10-301/601: Introduction to Machine Learning Lecture 10 – Regularization

Henry Chai & Matt Gormley & Hoda Heidari

19/2/23

Recall: Logistic Regression

- Model:

$p(y=1|x,\theta)$

$$p(y|\boldsymbol{x},\boldsymbol{\theta}) = \begin{cases} \sigma(\boldsymbol{\theta}^T\boldsymbol{x}) & \text{if } y = 1 \\ 1 - \sigma(\boldsymbol{\theta}^T\boldsymbol{x}) & \text{if } y = 0 \end{cases}$$

where $\sigma(z) = 1/1 + \exp(-z)$

- Derivatives

$$\frac{\partial J^{(i)}}{\partial \theta_m} = \frac{\partial}{\partial \theta_m}\left(-\log p\left(y^{(i)}\middle|\boldsymbol{x}^{(i)},\boldsymbol{\theta}\right)\right)$$

$$\vdots$$

$$= -\left(y^{(i)} - \sigma\left(\boldsymbol{\theta}^T\boldsymbol{x}^{(i)}\right)\right)x_m^{(i)}$$

- Optimization: use GD or SGD; logistic regression does not permit a closed form solution

- Objective: minimize the negative conditional log-likelihood

$$J(\theta) = \frac{1}{N}\sum_{i=1}^{N} -\log p\left(y^{(i)}\middle|\boldsymbol{x}^{(i)},\boldsymbol{\theta}\right)$$

- Gradients

$$\nabla J^{(i)}(\boldsymbol{\theta}) = -\left(y^{(i)} - \sigma\left(\boldsymbol{\theta}^T\boldsymbol{x}^{(i)}\right)\right)\boldsymbol{x}^{(i)}$$

$$\nabla J(\boldsymbol{\theta}) = \frac{1}{N}\sum_{i=1}^{N}\nabla J^{(i)}$$

- Predictions

$$\hat{y} = \underset{y\in\{0,1\}}{\operatorname{argmax}}\ p\left(y\middle|\boldsymbol{x}',\hat{\boldsymbol{\theta}}\right)$$

$$\vdots$$

$$= \text{“sign”}\left(\hat{\boldsymbol{\theta}}^T\boldsymbol{x}'\right)$$

Logistic Regression Decision Boundary

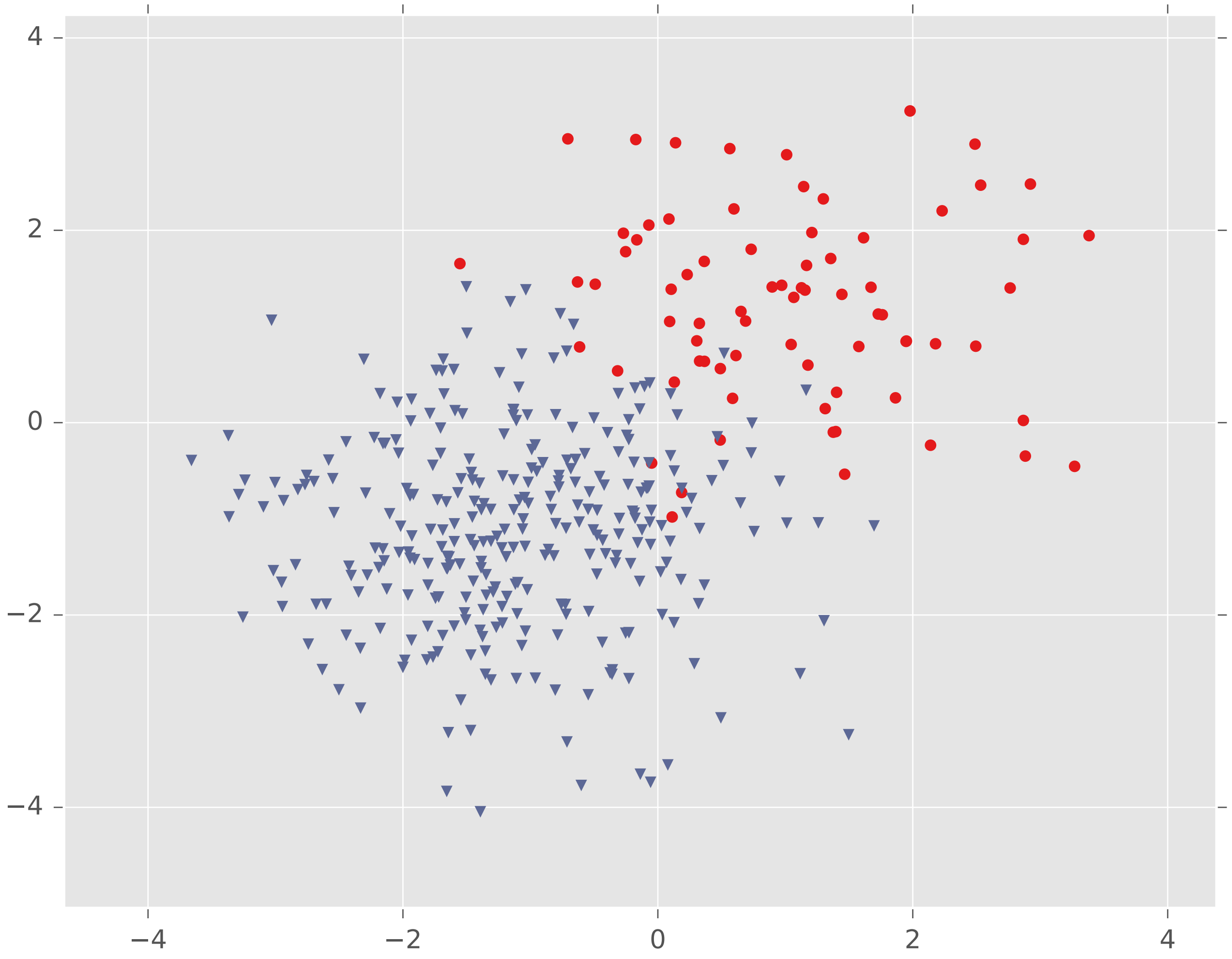


Figure courtesy of Matt Gormley

Logistic Regression Decision Boundary

Figure courtesy of Matt Gormley

Logistic Regression Decision Boundary

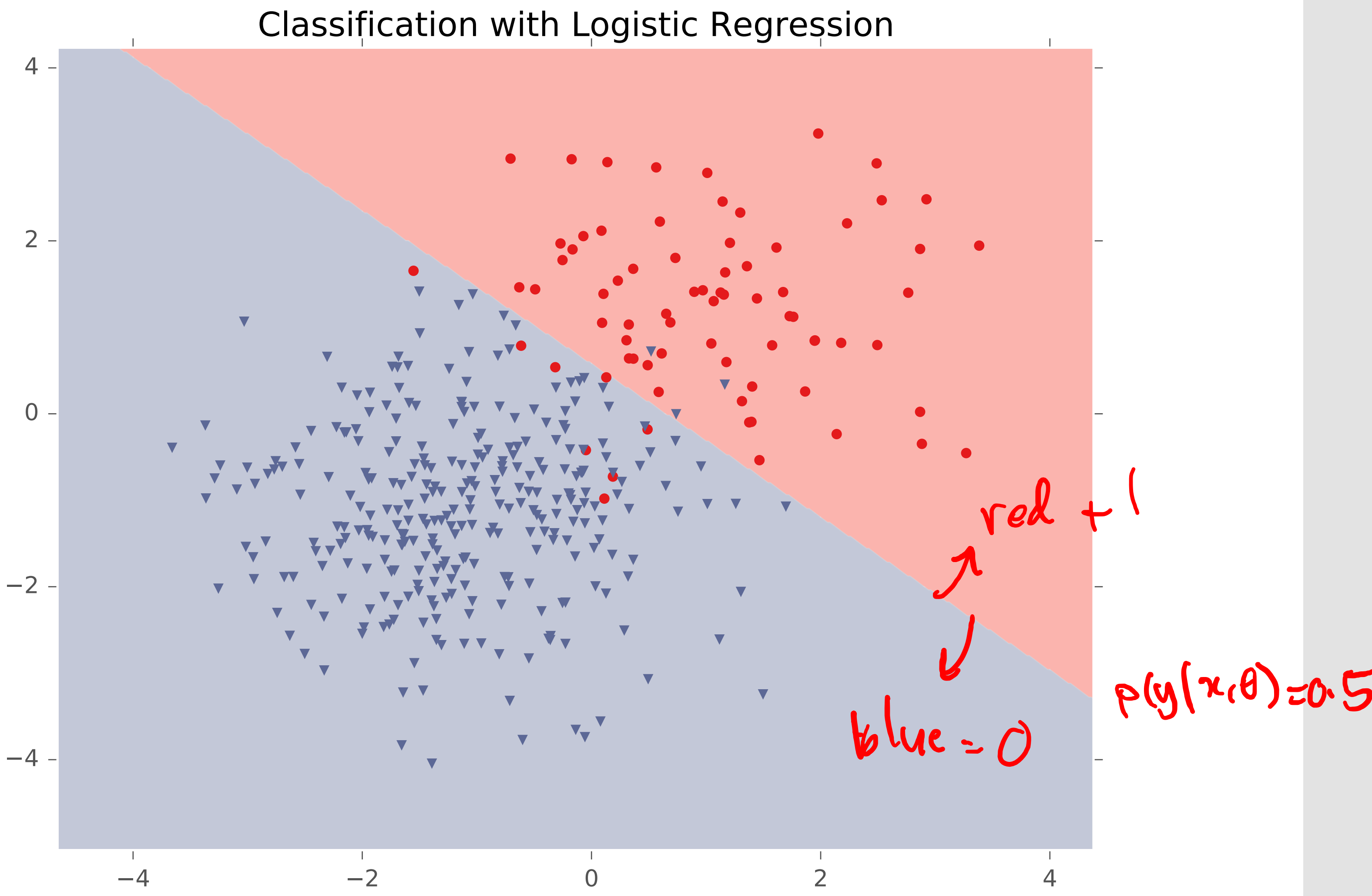


Figure courtesy of Matt Gormley

$p(y=1 \mid x, \theta) = p^*(y=1 \mid x)$

But is this the best that we could do, even if we knew p^*?

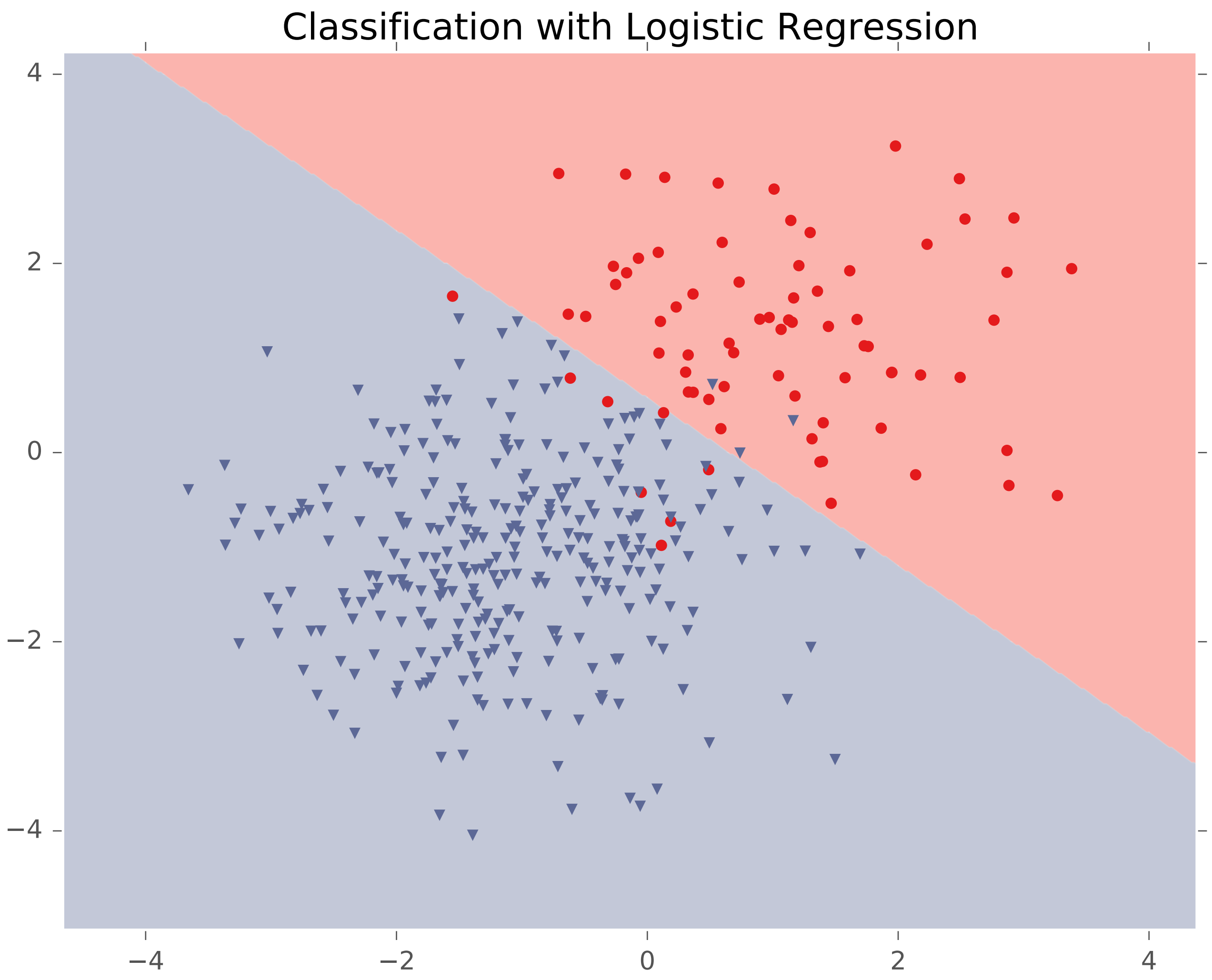


Figure courtesy of Matt Gormley

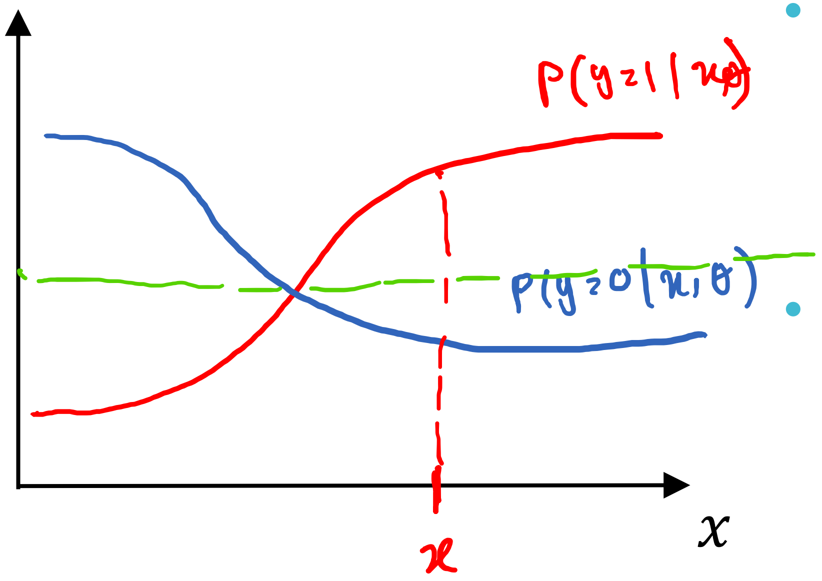
Bayes Optimal Classifier

- Suppose you knew $p^*(Y = 1|\boldsymbol{x})$ for all $\boldsymbol{x}$ and wanted to minimize the 0-1 loss

$$\ell(\hat{y}, y) = \mathbb{1}(\hat{y} \neq y)$$

- Then the optimal classifier in this setting, called the *Bayes optimal classifier*, is

$$\hat{y} = \begin{cases} 1 \text{ if } p^*(Y = 1|\boldsymbol{x}) \geq 0.5 \\ 0 \text{ otherwise} \end{cases}$$



- The *reducible error* of a classifier is the expected loss that could be eliminated if we knew p^*
- The *irreducible error* of a classifier is the expected loss even if we knew p^*

Stochastic Gradient Descent (SGD) for Logistic Regression

- Input: training dataset $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$ and step size γ

1. Initialize $\boldsymbol{\theta}^{(0)}$ to all zeros and set $t = 0$
2. While TERMINATION CRITERION is not satisfied
   a. For $i \in \text{shuffle}(\{1, \dots, N\})$
      i. Compute the pointwise gradient:
         $$\nabla J^{(i)}(\boldsymbol{\theta}^{(t)}) = -\left(y^{(i)} - \sigma(\boldsymbol{\theta}^T \boldsymbol{x}^{(i)})\right) \boldsymbol{x}^{(i)}$$
      ii. Update $\boldsymbol{\theta}$: $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \gamma \nabla J^{(i)}(\boldsymbol{\theta}^{(t)})$
      iii. Increment t: $t \leftarrow t + 1$

- Output: $\boldsymbol{\theta}^{(t)}$

Stochastic Gradient Descent (SGD) for Logistic Regression

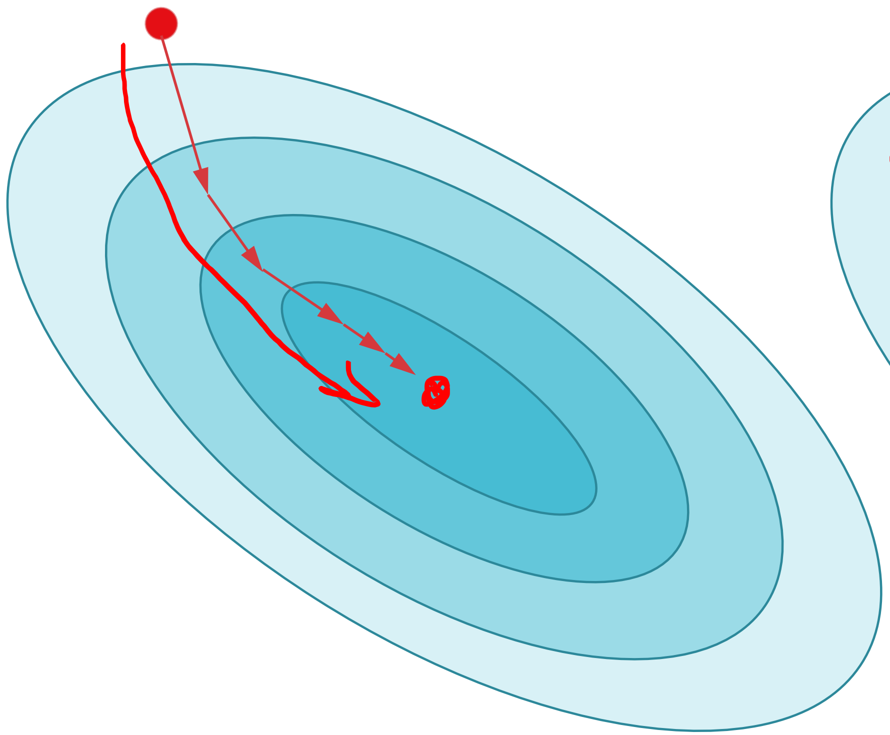
- Input: training dataset $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$ and step size γ

1. Initialize $\boldsymbol{\theta}^{(0)}$ to all zeros and set $t = 0$
2. While TERMINATION CRITERION is not satisfied
   a. For $i \in \text{shuffle}(\{1, \dots, N\})$
      i. Compute the pointwise gradient:
      $$\nabla J^{(i)}(\boldsymbol{\theta}^{(t)}) = \left(P\left(Y = 1 \middle| \boldsymbol{x}^{(i)}, \boldsymbol{\theta}^{(t)}\right) - y^{(i)}\right)\boldsymbol{x}^{(i)}$$
      ii. Update $\boldsymbol{\theta}$: $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \gamma \nabla J^{(i)}(\boldsymbol{\theta}^{(t)})$
      iii. Increment t: $t \leftarrow t + 1$

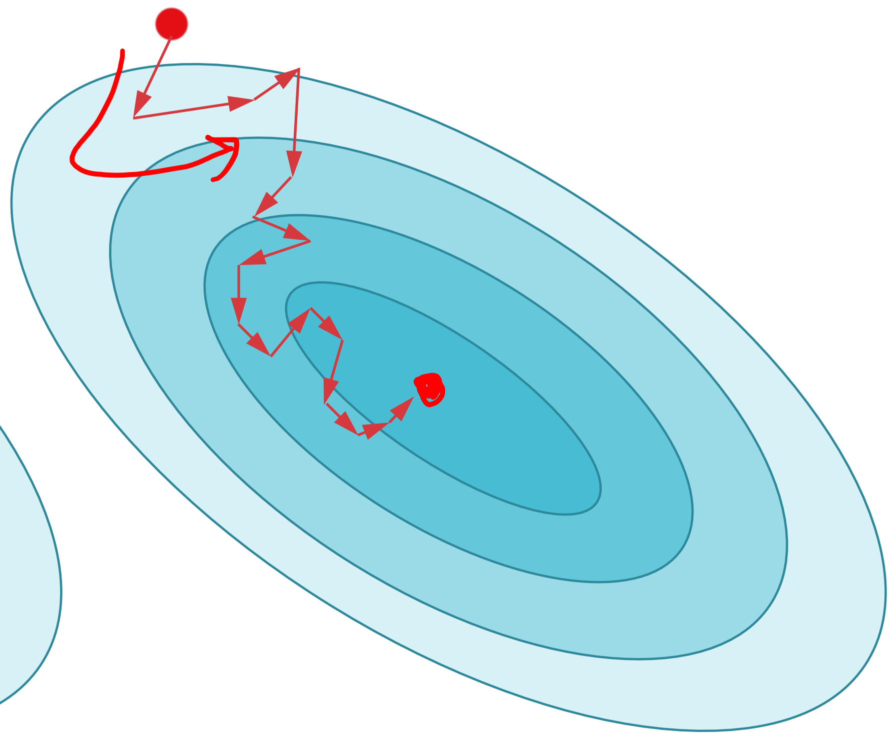
- Output: $\boldsymbol{\theta}^{(t)}$

$\sum_i$ 0.75 0

Stochastic Gradient Descent vs. Gradient Descent

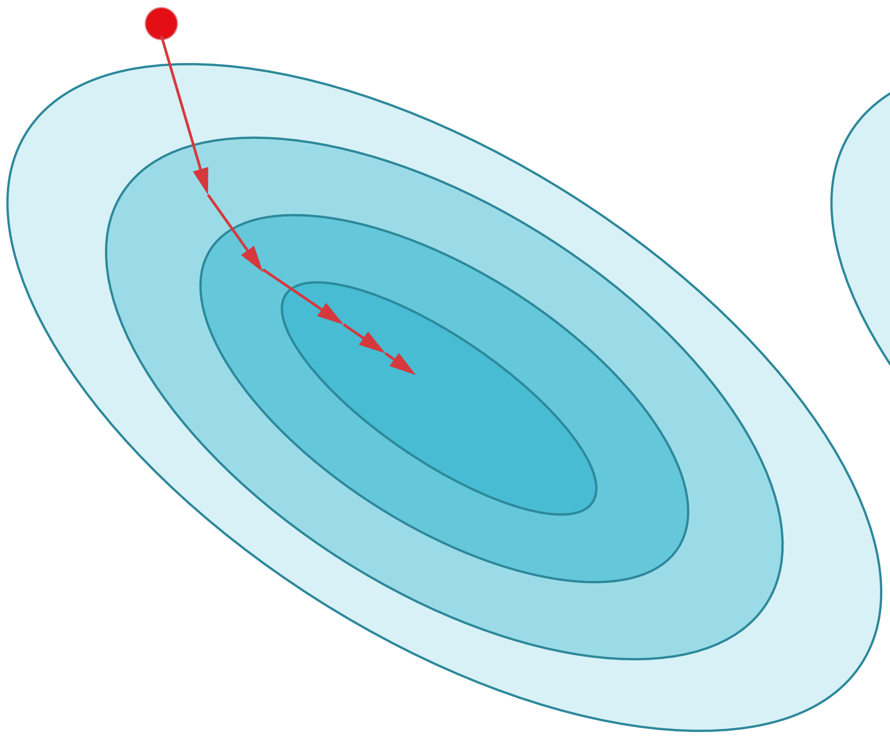


Gradient Descent

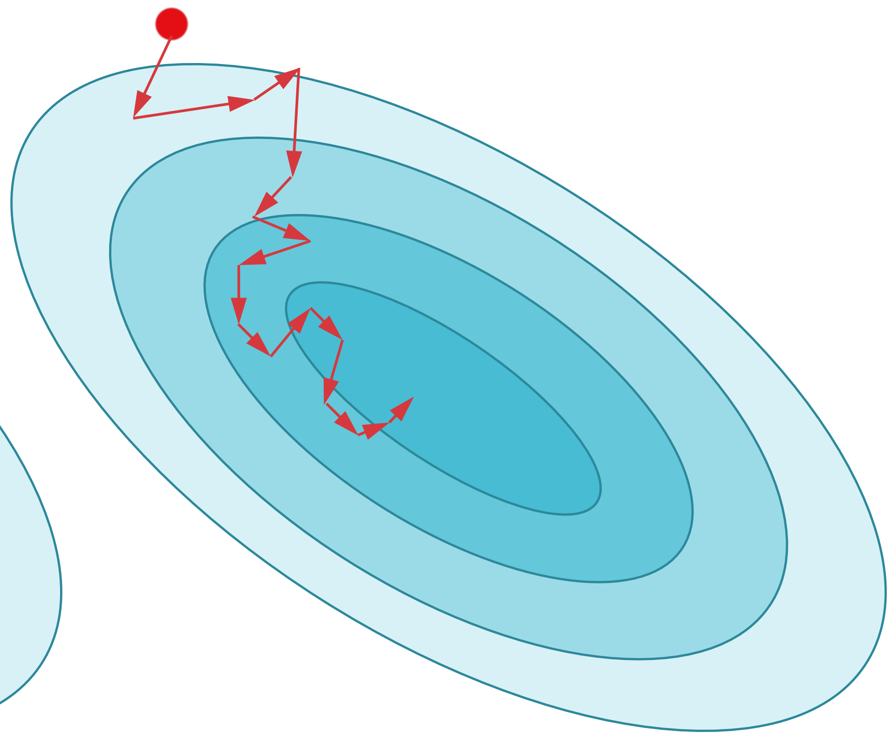


Stochastic Gradient Descent

Can we find some middle ground here?



Gradient Descent



Stochastic Gradient Descent

Mini-batch Stochastic Gradient Descent for Neural Networks

- Input: training dataset $\mathcal{D} = \{(\boldsymbol{x}^{(i)}, y^{(i)})\}_{i=1}^{N}$, step size γ, and batch size B

1. Initialize $\boldsymbol{\theta}^{(0)}$ to all zeros and set $t = 0$
2. While TERMINATION CRITERION is not satisfied
   a. Randomly sample B data points from $\mathcal{D}$, $\{(\boldsymbol{x}^{(b)}, y^{(b)})\}_{b=1}^{B}$
   b. Compute the gradient w.r.t. the sampled *batch*,

      $$\nabla J^{(B)}(\boldsymbol{\theta}^{(t)}) = \frac{1}{B}\sum_{b=1}^{B}\left(P\left(Y = 1 \middle| \boldsymbol{x}^{(b)}, \boldsymbol{\theta}^{(t)}\right) - y^{(b)}\right)\boldsymbol{x}^{(b)}$$

   c. Update $\boldsymbol{\theta}$: $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \gamma \nabla J^{(B)}(\boldsymbol{\theta}^{(t)})$
   d. Increment t: $t \leftarrow t + 1$

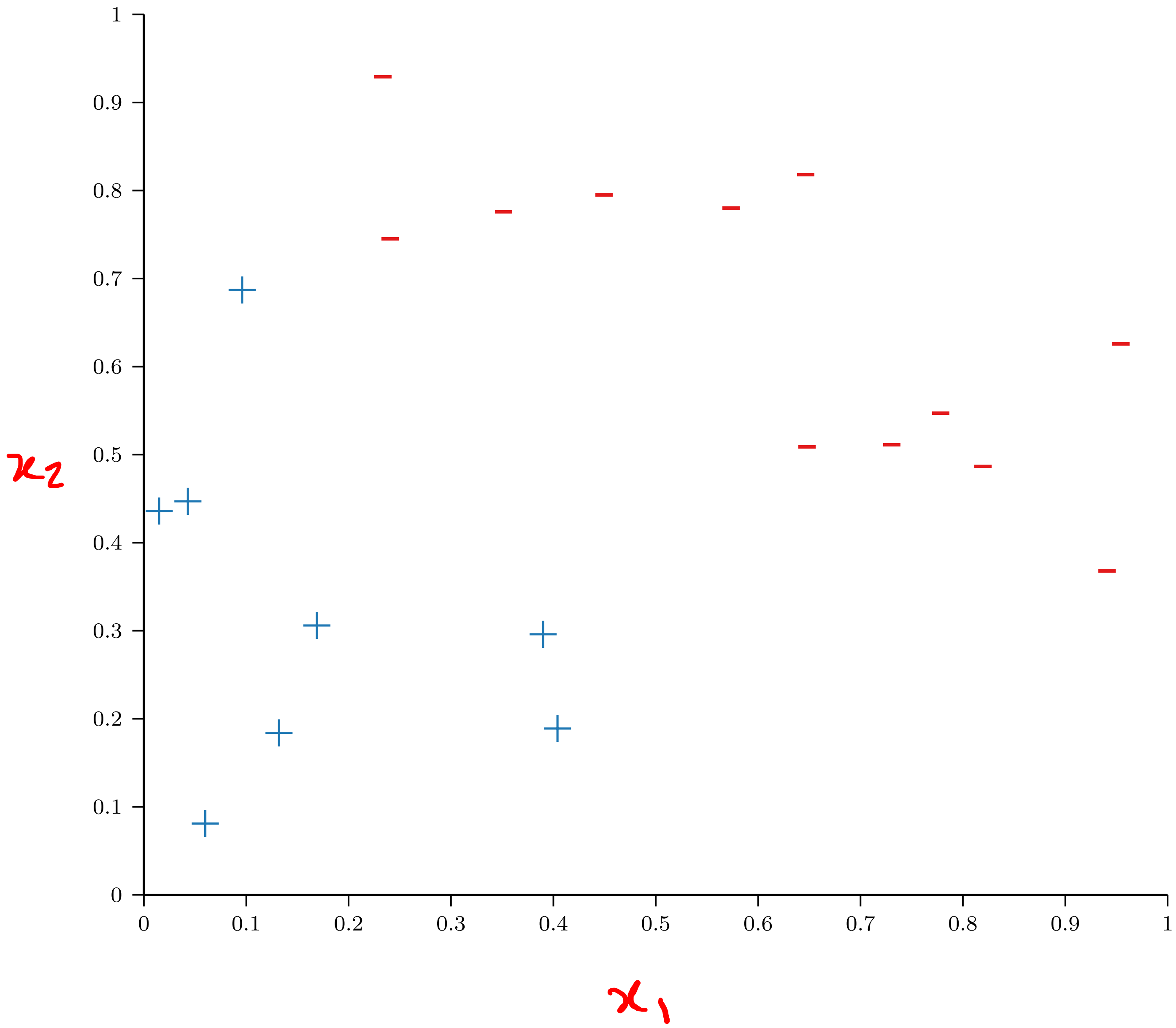
- Output: $\boldsymbol{\theta}^{(t)}$

Logistic Regression Learning Objectives

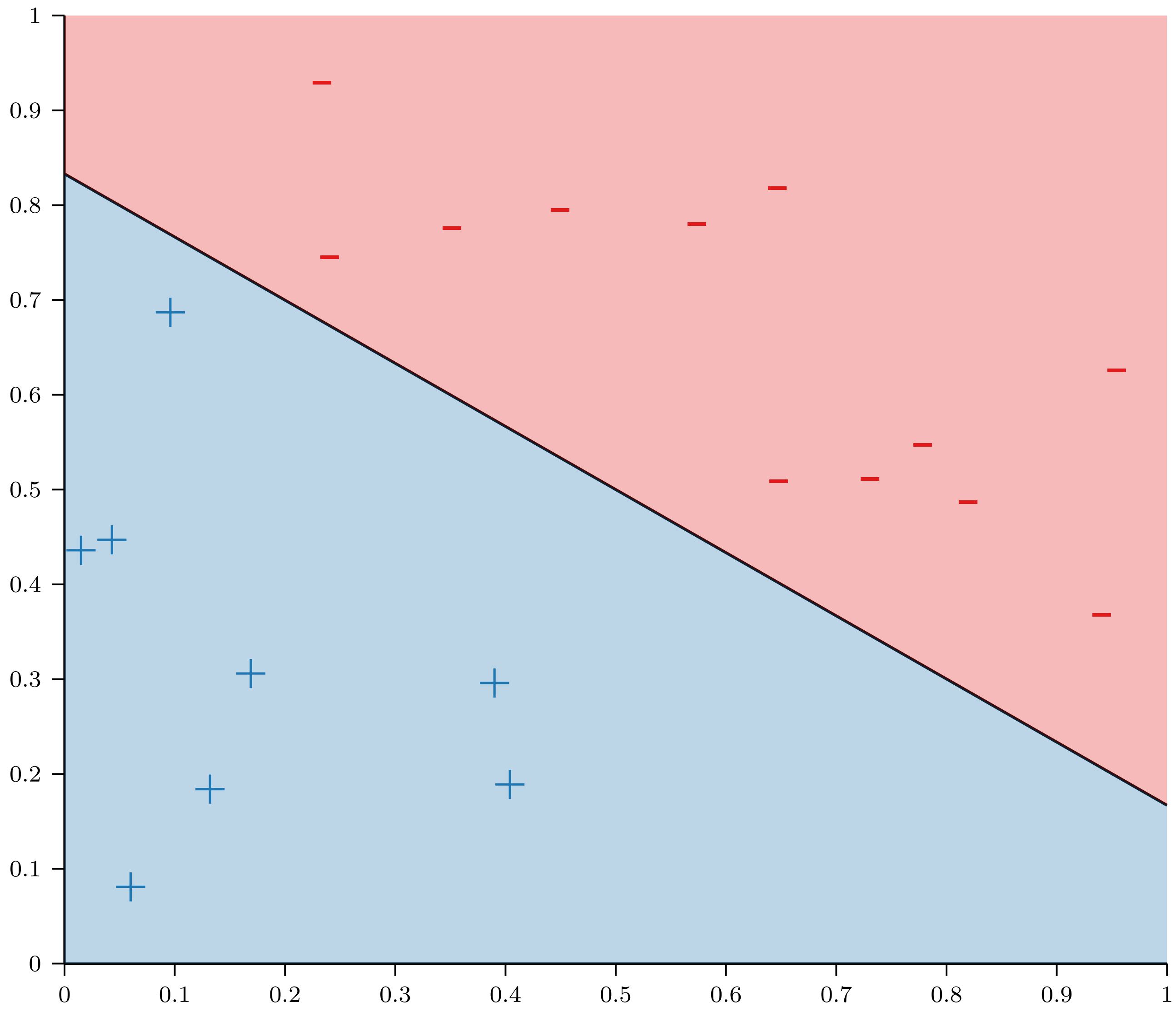
You should be able to…

- Apply the principle of maximum likelihood estimation (MLE) to learn the parameters of a probabilistic model
- Given a discriminative probabilistic model, derive the conditional log-likelihood, its gradient, and the corresponding Bayes Classifier
- Explain the practical reasons why we work with the log of the likelihood
- Implement logistic regression for binary classification
- Prove that the decision boundary of binary logistic regression is linear

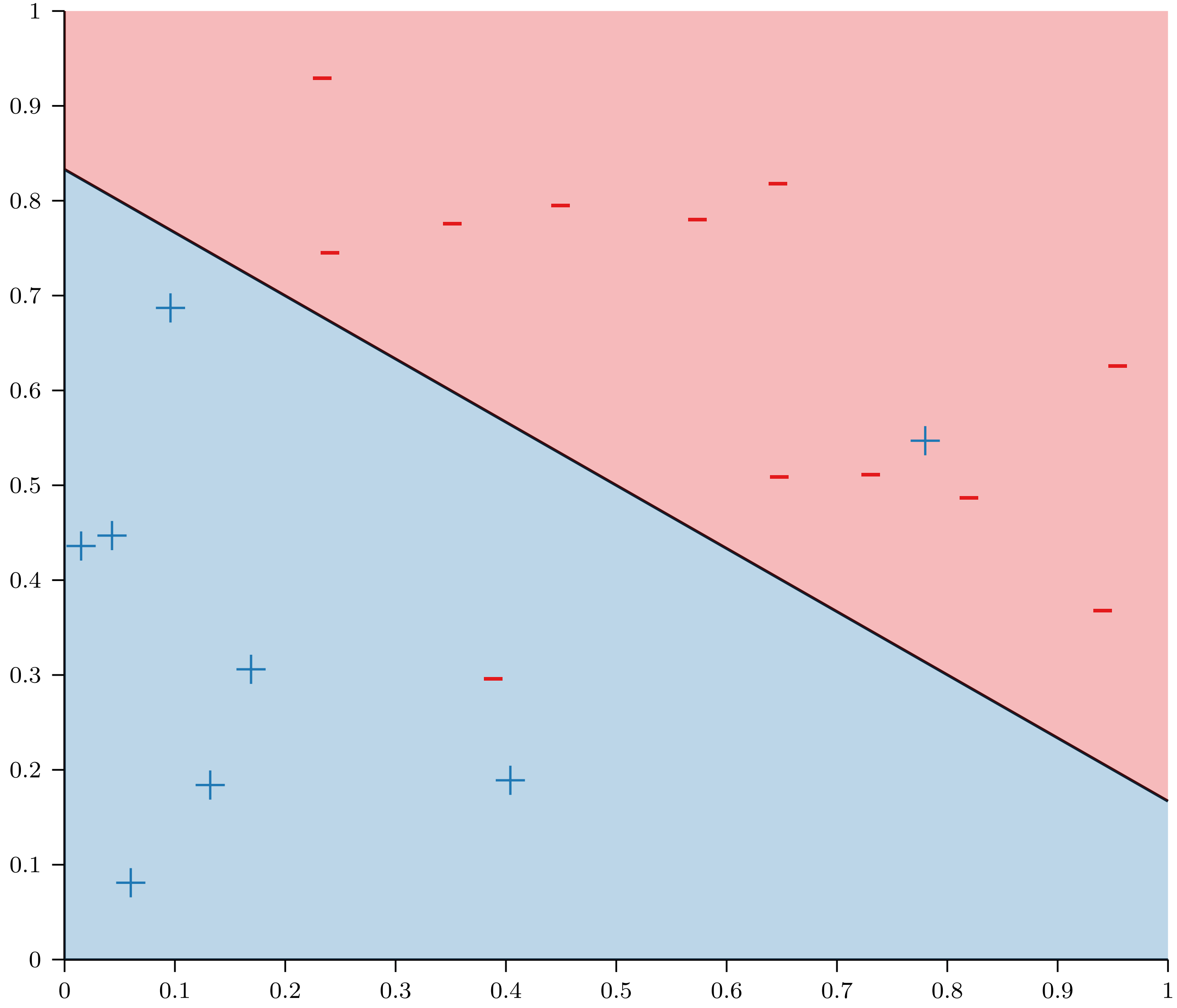
Linear Models



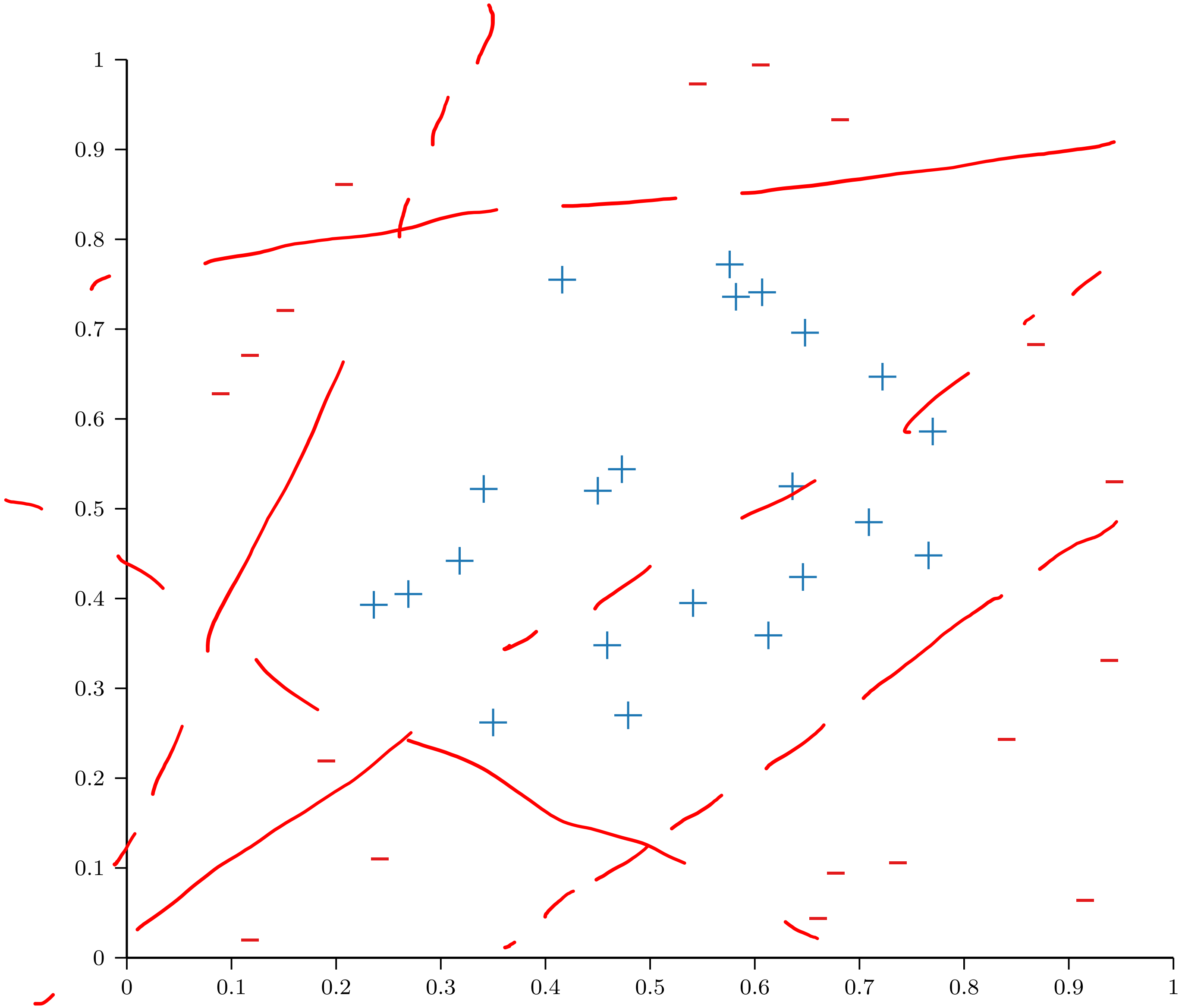
Linear Models



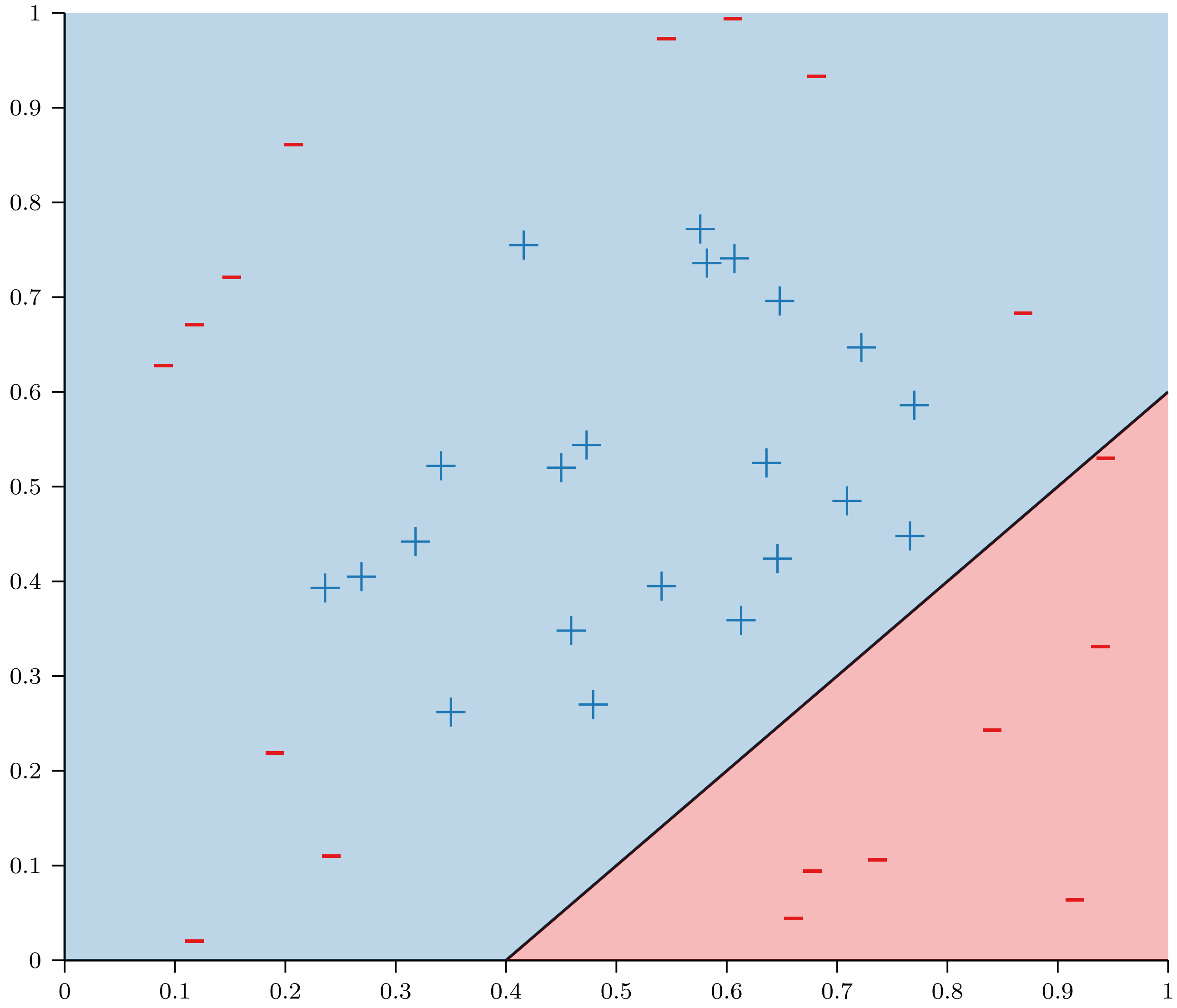
Linear Models



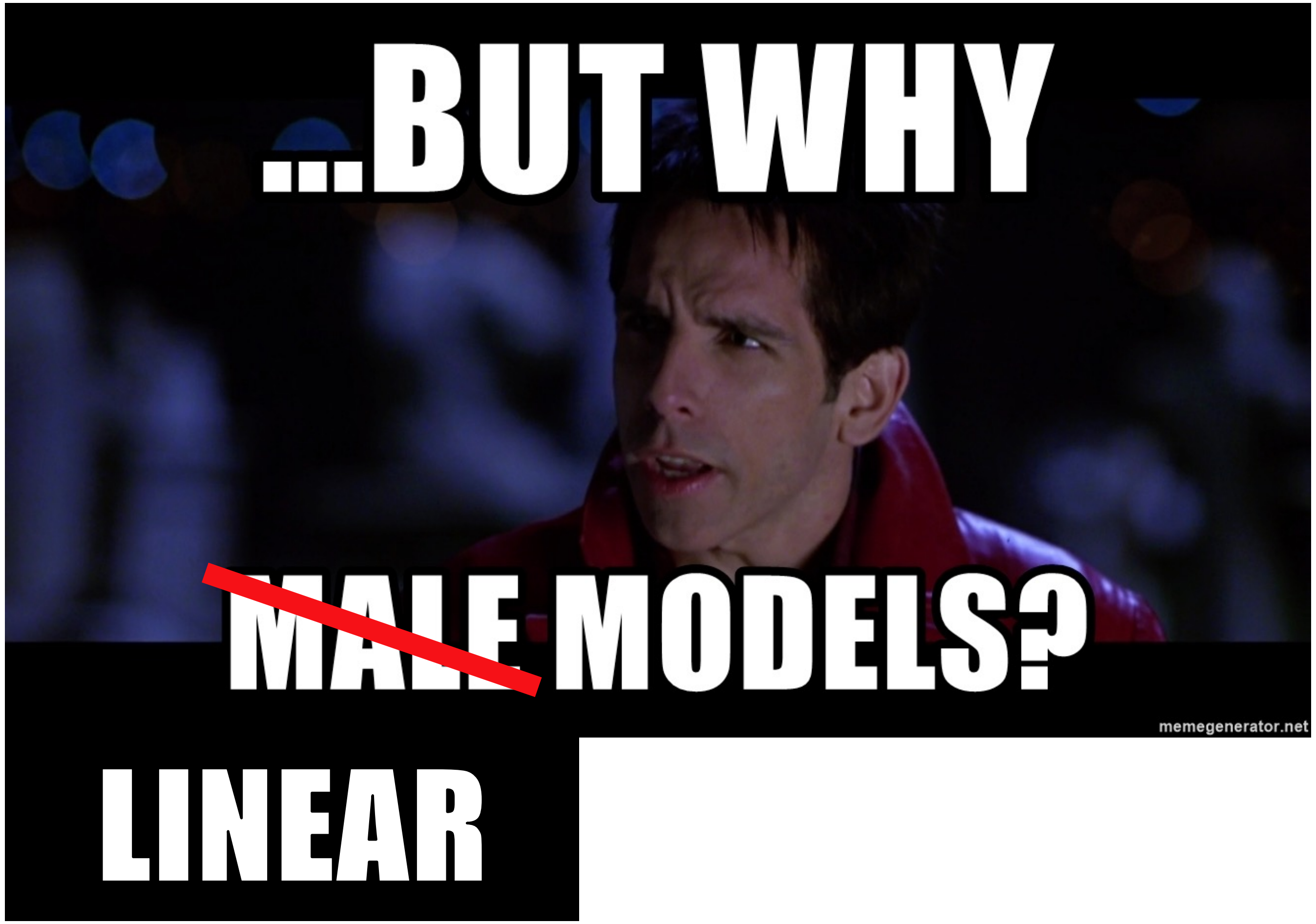
Linear Models?



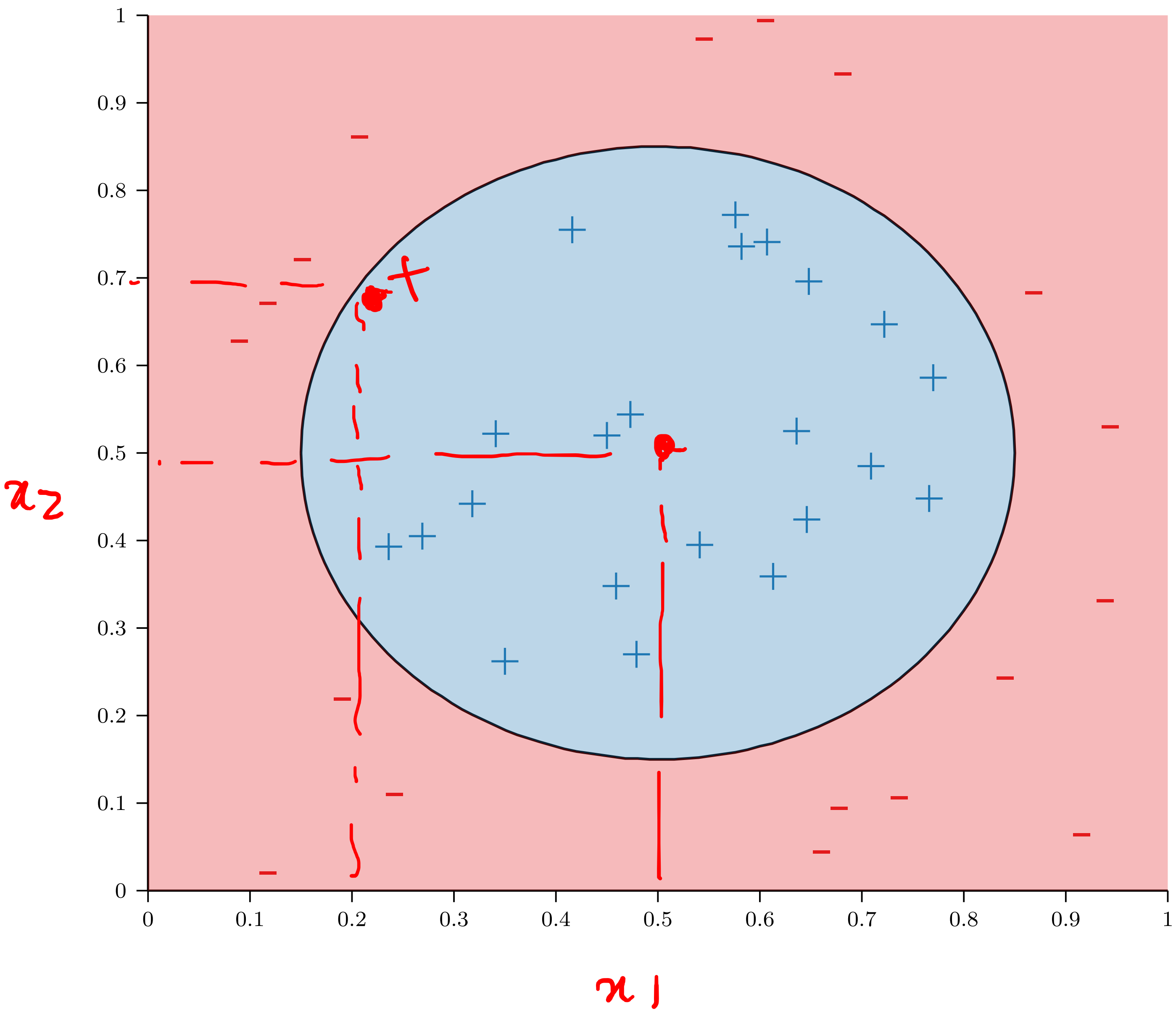
Linear Models?



Linear Models?



Nonlinear Models



Feature Transforms

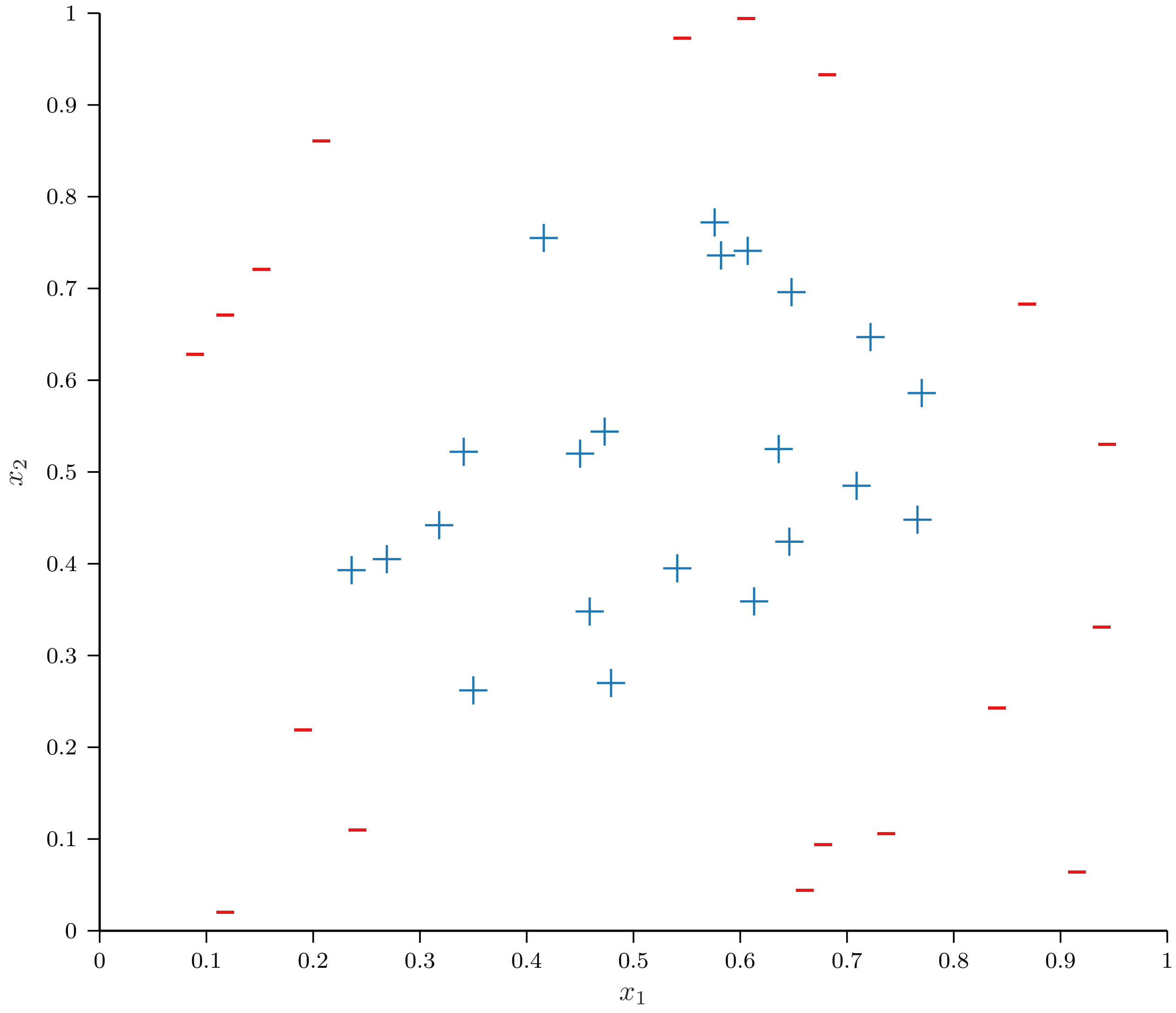
- Given D-dimensional inputs $\boldsymbol{x} = [x_1, \dots, x_D]$, first compute some transformation of our input, e.g.,

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

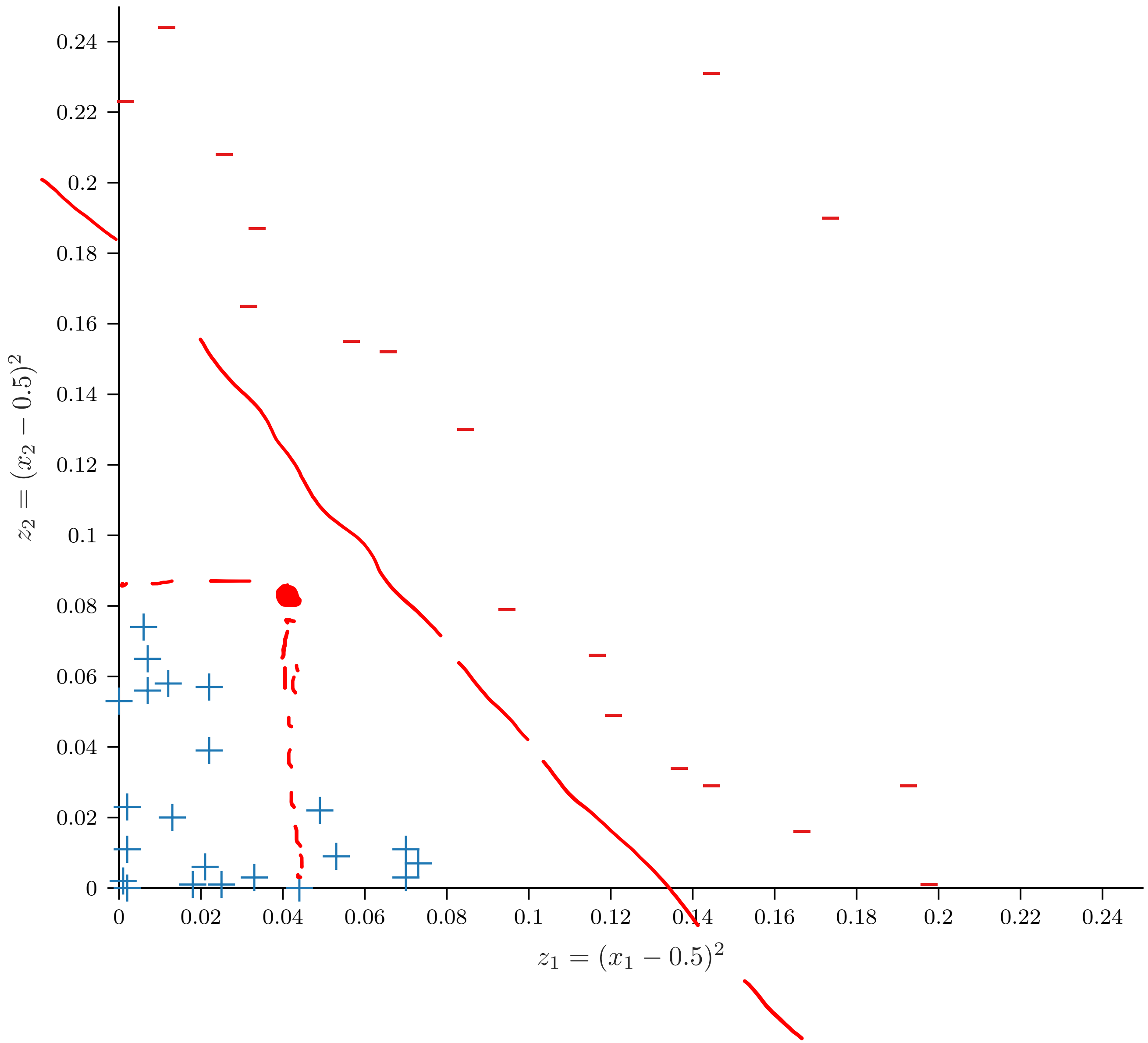
$$[x_1 = 0.7, x_2 = 0.2, y = +1] \rightarrow [z_1 = 0.04, z_2 = 0.09, y = +1]$$

$$D \longrightarrow D'$$

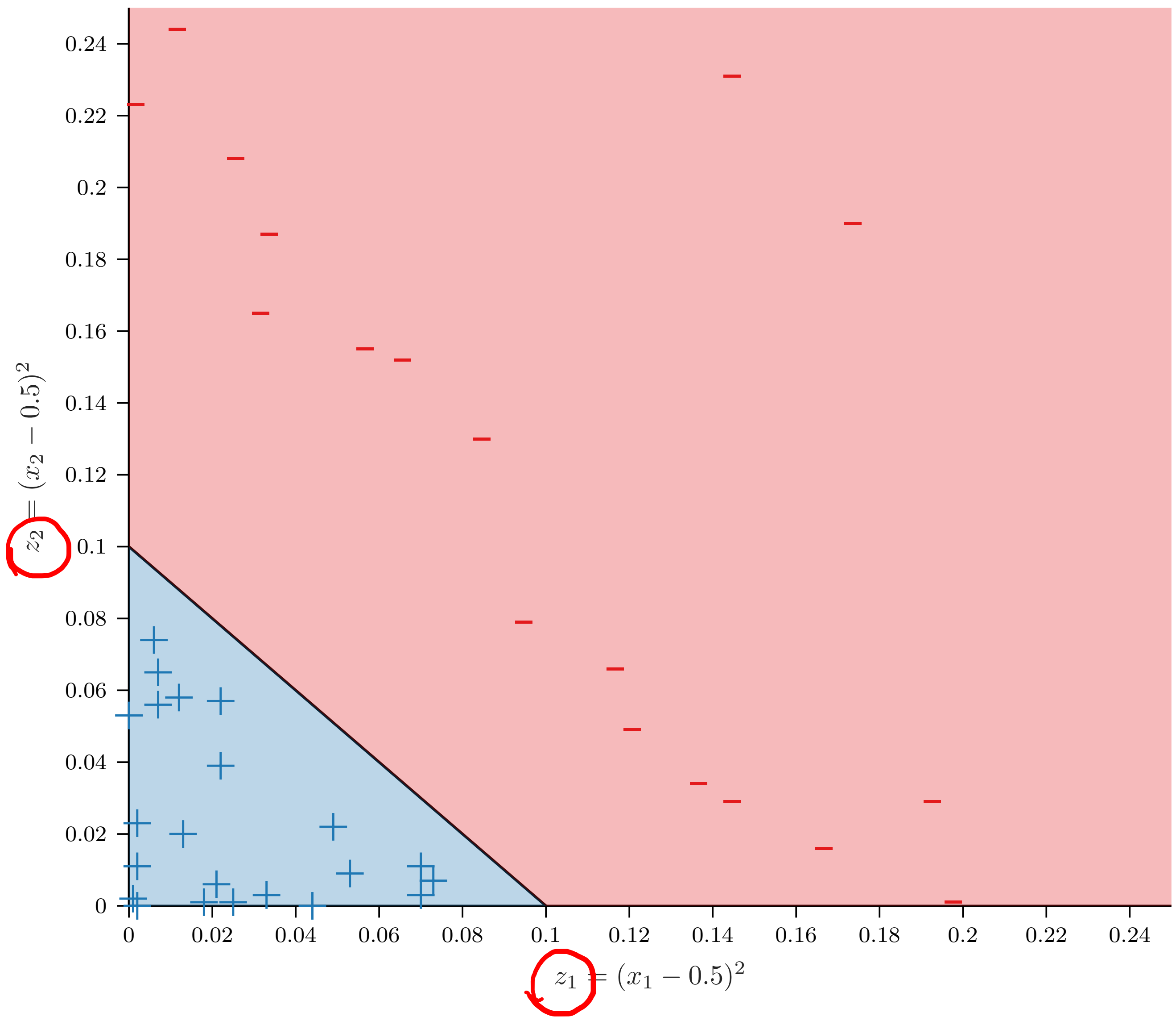
Nonlinear Models



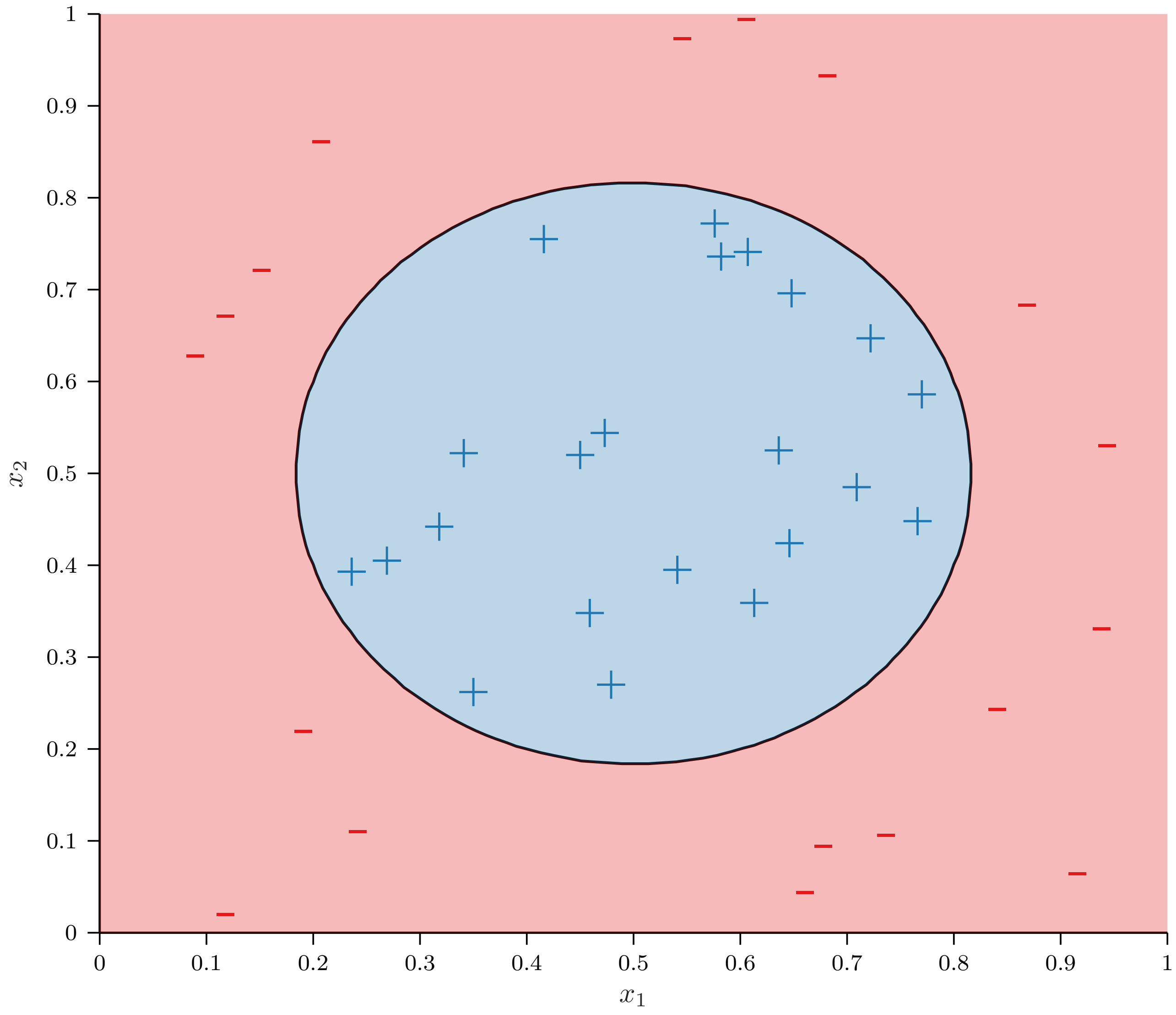
Nonlinear Models



Nonlinear Models



Nonlinear Models



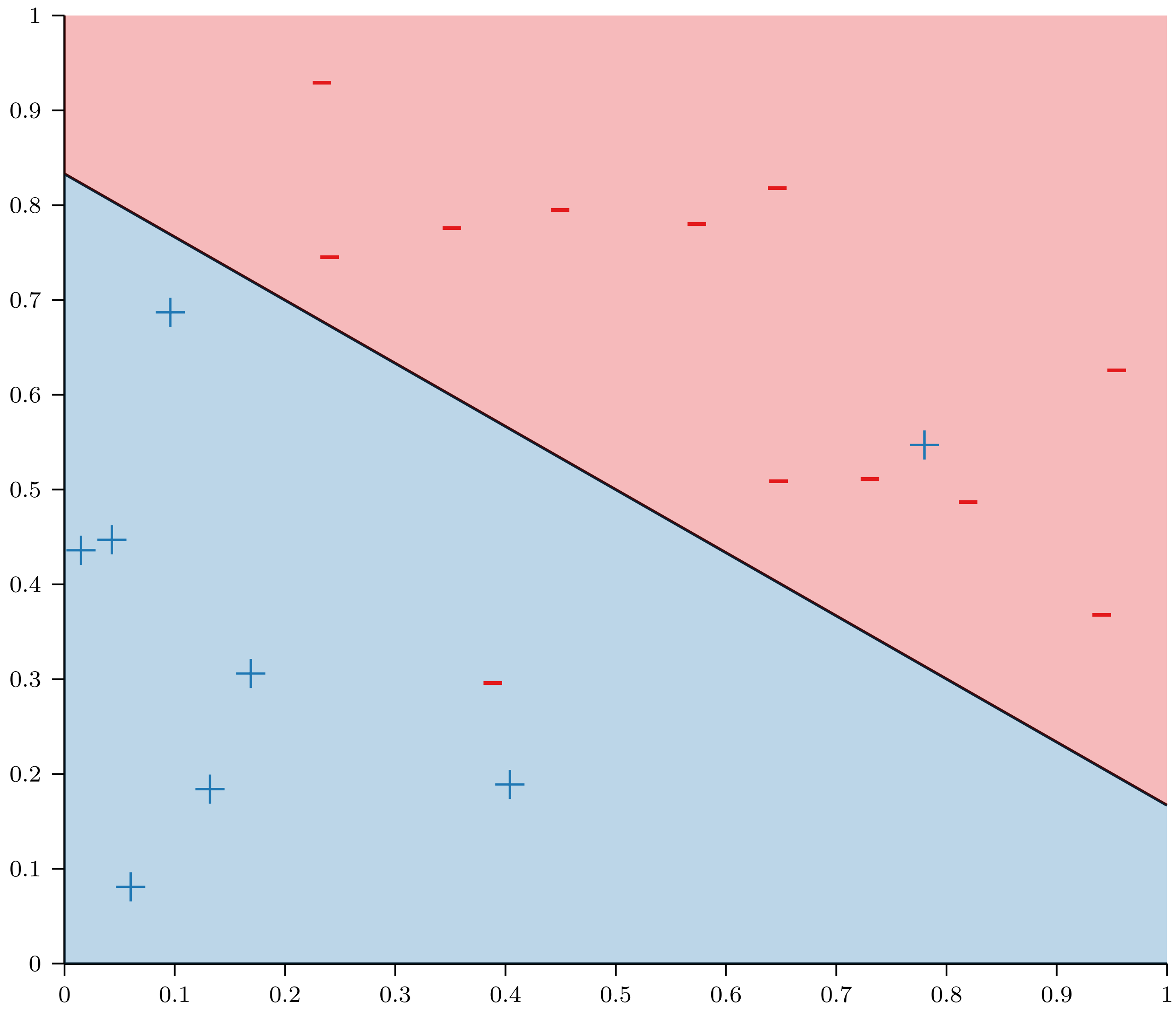
General Q^{th}-order Transforms

#feature [x1, x2]

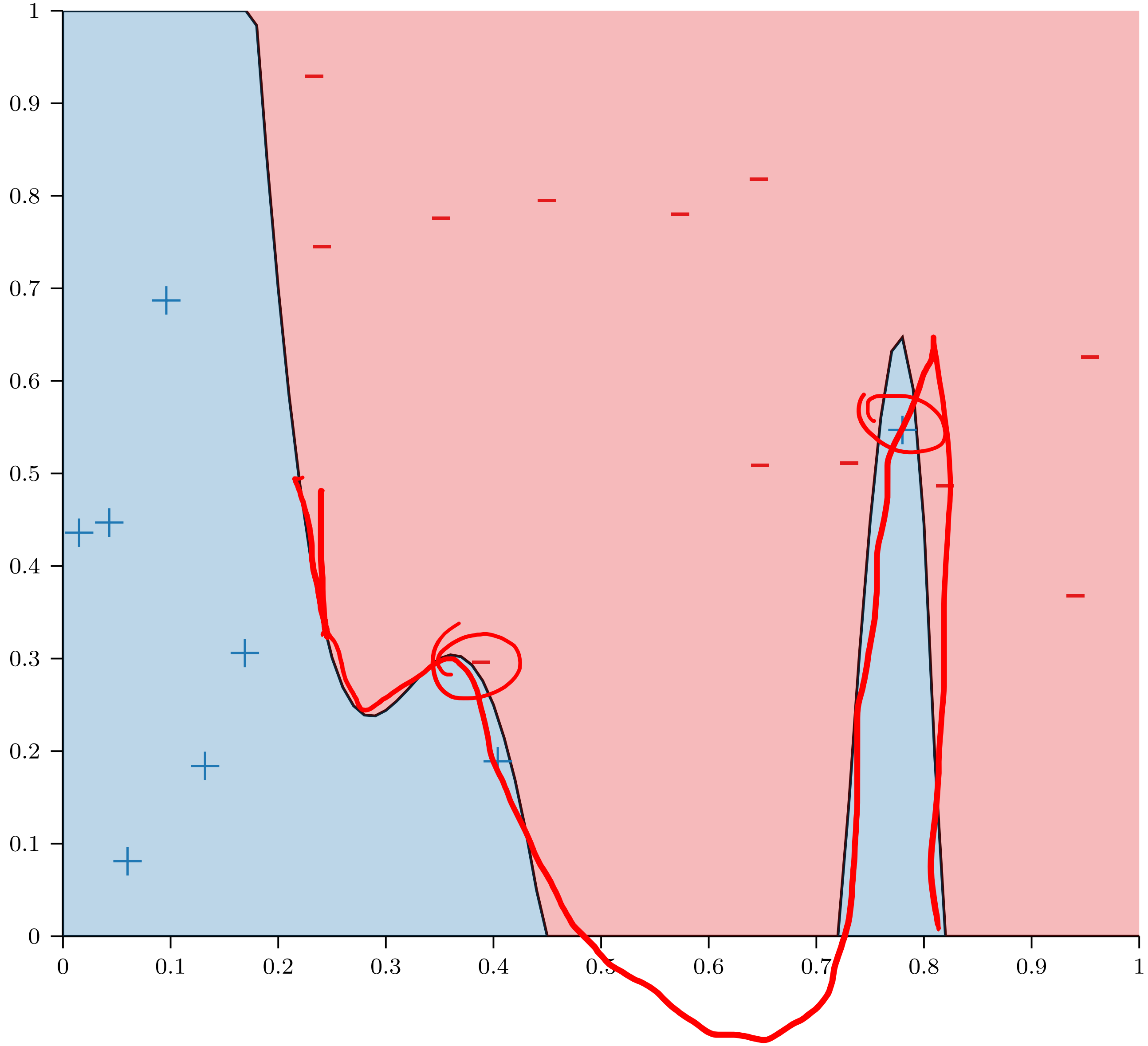
Q

- $\phi_{2,2}([x_1, x_2]) = [x_1, x_2, x_1^2, x_1 x_2, x_2^2]$
- $\phi_{2,3}([x_1, x_2]) = [x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3]$
- $\phi_{2,4}([x_1, x_2]) =$
  $[x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3, x_1^4, x_1^3 x_2, x_1^2 x_2^2, x_1 x_2^3, x_2^4]$
- $\phi_{2,Q}$ maps a 2-dimensional input to a $\frac{Q(Q+3)}{2}$ -dimensional 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:

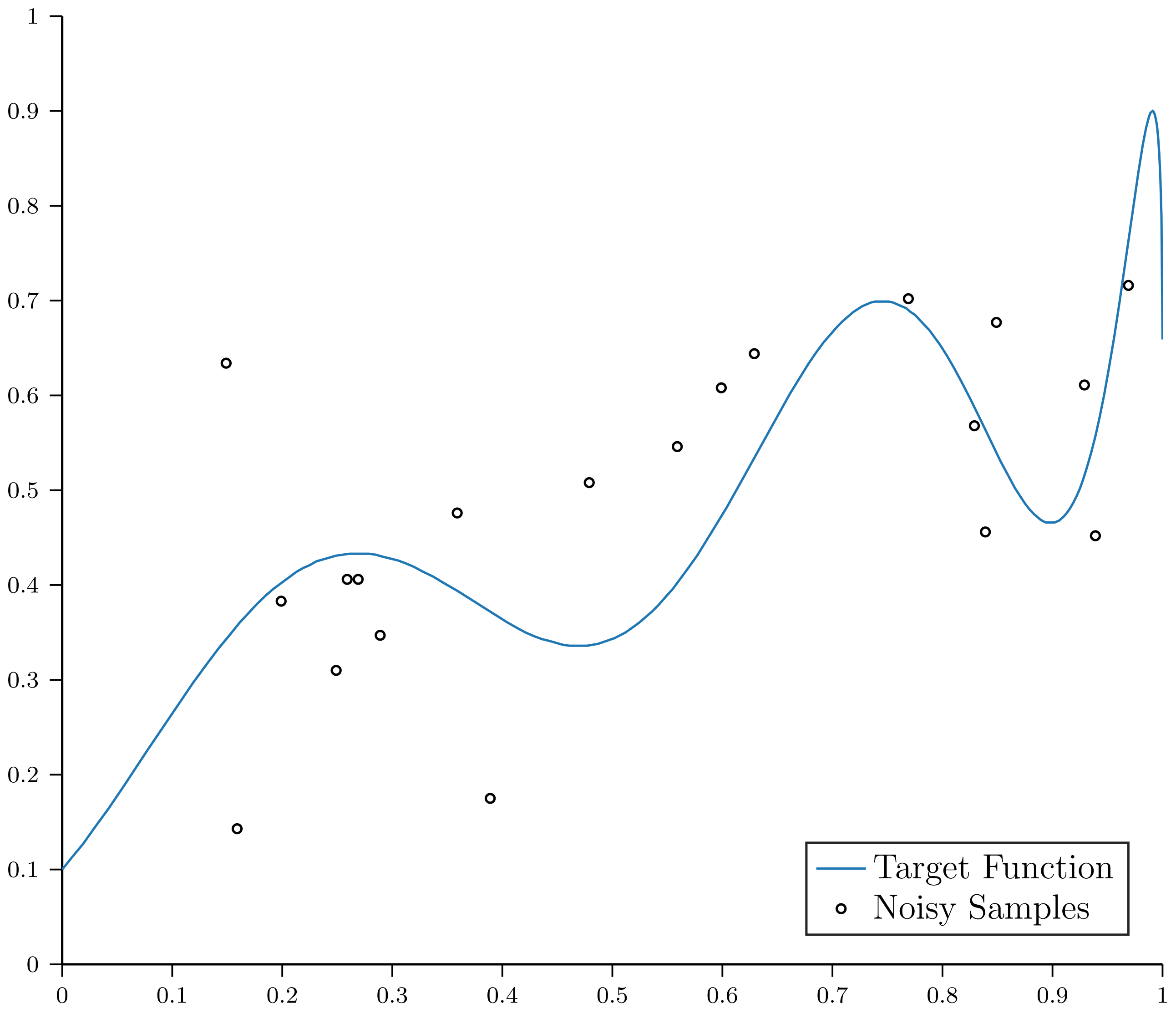
$$y = \sum_{d=0}^{10} a_d x^d + \epsilon \text{ where } \epsilon \sim N(0, \sigma^2)$$

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

$h^*(x) \in \mathcal{H}_{10}$

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



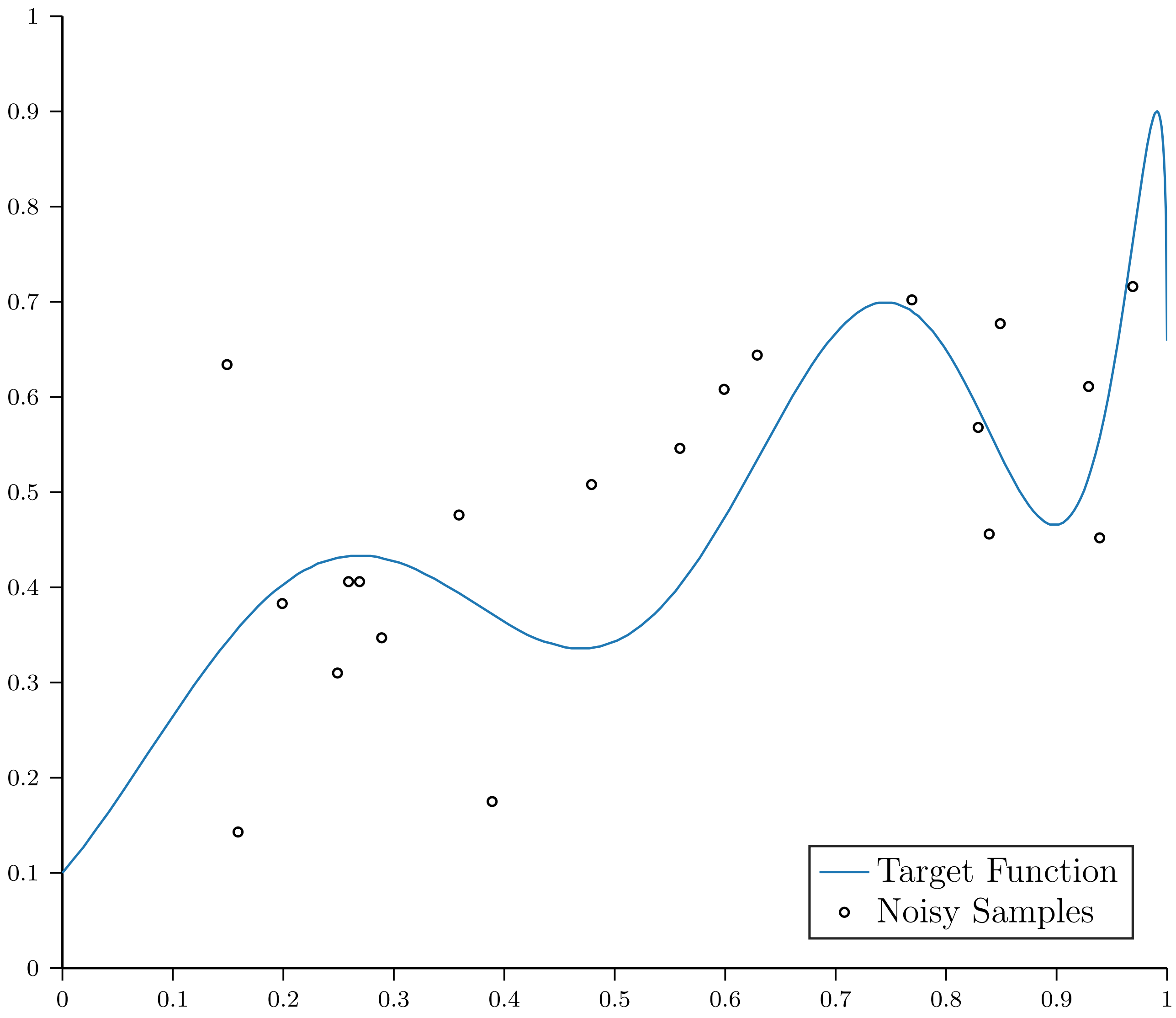
Poll Question 1

Which model do you think will have a lower true error?

A. $\mathcal{H}_2$

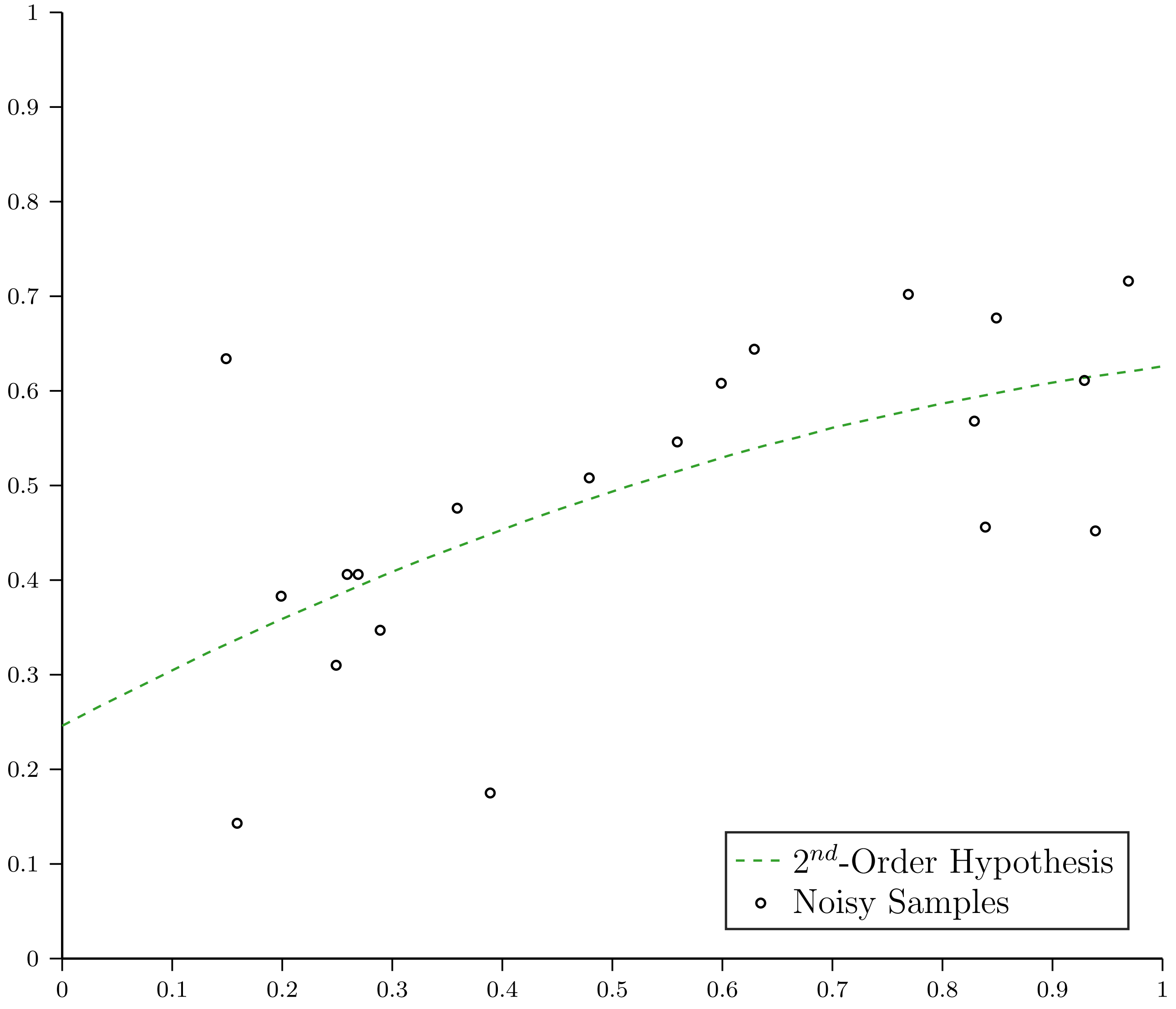
B. TOXIC

C. $\mathcal{H}_{10}$



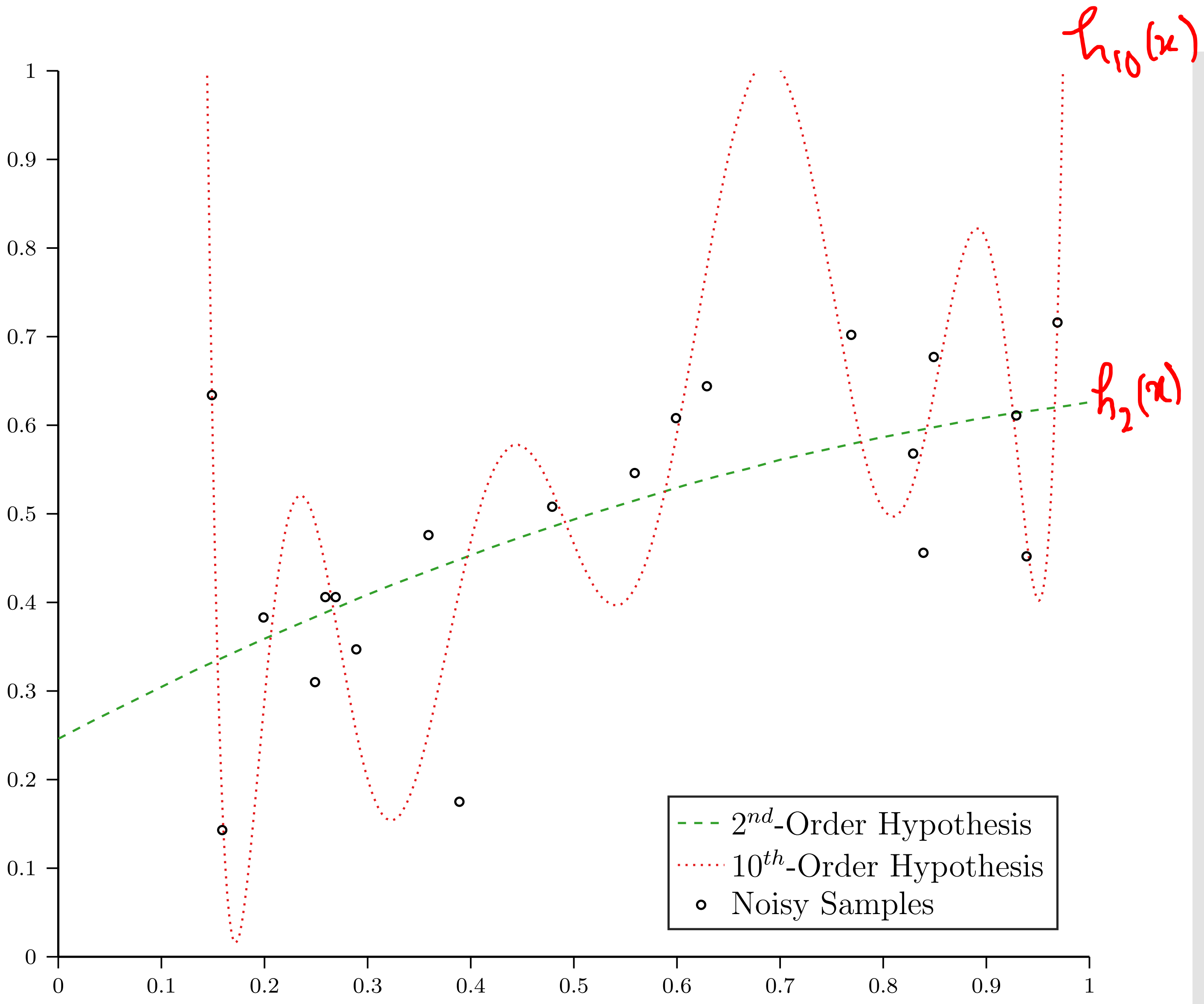
Noisy Targets

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



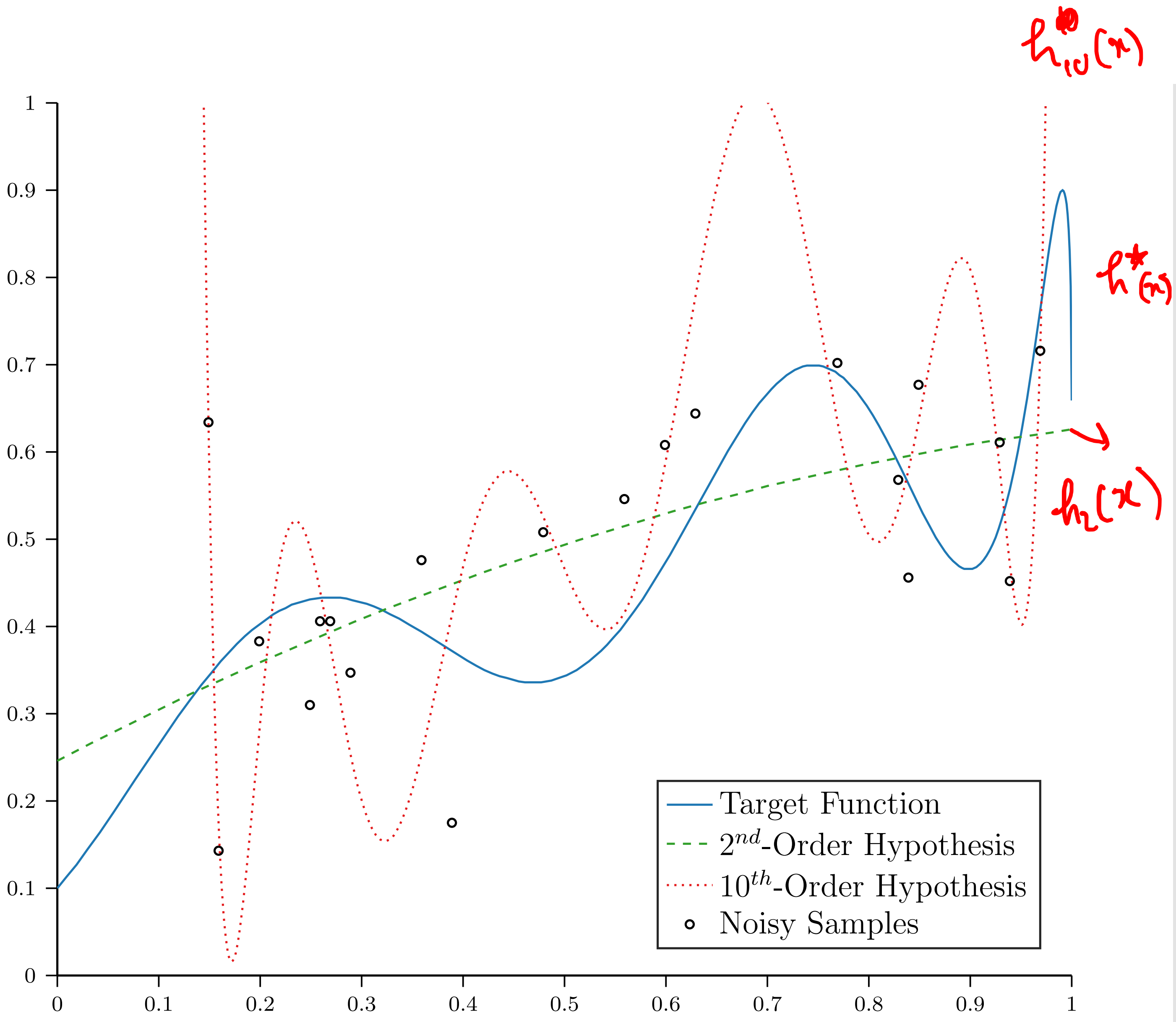
Noisy Targets

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



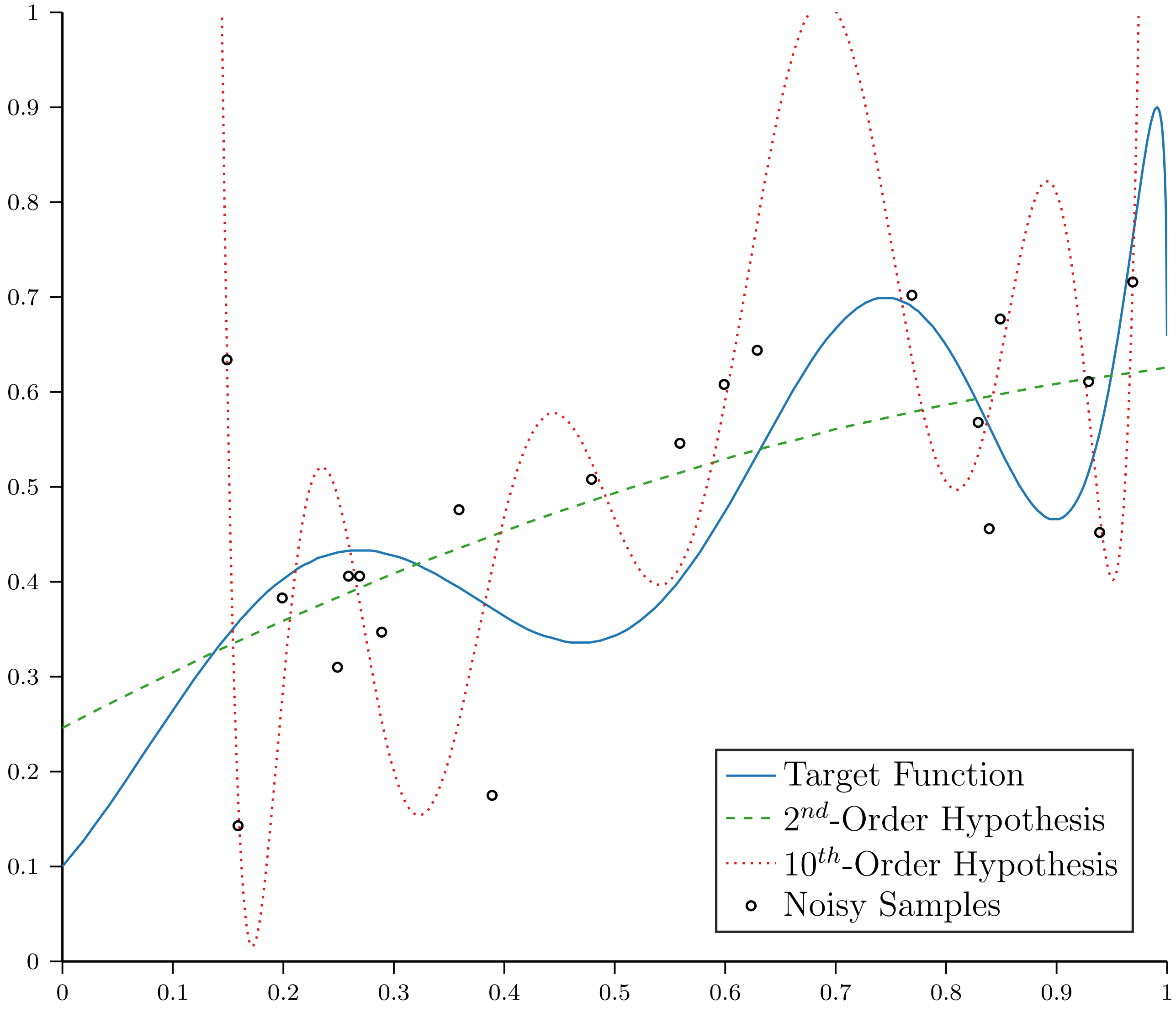
Noisy Targets

- 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

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

Poll Question 2

Now wich model do you think will have a lower true error?

A. TOXIC

B. $\mathcal{H}_2$

C. $\mathcal{H}_{10}$

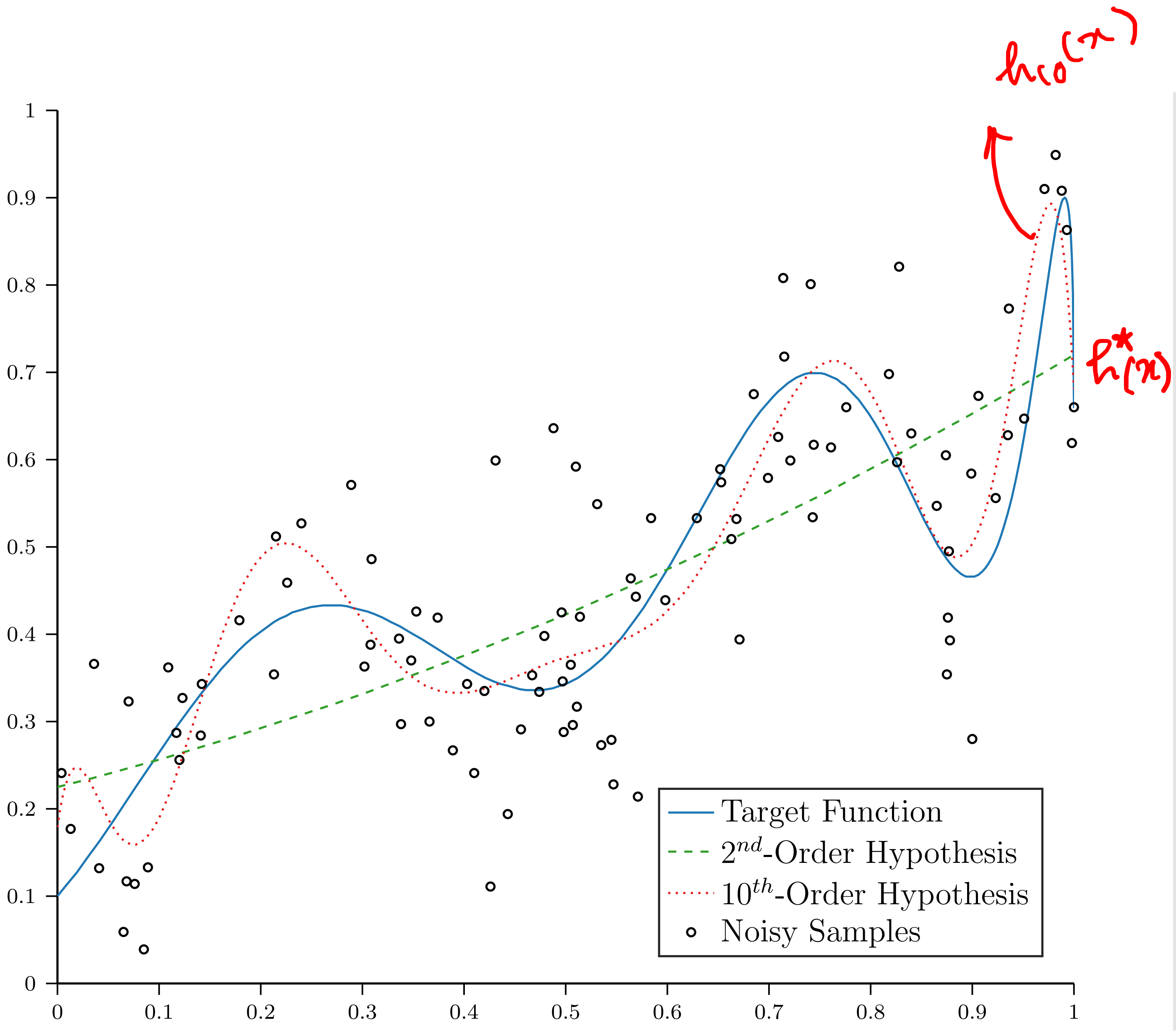
- $x \in \mathbb{R}$, $y \in \mathbb{R}$ and $N = 100$
- Targets are generated by a 10^{th}-order polynomial in x with additive Gaussian noise:

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

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

  $\theta_1, \theta_2, \ldots, \theta_{10}$

- Given $\mathrm{X} = \begin{bmatrix} 1 & \phi_{1,10}(x^{(1)}) \\ 1 & \phi_{1,10}(x^{(2)}) \\ \vdots & \vdots \\ 1 & \phi_{1,10}(x^{(N)}) \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$ find

  $\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \theta_9, \theta_{10}]$
  that minimizes

  $$\min_{\theta \in \mathbb{R}^{11}} \text{MSE} = (X\boldsymbol{\theta} - \boldsymbol{y})^T (X\boldsymbol{\theta} - \boldsymbol{y})$$

- Subject to

  $$\theta_3 = \theta_4 = \theta_5 = \theta_6 = \theta_7 = \theta_8 = \theta_9 = \theta_{10} = 0$$

Hard Constraints

- $\mathcal{H}_{10}$ = 10^{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 $\mathrm{X} = \begin{bmatrix} 1 & \phi_{1,10}(x^{(1)}) \\ 1 & \phi_{1,10}(x^{(2)}) \\ \vdots & \vdots \\ 1 & \phi_{1,10}(x^{(N)}) \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$ find

  $\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \theta_9, \theta_{10}]$
  that minimizes

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

- Subject to

$$\theta_3 = \theta_4 = \theta_5 = \theta_6 = \theta_7 = \theta_8 = \theta_9 = \theta_{10} = 0$$

Hard Constraints

- $\mathcal{H}_{10}$ = 10^{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 $\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2, \theta_3, \theta_4, \theta_5, \theta_6, \theta_7, \theta_8, \theta_9, \theta_{10}]$ that minimizes

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

- Subject to nothing!

Hard Constraints

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

  $\boldsymbol{\theta} = [\theta_0, \theta_1, \theta_2]$
  that minimizes

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

- Subject to nothing!

Soft Constraints

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

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

  find $\boldsymbol{\theta}$ that minimizes

$$\text{min MSE} = (X\boldsymbol{\theta} - \boldsymbol{y})^T (X\boldsymbol{\theta} - \boldsymbol{y})$$

- Subject to:

constraint

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

Soft Constraints

MSE

constrained optimization (*)

$$\begin{cases} \text{minimize } \ell_D(\boldsymbol{\theta}) = (X\boldsymbol{\theta} - \boldsymbol{y})^T(X\boldsymbol{\theta} - \boldsymbol{y}) \\ \text{subject to } \boldsymbol{\theta}^T\boldsymbol{\theta} \le C \end{cases}$$

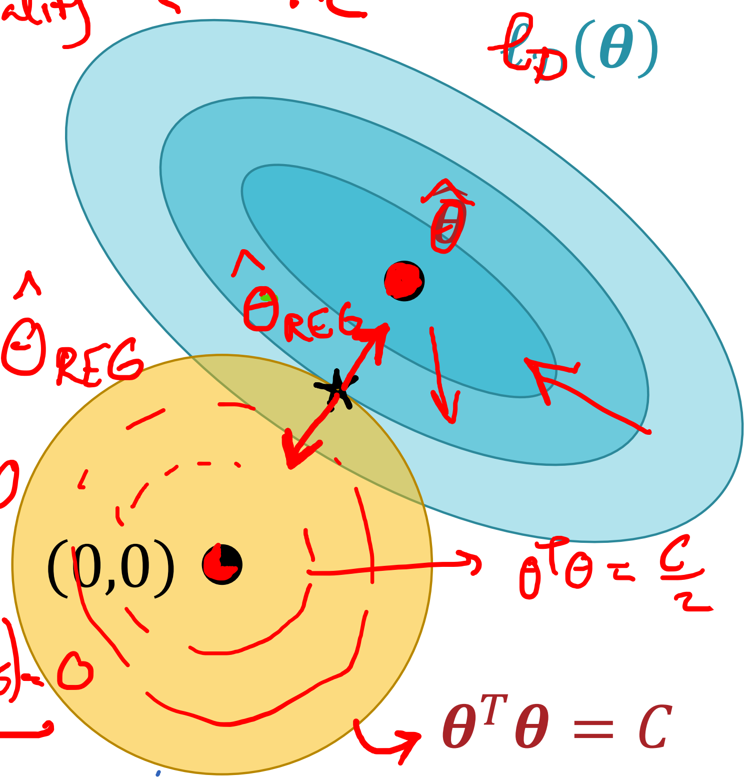
proportionality ← λ_C

$$\nabla_\theta \ell_D(\hat{\theta}_{REG}) \propto -2\hat{\theta}_{REG}$$

$$\Rightarrow \exists \lambda_C \geq 0 : \nabla_\theta \ell_D(\hat{\theta}_{REG}) = -2\lambda_C \hat{\theta}_{REG}$$

$$\Rightarrow \nabla_\theta \ell_D(\hat{\theta}_{REG}) + 2\lambda_C \hat{\theta}_{REG} = 0$$

$$\Rightarrow \nabla_\theta \underbrace{\left[\ell_D(\hat{\theta}_{REG}) + \lambda_C \hat{\theta}_{REG}^T \hat{\theta}_{REG}\right]}_{\ell_D^{AUG}} = 0$$

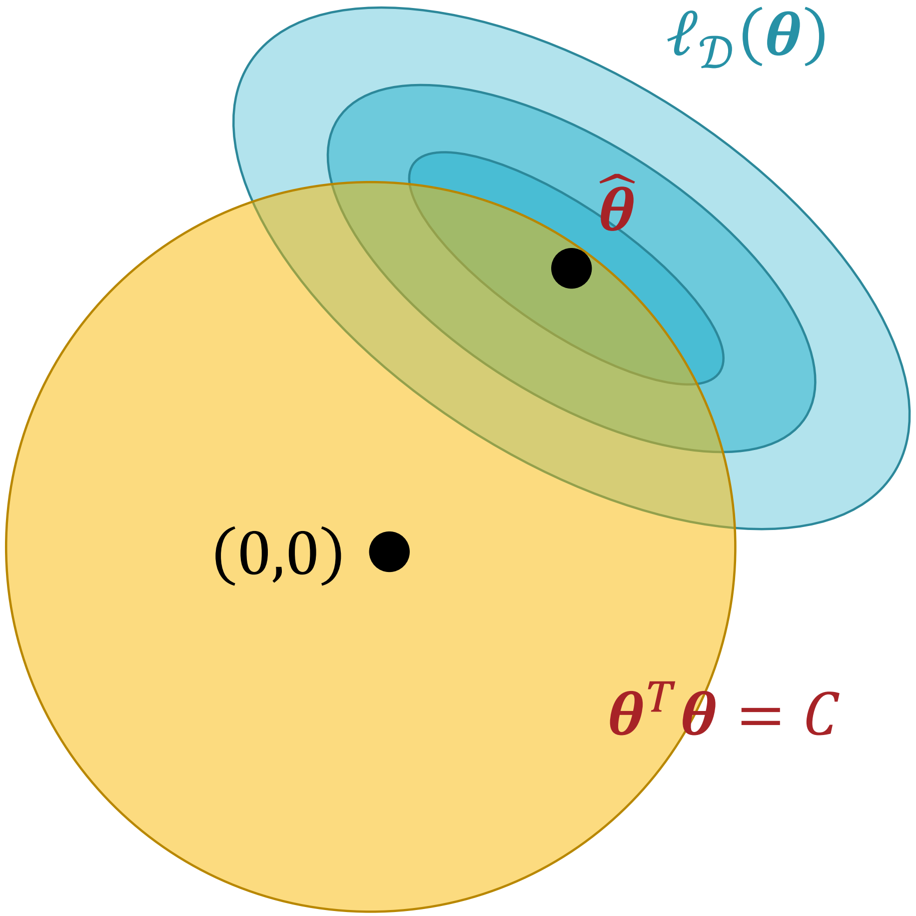


(**) $\min_\theta \; \ell_D(\theta) + \lambda_C \theta^T\theta \qquad \lambda_C \geq 0$

Soft Constraints

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

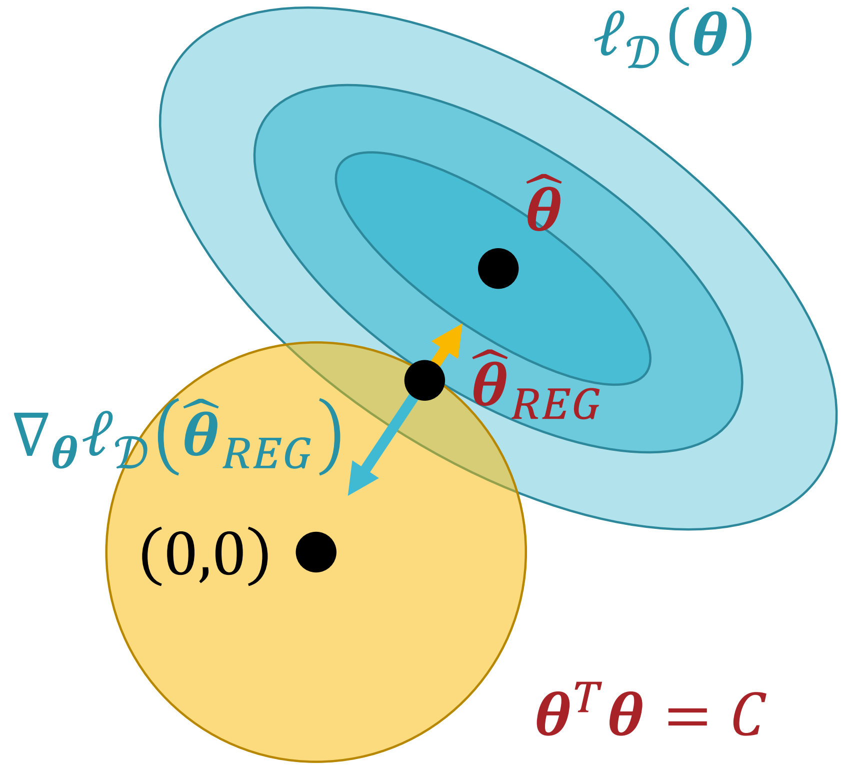
subject to $\boldsymbol{\theta}^T\boldsymbol{\theta} \leq C$



Soft Constraints

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

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



Soft Constraints: Solving for $\hat{\boldsymbol{\theta}}_{REG}$

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

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

$\Updownarrow$

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

Ridge Regression

$$MSE = (X^T\theta - Y)^T (X^T\theta - Y)$$

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

$$\nabla \ell^{AUG}(\hat{\theta}_{REG}) = 2X^TX\hat{\theta}_{REG} - 2X^TY + 2\lambda_C\hat{\theta}_{REG} = 0$$

$$\Rightarrow X^TX\hat{\theta}_{REG} + \lambda_C\hat{\theta}_{REG} = X^TY$$

$$\Rightarrow (X^TX + \lambda I)\hat{\theta}_{REG} = X^TY$$

$$\Rightarrow \boxed{\hat{\theta}_{REG} = (X^TX + \lambda_C I)^{-1} X^TY}$$

Poll Question 3

- Suppose we are minimizing $\ell_{\mathcal{D}}^{AUG}(\boldsymbol{\theta}) = \ell_{\mathcal{D}}(\boldsymbol{\theta}) + \lambda_C r(\boldsymbol{\theta})$. As λ_C increases, the minimum of $\ell_{\mathcal{D}}^{AUG}$ …

A. … moves towards the midpoint of $\ell_{\mathcal{D}}$ and r

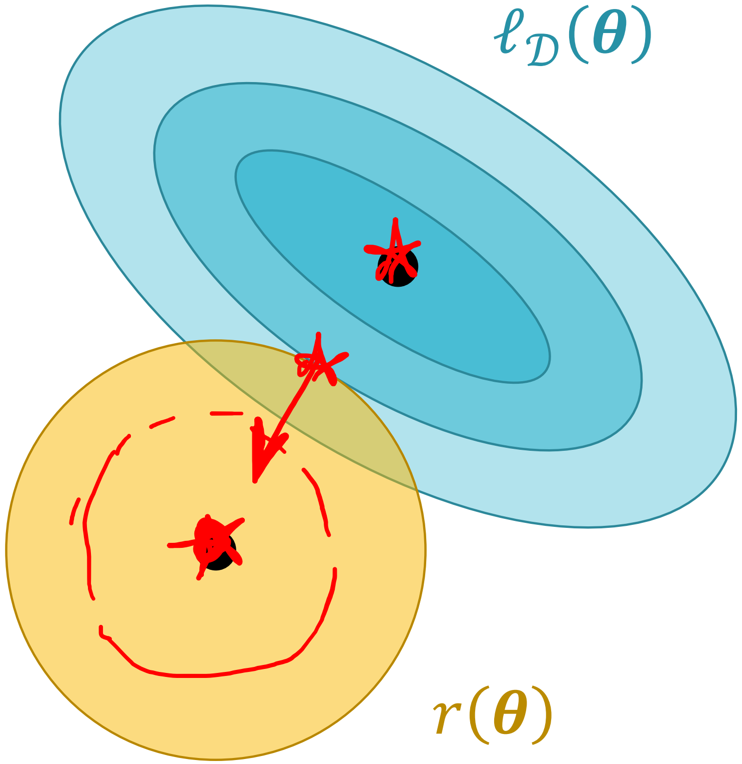
B. … moves towards the minimum of $\ell_{\mathcal{D}}$

C. … moves towards the minimum of r

D. … moves towards the vector of all infinities

E. … moves towards the vector of all ones

F. … stays the same



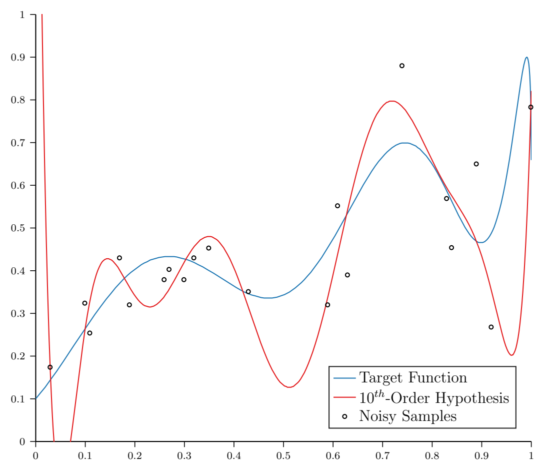
Regularization: Q & A

- Should we regularize the bias/intercept parameter, θ_0?

- Is feature scale a concern with regularization?

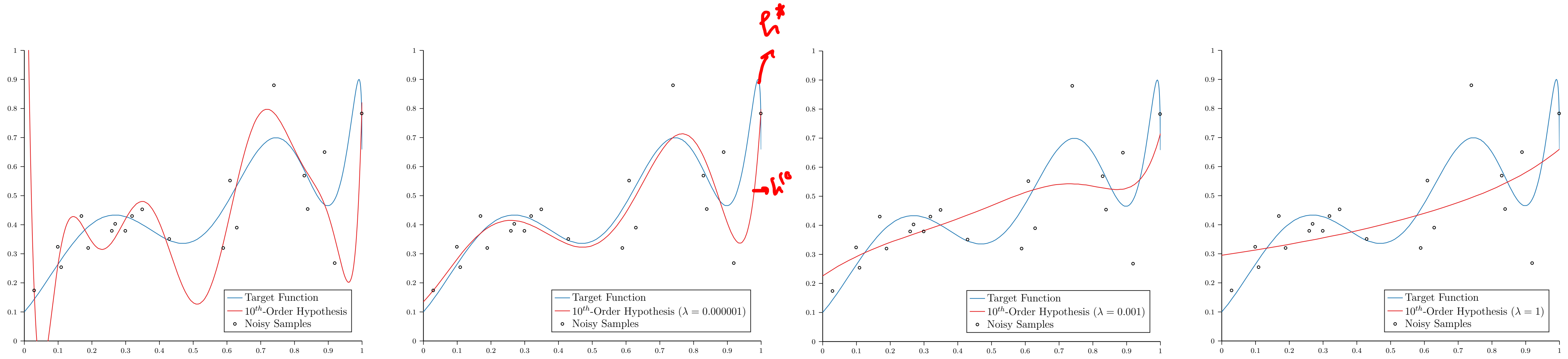
Regularization: Best Practices

- Should we regularize the bias/intercept parameter, θ_0?
  - No!
  - Regularizers typically avoid penalizing this term so that our classifiers can adapt to shifts in the y values
- Is feature scale a concern with regularization?
  - Yes!
  - Features at dramatically different scales might have vastly different coefficient values
  - When using regularization, it is common to *standardize* the features first by subtracting the mean and dividing by the standard deviation



Ridge Regression

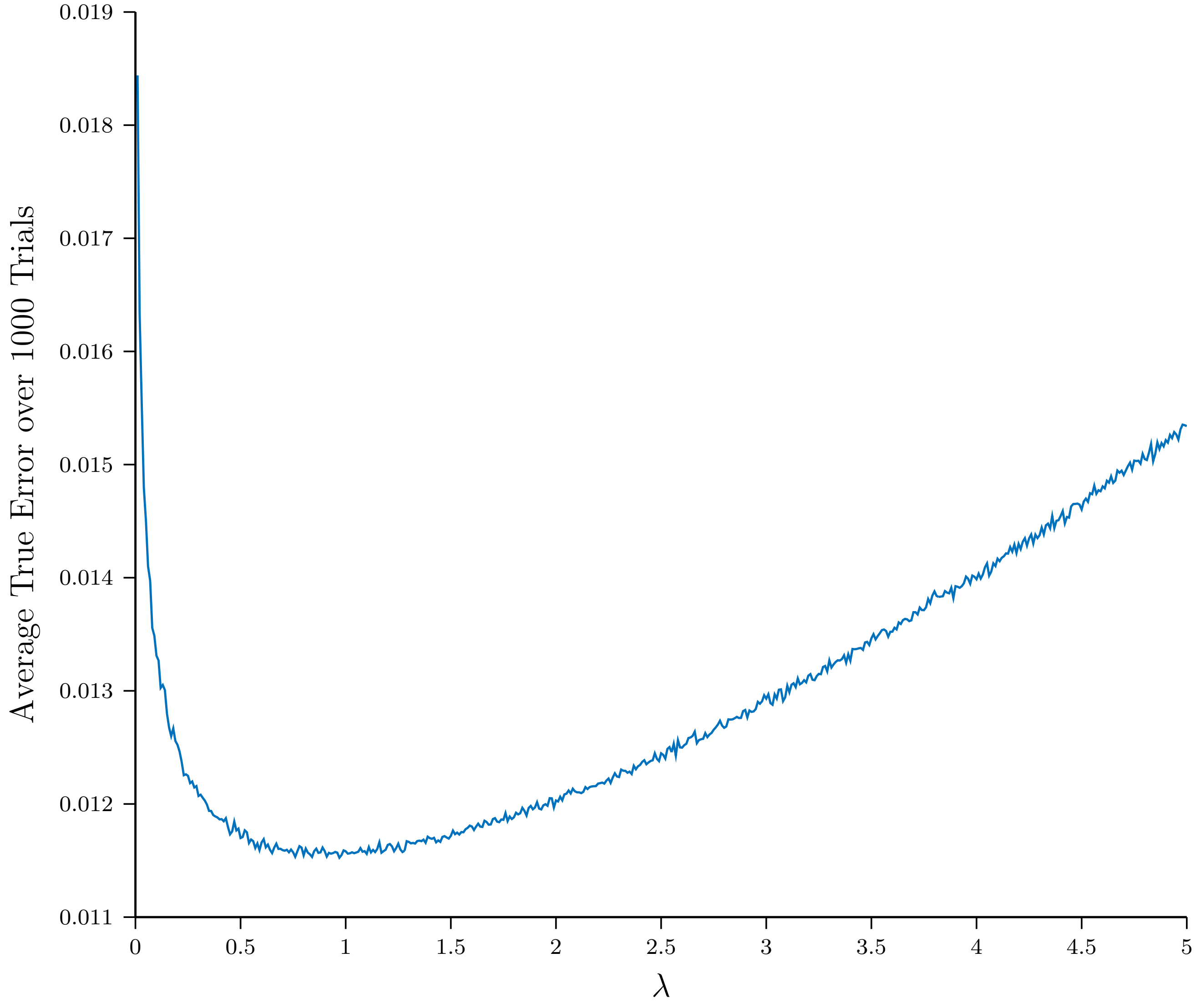
- 10-dimensional target function with additive Gaussian noise
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomial



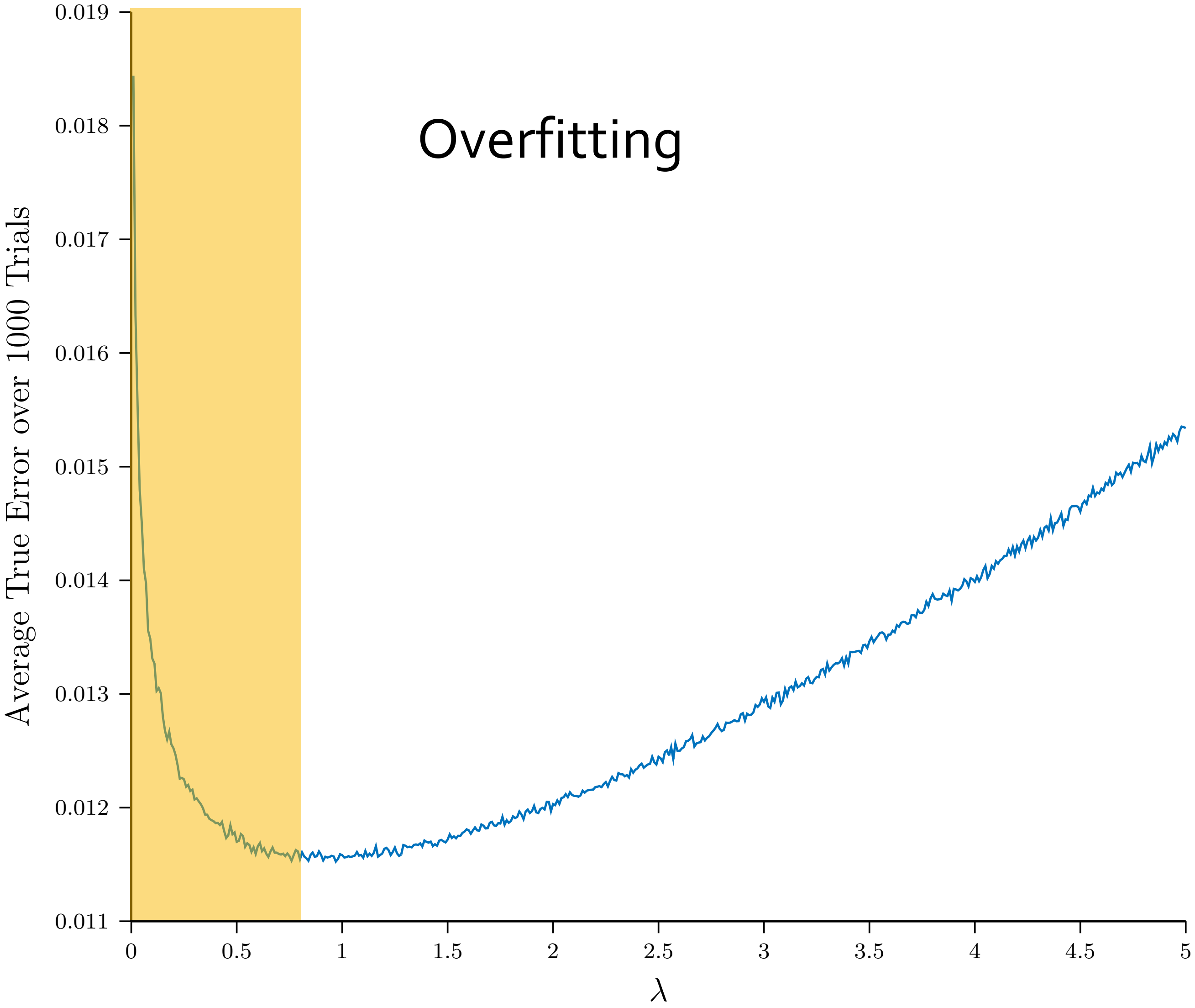
Ridge Regression

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

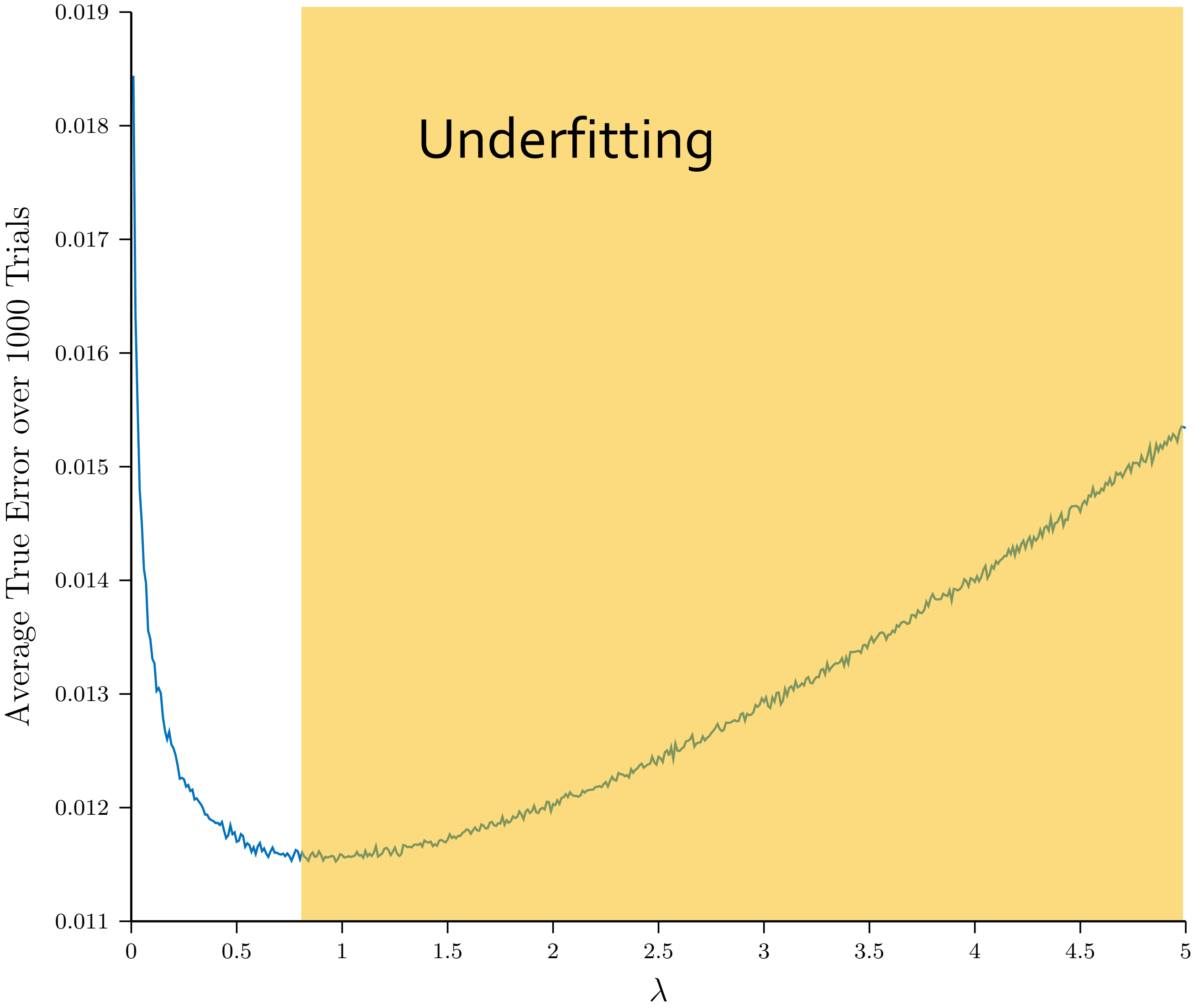
Setting λ



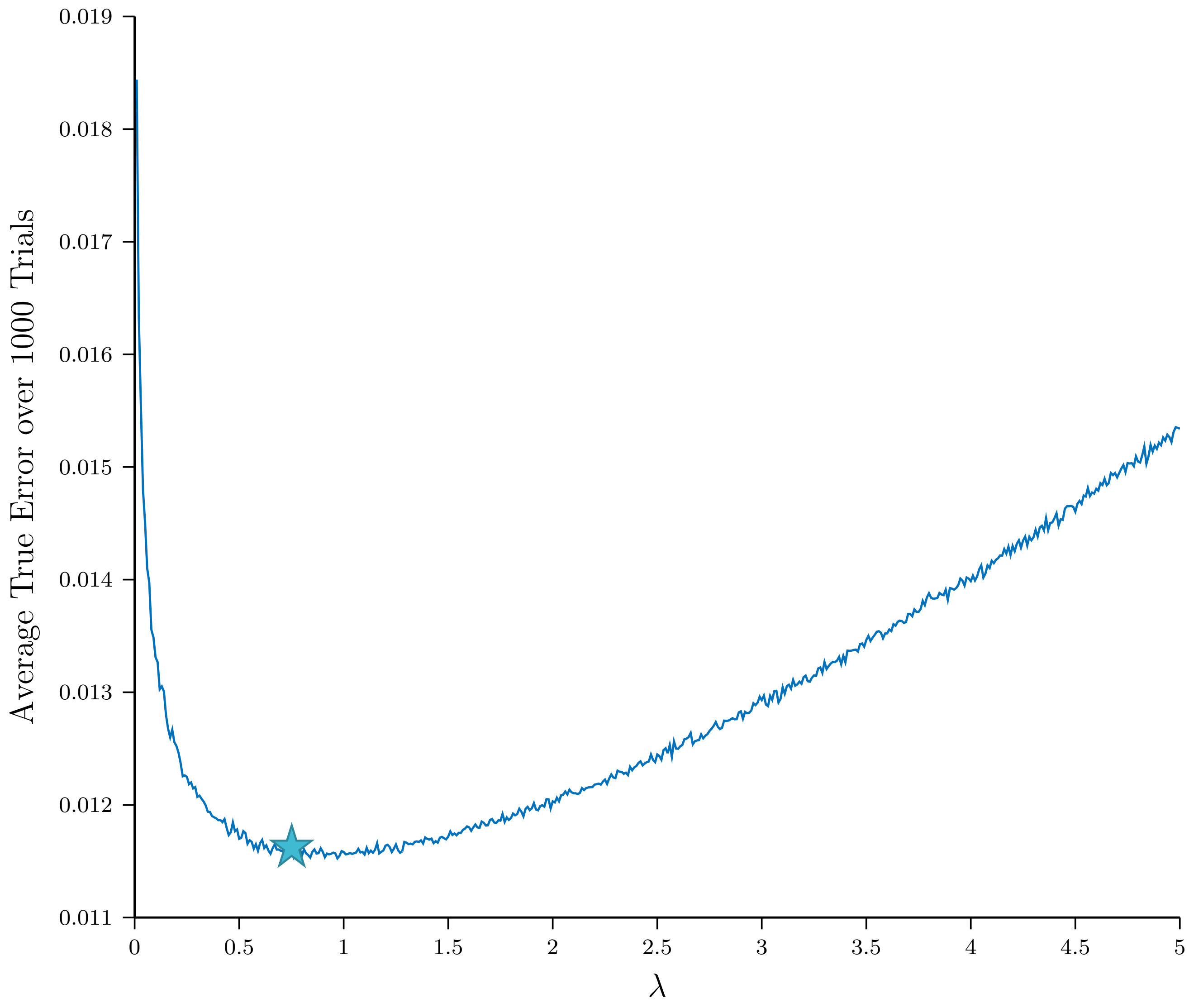
Setting λ



Setting λ

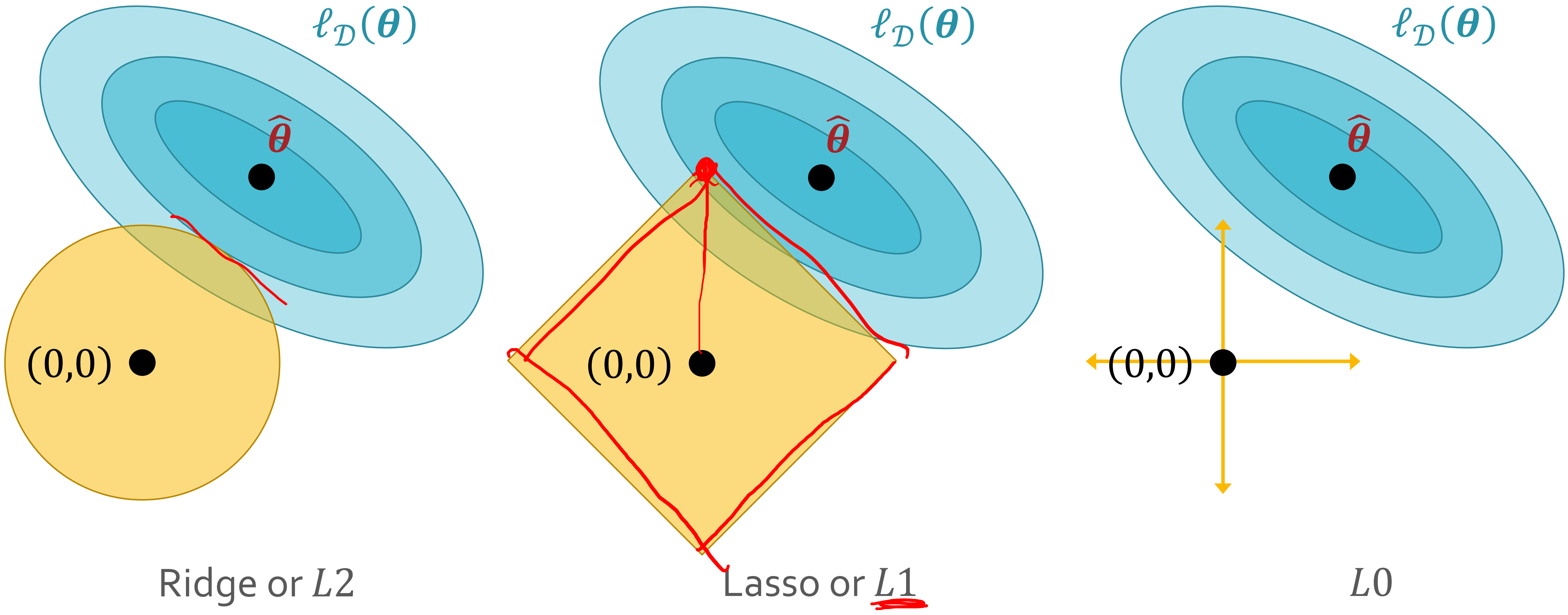


Setting λ



Other Regularizers

$\ell_{\mathcal{D}}(\boldsymbol{\theta}) + \lambda r(\theta)$		
Ridge or $L2$	$r(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_2^2 = \sum_{d=0}^{D} \theta_d^2$	Encourages small weights
Lasso or $L1$	$r(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_1 = \sum_{d=0}^{D} \lvert\theta_d\rvert$	Encourages sparsity
$L0$	$r(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_0 = \sum_{d=0}^{D} \mathbb{1}(\theta_d \neq 0)$	Encourages sparsity (intractable)



Other Regularizers

Regularization Learning Objectives

You should be able to…

- Identify when a model is overfitting
- Add a regularizer to an existing objective in order to combat overfitting
- Explain why we should not regularize the bias term
- Convert linearly inseparable dataset to a linearly separable dataset in higher dimensions