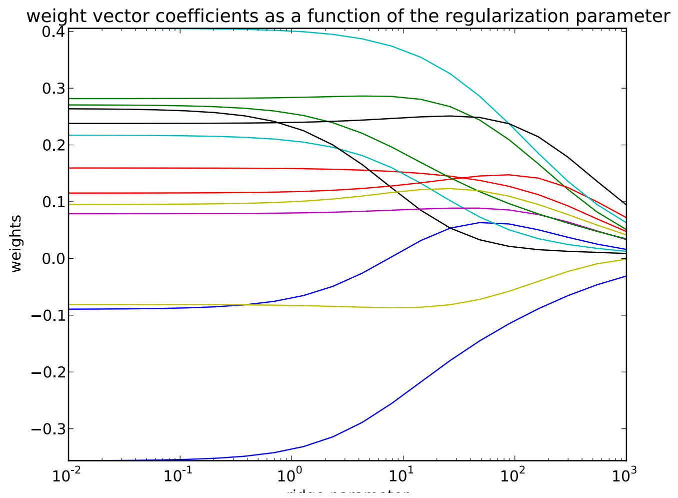
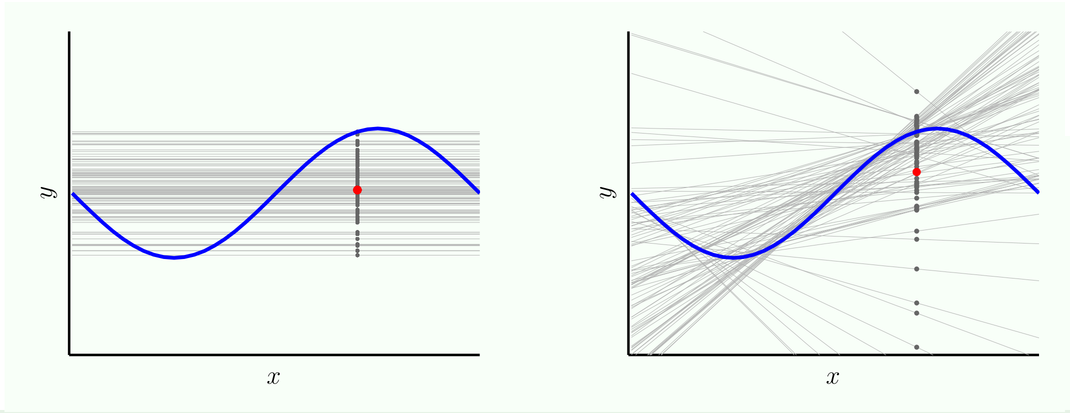
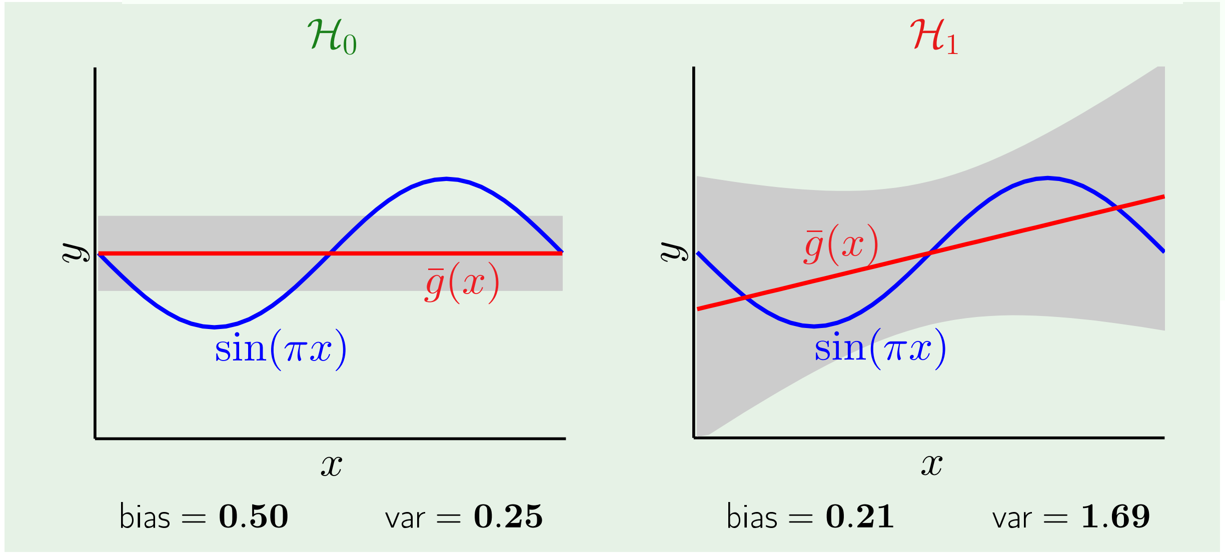
Regularization and model selection

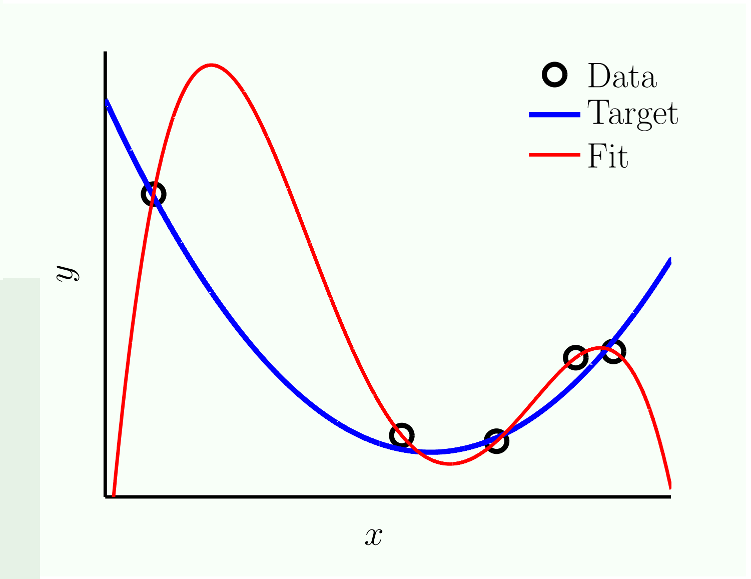
Chapter 4



Reminder: bias vs variance, overfitting

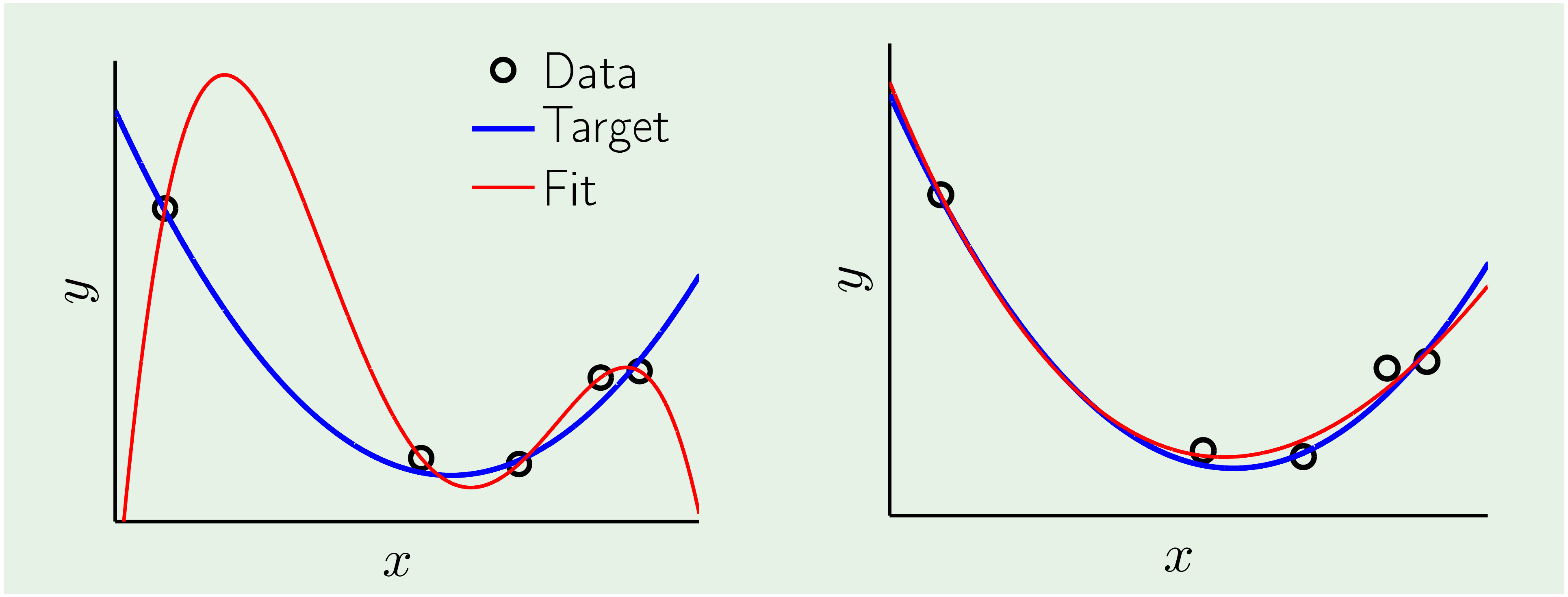






Regularization

The cure for overfitting - regularization



Without regularization

With regularization

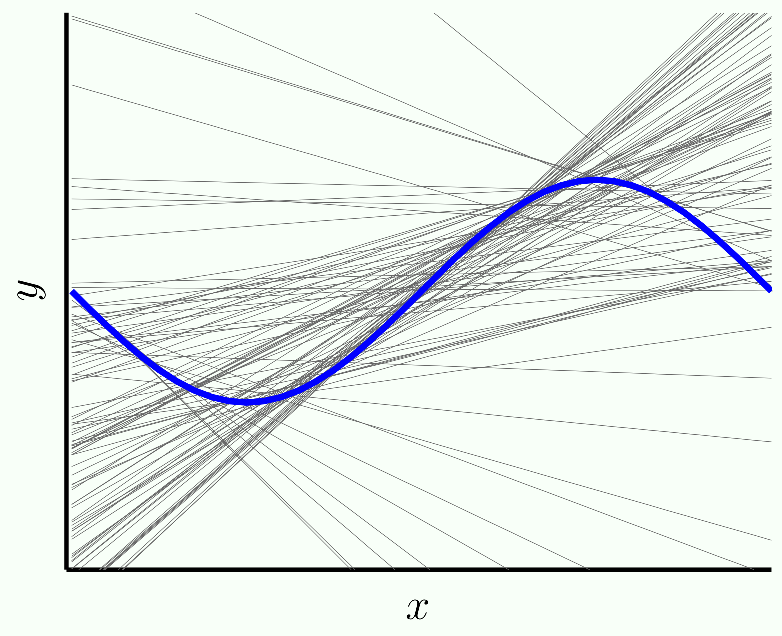
Regularization

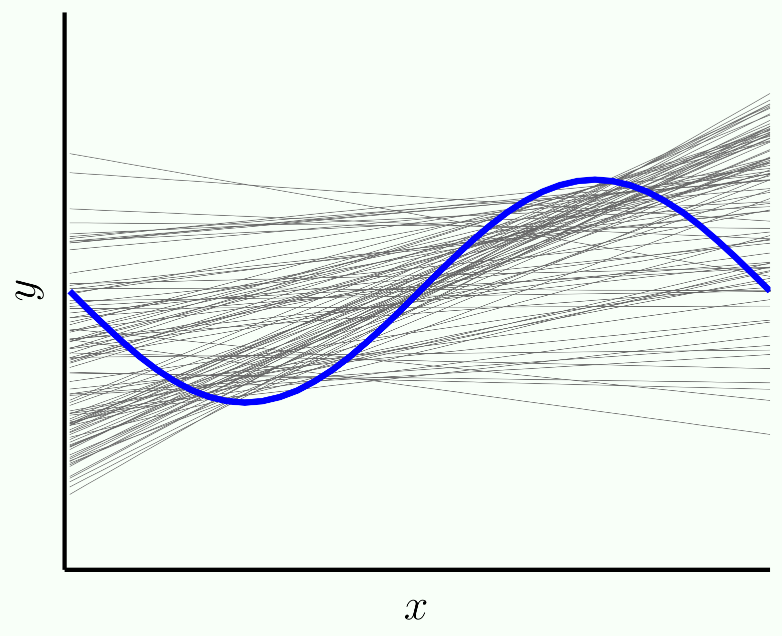
How does it work?

- Constrains the model so it cannot fit the noise
- Potential side effect: if it cannot fit the noise, can it fit the target function?
- Introduces bias and reduces variance, so that (hopefully) out-of-sample error is lower

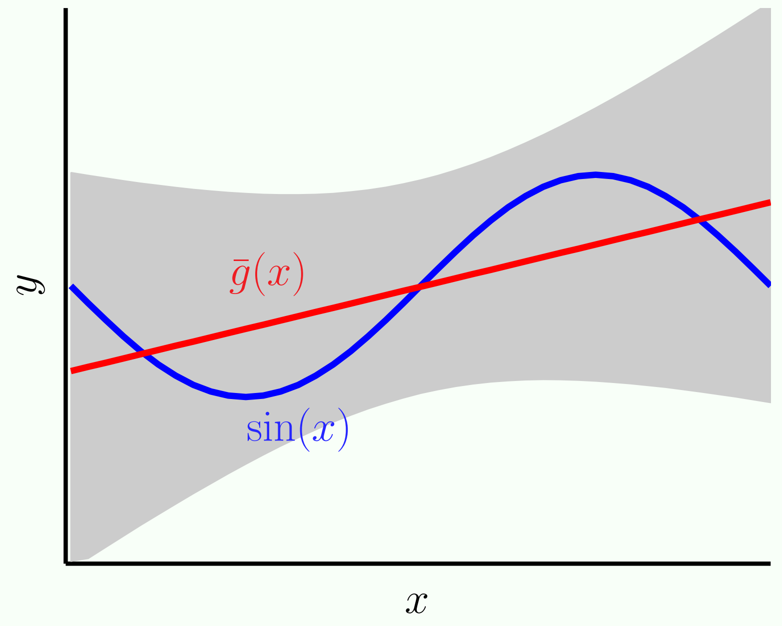
Constraining the model

Let's penalize large weights

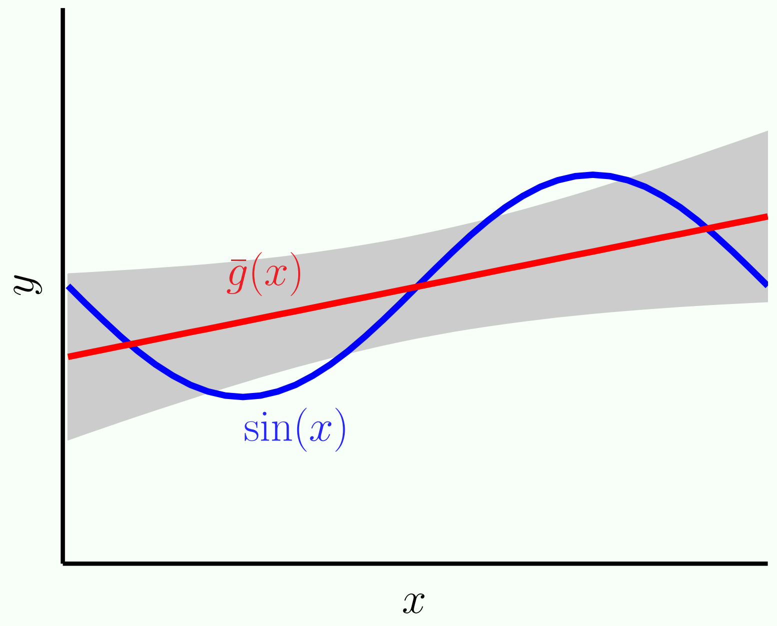




One effect: increased bias



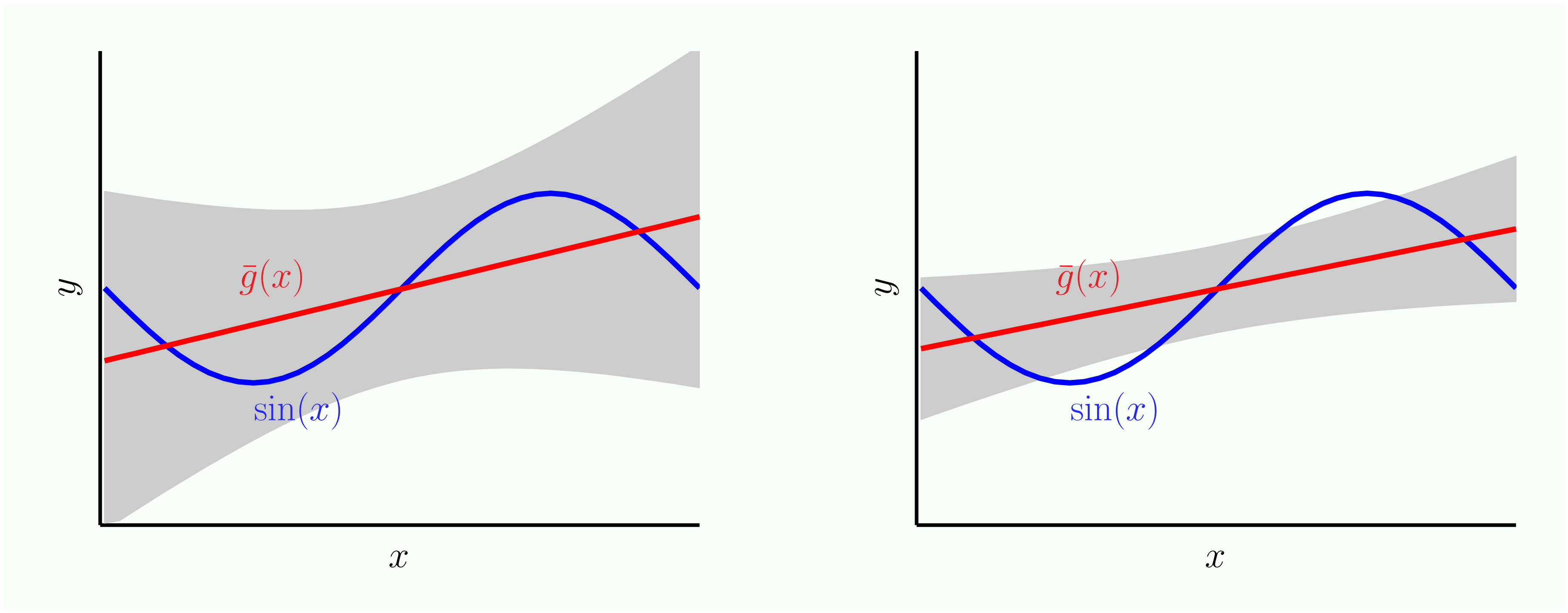
no regularization



regularization

bias $= 0.21$ bias $= 0.23$

Second effect: dramatic reduction in variance



no regularization	regularization
bias $= 0.21$	bias $= 0.23$
var $= 1.69$	var $= 0.33$

Constraining the complexity of the model

Replace E_{in} with:

$$E_{\text{aug}}(h) = E_{\text{in}}(h) + \frac{\lambda}{N}\Omega(h)$$

Regularization term

λ regularization constant

Choosing a regularizer

We want to constrain the learned function in the direction of the target function.

Intuition: noise is non-smooth

Common choice for the augmented in-sample-error:

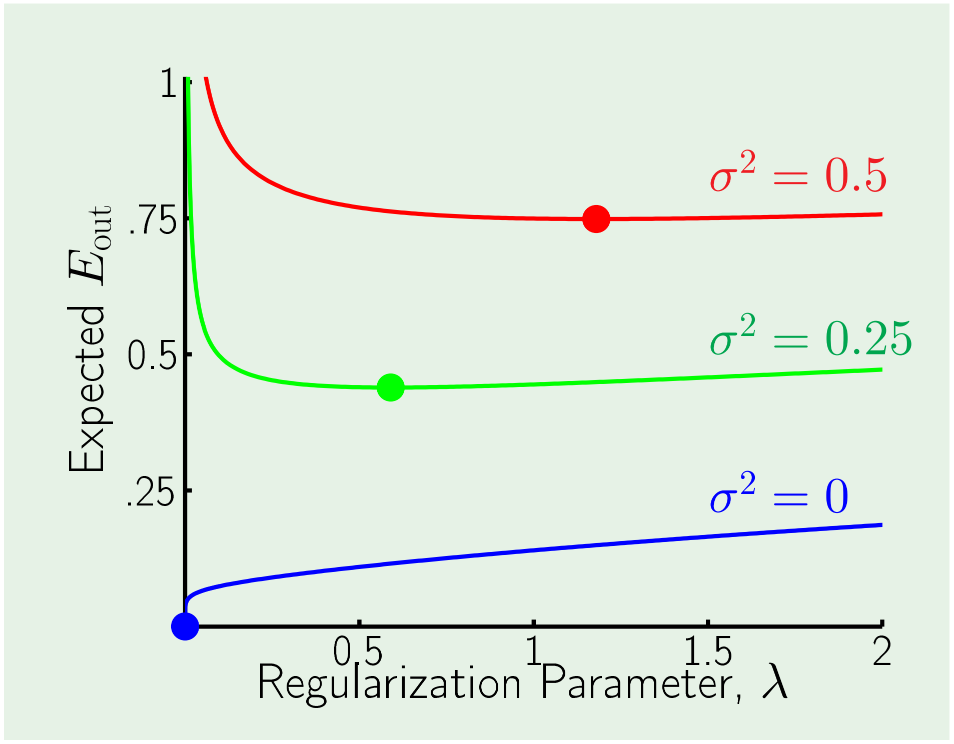
$$E_{aug}(\mathbf{w}) = E_{in}(\mathbf{w}) + \lambda \mathbf{w}^{\intercal}\mathbf{w}$$

weight decay regularizer

This regularization term controls the size of the components of the weight vector.

Is there an optimal value for λ?

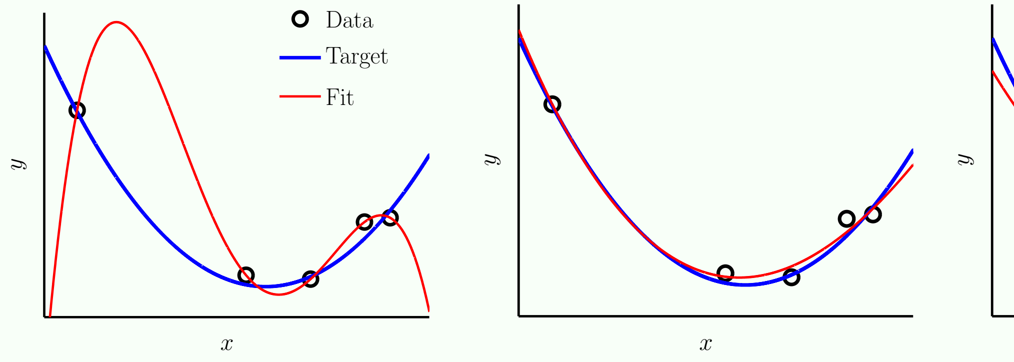
The behavior of E_{out} as a function of the regularization parameter for varying levels of noise:

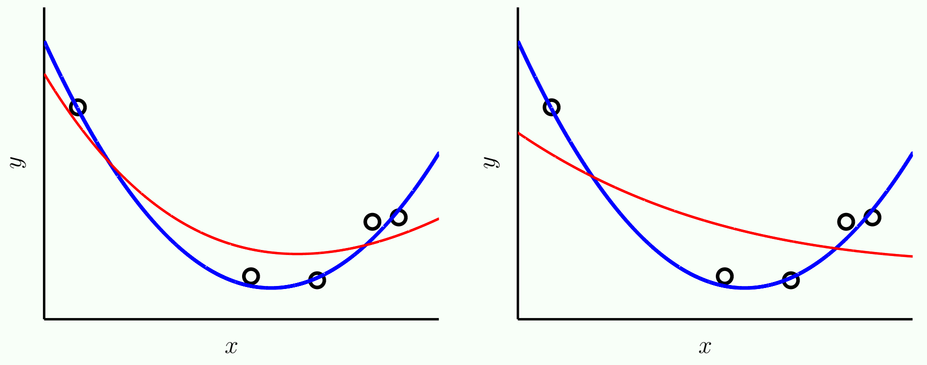


Is there an optimal value for λ?

Minimizing $E_{aug}(\mathbf{w}) = E_{in}(\mathbf{w}) + \lambda \mathbf{w}^\intercal \mathbf{w}$

$\underline{\lambda = 0}$ $\underline{\lambda = 0.0001}$ $\underline{\lambda = 0.01}$ $\underline{\lambda = 1}$





Overfitting $\longrightarrow$ $\longrightarrow$ **Underfitting**

Regularized least-squares

Ridge regression:

$$E_{\text{aug}}(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^{\intercal}(\mathbf{y} - \mathbf{X}\mathbf{w}) + \lambda||\mathbf{w}||^2$$

To solve:

$$\frac{\partial E_{\text{aug}}(\mathbf{w})}{\partial \mathbf{w}} = -2\mathbf{X}^{\intercal}\mathbf{y} + 2\mathbf{X}^{\intercal}X\mathbf{w} + 2\lambda\mathbf{w} = 0$$

$$\mathbf{w} \; = (\mathbf{X}^{\intercal}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^{\intercal}\mathbf{y}$$

Compare to the solution without regularization:

$$\mathbf{w} = (\mathbf{X}^{\intercal}\mathbf{X})^{-1}\mathbf{X}^{\intercal}\mathbf{y}$$

Regularized least-squares

Ridge regression:

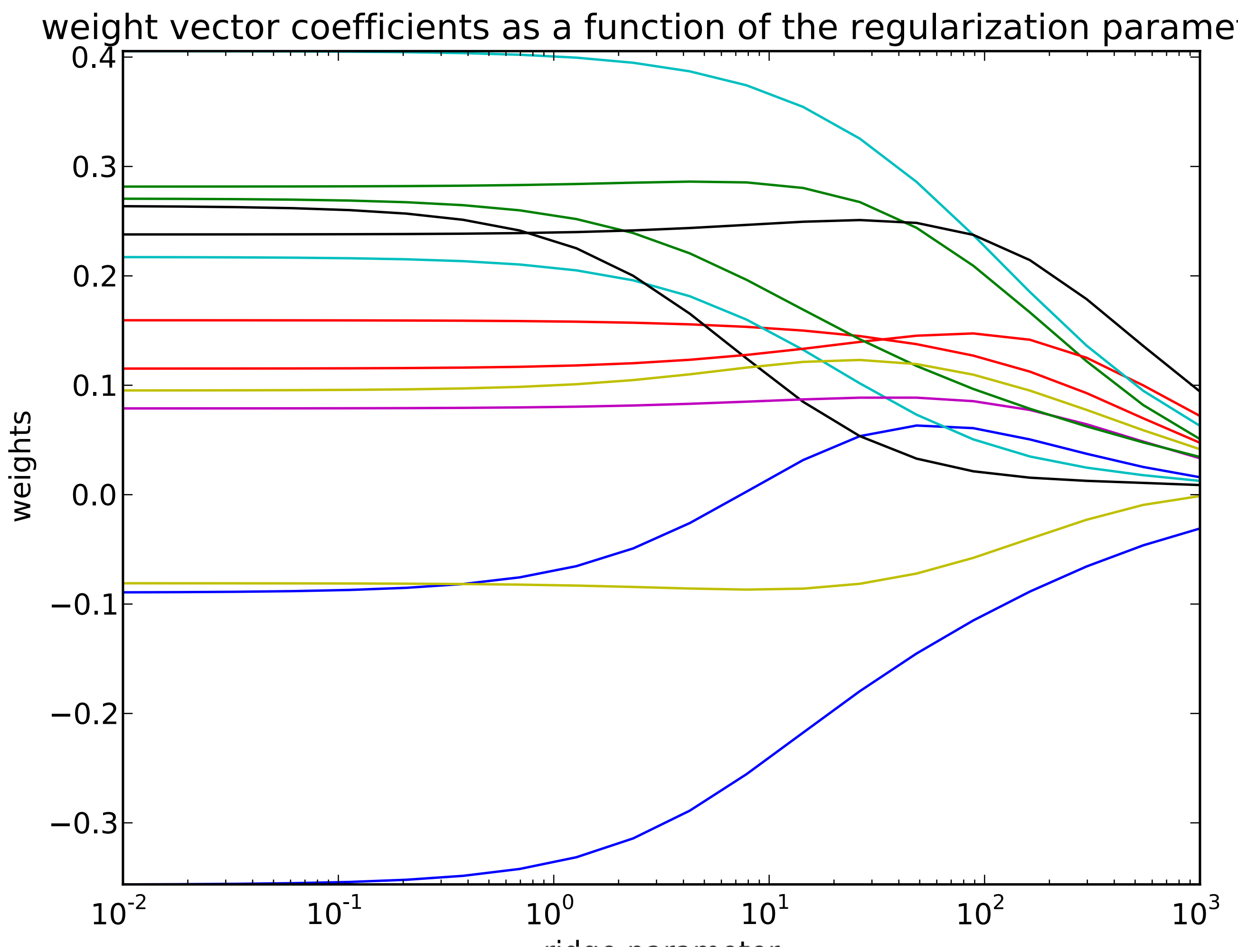
$$E_{\mathrm{aug}}(\mathbf{w}) = (\mathbf{y} - \mathbf{X}\mathbf{w})^{\mathsf{T}}(\mathbf{y} - \mathbf{X}\mathbf{w}) + \lambda||\mathbf{w}||^2$$

There is a tradeoff between fitting (the error term) and regularization. The regularization terms can therefore prevent overfitting. The parameter λ controls this tradeoff.

Many ML methods can be expressed as solution to a cost function of the form:

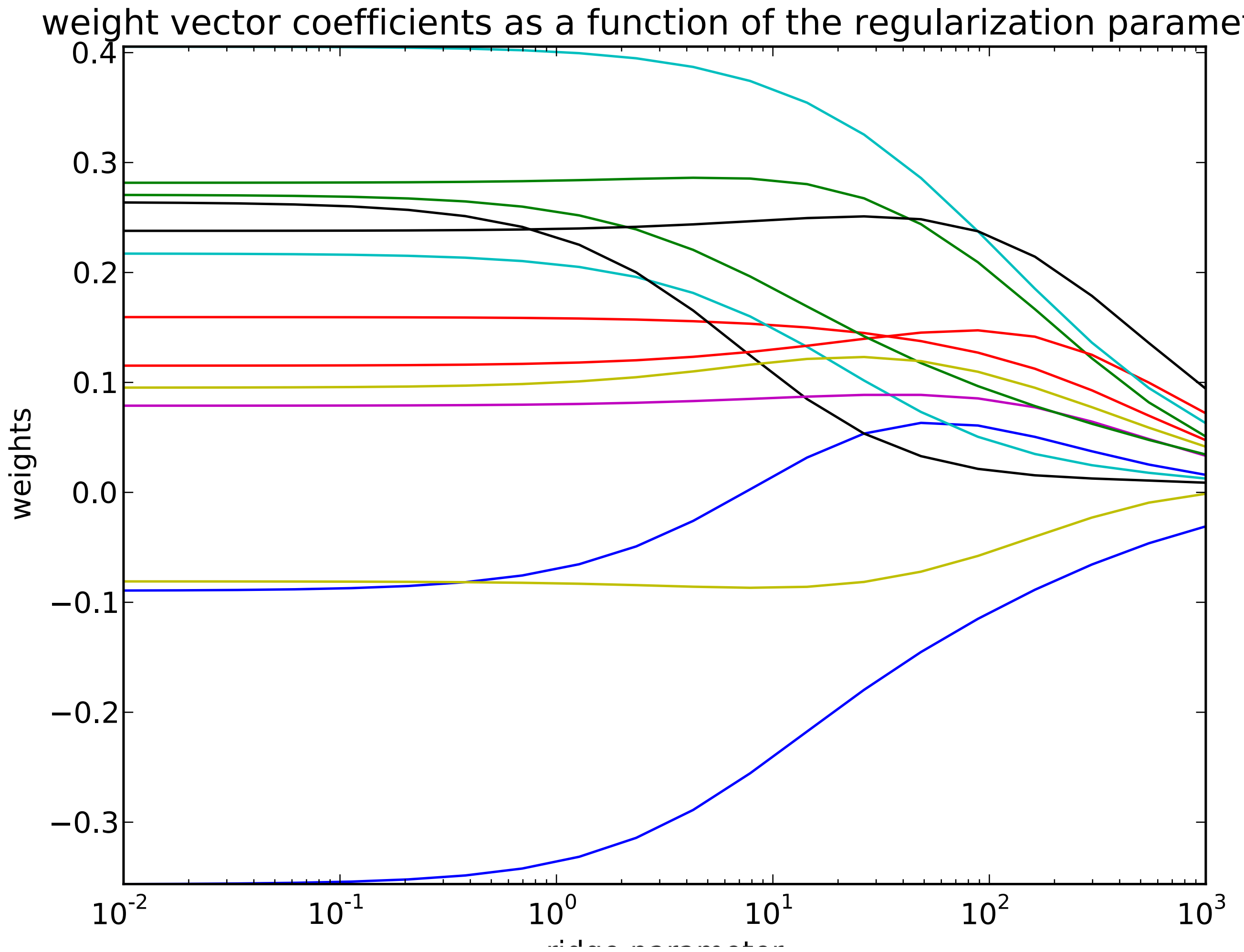
error term + regularization term

The effect of the regularization parameter



Each curve is the magnitude of the weight vector associated with a given feature.
Computed on the scaled version of the "heart" dataset.

The effect of the regularization parameter



As the regularization parameter increases, w_i shrinks toward 0

The validation set

How to choose the value of the regularization parameter?

Take a sneak peak at E_{out} using a validation set!

On a validation set $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_K, y_K)$, the error is $E_{\text{val}}(h) = \frac{1}{K} \sum_{k=1}^{K} \mathbf{e}(h(\mathbf{x}_k), y_k)$

Choosing the size of the validation set

Given the data set $\mathcal{D} = (\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$

$\underbrace{K \text{ points}}_{\mathcal{D}_{\text{val}}} \rightarrow$ validation $\qquad \underbrace{N - K \text{ points}}_{\mathcal{D}_{\text{train}}} \rightarrow$ training

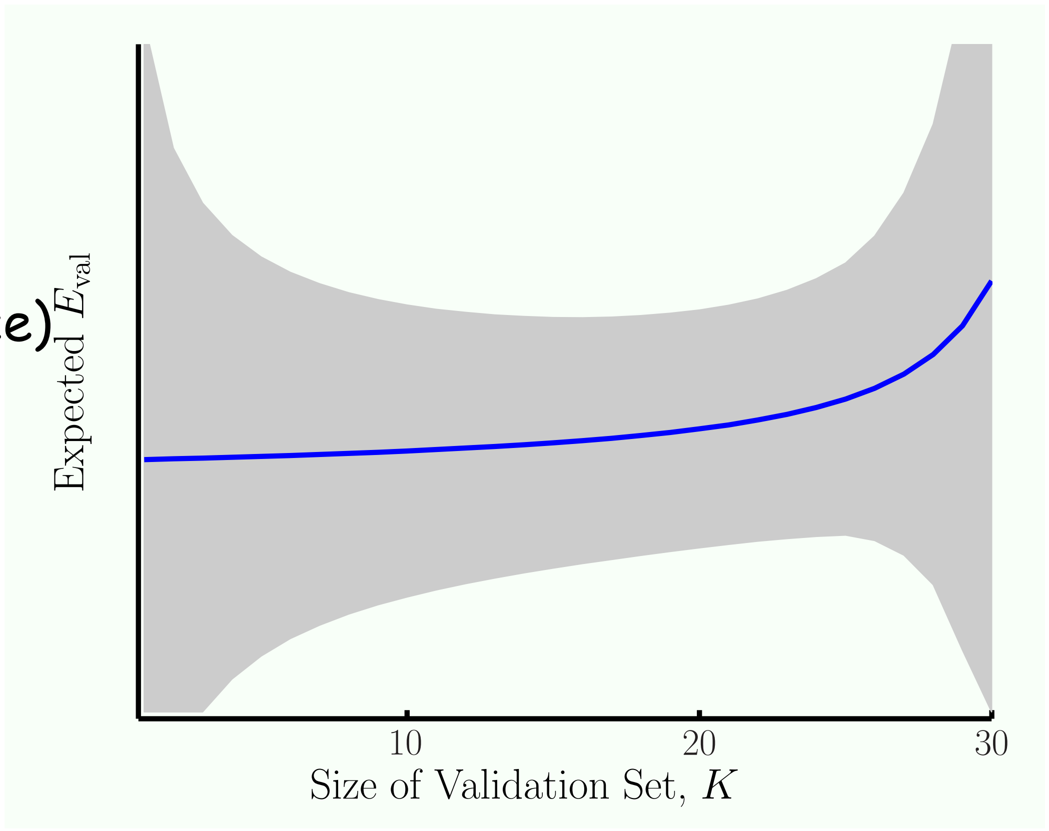
Small $K \implies$ bad estimate

Large $K \implies$?

Rule of thumb: use 20% of the data for validation

Choosing the size of the validation set

Shaded region:
the uncertainty (variance)
of the estimate



Observations:

- As we increase the size of the validation set, the estimate goes up because of a small training set
- The uncertainty in E_{val} decreases as we increase K, up to a point, where a small training set size generates uncertainty in the estimate

Using the validation set

The validation set is used to get estimates that allow us to choose a value for the regularization parameter.

M models $\mathcal{H}_1, \ldots, \mathcal{H}_M$

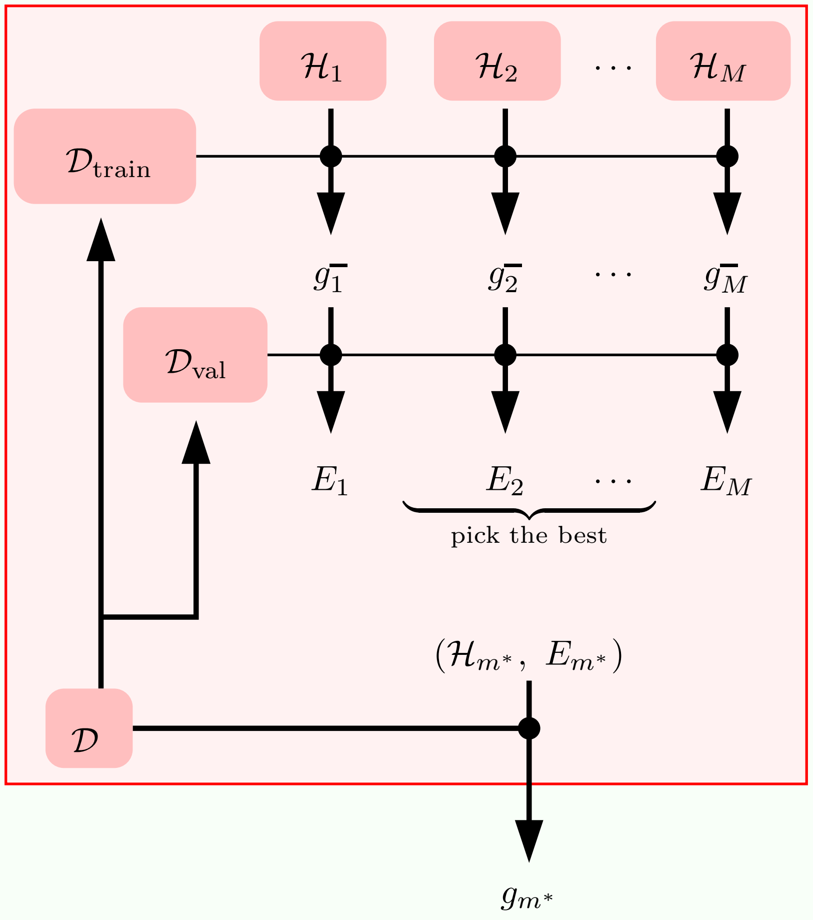
Use $\mathcal{D}_{\text{train}}$ to learn g_m^- for each model

Evaluate g_m^- using $\mathcal{D}_{\text{val}}$:

$$E_m = E_{\text{val}}(g_m^-); \quad m = 1, \ldots, M$$

Pick model $m = m^*$ with smallest E_m

At the end: train a model on all the data using the parameters of H_{m^*}.



We have a dilemma…

We would like to have the following:

$$E_{\text{out}}(g) \approx E_{\text{out}}(g^{-}) \approx E_{\text{val}}(g^{-})$$

(small K) (large K)

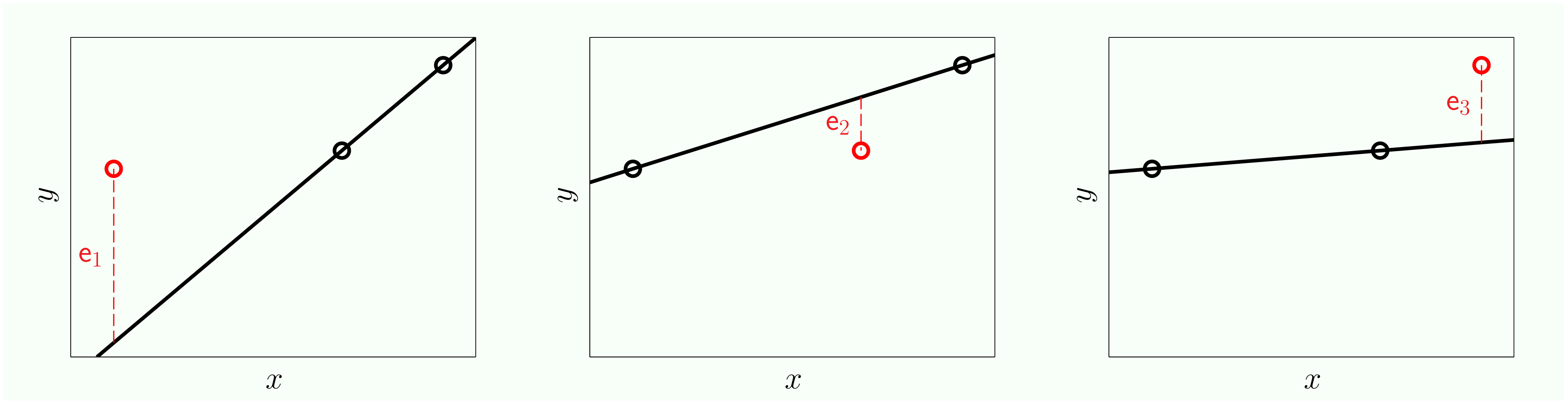
g : the model as a result of training on all the data

g^{-}: the model trained on D_{train}

Can we have K both large and small?

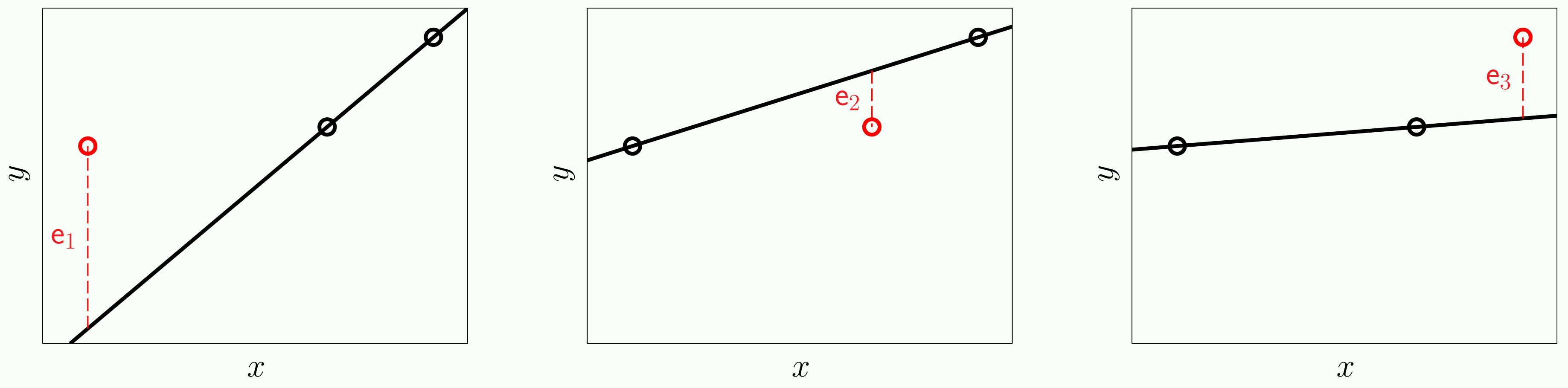
Leave-one-out errors

Extreme case: K=1



The leave-one-out estimate

Extreme case: K=1



$$E_{\mathrm{cv}} = \frac{1}{N} \sum_{n=1}^{N} \mathrm{e}_n$$

Theorem. E_{cv} is an unbiased estimate of $\bar{E}_{\mathrm{out}}(N-1)$.

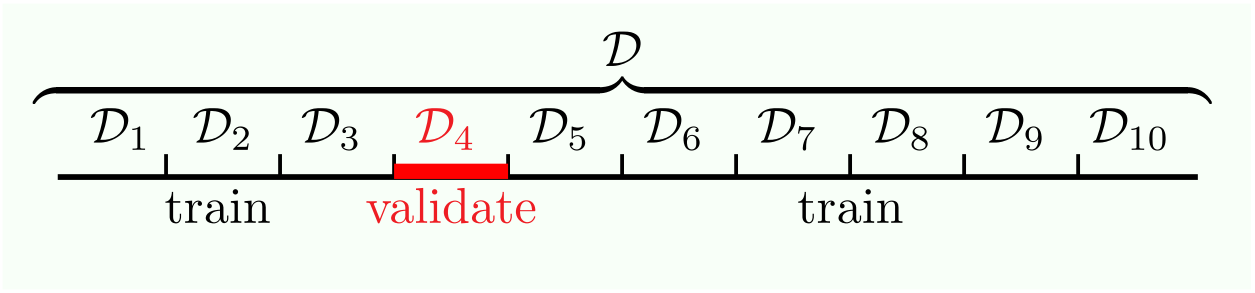
Expected E_{out} when learning with $N-1$ points.

Cross validation

The leave-one-out estimate is expensive to compute!

Cross validation:

- Randomly partition the data into k parts ("folds").
- Set one fold aside for evaluation and train a model on the remaining k-1 folds and evaluate it on the held-out fold.
- Repeat until each fold has been used for evaluation

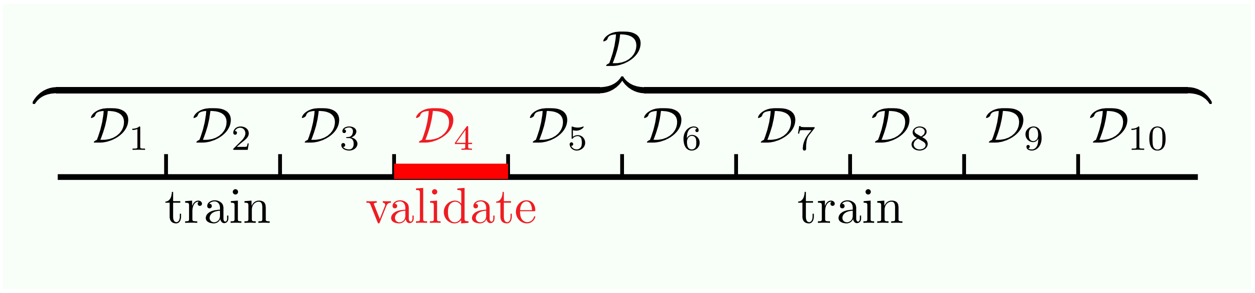

$\mathcal{D}$
$\mathcal{D}_1$ $\mathcal{D}_2$ $\mathcal{D}_3$ $\mathcal{D}_4$ $\mathcal{D}_5$ $\mathcal{D}_6$ $\mathcal{D}_7$ $\mathcal{D}_8$ $\mathcal{D}_9$ $\mathcal{D}_{10}$
train
validate
train

Cross validation

The leave-one-out estimate is expensive to compute!

Cross validation:

- Randomly partition the data into k parts ("folds").
- Set one fold aside for evaluation and train a model on the remaining k-1 folds and evaluate it on the held-out fold.
- Repeat until each fold has been used for evaluation


$\mathcal{D}$
$\mathcal{D}_1$ $\mathcal{D}_2$ $\mathcal{D}_3$ $\mathcal{D}_4$ $\mathcal{D}_5$ $\mathcal{D}_6$ $\mathcal{D}_7$ $\mathcal{D}_8$ $\mathcal{D}_9$ $\mathcal{D}_{10}$
train
validate
train

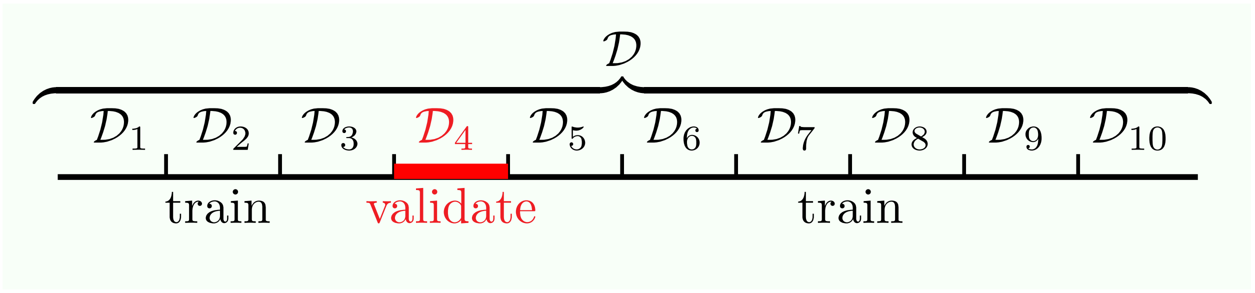
- The reported error is the average over the errors for each fold.

Cross validation

The leave-one-out estimate is expensive to compute!

Cross validation:

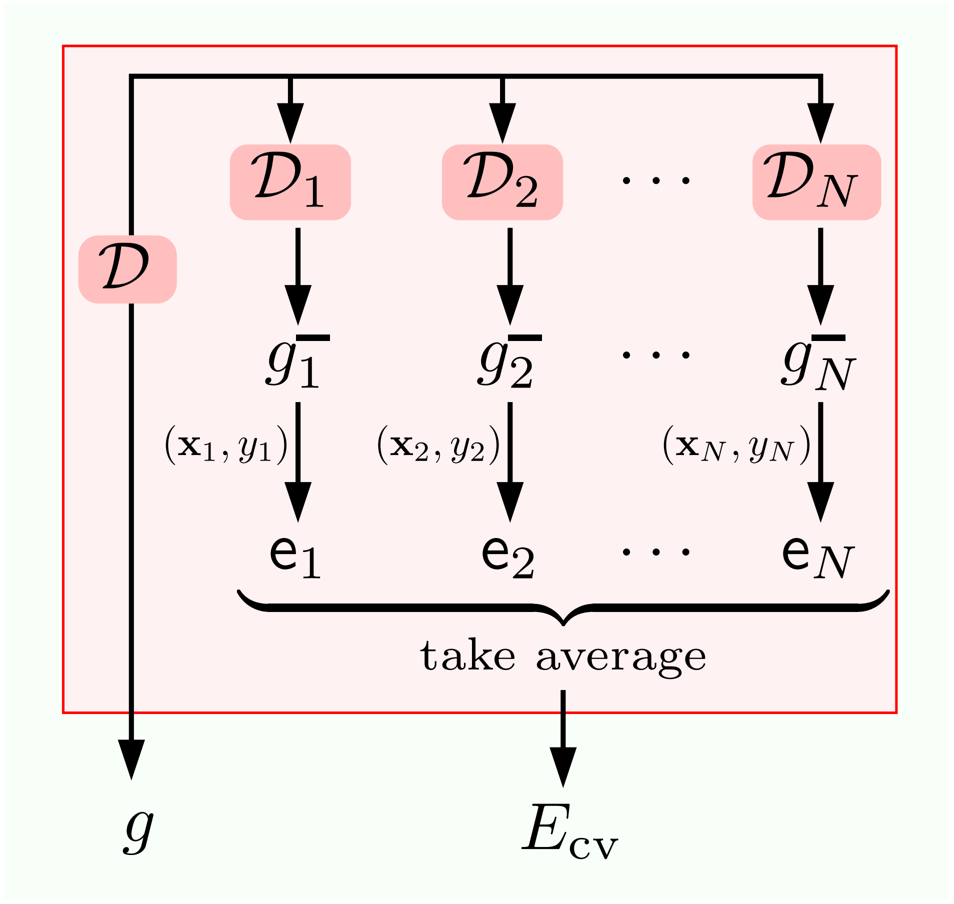
- Randomly partition the data into k parts ("folds").
- Set one fold aside for evaluation and train a model on the remaining k-1 folds and evaluate it on the held-out fold.
- Repeat until each fold has been used for evaluation



$\mathcal{D}$
$\mathcal{D}_1$ $\mathcal{D}_2$ $\mathcal{D}_3$ $\mathcal{D}_4$ $\mathcal{D}_5$ $\mathcal{D}_6$ $\mathcal{D}_7$ $\mathcal{D}_8$ $\mathcal{D}_9$ $\mathcal{D}_{10}$
train
validate
train

Stratified-cross validation aims at achieving roughly the same class distribution in each fold.

Using cross-validation



customer

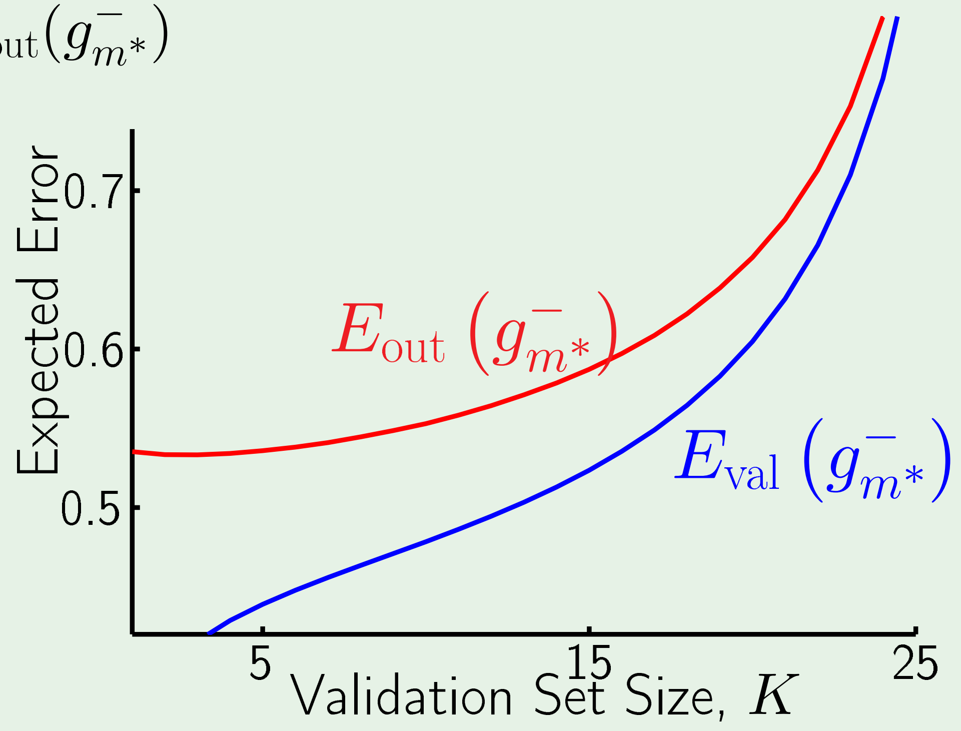
select the best model

Bias

The error estimates using the validation set are optimistic estimates of E_{out}!

We selected the model $\mathcal{H}_{m^*}$ using $\mathcal{D}_{\text{val}}$

$E_{\text{val}}(g^-_{m^*})$ is a biased estimate of $E_{\text{out}}(g^-_{m^*})$



Bias

The error estimates using the validation set are optimistic estimates of E_{out}!

We selected the model $\mathcal{H}_{m^*}$ using $\mathcal{D}_{\text{val}}$

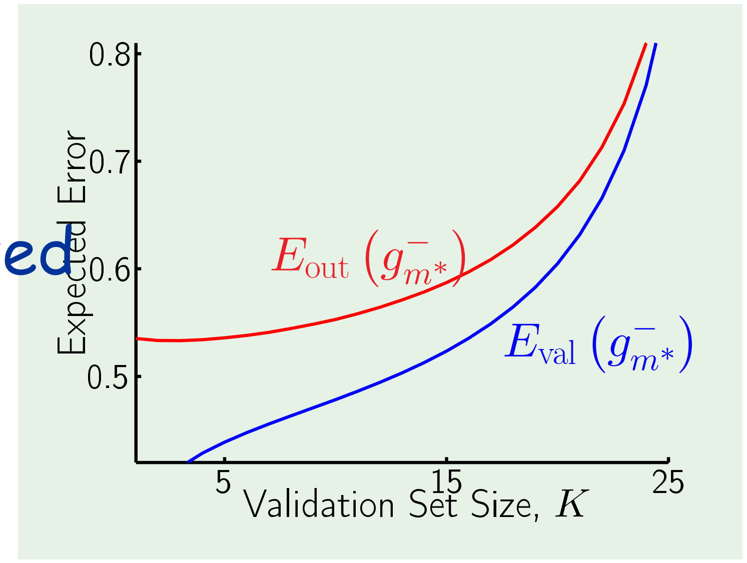
$E_{\text{val}}(g^-_{m^*})$ is a biased estimate of $E_{\text{out}}(g^-_{m^*})$

So you need to have a separate test set.

Training set: totally contaminated

Validation set: slightly contaminated

Test set: "clean"


Expected Error
$E_{\text{out}}(g^-_{m^*})$
$E_{\text{val}}(g^-_{m^*})$
Validation Set Size, K