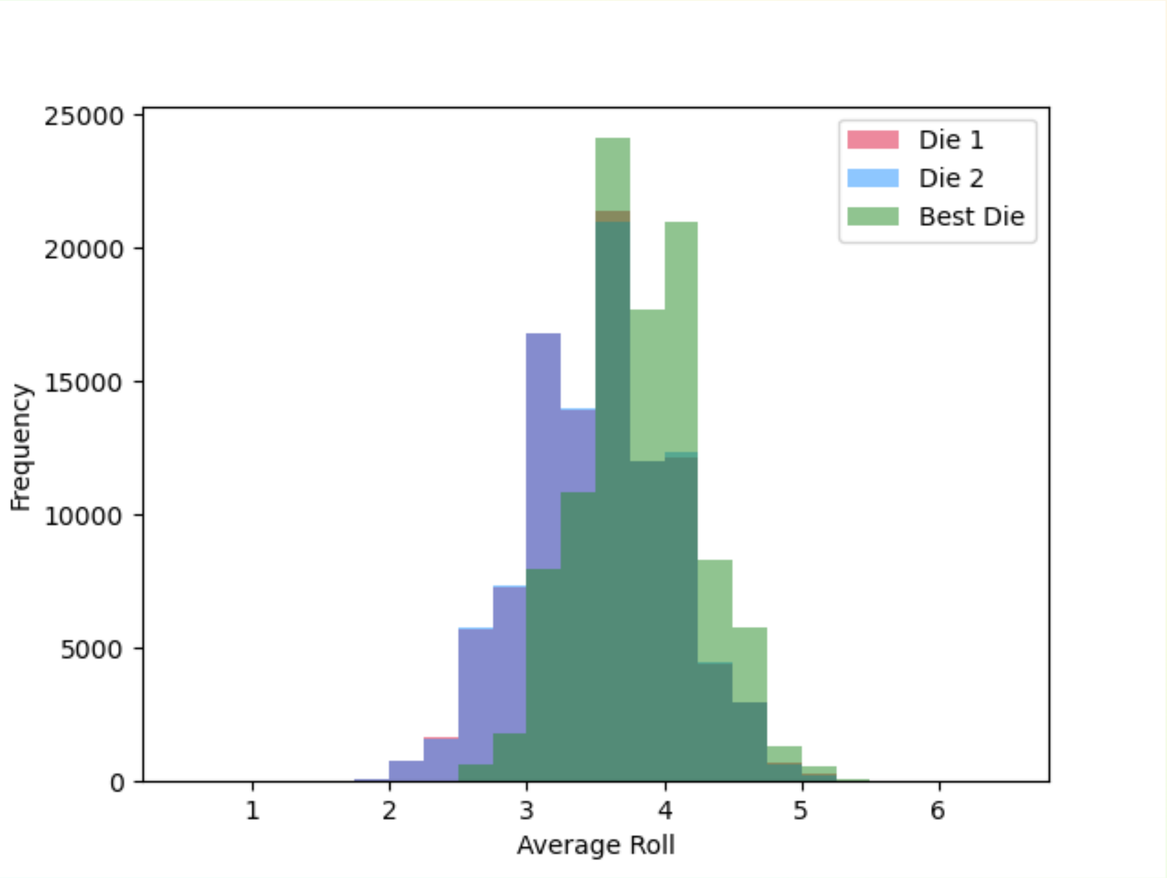
Run this experiment with two identical dies 100,000 times.

Dies 1 and 2 have same distribution

The best die has a different distribution

Its mean is higher. This is bias!

$$\mathbf{E}\left[\bar{X}_m\right] = \mathbf{E}\left[\bar{Y}_m\right] \neq \mathbf{E}\left[\max\{\bar{X}_m, \bar{Y}_m\}\right]$$



MAXIMIZATION BIAS

IN ML

Before getting the data D choose some weights w^1, w^2, etc

We can then evalute $w^1, w^2, \ldots$ using data D,

$l_D(w)$ is the empirical loss using data D

$$\mathbf{E}\left[l_D(w^1)\right] = l(w^1), \mathbf{E}\left[l_D(w^2)\right] = l(w^2)$$

We can then choose the best w of this pre chosen set with unbiased evaluations.

However, we cannot say how good these chosen weights are because we already used D.

MAXIMIZATION BIAS

IN ML

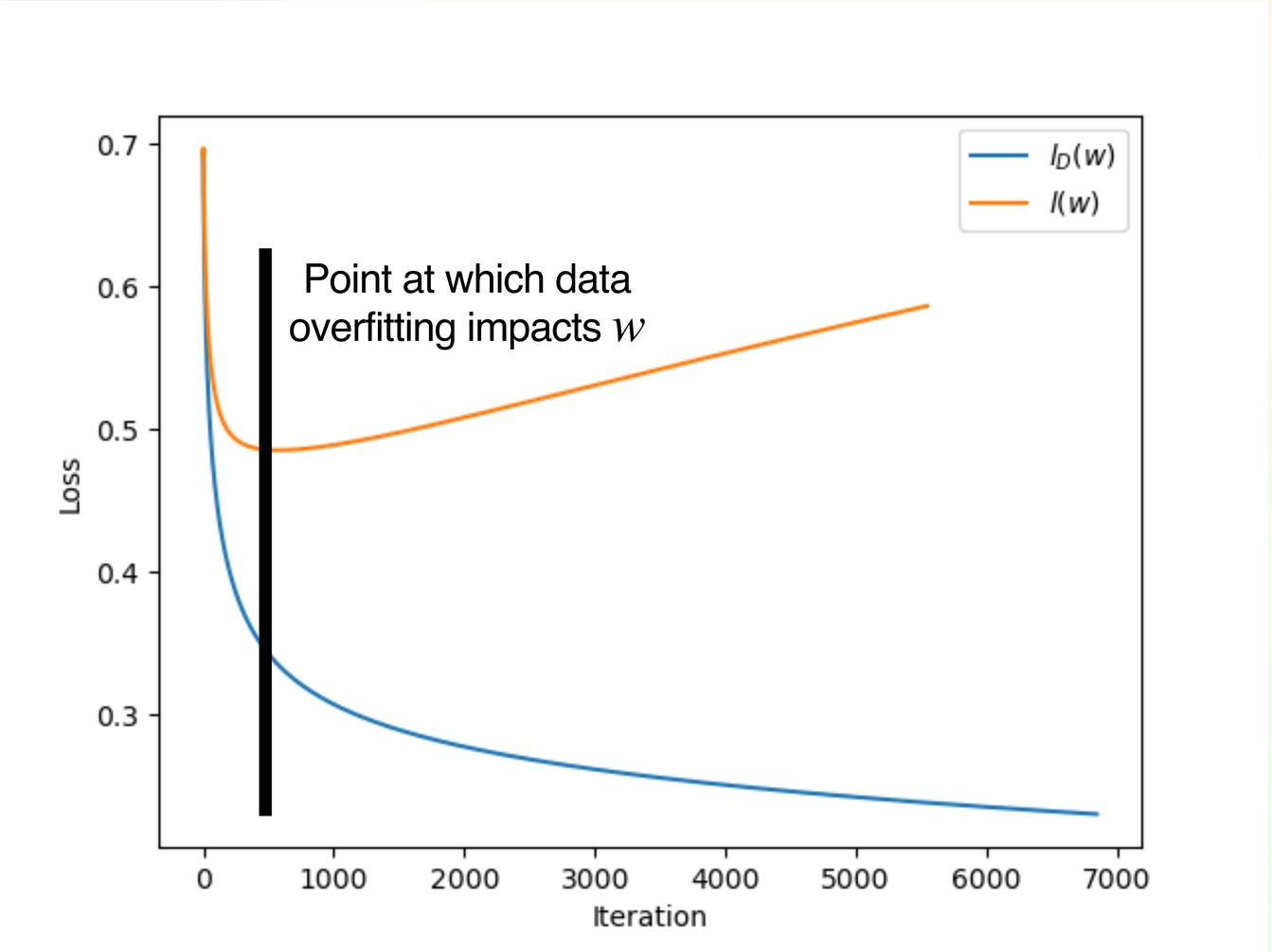
When we optimize w on D, we are choosing the w that does the best on D.

This biases the $\min_w l_D(w)$ to be less than $l(w)$.

Need new data D' then we can evalaute $l_{D'}\left(\arg\min_w l_D(w)\right)$

MAXIMIZATION BIAS

IN ML



TEST SETS

UNBIASED EVALUATIONS

Having both D to train on and D' to evaluate the model on is necessary to get an unbaised assement of our model.

Solution: Create D_{train} and D_{test} from a single data set D

- D_{train} contains some proportion $p \in (0,1)$ of the data and is used to train the model
- D_{test} contains some proportion $1 - p$ of the data and is used to provide an unbaised evaluation of the model

TEST SETS

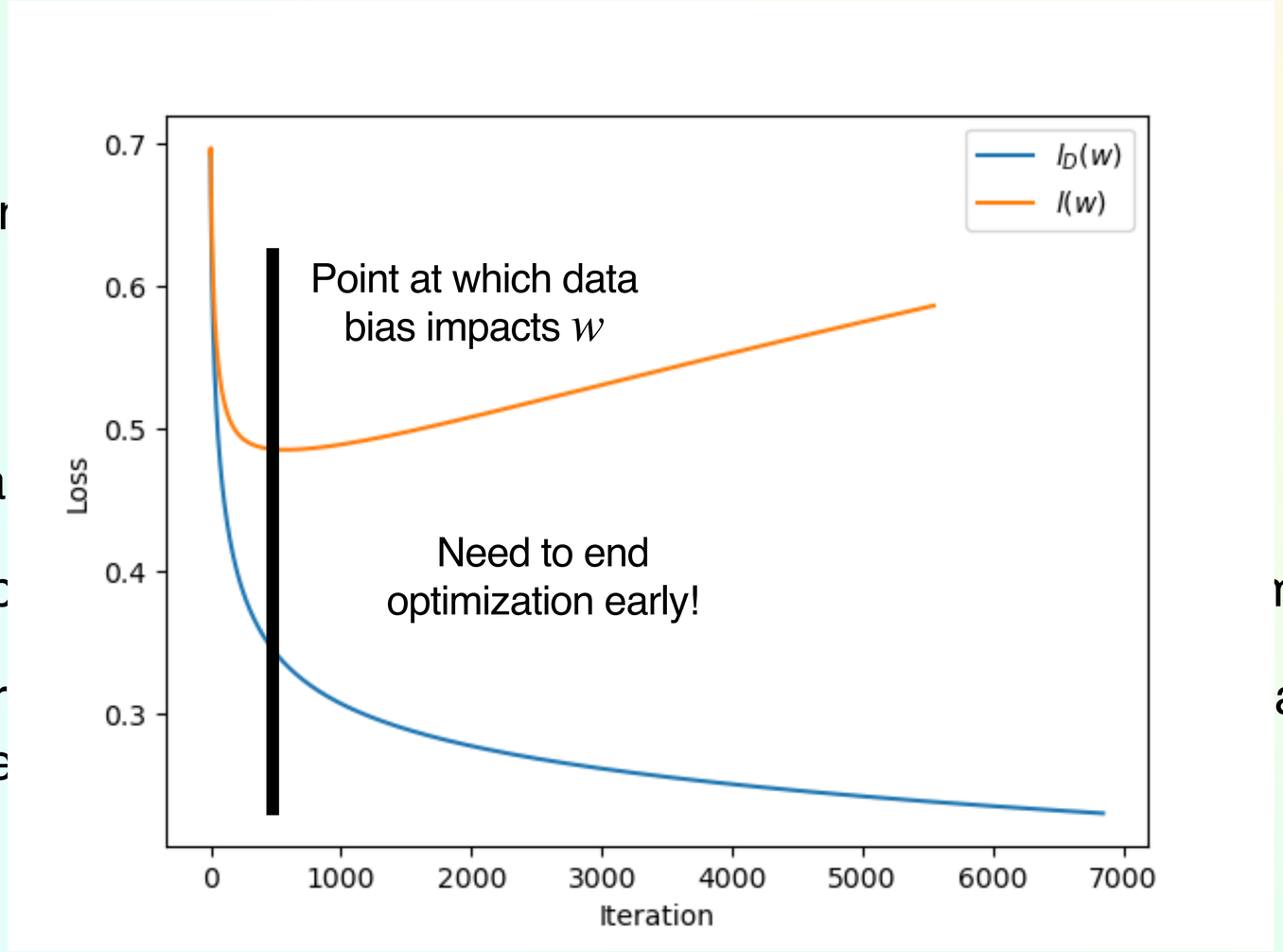
UNBIASED EVALUATIONS

Having both D to train or unbaised assement of our model.

Solution: Create D_{train} a

D_{train} contains some p model

D_{test} contains some pr aised evaluation of the mode



EARLY STOPPING

USING VALIDATION DATA

Need to decide when to stop — Cannot use $|l_{D_{train}}(w^{k+1}) - l_{D_{train}}(w^k)| < \epsilon$

Split D into three data sets $D_{train}, D_{val}, D_{test}$

D_{train} — search for the best weights (cannot evaluate if overtraining without bias)

D_{val} — Validation data used to evaluate if over training and we should stop.

D_{test} — used to evaluate the final model, i.e., say how accurate it is

EARLY STOPPING

USING VALIDATION DATA

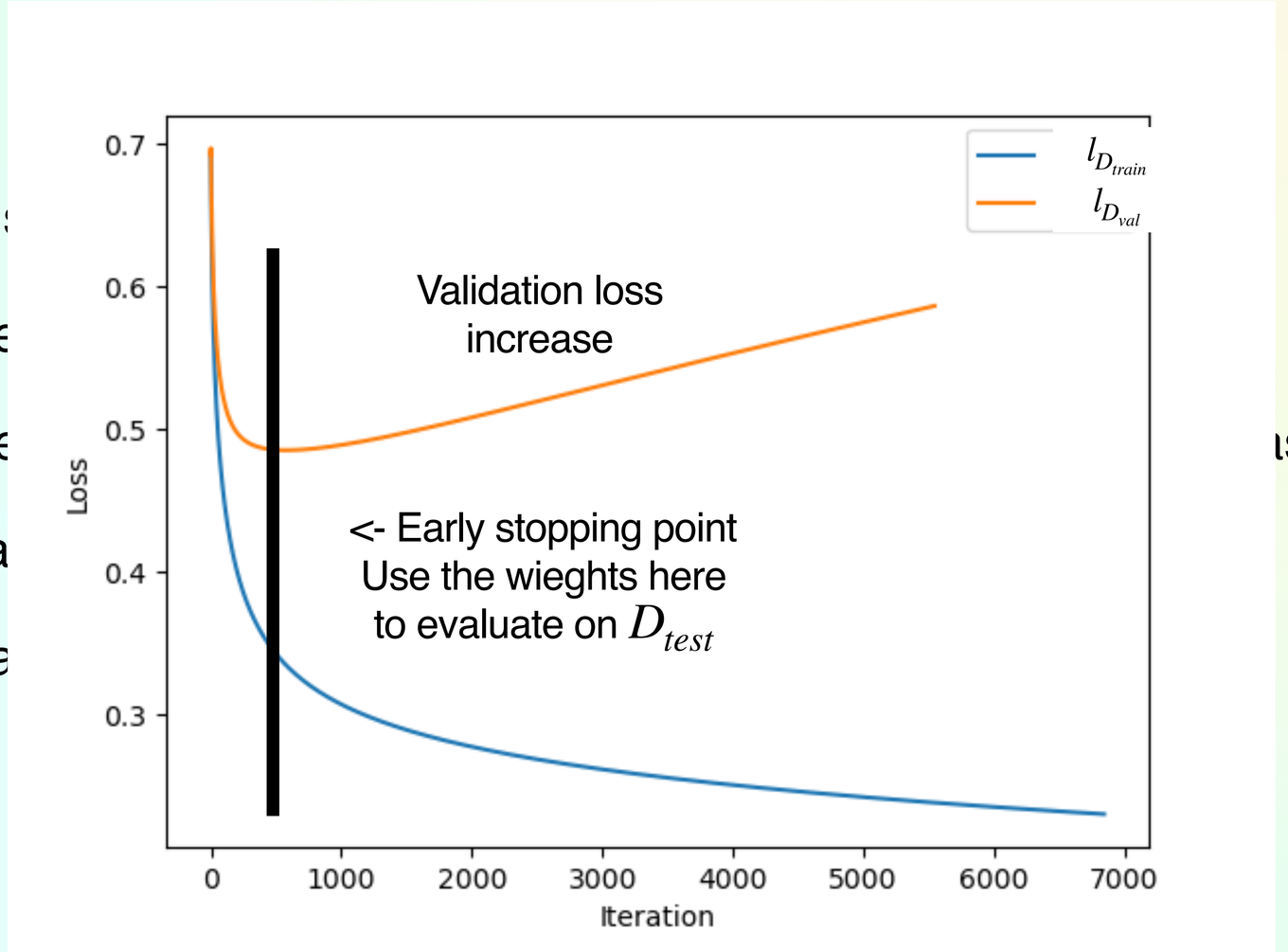
Need to decide when to s

Split D into three data se

D_{train} — search for the ıs)

D_{val} — Validation data

D_{test} — used to evalua



EARLY STOPPING

USING VALIDATION DATA

gradient_descent_wearlystopping($D, w_0, \rho_{test}, \rho_{val}, \epsilon, \eta$)

$D_{train}, D_{val}, D_{test} \leftarrow$ split_data($D, \rho_{test}, \rho_{val}$)

$w = w_0$

$w_{best} = w$

Loop:

$w \leftarrow w - \eta \nabla l_{D_{train}}(w)$

If $l_{D_{val}}(w) < l_{D_{val}}(w_{best})$

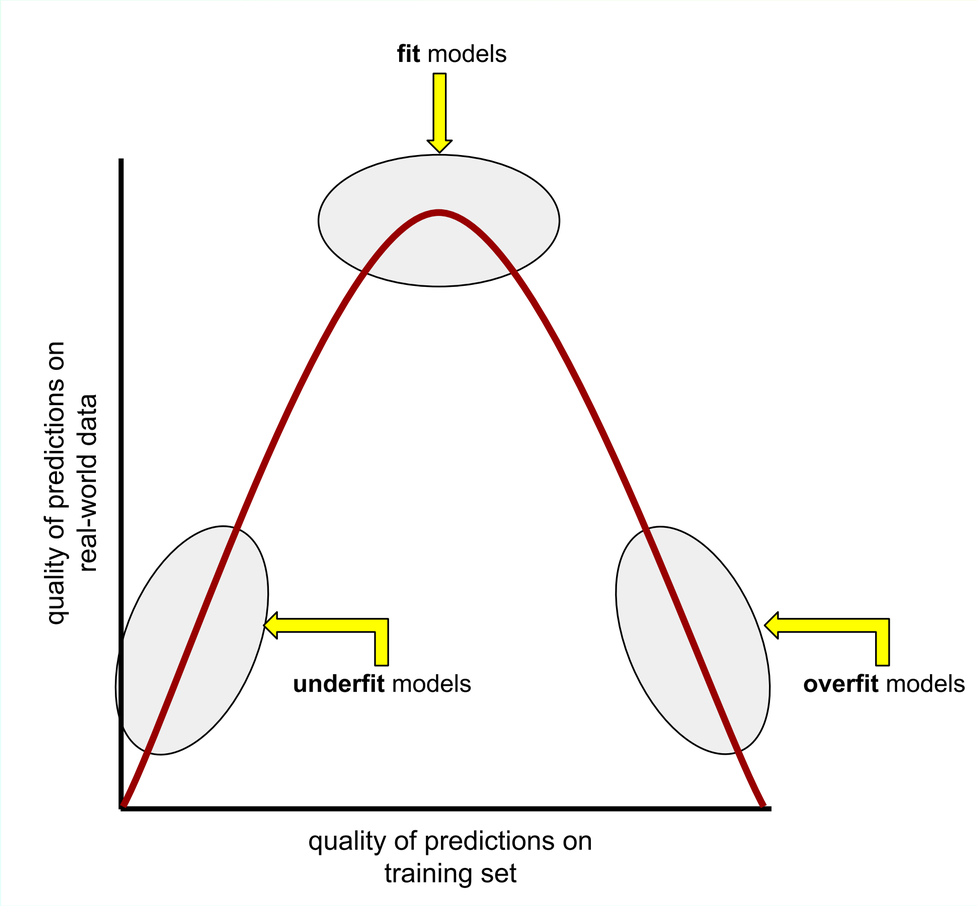
$w_{best} = w$

If $l_{D_{val}}(w) > l_{D_{val}}(w_{best}) + \epsilon$ // check if validation loss is sufficiently increasing

Return w_{best}

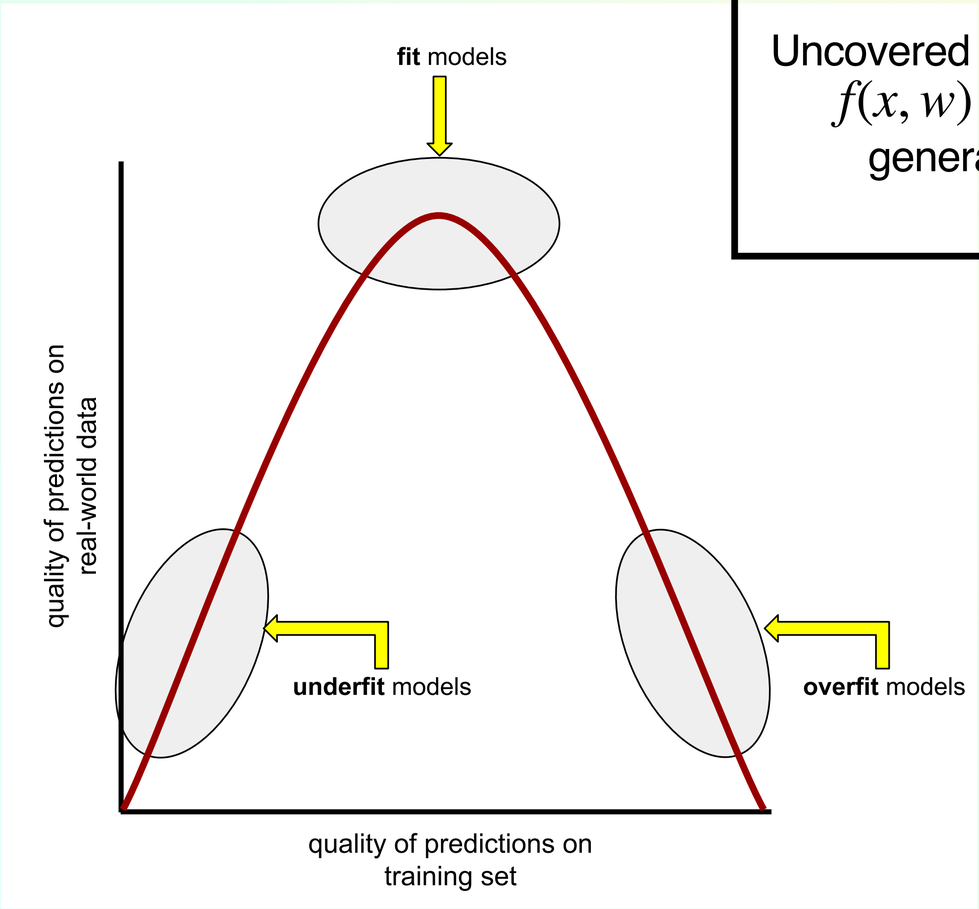
DEFINING OVERFITTING

BEING PRECISE :(



DEFINING OVERFITTING

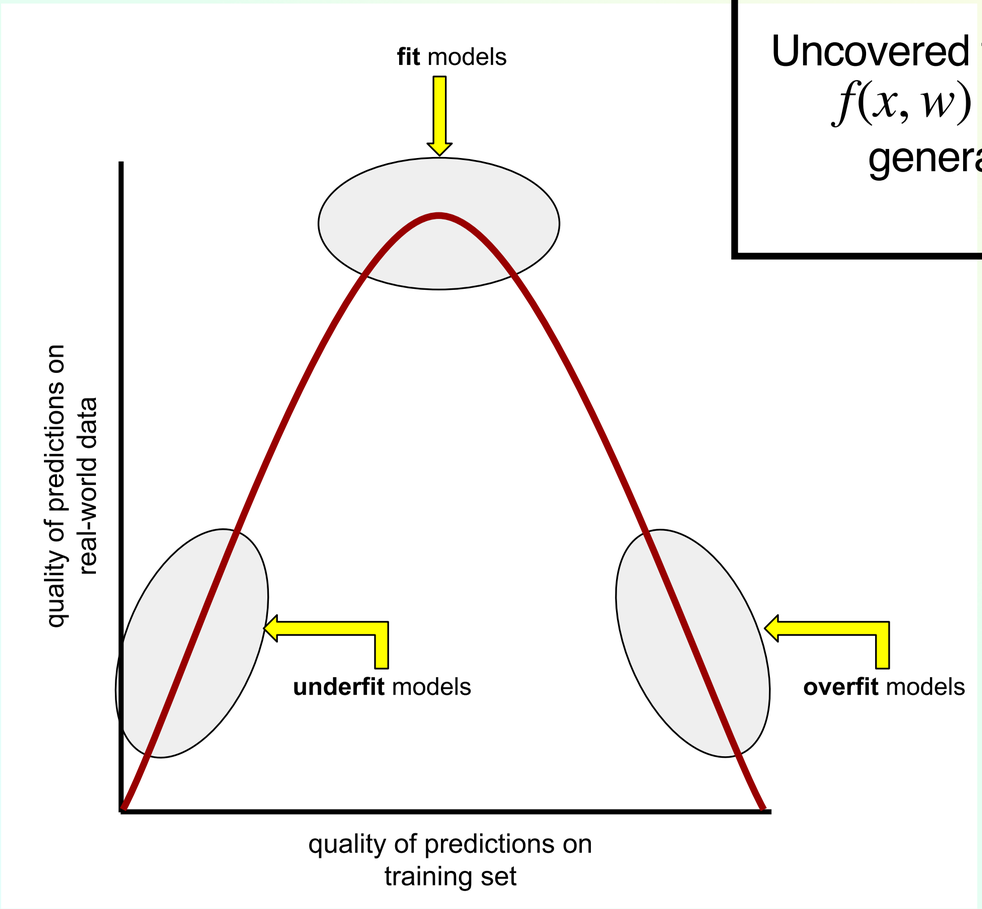
BEING PRECISE :(



Uncovered the characteristics of $f(x, w)$ in D_{train} that are generalizable to D_{test}

DEFINING OVERFITTING

BEING PRECISE :(

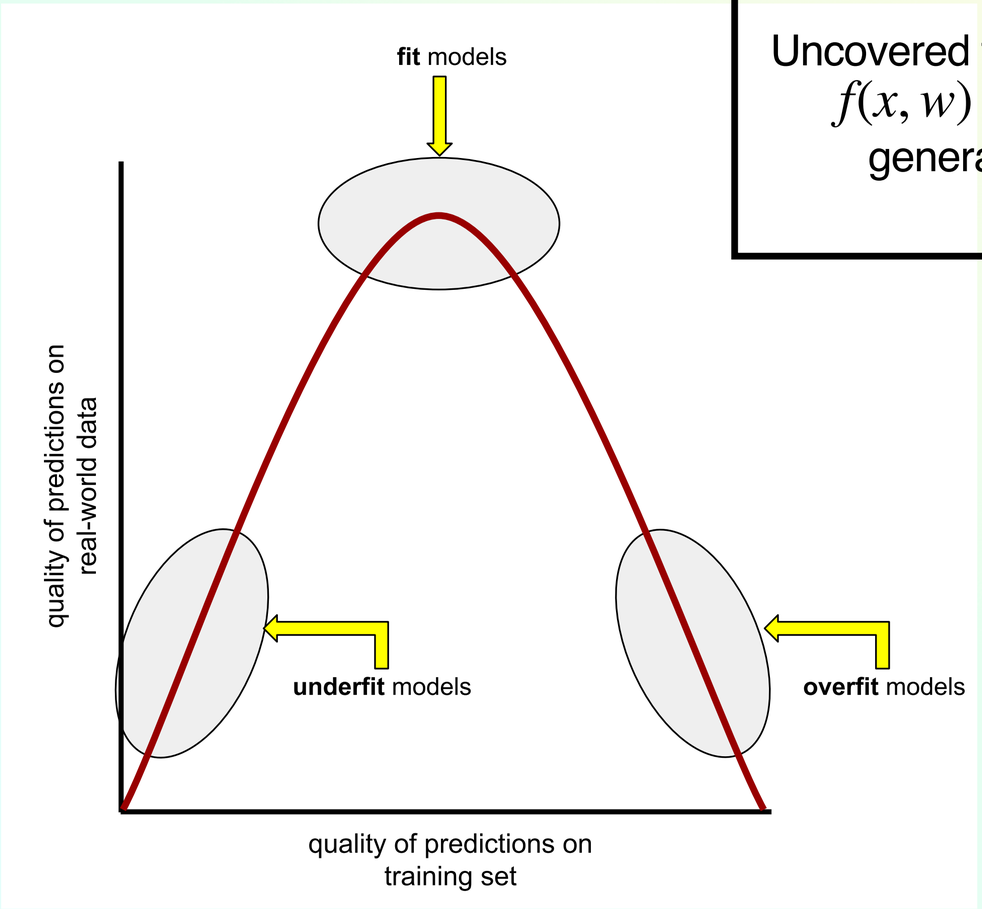


Uncovered the characteristics of $f(x, w)$ in D_{train} that are generalizable to D_{test}

Identified the characteristics $f(x, w)$ that are only important for doing well on the training data D_{train}

DEFINING OVERFITTING

BEING PRECISE :(



Uncovered the characteristics of $f(x, w)$ in D_{train} that are generalizable to D_{test}

Did not uncover characteristics of $f(x, w)$ that are present in D_{train} and did not even able to model D_{train} well

Identified the characteristics $f(x, w)$ that are only important for doing well on the training data D_{train}

OVER/UNDER FITTING

REASONS

Overfitting:

- f models noise present in D_{train}
- f is too complex for the data
  - Number of data points < model parameters
- Ran gradient descent too long on D_{train}

$$\arg\min_w l_{D_{train}}(w) \neq \arg\min_w l(w)$$

Underfitting:

- The model is too simple
  - Ex. Fitting a line to a quadratic
  - Can still overfit a line to D_{train}
    - (Overfit to data vs. overfit of model class)
- Not extracting enough information out of D_{train}
  - Didn't run gradient descent long enough

Class question

BIAS-VARIANCE

DECOMPOSITION OF ERROR

$$Y = f_*(X) + \xi$$

$$\mathbf{E}[\xi] = 0$$

f_D is approximation produced by an algorithm on a data set D

$$\mathbf{E}\left[\left(Y - f_D(x_0)\right)^2 \middle| X = x_0\right] = \text{Irreducible Error} + \text{Bias}^2 + \text{Variance}$$

BIAS-VARIANCE

DECOMPOSITION OF ERROR

$$
\begin{aligned}
\left(Y - f_D(x_0)\right)^2 &= \left(Y - f_*(x_0) + f_*(x_0) - f_D(x_0)\right)^2 \\
&= \left(\left[Y - f_*(x_0)\right] + \left[f_*(x_0) - f_D(x_0)\right]\right)^2 \\
&= \left(Y - f_*(x_0)\right)^2 + 2\left[Y - f_*(x_0)\right]\left[f_*(x_0) - f_D(x_0)\right] + \left(f_*(x_0) - f_D(x_0)\right)^2
\end{aligned}
$$

BIAS-VARIANCE

DECOMPOSITION OF ERROR

$$\begin{aligned}\mathbf{E}\left[\left(Y - f_D(x_0)\right)^2 \middle| X = x_0\right] = \; & \mathbf{E}\left[\left(Y - f_*(x_0)\right)^2 \middle| X = x_0\right] \\ & + 2\mathbf{E}\left[\left(Y - f_*(x_0)\right) \middle| X = x_0\right] \mathbf{E}\left[\left(f_*(x_0) - f_D(x_0)\right) \middle| X = x_0\right] \\ & + \mathbf{E}\left[\left(f_*(x_0) - f_D(x_0)\right)^2 \middle| X = x_0\right]\end{aligned}$$

BIAS-VARIANCE

DECOMPOSITION OF ERROR

$$\mathbf{E}\left[\left(Y - f_D(x_0)\right)^2 \middle| X = x_0\right] = \underbrace{\mathbf{E}\left[\left(Y - f_*(x_0)\right)^2 \middle| X = x_0\right]}_{\text{Irreducible Error}} + \mathbf{E}\left[\left(f_*(x_0) - f_D(x_0)\right)^2 \middle| X = x_0\right]$$

BIAS-VARIANCE

DECOMPOSITION OF ERROR

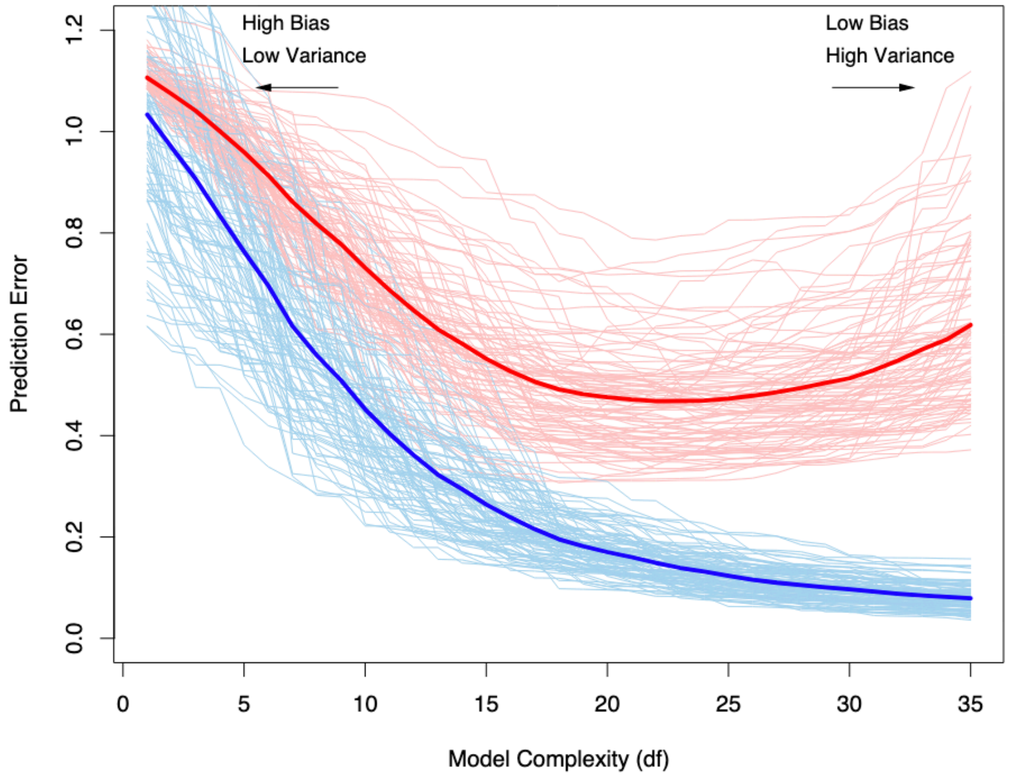
$$\mathbf{E}f \doteq \mathbf{E}\left[f_D\right]$$

$$\begin{aligned}
\mathbf{E}\left[\left(f_*(x_0) - f_D(x_0)\right)^2 \middle| X = x_0\right] &= \mathbf{E}\left[\left(f_*(x_0) - \mathbf{E}f(x_0) + \mathbf{E}f(x_0) - f_D(x_0)\right)^2\right] \\
&= \left(f_*(x_0) - \mathbf{E}f(x_0)\right)^2 + 2\left(f_*(x_0) - \mathbf{E}f(x_0)\right)\mathbf{E}\left[\mathbf{E}f(x_0) - f_D(x_0)\right] + \mathbf{E}\left[\left(\mathbf{E}f(x_0) - f_D(x_0)\right)^2\right] \\
&= \underbrace{\left(f_*(x_0) - \mathbf{E}f(x_0)\right)^2}_{=\mathrm{Bias}(f)^2_{x_0}} + \underbrace{\mathbf{E}\left[\left(\mathbf{E}f(x_0) - f_D(x_0)\right)^2\right]}_{=\mathrm{Var}(f)_{x_0}}
\end{aligned}$$

BIAS-VARIANCE

DECOMPOSITION OF ERROR

High bias^2 — the expected model has a large error



High Variance — The model greatly changes its output for different training data sets.

BIAS

BREAKING DOWN BIAS FOR LINEAR MODELS

$$w_* \in \arg\min_w \mathbf{E}\left[\left(f_*(X) - f(X,w)\right)^2\right]$$

$$\mathbf{E}\left[f_*(X) - \mathbf{E}f(X,w)\right]^2 = \underbrace{\mathbf{E}\left[f_*(X) - f(X,w_*)\right]^2}_{\text{Avg}[\textit{Model Bias}]^2} + \underbrace{\mathbf{E}\left[f(X,w_*) - f(X,w)\right]^2}_{\text{Avg}[\textit{Estimation Bias}]^2}$$

Model Bias — limits on what the model can represent

Estimation Bias — error that the optimization algorithm has in finding the optimal parameters

OVERFITTING

RECAP

We do not have access to the loss function l to optimize w

We want the best fit possible, but we do not want to fit noise (overfitting)

We want to find the best w for l using data D

Early stopping helps reduce overfitting w on D

OVERFITTING

RECAP

We do not have access to the loss function l to optimize w

We want the best fit possible, but we do not want to fit noise (overfitting)

We want to find the best w for l using data D

Early stopping helps reduce overfitting w on D

We care about finding the best approximate function f, not just optimal weights

Need to be careful with other sources of overfitting

OTHER SOURCES OF BIAS

MODEL SELECTION

We are not trying to find optimal weights w^*, but look for the class of functions f

We choose hyperparameters (anything we choose that leads to an approximation $\hat{f}$):

- Step size
- basis function ϕ
- initial weights w^0
- many other little choices

OTHER SOURCES OF BIAS

MODEL SELECTION

We are not trying to find optimal weights $w*$, but look for the class of functions f

We choose hyperparameters (anything we choose that leads to an approximation $\hat{f}$):

- Step size
- basis function ϕ
- initial weights w^0
- many other little choices

We guess at some of these, look at the test loss, and see if they were a good choice

If not good enough, we choose new ones and rerun them. —**This is introducing bias based on the D_{test}**

MODEL DESIGN OPTIMIZATION

BROADER PROBLEM

$$\phi^*, w^* \in \arg\min_{\phi,\varphi} l_{D_{test}}\left(\phi, w_{alg}(\phi,\varphi)\right)$$

MODEL DESIGN OPTIMIZATION

BROADER PROBLEM

$$\phi^*, w^* \in \arg\min_{\phi,\varphi} l_{D_{test}}\left(\phi, w_{alg}(\phi,\varphi)\right)$$

Bias evaluation because we evaluate on D_{test} — overfitting to D_{test}

Idea: select a model based on $l_{D_{val}}$

MODEL DESIGN OPTIMIZATION

BROADER PROBLEM

$$\phi^*, w^* \in \arg\min_{\phi,\varphi} l_{D_{test}}\left(\phi, w_{alg}(\phi,\varphi)\right)$$

Bias evaluation because we evalaute on D_{test} — overfitting to D_{test}

Idea: select a model based on $l_{D_{val}}$

Problem: overfitting to D_{val} — could get a validation set that is especially bad for a few cases

What if we had multiple D_{val}?

CROSS-VALIDATION

OUTLINE

General idea:

1. Create multiple D_{train}, D_{val}, find the best hyperparameters across these data sets

   Finds hyperparameters that are good at finding a solution across several D_{train}, D_{test}

CROSS-VALIDATION

OUTLINE

General idea:

1. Create multiple D_{train}, D_{val}, find the best hyperparameters across these data sets

   Finds hyperparameters that are good at finding a solution across several D_{train}, D_{test}

2. Refit the model on all the data (except D_{test})

   Find the best fit with as much data as possible

CROSS-VALIDATION

OUTLINE

General idea:

1. Create multiple D_{train}, D_{val}, find the best hyperparameters across these data sets

   Finds hyperparameters that are good at finding a solution across several D_{train}, D_{test}

2. Refit the model on all the data (except D_{test})

   Find the best fit with as much data as possible

3. Evaluate on D_{test}

   Unbiased evaluation of the final model

CROSS-VALIDATION

OUTLINE

Procedure:

1. Split the data set D into D', D_{test}
2. Create (disjoint) data sets $D_1, D_2, \ldots, D_k$ from D'
3. Create k different train and validation data sets

$$D^1_{train}, D^1_{val} = \bigcup_{i \neq 1}^{k} D_i, D_1 \qquad D^2_{train}, D^2_{val} = \bigcup_{i \neq 2}^{k} D_i, D_2, \ldots, \qquad D^k_{train}, D^k_{val} = \bigcup_{i \neq k}^{k} D_i, D_k$$

CROSS-VALIDATION

OUTLINE

Procedure:

4. find the best hyperparameters that work well on average across D^i_{train}, D^i_{val}

$$\phi^*, \varphi^* = \arg\min_{\phi,\varphi} \frac{1}{k} \sum_{i=1}^{k} l_{D^i_{val}} \Big(\phi, w_{alg}(D^i_{train}, \phi, \varphi) \Big),$$

$w_{alg}(D, \phi, \varphi)$ returns the weights output by an algorithm alg using the hyperparameters ϕ and φ and data D

5. Retrain on whole train set and evaluate on D_{test}

$$w^* = w_{alg}(D', \phi^*, \varphi^*), \qquad l_{D_{test}}(\phi^*, w^*)$$

CROSS-VALIDATION

OUTLINE

Procedure:

4. find the best hyperparameters that work well on average across D^i_{train}, D^i_{val}

$$\phi^*, \varphi^* = \arg\min_{\phi,\varphi} \frac{1}{k} \sum_{i=1}^{k} l_{D^i_{val}} \left(\phi, w_{alg}(D^i_{train}, \phi, \varphi) \right),$$

To find good ϕ, φ:

Sample a large number of choices, ϕ, φ then use the best one.

ITERATIVE DEVELOPMENT

TYPICAL ML PROCESS

Sussie gets some dataset (D', D_{test}), finds some good model f, and reports performance on D_{test}.

Sam learns about what Sussie did and thinks they can do better. They find some new model f' and reports better performance on D_{test}.

Is Sam's result biased or unbiased (discuss)?

ITERATIVE DEVELOPMENT

TYPICAL ML PROCESS

Sam's result is biased because they already learned what Sussie did, which provides information about how to do well on D_{test}.

If Sam collected new data D'_{test} and evaluated both f and f' on it, then the comparison of the **models** would be unbiased. However, the comparison of how to **search** for the models is not without bias.

Research in ML/Deep Learning has both of these biases!

NEXT CLASS

Next Class —Optimizing Basis Functions