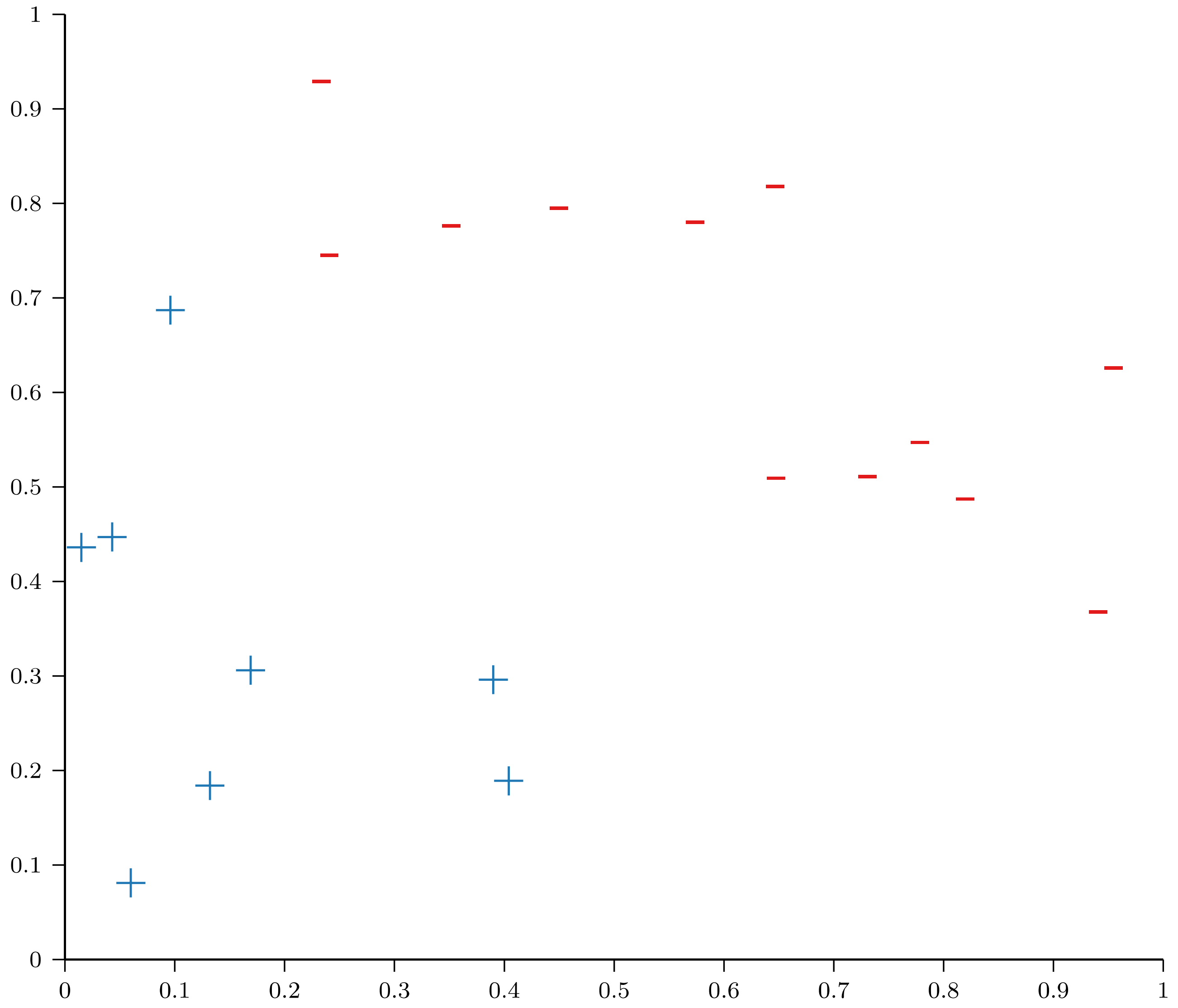
10-701: Introduction to Machine Learning

Lecture 8 – Regularization

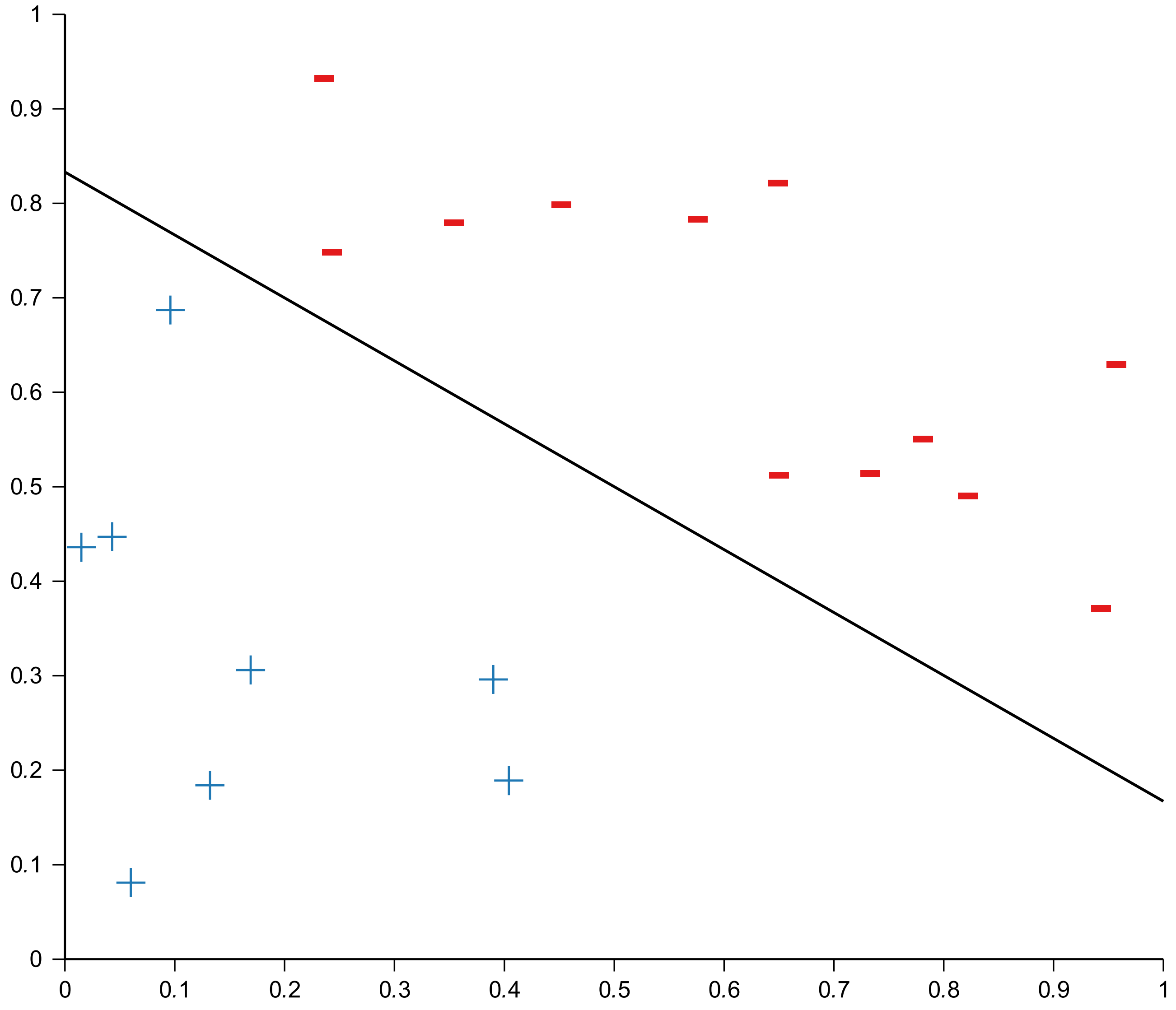
Hoda Heidari

* Slides adopted from F24 offering of 10701 by Henry Chai.

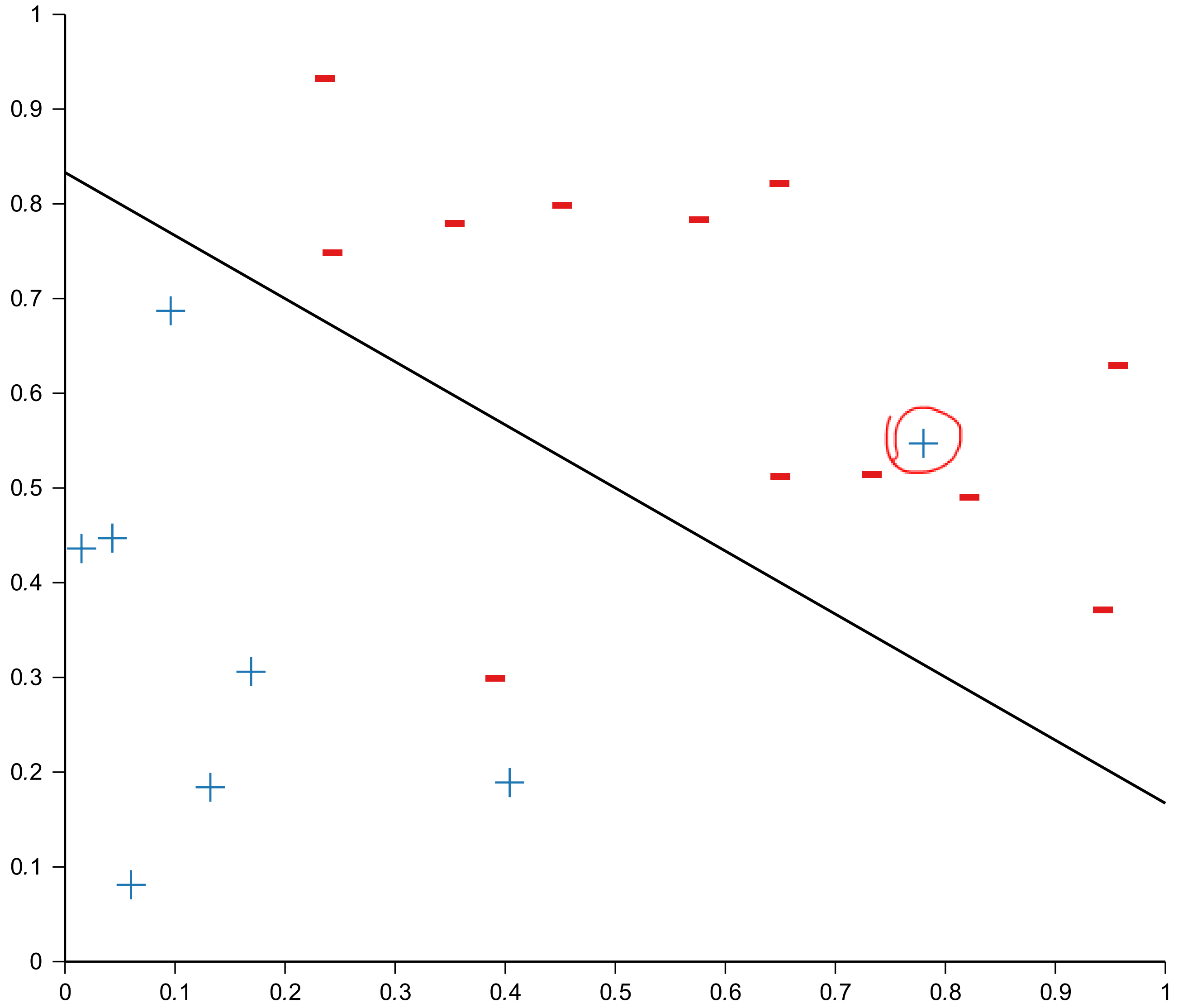
Linear Models



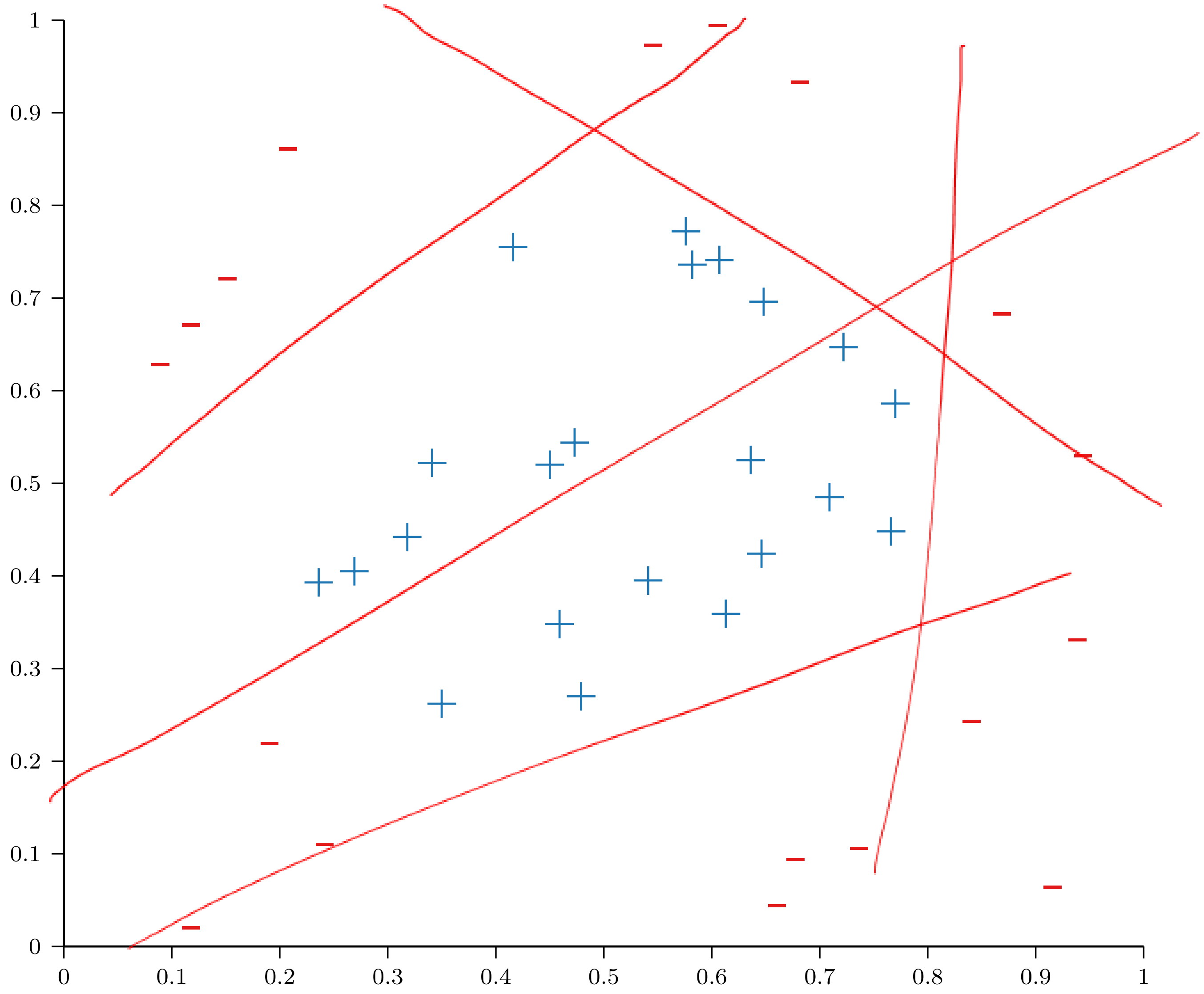
Linear Models



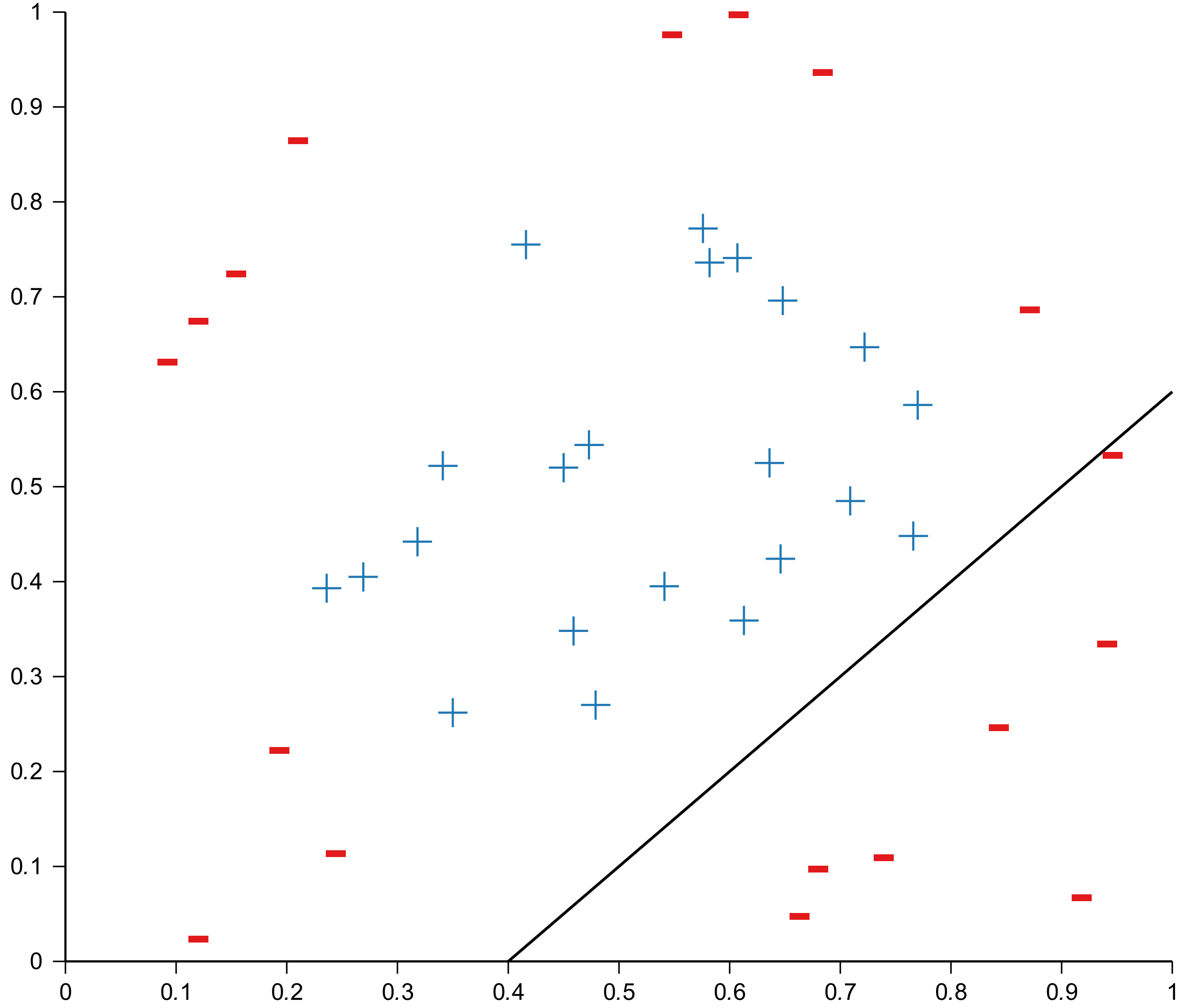
Linear Models



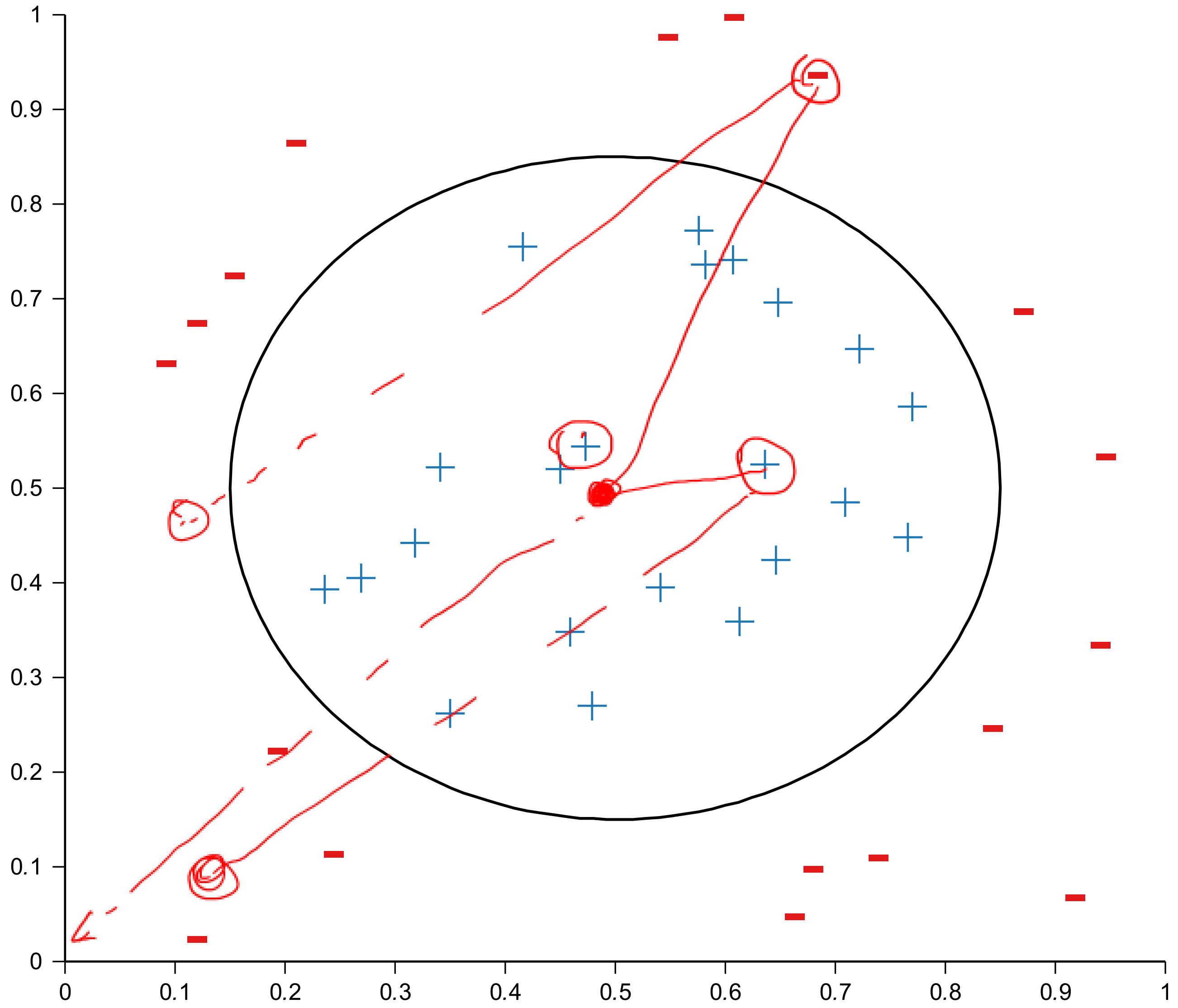
Linear Models?



Linear Models?



Nonlinear Models



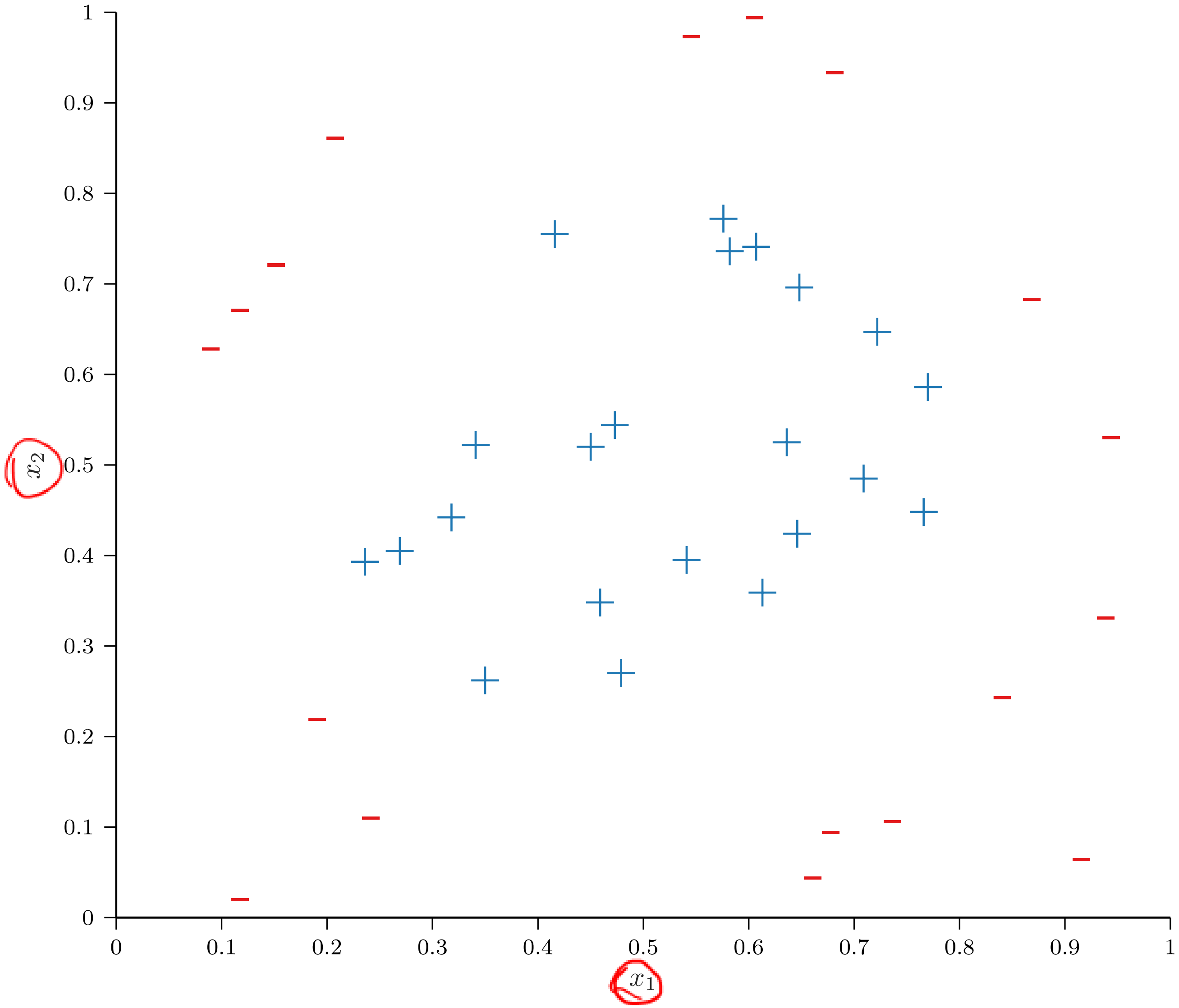
Feature Transforms

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

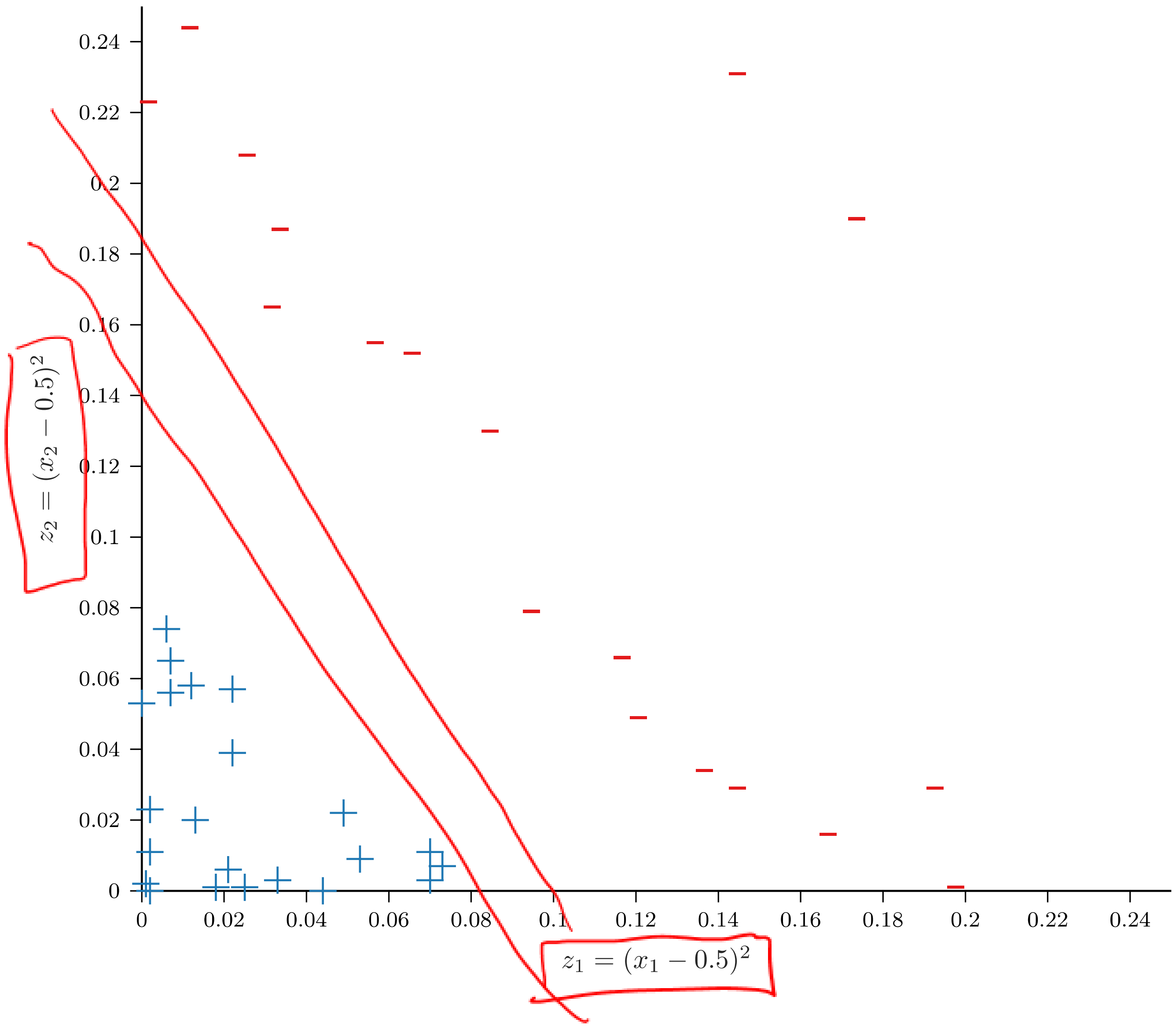
$$\phi([x_1, x_2]) = [z_1 = (x_1 - 0.5)^2, z_2 = (x_2 - 0.5)^2]$$

(0.5, 0.5) is the center of the red circle.

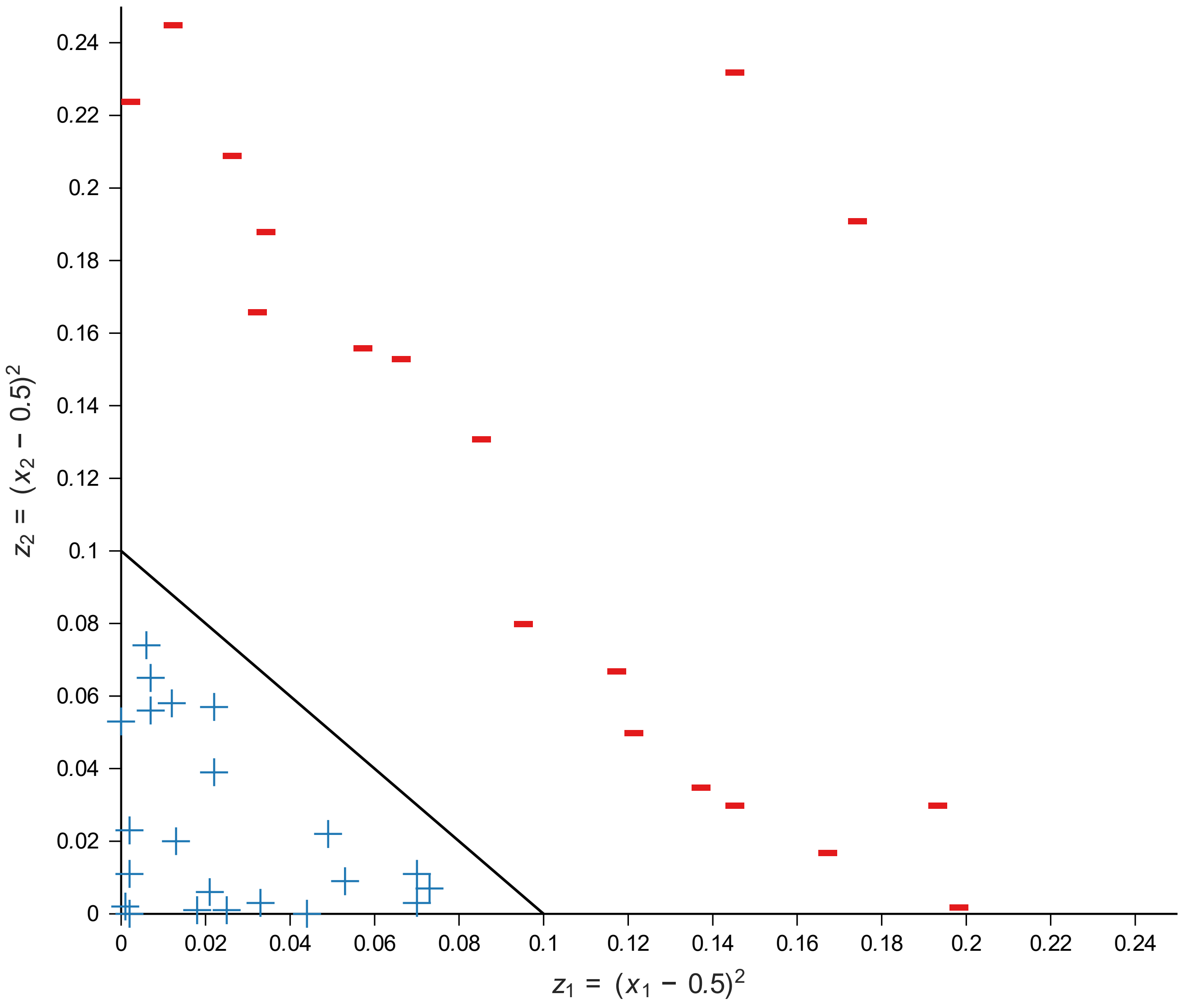
Nonlinear Models



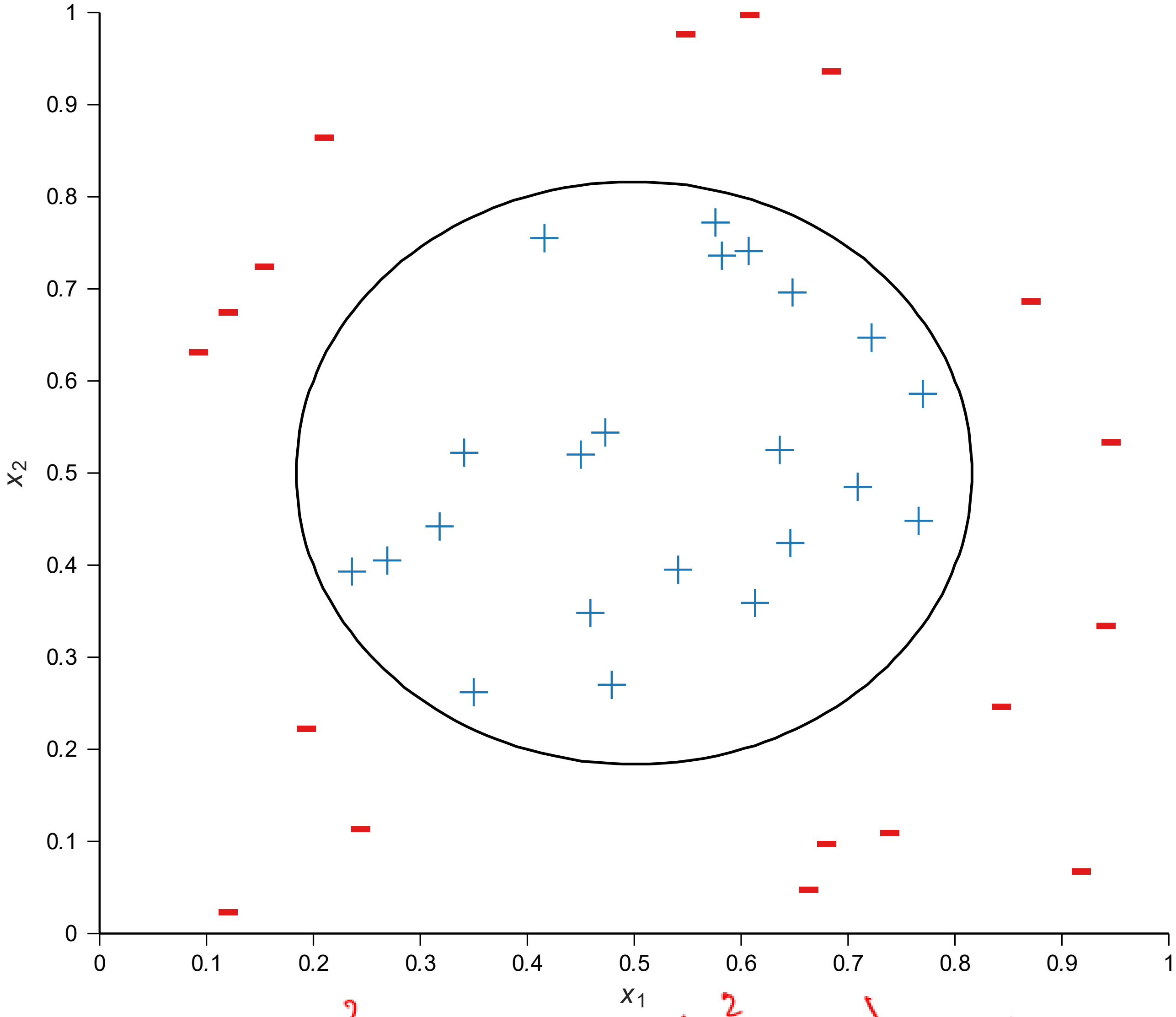
Nonlinear Models



Nonlinear Models



Nonlinear Models



$$\underbrace{w_1(x_1^2 + \cdots)}_{z_1} + \underbrace{w_2(x_2^2 + \cdots)}_{z_2} + w_0$$

General Q^{th}-order Transforms

\# of features or D

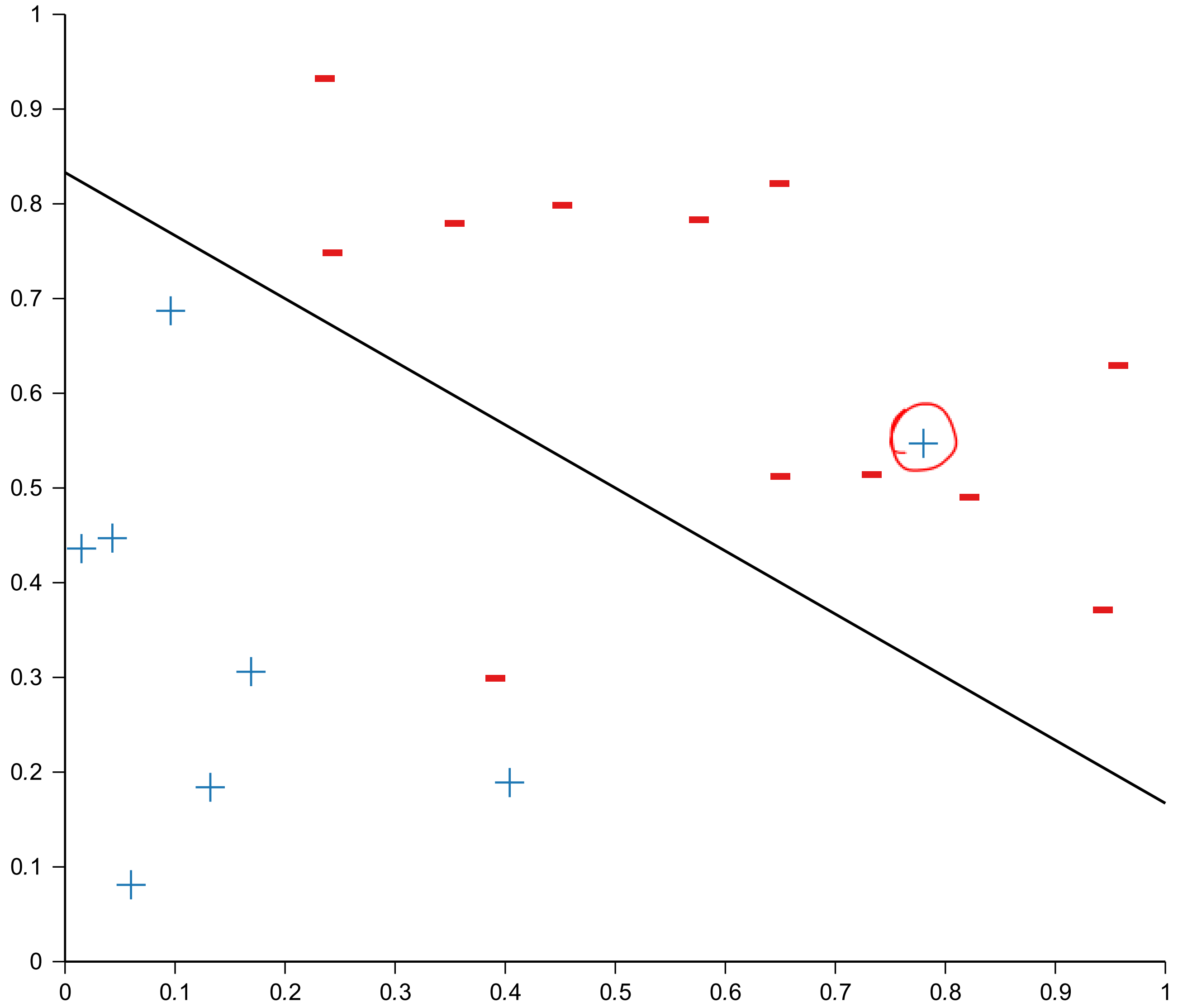
highest degree polynomial term allowed

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

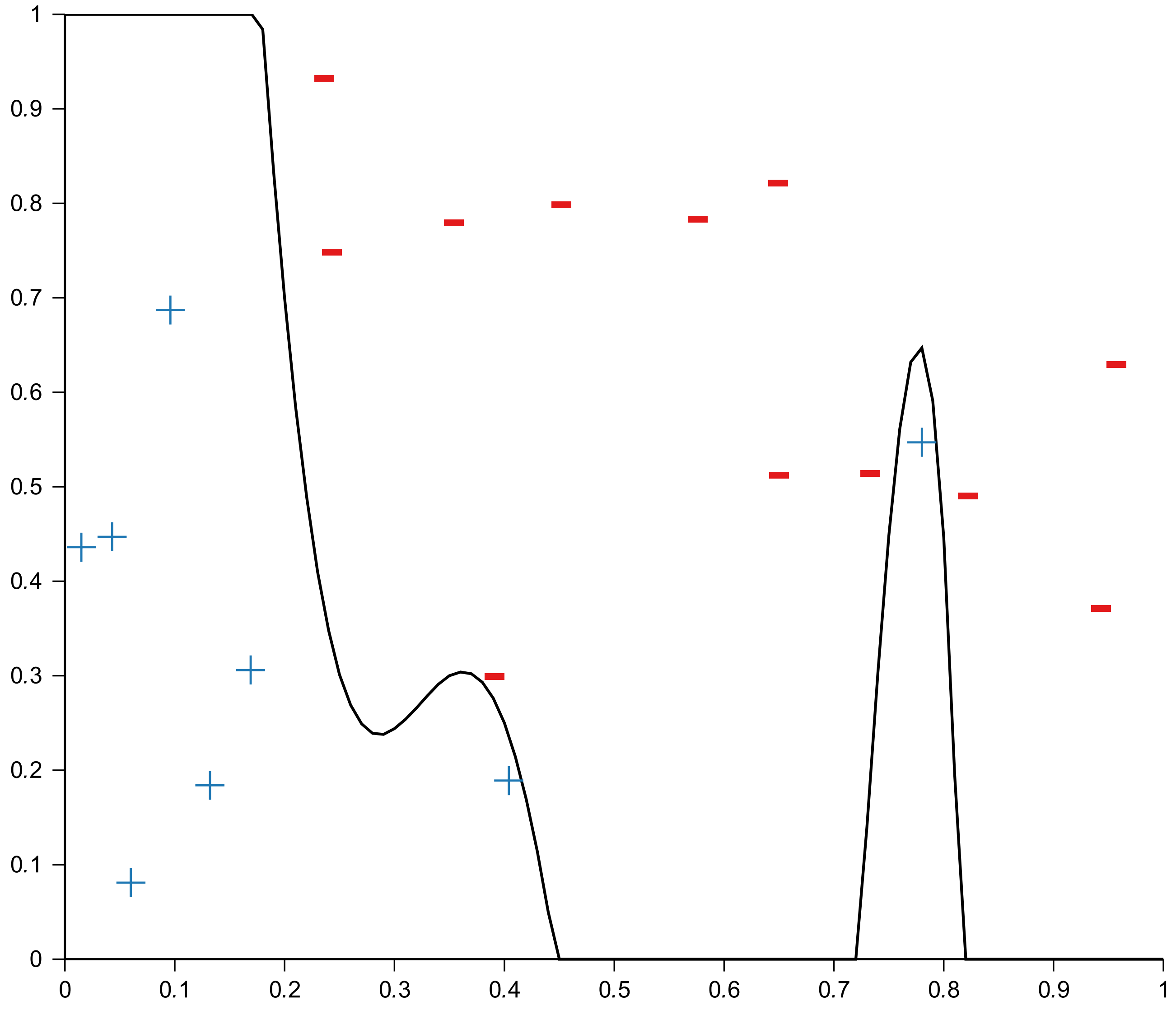
If feature space is D-dimensional, $\phi_{D,Q}$ is defined similarly

how many features are in $\phi_{D,Q}$? $O(Q^D)$ $\binom{Q+D}{D}$

Linear Models



Nonlinear Models?



Feature Transforms: Tradeoffs

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

Feature Transforms: Experiment

1-dim feature space

- $x \in \mathbb{R}$, $y \in \mathbb{R}$ and $N = 20$ $D = \{(x_1, y_1)(x_2, y_2) \ldots (x_{20}, y_{20})\}$

- Targets are generated by a 10^{th}-order polynomial in x with additive Gaussian noise:

$$y_i = \underbrace{\sum_{d=0}^{10} a_d x_i^d}_{} + \epsilon \text{ where } \epsilon \sim \underbrace{N(0, \sigma^2)}_{}$$

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

In-Class Poll:

- $x \in \mathbb{R}$, $y \in \mathbb{R}$ and $N = 20$

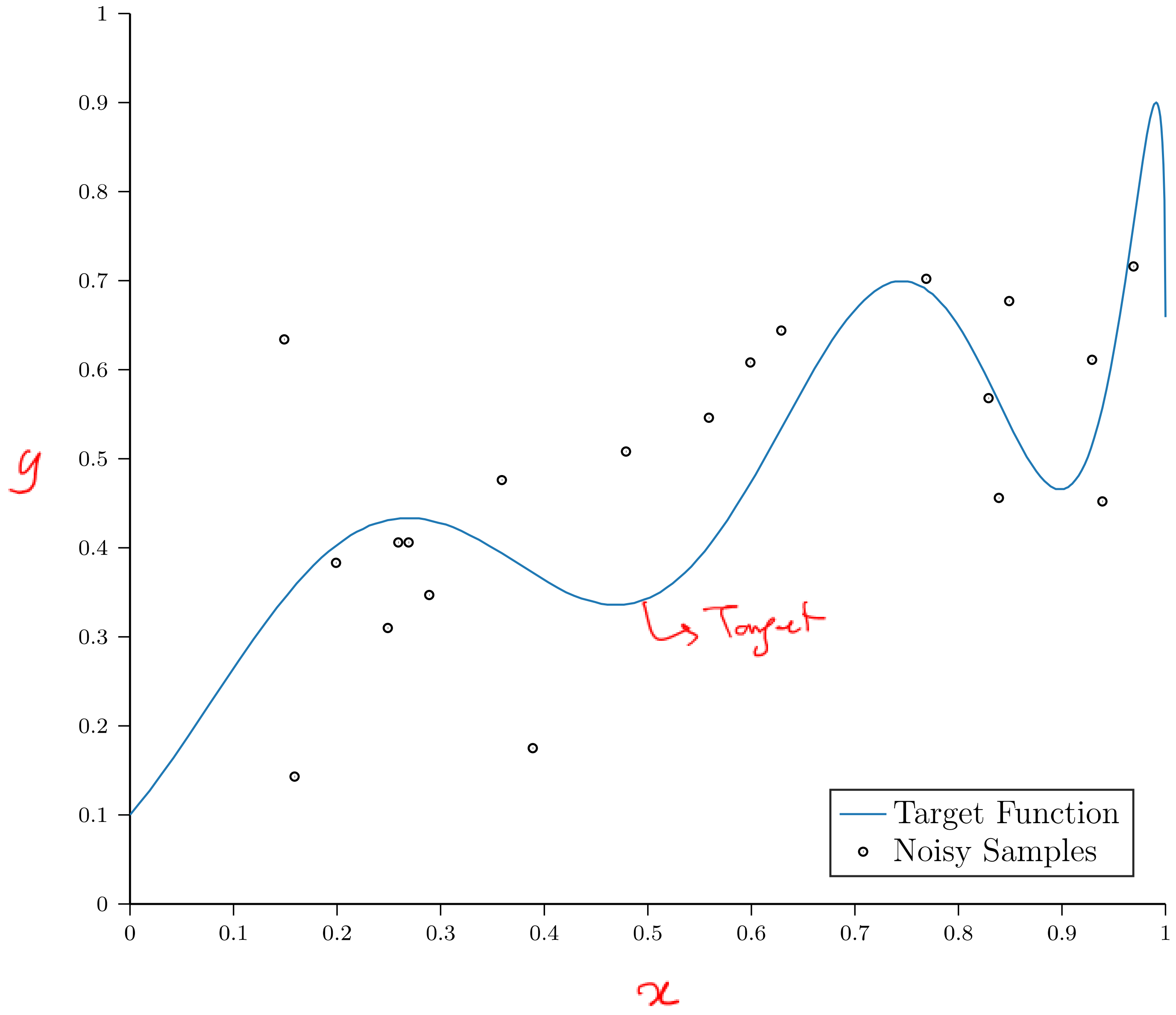
$$y = \sum_{d=0}^{10} a_d x^d + \epsilon \quad \text{where } \epsilon \sim N(0, \sigma^2)$$

- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomials
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomials

Which hypothesis space ($\mathcal{H}_2$ or $\mathcal{H}_{10}$) is a better choice to fit to this data set?

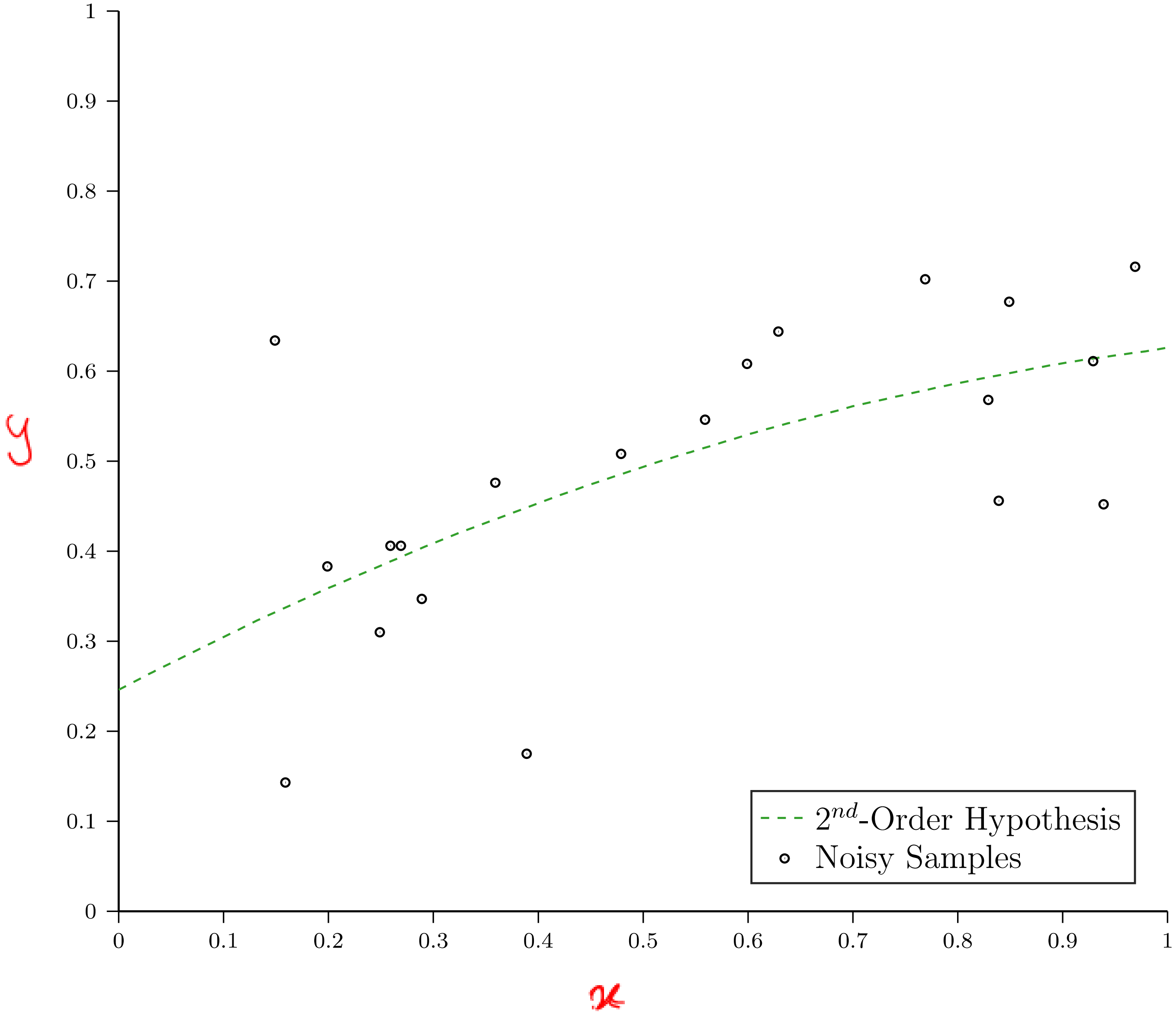
Noisy Targets

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



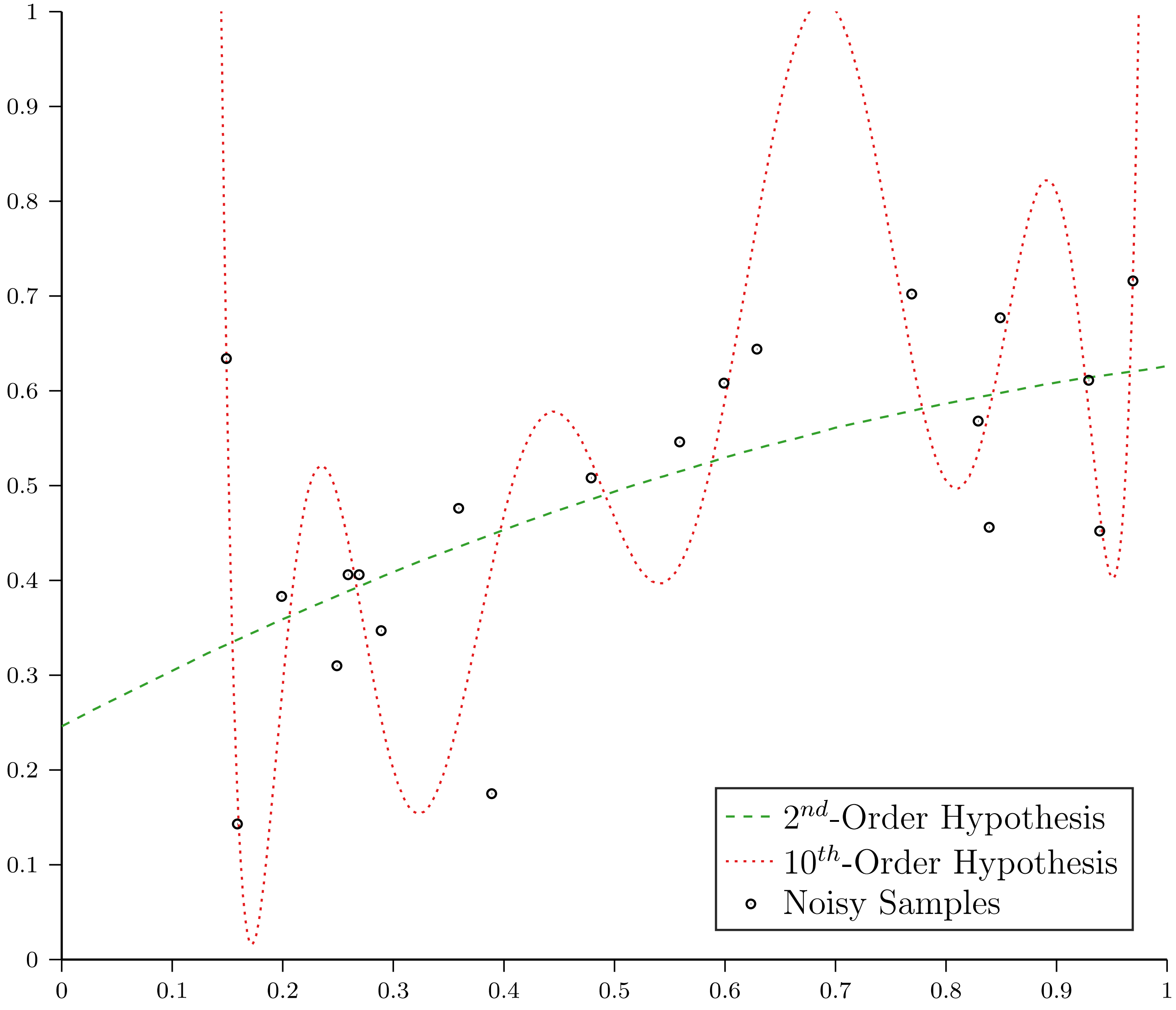
Noisy Targets

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



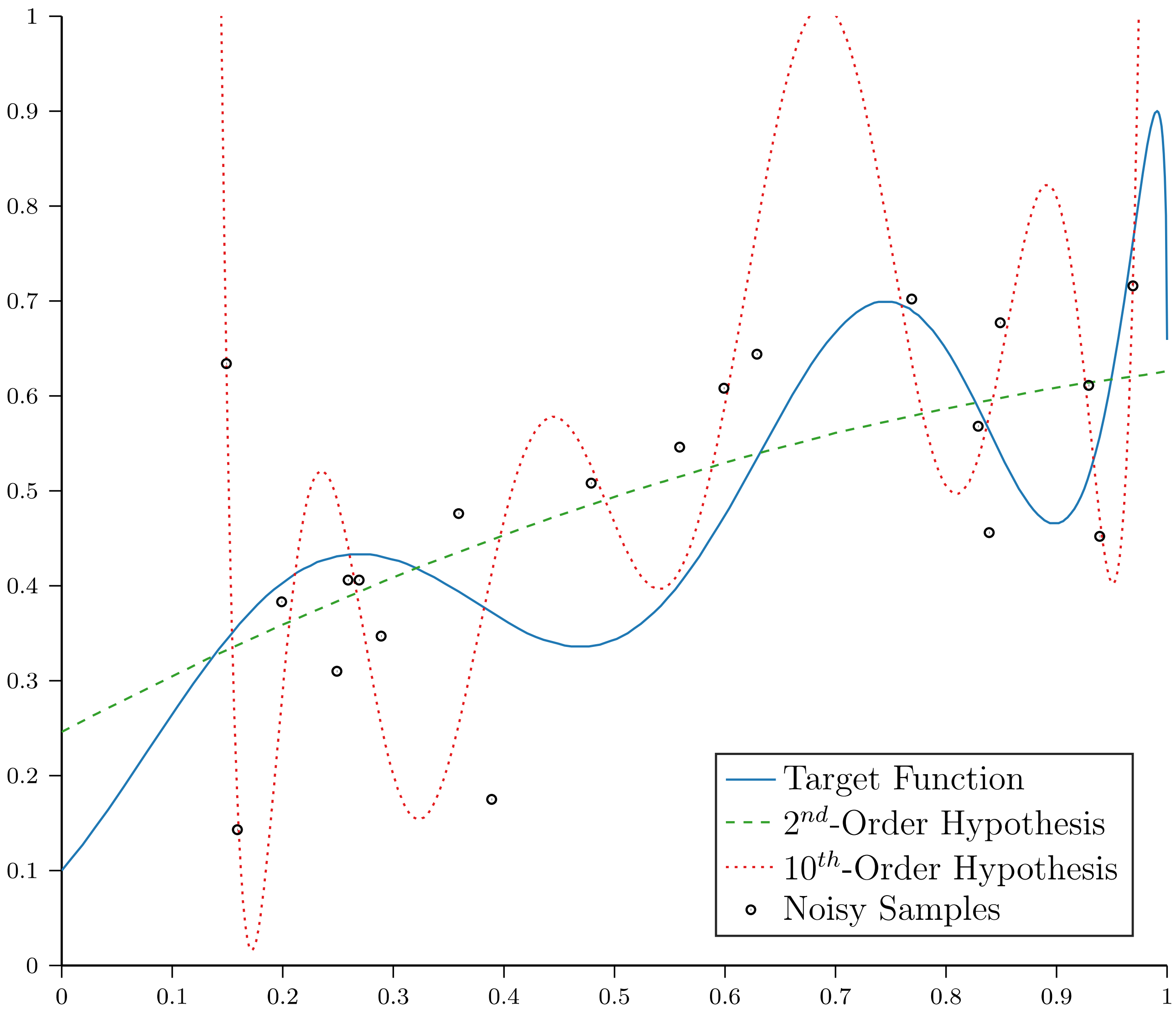
Noisy Targets

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



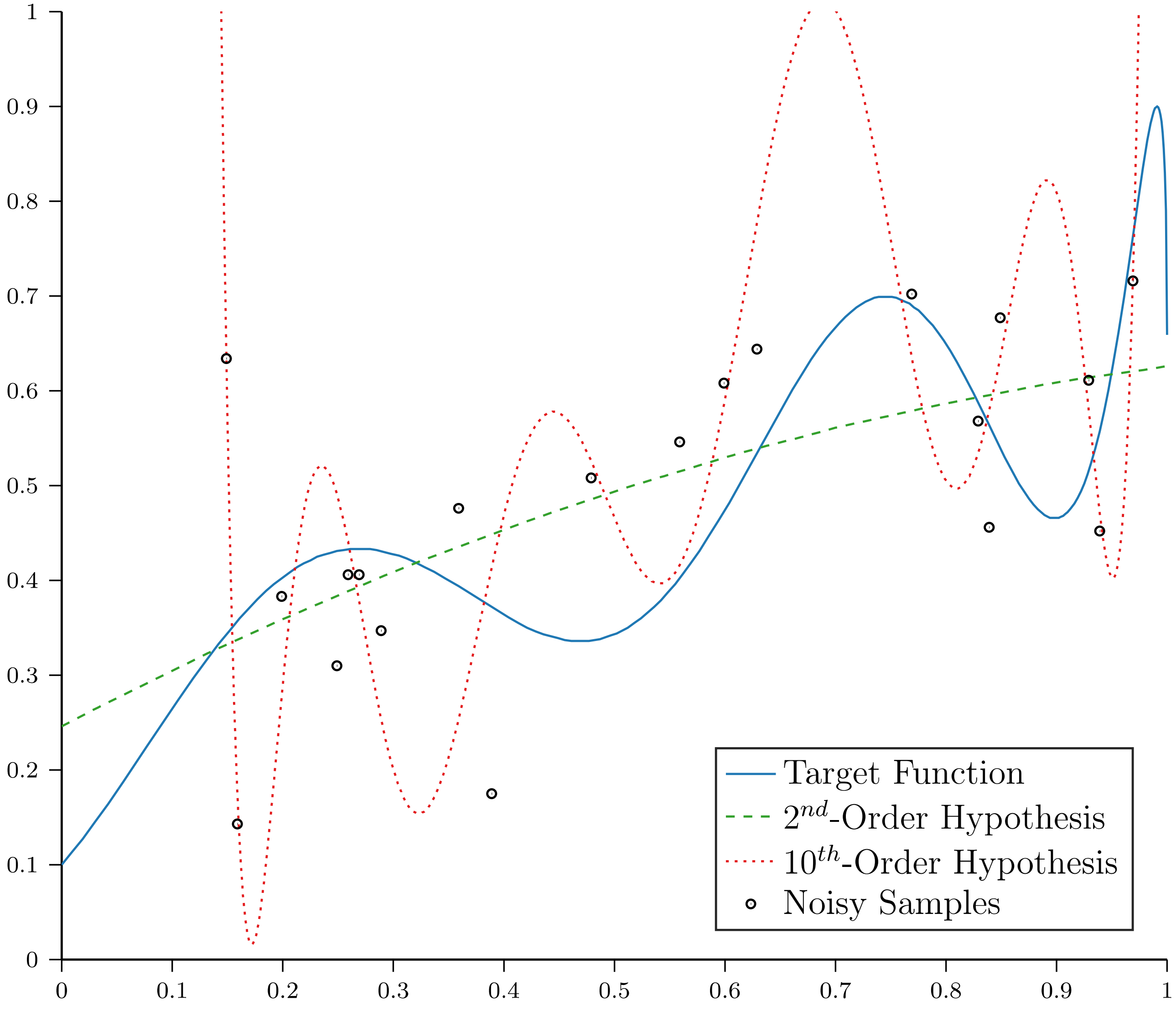
Noisy Targets

- 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

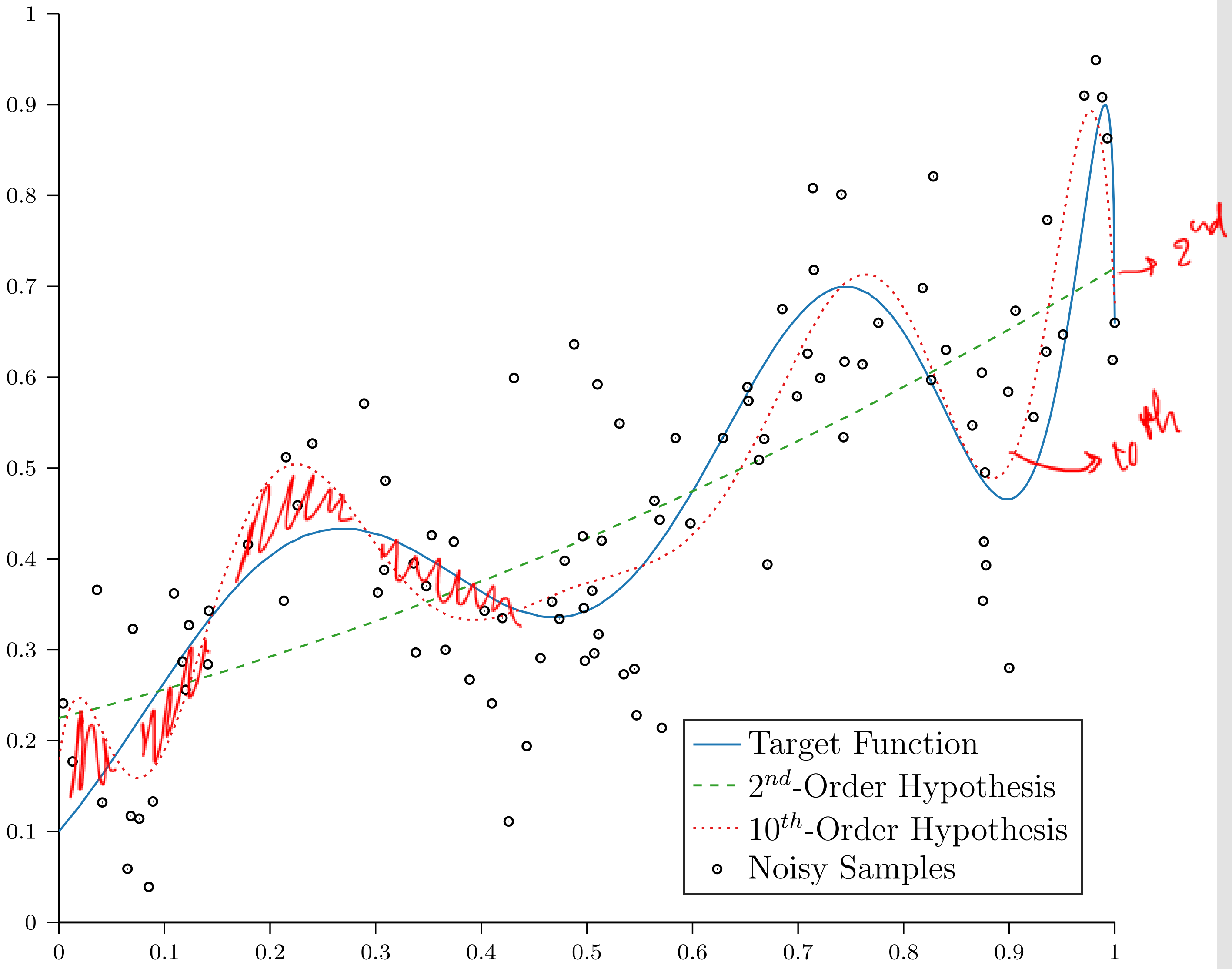
	$\mathcal{H}_2$	$\mathcal{H}_{10}$
Training Error	0.016	0.011
True Error	0.009	3797



More Data

Test error

	$\mathcal{H}_2$	$\mathcal{H}_{10}$
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 $D = \{(x_1, y_1), \ldots, (x_{20}, y_{20})\}$
  - $\phi_{1,10}(x) = [x, x^2, x^3, x^4, x^5, x^6, x^7, x^8, x^9, x^{10}]$
- 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

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

- Subject to

  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

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

- Subject to nothing!

Hard Constraints

- $\mathcal{H}_2 = 2^{\text{nd}}$-order polynomials
  - $\phi_{1,2}(x) = \boxed{[x, x^2]}$
- 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

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

- Subject to nothing!

Soft Constraints

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

transformed design matrix

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

  find $\boldsymbol{\omega}$ that minimizes

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

- Subject to: Some constraint on $\boldsymbol{\omega}$, e.g.,

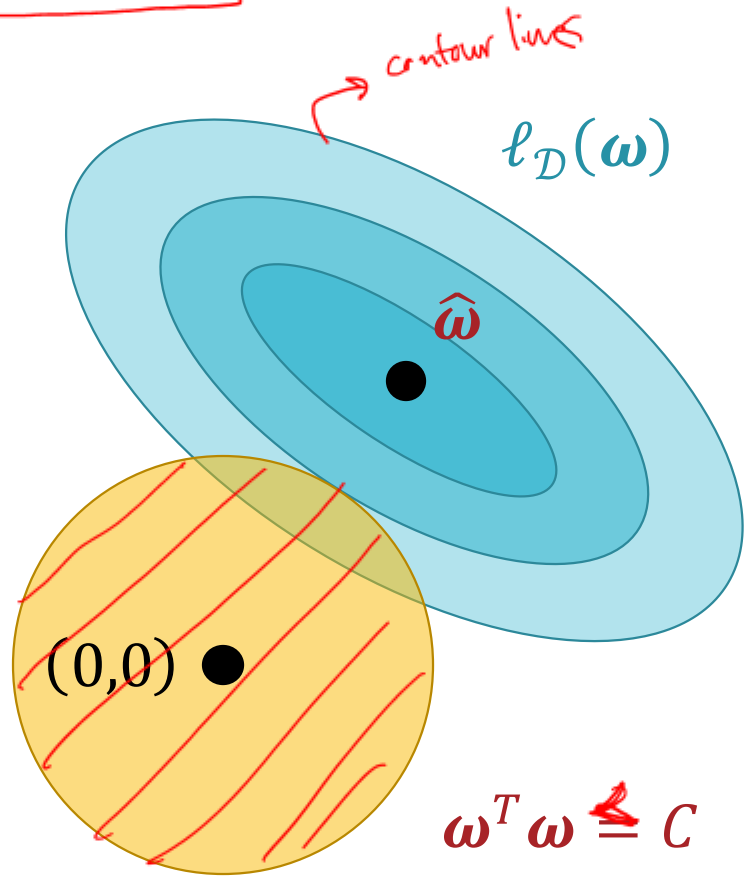
Ridge ~~regression~~ regularizer $\|\boldsymbol{\omega}\|_2^2 = \boldsymbol{\omega}^T\boldsymbol{\omega} = \underbrace{\sum_{d=0}^{D} \omega_d^2 \leq C}$

Soft Constraints

constrained optimization

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

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

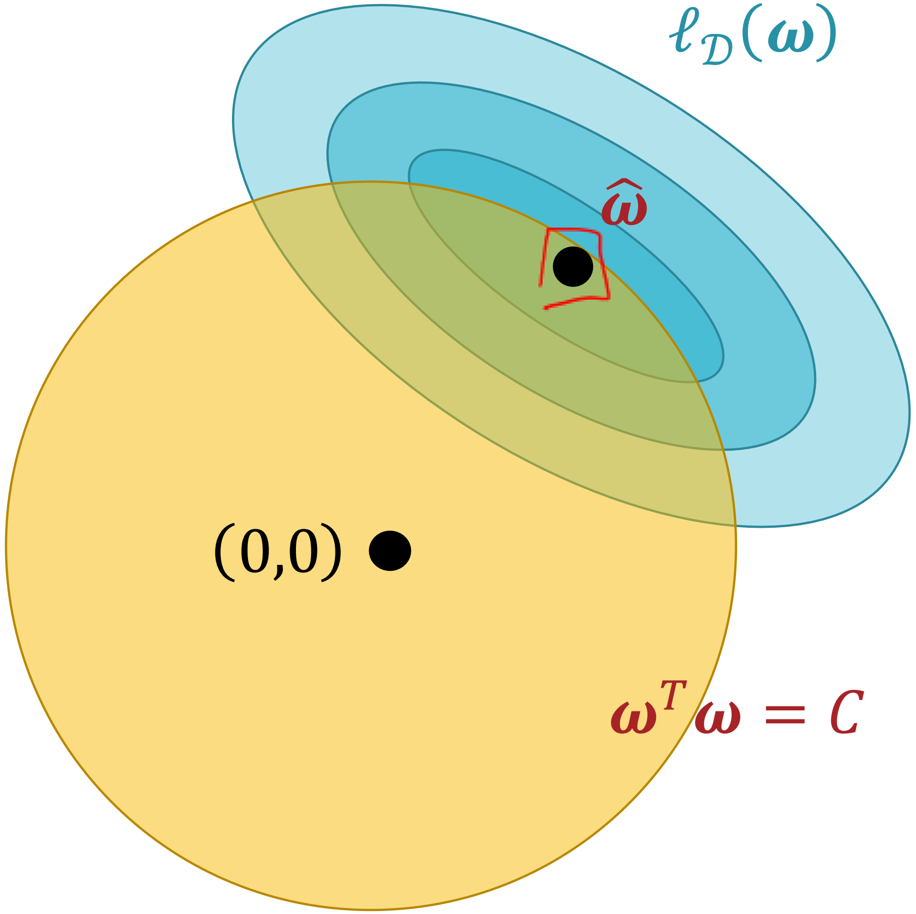


$\ell_{\mathcal{D}}(\omega) + 0.00001\, \omega^T \omega$

Soft Constraints

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

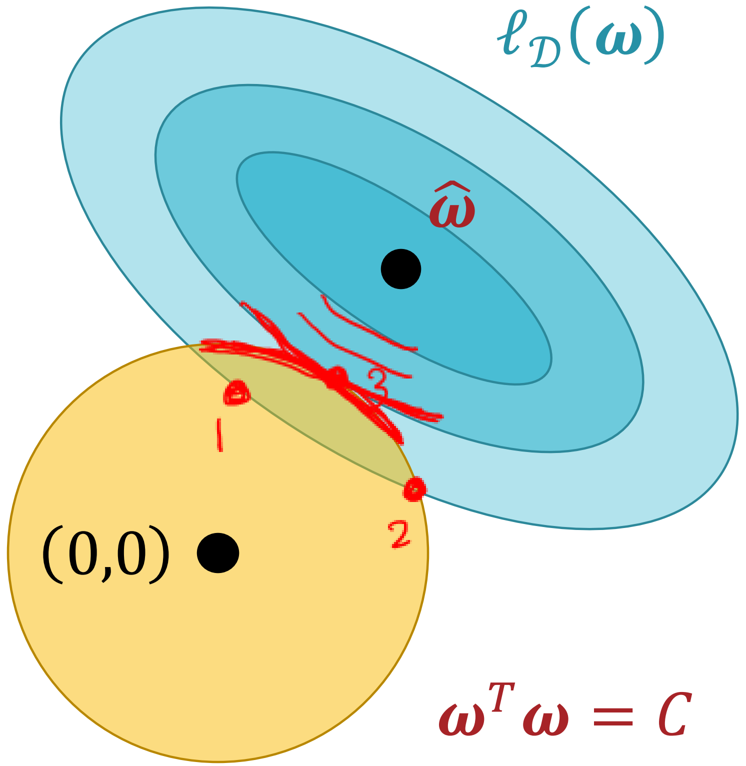
subject to $\boldsymbol{\omega}^T\boldsymbol{\omega} \leq C$



Soft Constraints

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

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



Soft Constraints

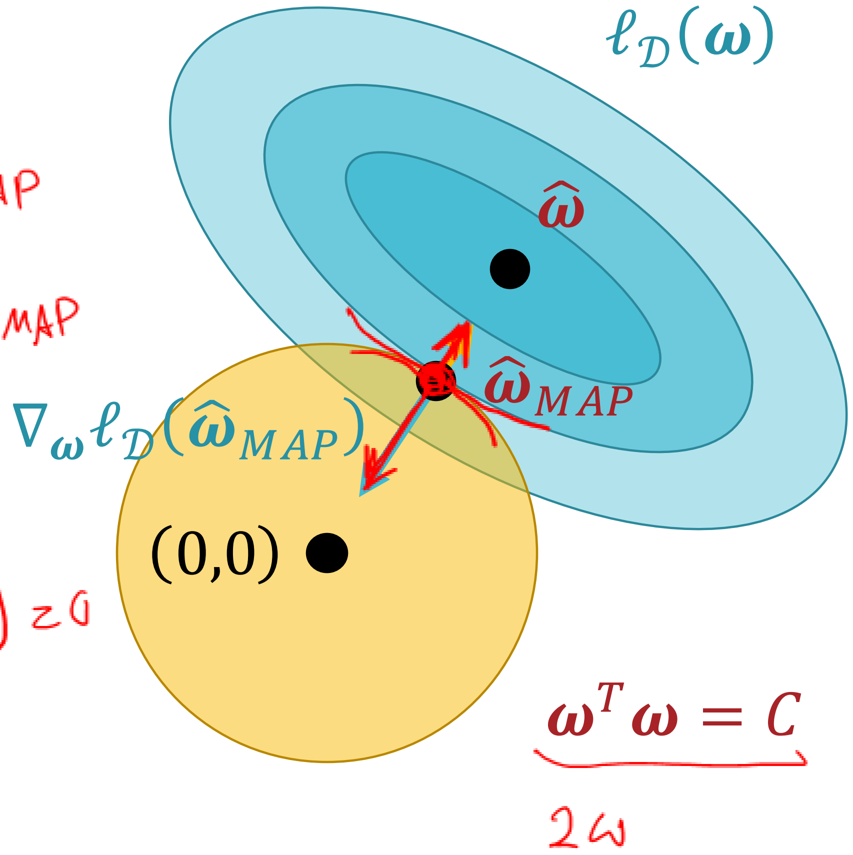
$$\begin{cases} \text{minimize } \ell_{\mathcal{D}}(\boldsymbol{\omega}) = (\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})^T(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y}) \\ \text{subject to } \boldsymbol{\omega}^T\boldsymbol{\omega} \le C \end{cases}$$

$\nabla_\omega \ell_D(\hat{\omega}_{MAP}) \propto -2\hat{\omega}_{MAP}$

$\Leftrightarrow \nabla_\omega \ell_D(\hat{\omega}_{MAP}) = -2\lambda_C \hat{\omega}_{MAP}$

$\Leftrightarrow \nabla_\omega \ell_D(\hat{\omega}_{MAP}) + 2\lambda_C \hat{\omega}_{MAP} = 0$

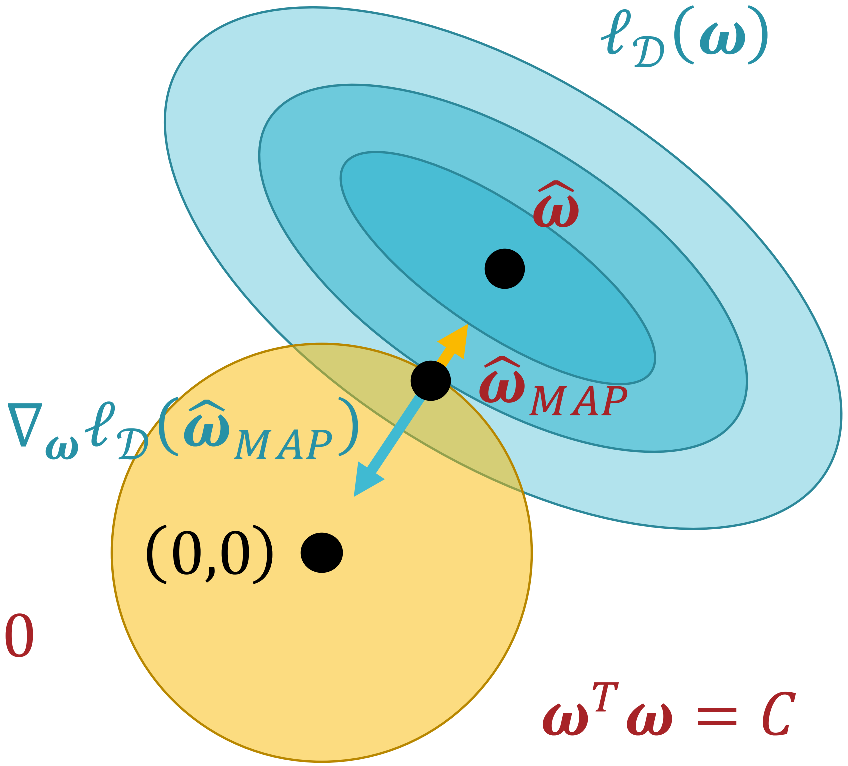
$\Leftrightarrow \nabla_\omega \Big(\underbrace{\ell_D(\hat{\omega}_{MAP}) + \lambda_C \hat{\omega}_{MAP}^T \hat{\omega}_{MAP}}_{\text{new objective function for unconstrained problem.}} \Big) = 0$



Soft Constraints

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

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



$$\nabla_{\omega}\ell_{\mathcal{D}}(\widehat{\boldsymbol{\omega}}_{MAP}) \propto -2\widehat{\boldsymbol{\omega}}_{MAP}$$

$$\nabla_{\omega}\ell_{\mathcal{D}}(\widehat{\boldsymbol{\omega}}_{MAP}) = -2\lambda_C\widehat{\boldsymbol{\omega}}_{MAP}$$

$$\nabla_{\omega}\ell_{\mathcal{D}}(\widehat{\boldsymbol{\omega}}_{MAP}) + 2\lambda_C\widehat{\boldsymbol{\omega}}_{MAP} = 0$$

$$\nabla_{\omega}\left(\ell_{\mathcal{D}}(\widehat{\boldsymbol{\omega}}_{MAP}) + \lambda_C(\widehat{\boldsymbol{\omega}}_{MAP})^T\widehat{\boldsymbol{\omega}}_{MAP}\right) = 0$$

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

$$\left\{\begin{array}{l}\text{minimize } \ell_{\mathcal{D}}(\boldsymbol{\omega}) = (\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y})^T(\mathrm{X}\boldsymbol{\omega} - \boldsymbol{y}) \\ \\ \text{subject to } \boldsymbol{\omega}^T\boldsymbol{\omega} \leq C\end{array}\right. \qquad (\star)$$

$$\Updownarrow$$

$$\text{minimize } \ell_{\mathcal{D}}^{AUG}(\boldsymbol{\omega}) = \underbrace{\ell_{\mathcal{D}}(\boldsymbol{\omega}) + \lambda_C\boldsymbol{\omega}^T\boldsymbol{\omega}} \qquad (\star) \leftarrow$$

$$\nabla_\omega \ell_D^{AUG}(\omega) = \left(2X^TX\omega - 2X^TY + 2\lambda_C\omega\right)$$

$$\Rightarrow 2X^TX\hat{\omega}_{MAP} - 2X^TY + 2\lambda_C\hat{\omega}_{MAP} = 0$$

$$\Rightarrow X^TX\hat{\omega}_{MAP} + \underbrace{\lambda_C\hat{\omega}_{MAP}}_{\equiv \lambda_C I_{D+1}\hat{\omega}_{MAP}} = X^TY$$

$$\Rightarrow \underbrace{(X^TX + \lambda_C I_{D+1})}\hat{\omega}_{MAP} = X^TY$$

$$\Rightarrow \hat{\omega}_{MAP} = (X^TX + \lambda_C I_{D+1})^{-1}X^TY$$

Ridge Regression

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

Ridge Regression

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

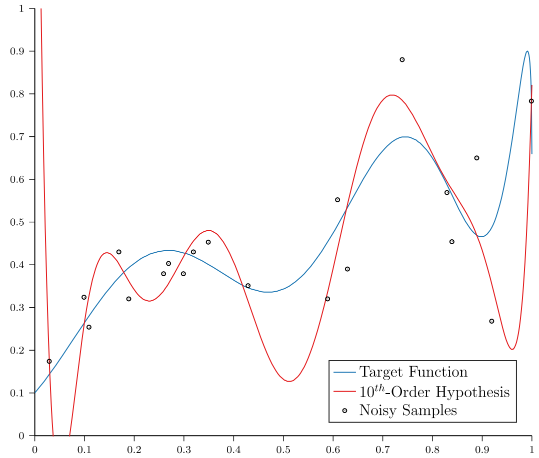
$$\nabla_{\boldsymbol{\omega}} \ell_{\mathcal{D}}^{AUG}(\boldsymbol{\omega}) = 2(X^T X \boldsymbol{\omega} - X^T \boldsymbol{y} + \lambda_C \boldsymbol{\omega})$$

$$2(X^T X \widehat{\boldsymbol{\omega}}_{MAP} - X^T \boldsymbol{y} + \lambda_C \widehat{\boldsymbol{\omega}}_{MAP}) = 0$$

$$(X^T X + \lambda_C I_{D+1}) \widehat{\boldsymbol{\omega}}_{MAP} = X^T \boldsymbol{y}$$

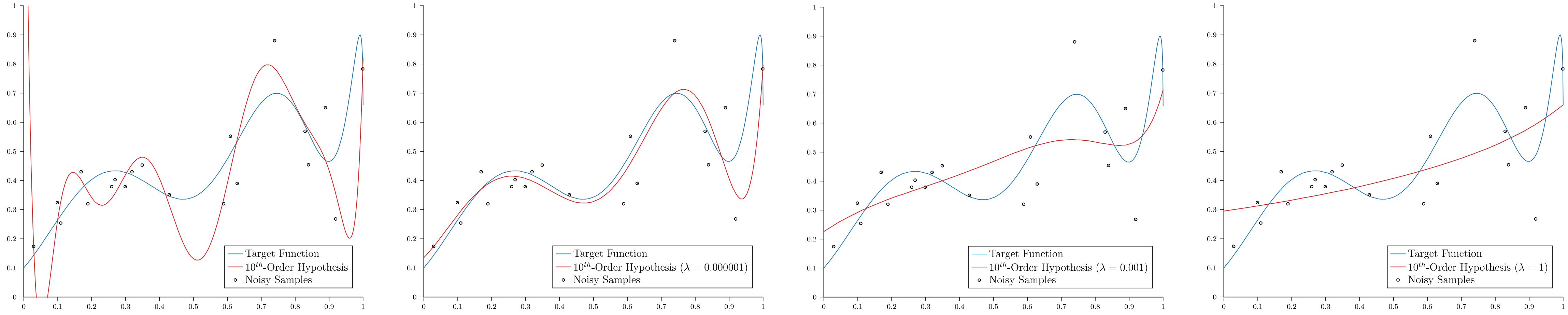
$$\widehat{\boldsymbol{\omega}}_{MAP} = (X^T X + \underbrace{\lambda_C I_{D+1}})^{-1} X^T \boldsymbol{y}$$

Adding this positive ($\lambda_C \geq 0$) diagonal matrix can help if $X^T X$ is not invertible!



Ridge Regression

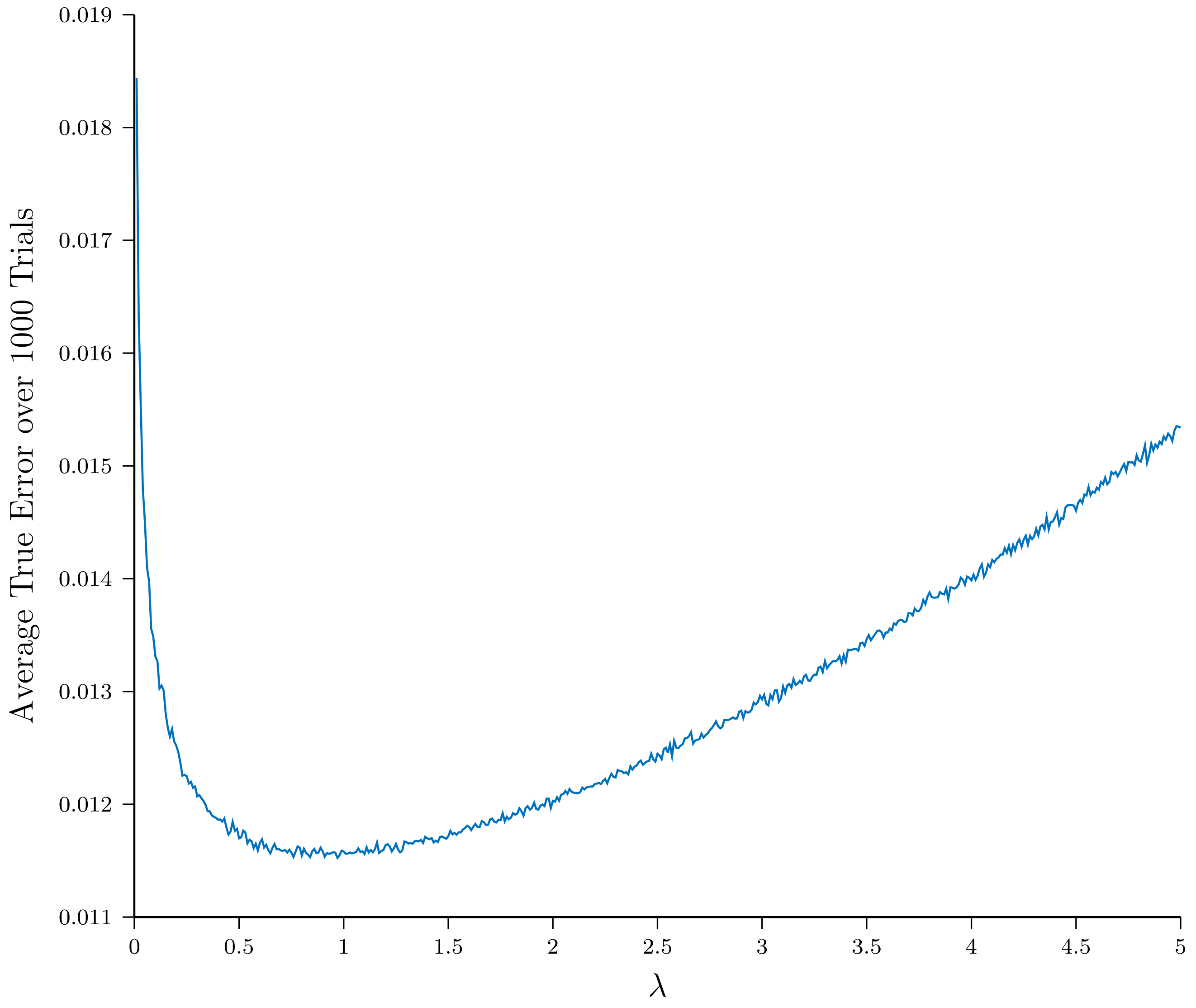
- 10-dimensional target function with additive Gaussian noise
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomial



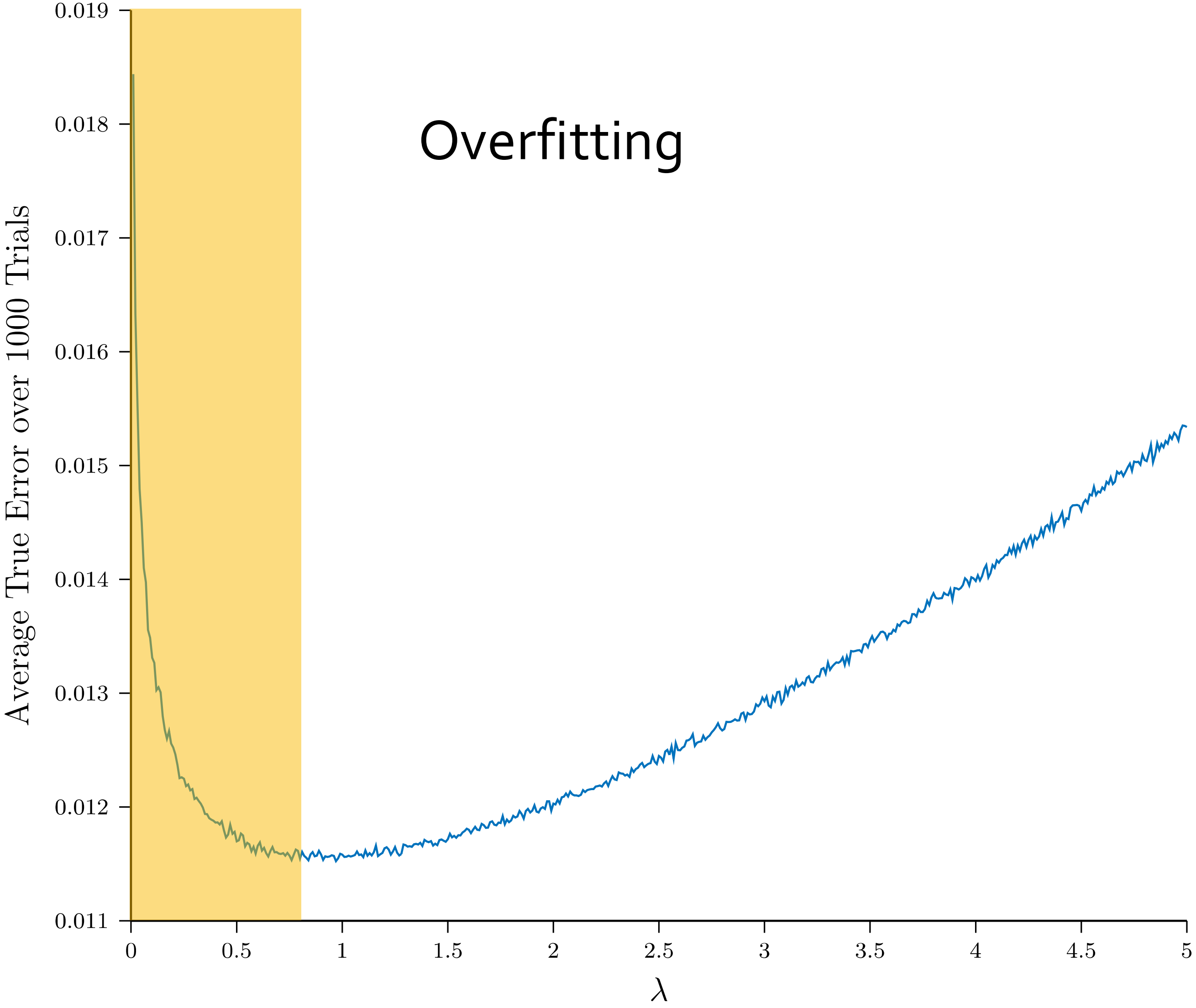
Ridge Regression

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

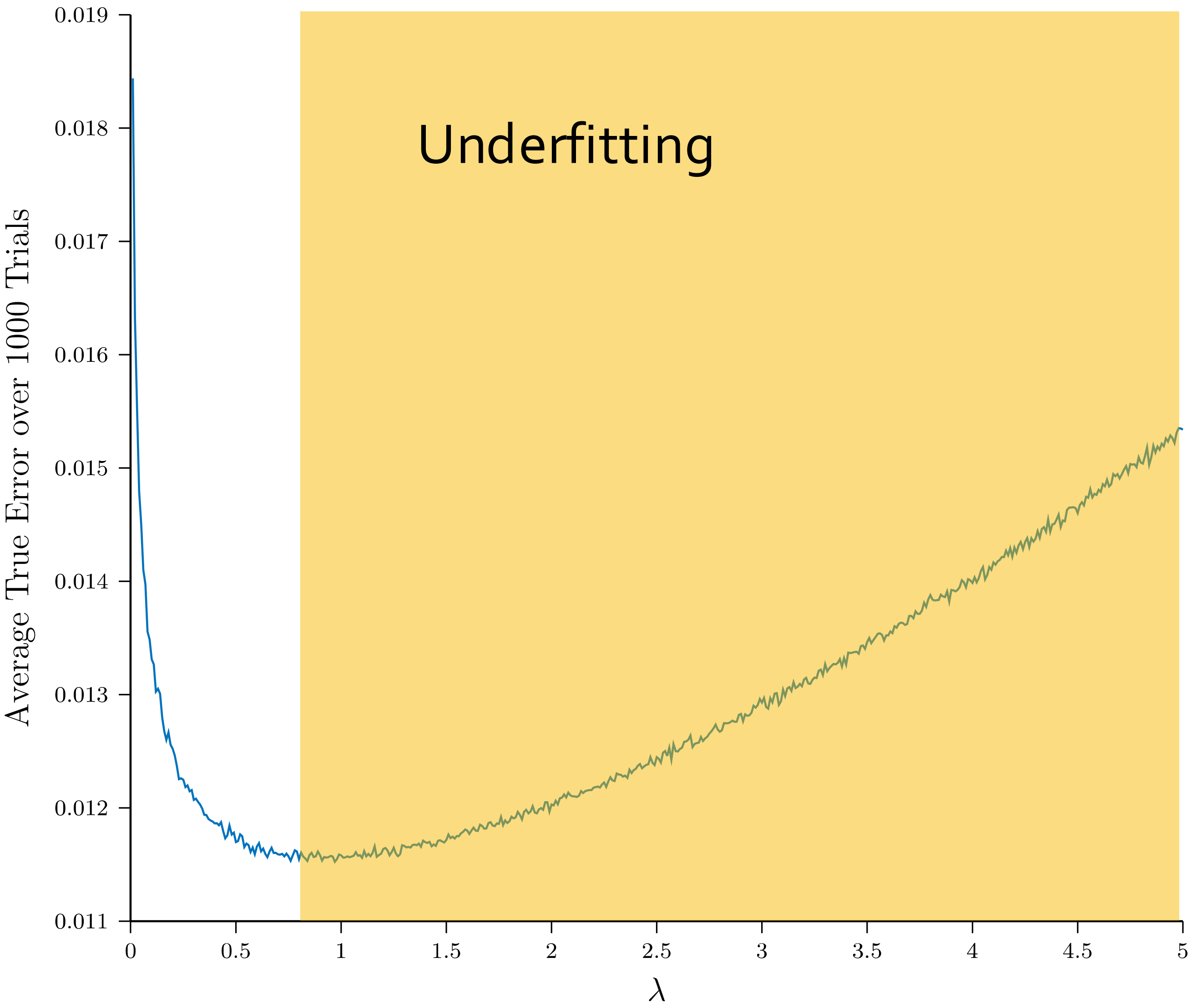
Setting λ



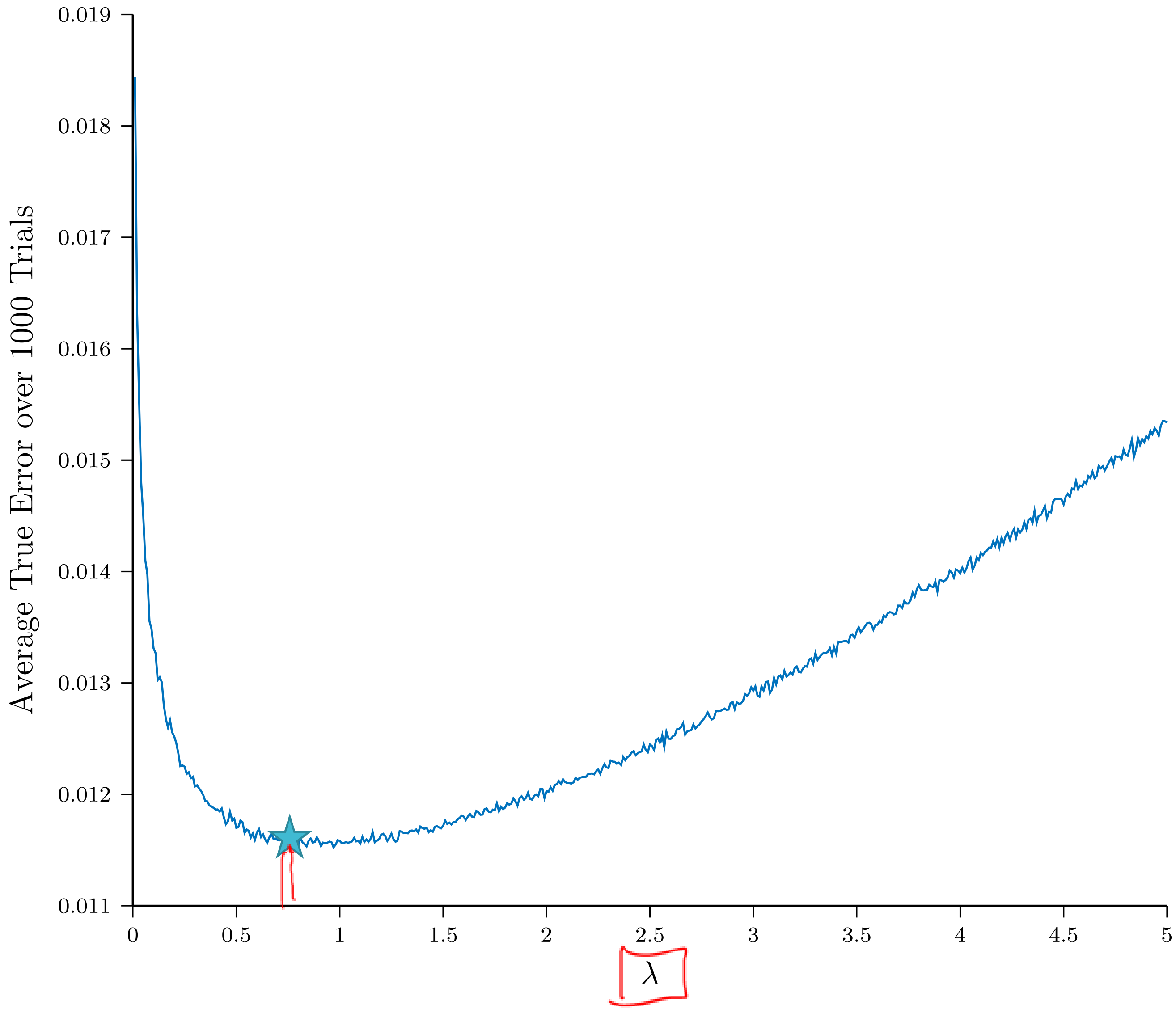
Setting λ



Setting λ

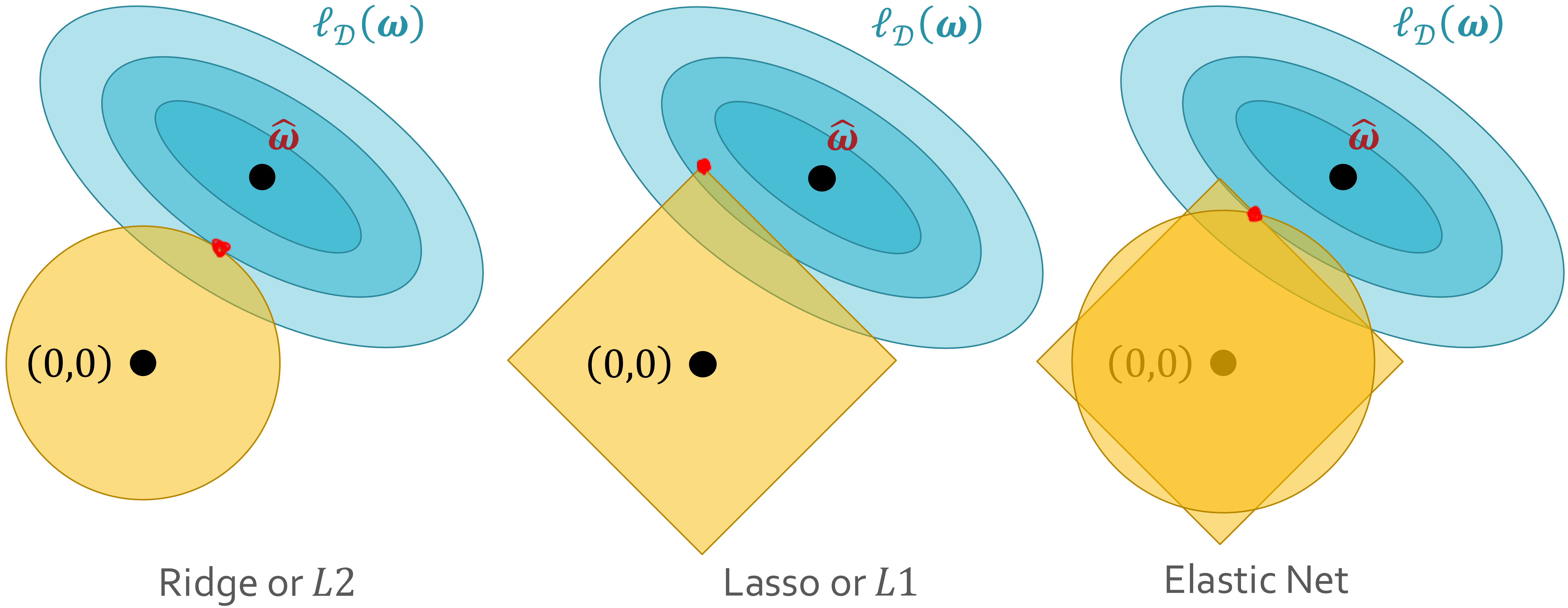


Setting λ



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
Elastic Net $L1 + L2$	$r(\boldsymbol{\omega}) = \lambda\lVert\boldsymbol{\omega}\rVert_1 + \lambda\lVert\boldsymbol{\omega}\rVert_2^2$	Encourages sparsity & shrinkage



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!

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

$$\vdots$$

$$= \boxed{(X^T X + \lambda_C I_{D+1})^{-1} X^T \boldsymbol{y}} \equiv \text{ridge reg solution}$$

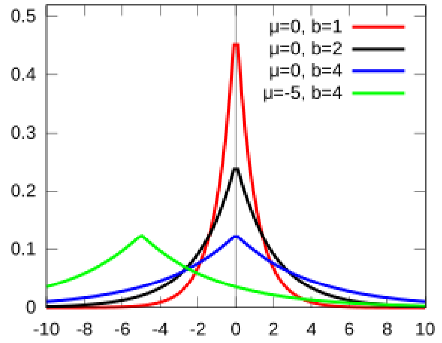
MAP for Linear Regression

- 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