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

Henry Chai & Matt Gormley

10/2/23

Front Matter

- Announcements:
  - HW4 released 9/29, due 10/9 at 11:59 PM

Recall: Logistic Regression

- Model:

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

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

- Objective: minimize the negative conditional log-likelihood

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

- Derivatives

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

- Gradients

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

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

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

- Predictions

$$\hat{y} = \underset{y\in\{0,1\}}{\text{argmax}}\ p\left(y\middle|\boldsymbol{x}',\hat{\boldsymbol{\theta}}\right)$$
$$\vdots$$
$$= \text{“sign”}\left(\hat{\boldsymbol{\theta}}^T\boldsymbol{x}'\right)$$

Logistic Regression Decision Boundary

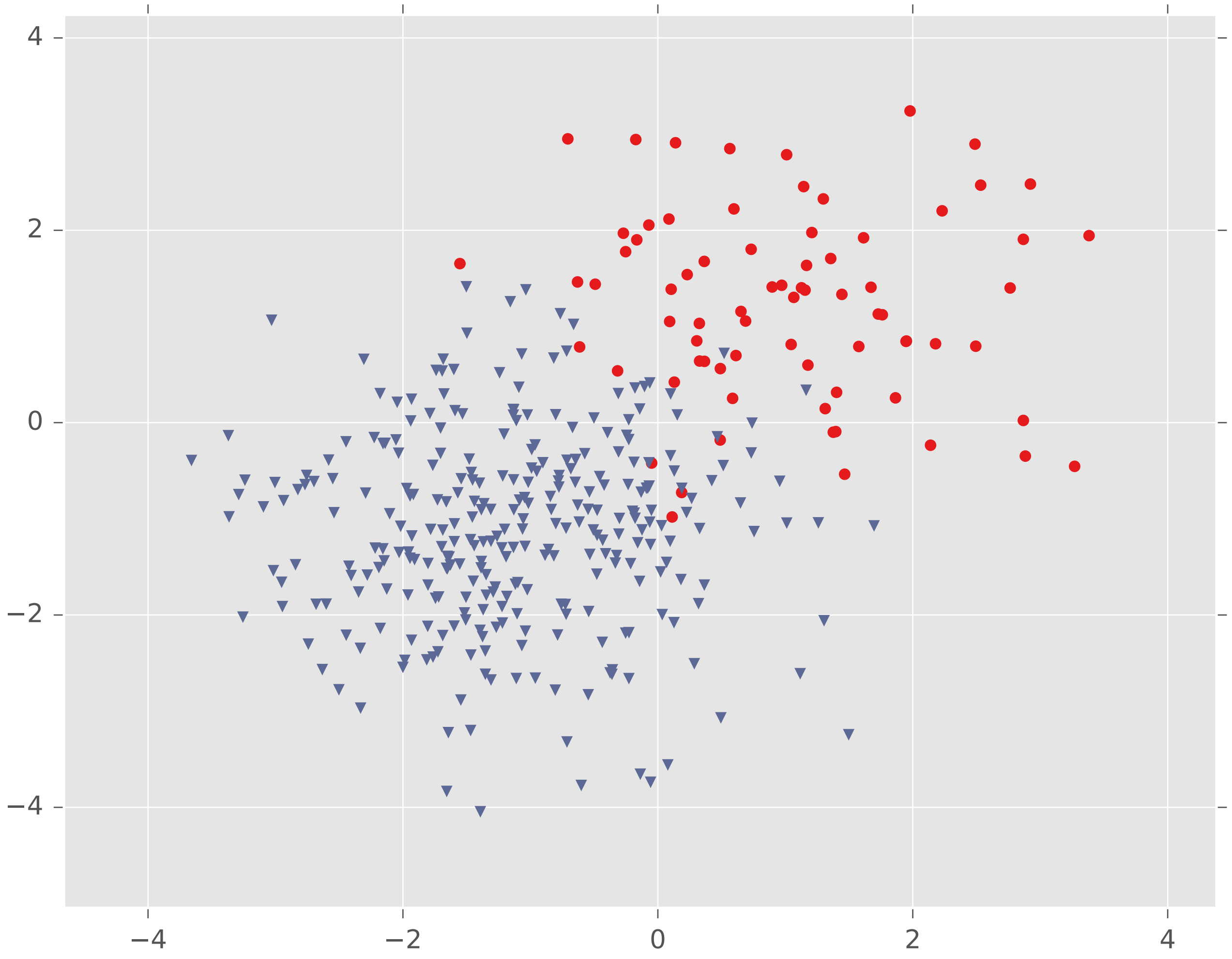


Figure courtesy of Matt Gormley

Logistic Regression Decision Boundary

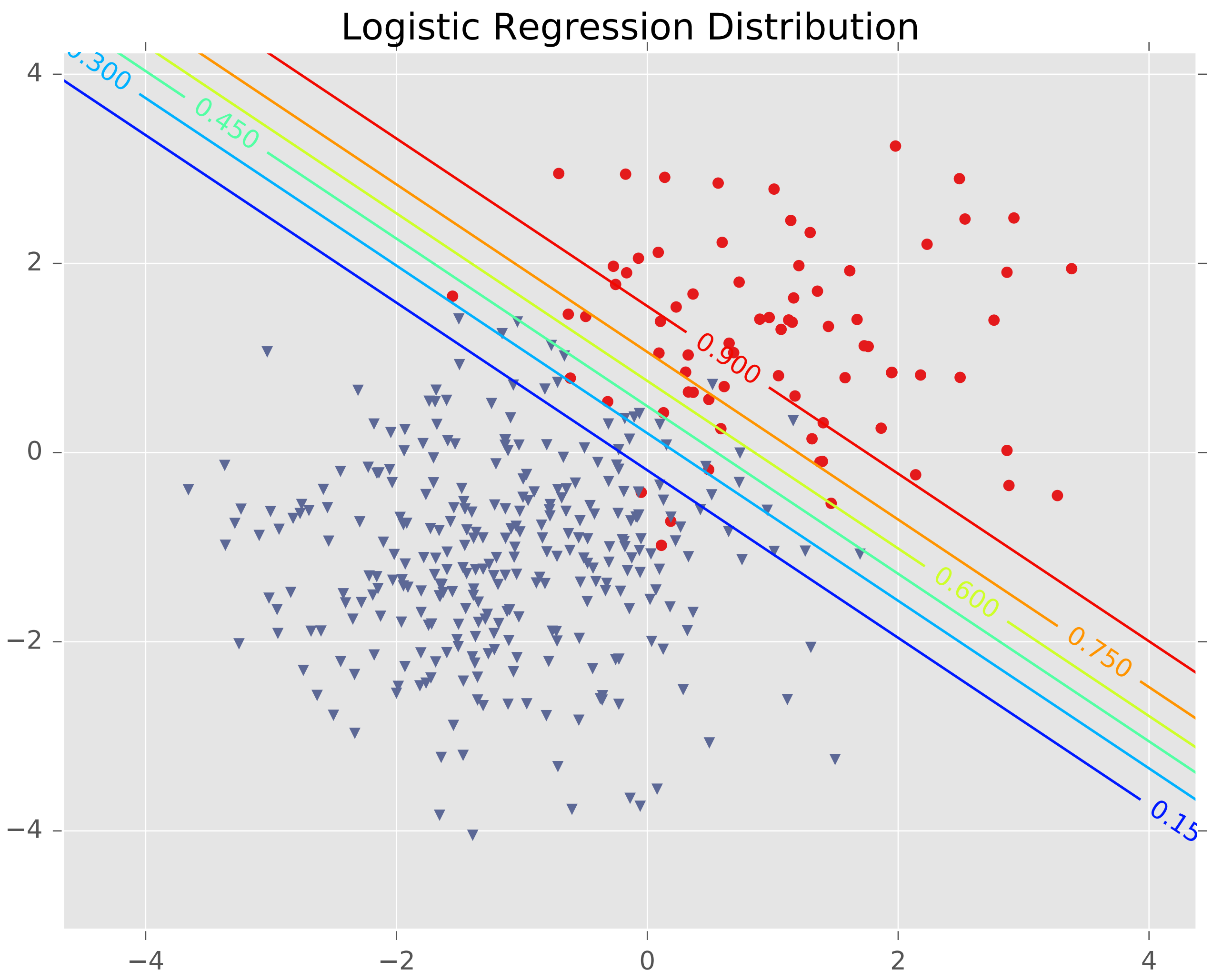


Figure courtesy of Matt Gormley

Logistic Regression Decision Boundary

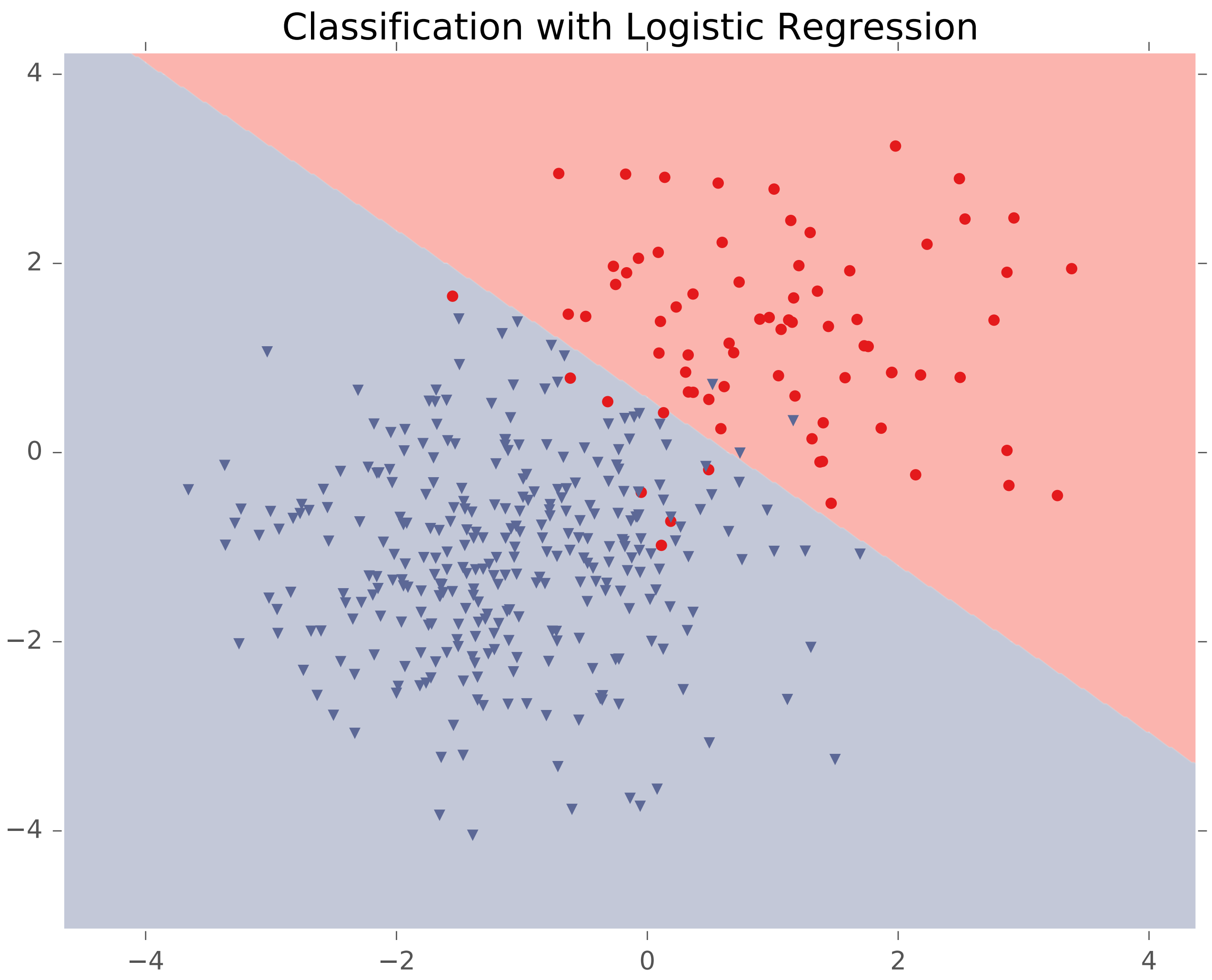


Figure courtesy of Matt Gormley

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 $i \in \text{shuffle}(\{1, \ldots, N\})$
      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

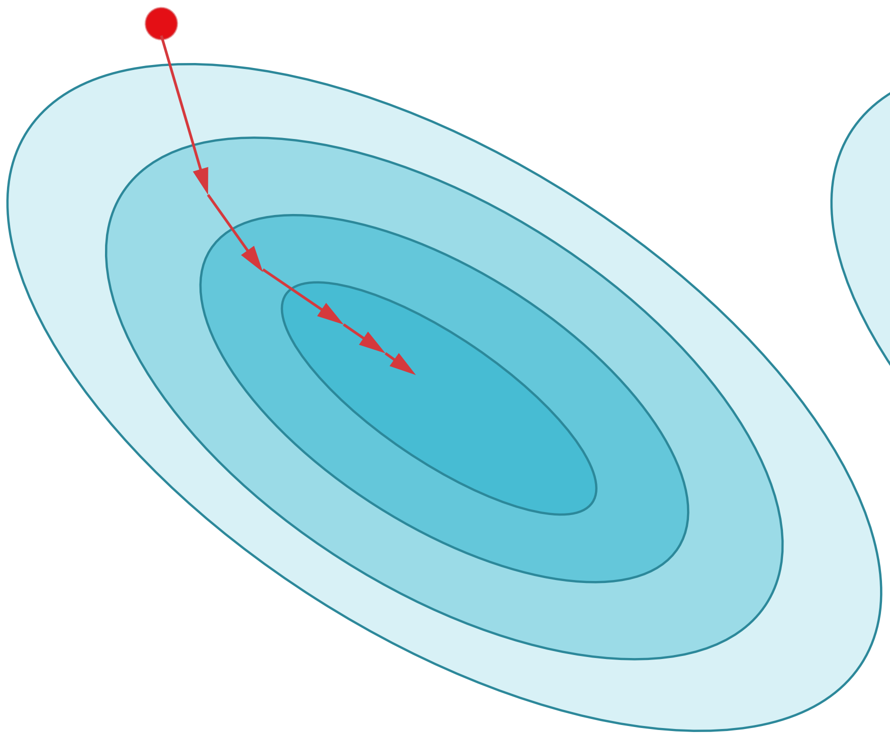
1. Initialize $\boldsymbol{\theta}^{(0)}$ to all zeros and set $t = 0$
2. While TERMINATION CRITERION is not satisfied
   a. For $i \in \text{shuffle}(\{1, \dots, N\})$
      i. Compute the pointwise gradient:

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

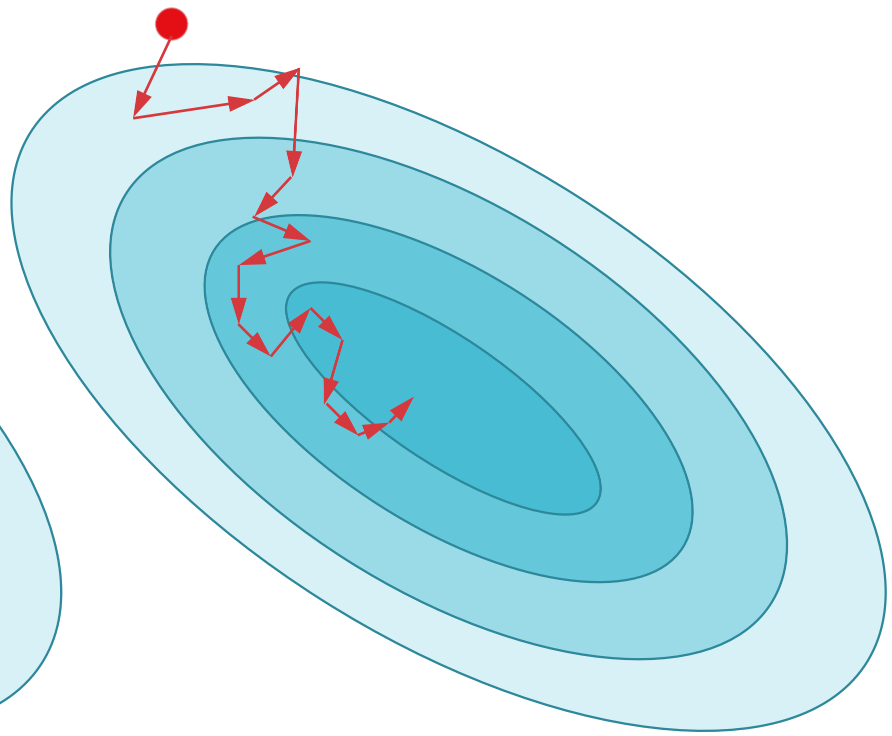
      ii. Update $\boldsymbol{\theta}$: $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t)} - \gamma \nabla J^{(i)}(\boldsymbol{\theta}^{(t)})$
      iii. Increment t: $t \leftarrow t + 1$

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

Stochastic Gradient Descent vs. Gradient Descent



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: $t \leftarrow t + 1$

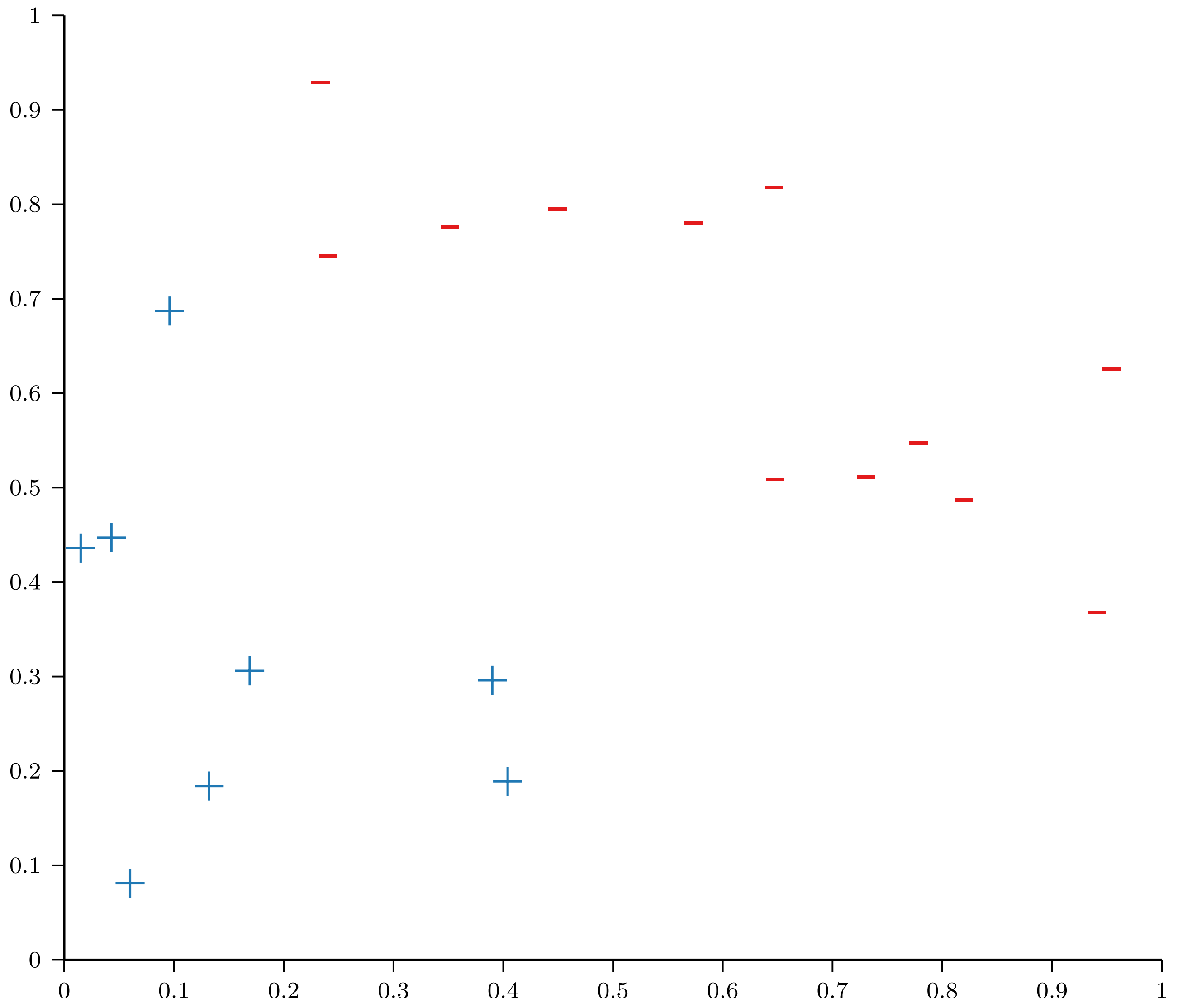
- Output: $\boldsymbol{\theta}^{(t)}$

Logistic Regression Learning Objectives

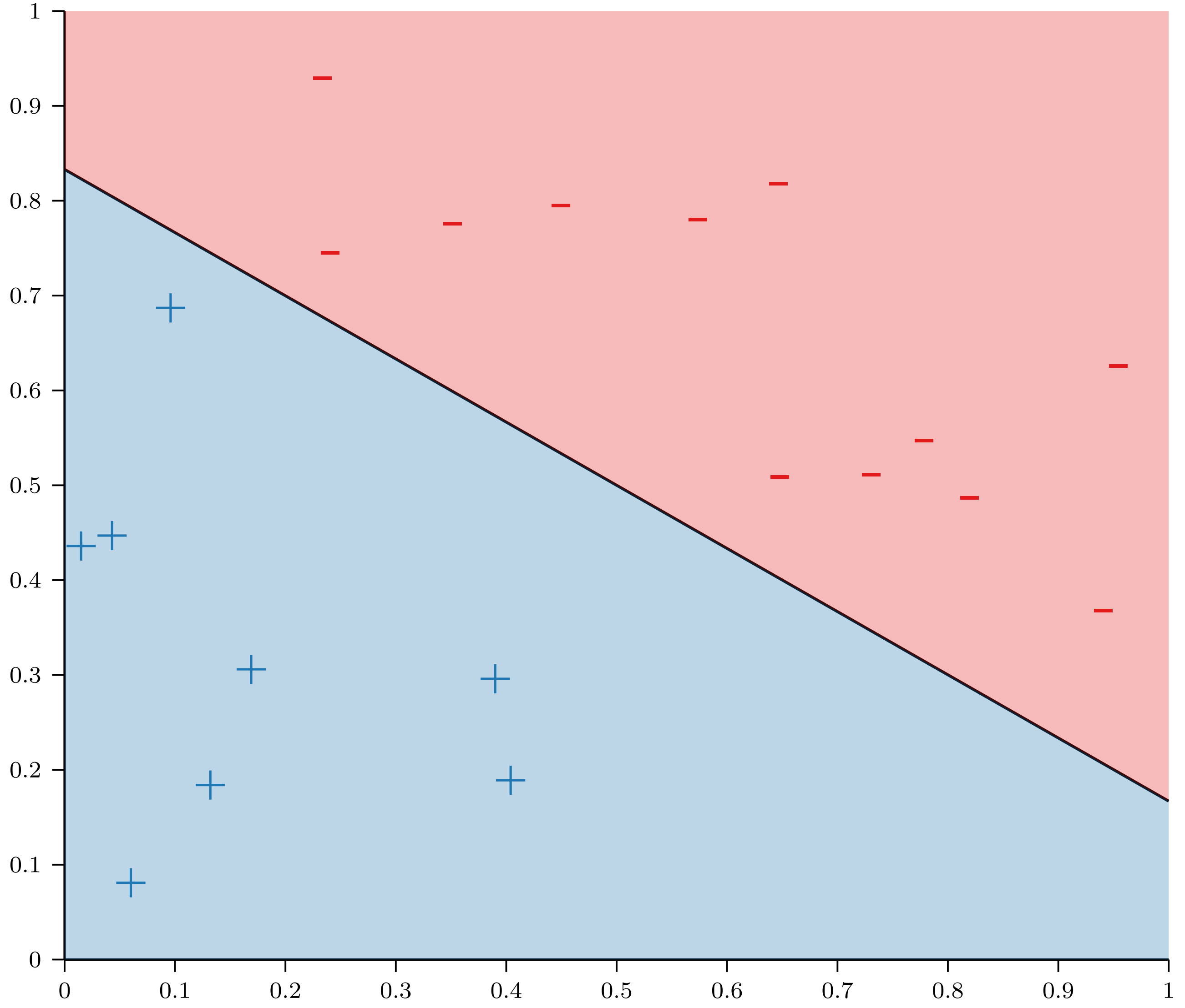
You should be able to…

- 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

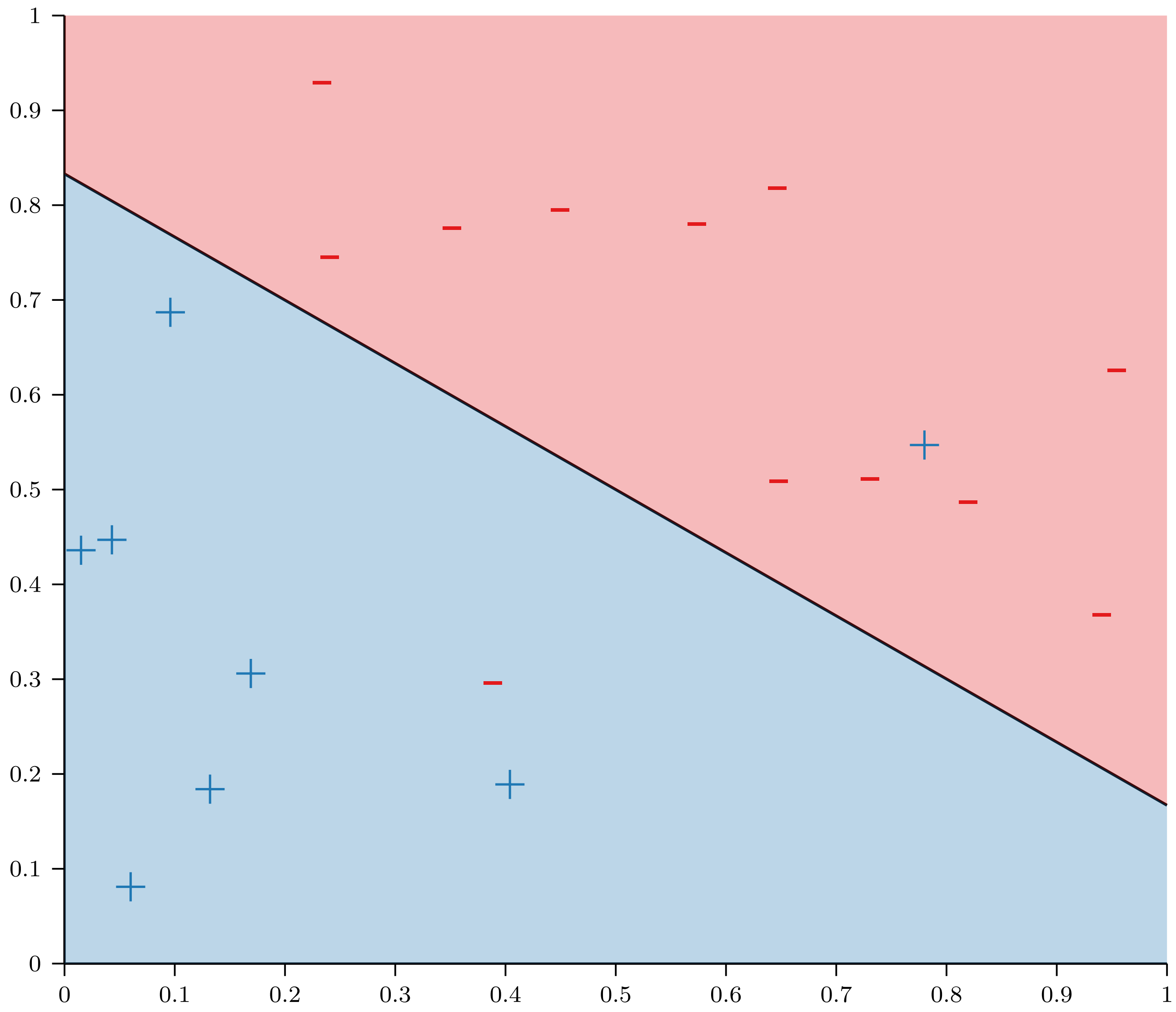
Linear Models



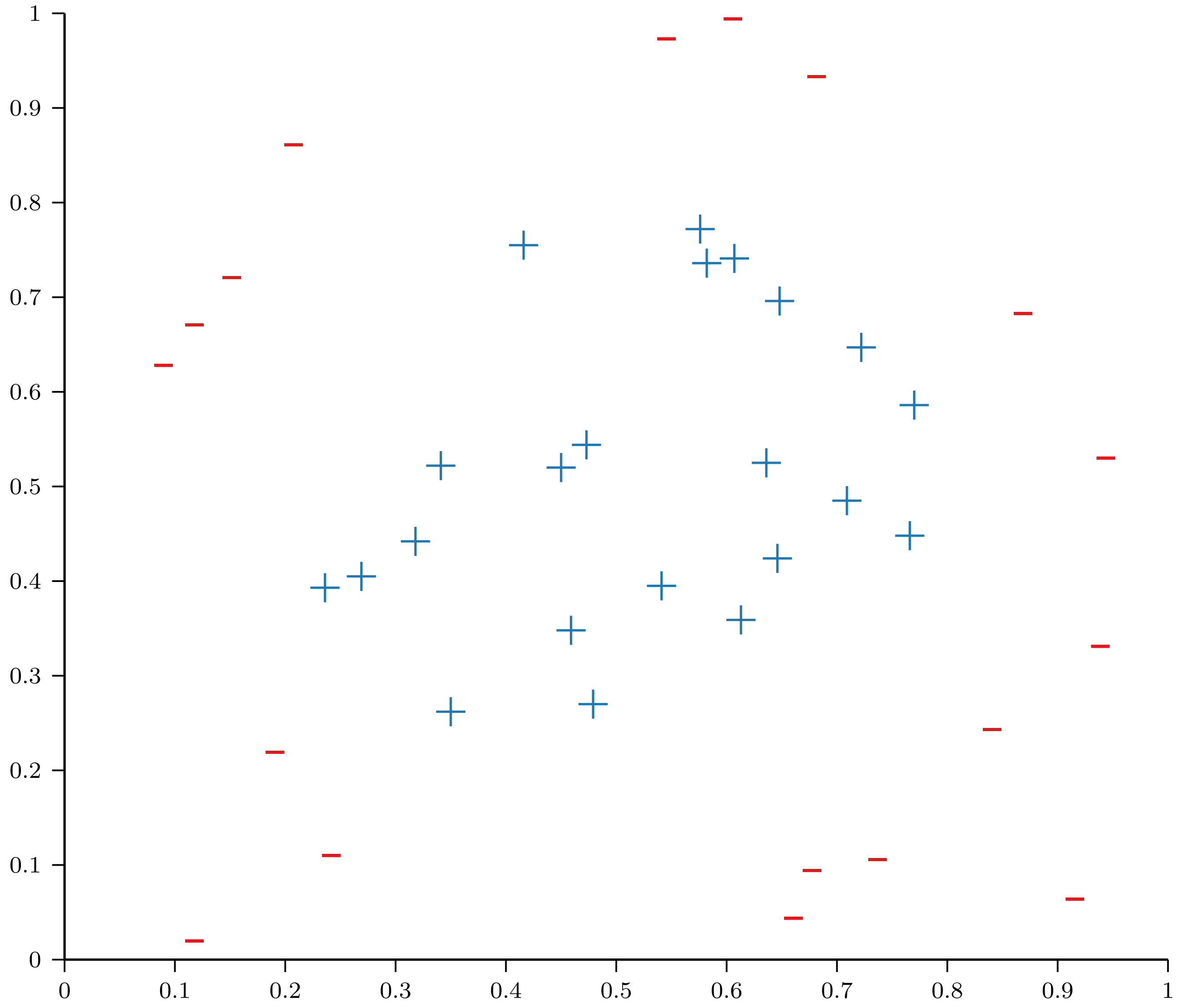
Linear Models



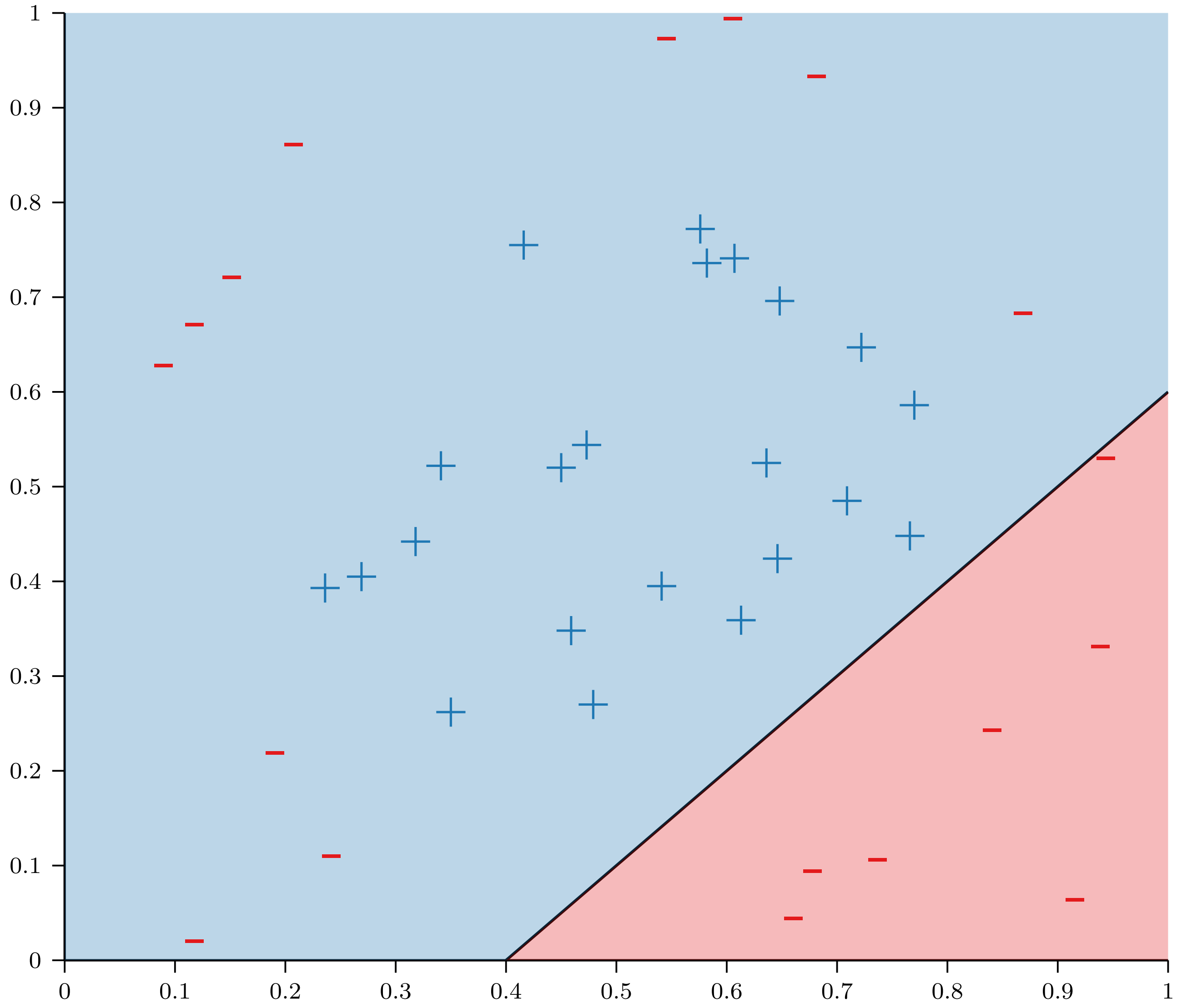
Linear Models



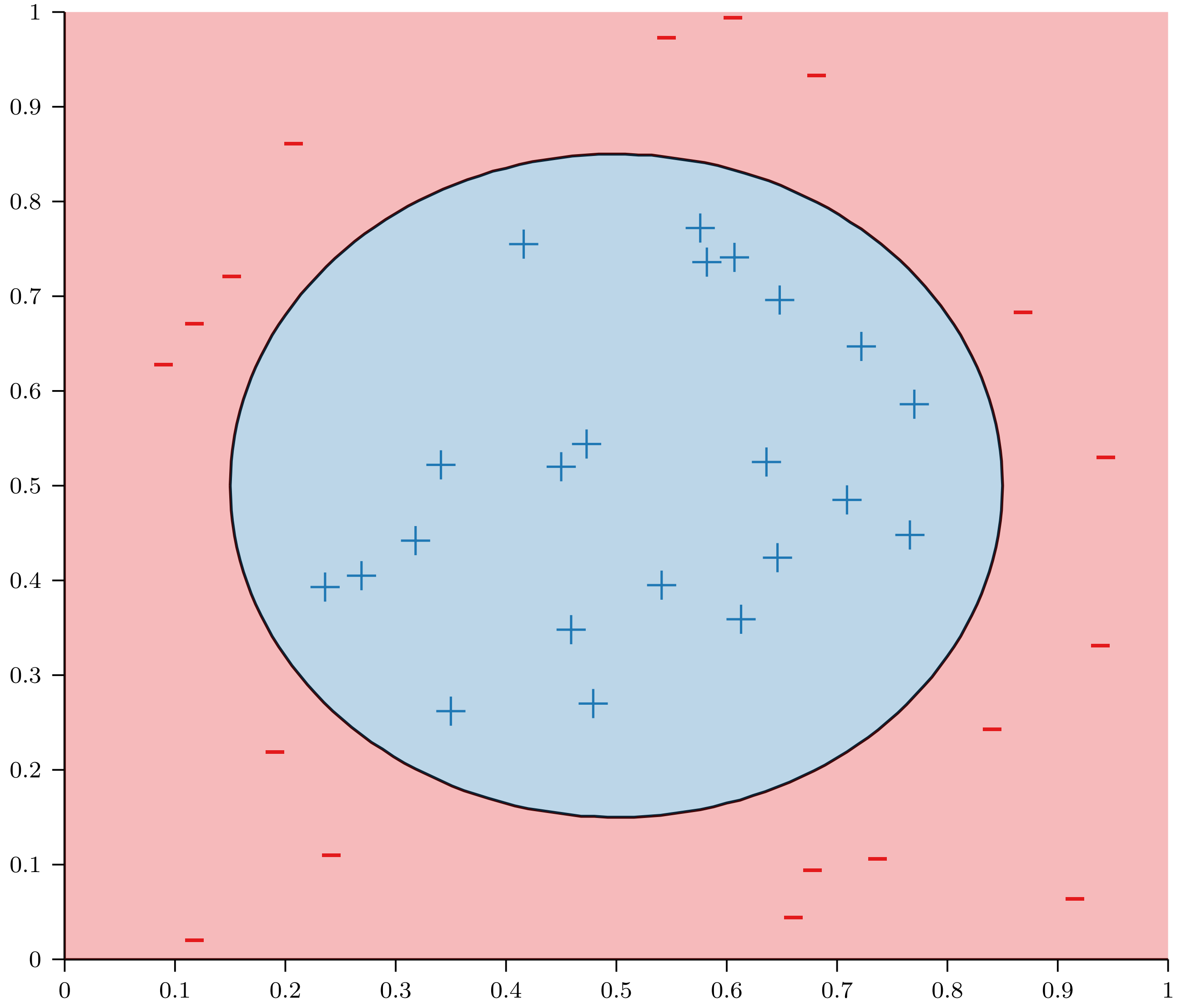
Linear Models?



Linear Models?



Nonlinear Models

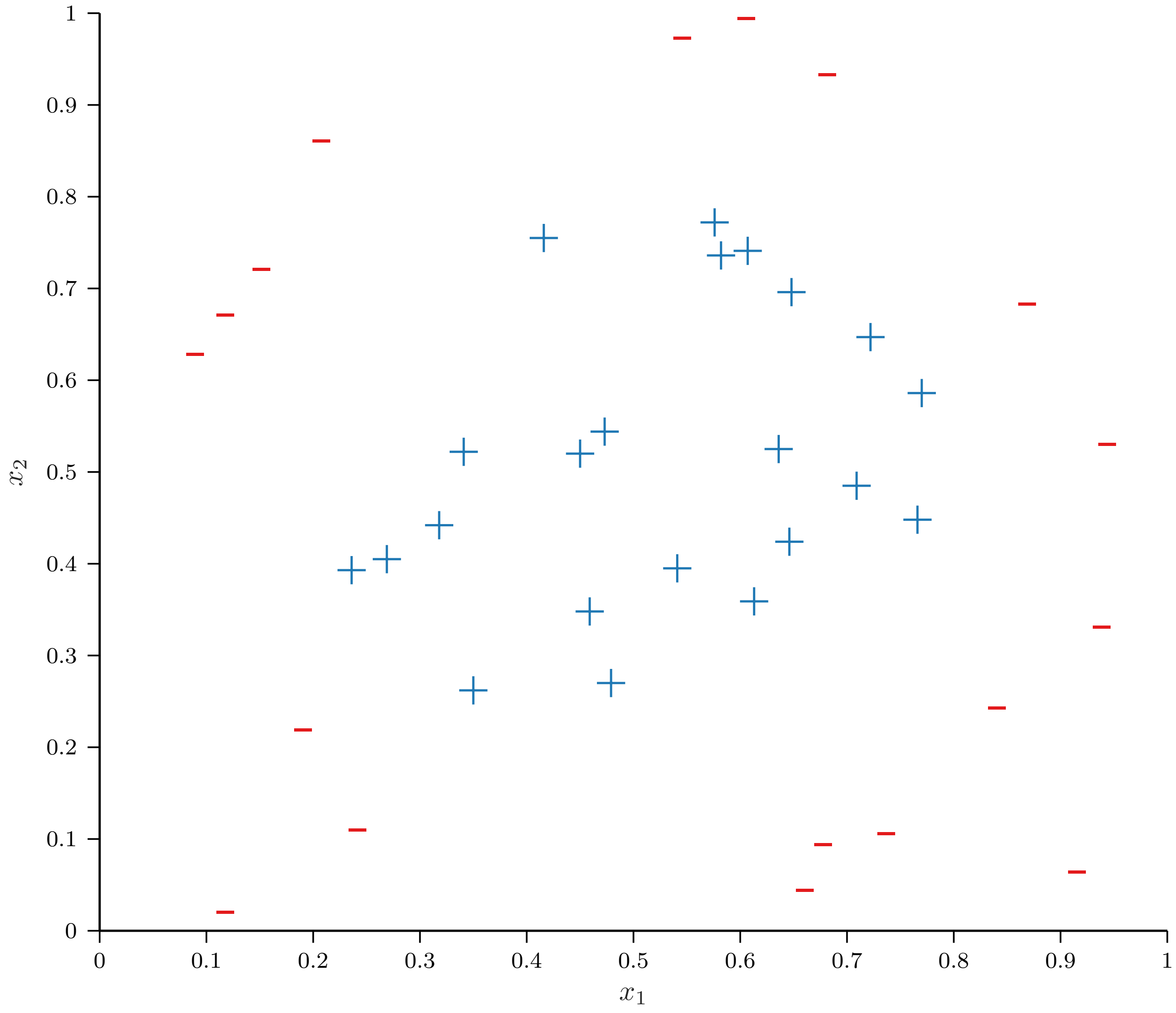


Feature Transforms

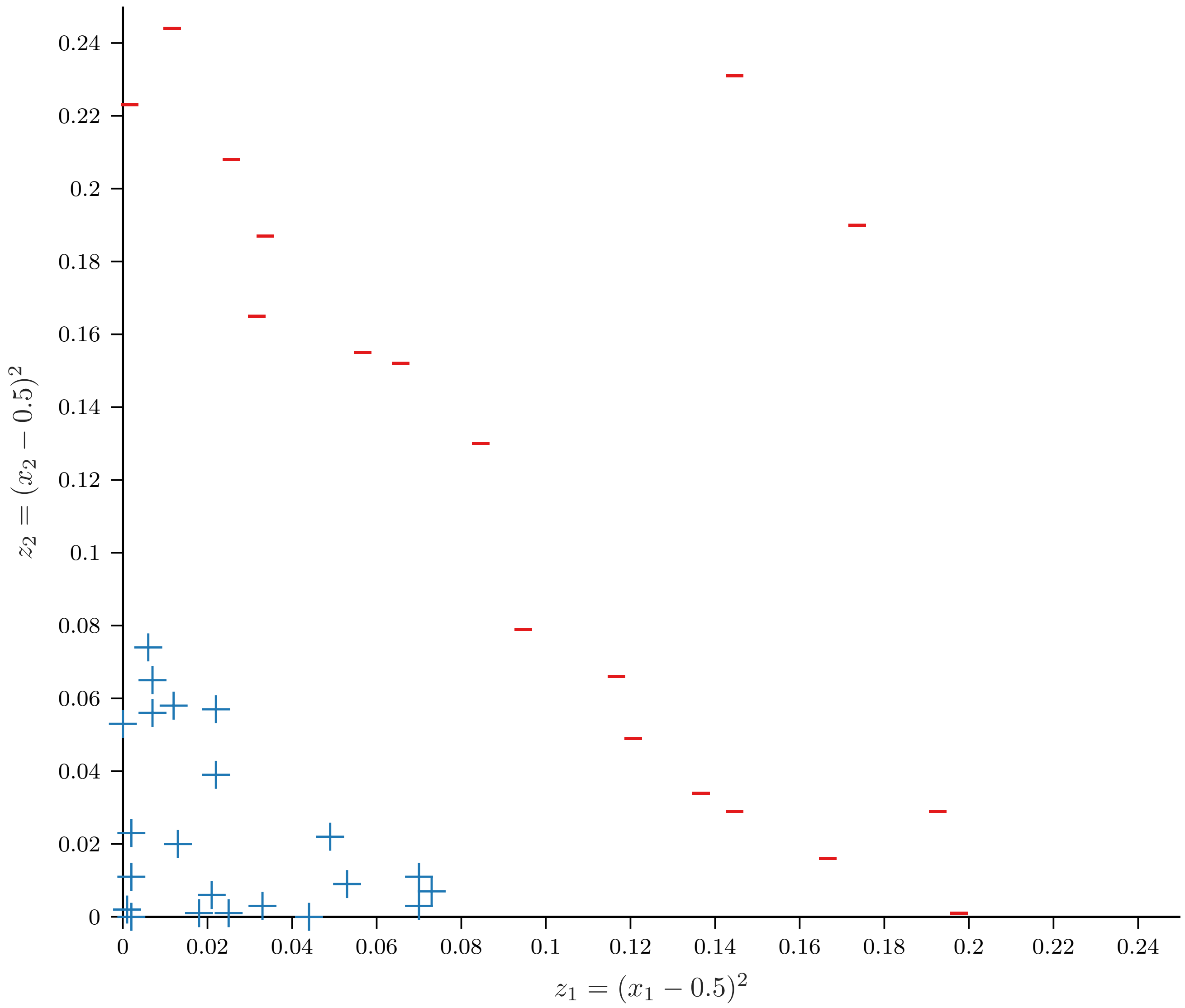
- Given D-dimensional inputs $\boldsymbol{x} = [x_1, \dots, x_D]$, first compute some transformation of our input, e.g.,

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

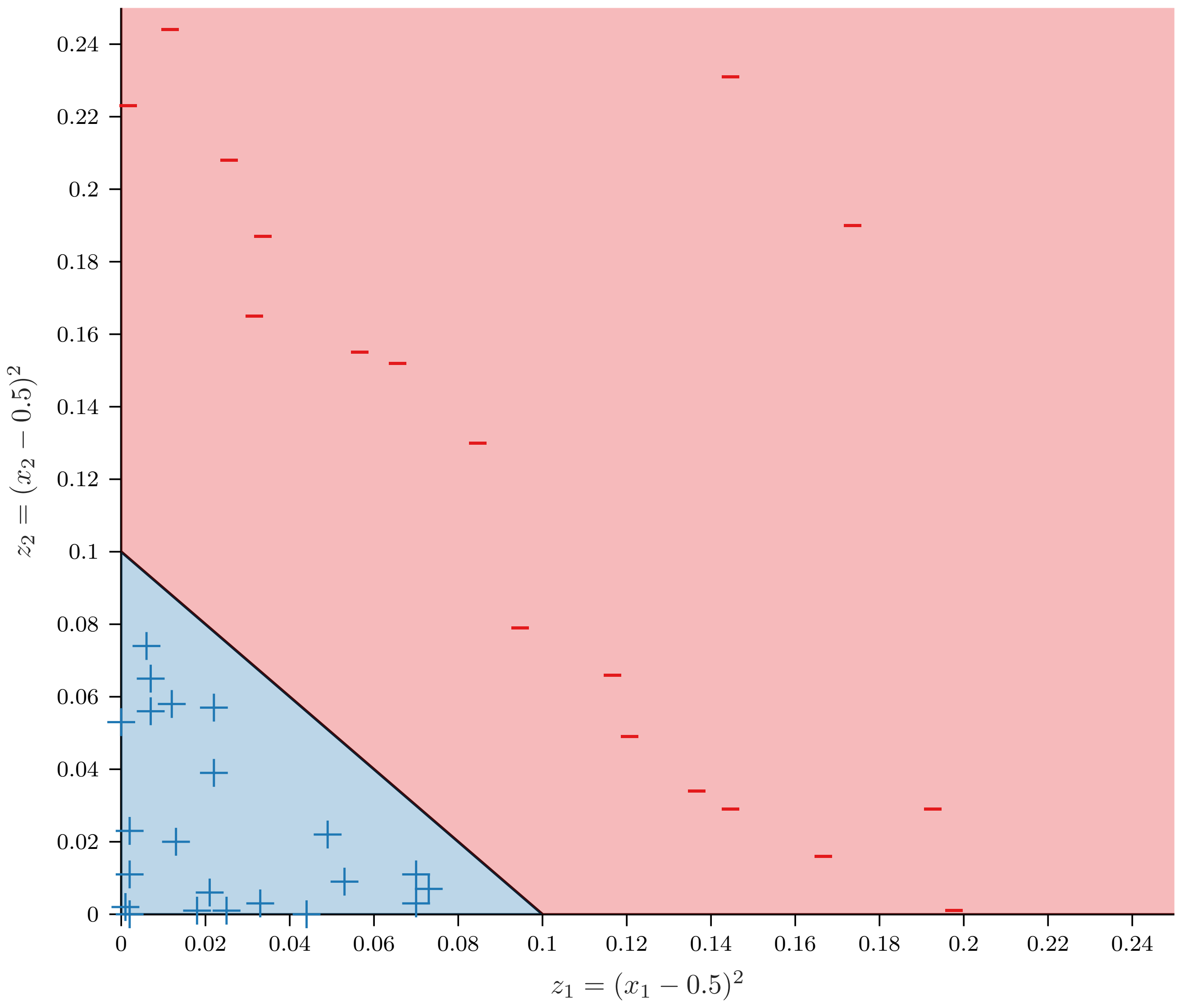
Nonlinear Models



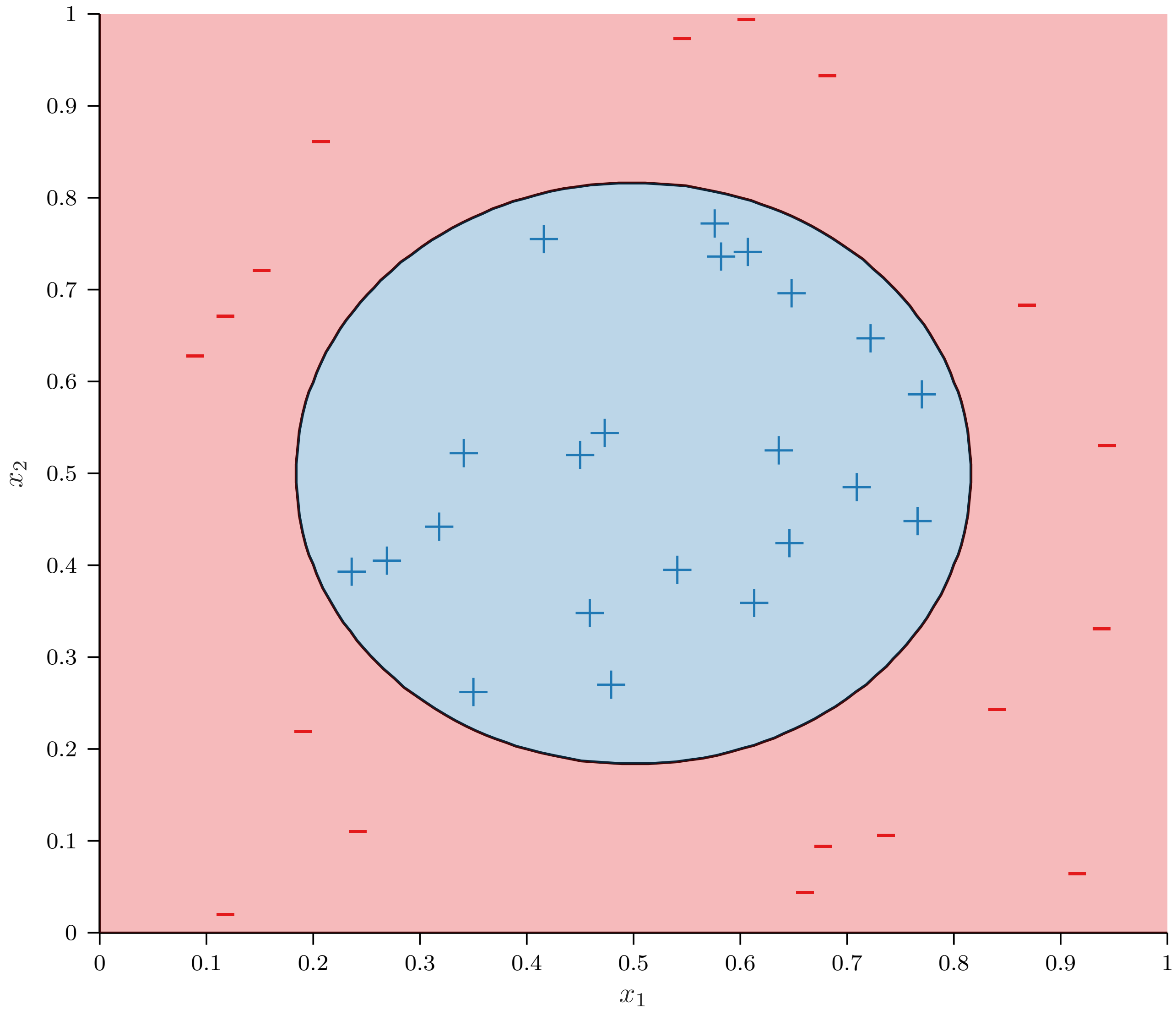
Nonlinear Models



Nonlinear Models



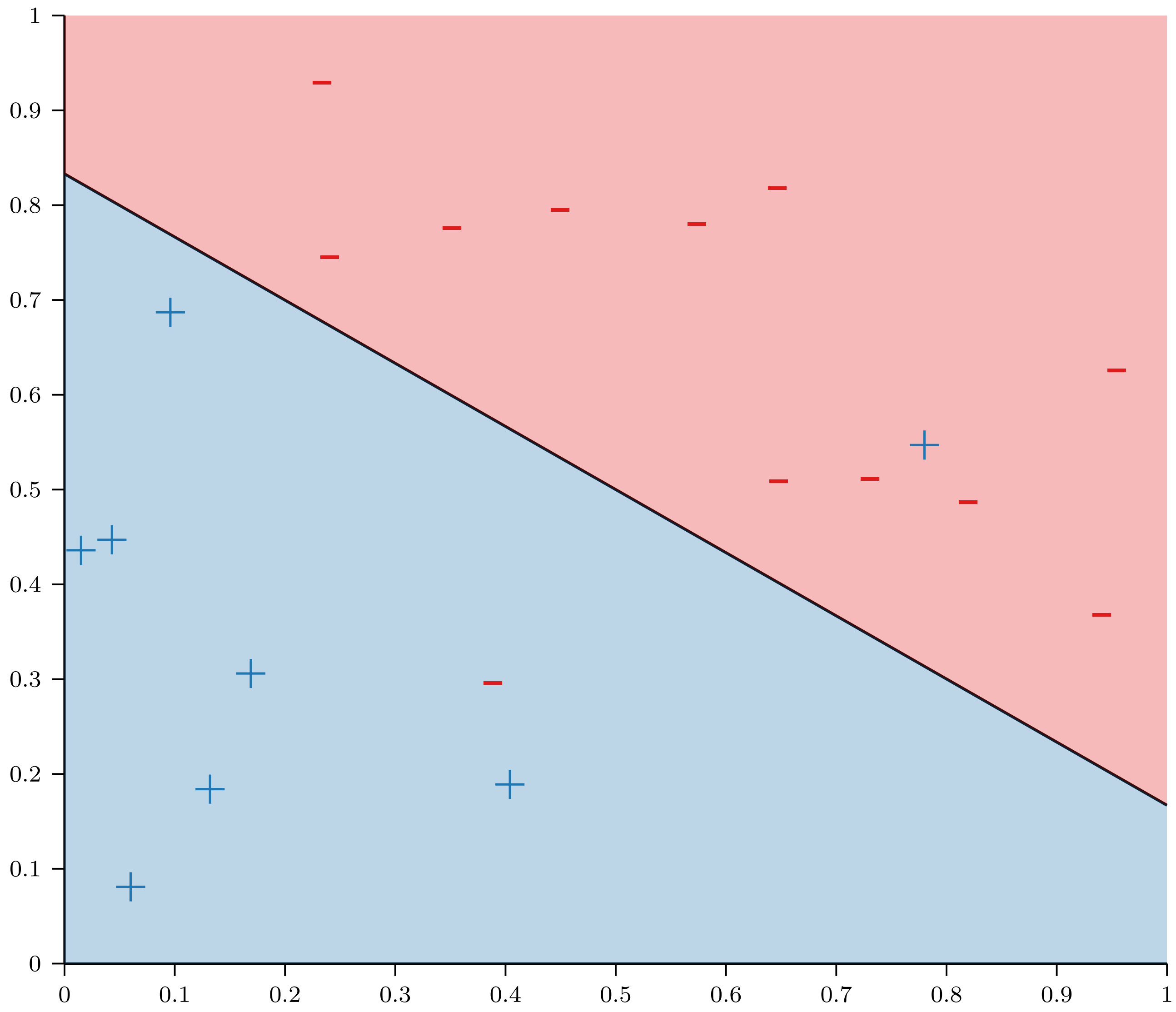
Nonlinear Models



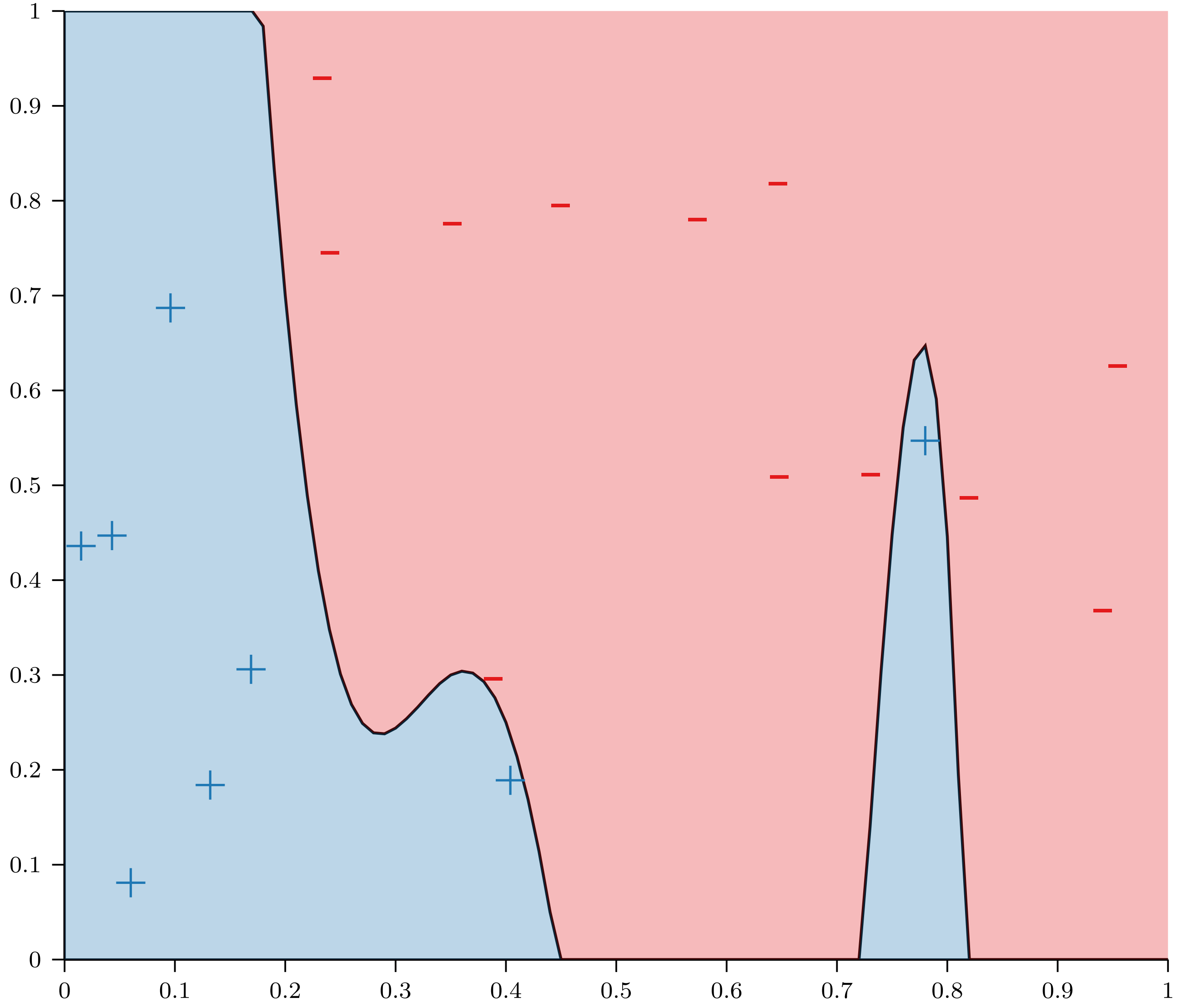
General Q^{th}-order Transforms

- $\phi_{2,2}([x_1, x_2]) = [x_1, x_2, x_1^2, x_1x_2, x_2^2]$
- $\phi_{2,3}([x_1, x_2]) = [x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3]$
- $\phi_{2,4}([x_1, x_2]) =$
  $[x_1, x_2, x_1^2, x_1x_2, x_2^2, x_1^3, x_1^2x_2, x_1x_2^2, x_2^3, x_1^4, x_1^3x_2, x_1^2x_2^2, x_1x_2^3, x_2^4]$
- $\phi_{2,Q}$ maps a 2-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

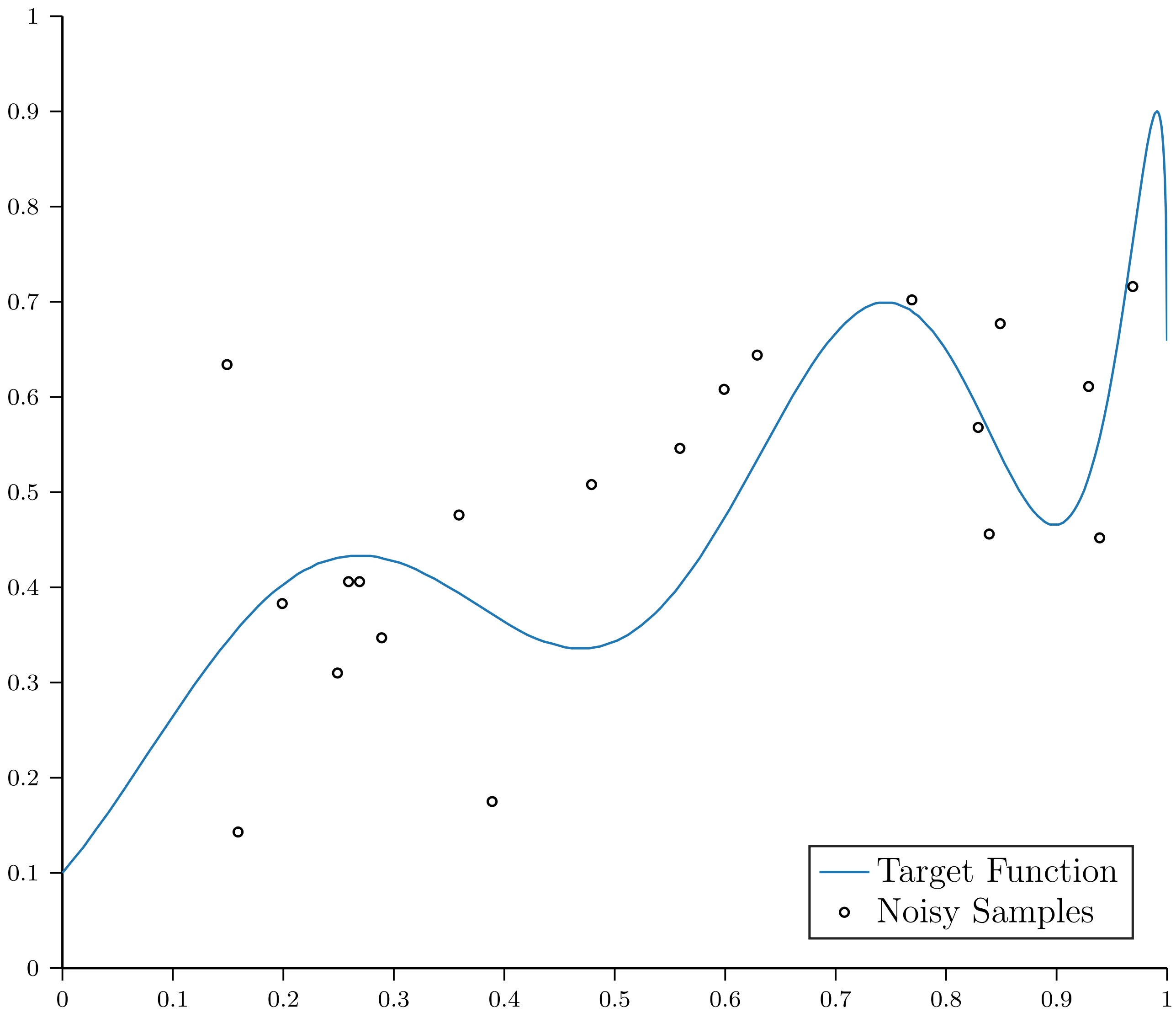
- $x \in \mathbb{R}$, $y \in \mathbb{R}$ and $N = 20$
- Targets are generated by a 10^{th}-order polynomial in x with additive Gaussian noise:

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

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

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



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

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

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

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

- Subject to

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

Hard Constraints

- $\mathcal{H}_2 = 2^{\text{nd}}$-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 $\mathrm{X} = \begin{bmatrix} 1 & \phi_1\left(\boldsymbol{x}^{(1)}\right) & \cdots & \phi_m\left(\boldsymbol{x}^{(1)}\right) \\ \vdots & \vdots & \ddots & \vdots \\ 1 & \phi_1\left(\boldsymbol{x}^{(N)}\right) & \cdots & \phi_m\left(\boldsymbol{x}^{(N)}\right) \end{bmatrix}$ and $\boldsymbol{y} = \begin{bmatrix} y^{(1)} \\ y^{(2)} \\ \vdots \\ y^{(N)} \end{bmatrix}$, find $\boldsymbol{\omega}$ that minimizes

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

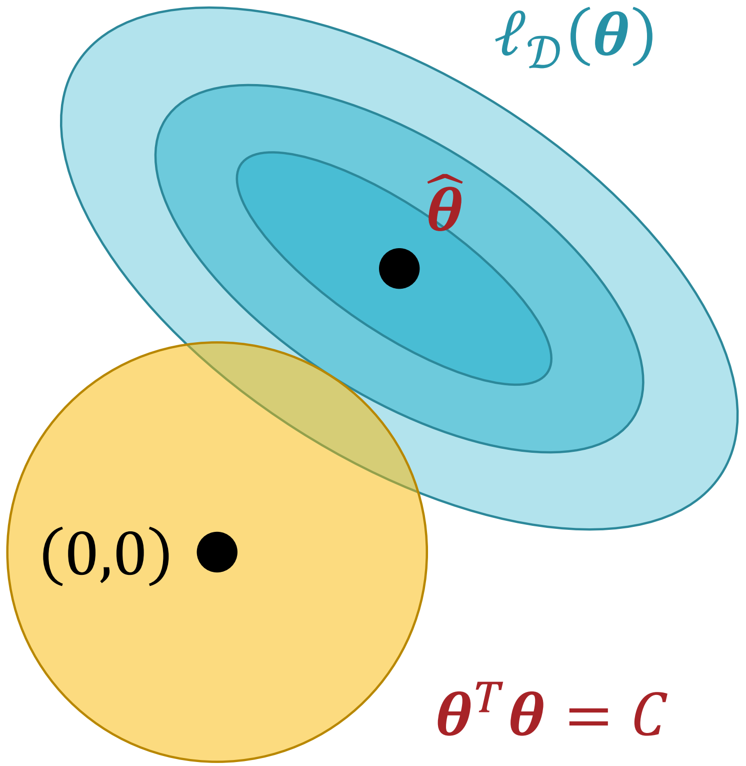
- Subject to:

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

Soft Constraints

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

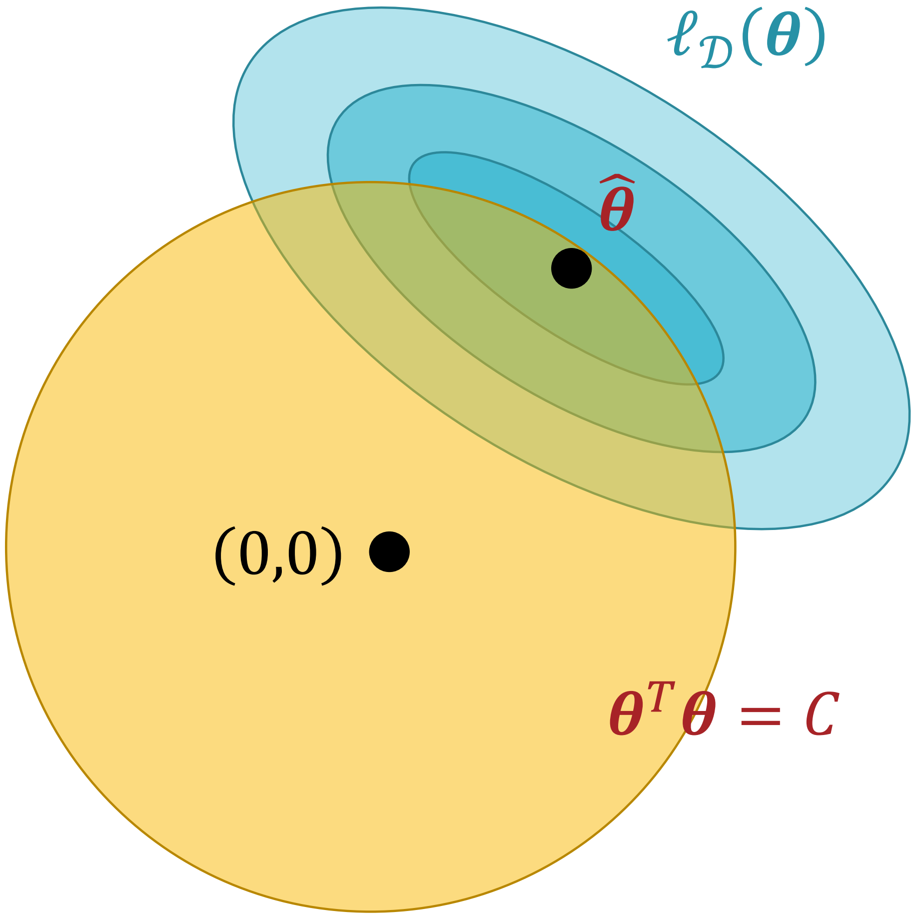
subject to $\boldsymbol{\theta}^T\boldsymbol{\theta} \leq C$



Soft Constraints

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

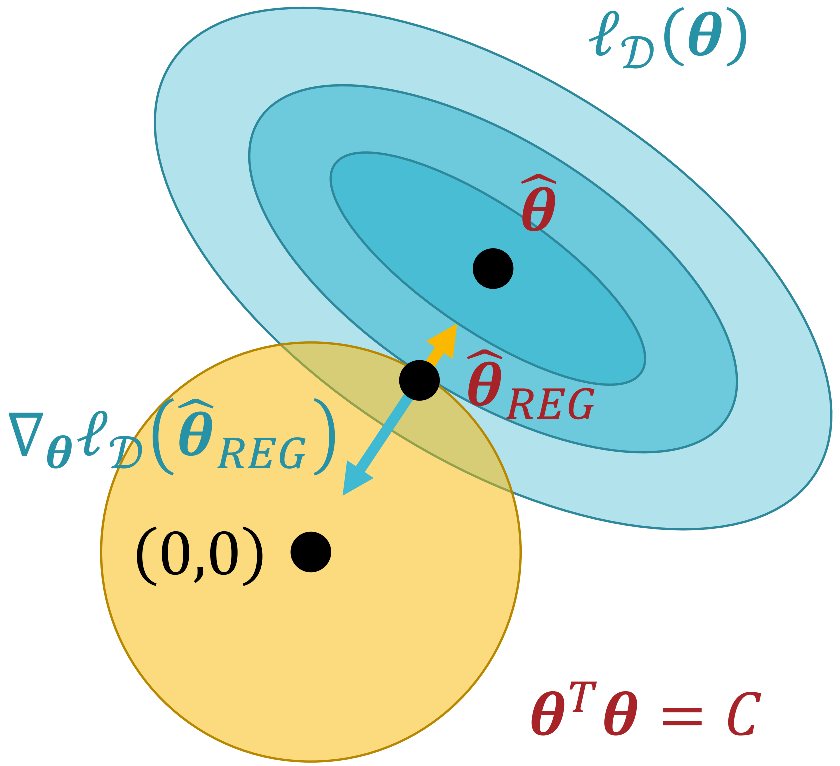
subject to $\boldsymbol{\theta}^T\boldsymbol{\theta} \le C$



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 $\widehat{\boldsymbol{\theta}}_{REG}$

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

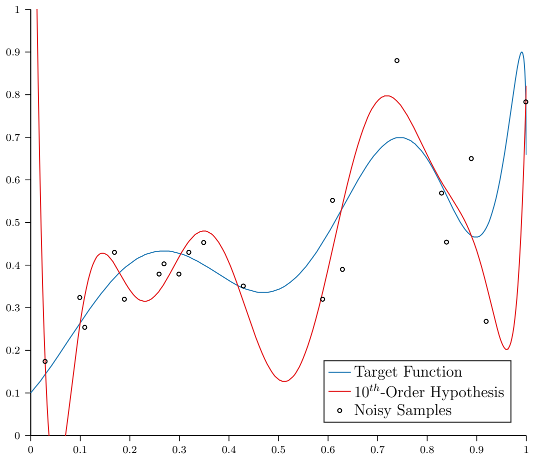
Ridge Regression

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

Regularization: Q & A

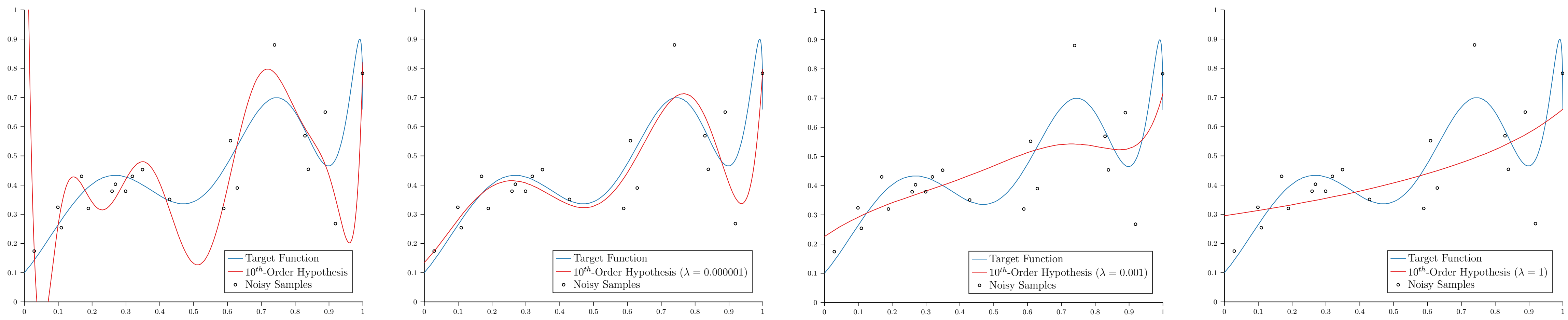
- Should we regularize the bias/intercept parameter, θ_0?

- Is feature scale a concern with regularization?



Ridge Regression

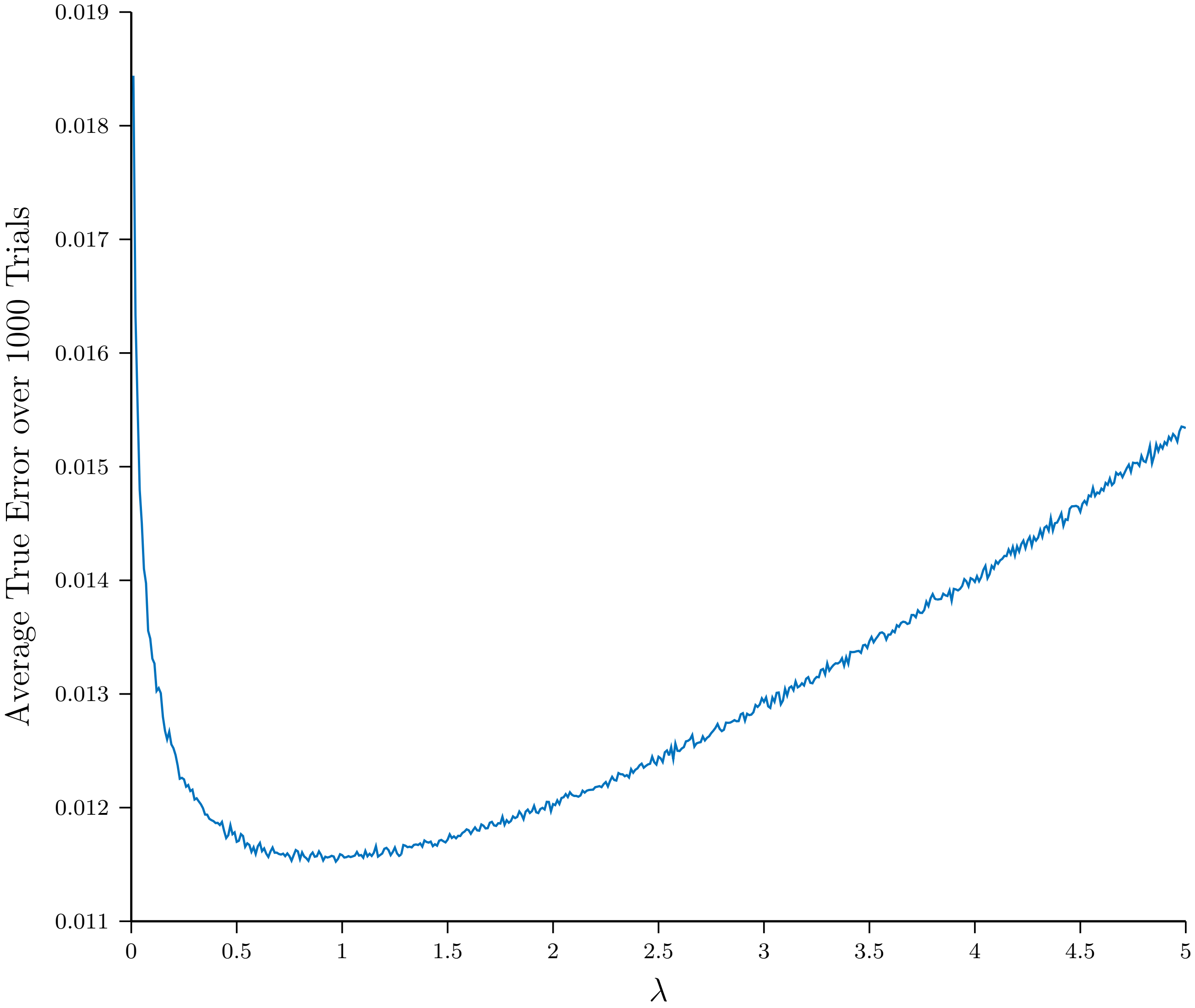
- 10-dimensional target function with additive Gaussian noise
- $\mathcal{H}_{10} = 10^{\text{th}}$-order polynomial



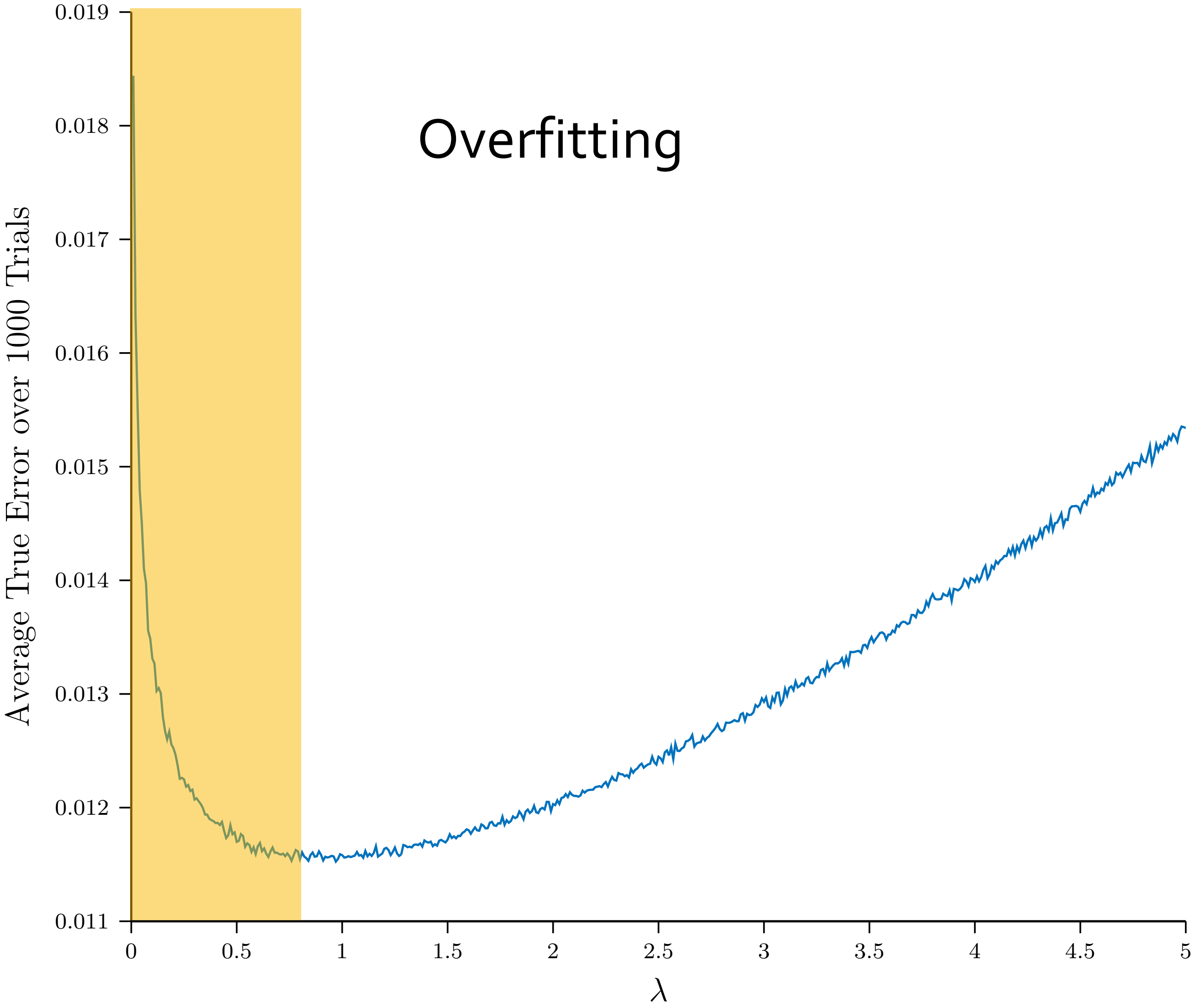
Ridge Regression

	$\lambda_C = 0$
True Error	0.059
	Overfit

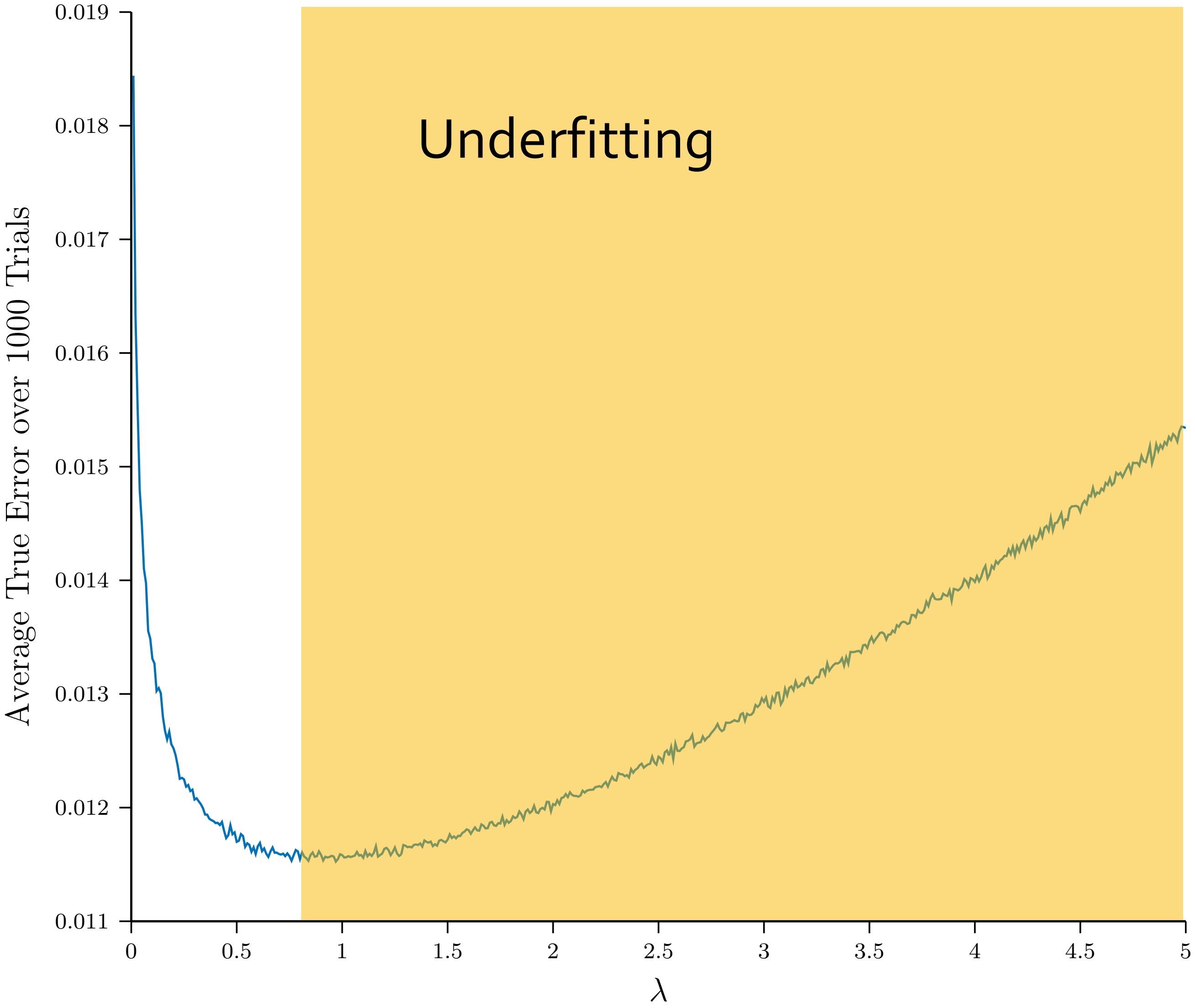
Setting λ



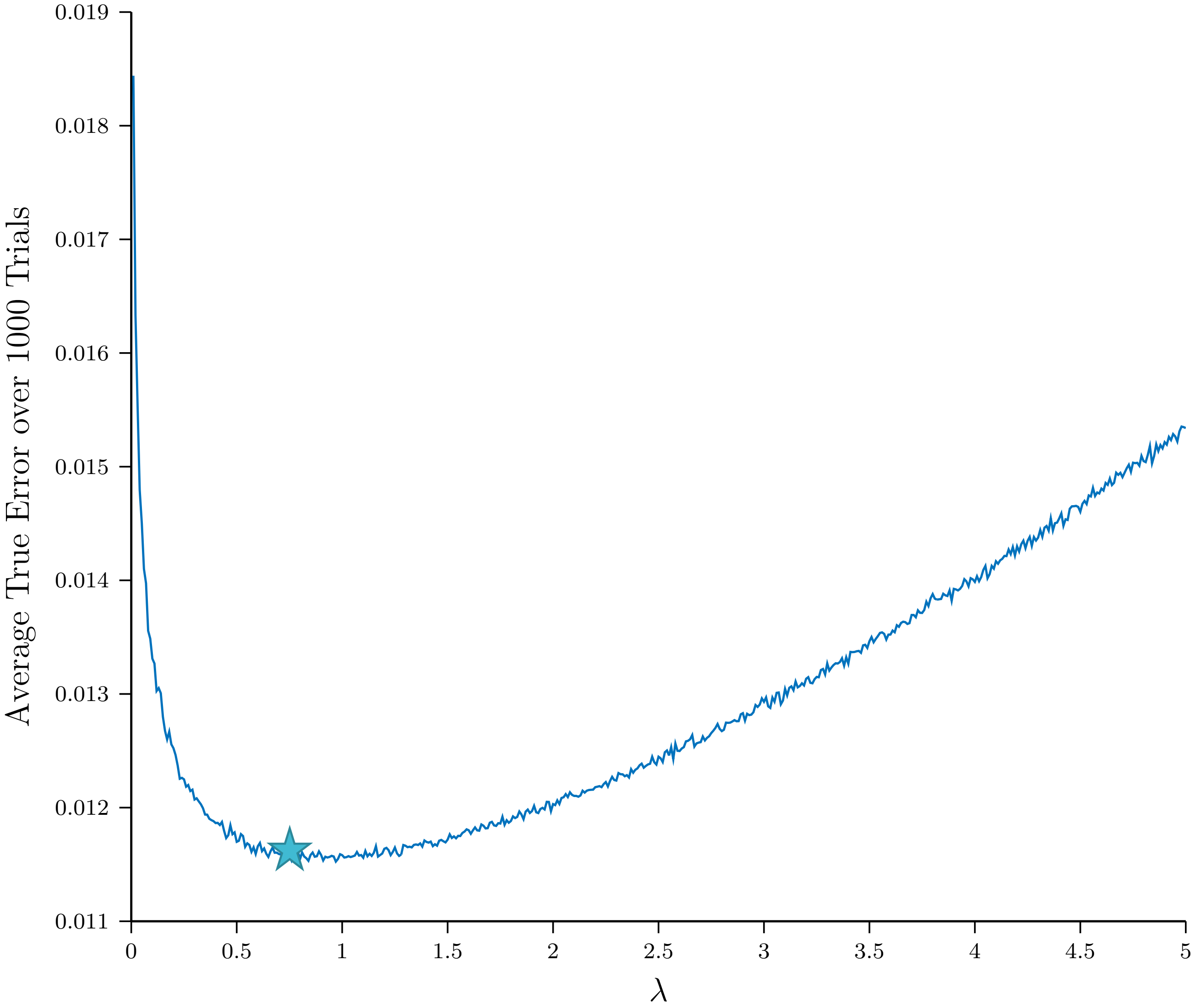
Setting λ



Setting λ

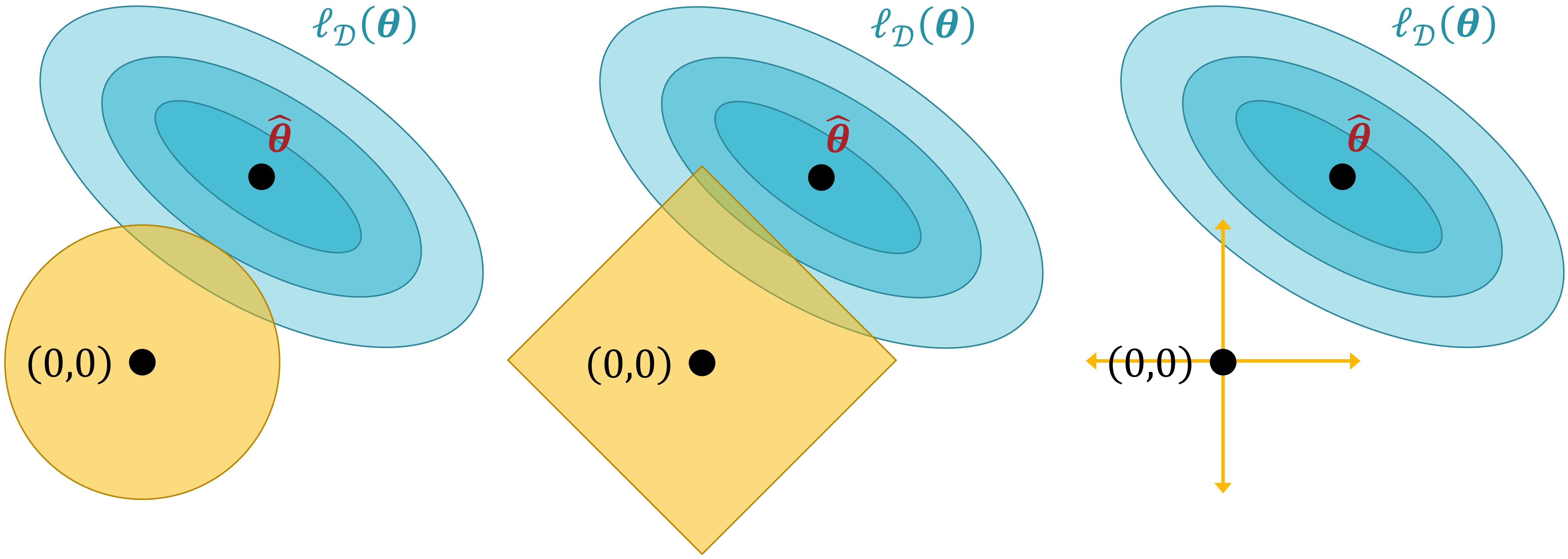


Setting λ



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 $L1$	$r(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_1 = \sum_{d=0}^{D} \|\theta_d\|$	Encourages sparsity
$L0$	$r(\boldsymbol{\theta}) = \|\boldsymbol{\theta}\|_0 = \sum_{d=0}^{D} \mathbb{1}(\theta_d \neq 0)$	Encourages sparsity (intractable)



Ridge or $L2$ | Lasso or $L1$ | $L0$

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