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

Pradeep Ravikumar

Spring 2026

Front Matter

- Announcements:
  - Homework 2 due tomorrow
  - Quiz 2 on Friday
- Recommended Readings:
  - Murphy, Sections 7.5 & 14.4

Feature Transforms: Experiment

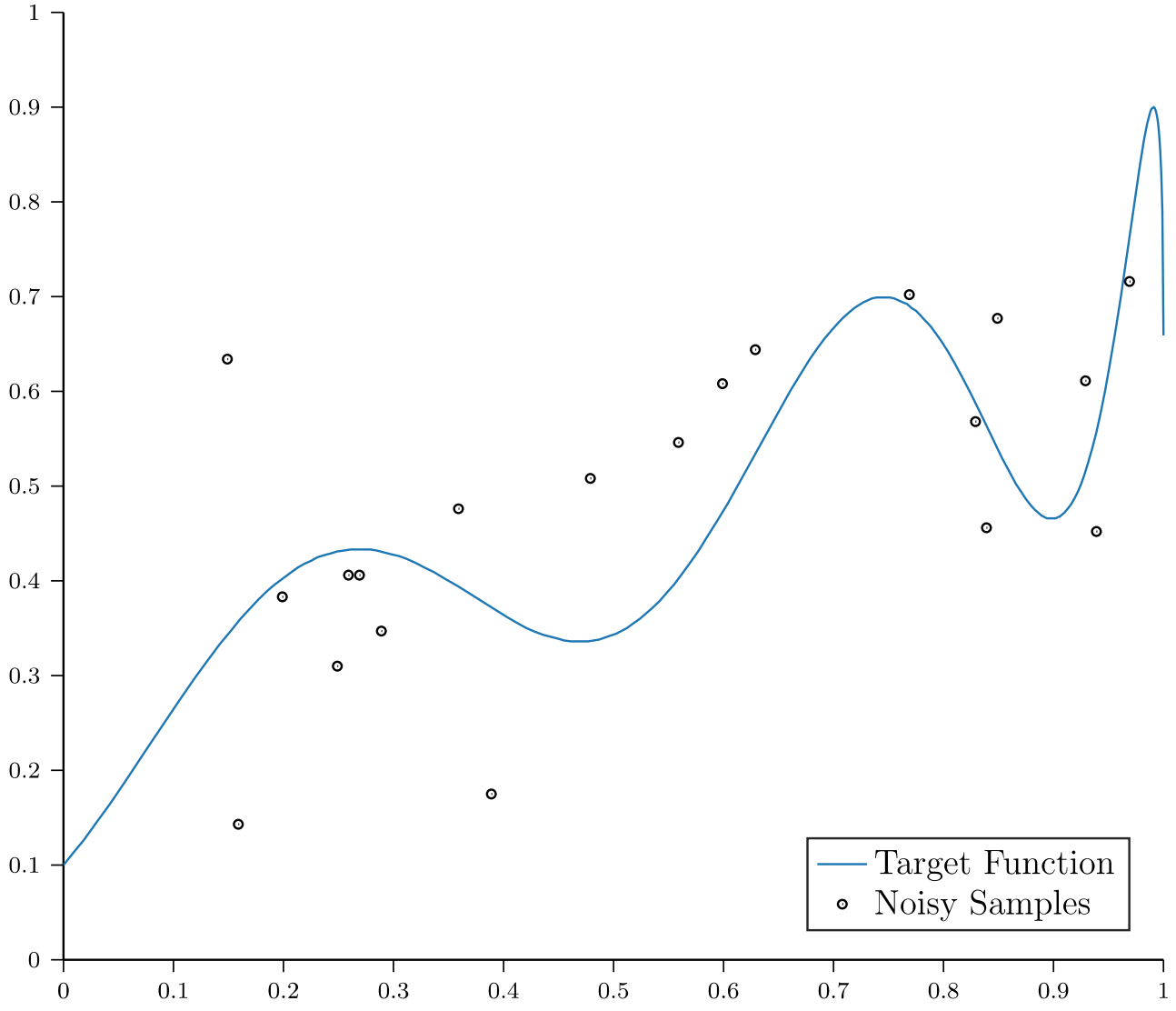
- $x \in \mathbb{R}$, $y \in \mathbb{R}$ and $N = 20$
- Targets are generated by a 10^{th}-order polynomial in x with additive Gaussian noise:

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

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

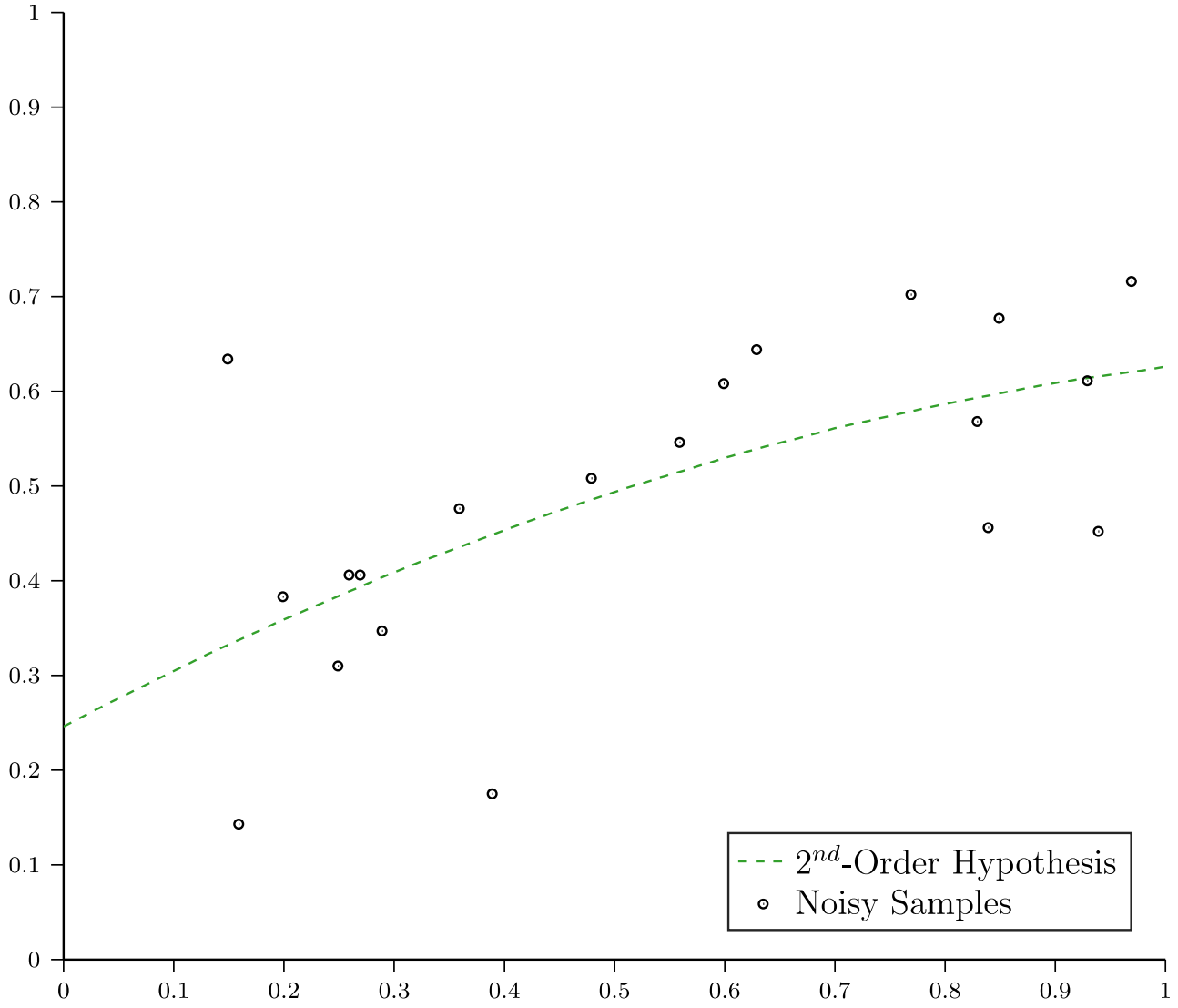
Noisy Targets

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



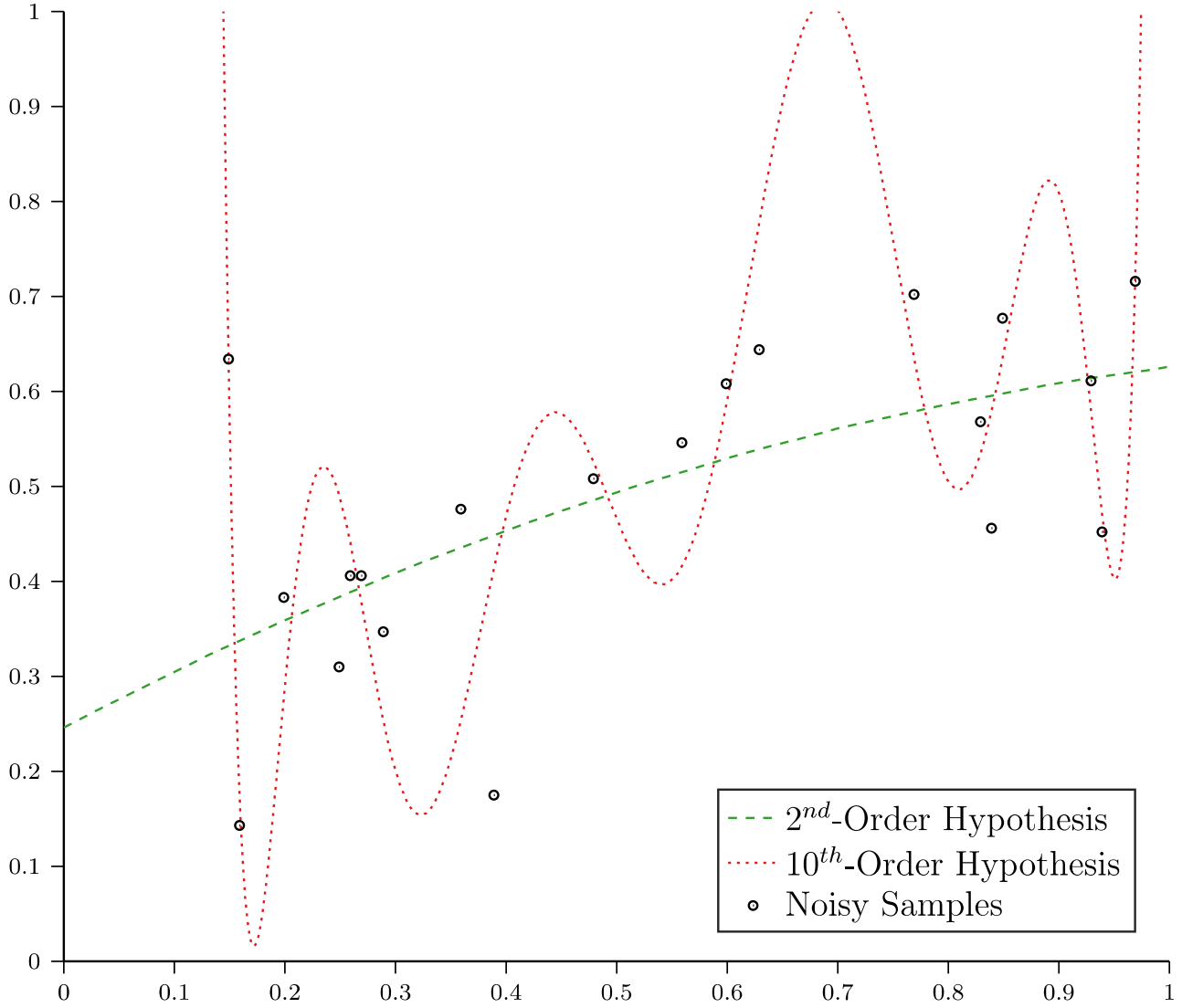
Noisy Targets

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



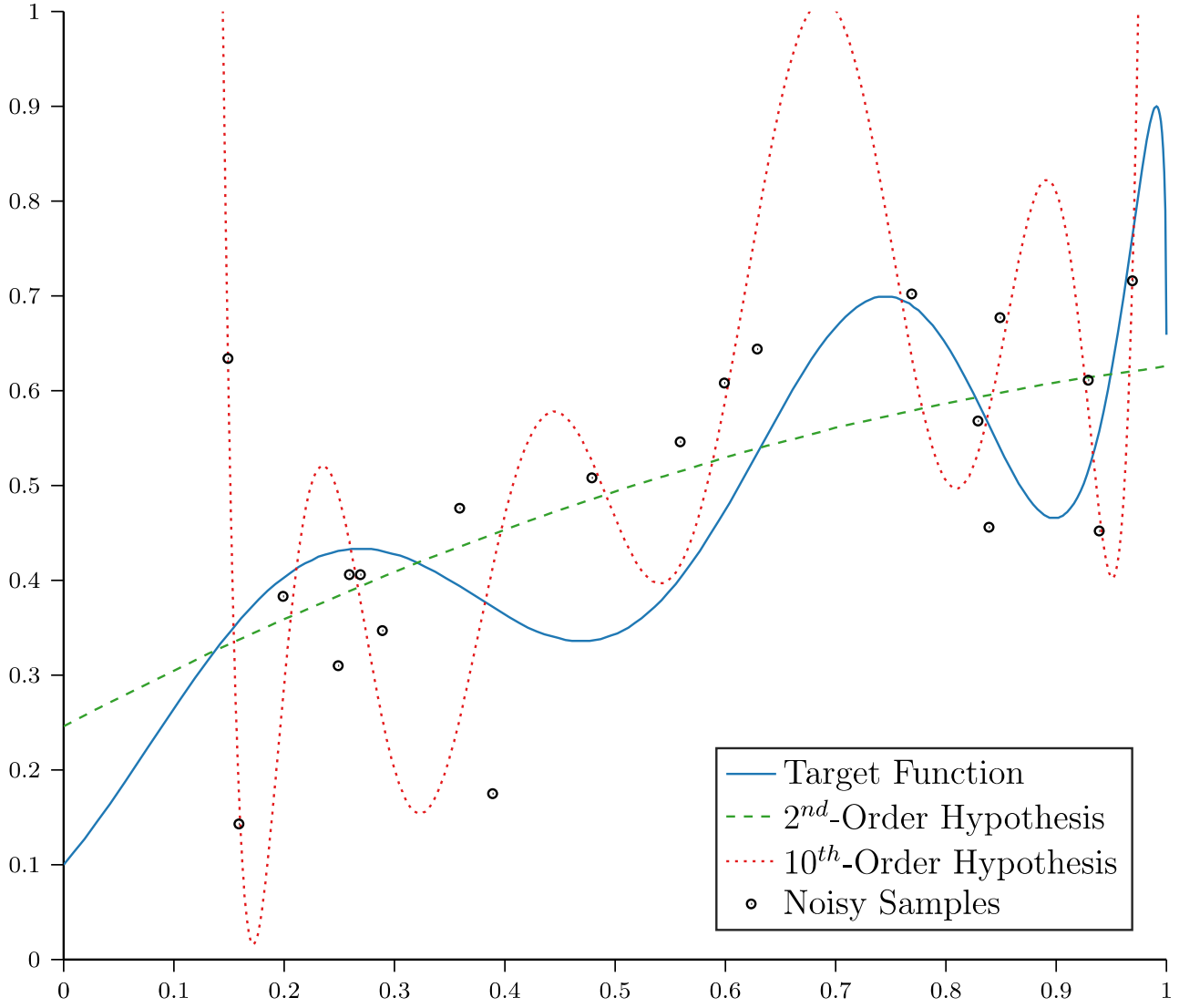
Noisy Targets

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



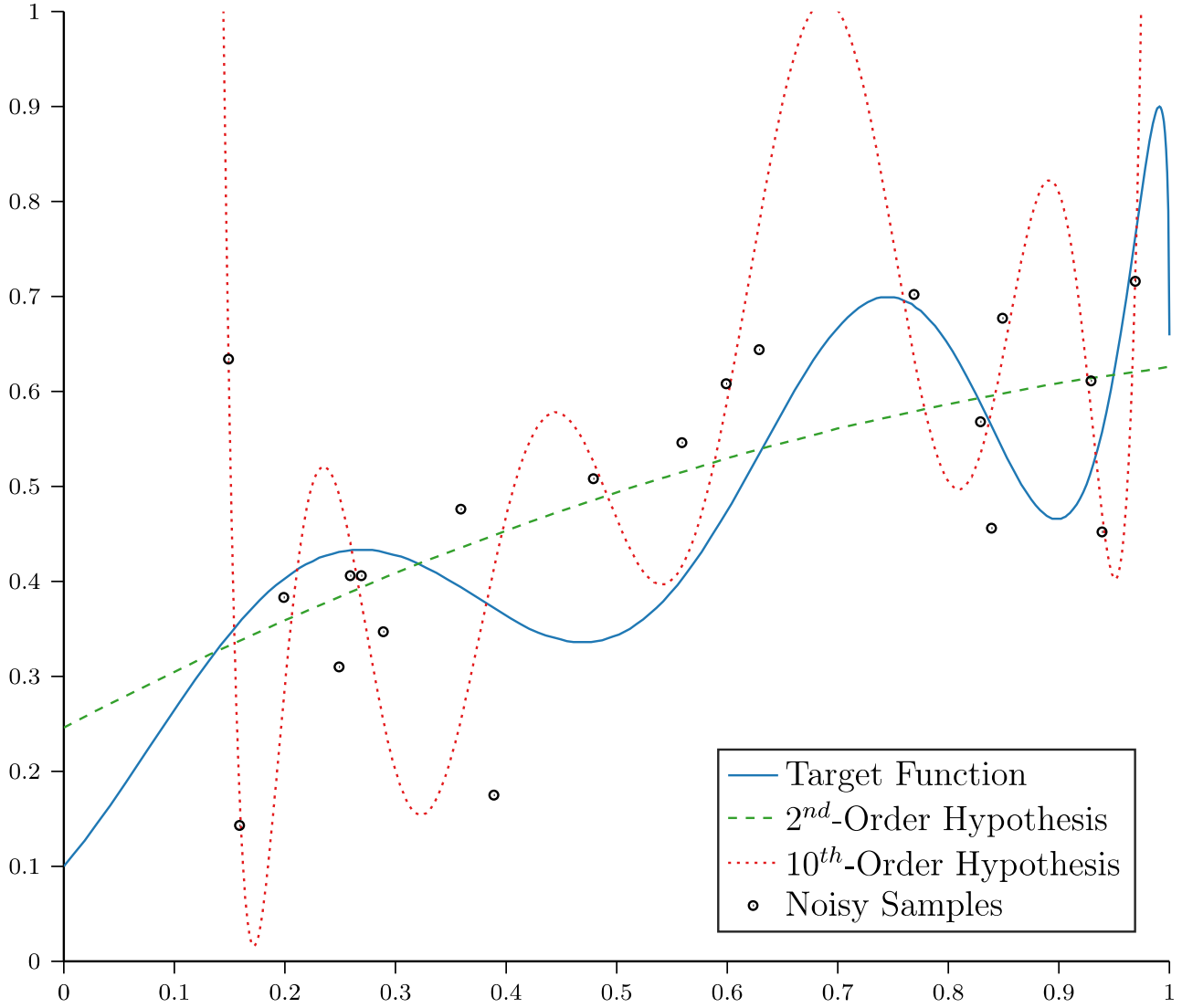
Noisy Targets

- 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



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

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

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

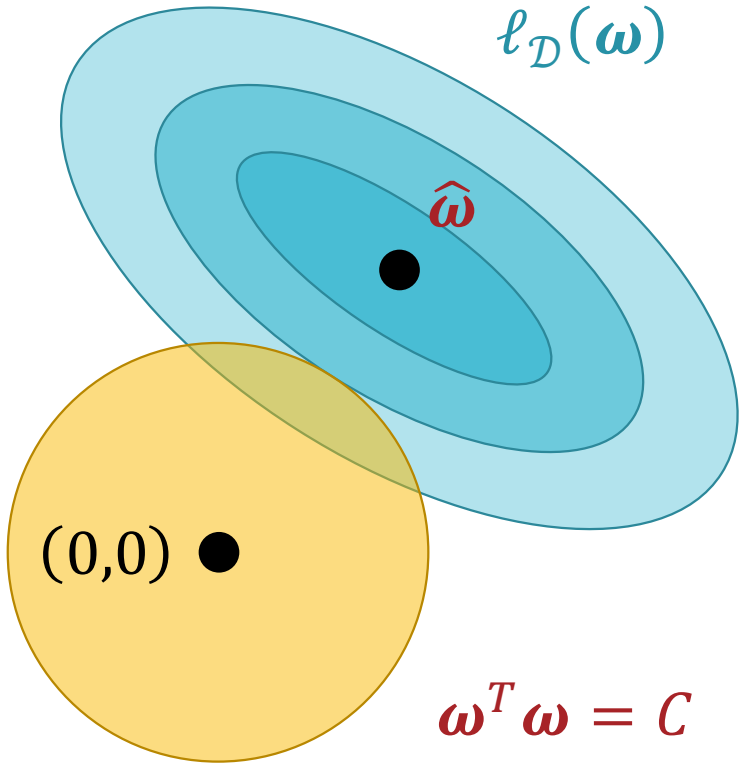
- Subject to

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

Soft Constraints

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

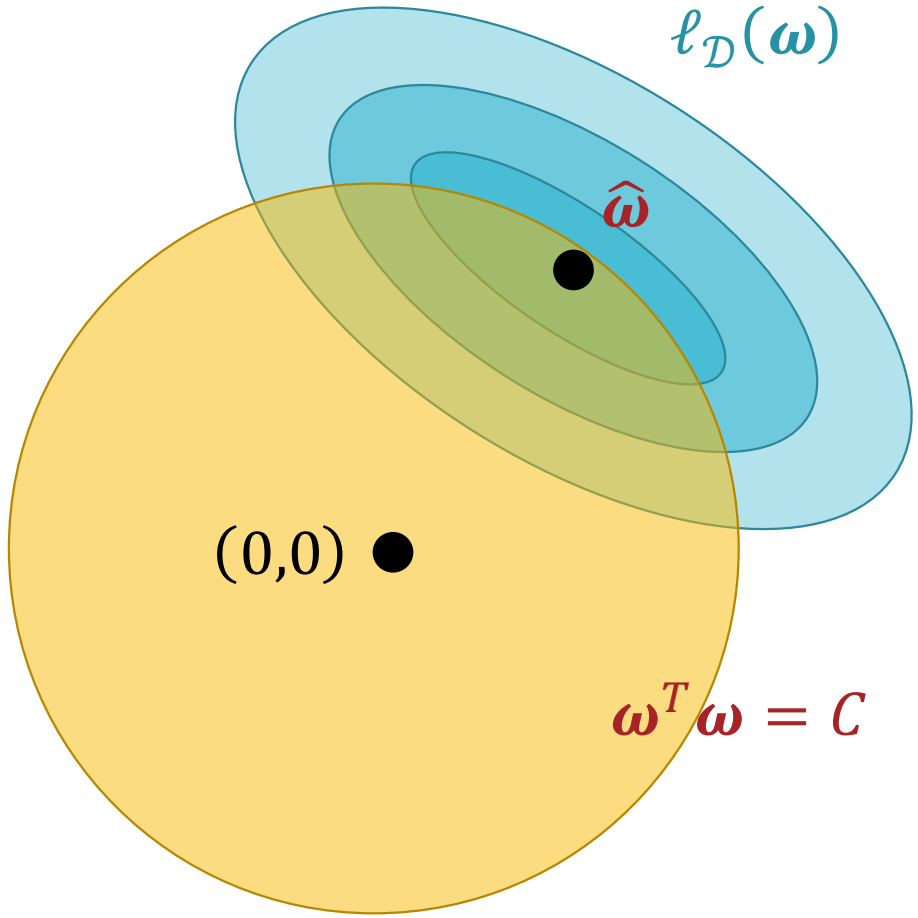
subject to $\boldsymbol{\omega}^T\boldsymbol{\omega} \leq C$

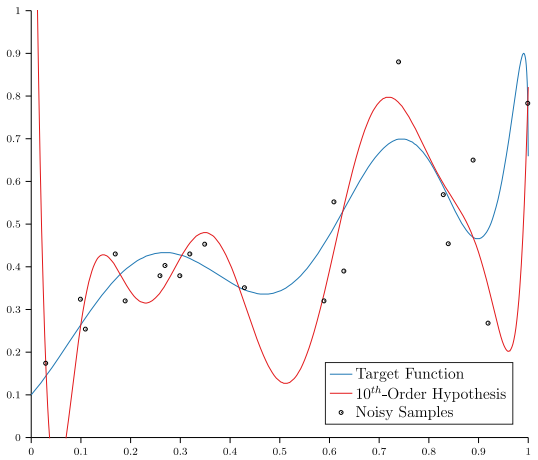


Soft Constraints

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

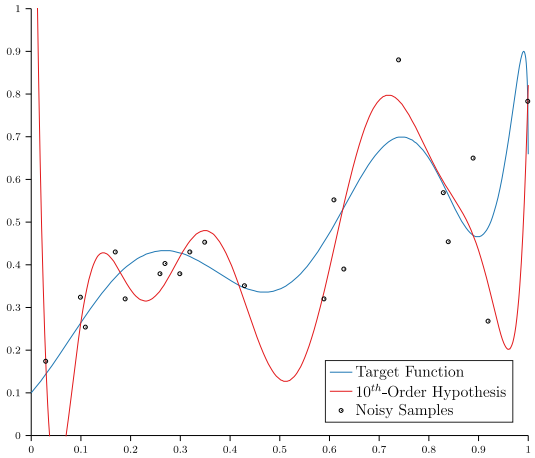
subject to $\boldsymbol{\omega}^T\boldsymbol{\omega} \leq C$

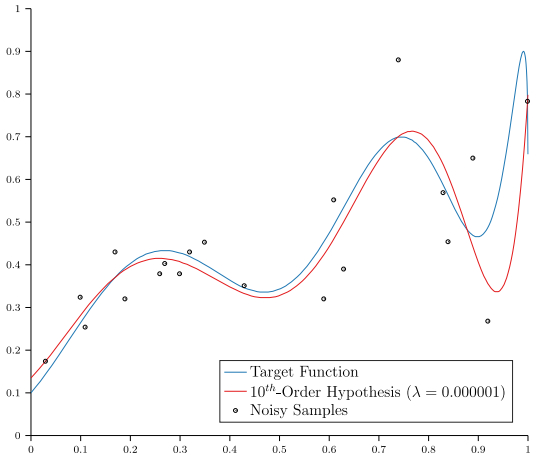


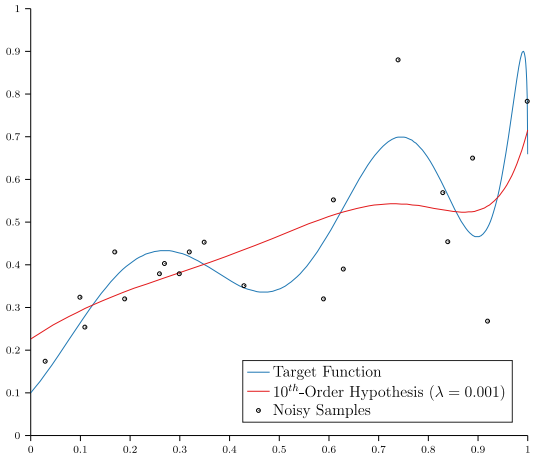


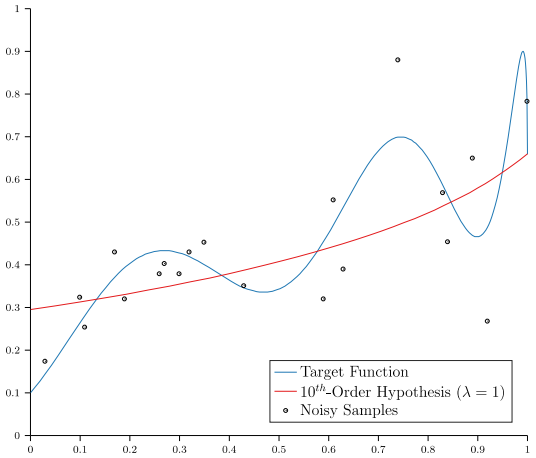
Ridge Regression

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









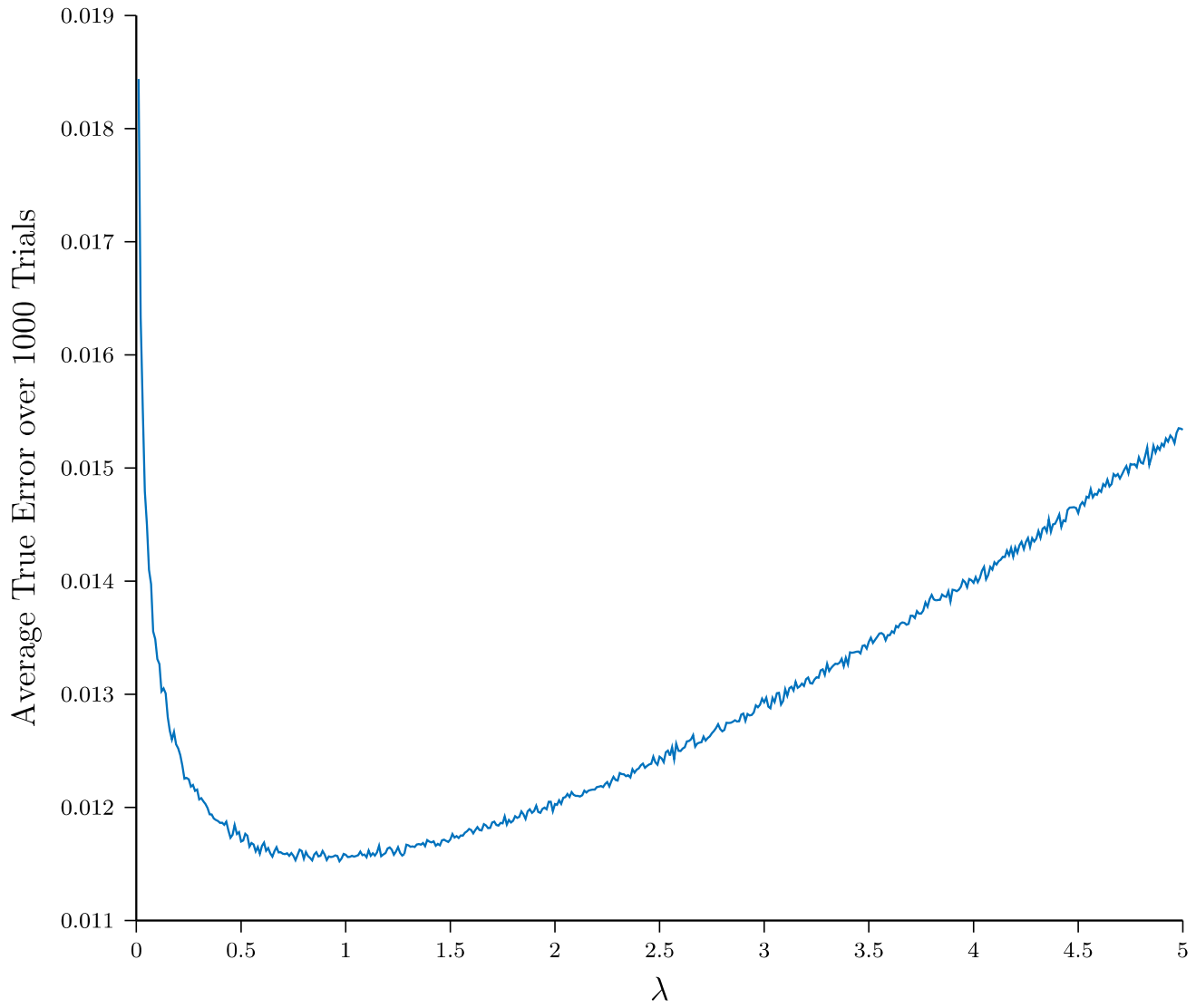
Ridge Regression

$\boldsymbol{\lambda_C = 0}$

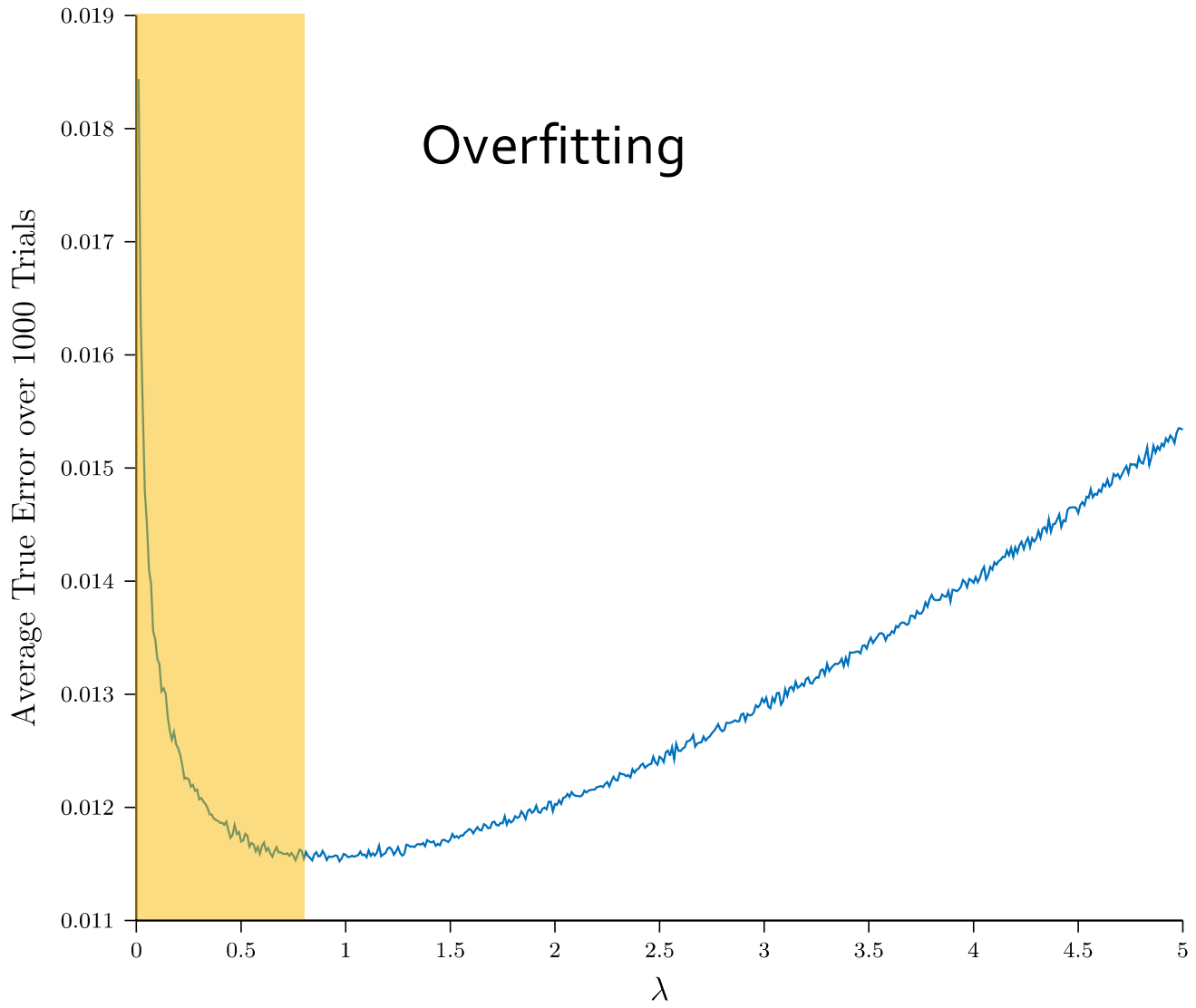
True Error	0.059

Overfit

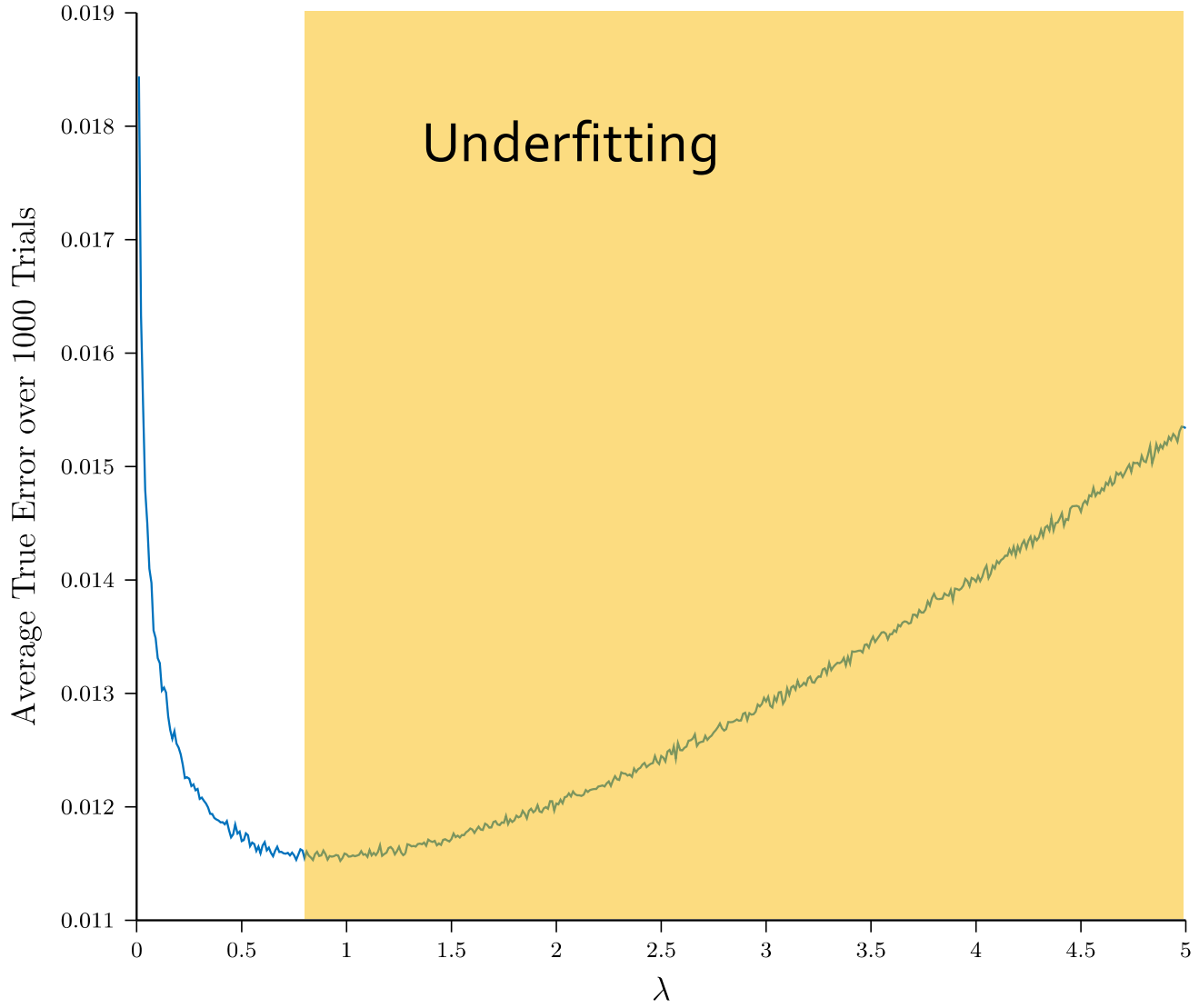
Setting λ



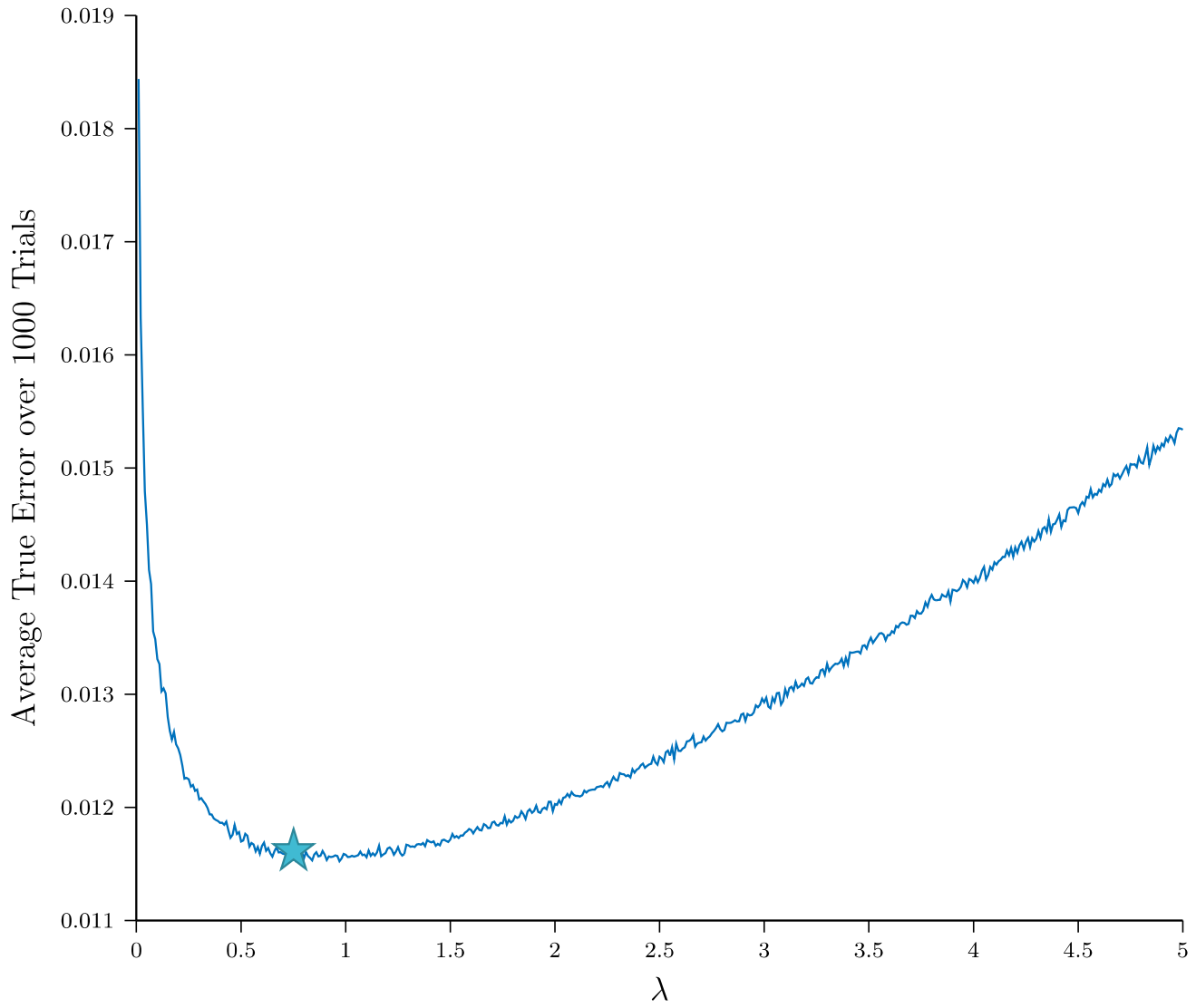
Setting λ



Setting λ



Setting λ



True Risk vs. Empirical Risk

True Risk: Target performance measure

Classification – Probability of misclassification $P(f(X) \neq Y)$

Regression – Mean Squared Error $\mathbb{E}[(f(X) - Y)^2]$

Expected performance on a random test point (X,Y)

True Risk vs. Empirical Risk

True Risk: Target performance measure

Classification – Probability of misclassification $P(f(X) \neq Y)$

Regression – Mean Squared Error $\mathbb{E}[(f(X) - Y)^2]$

Expected performance on a random test point (X,Y)

Empirical Risk: Performance on training data

Classification – Proportion of misclassified examples $\frac{1}{n}\sum_{i=1}^{n} 1_{f(X_i)\neq Y_i}$

Regression – Average Squared Error $\frac{1}{n}\sum_{i=1}^{n}(f(X_i) - Y_i)^2$

MythBusters

MythBusters #1

“Zero training error means the model is great.”

Myth: If empirical risk is tiny, true risk must be tiny too.

Reality: We minimize empirical risk (on training data) but care about true risk (on new samples). With high complexity, ERM can overfit: lo training error, high test error.

Key phrase: “mismatch between empirical and true risk.”

Some quick notation

$$\text{True Risk} : R(f) := \mathbb{E}(\ell(f(X), Y))$$

$$\text{Empirical Risk given data D} : \widehat{R}_D(f) := \frac{1}{|D|} \sum_{i \in D} \ell(f(X_i), Y_i)$$

True Risk vs Empirical Risk

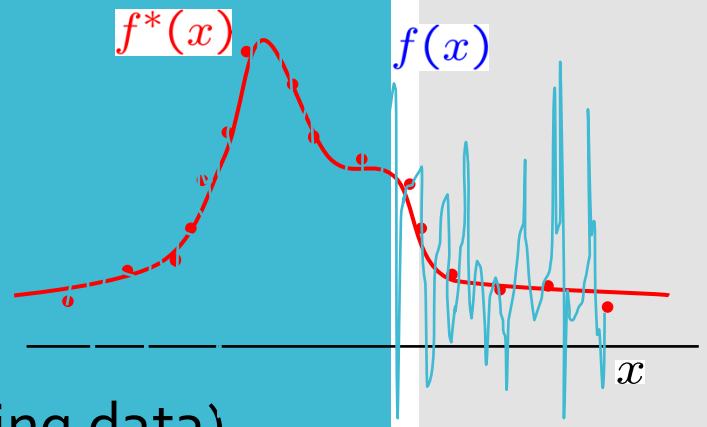
- So we minimize with respect to empirical risk
- And evaluate with respect to true risk

- Is there any danger to this mismatch?
  - Overfitting!!

Overfitting

Is the following predictor a good one?

$$f(x) = \begin{cases} Y_i, & x = X_i \text{ for } i = 1, \ldots, n \\ \text{any value}, & \text{otherwise} \end{cases}$$



What is its empirical risk? (performance on training data)

zero !

What about true risk?

>> zero

Will predict very poorly on new random test point:

Large generalization error !

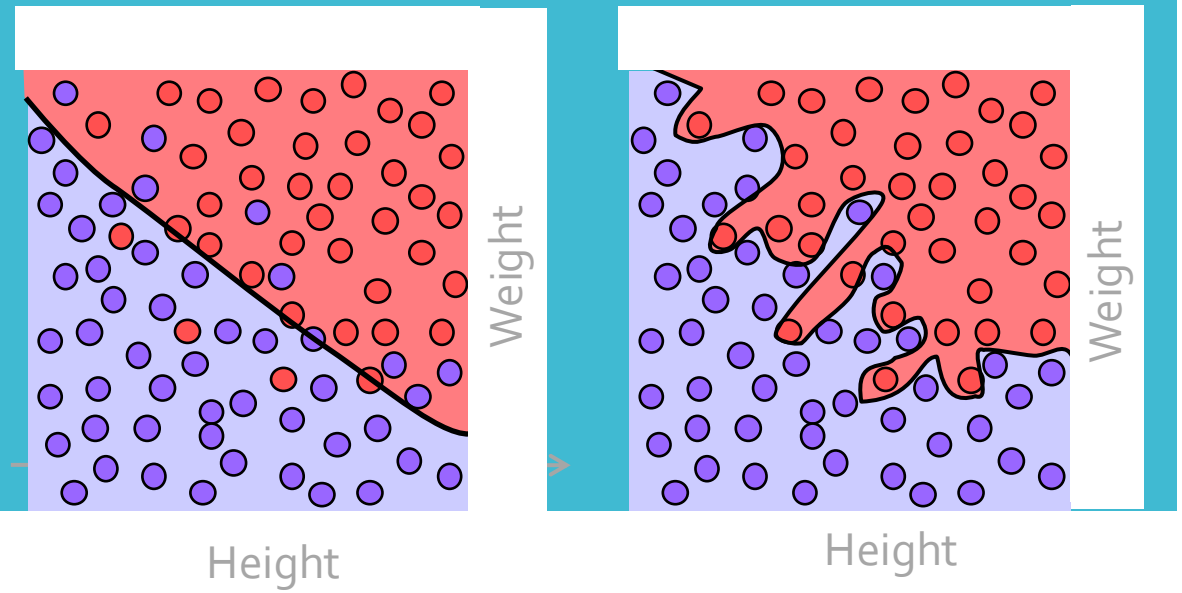
Overfitting

If we allow very complicated predictors, we could overfit the training data.

Examples: Classification

Football player ?

- No
- Yes

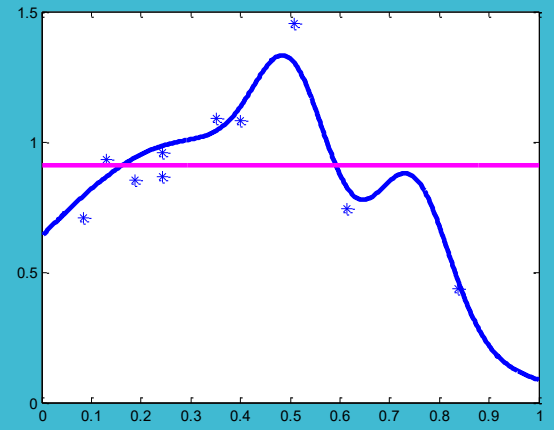


Overfitting

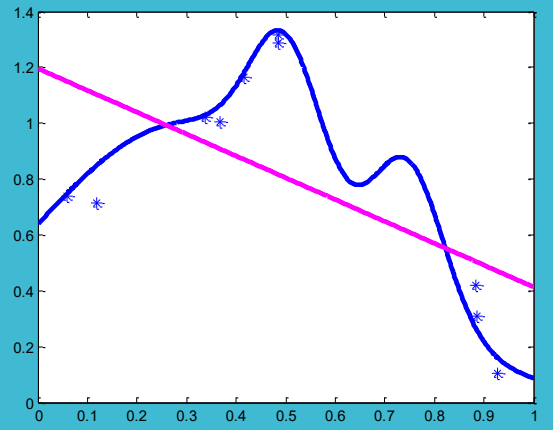
If we allow very complicated predictors, we could overfit the training data.

Examples: Regression (Polynomial of order k – degree up to k-1)

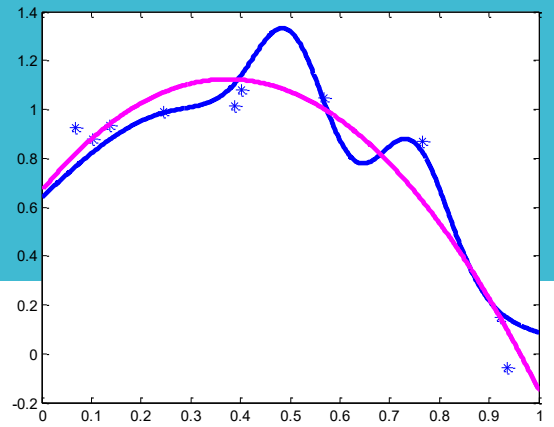
k=1



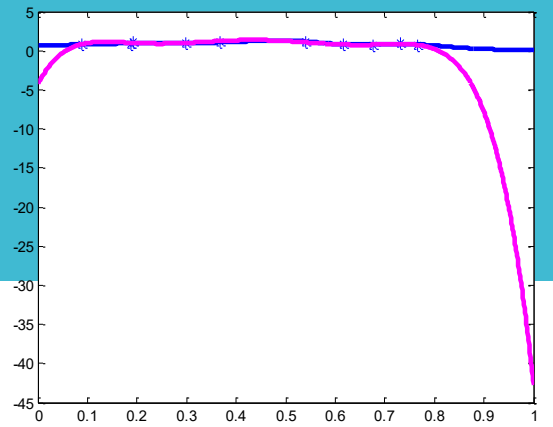
k=2



k=3



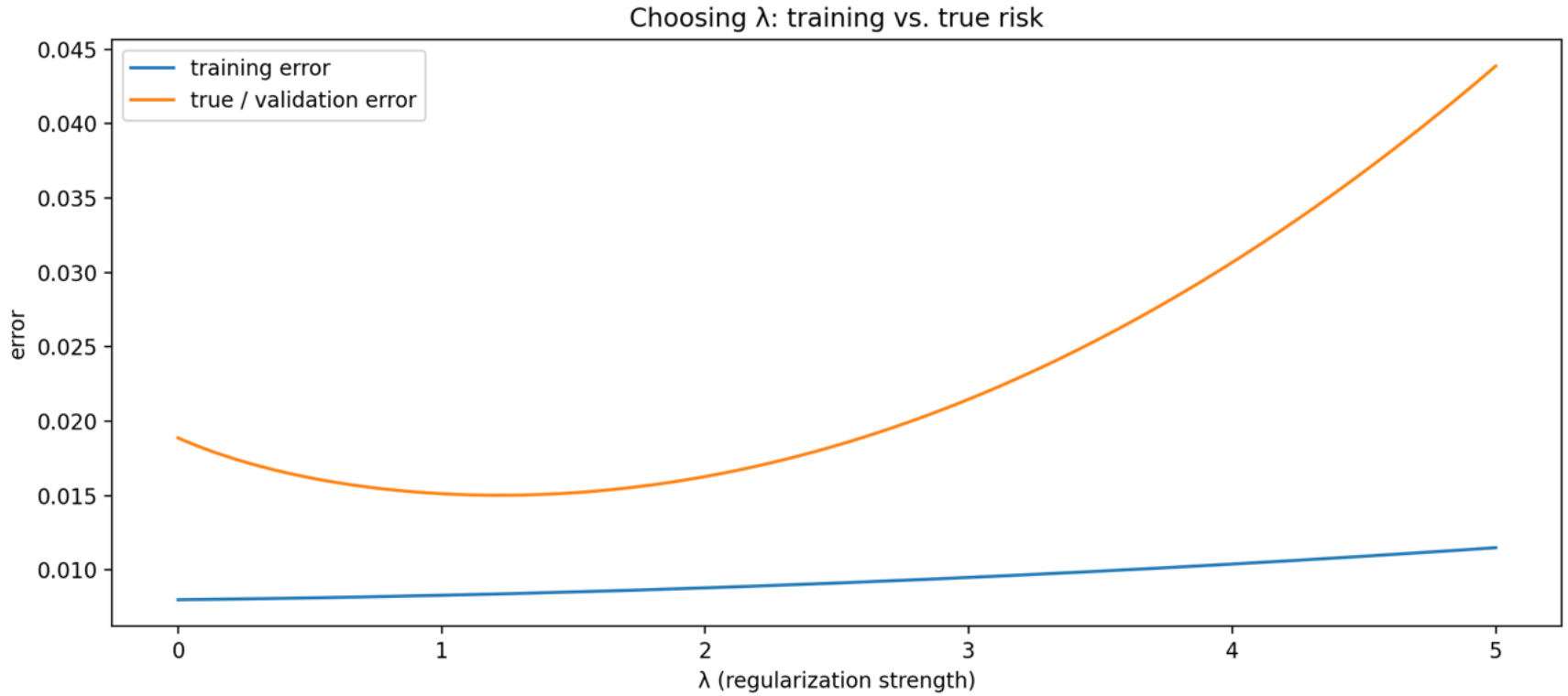
k=7



Quick Game

Quick game: pick λ

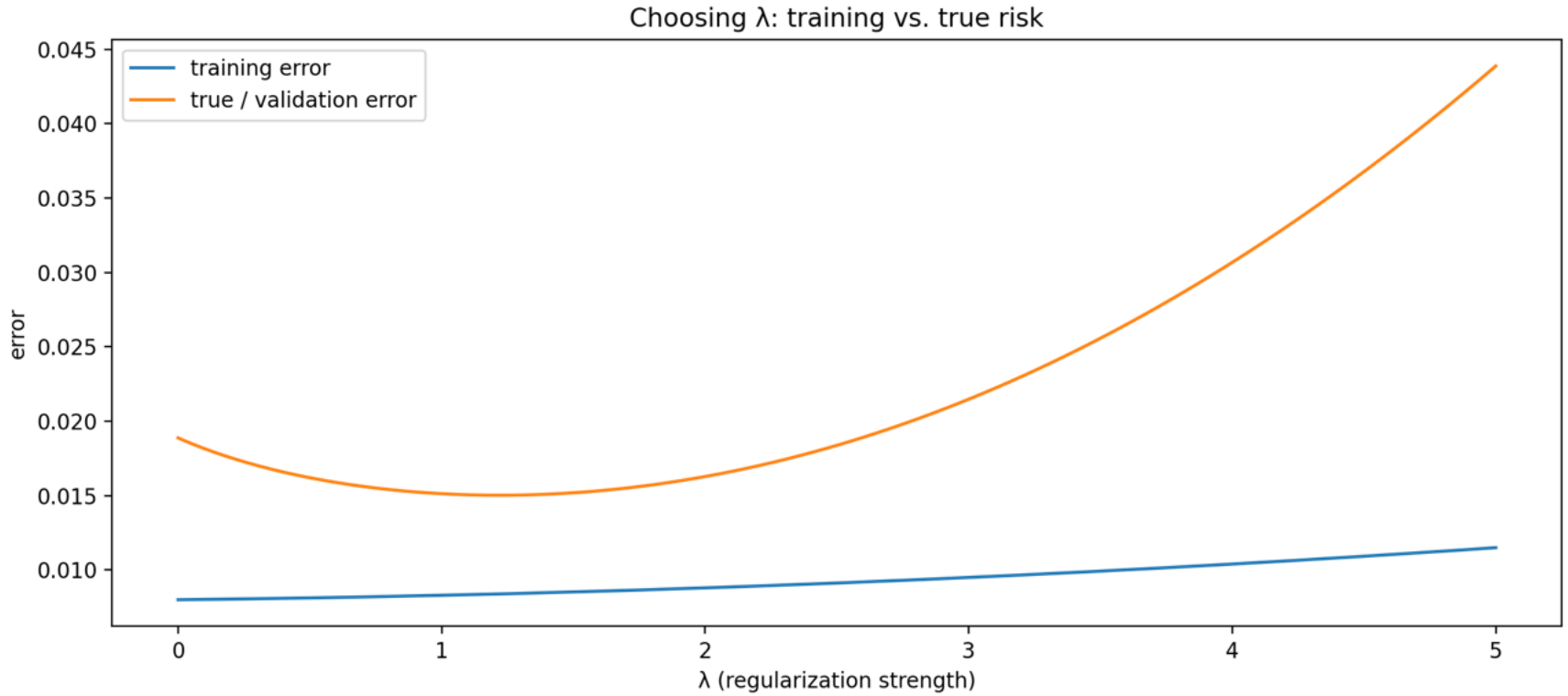
Where would you choose λ to get the best generalization?



Quick Game

Quick game: pick λ

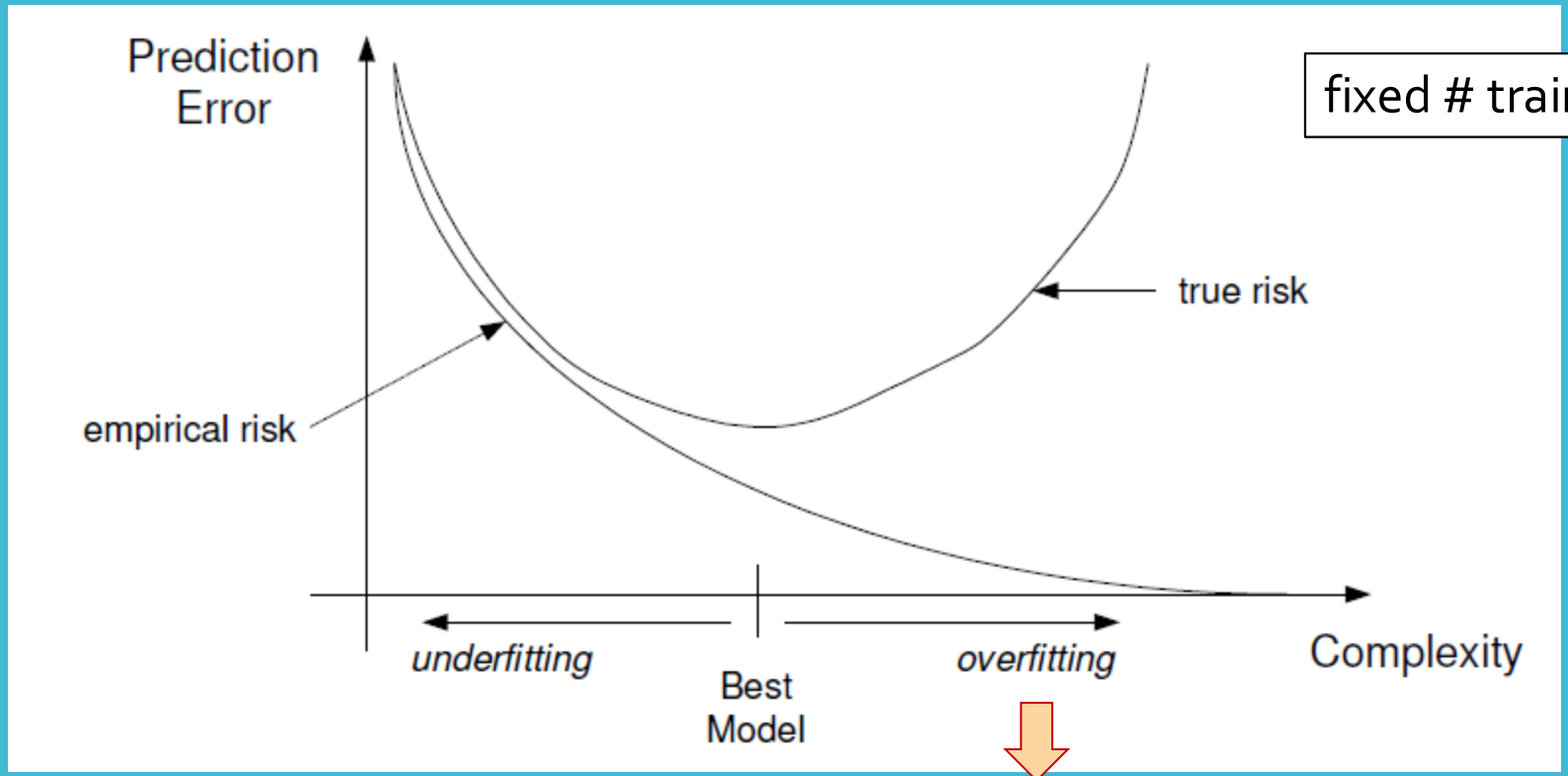
Where would you choose λ to get the best generalization?



Answer: near the minimum of the validation/true-error curve.

Overfitting: Effect of discrepancy between empirical and true risks

If we allow very complicated predictors, we could overfit the training data.



Empirical risk is no longer a good indicator of true risk

Questions

- So, Empirical risk minimization (ERM) might “overfit” when the model complexity is high, due to mismatch between empirical risk and true risk
- But we do not have access to true risk since it depends on unknown distribution :(
- And so we estimate true risk via empirical risk!
- **Can we do better?**

Structural Risk Minimization

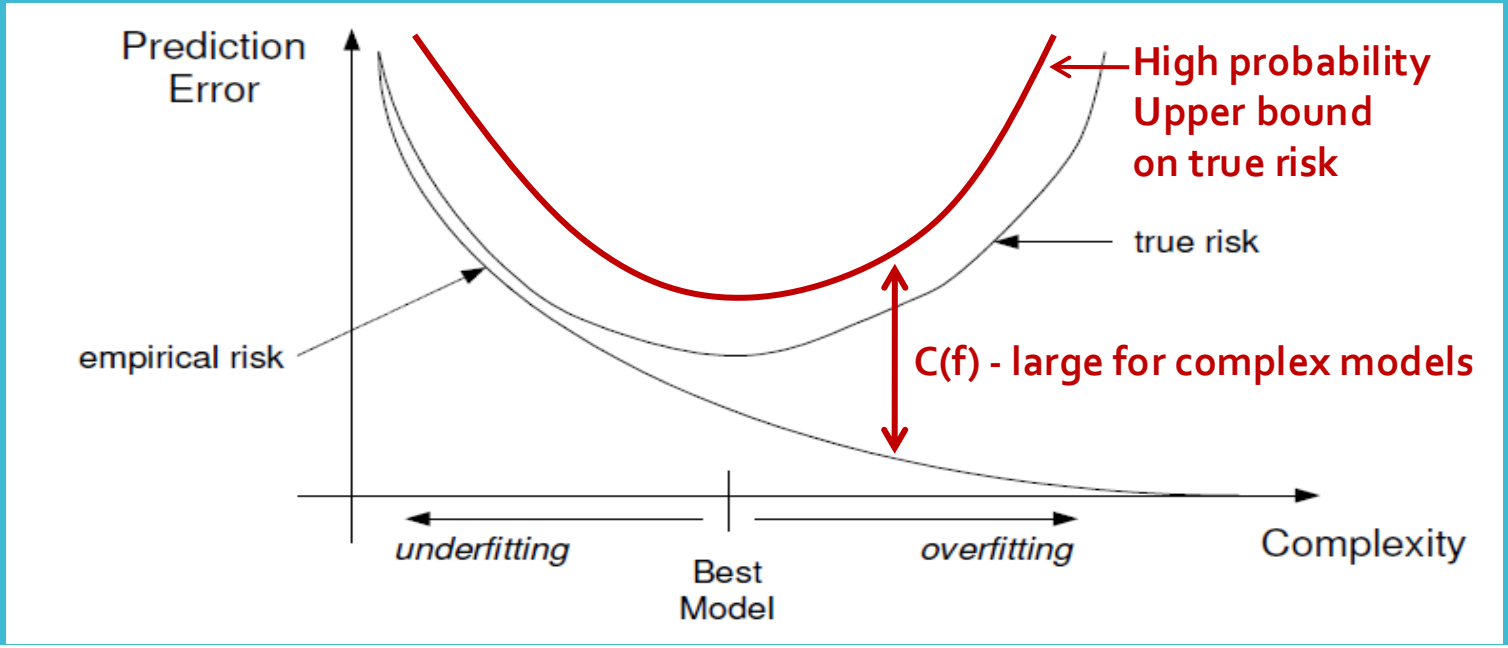
Penalize models using bound on **deviation of true and empirical risks**.

$$\hat{f}_n = \arg\min_{f \in \mathcal{F}} \left\{ \hat{R}_n(f) + C(f) \right\}$$

Bound on deviation from true risk

With high probability, $|R(f) - \hat{R}_n(f)| \leq C(f) \quad \forall f \in \mathcal{F}$

Concentration bounds (later)



Structural Risk Minimization

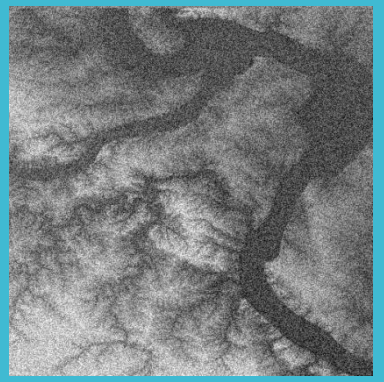
Deviation bounds are typically pretty loose, for small sample sizes. In practice,

$$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \left\{ \widehat{R}_n(f) + \lambda \mathbb{C}(f) \right\}$$

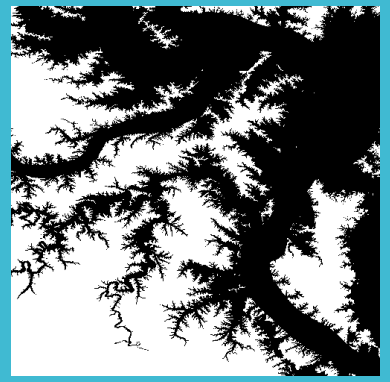
Choose by **model selection**!

Problem: Identify flood plain from noisy satellite images

Noiseless image



Noisy image



True Flood plain
(elevation level > x)

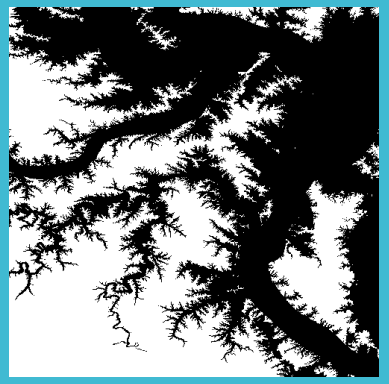
Structural Risk Minimization

Deviation bounds are typically pretty loose, for small sample sizes. In practice,

$$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \left\{ \widehat{R}_n(f) + \lambda \mathbb{C}(f) \right\}$$

Choose by **model selection!**

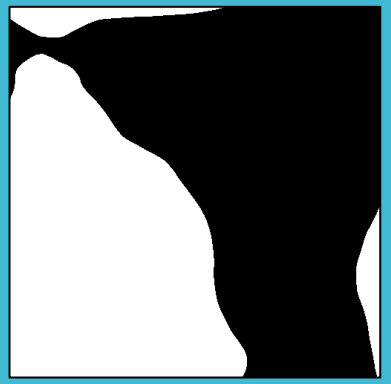
Problem: Identify flood plain from noisy satellite images



True Flood plain
(elevation level > x)

Zero penalty

CV penalty

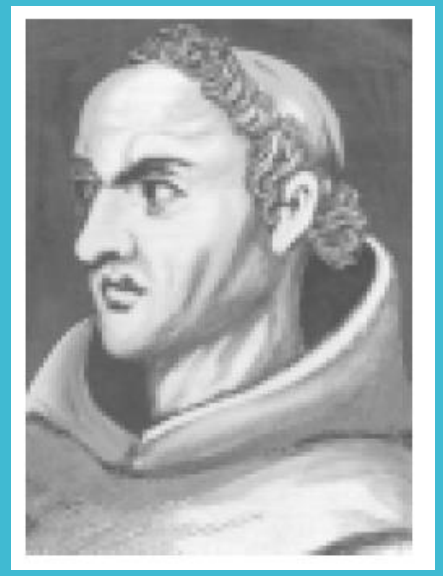


Theoretical penalty

Occam's Razor

William of Ockham (1285-1349) *Principle of Parsimony:*

"One should not increase, beyond what is necessary, the number of entities required to explain anything."

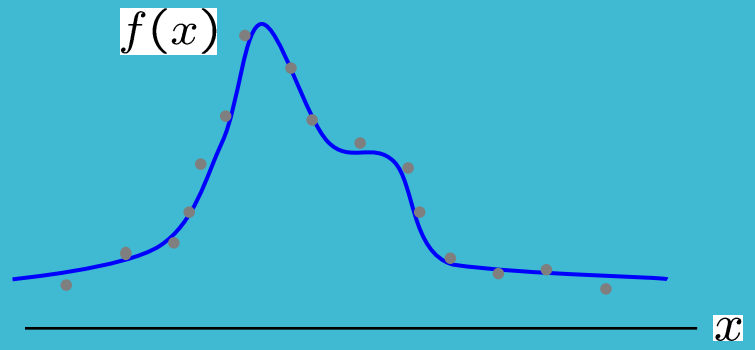


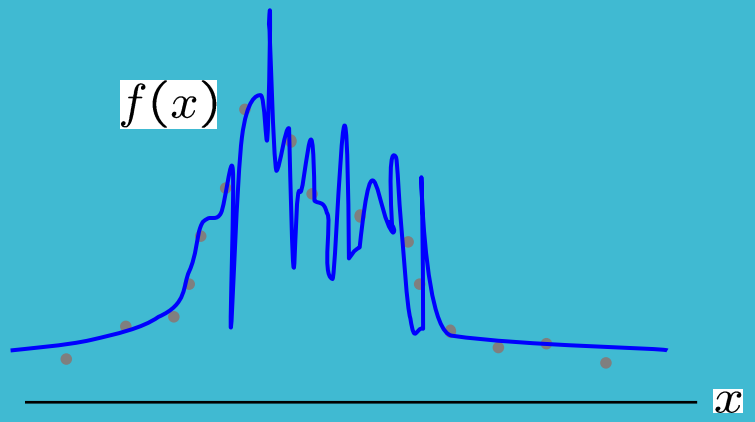
Alternatively, seek the simplest explanation.

Penalize complex models based on

- Prior information (bias)
- Information Criterion (MDL, AIC, BIC)

Importance of Domain Knowledge





Distribution of photon arrivals

Oil Spill Contamination

Compton Gamma-Ray Observatory Burst and Transient Source Experiment (BATSE)

Complexity Regularization

Penalize complex models using **prior knowledge**.

$$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \left\{ \widehat{R}_n(f) + C(f) \right\}$$

Cost of model (log prior)

Bayesian viewpoint:

prior probability of *f*, p(*f*) ≡ $e^{-C(f)}$

cost is small if *f* is highly probable, cost is large if *f* is improbable

ERM (empirical risk minimization) over a restricted class *F*

≡ uniform prior on $f \in F$, zero probability for other predictors

$$\widehat{f}_n^L = \arg\min_{f \in \mathcal{F}_L} \widehat{R}_n(f)$$

Complexity Regularization

Penalize complex models using **prior knowledge**.

$$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \left\{ \widehat{R}_n(f) + C(f) \right\}$$

Cost of model (log prior)

Examples: MAP estimators

Regularized Linear Regression - Ridge Regression, Lasso

$$\widehat{\theta}_{\text{MAP}} = \arg\max_{\theta} \log p(D|\theta) + \log p(\theta)$$

$$\widehat{\beta}_{\text{MAP}} = \arg\min_{\beta} \sum_{i=1}^{n} (Y_i - X_i\beta)^2 + \lambda\|\beta\|$$

Penalize models based on some norm of regression coefficients

How to choose tuning parameter λ? **Model Selection**

Information Criteria – AIC, BIC

Penalize complex models based on their **information content**.

$$\widehat{f}_n = \arg\min_{f \in \mathcal{F}} \left\{ \widehat{R}_n(f) + C(f) \right\}$$

→ # bits needed to describe *f* (description length)

AIC (Akiake IC) C(*f*) = # parameters

Allows # parameters to be infinite as # training data n become large

BIC (Bayesian IC) C(*f*) = # parameters * log n

Penalizes complex models more heavily – limits complexity of models as # training data n become large

Model selection

Model Selection

- Model classes with increasing complexity
  - Regularization parameter λ in structural risk estimators
    - Larger values of λ => Lower complexity
    - Question: How to select λ?
  - Regression with polynomials of order k = 0, 1, 2, …
    - Higher degree => Higher complexity
    - Question: How to select k?
  - k and λ are called "tuning" parameters

- General setup:
  - Define a finite set of model classes
    - Regression: $\{\mathcal{F}_{k=0}, \mathcal{F}_{k=1}, \mathcal{F}_{k=2}\}$
    - Structural risk: $\{\mathcal{F}_{\lambda=0.01}, \mathcal{F}_{\lambda=0.1}, \mathcal{F}_{\lambda=1}\}$
  - For each model class, find best estimator in model class, and estimate corresponding true risks: $\{\hat{R}(\hat{f}_1), \hat{R}(\hat{f}_2), \hat{R}(\hat{f}_3)\}$
  - Model selection: Select best model class: $\arg\min_i \hat{R}(\hat{f}_i)$

Model Selection

Formal setup:

Model Classes $\{\mathcal{F}_\lambda\}_{\lambda \in \Lambda}$ of increasing complexity $\mathcal{F}_1 \prec \mathcal{F}_2 \prec \ldots$

$$\min_{\lambda} \min_{f \in \mathcal{F}_\lambda} J(f, \lambda)$$

Stage I: *Given* λ, estimate $\hat{f}_\lambda$ using

- Empirical risk minimization
- Structural risk minimization
- Complexity regularized risk minimization

Stage II: *Select* λ for which $\hat{f}_\lambda$ has minimum value of true risk estimated using

- **Cross-validation**
- **Hold-out**
- **Information-theoretic risk estimates (AIC, BIC)**

Estimating True Risk of estimators

Estimating True Risk of Estimators

- Suppose we train an estimator $\widehat{f}_D$ on data D
- How do we estimate its true risk $R(\widehat{f}_D)$?
- We could use the training data D itself i.e. use empirical risk on training data $\widehat{R}_D(\widehat{f}_D)$
- Not such a good idea
- If the midterm questions are comprised entirely of homework questions, would the midterm grade be an optimistic estimate of the "true" midterm grade?
  - Yes!
- Similarly, using the empirical risk on training data would be an optimistic estimate of the true risk

Algorithmic and Closed Form Estimates of True Risk

- Algorithmic Estimates of True Risk:
  - Empirical Risk
    - Optimistic
  - Evaluating Risk on a holdout set
  - Cross-validation
- Closed form Estimates of True Risk
  - Structural Risk

Hold-out method

Can judge generalization error by using an independent sample of data.

Hold – out procedure:

n data points available $D \equiv \{X_i, Y_i\}_{i=1}^{n}$

1) Split into two sets: Training dataset Holdout dataset

$D_T = \{X_i, Y_i\}_{i=1}^{m}$ $D_V = \{X_i, Y_i\}_{i=m+1}^{n}$

2) Use D_T for training a predictor $\widehat{f}_{D_T}$

3) Use D_V for evaluating the predictor

$\widehat{R}_{D_V}(\widehat{f}_{D_T})$

Hold-out method

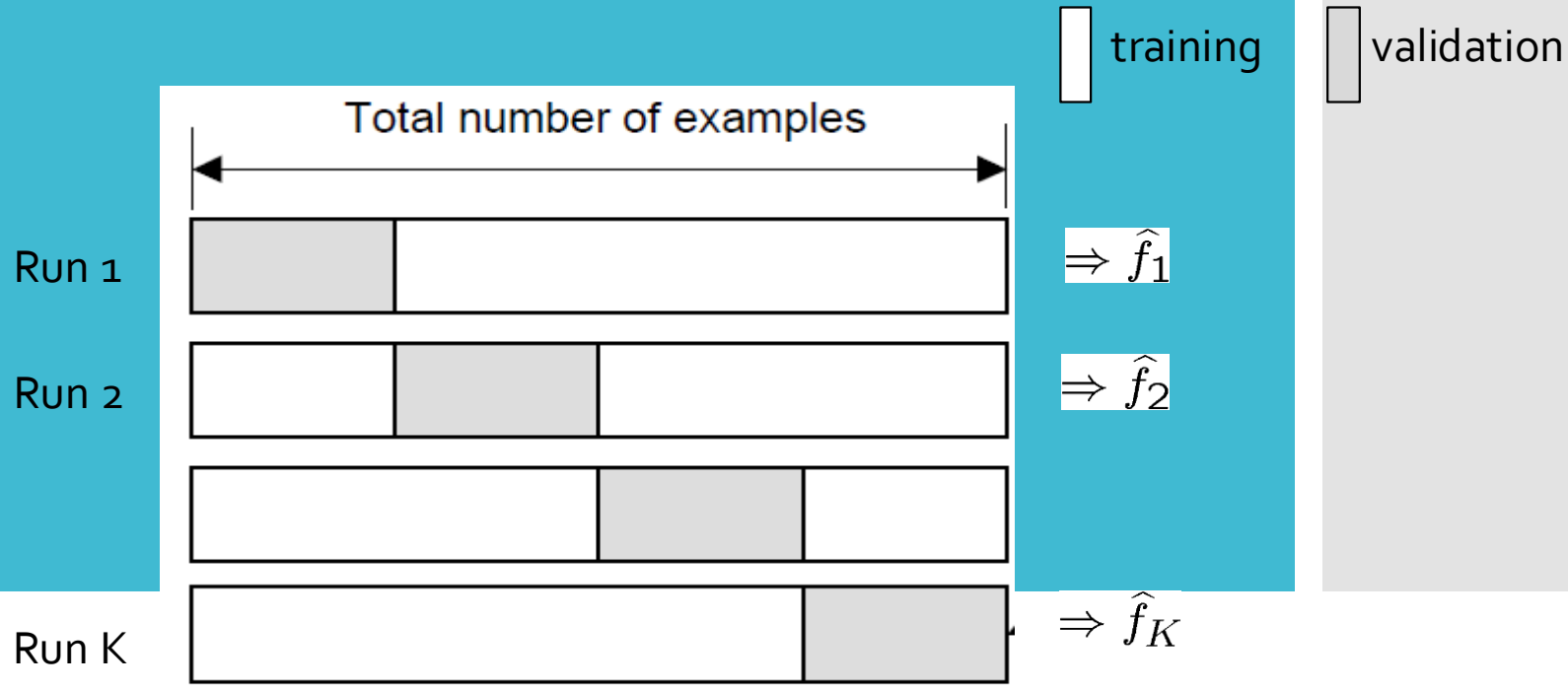
Drawbacks:

- May not have enough data to afford setting one subset aside for getting a sense of generalization abilities
- Holdout error may be misleading (bad estimate of generalization error) if we get an "unfortunate" split

Cross-validation

K-fold cross-validation

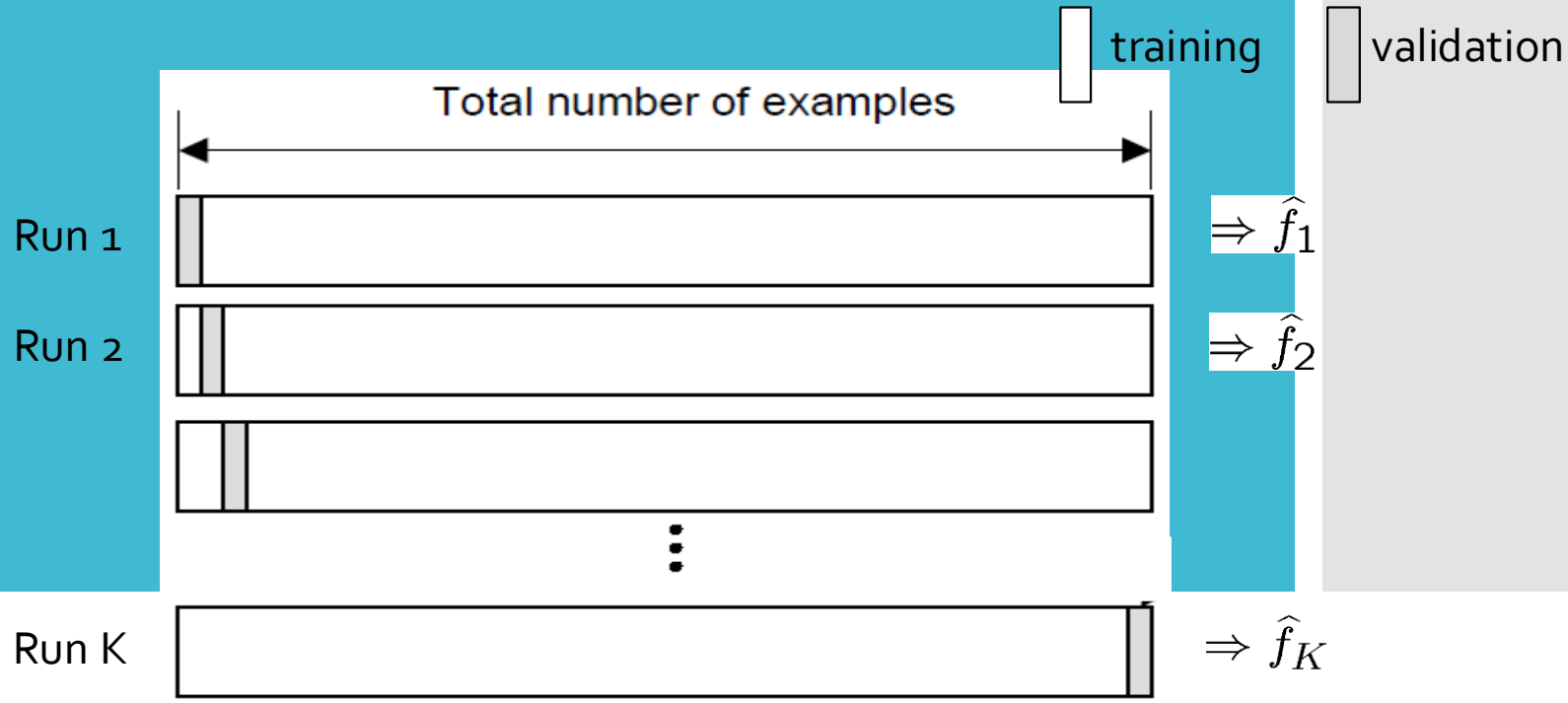
Create K-fold partition of the dataset.
Form K hold-out predictors, each time using one partition as validation and rest K-1 as training datasets.
Final predictor is average/majority vote over the K hold-out estimates.



Cross-validation

Leave-one-out (LOO) cross-validation

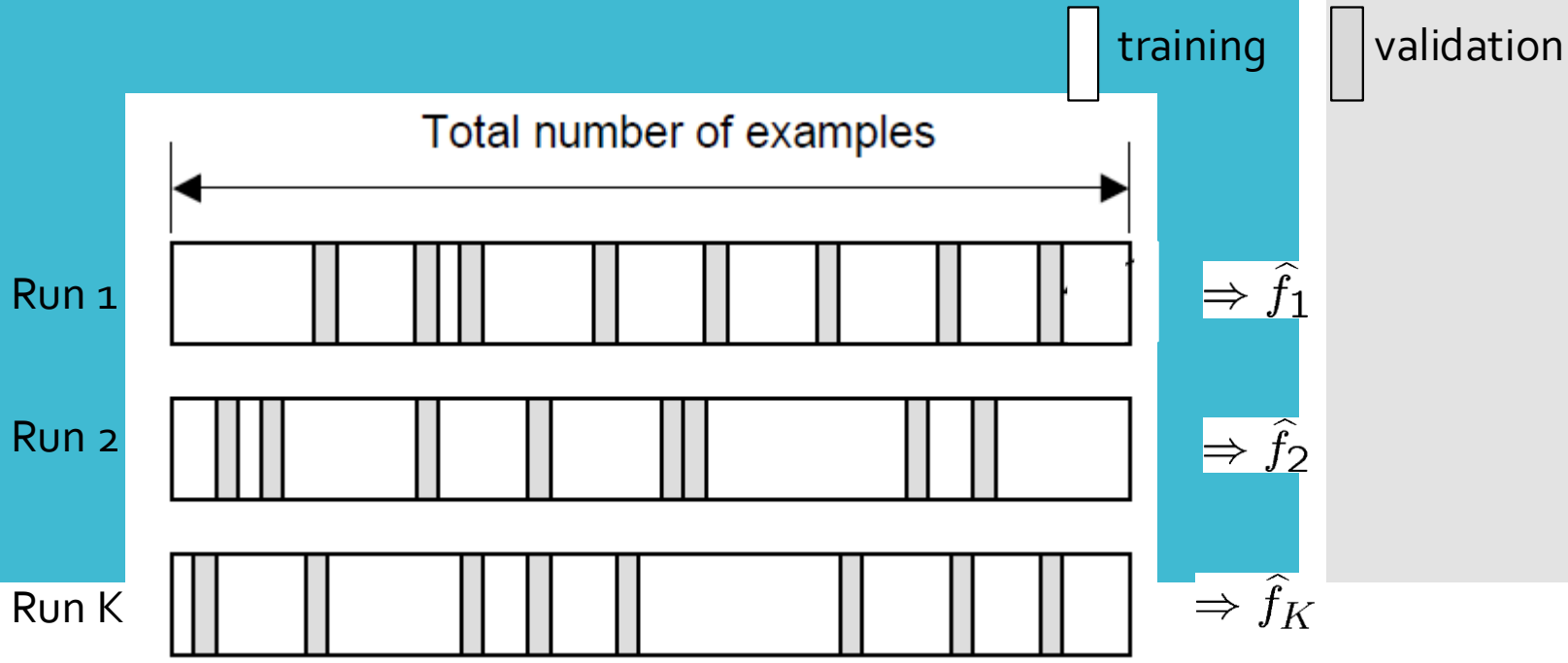
Special case of K-fold with K=n partitions
Equivalently, train on n-1 samples and validate on only one sample per run for n runs



Cross-validation

Random subsampling

Randomly subsample a fixed fraction αn ($0 < \alpha < 1$) of the dataset for validation.
Form hold-out predictor with remaining data as training data.
Repeat K times
Final predictor is average/majority vote over the K hold-out estimates.



Estimating true risk

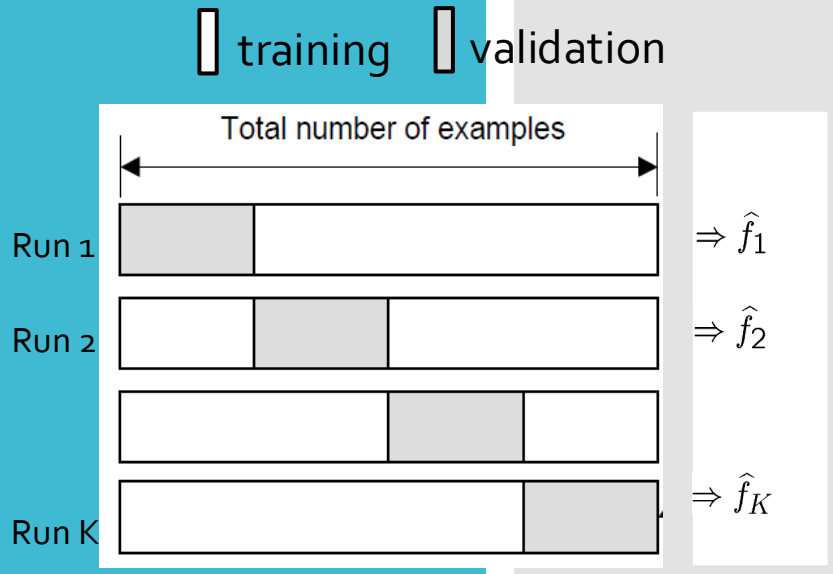
K-fold/LOO/random sub-sampling:

Error estimate = $\frac{1}{K}\sum_{k=1}^{K}\hat{R}_{V_k}(\hat{f}_{T_k})$

We want to estimate the error of a predictor based on n data points.

If K is large (close to n), bias of error estimate is small since each training set has close to n data points.

However, variance of error estimate is high since each validation set has fewer data points and $\hat{R}_{V_k}$ might deviate a lot from the mean.



Practical Issues in Cross-validation

How to decide the values for *K* and α ?

- Large K
  - + The bias of the error estimate will be small
  - - The variance of the error estimate will be large (few validation pts)
  - - The computational time will be very large as well (many experiments)

- Small K
  - + The # experiments and, therefore, computation time are reduced
  - + The variance of the error estimate will be small (many validation pts)
  - - The bias of the error estimate will be large

Common choice: K = 10, α = 0.1 ☺

Quick Game

Two truths and a lie

Model selection & estimating true risk

A. A hold-out set can estimate generalization error, but an “unfortunate split” can be misleading.

B. K-fold CV trades bias vs. variance: larger K → lower bias but higher variance and more compute.

C. Training-set error is an unbiased estimate of true risk, so you can tune λ using only training error.

Quick Game

Two truths and a lie

Model selection & estimating true risk

A. A hold-out set can estimate generalization error, but an “unfortunate split” can be misleading.

B. K-fold CV trades bias vs. variance: larger K → lower bias but higher variance and more compute.

C. Training-set error is an unbiased estimate of true risk, so you can tune λ using only training error.

Reveal: C is the lie (training error is optimistic).

Structural Risk

Add a penalty based on deviation of true and empirical risks:

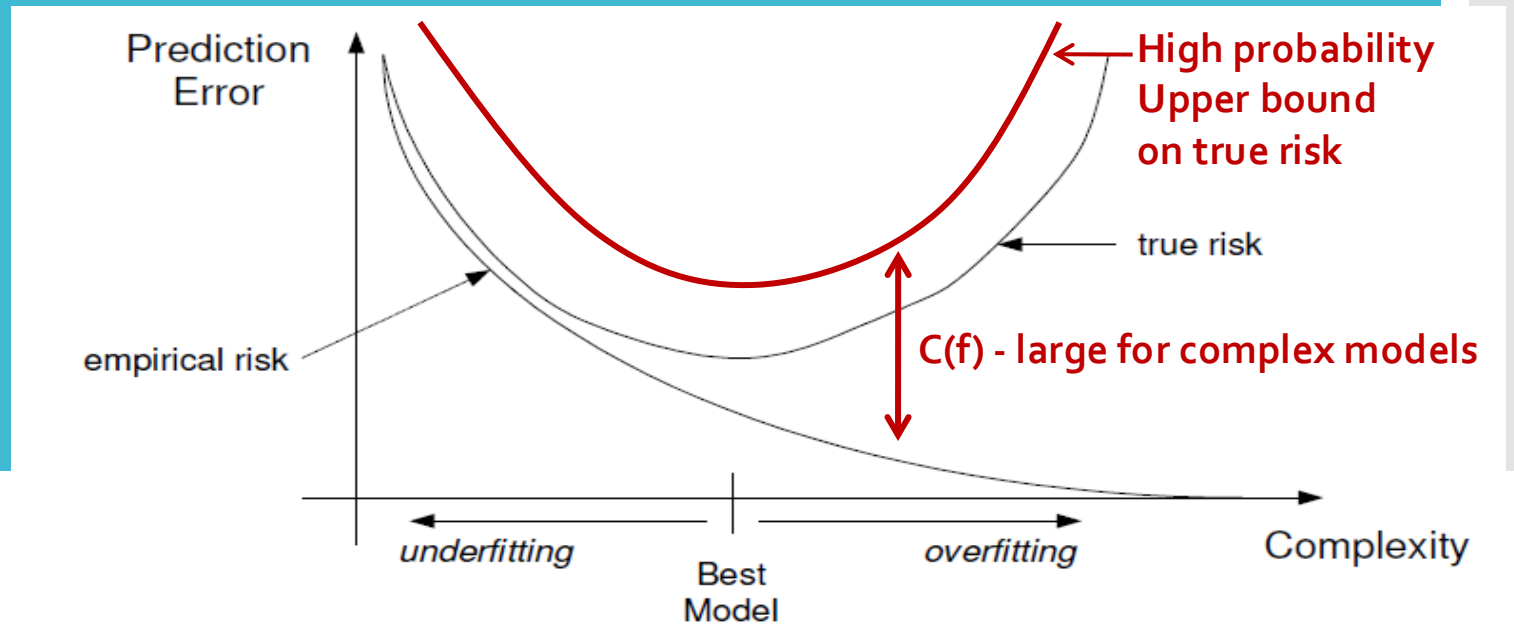
Suppose we have a bound, that with high probability:

$$|R(f) - \widehat{R}_n(f)| \leq C(f) \quad \forall f \in \mathcal{F}$$

Concentration bounds (later)

$$R(f) \leq \widehat{R}_n(f) + C(f), \quad \forall f \in \mathcal{F}$$

Use $\widehat{R}_n(\widehat{f}_n) + C(\widehat{f}_n)$ as a *pessimistic* estimate of true risk!



Analyzing generalization error Via True Risk

Estimation and Approximation Errors

Estimated Predictor : $\widehat{f}_n$

Optimal Predictor : f^*

Risk of Estimated Predictor : $R(\widehat{f}_n)$

Above is random due to samples in training data

Expectation of above wrt training data : $\mathbb{E}(R(\widehat{f}_n))$

Risk of Optimal Predictor : $R(f^*)$

Players in the risk minimization story

Estimated Predictor : $\widehat{f}_n$

Optimal Predictor : f^*

Risk of Estimated Predictor : $R(\widehat{f}_n)$

Above is random due to samples in training data

Expectation of above wrt training data : $\mathbb{E}(R(\widehat{f}_n))$

Risk of Optimal Predictor : $R(f^*)$

Interested in the excess risk: $\mathbb{E}(R(\widehat{f}_n)) - R(f^*)$

Behavior of True Risk

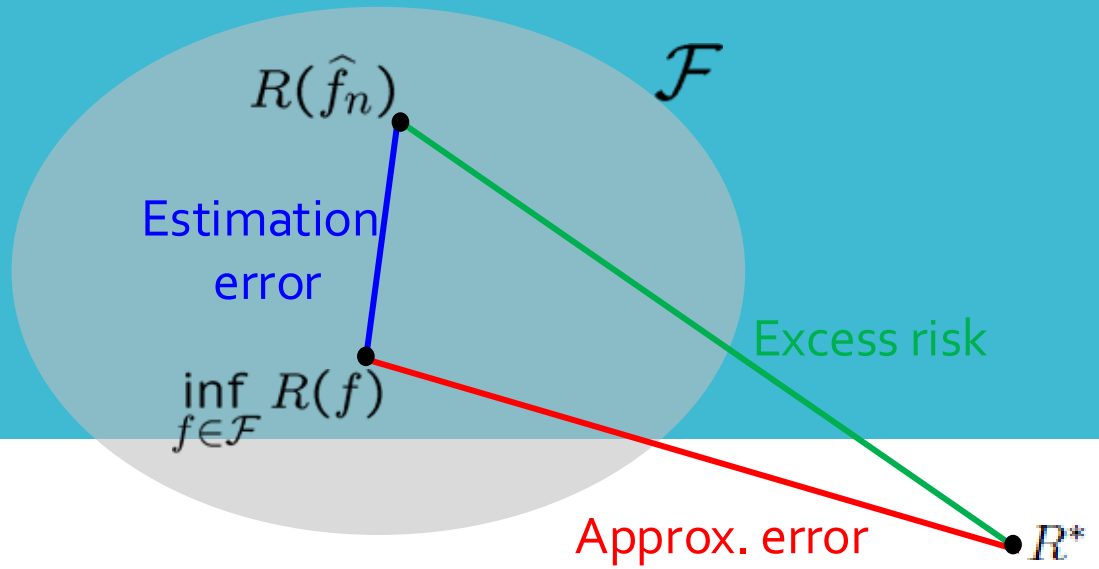
Want $\widehat{f}_n$ to be as good as optimal predictor f^*

Excess Risk

$$E\left[R(\widehat{f}_n)\right] - R^* = \underbrace{\left(E[R(\widehat{f}_n)] - \inf_{f\in\mathcal{F}} R(f)\right)}_{\text{estimation error}} + \underbrace{\left(\inf_{f\in\mathcal{F}} R(f) - R^*\right)}_{\text{approximation error}}$$

finite sample size + noise ← Due to randomness of training data

Due to restriction of model class

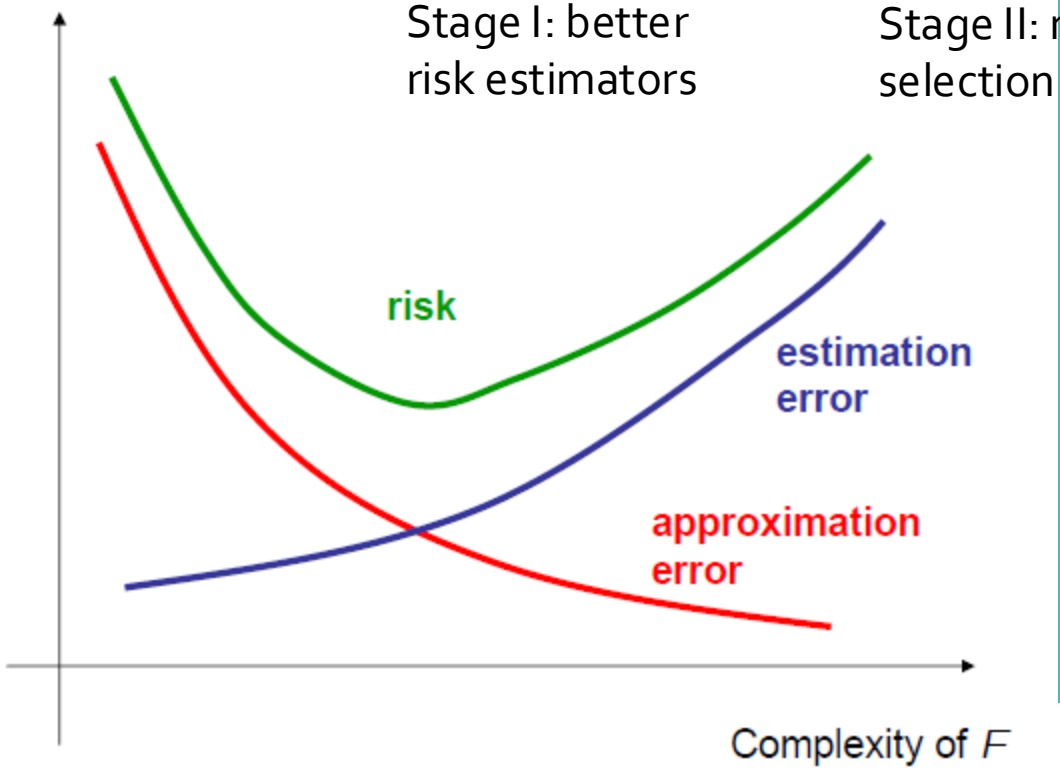


Behavior of True Risk

$$E\left[R(\widehat{f}_n)\right] - R^* = \underbrace{\left(E[R(\widehat{f}_n)] - \inf_{f\in\mathcal{F}} R(f)\right)}_{\text{estimation error}} + \underbrace{\left(\inf_{f\in\mathcal{F}} R(f) - R^*\right)}_{\text{approximation error}}$$

Stage I: better risk estimators

Stage II: model selection



Key Takeaways

- Non-linear feature transformations allow for learning non-linear functions/decision boundaries
  - Can lead to overfitting…
    - Address with regularization!
    - Regularization level is a hyperparameter
  - Can be computationally expensive…
    - Address with kernels!
    - Alternative to explicitly computing feature transformations for inner product methods

End Poll

Quick poll (pick one)

In ridge regression / complexity regularization, what typically happens as λ increases?

A. Model complexity increases; overfitting risk increases

B. Model complexity decreases; overfitting risk often decreases (but too large λ can underfit)

C. Training and test error both always decrease

D. λ only affects runtime, not the learned predictor

E. Empirical risk becomes an unbiased estimate of true risk

Poll: https://forms.gle/tttDMCujzHCFowVm6