FSAN/ELEG815: Statistical Learning

Gonzalo R. Arce

Department of Electrical and Computer Engineering
University of Delaware

6: Overfitting and Regularization

Outline of the Course

1. Review of Probability
2. Eigen Analysis, Singular Value Decomposition (SVD) and Principal Component Analysis (PCA)
3. The Learning Problem and the VC Dimension
4. Training vs Testing
5. Nonlinear Transformation and Logistic Regression
6. Overfitting and Regularization (Ridge Regression)
7. Lasso Regression
8. Neural Networks
9. Convolutional Neural Networks

Approximation- Generalization Tradeoff

Balance between approximating f in the training data and generalizing on new data.

Goal: small $E_{out} \rightarrow$ good approximation of f out of sample.

More complex $\mathcal{H} \implies$ better chance of **approximating** f

Less complex $\mathcal{H} \implies$ better chance of **generalizing** out of sample

A more complex $\mathcal{H}$ better approximates f, however, it might be more difficult for the algorithm to zoom in on the right hypothesis.

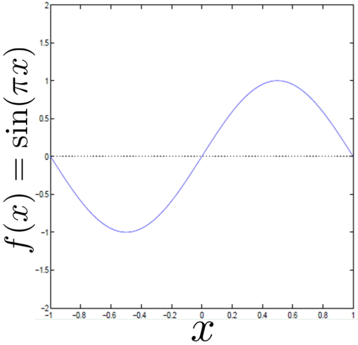
The ideal $\mathcal{H}$ is a singleton hypothesis set containing only the target function.

$$\mathcal{H} = \{f\} \equiv \text{Wining the lottery!}$$

Example: Sine Target

$f : [-1,1] \to \mathbb{R} \quad f(x) = \sin(\pi x)$ **unknown**

We sample x uniformly in $[-1,1]$ to generate two training samples $(N = 2)$



Two models used for learning:

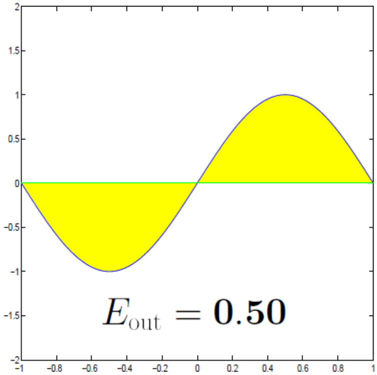
$$\begin{aligned} \mathcal{H}_0 : &\quad h(x) = b \\ \mathcal{H}_1 : &\quad h(x) = ax + b \end{aligned}$$

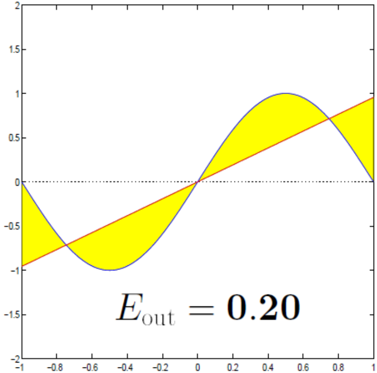
Which is better, $\mathcal{H}_0$ or $\mathcal{H}_1$?

Approximation - $\mathcal{H}_0$ versus $\mathcal{H}_1$

Based on the two models and assuming we know f, try to find the two functions that minimize the squared error:

$$\mathcal{H}_0 : h(x) = b \qquad\qquad \mathcal{H}_1 : h(x) = ax + b$$

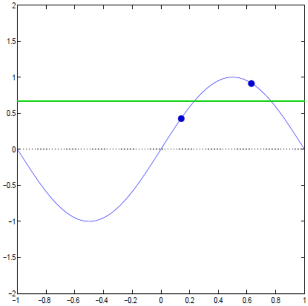




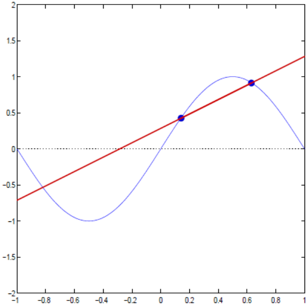
Learning - $\mathcal{H}_0$ versus $\mathcal{H}_1$

In learning, we do not know f. We use the two examples (x_1, y_1),(x_2, y_2) to learn the two functions that best fits the data.

$\mathcal{H}_0$: midpoint $\left(b = \frac{y_1+y2}{2}\right)$



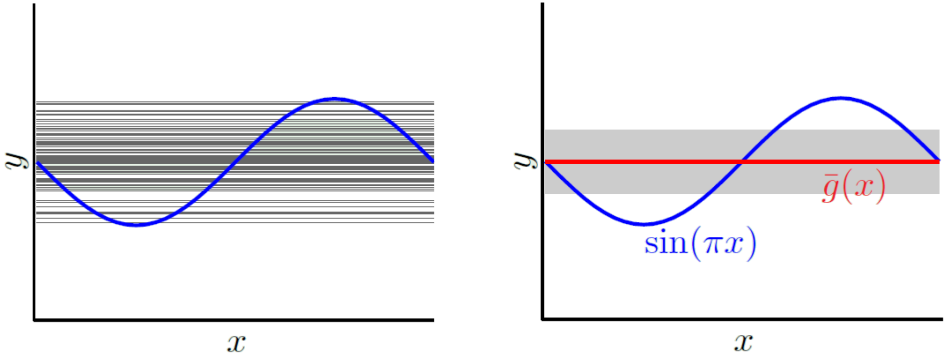
$\mathcal{H}_1$: line passes through the two points



The result varies depending on the data points. We need bias-variance analysis to evaluate our result (considering other possible data sets).

Bias and Variance - $\mathcal{H}_0$

Repeating the process with many data sets, we can estimate the bias and the variance.



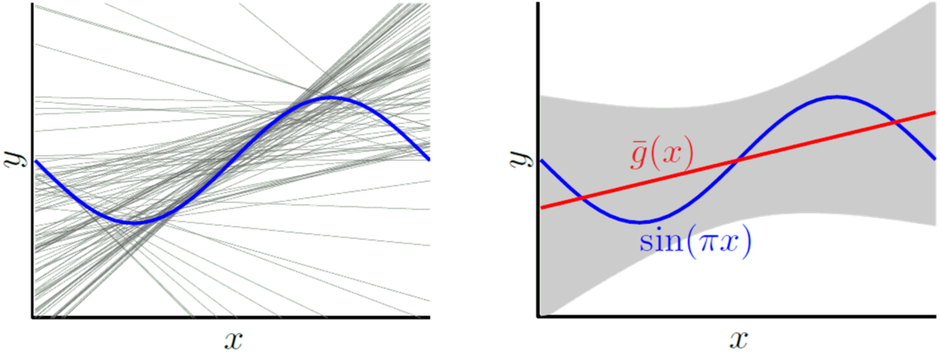
Average hypothesis $\bar{g}(x)$. In this case $\bar{g}(x) \approx 0$ that is close to the best approximation computed using f.

bias: difference between red function $\bar{g}(x)$ and blue function f.

var(x) is indicated by the gray shaded region that is $\bar{g}(x) \pm \sqrt{\textbf{var}(x)}$

Bias and Variance - $\mathcal{H}_1$

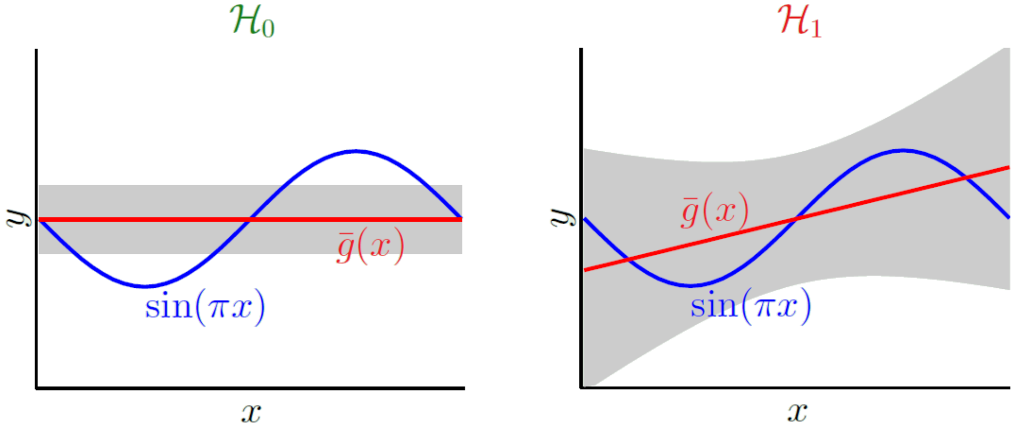
Using the same data sets as before, for the second model we get



bias: difference between red function $\bar{g}(x)$ and blue function f.

var(x) is indicated by the gray shaded region that is $\bar{g}(x) \pm \sqrt{\textbf{var}(x)}$

The Winner is ...



bias = 0.50 var=0.25 bias=0.21 var=1.69

The simpler model wins by significantly decreasing the **var** at the expense of a smaller increase in **bias**

Lesson Learned

However, the **var** term decreases as N increases, so if we get a bigger data set, the **bias** term will be dominant in E_{out}, and $\mathcal{H}_1$ will win.

Match the '**model complexity**'

to the **data resources**, not to the **target complexity**

Outline

- Bias and Variance
- Learning Curves

Expected E_{out} and E_{in}

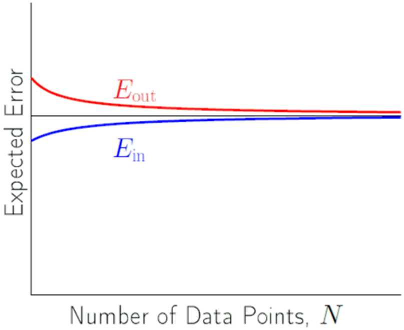
Consider learning with a data set $\mathcal{D}$ of size N,

the final hypothesis has a expected out-of-sample error $\mathbb{E}_{\mathcal{D}}\left[E_{out}(g^{(\mathcal{D})})\right]$ and

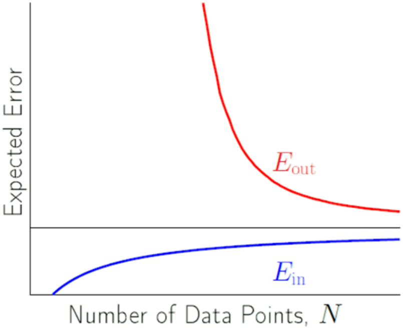
expected in-sample error $\mathbb{E}_{\mathcal{D}}\left[E_{in}(g^{(\mathcal{D})})\right]$

How do they vary with N?

The Curves



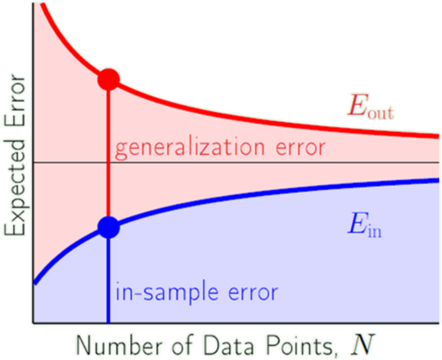
Simple Model



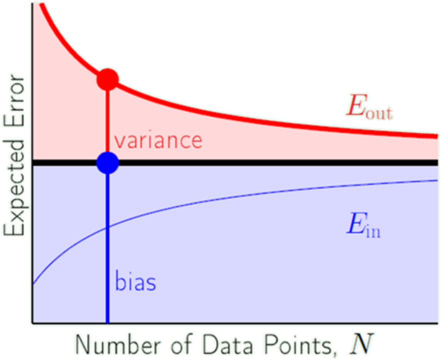
Complex Model

Note: the simple model converges more quickly but to a higher error. In both models, E_{out} decreases while E_{in} increases toward the smallest error the learning model can achieve in approximating f.

VC versus Bias-Variance



VC analysis



bias-variance

In the VC analysis, $E_{out} \leq E_{in} + \Omega$. In the **bias-variance**, it is assumed that, for every N, $\bar{g}$ has the same performance as the best approximation to f in the learning model.

Both capture the tradeoff: Approximation-Generalization

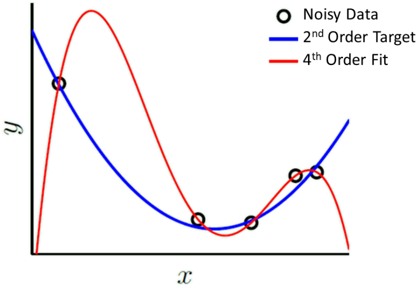
Outline

- What is overfitting?
- The role of noise
- Deterministic noise
- Dealing with overfitting

Illustration of Overfitting

- Simple target function → 2nd order polynomial.
- Generate 5 data points (noisy).
- Solve regression problem → 5 points fit by a 4th order polynomial.

$$E_{in} = 0$$



However, result does not match the target.

The complex model uses additional degrees of freedom to learn noise.

Overfitting: Process of picking a hypothesis with lower E_{in} and higher E_{out}.

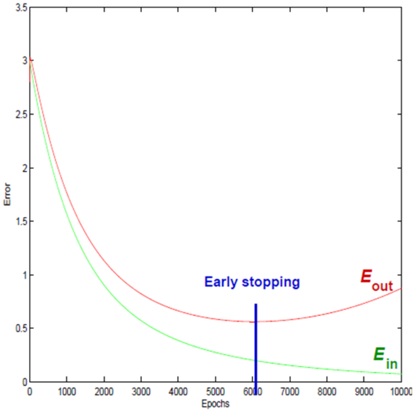
Overfitting vs Bad Generalization

Neural network fitting noisy data:

- Green curve: Running gradient descent and evaluate E_{in} for each epoch.
- Red curve: Use test set to evaluate E_{out} for each epoch.
- Generalization error (difference between the two curves) is increasing.

Overfitting: $E_{in} \downarrow$ $E_{out} \uparrow$

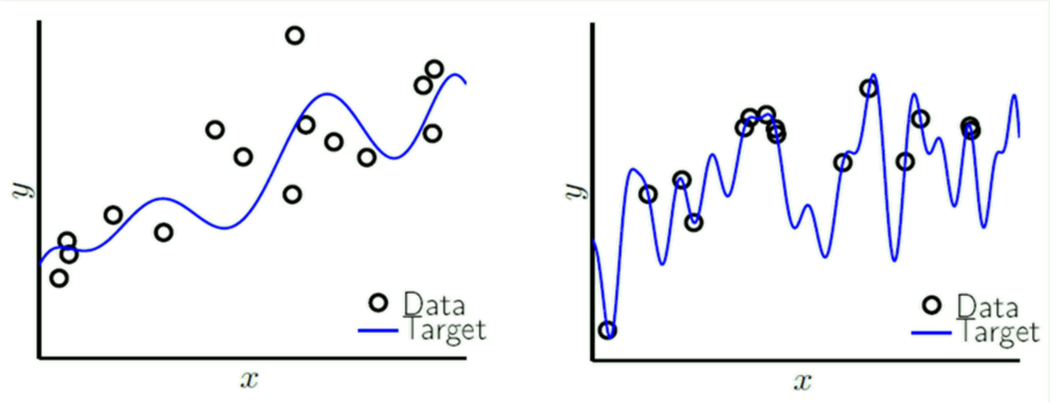
Possible solution: Early stopping



Case Study

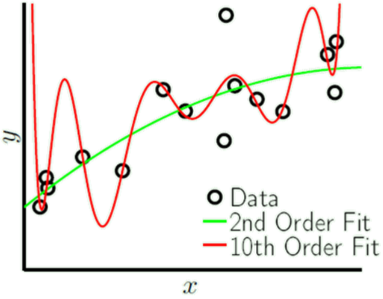
Polynomial regression: $x \rightarrow (1, x, x^2, \cdots)$.

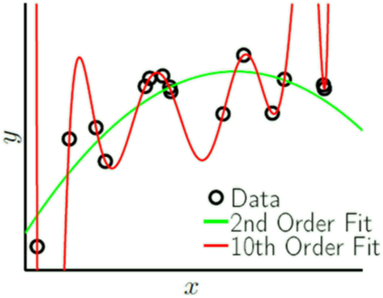
- 10th order target function +noise
- 50th order target function (noiseless)



Data set $\mathcal{D}$ contains 15 data points.

Two Fits for Each Target





Noisy low-order target (10th)

	2nd Order	10th Order
E_{in}	0.05	0.034
E_{out}	0.127	9.00

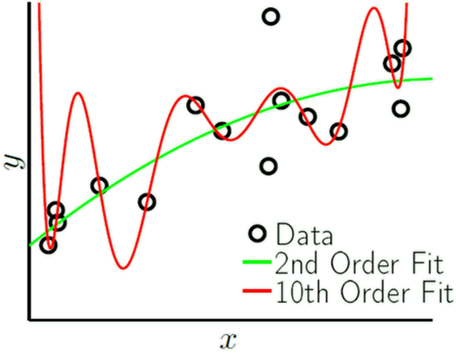
Noiseless high-order target (50th)

	2nd Order	10th Order
E_{in}	0.029	10^{-5}
E_{out}	0.120	7680

The 10th order polynomial heavily overfits the data.

An Irony of Two Learners

- Two learners O and R
- They know the target is 10th order.
- O chooses $\mathcal{H}_{10}$
- R chooses $\mathcal{H}_2$.
  - Give up implementing the true target function.
  - Best you can do considering # data points ($N \geq 10d_{\text{VC}}$)

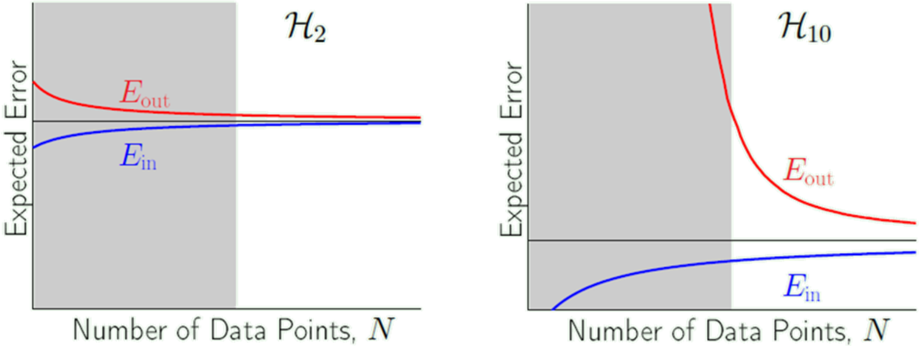


Learning a 10th-order target (noisy)

Match the resources, rather than the target complexity.

Irony: The belief that the best results are obtained by incorporating as much information about the target function as it is available.

Learning Curves



Overfitting is occurring in the shaded region by choosing $\mathcal{H}_{10}$ which has better E_{in} but worse E_{out}.

What matters is how the model complexity matches quantity and quality of the data, instead of only matching the target function.

A Detailed Experiment

Goal: Study impact of **noise level** σ^2, **target complexity** Q_f and **number of data points** N.

$$y = f(x) + \underbrace{\epsilon(x)}_{\sigma^2} = \sum_{q=0}^{Q_f} a_q L_q(x) + \epsilon(x)(*)$$

where $\epsilon(x)$ are iid standard Normal random variables.

Interesting targets → $L_i(x)$: increasing complexity polynomials (Legendre polynomials $^{(*)}$). a_q's selected independently from a standard Normal.

$$y = \underbrace{\sum_{q=0}^{Q_f} \alpha_q x^q}_{\text{normalized}} + \epsilon(x) \qquad \alpha_q : \text{ sum of coefficients paired with } x^q$$

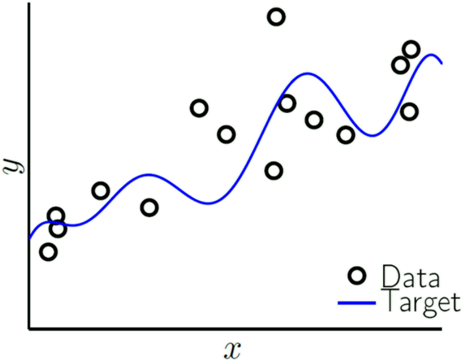
Rescale α_i's so that $\mathbb{E}_{\alpha,x}[f^2] = 1$

$^{(*)}$A Legendre polynomial $L_i(x)$ has specific coefficients such that they are orthogonal.

A Detailed Experiment

Goal: Study impact of **noise level** σ^2, **target complexity** Q_f and **number of data points** N.

Example



$$y = f(x) + \underbrace{\epsilon(x)}_{\sigma^2} = \sum_{q=0}^{Q_f} \alpha_q x^p + \epsilon(x)$$

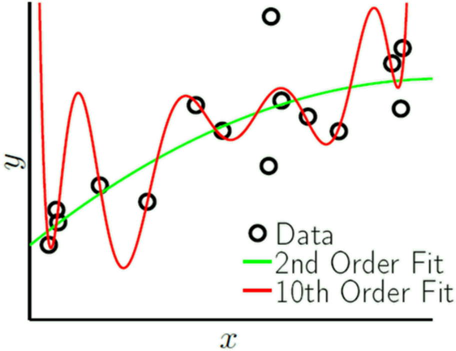
- Noise level: σ^2
- Target complexity: $Q_f = 10$
- Data size: $N = 15$

The Results

Fit the data set $(x_1, y_1), \cdots, (x_N, y_N)$ using our two models:

$\mathcal{H}_2$: 2nd-order polynomials $\mathcal{H}_{10}$: 10th-order polynomials

Target: 10th order polynomial (noisy)



Compare out-of-sample errors of

- $g_2 \in \mathcal{H}_2$
- $g_{10} \in \mathcal{H}_{10}$

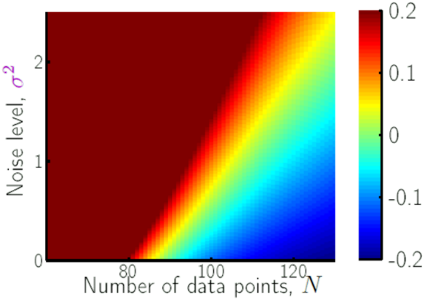
Overfit Measure:

$E_{out}(g_{10}) - E_{out}(g_2)$

More positive → More overfitting

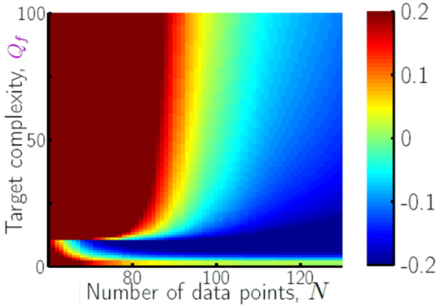
The Results

The colors map to overfit measure: $E_{out}(g_{10}) - E_{out}(g_2)$



Impact of σ^2

Less overfitting when σ^2 drops or N increases ($Q_f = 20$).



Impact of Q_f

Less overfitting when Q_f drops or N increases ($\sigma^2 = 0.1$).

Number of Data Points	↑	Overfitting	↓
Noise	↑	Overfitting	↑
Target Complexity	↑	Overfitting	↑

Definition of Deterministic Noise (DN)

Part of f that $\mathcal{H}$ cannot capture: $f(\mathbf{x}) - h^*(\mathbf{x})$

Why called "noise"?

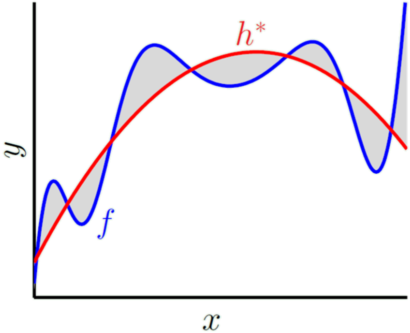
Similarities with stochastic noise:

- It cannot be modeled.
- Trying to learn model it results in overfitting and a spurious final hypothesis.

Differences with stochastic noise:

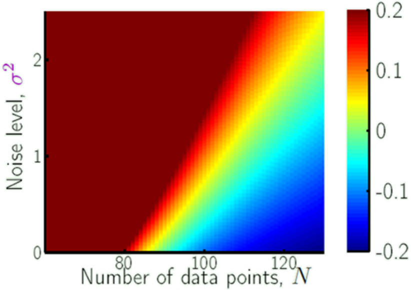
- DN depends on $\mathcal{H}$ (↑ Complexity ↓ DN)
- DN is fixed for a given $\mathbf{x}$.

For a given learning model, there is a best approximation h^* to the target function f.

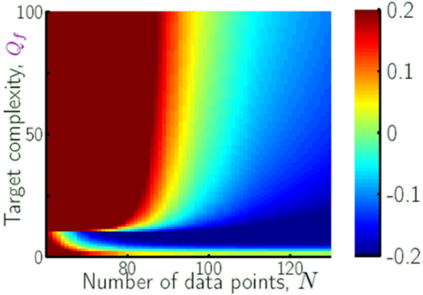


Shading area: Deterministic noise.

Impact of “Noise”



Stochastic noise



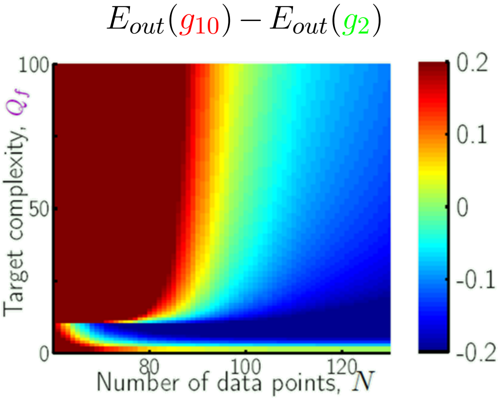
Deterministic noise

Number of Data Points	↑	Overfitting	↓
Stochastic Noise	↑	Overfitting	↑
Deterministic Noise	↑	Overfitting	↑

Deterministic Noise- Impact on Overfitting

Deterministic noise and target complexity Q_f

- As Q_f increases, deterministic noise increases.
- Why overfit starts at $Q_f = 10$? $\mathcal{H}_{10}$ cannot completely capture targets of order greater than 10 (Deterministic Noise).
- For a finite N: $\mathcal{H}$ tries to fit stochastic and deterministic noise.



How much overfit

Noise and Bias-Variance

For f a noiseless target:

$$\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - f(\mathbf{x}))^2] = \underbrace{\mathbb{E}_{\mathcal{D}}[(g^{(\mathcal{D})}(\mathbf{x}) - \bar{g}(\mathbf{x}))^2]}_{\text{var}(\mathbf{x})} + \underbrace{(\bar{g}(\mathbf{x}) - f(\mathbf{x}))^2}_{\text{bias}(\mathbf{x})}$$

- The best approximation h^* to the target function f is approximately the 'average' hypothesis $\bar{g}$.
- What if f is a noisy target?

$$y = f(\mathbf{x}) + \epsilon(\mathbf{x}) \qquad \mathbb{E}[\epsilon(\mathbf{x})] = 0$$

Actually, Two Noise Terms

$$\mathbb{E}_{\mathcal{D},\mathbf{x},\epsilon}[(g^{(\mathcal{D})}(\mathbf{x})-y)^2] \;=\; \underbrace{\mathbb{E}_{\mathcal{D},\mathbf{x}}[(g^{(\mathcal{D})}(\mathbf{x})-\bar{g}(\mathbf{x}))^2]}_{\mathsf{var}(\mathbf{x})}+\underbrace{\mathbb{E}_{\mathbf{x}}[(\bar{g}(\mathbf{x})-f(\mathbf{x}))^2]}_{\mathsf{bias}(\mathbf{x})}+\underbrace{(\epsilon(\mathbf{x}))^2}_{\sigma^2}$$

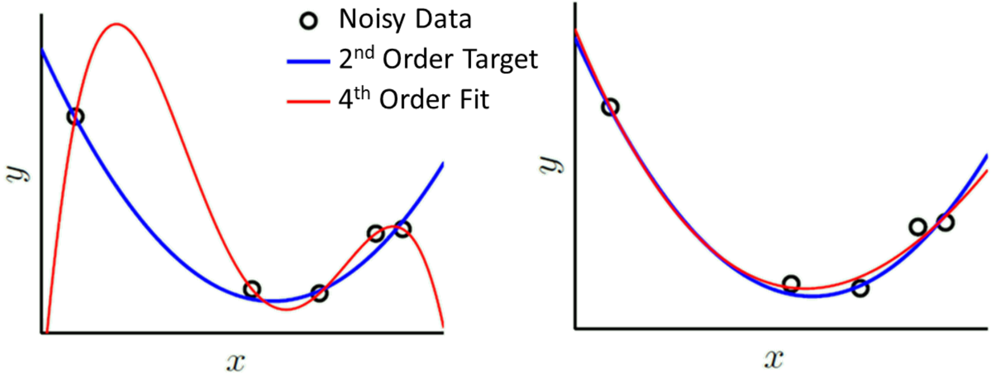
- $\sigma^2 \rightarrow$ Stochastic Noise
- **bias** $\rightarrow$ Deterministic Noise
  Captures model's inability to approximate f.
- **var** $\rightarrow$ Variance of the model
  Captures model's susceptibility to being led in the wrong direction by the two types of noise.

Size of set $N \uparrow$ **var** $\downarrow$.

Given a hypothesis set $\mathcal{H}$, **bias** and σ^2 are fix (irreducible error).

Dealing with Overfitting

- **Regularization:** Putting the brakes.
- **Validation:** Checking the bottom line.



free fit

restrained fit

Two Approaches to Regularization

- **Mathematical**:
  - Ill-posed problems in function approximation (solved by smoothness constrains).
  - Bayesian Approach (prior knowledge). Assumptions might not be realistic
- **Heuristic:**
  - Constraining on the minimization of E_{in}

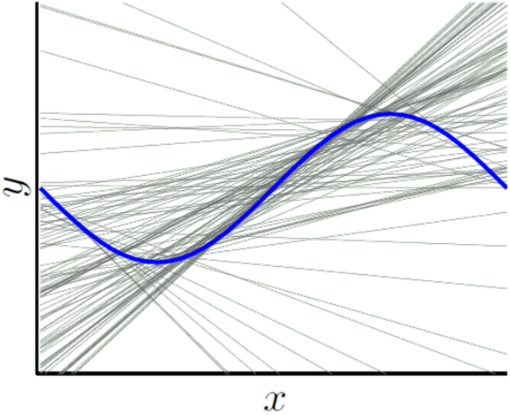
A Familiar Example

$f : [-1,1] \rightarrow \mathbb{R} \quad f(x) = \sin(\pi x)$ **unknown**

We sample x uniformly in $[-1,1]$ to generate two training samples ($N = 2$)

Two models used for learning:

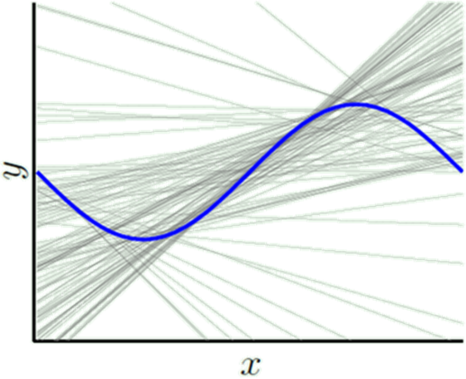
$$\begin{aligned} \mathcal{H}_0 : &\quad h(x) = b \\ \mathcal{H}_1 : &\quad h(x) = ax + b \end{aligned}$$



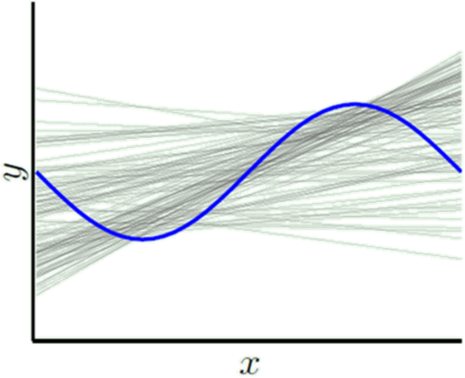
Which was better, $\mathcal{H}_0$ or $\mathcal{H}_1$?

$\mathcal{H}_0$ beats $\mathcal{H}_1$

A Familiar Example



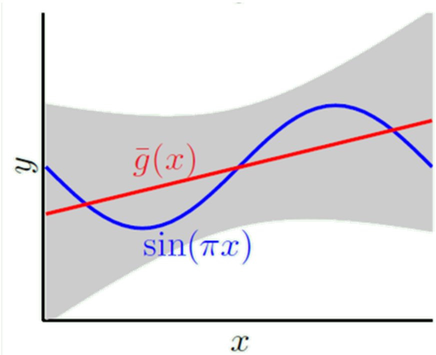
Without Regularization:
Learned function varies extensively depending on the data set.



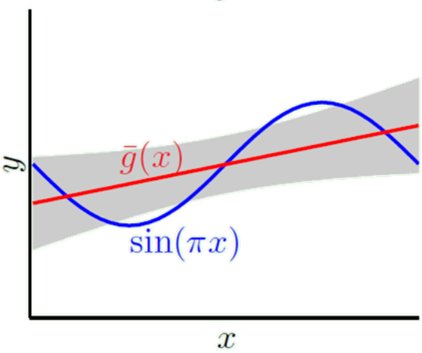
With regularization:
The same data sets are less volatile.

Bias-Variance Decomposition

var(x) gray shaded region ($\bar{g}(x) \pm \sqrt{\textbf{var}(x)}$).



Without Regularization:
bias $= 0.21$ **var** $= 1.69$



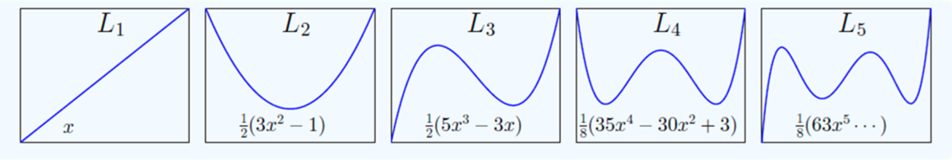
With regularization:
bias $= 0.23$ **var** $= 0.33.$

Regularized $\mathcal{H}_1$ also beats the constant model $\mathcal{H}_0$ (**bias**=0.50, **var**=0.25)

Legendre Polynomials

Standard set of polynomials in one variable $x \in [-1, 1]$ with nice analytic properties:

- Curves get more complex when order increases.
- Orthogonal to each other within $x \in [-1, 1]$.
- Any regular polynomial can be written as a linear combination of Legendre Polynomials.



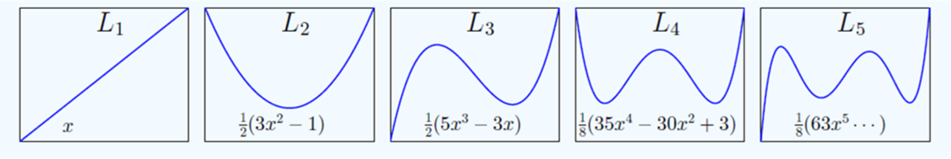
The Polynomial Model

$\mathcal{H}_Q$: polynomials of order Q

$$\mathcal{H}_Q = \left\{ h \middle| h(x) = \mathbf{w}^T\mathbf{z} = \sum_{q=0}^{Q} w_q L_q(x) \right\}_{\mathbf{w}\in\mathbb{R}^{Q+1}}$$

where $\mathbf{z} = [1, L_1(x), \ldots L_Q(x)]^T$ (L_q: Legendre Polynomials).

Using Legendre Polynomials, coefficients w_q can be treated as independent (dealing with orthogonal coordinates).



Note: h is linear in $\mathbf{w}$ → Apply Linear Regression in $\mathcal{Z}$ space.

Unconstrained Solution

Given $(x_1, y_1), \cdots, (x_N, y_N) \quad \overset{\Phi}{\rightarrow} \quad (\mathbf{z}_1, y_1), \cdots, (\mathbf{z}_1, y_1), \cdots, (\mathbf{z}_N, y_N)$

where $\Phi : \mathcal{X} \rightarrow \mathcal{Z}$ is a nonlinear transformation.

$$\begin{aligned} E_{in}(\mathbf{w}) &= \frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^T\mathbf{z}_n - y_n)^2 \\ &= \frac{1}{N}||\mathbf{Z}\mathbf{w} - \mathbf{y}||_2^2 \\ &= \frac{1}{N}(\mathbf{Z}\mathbf{w} - \mathbf{y})^T(\mathbf{Z}\mathbf{w} - \mathbf{y}) \end{aligned}$$

$$\begin{aligned} \mathbf{w}_{\text{lin}} &= \arg \min_{\mathbf{w}\in\mathbb{R}^{Q+1}} \quad E_{in} \\ &= (\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{Z}^T y \end{aligned}$$

Constraining the Weights

- Hard constraint: $\mathcal{H}_2$ is constrained version of $\mathcal{H}_{10}$ with $w_q = 0$ for $q > 2$

- Softer version: $\sum_{q=0}^{Q} w_q^2 \leq C$ "soft-order" constraint

  It encourages each weight to be small.

  C determines the amount of regularization.

  Larger C, weaker constraint → less regularization.

The optimization problems becomes:

$$\mathbf{w}_{\text{reg}} = \arg\min_{\mathbf{w}} \frac{1}{N}(\mathbf{Zw}-\mathbf{y})^T(\mathbf{Zw}-\mathbf{y}) \qquad \text{subject to:} \qquad \mathbf{w}^T\mathbf{w} \leq C$$

Augmented Error

$$E_{\text{aug}}(\mathbf{w}) = E_{in}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$$

$\mathbf{w}_{\text{reg}} = \arg\min_{\mathbf{w}} \quad E_{in}(\mathbf{w}) + \frac{\lambda}{N}\mathbf{w}^T\mathbf{w}$ unconditionally (Ridge Regression)

Solves:

$$\mathbf{w}_{\text{reg}} = \arg\min_{\mathbf{w}} \frac{1}{N}(\mathbf{Zw}-\mathbf{y})^T(\mathbf{Zw}-\mathbf{y}) \qquad \text{subject to:} \qquad \mathbf{w}^T\mathbf{w} \leq C$$

$$C\uparrow \qquad \lambda\downarrow$$

- $\lambda = 0 \implies C \to \infty$ Least Squares Solution
- $\lambda = \infty \implies C = 0$ $\mathbf{w}_{\text{reg}} = 0$

Ridge Regression

Given the data set $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$, Ridge regression shrinkage fit minimizes a penalized residual sum of squares,

$$\begin{aligned}\hat{\mathbf{w}}^{ridge} &= \arg\min_{\mathbf{w}\in\mathbb{R}^d}\left[\sum_{i=1}^{N}(y_i - w_0 - \sum_{j=1}^{d} x_{ij}w_j)^2 + \lambda\sum_{j=1}^{d} w_j^2\right] \\ &= \arg\min_{\mathbf{w}\in\mathbb{R}^d}\left[\underbrace{||\mathbf{y} - w_0 - \mathbf{X}\mathbf{w}||_2^2}_{\text{Loss}} + \underbrace{\lambda||\mathbf{w}||_2^2}_{\text{Penalty}}\right],\end{aligned}$$

where $||\mathbf{w}||_2$ is the ℓ_2 norm $||\mathbf{w}||_2 = \sqrt{\sum_{j=1}^{d} w_j^2}$.
Here $\lambda \geq 0$ is a tuning parameter, which controls the strength of the penalty term.

- When $\lambda = 0$, we get the linear regression estimate.
- When $\lambda = \infty$, we get $\mathbf{w}^{ridge} = 0$.

Ridge Regression

- For λ in between, we balance two ideas: a linear model of **y** on **X**, and shrinking the coefficients.
  Given

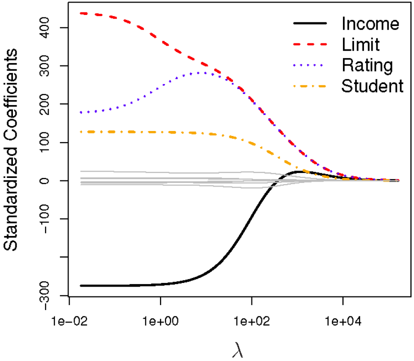
$$\mathbf{y} = w_0 + w_1\mathbf{x}_1 + w_2\mathbf{x}_2 + w_3\mathbf{x}_3 + \ldots + w_{d-1}\mathbf{x}_{d-1} + w_d\mathbf{x}_d + \epsilon.$$

- If the columns of $\mathbf{X}$ are centered, then the intercept estimate is $\hat{w}_0 = \bar{y}$, so we usually assume that $\mathbf{y}$, $\mathbf{X}$ have been centered (zero mean) and don't include an intercept.
- The penalty term $||\mathbf{w}||_2^2$ is unfair if the predictor variables are not on the same scale. Variables are not measured in the same units, we typically scale the columns of $\mathbf{X}$ (to have sample variance 1), and then perform ridge regression.

Ridge Regression

Credit data set: **balance** (average credit card debt for a number of individuals), **age**, **cards** (number of credit cards), **education** (years of education), **income** (in thousand of dollars), **limit** (credit limit), and **rating** (credit rating).

Each curve corresponds to estimate for one of the seven variables.



- As $\lambda \uparrow$, the ridge estimates $\hat{w}_k \rightarrow 0$.

Ridge Regression

The penalized residual sum of squares (PRSS):

$$\begin{aligned} PRSS &= (\mathbf{y}-\mathbf{X}\mathbf{w})^T(\mathbf{y}-\mathbf{X}\mathbf{w}) + \lambda||\mathbf{w}||_2^2 \\ PRSS &= \mathbf{y}^T\mathbf{y} - \mathbf{w}^T\mathbf{X}^T\mathbf{y} + \mathbf{y}\mathbf{X}\mathbf{w} - \mathbf{w}^T\mathbf{X}^T\mathbf{X}\mathbf{w} + \lambda\mathbf{w}^T\mathbf{w} \end{aligned}$$

Differentiating with respect to **w**, we obtain,

$$\begin{aligned} \frac{\partial PRSS}{\partial \mathbf{w}} &= -2\mathbf{X}^T(\mathbf{y}-\mathbf{X}\mathbf{w}) + 2\lambda\mathbf{w} \\ \frac{\partial PRSS}{\partial \mathbf{w}} &= -2\mathbf{X}^T\mathbf{y} - 2\mathbf{X}^T\mathbf{X}\mathbf{w} + 2\lambda\mathbf{w} \end{aligned}$$

PRSS(**w**) is convex. Set the first derivative to zero,

$$\lambda\mathbf{w} = \mathbf{X}^T(\mathbf{y}-\mathbf{X}\mathbf{w}) \tag{1}$$

The ridge regression solution is

$$\hat{\mathbf{w}}^{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}. \tag{2}$$

- Inclusion of λ makes problem non-singular even if $\mathbf{X}^T\mathbf{X}$ is not invertible.

Ridge Regression

The ridge regression estimate

$$\hat{\mathbf{w}}^{ridge} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}.$$

- Solution indexed by the parameter λ
- For each shrinkage λ value, we have a solution.(path of solutions).
- λ controls the size of the coefficients and the amount of regularization.
- As $\lambda \to 0$, we obtain the LS solutions.
- As $\lambda \to \infty$, we have $\hat{\mathbf{w}}^{\text{ridge}}_{\lambda=\infty} = 0$.

Ridge Regression

Setting $\mathbf{R} = \mathbf{X}^T\mathbf{X}$,

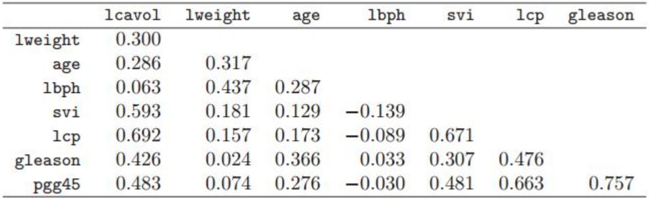
$$
\begin{aligned}
\hat{\mathbf{w}}_{\lambda}^{\text{ridge}} &= (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}^T\mathbf{y} \\
&= (\mathbf{R} + \lambda\mathbf{I}_d)^{-1}\mathbf{R}(\mathbf{R}^{-1}\mathbf{X}^T\mathbf{y}) \\
&= (\mathbf{R}(\mathbf{I}_d + \lambda\mathbf{R}^{-1}))^{-1}\mathbf{R}\underbrace{((\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y})}_{\mathbf{w}^{ls}} \\
&= (\mathbf{I}_d + \lambda\mathbf{R}^{-1})^{-1}\mathbf{R}^{-1}\mathbf{R}\hat{\mathbf{w}}^{ls} \\
&= (\mathbf{I}_d + \lambda\mathbf{R}^{-1})^{-1}\hat{\mathbf{w}}^{ls}
\end{aligned}
$$

- If $\mathbf{X}$ is orthonormal and $\mathbf{X}^T\mathbf{X} = \mathbf{I}_d$, then:

$$
\begin{aligned}
\hat{\mathbf{w}}_{\lambda}^{ridge} &= (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}_d)^{-1}\mathbf{X}^T\mathbf{y} \\
\hat{\mathbf{w}}_{\lambda}^{ridge} &= (1 + \lambda)^{-1}\mathbf{I}_d^{-1}\mathbf{X}\mathbf{y} \\
\hat{\mathbf{w}}_{\lambda}^{ridge} &= \frac{1}{1+\lambda}\hat{\mathbf{w}}^{ls}.
\end{aligned}
$$

Ridge Regression- Prostate cancer example

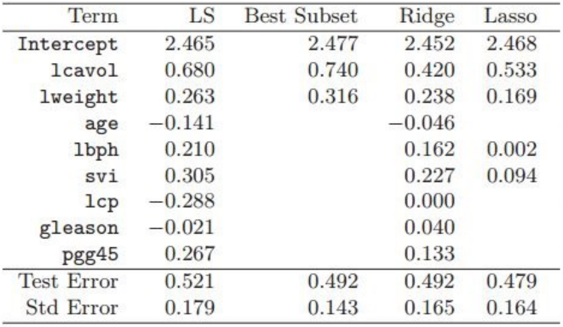
Correlation between the level of prostate-specific antigen and clinical measures in men who were about to receive a radical prostatectomy: log cancer volume (lcavol), log prostate weight (lweight), age, log of the amount of benign prostatic hyperplasia (lbph), seminal vesicle invasion (svi), log of capsular penetration (lcp), Gleason score (gleason), and percent of Gleason scores 4 or 5 (pgg45). The correlation matrix of the predictors is:



	lcavol	lweight	age	lbph	svi	lcp	gleason
lweight	0.300						
age	0.286	0.317					
lbph	0.063	0.437	0.287				
svi	0.593	0.181	0.129	−0.139			
lcp	0.692	0.157	0.173	−0.089	0.671		
gleason	0.426	0.024	0.366	0.033	0.307	0.476	
pgg45	0.483	0.074	0.276	−0.030	0.481	0.663	0.757

Ridge Regression- Prostate cancer example

Estimated coefficients and test error results, for different subset and shrinkage methods applied to the prostate data. The blank entries correspond to variables omitted.



Term	LS	Best Subset	Ridge	Lasso
Intercept	2.465	2.477	2.452	2.468
lcavol	0.680	0.740	0.420	0.533
lweight	0.263	0.316	0.238	0.169
age	−0.141		−0.046	
lbph	0.210		0.162	0.002
svi	0.305		0.227	0.094
lcp	−0.288		0.000	
gleason	−0.021		0.040	
pgg45	0.267		0.133	
Test Error	0.521	0.492	0.492	0.479
Std Error	0.179	0.143	0.165	0.164

Prediction Error And The Bias-Variance Tradeoff

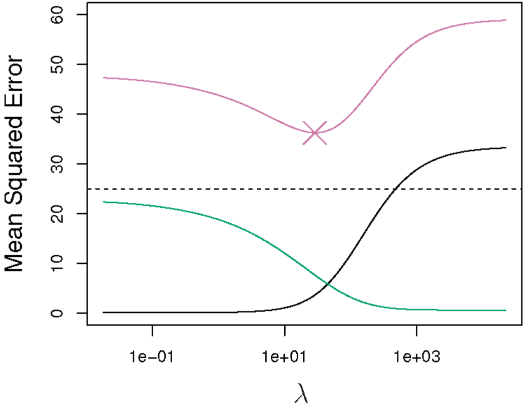
- Good estimators should have small prediction errors.
- Consider the PE at a particular point $\mathbf{x}_0$:

$$\mathrm{PE}(\mathbf{x}_0) = \sigma_\epsilon^2 + \mathrm{Bias}^2(\mathrm{f}(\mathbf{x}_0)) + \mathrm{Var}(\mathrm{f}(\mathbf{x}_0)). \tag{3}$$

- Bias-variance tradeoff.
  - As model becomes more complex, local structure/curvature can be picked up.
  - But coefficient estimates suffer from high variance as more terms are included in the model.
- Introducing a little bias in estimate for β might lead to decrease in variance, and to decrease PE.

Ridge Regression

Bias-variance trade-off.



Squared bias (black), variance (green), and test mean squared error (purple).

- $\lambda = 0$, the variance is high but there is no bias.
- As λ increases, the variance decreases, at the expense of bias.