10-701: Introduction to Machine Learning Lecture 8 – Regularization

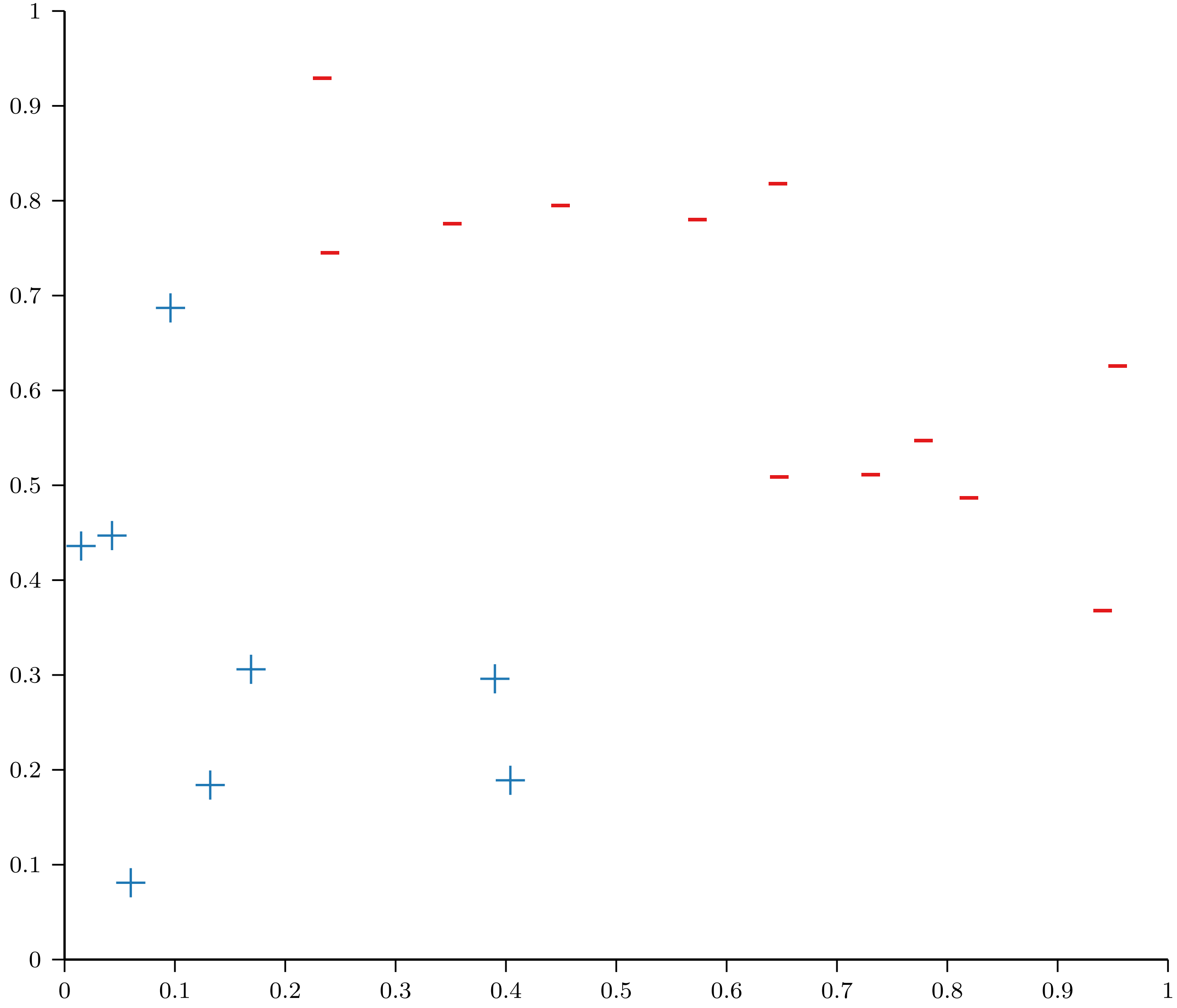
Henry Chai

2/12/24

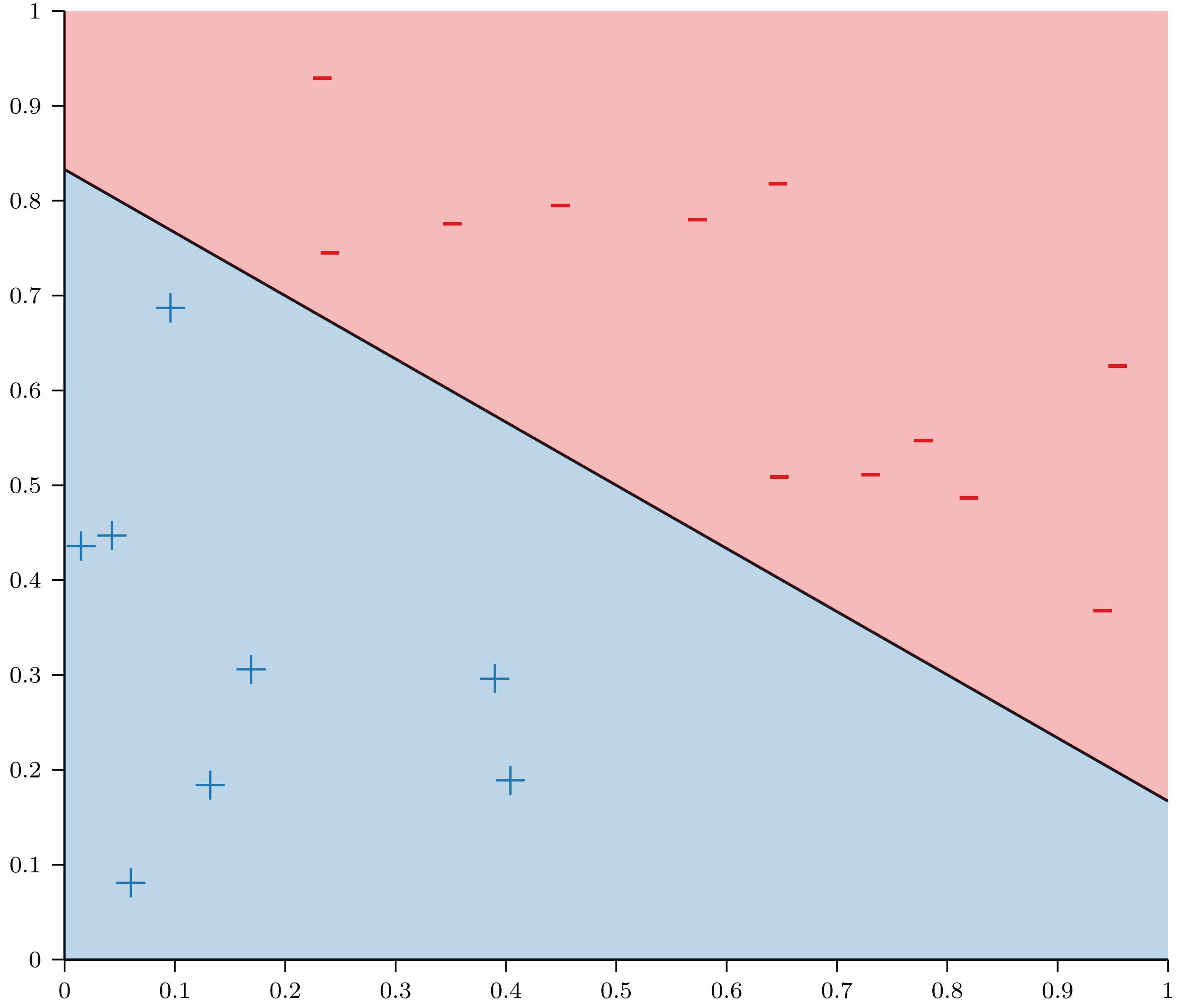
Front Matter

- Announcements:
  - HW2 released 2/7, due **2/19** (previously 2/16) at 11:59 PM
  - HW3 released **2/19** (previously 2/16), due **2/28** (previously 2/26) at 11:59 PM
  - Lecture schedule has been updated, see the course website for full details
    - Lecture on 2/21 (Wednesday) and Recitation on 2/23 (Friday) have been swapped
- Recommended Readings:
  - Murphy, Sections 7.5 & 14.4

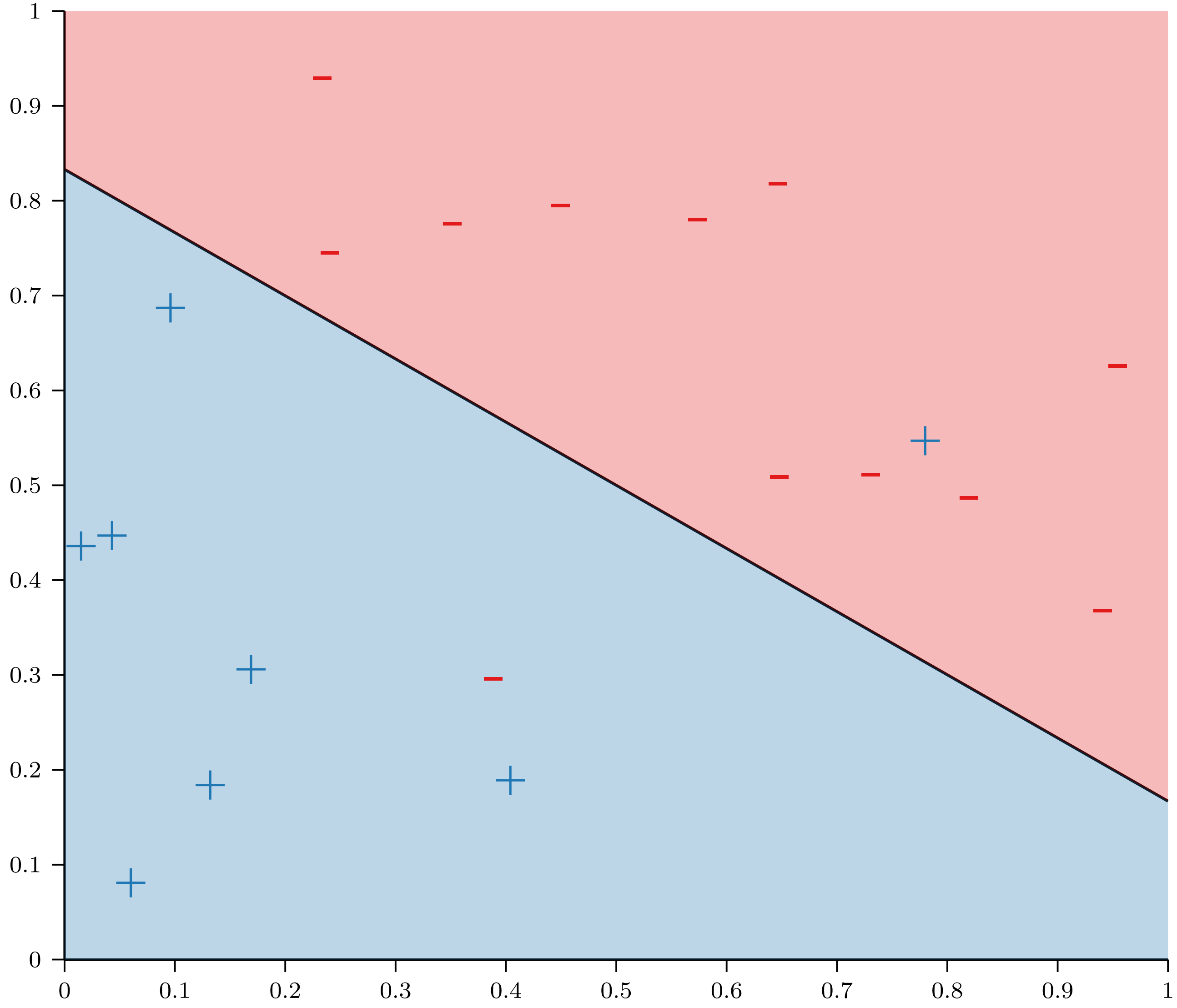
Linear Models



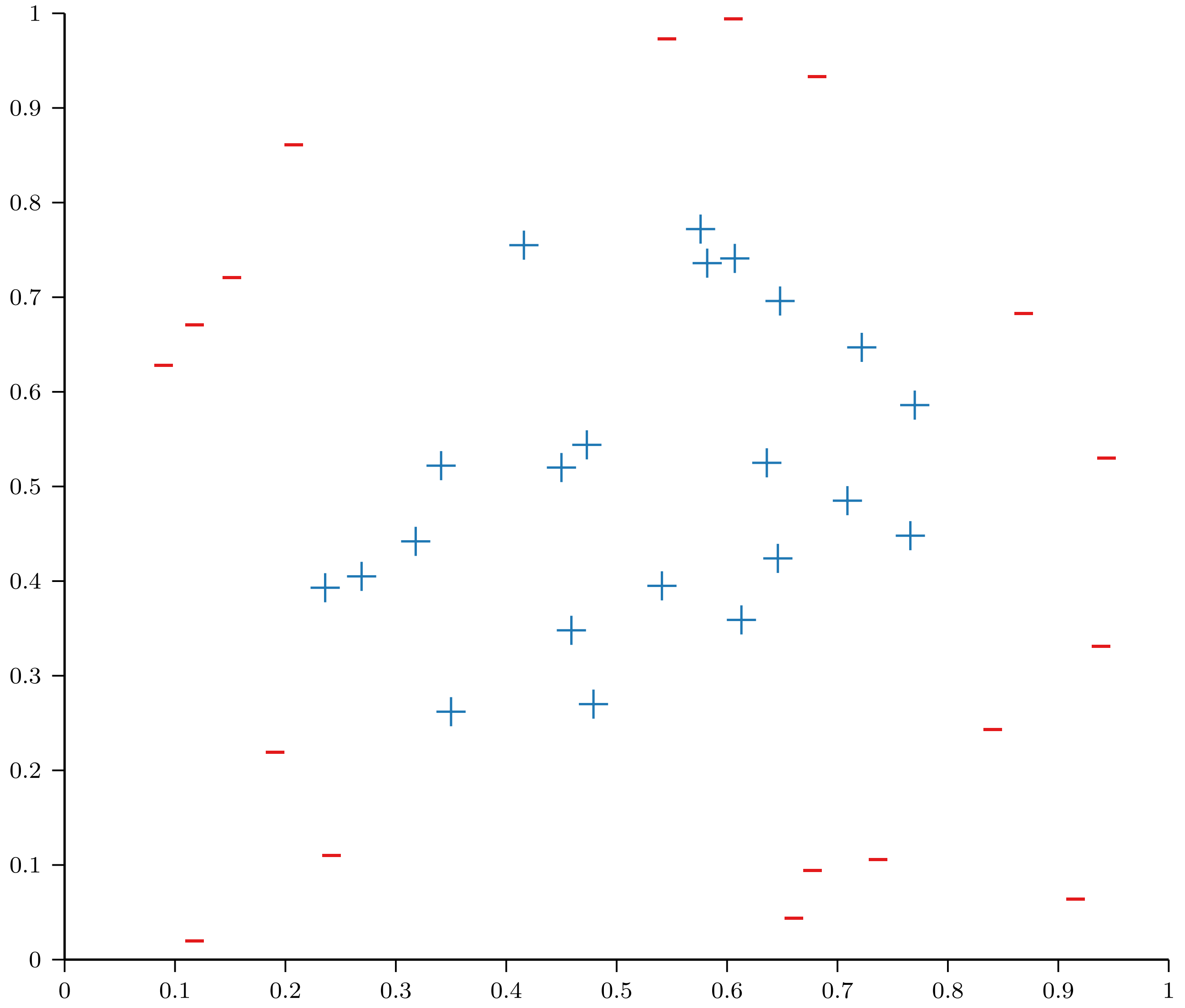
Linear Models



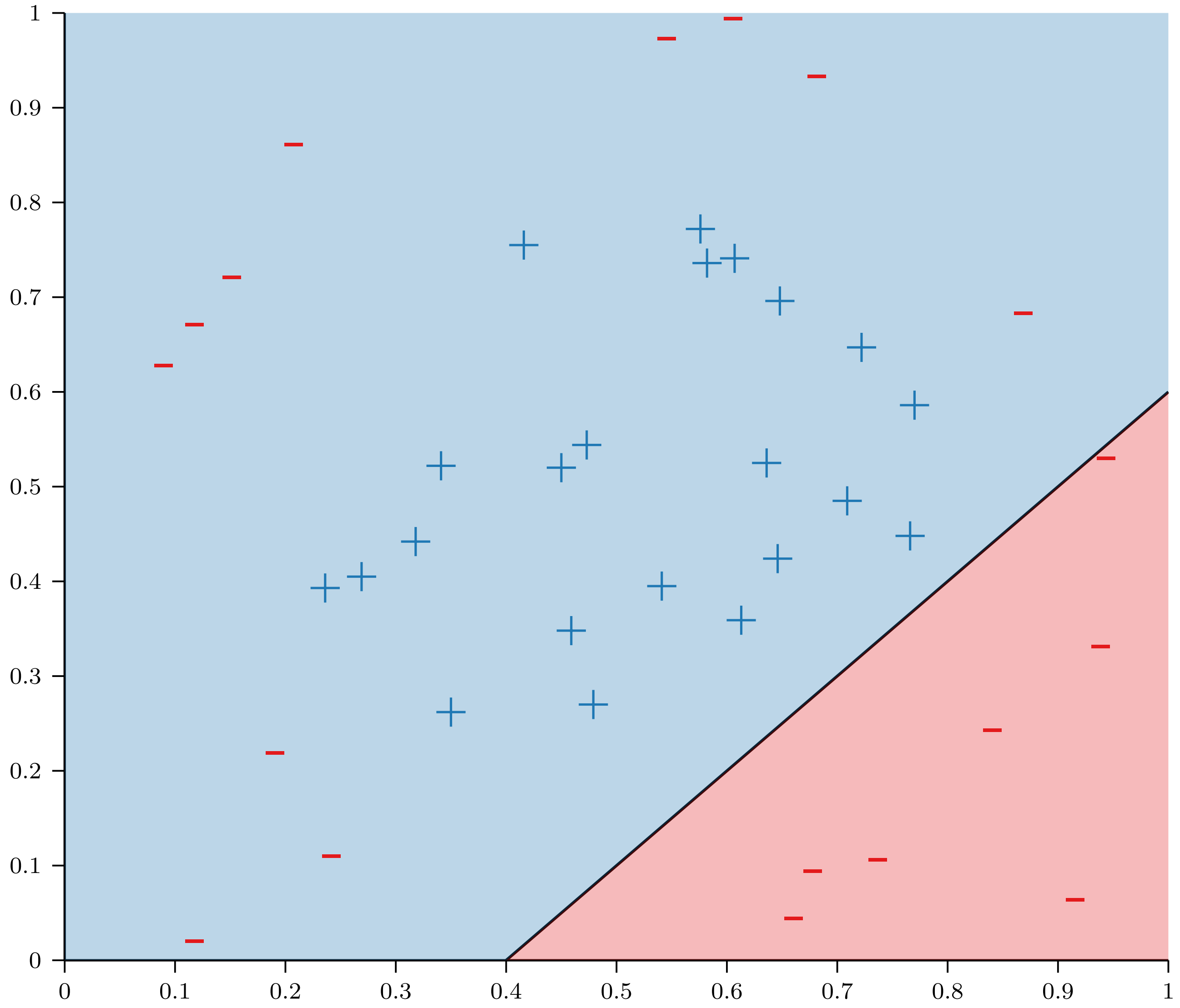
Linear Models



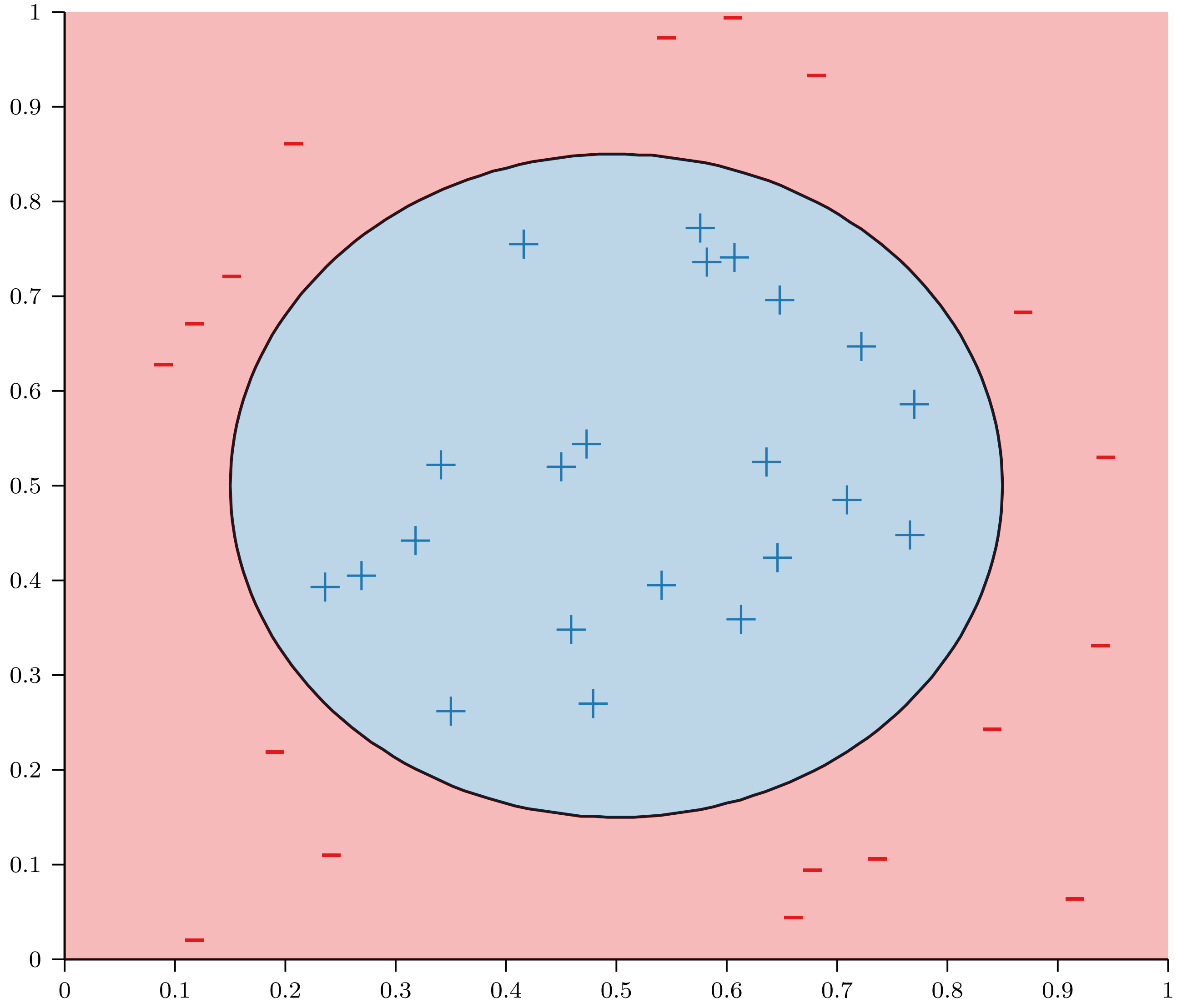
Linear Models?



Linear Models?



Nonlinear Models

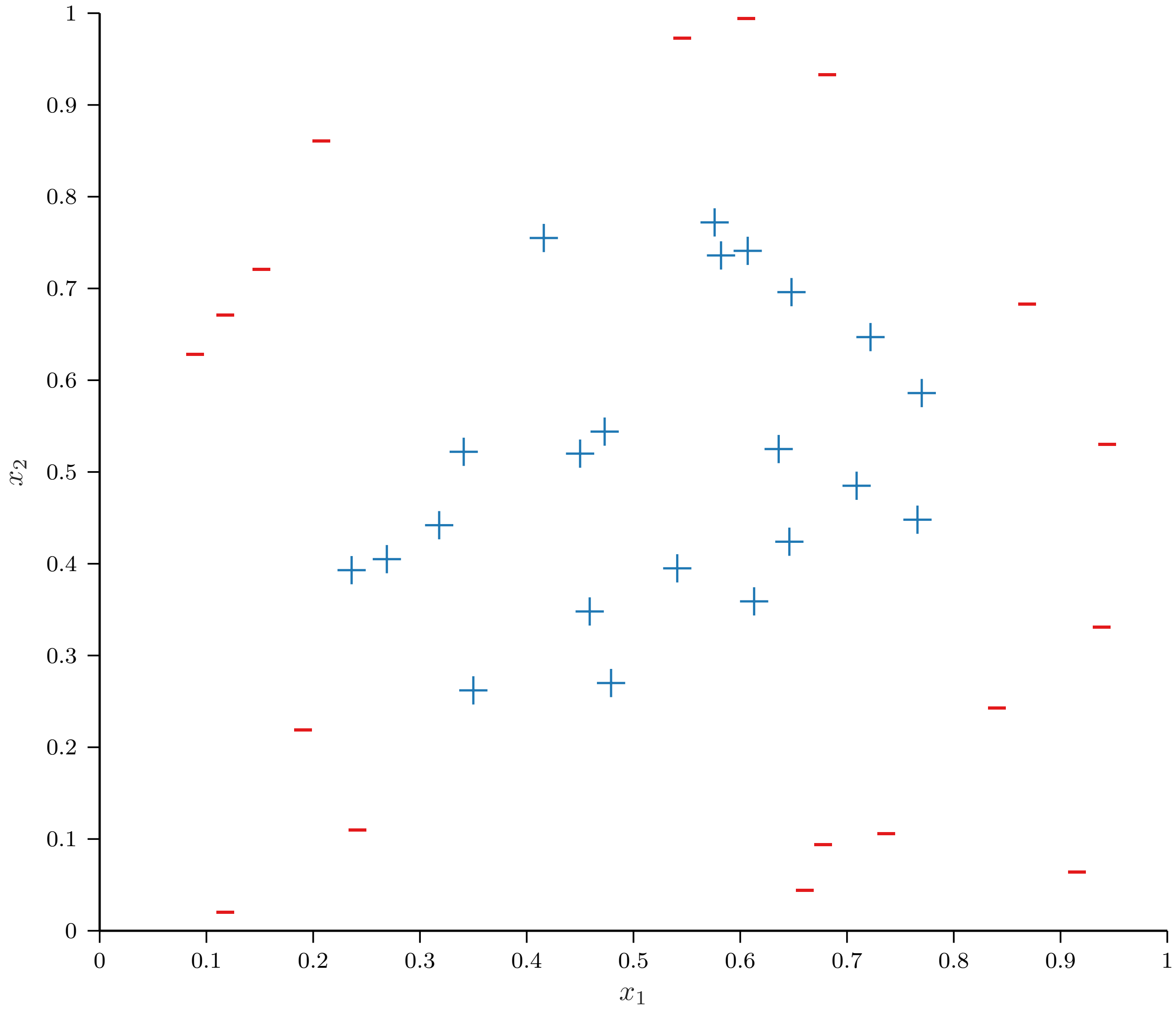


Feature Transforms

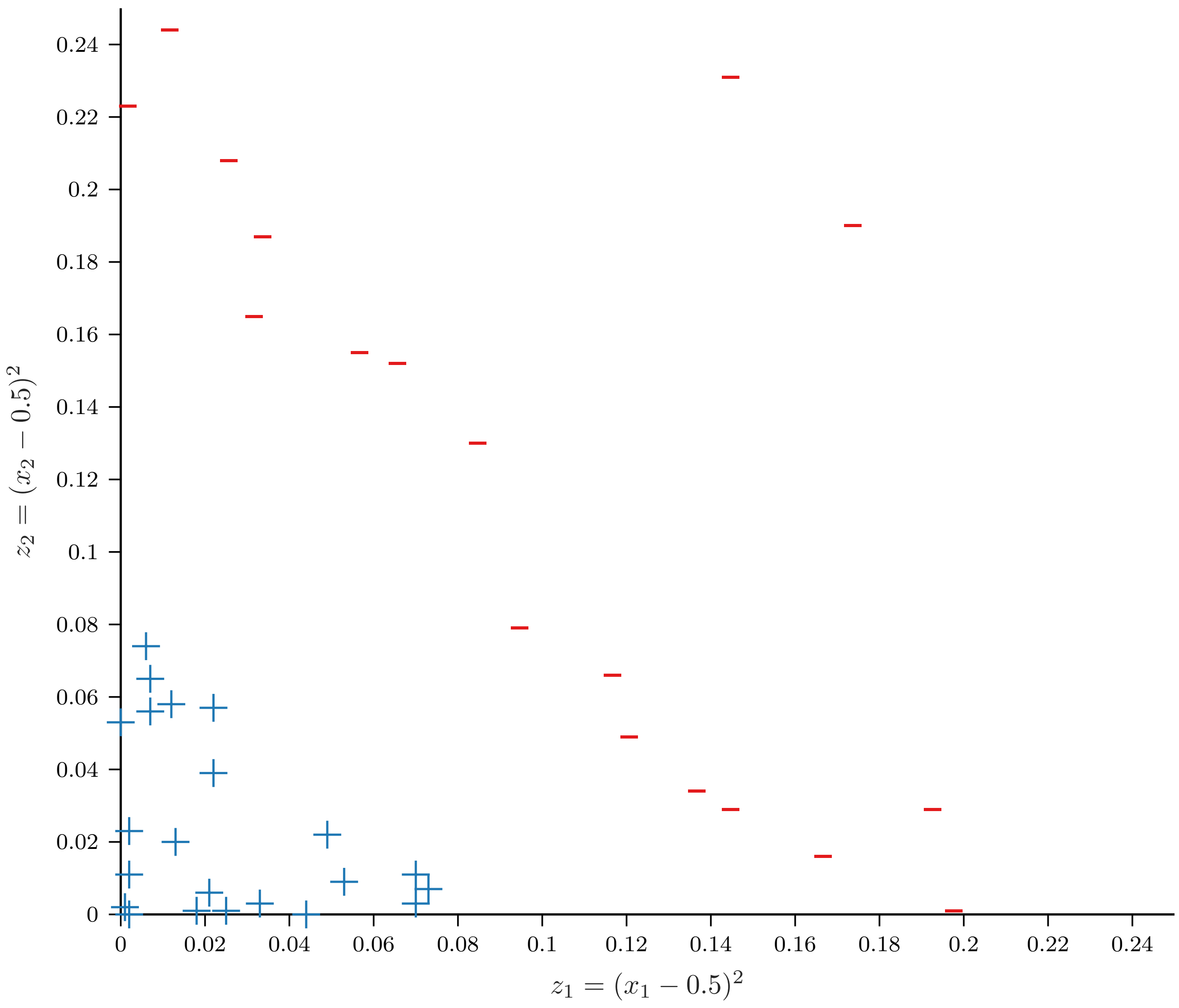
- Given D-dimensional inputs $\boldsymbol{x} = [x_1, \ldots, x_D]$, first compute some transformation of our input, e.g.,

$$\phi([x_1, x_2]) = [z_1 = (x_1 - 0.5)^2, z_2 = (x_2 - 0.5)^2]$$

Nonlinear Models

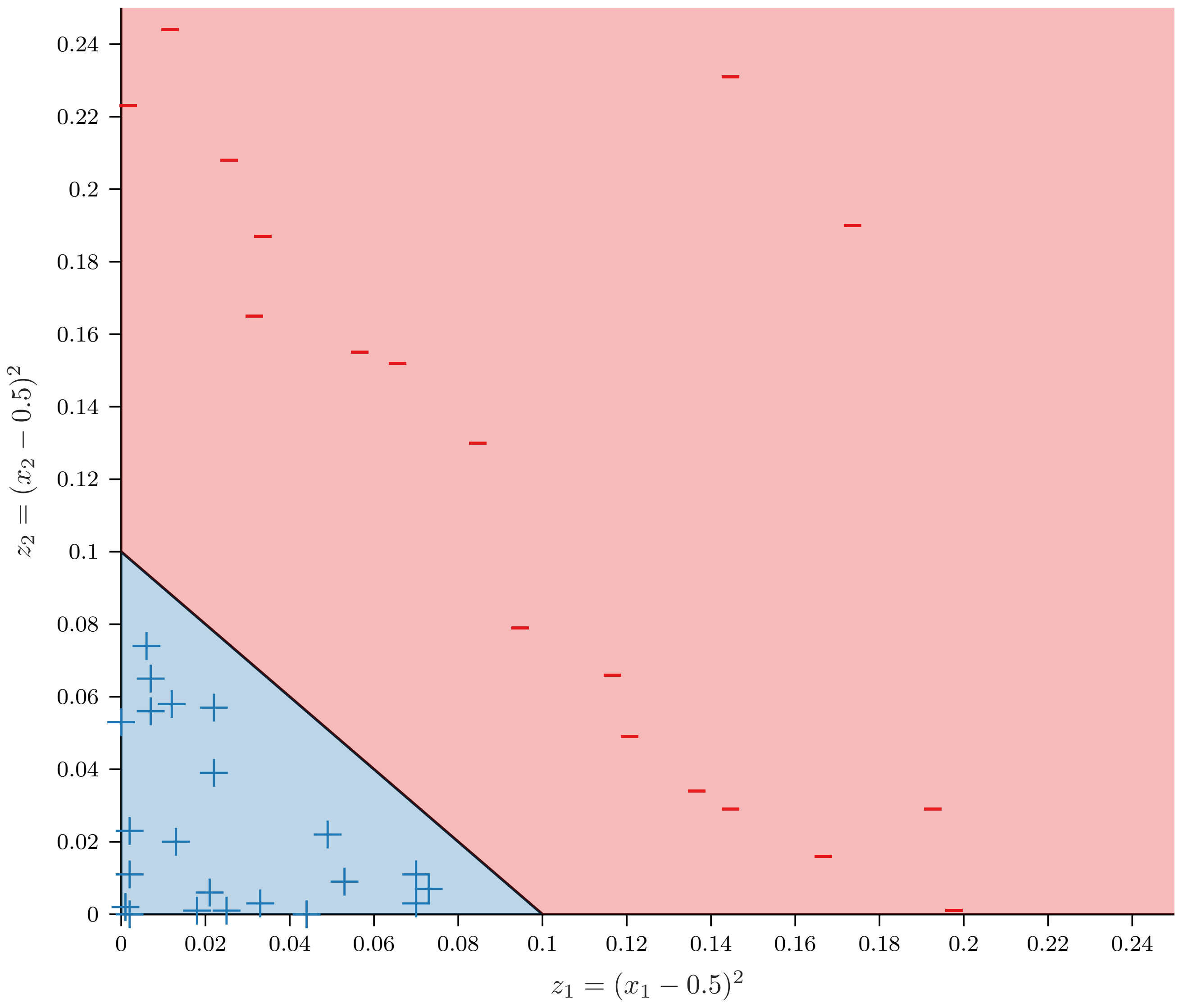


Nonlinear Models

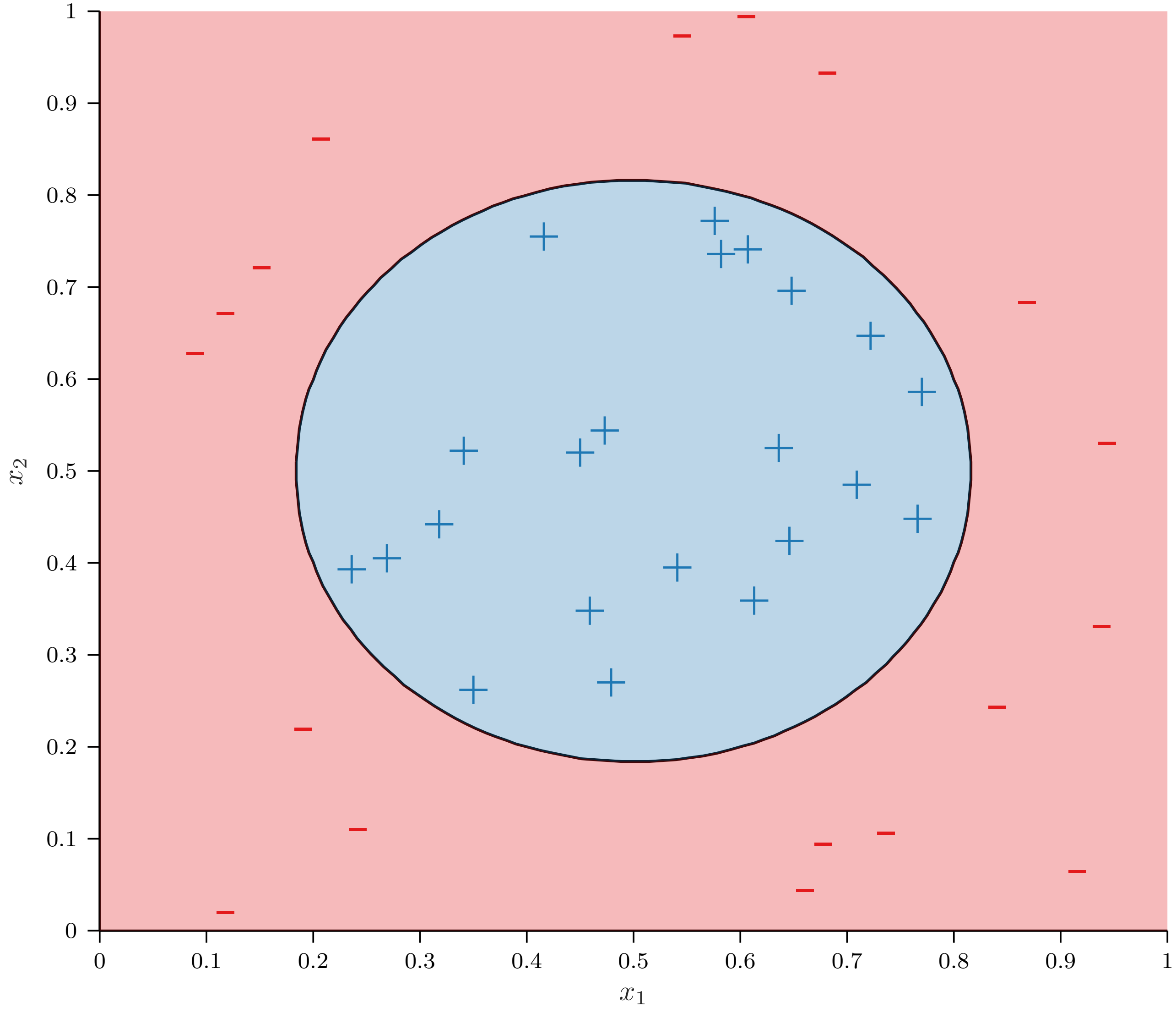


Nonlinear Models

$w_1 z_1 + w_2 z_2 + b = 0$



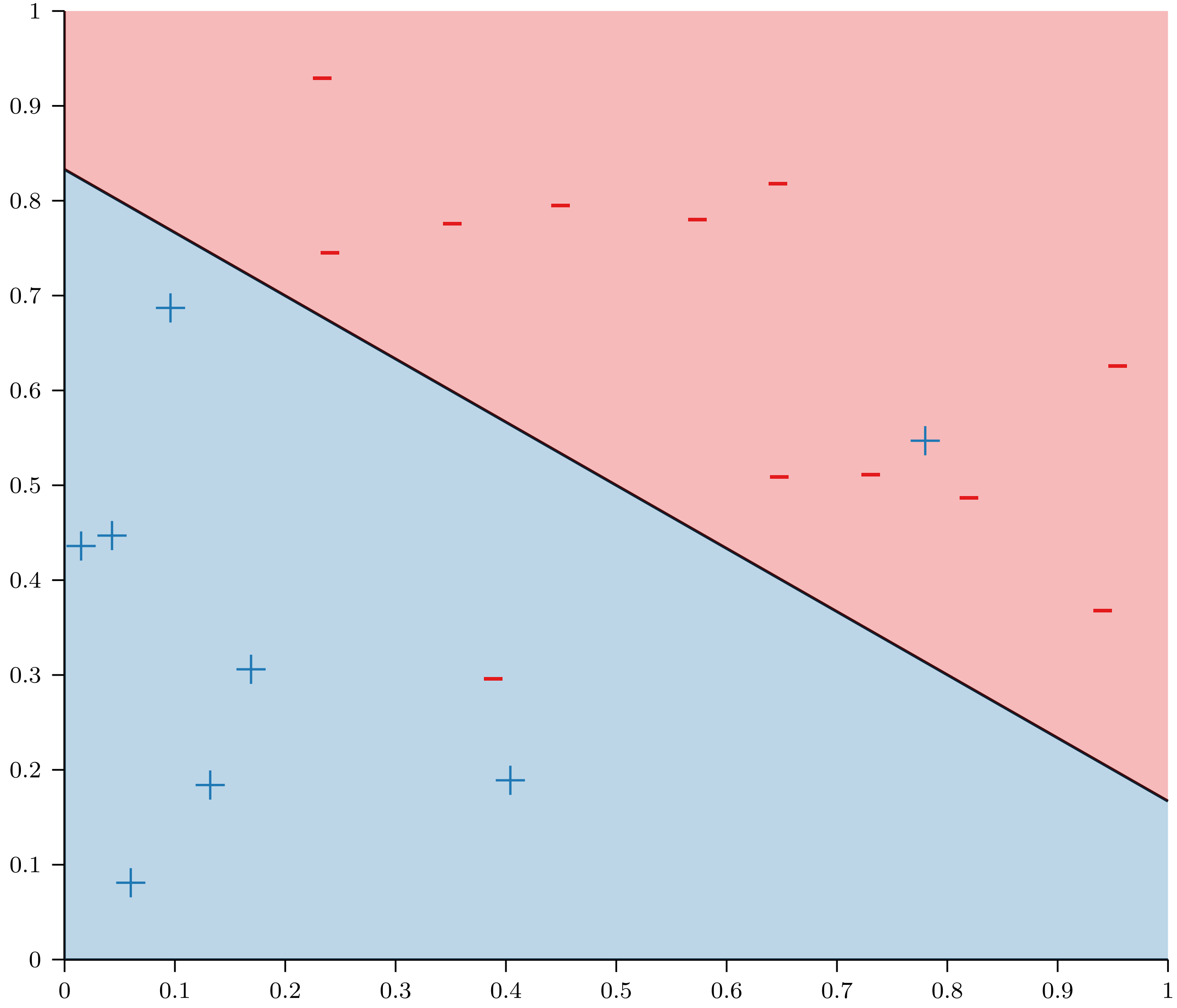
Nonlinear Models



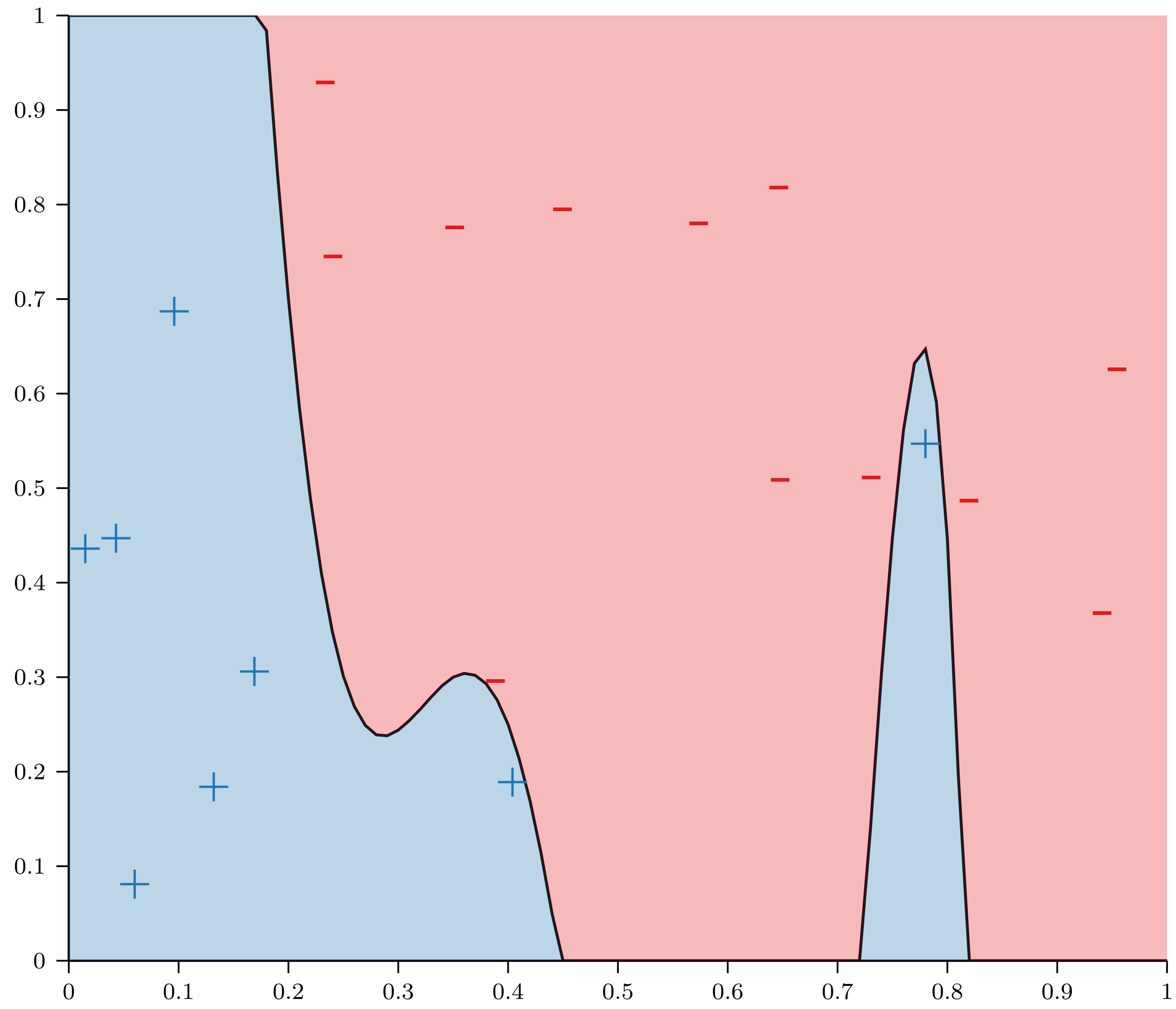
General Q^{th}-order Transforms

- $\phi_{2,2}([x_1, x_2]) = [x_1, x_2, x_1^2, x_1x_2, x_2^2]$
- $\phi_{2,3}([x_1, x_2]) = [x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3]$
- $\phi_{2,4}([x_1, x_2]) =$
  $[x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3, x_1^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4]$
- $\phi_{2,Q}$ maps a 2-D input to a $O(Q^2)$-D output
- Scales even worse for higher-dimensional inputs…

Linear Models



Nonlinear Models?



Feature Transforms: Tradeoffs

	Low-Dimensional Input Space	High-Dimensional Input Space
Training Error	High	Low
Generalization	Good	Bad

Feature Transforms: Experiment

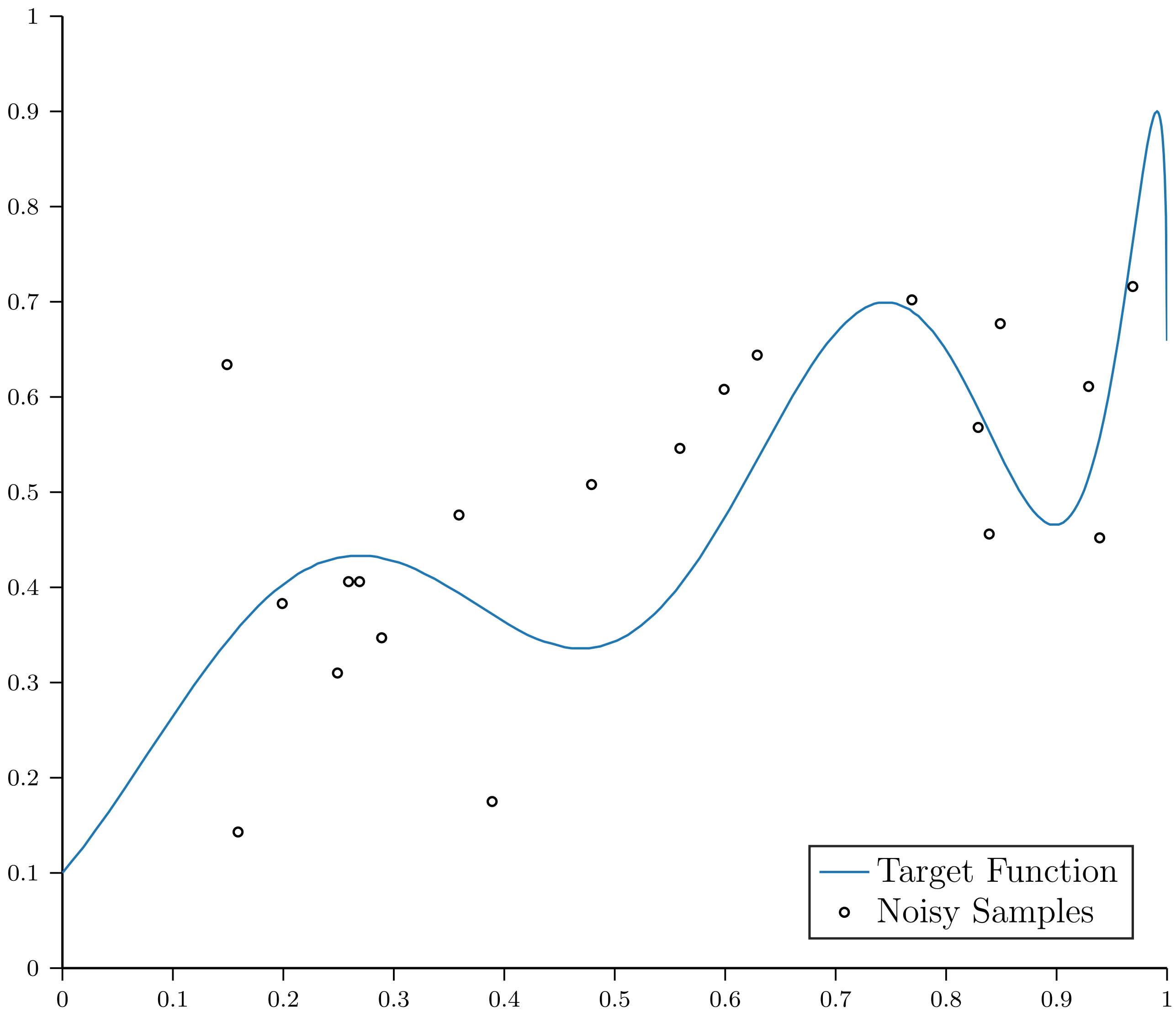
- $x \in \mathbb{R}$, $y \in \mathbb{R}$ and $N = 20$
- Targets are generated by a 10^{th}-order polynomial in x with additive Gaussian noise:

$$y = \sum_{d=0}^{10} a_d x^d + \epsilon \text{ where } \epsilon \sim N(0, \sigma^2)$$

- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomials
  - $\phi_{1,2}(x) = [x, x^2]$
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomials
  - $\phi_{1,10}(x) = [x, x^2, x^3, x^4, x^5, x^6, x^7, x^8, x^9, x^{10}]$

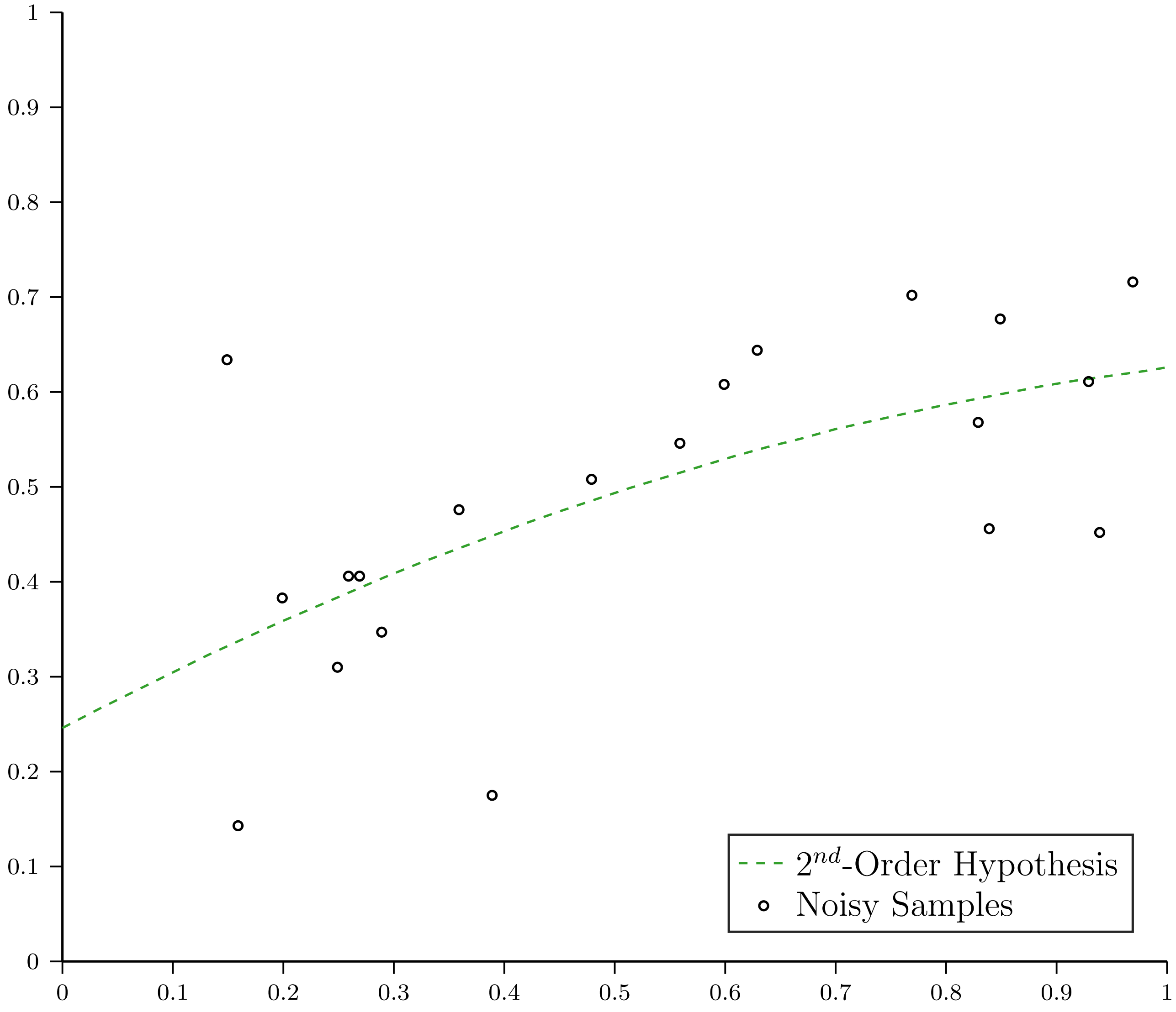
Noisy Targets

- 10-dimensional target function with additive Gaussian noise
- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomial
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomial



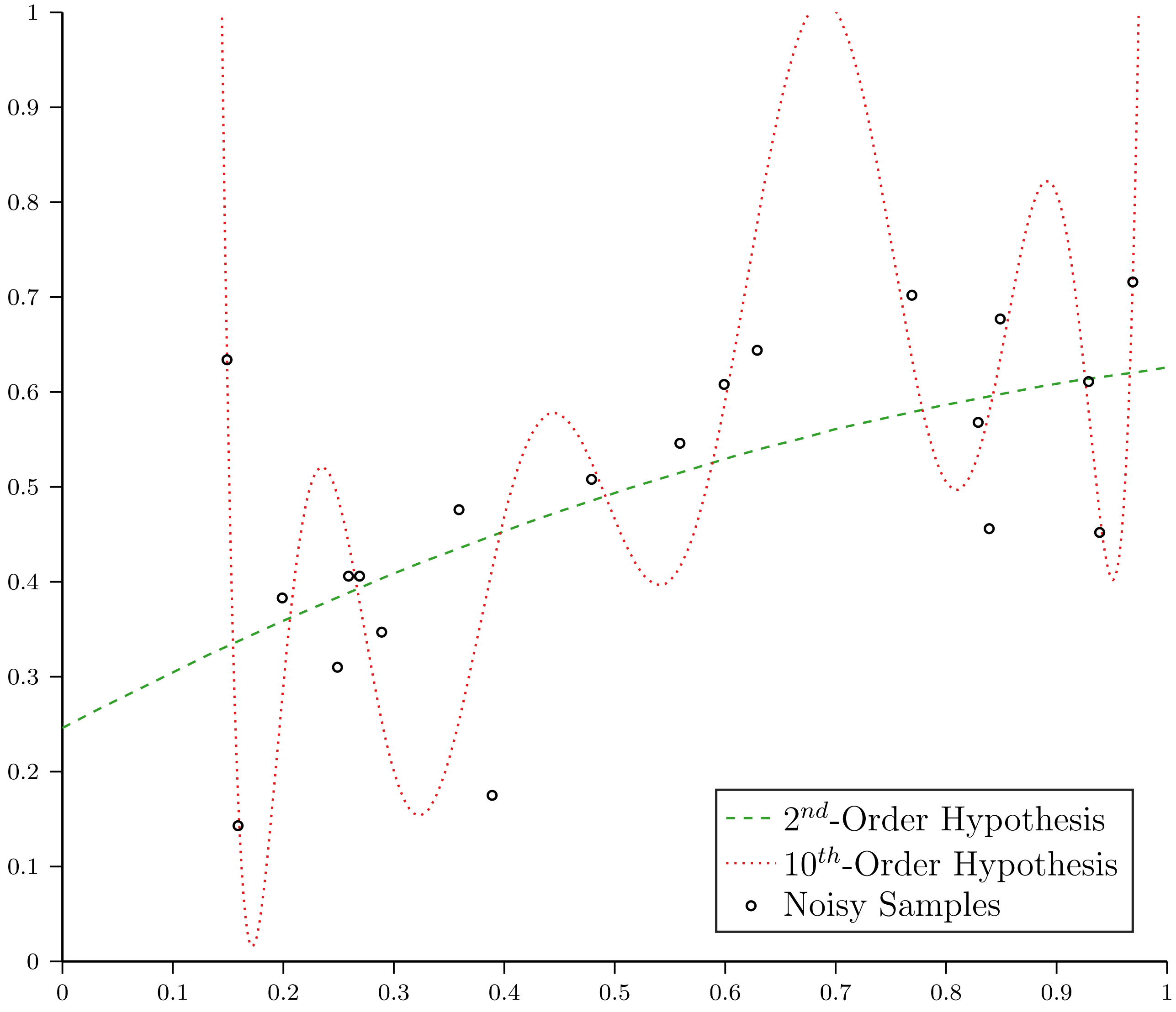
Noisy Targets

- 10-dimensional target function with additive Gaussian noise
- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomial
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomial



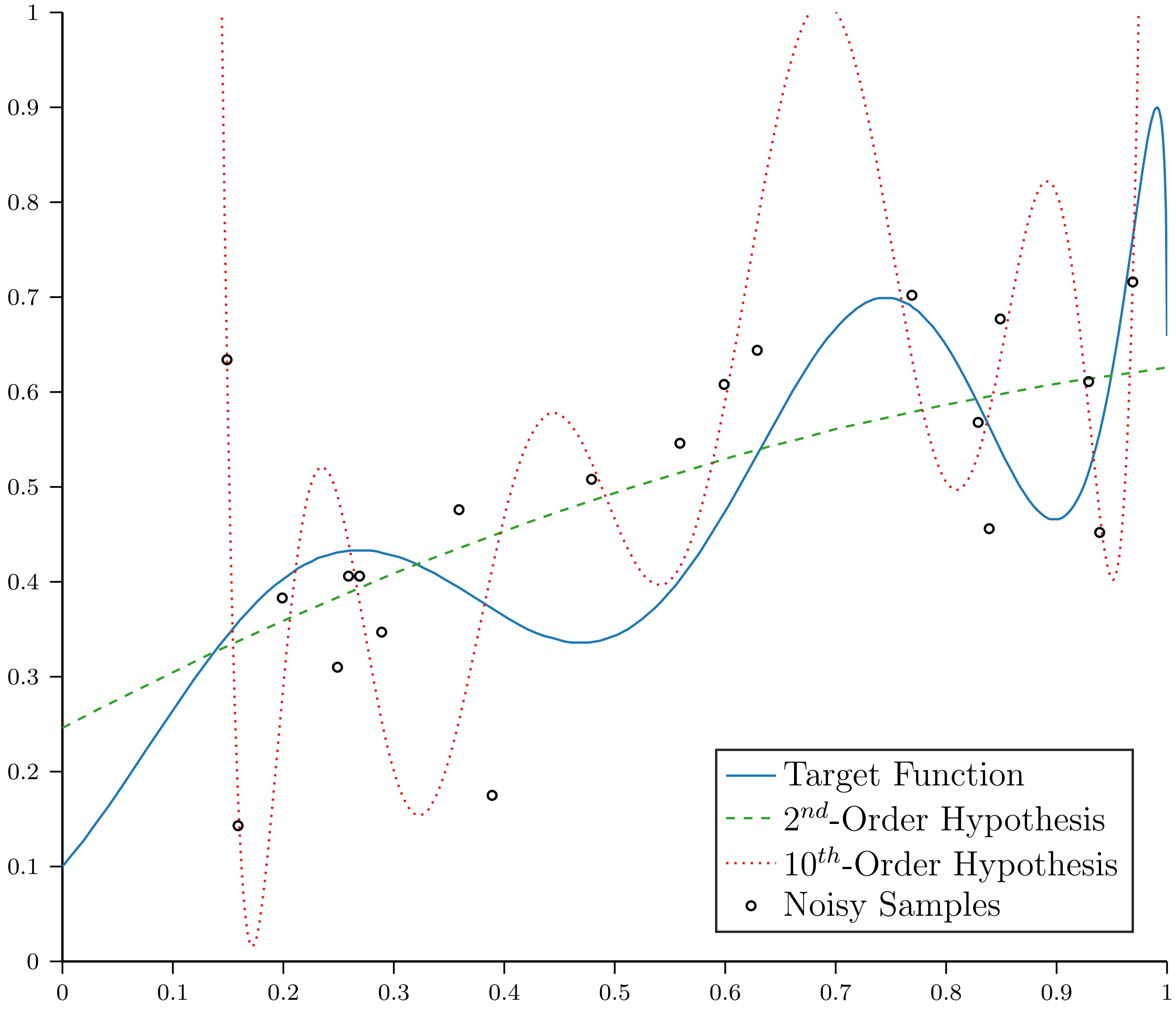
Noisy Targets

- 10-dimensional target function with additive Gaussian noise
- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomial
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomial



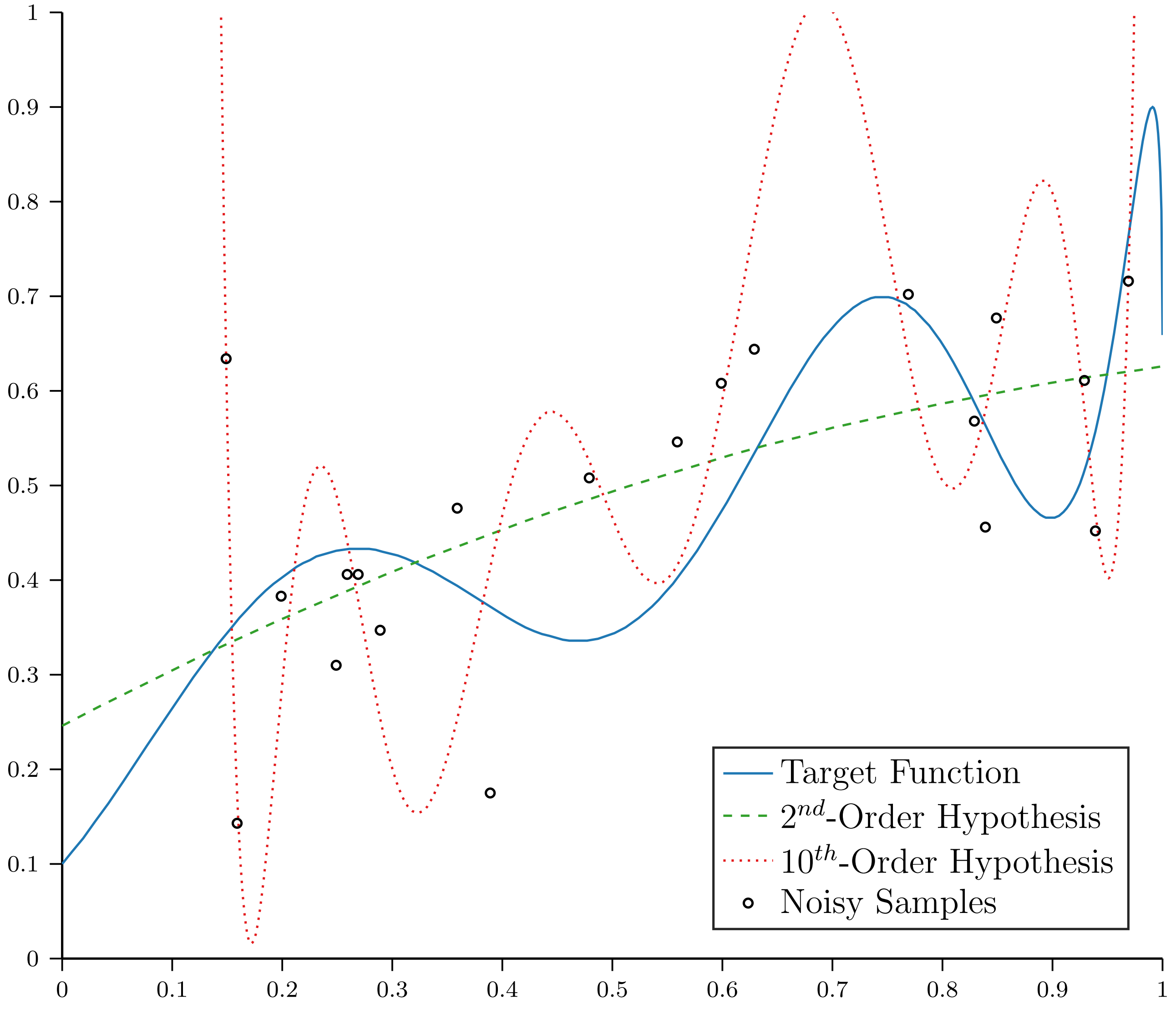
Noisy Targets

- 10-dimensional target function with additive Gaussian noise
- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomial
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomial



Noisy Targets

	$\mathcal{H}_2$	$\mathcal{H}_{10}$
Training Error	0.016	0.011
True Error	0.009	3797



Feature Transforms: Experiment

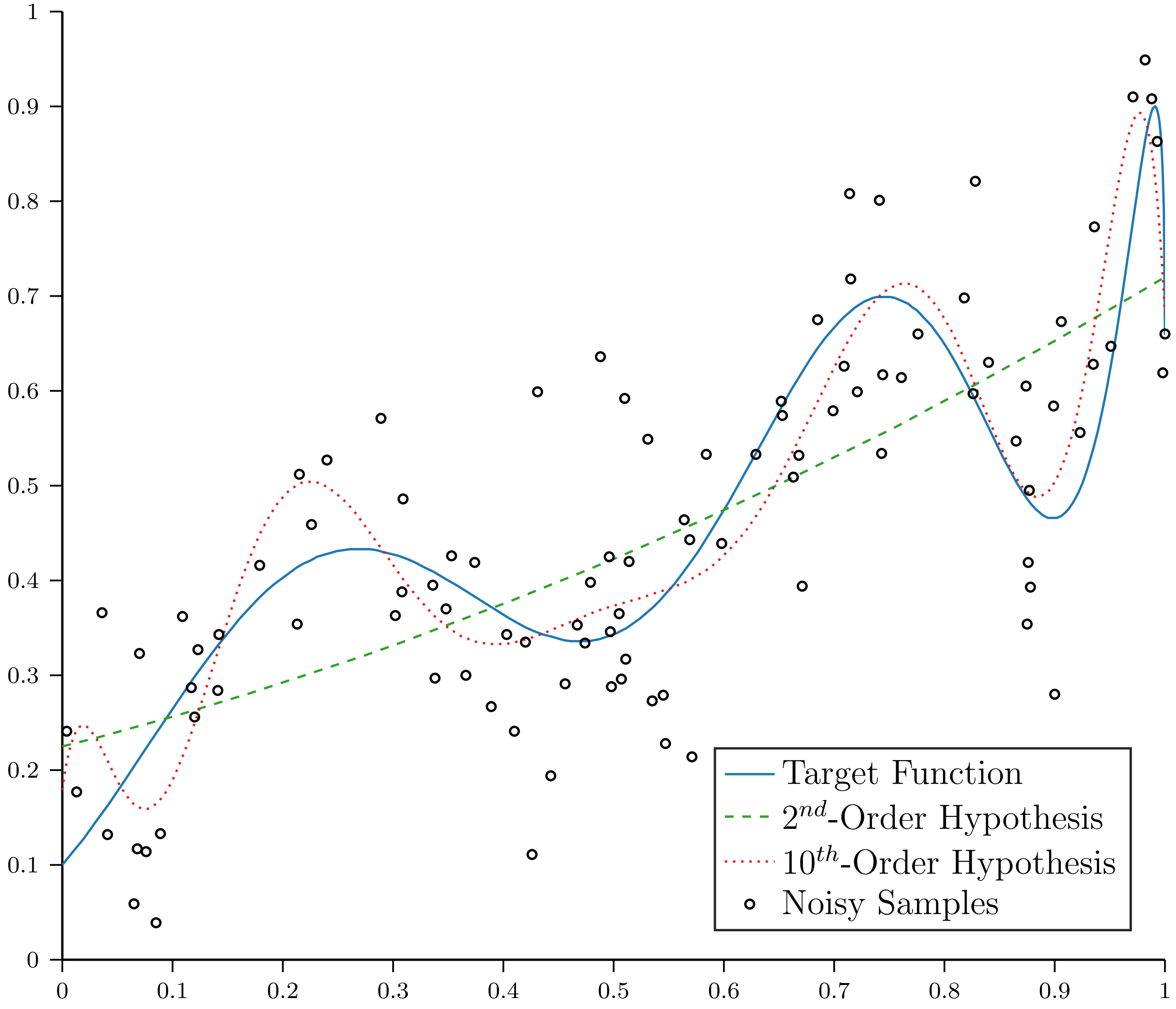
- $x \in \mathbb{R}$, $y \in \mathbb{R}$ and $N = 100$
- Targets are generated by a 10^{th}-order polynomial in x with additive Gaussian noise:

$$y = \sum_{d=0}^{10} a_d x^d + \epsilon \text{ where } \epsilon \sim N(0, \sigma^2)$$

- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomials
  - $\phi_{1,2}(x) = [x, x^2]$
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomials
  - $\phi_{1,10}(x) = [x, x^2, x^3, x^4, x^5, x^6, x^7, x^8, x^9, x^{10}]$

Noisy Targets

	$\mathcal{H}_2$	$\mathcal{H}_{10}$
Training Error	0.018	0.010
True Error	0.009	0.003



Regularization

- Constrain models to prevent them from overfitting
- Learning algorithms are optimization problems and regularization imposes constraints on the optimization

Hard Constraints

- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomials
  - $\phi_{1,10}(x) = [x, x^2, x^3, x^4, x^5, x^6, x^7, x^8, x^9, x^{10}]$
- Given $\mathrm{X} = \begin{bmatrix} 1 & \phi_{1,10}\left(x^{(1)}\right) \\ 1 & \phi_{1,10}\left(x^{(2)}\right) \\ \vdots & \vdots \\ 1 & \phi_{1,10}\left(x^{(N)}\right) \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$ find

  $\boldsymbol{\omega} = [\omega_0, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6, \omega_7, \omega_8, \omega_9, \omega_{10}]$ that minimizes

$$\frac{1}{N}(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})^T(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})$$

- Subject to

$$\omega_3 = \omega_4 = \omega_5 = \omega_6 = \omega_7 = \omega_8 = \omega_9 = \omega_{10} = 0$$

Hard Constraints

- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomials
  - $\phi_{1,10}(x) = [x, x^2, x^3, x^4, x^5, x^6, x^7, x^8, x^9, x^{10}]$
- Given $\mathrm{X} = \begin{bmatrix} 1 & \phi_{1,10}\left(x^{(1)}\right) \\ 1 & \phi_{1,10}\left(x^{(2)}\right) \\ \vdots & \vdots \\ 1 & \phi_{1,10}\left(x^{(N)}\right) \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$ find

  $\boldsymbol{\omega} = [\omega_0, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6, \omega_7, \omega_8, \omega_9, \omega_{10}]$ that minimizes

$$\frac{1}{N}\sum_{n=1}^{N}\left(\left(\sum_{d=0}^{10} x_d^{(n)}\omega_d\right) - y^{(n)}\right)^2$$

- Subject to

$$\omega_3 = \omega_4 = \omega_5 = \omega_6 = \omega_7 = \omega_8 = \omega_9 = \omega_{10} = 0$$

Hard Constraints

- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomials
  - $\phi_{1,10}(x) = [x, x^2, x^3, x^4, x^5, x^6, x^7, x^8, x^9, x^{10}]$
- Given $\mathrm{X} = \begin{bmatrix} 1 & \phi_{1,10}\left(x^{(1)}\right) \\ 1 & \phi_{1,10}\left(x^{(2)}\right) \\ \vdots & \vdots \\ 1 & \phi_{1,10}\left(x^{(N)}\right) \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$ find

  $\boldsymbol{\omega} = [\omega_0, \omega_1, \omega_2, \omega_3, \omega_4, \omega_5, \omega_6, \omega_7, \omega_8, \omega_9, \omega_{10}]$ that minimizes

$$\frac{1}{N}\sum_{n=1}^{N}\left(\left(\sum_{d=0}^{2} x_d^{(n)}\omega_d\right) - y^{(n)}\right)^2$$

- Subject to nothing!

Hard Constraints

- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomials
  - $\phi_{1,2}(x) = [x, x^2]$
- Given $\mathrm{X} = \begin{bmatrix} 1 & \phi_{1,2}(x^{(1)}) \\ 1 & \phi_{1,2}(x^{(2)}) \\ \vdots & \vdots \\ 1 & \phi_{1,2}(x^{(N)}) \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$ find

  $\boldsymbol{\omega} = [\omega_0, \omega_1, \omega_2]$

  that minimizes

$$\frac{1}{N}(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})^T(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})$$

- Subject to nothing!

Soft Constraints

- More generally, ϕ can be any nonlinear transformation, e.g., exp, log, sin, sqrt, etc...

- Given $\mathrm{X} = \begin{bmatrix} 1 & \phi_1\left(\boldsymbol{x}^{(1)}\right) & \cdots & \phi_m\left(\boldsymbol{x}^{(1)}\right) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \phi_1\left(\boldsymbol{x}^{(N)}\right) & \cdots & \phi_m\left(\boldsymbol{x}^{(N)}\right) \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$,

  find $\boldsymbol{\omega}$ that minimizes

$$\frac{1}{N}(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})^T(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})$$

- Subject to:

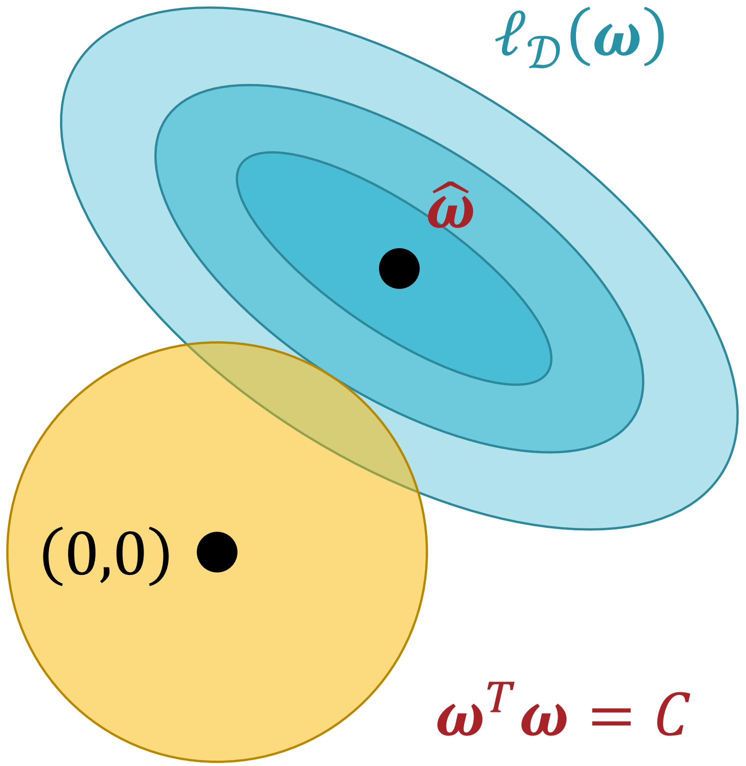
$$\|\boldsymbol{\omega}\|_2^2 = \boldsymbol{\omega}^T\boldsymbol{\omega} = \sum_{d=0}^{D} \omega_d^2 \leq C$$

Soft Constraints

minimize $\ell_{\mathcal{D}}(\boldsymbol{\omega}) = (\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})^T(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})$

$\omega_1^2 + \omega_2^2 \leq C$

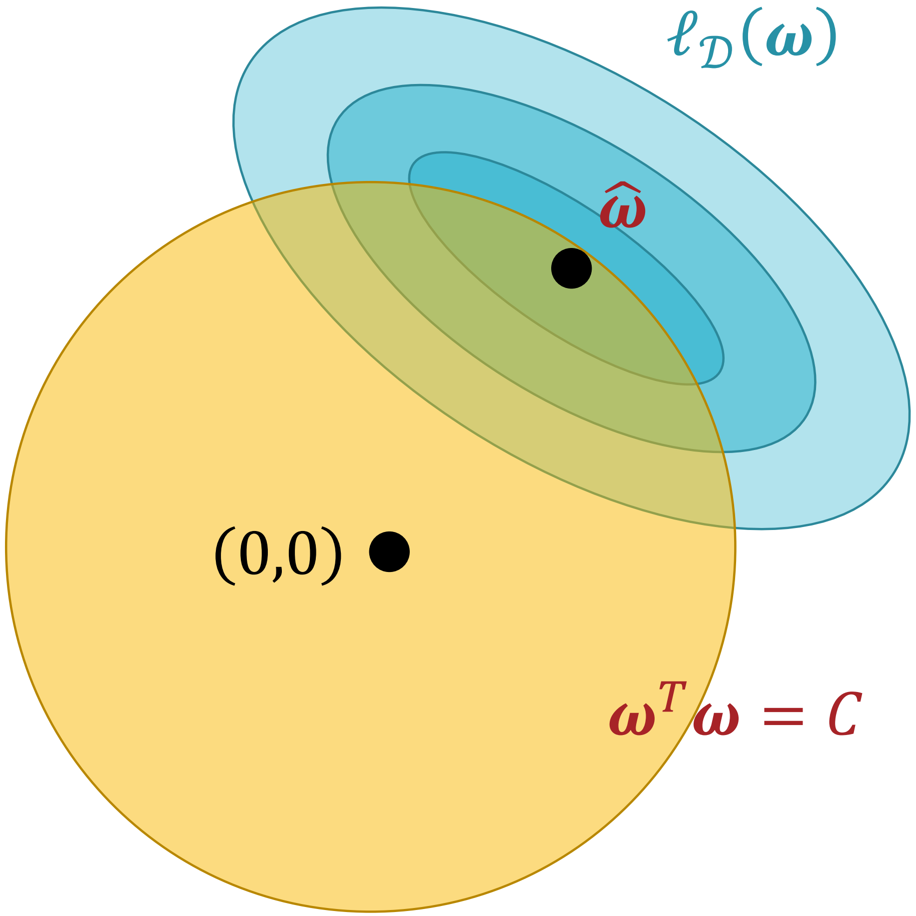
subject to $\boldsymbol{\omega}^T\boldsymbol{\omega} \leq C$



Soft Constraints

minimize $\ell_{\mathcal{D}}(\boldsymbol{\omega}) = (\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})^T(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})$

subject to $\boldsymbol{\omega}^T\boldsymbol{\omega} \leq C$



Soft Constraints

minimize $\ell_{\mathcal{D}}(\boldsymbol{\omega}) = (X\boldsymbol{\omega} - \boldsymbol{y})^T(X\boldsymbol{\omega} - \boldsymbol{y})$

$(w_1 + \mu_1)^2 + (w_2 + \mu_2)^2 \le C$

subject to $\boldsymbol{\omega}^T\boldsymbol{\omega} \le C$

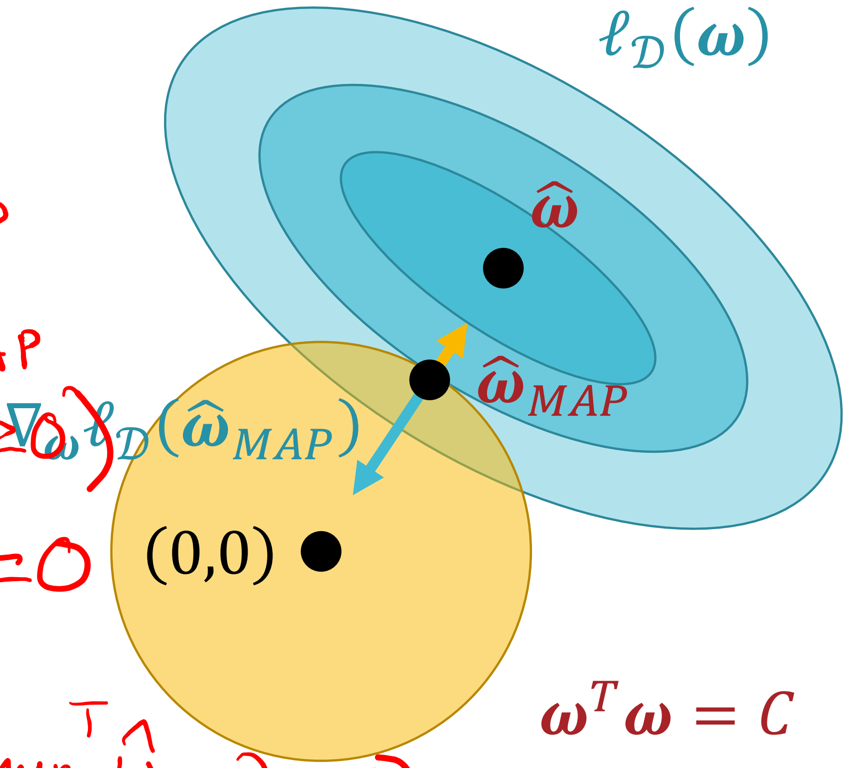
$\nabla_w \ell_D(\hat{\omega}_{MAP}) \propto -\hat{\omega}_{MAP}$

$\nabla_w \ell_D(\hat{\omega}_{MAP}) = -2\lambda_c \hat{\omega}_{MAP}$ $(\lambda_c \ge 0)$

$\nabla_w \ell_D(\hat{\omega}_{MAP}) + 2\lambda_c \hat{\omega}_{MAP} = 0$

$\nabla_w \ell_D(\hat{\omega}_{MAP}) + \lambda_c \nabla_w (\hat{\omega}_{MAP}^T \hat{\omega}_{MAP}) = 0$

$\nabla_w \left(\ell_D(\hat{\omega}_{MAP}) + \lambda_c \hat{\omega}_{MAP}^T \hat{\omega}_{MAP}\right) = 0$



Soft Constraints: Solving for $\hat{\boldsymbol{\omega}}_{MAP}$

minimize $\ell_{\mathcal{D}}(\boldsymbol{\omega}) = (\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})^T(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})$

subject to $\boldsymbol{\omega}^T\boldsymbol{\omega} \leq C$

$$\Updownarrow$$

minimize $\ell_{\mathcal{D}}^{AUG}(\boldsymbol{\omega}) = \ell_{\mathcal{D}}(\boldsymbol{\omega}) + \lambda_C \boldsymbol{\omega}^T\boldsymbol{\omega}$

Ridge Regression

minimize $\ell_{\mathcal{D}}^{AUG}(\boldsymbol{\omega}) = \ell_{\mathcal{D}}(\boldsymbol{\omega}) + \lambda_C \boldsymbol{\omega}^T \boldsymbol{\omega}$

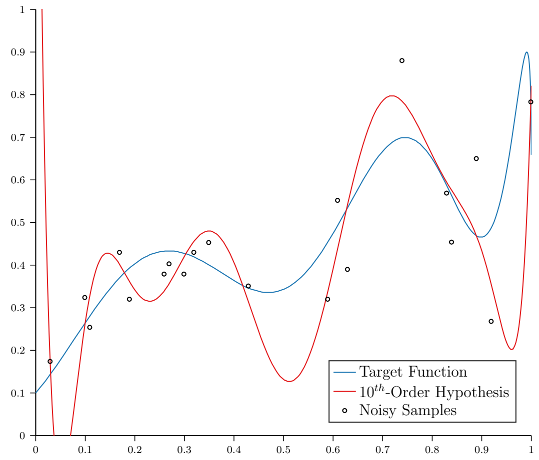
$$\nabla_{\omega} \ell_{D}^{AUG}(\omega) = 2(X^T X \omega - X^T y + \lambda_c \omega)$$

$$2(X^T X \hat{\omega}_{MAP} - X^T y + \lambda_c \hat{\omega}_{MAP}) = 0$$

$$(X^T X + \lambda_c I_{D+1}) \hat{\omega}_{MAP} = X^T y$$

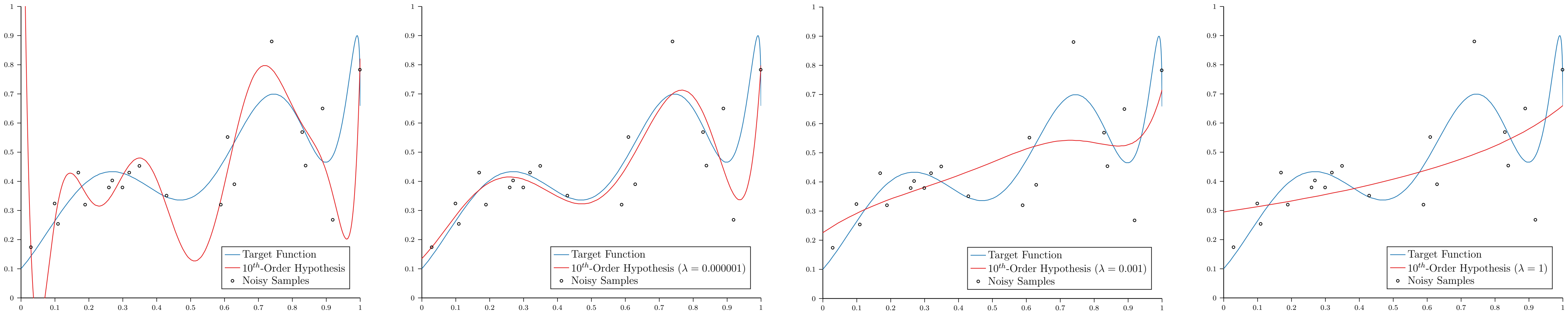
I_{D+1}: $(D+1 \times D+1)$ identity matrix

$$\hat{\omega}_{MAP} = (X^T X + \lambda_c I_{D+1})^{-1} X^T y$$



Ridge Regression

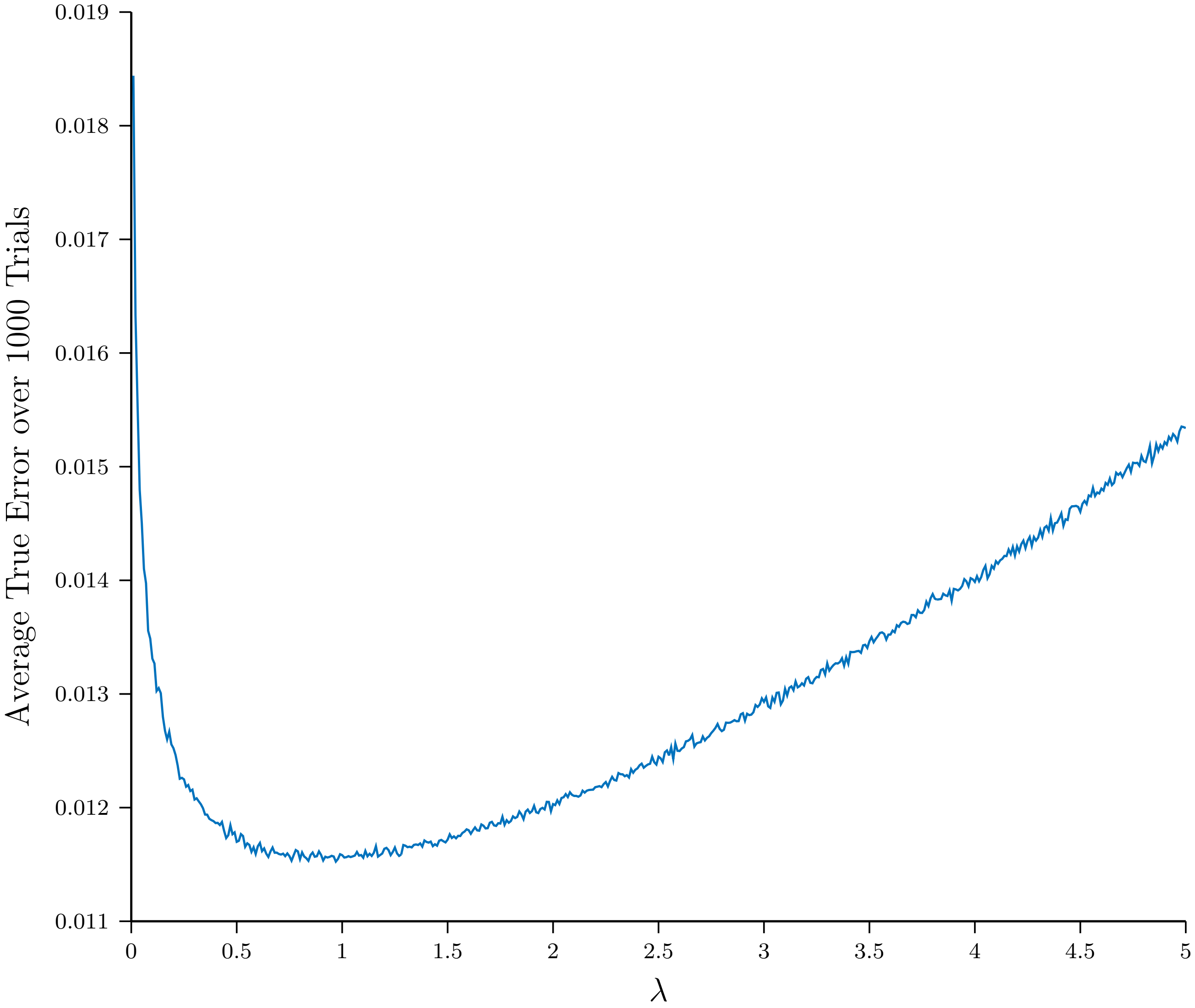
- 10-dimensional target function with additive Gaussian noise
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomial



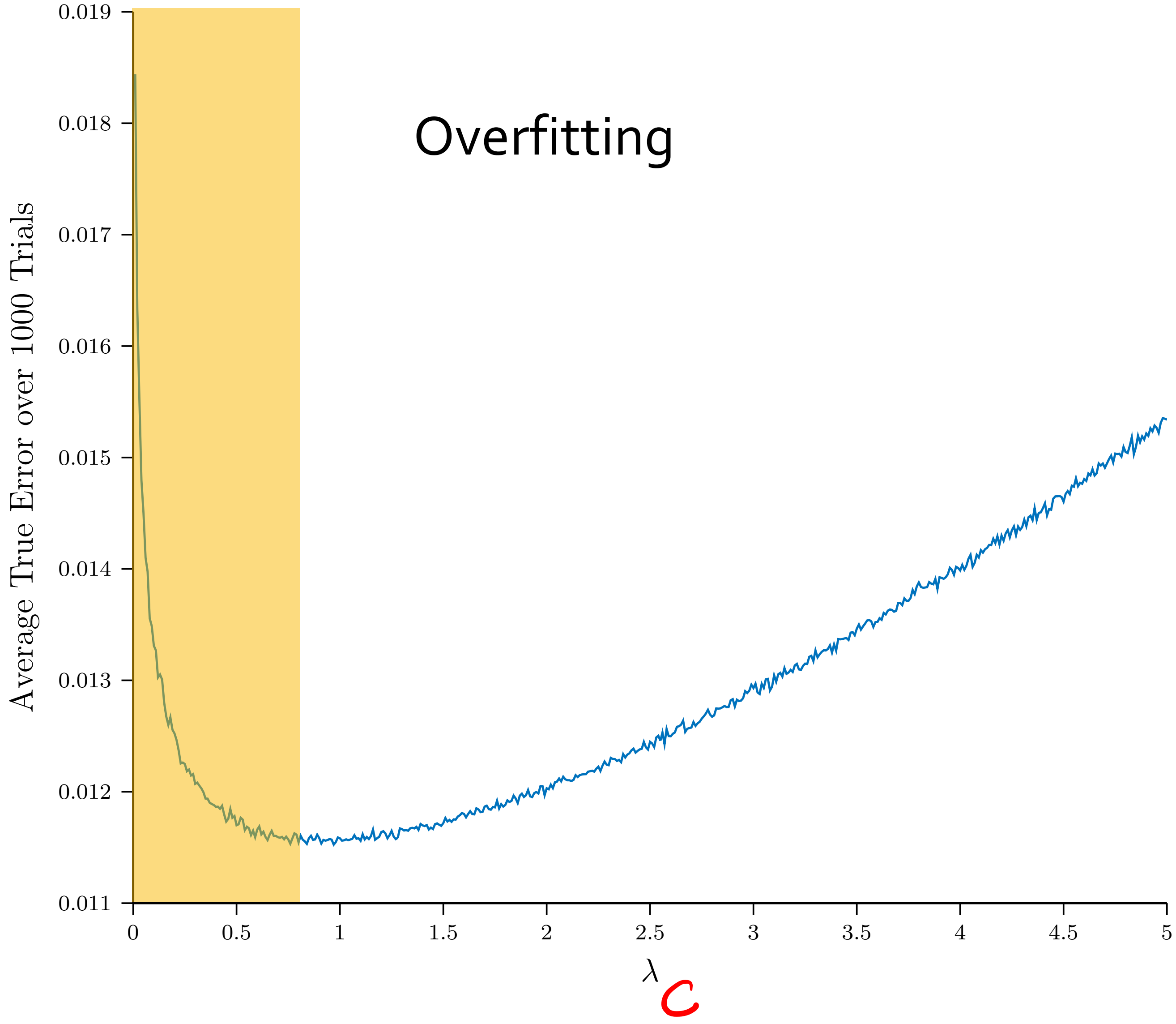
Ridge Regression

	$\boldsymbol{\lambda_C = 0}$
True Error	0.059
	Overfit

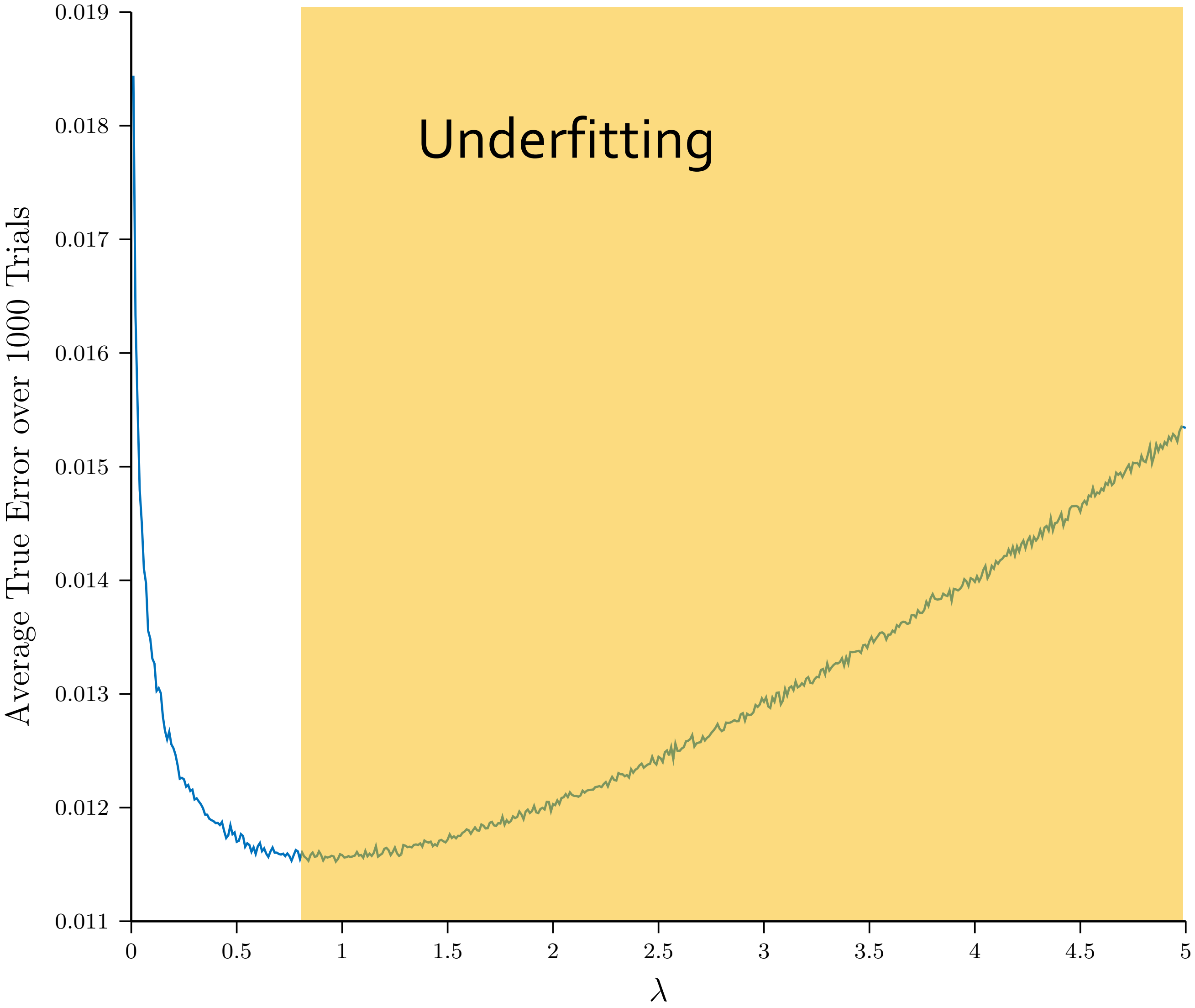
Setting λ



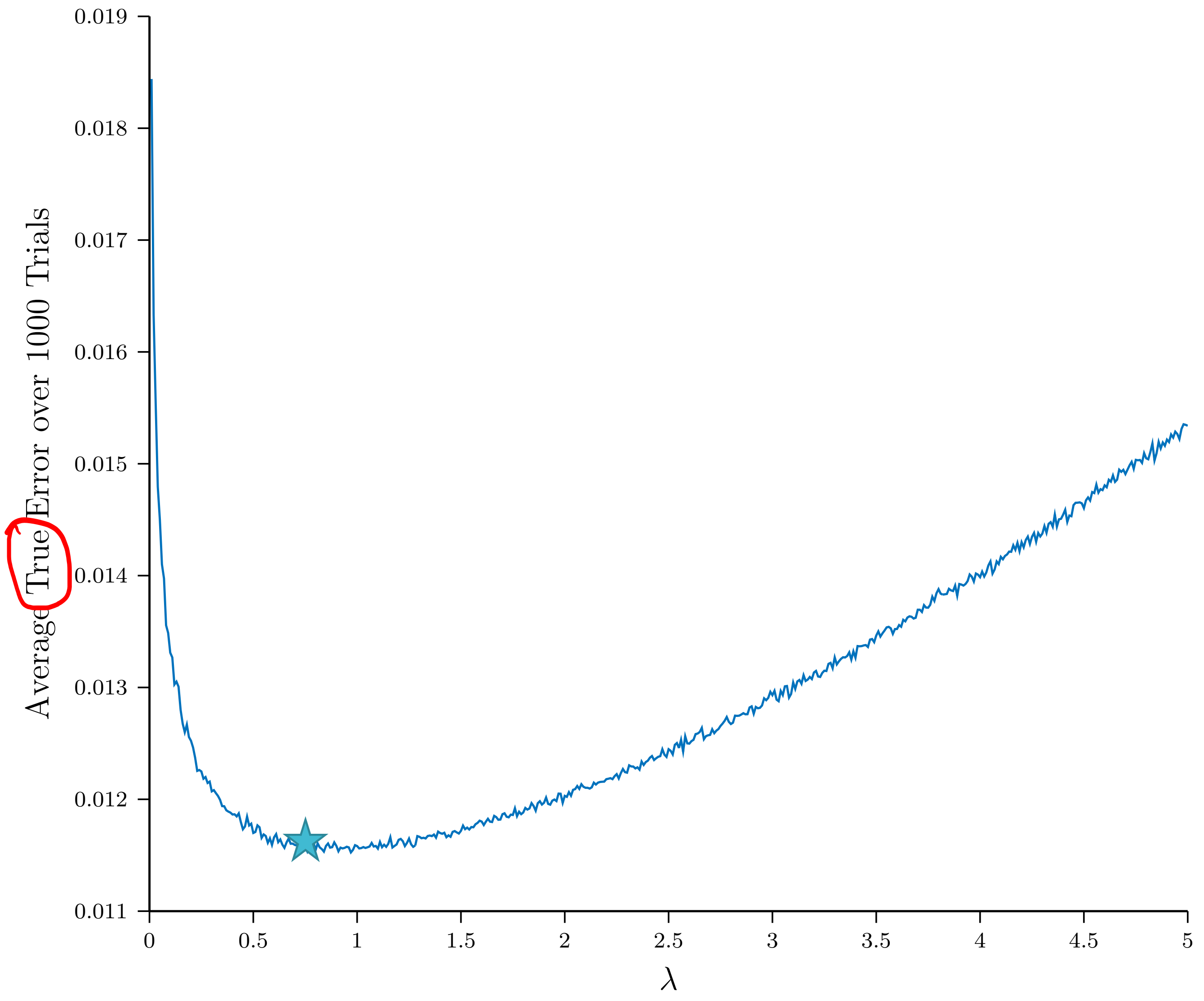
Setting λ



Setting λ

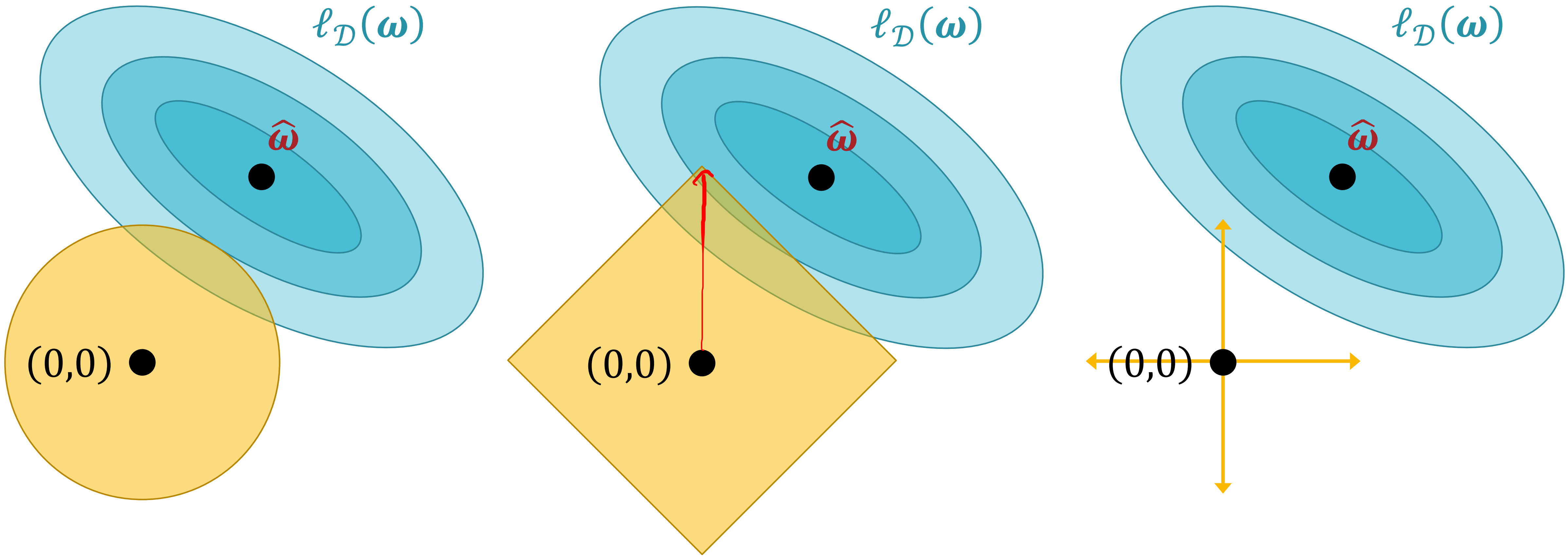


Setting λ



Other Regularizers

$\ell_{\mathcal{D}}(\boldsymbol{\omega}) + \lambda r(\boldsymbol{\omega})$		
Ridge or $L2$	$r(\boldsymbol{\omega}) = \lVert\boldsymbol{\omega}\rVert_2^2 = \sum_{d=0}^{D} \omega_d^2$	Encourages small weights
Lasso or $L1$	$r(\boldsymbol{\omega}) = \lVert\boldsymbol{\omega}\rVert_1 = \sum_{d=0}^{D} \lvert\omega_d\rvert$	Encourages sparsity
$L0$	$r(\boldsymbol{\omega}) = \lVert\boldsymbol{\omega}\rVert_0 = \sum_{d=0}^{D} \mathbb{1}(\omega_d \neq 0)$	Encourages sparsity (intractable)



Ridge or $L2$

Lasso or $L1$

$L0$

Other Regularizers

M(C)LE for Linear Regression

- If we assume a linear model with additive Gaussian noise

$$y = \boldsymbol{\omega}^T \boldsymbol{x} + \epsilon \text{ where } \epsilon \sim N(0, \sigma^2) \rightarrow y \sim N(\boldsymbol{\omega}^T \boldsymbol{x}, \sigma^2)$$

- Then given $X = \begin{bmatrix} 1 & \boldsymbol{x}^{(1)} \\ 1 & \boldsymbol{x}^{(2)} \\ \vdots & \vdots \\ 1 & \boldsymbol{x}^{(N)} \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$ the MLE of $\boldsymbol{\omega}$ is

$$\widehat{\boldsymbol{\omega}} = \underset{\boldsymbol{\omega}}{\operatorname{argmax}} \log P(\boldsymbol{y} | X, \boldsymbol{\omega})$$

$$\vdots$$

$$= (X^T X)^{-1} X^T \boldsymbol{y}$$

MAP for Linear Regression

- If we assume a linear model with additive Gaussian noise

$$y = \boldsymbol{\omega}^T \boldsymbol{x} + \epsilon \text{ where } \epsilon \sim N(0, \sigma^2) \rightarrow y \sim N(\boldsymbol{\omega}^T \boldsymbol{x}, \sigma^2)$$

and independent Gaussian priors on all the weights…

$$\omega_d \sim N\left(0, \frac{\sigma^2}{\lambda}\right)$$

- … then, the MAP of $\boldsymbol{\omega}$ is the ridge regression solution!

$$\widehat{\boldsymbol{\omega}} = \underset{\boldsymbol{\omega}}{\operatorname{argmax}} \log P(\boldsymbol{\omega} | X, \boldsymbol{y}) = \underset{\boldsymbol{\omega}}{\operatorname{argmax}} \log P(\boldsymbol{y} | X, \boldsymbol{\omega}) P(\boldsymbol{\omega})$$

$$\vdots$$

$$= (X^T X + \lambda_C I_{D+1})^{-1} X^T \boldsymbol{y}$$

MAP for Linear Regression

- If we assume a linear model with additive Gaussian noise

$$y = \boldsymbol{\omega}^T \boldsymbol{x} + \epsilon \text{ where } \epsilon \sim N(0, \sigma^2) \rightarrow y \sim N(\boldsymbol{\omega}^T \boldsymbol{x}, \sigma^2)$$

and independent *Laplace* priors on all the weights…

$$\omega_d \sim \text{Laplace}\left(0, \frac{2\sigma^2}{\lambda}\right)$$

- … then, the MAP of $\boldsymbol{\omega}$ is the Lasso regression solution!
- No closed form solution exists but we can solve via (sub−)gradient descent

Key Takeaways

- Polynomial/non-linear feature transformations allow for learning non-linear functions/decision boundaries
  - Can lead to overfitting…
    - Address with regularization!
    - Analogous to constrained optimization, solve via method of Lagrange multipliers
    - Regularization level is a hyperparameter
  - Can be computationally expensive…
    - Address with kernels!
    - Alternative to explicitly computing feature transformations for inner product methods