南方科技大学

MAT8034: Machine Learning

Generalization and Regularization

Fang Kong

https://fangkongx.github.io/Teaching/MAT8034/Spring2026/index.html

Part of slide credit: Stanford CS229

Outline

- Intuition
- Bias-variance tradeoff
- Sample complexity bounds
- The double descent phenomenon
- Implicit regularization effect
- Bayesian statistics and regularization

Intuition

Intuition

- Recall in previous classes
  - We typically learn a model h_θ by minimizing the training loss/error
  - $J_\theta = \frac{1}{n}\sum_{i=1}^{n}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2$
  - This is not the ultimate goal
- The ultimate goal
  - Sample a test data from the test distribution $\mathcal{D}$
  - Measure the model's error on the test data (test loss/error)

$$L(\theta) = \mathbb{E}_{(x,y)\sim\mathcal{D}}[(y - h_\theta(x))^2]$$

  - Can be approximated by the average error on many sampled test examples

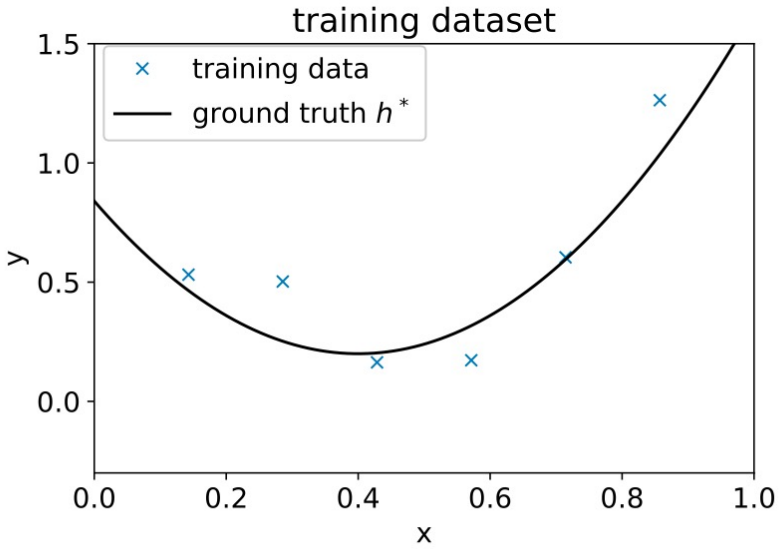
Challenges

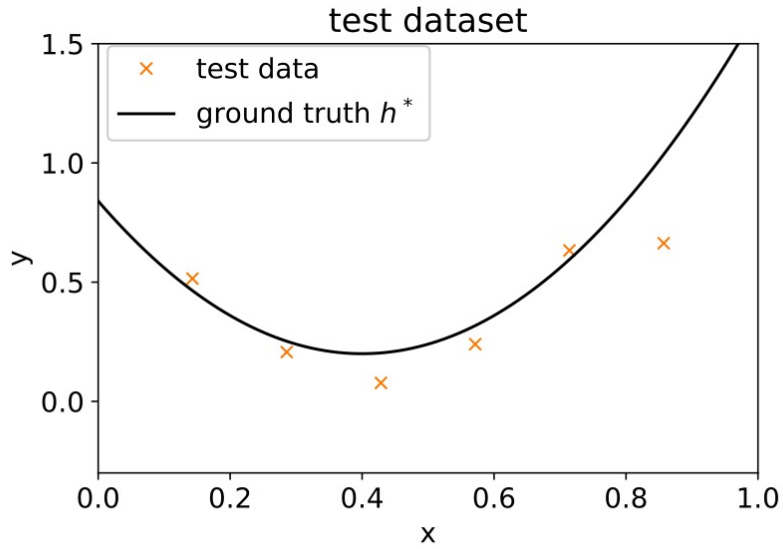
- The test examples are unseen
  - Even though the training set is sampled from the same distribution $\mathcal{D}$, it can not guaranteed that the test error is close to the training error
  - Minimizing training error may not lead to a small test error
- Important concepts
  - Overfitting: the model predicts accurately on the training dataset but doesn't generalize well to other test examples
  - Underfitting: the training error is relatively large (typically the test error is also relatively large)

- How the test error is influenced by the learning procedure, especially the choice of model parameterizations?

Bias-variance tradeoff

Problem setting





- The training inputs are randomly chosen
- The outputs are generated by $y^{(i)} = h^{\star}(x^{(i)}) + \xi^{(i)}$
  - $h^*(\cdot)$: a quadratic function
  - $\xi^{(i)} \sim N(0, \sigma^2)$: noise
- Our goal is to recover the function $h^*(\cdot)$

How about fitting a linear model?

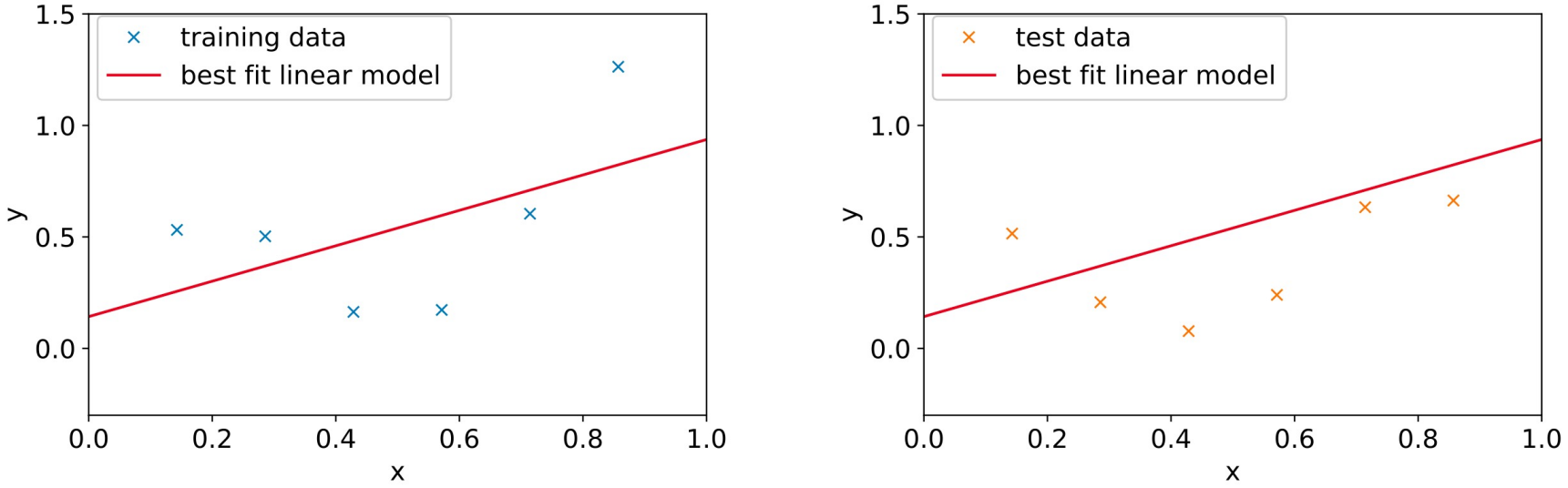


Figure 8.2: The best fit linear model has large training and test errors.

- The true relationship between y and x is not linear
- Any linear model is far away from the true function
- The training error is large, underfitting

How about fitting a linear model? (cont'd)

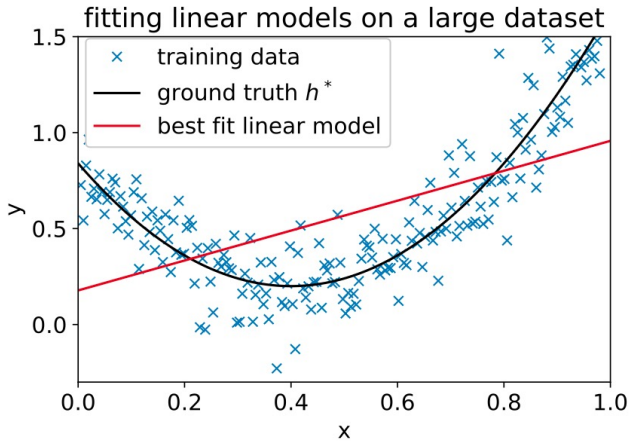


Figure 8.3: The best fit linear model on a much larger dataset still has a large training error.

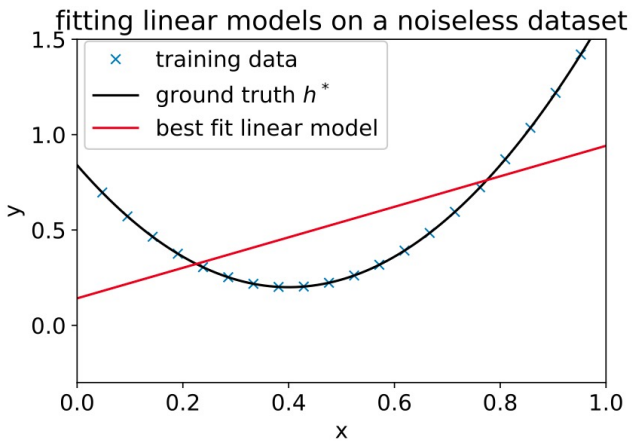


Figure 8.4: The best fit linear model on a noiseless dataset also has a large training/test error.

- Fundamental bottleneck: linear model family's inability to capture the structure in the data
- Define model bias: the test error even if we were to fit it to a very (say, infinitely) large training dataset

How about a 5th-degree polynomial?

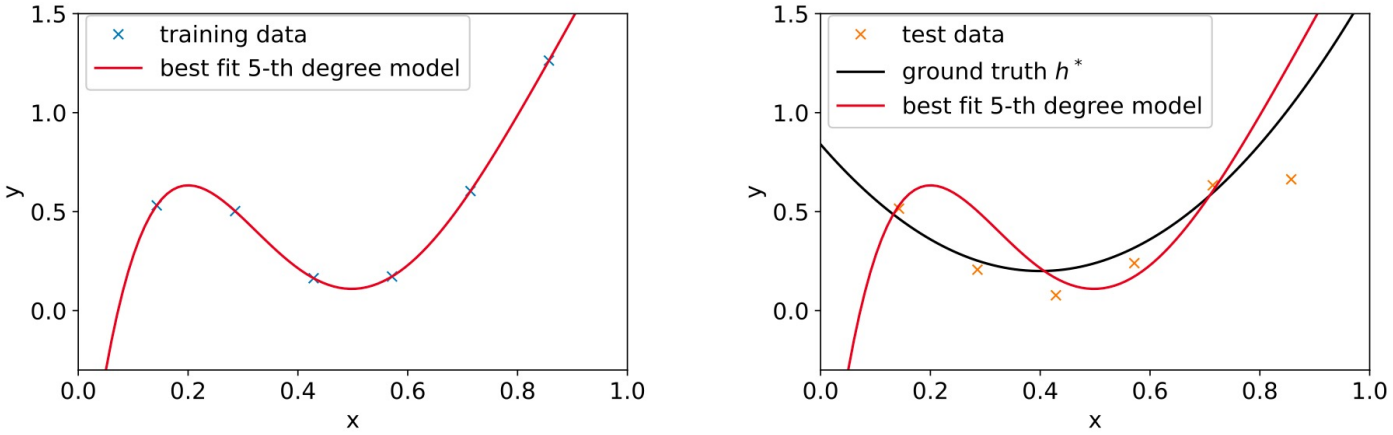
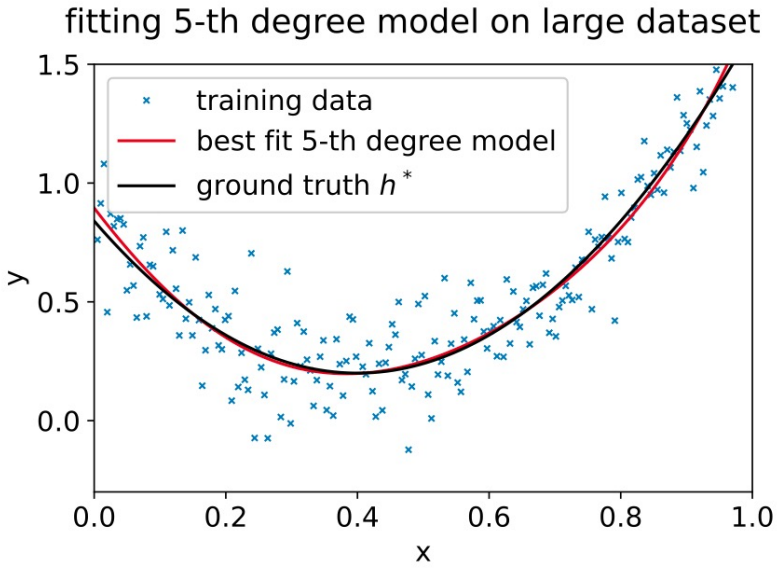


Figure 8.5: Best fit 5-th degree polynomial has zero training error, but still has a large test error and does not recover the the ground truth. This is a classic situation of overfitting.

- Predict well on the training set, does not work well on test examples

How about a 5th-degree polynomial? (cont'd)



- When the training set becomes huge, the model recovers the ground-truth

How about a 5th-degree polynomial? (cont'd)

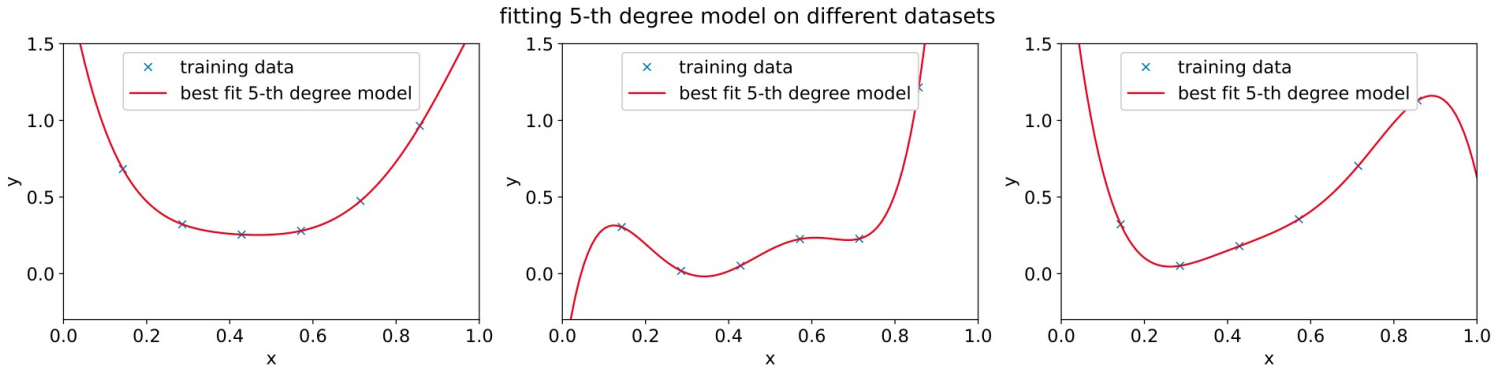


Figure 8.7: The best fit 5-th degree models on three different datasets generated from the same distribution behave quite differently, suggesting the existence of a large variance.

- Failure: fitting patterns in the data that happened to be present in the small, finite training set (NOT the real relationship between x and y)
- Define variance: the amount of variations across models learnt on multiple different training datasets (drawn from the same underlying distribution)

Bias-variance trade-off

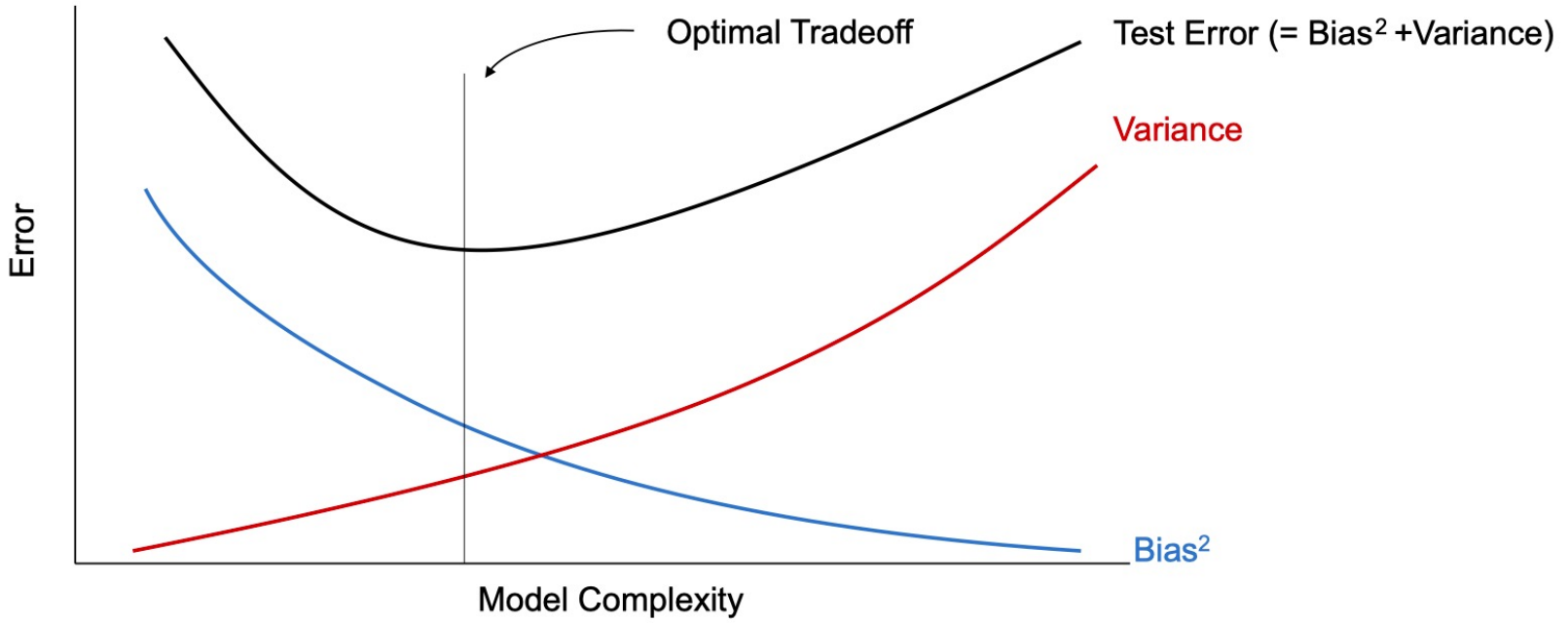
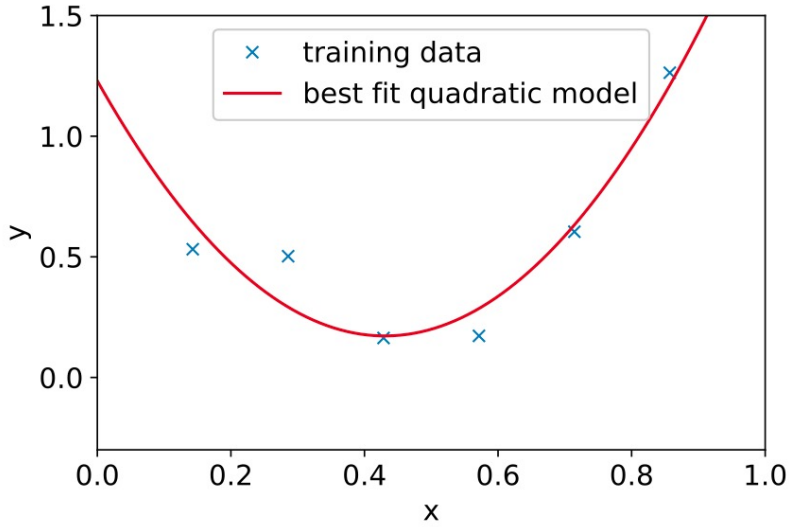


Figure 8.8: An illustration of the typical bias-variance tradeoff.

Bias-variance trade-off (cont'd)



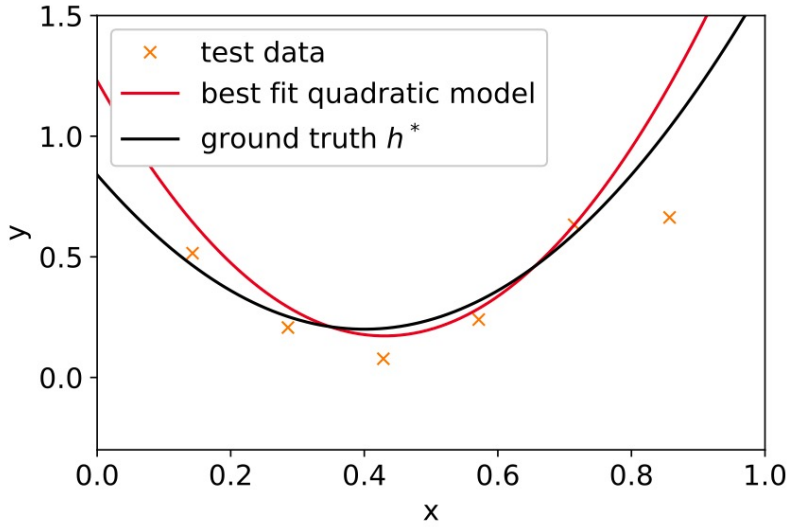


Figure 8.9: Best fit quadratic model has small training and test error because quadratic model achieves a better tradeoff.

A mathematical decomposition (for regression)

Problem setting: regression

- Draw a training dataset $S = \{x^{(i)}, y^{(i)}\}_{i=1}^{n}$ such that $y^{(i)} = h^{\star}(x^{(i)}) + \xi^{(i)}$ where $\xi^{(i)} \in N(0, \sigma^2)$.
- Train a model on the dataset S, denoted by $\hat{h}_S$.
- Take a test example (x, y) such that $y = h^{\star}(x) + \xi$ where $\xi \sim N(0, \sigma^2)$, and measure the expected test error (averaged over the random draw of the training set S and the randomness of ξ)

$$\mathrm{MSE}(x) = \mathbb{E}_{S,\xi}[(y - \hat{h}_S(x))^2] \tag{8.2}$$

Decomposition

- $$\begin{aligned}\mathrm{MSE}(x) = \mathbb{E}[(y - h_S(x))^2] &= \mathbb{E}[(\xi + (h^\star(x) - h_S(x)))^2] \\ &= \mathbb{E}[\xi^2] + \mathbb{E}[(h^\star(x) - h_S(x))^2] \\ &= \sigma^2 + \mathbb{E}[(h^\star(x) - h_S(x))^2]\end{aligned}$$

- Define $h_{avg}(x) = \mathbb{E}_S[(h_S(x))]$
  - The model obtained by drawing an infinite number of datasets, training on them, and averaging their predictions on x

- $$\begin{aligned}\mathrm{MSE}(x) &= \sigma^2 + \mathbb{E}[(h^\star(x) - h_S(x))^2] \\ &= \sigma^2 + (h^\star(x) - h_{\mathrm{avg}}(x))^2 + \mathbb{E}[(h_{\mathrm{avg}} - h_S(x))^2] \\ &= \underbrace{\sigma^2}_{\text{unavoidable}} + \underbrace{(h^\star(x) - h_{\mathrm{avg}}(x))^2}_{\triangleq \text{ bias}^2} + \underbrace{\mathrm{var}(h_S(x))}_{\triangleq \text{ variance}}\end{aligned}$$

Sample complexity bounds

Objective

- Some questions
  - Can we relate error on the training set to generalization error?
  - Can we make formal the bias/variance tradeoff that was just discussed?
  - Are there conditions under which we can actually prove that learning algorithms will work well?

Useful lemmas

- **Lemma.** (The union bound). Let $A_1, A_2, \ldots, A_k$ be k different events (that may not be independent). Then

$$P(A_1 \cup \cdots \cup A_k) \leq P(A_1) + \ldots + P(A_k).$$

- **Lemma.** (Hoeffding inequality) Let $Z_1, \ldots, Z_n$ be n independent and identically distributed (iid) random variables drawn from a Bernoulli(ϕ) distribution. I.e., $P(Z_i = 1) = \phi$, and $P(Z_i = 0) = 1 - \phi$. Let $\hat{\phi} = (1/n) \sum_{i=1}^{n} Z_i$ be the mean of these random variables, and let any $\gamma > 0$ be fixed. Then

$$P(|\phi - \hat{\phi}| > \gamma) \leq 2 \exp(-2\gamma^2 n)$$

Problem setting

- To simplify, consider the classification problem with $y \in \{0,1\}$
- Training set $S = \{(x^i, y^i); i = 1,2, \dots, n\}$, drawn iid from $\mathcal{D}$

- For hypothesis h, define training error (empirical risk/error)

$$\hat{\varepsilon}(h) = \frac{1}{n}\sum_{i=1}^{n} 1\{h(x^{(i)}) \neq y^{(i)}\}$$

- Define the generalization error $\varepsilon(h) = P_{(x,y)\sim\mathcal{D}}(h(x) \neq y)$

One of PAC assumption: training and testing set are from the same D

Problem setting (cont'd)

- Consider the linear classification $h_\theta(x) = 1\{\theta^\top x \geq 0\}$
- Objective: minimize the training error

$$\hat{\theta} = \arg\min_\theta \hat{\varepsilon}(h_\theta)$$

$$\hat{h} = h_{\hat{\theta}}$$

empirical risk minimization

- In learning theory, it will be useful to abstract away from the specific parameterization of hypotheses
- Define the hypothesis class $\mathcal{H}$, for linear classification

$$\mathcal{H} = \{h_\theta : h_\theta(x) = 1\{\theta^T x \geq 0\}, \theta \in \mathbb{R}^{d+1}\}$$

Problem setting (cont'd)

- ERM becomes finding $\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\varepsilon}(h)$
- For simplicity, first consider the finite hypothesis set

$$\mathcal{H} = \{h_1, \ldots, h_k\}$$

- Now, show the guarantee for the generalization error of $\hat{h}$
  - 1. $\forall h, \hat{\varepsilon}(h)$ is a reliable estimate of $\varepsilon(h)$
  - 2. $\hat{h}$ guarantees good generalization error

Guarantee for a fixed hypothesis function

- Fix any hypothesis function $h_i \in \mathcal{H}$
- Define $Z_j = 1\{h_i(x^j) \neq y^j\}$
- The training error is

$$\hat{\varepsilon}(h_i) = \frac{1}{n}\sum_{j=1}^{n} Z_j$$

- The empirical mean of n random variables with expectation $\varepsilon(h_i)$
- Applying Hoeffding inequality,

$$P(|\varepsilon(h_i) - \hat{\varepsilon}(h_i)| > \gamma) \leq 2\exp(-2\gamma^2 n)$$

Guarantee for any hypothesis function

- $$\begin{aligned} P(\exists h \in \mathcal{H}.|\varepsilon(h_i) - \hat{\varepsilon}(h_i)| > \gamma) &= P(A_1 \cup \cdots \cup A_k) \\ &\leq \sum_{i=1}^{k} P(A_i) \\ &\leq \sum_{i=1}^{k} 2\exp(-2\gamma^2 n) \\ &= 2k\exp(-2\gamma^2 n) \end{aligned}$$

- Thus $$\begin{aligned} P(\neg\exists h \in \mathcal{H}.|\varepsilon(h_i) - \hat{\varepsilon}(h_i)| > \gamma) &= P(\forall h \in \mathcal{H}.|\varepsilon(h_i) - \hat{\varepsilon}(h_i)| \leq \gamma) \\ &\geq 1 - 2k\exp(-2\gamma^2 n) \end{aligned}$$

Questions

- How large must n be before we can guarantee that with probability at least $1 - \delta$, training error will be within γ of generalization error? (sample complexity)

- What is the distance between the training error and generalization error with training set size n and confidence δ?

Guarantee for the output hypothesis function

- Recall $\hat{h} = \arg\min_{h \in \mathcal{H}} \hat{\varepsilon}(h)$
- Define the best hypothesis is $h^* = \arg\min_{h \in \mathcal{H}} \varepsilon(h)$
- Then
$$\begin{aligned} \varepsilon(\hat{h}) &\leq \hat{\varepsilon}(\hat{h}) + \gamma \\ &\leq \hat{\varepsilon}(h^*) + \gamma \\ &\leq \varepsilon(h^*) + 2\gamma \end{aligned}$$
- If uniform convergence occurs, then the generalization error of h is at most 2γ worse than the best possible hypothesis in $\mathcal{H}$!

Theorem of generalization error

- **Theorem.** Let $|\mathcal{H}| = k$, and let any n, δ be fixed. Then with probability at least $1 - \delta$, we have that

$$\varepsilon(\hat{h}) \leq \left(\min_{h \in \mathcal{H}} \varepsilon(h) \right) + 2\sqrt{\frac{1}{2n} \log \frac{2k}{\delta}}.$$

- Explanation of bias/variance
  - If we switch to a larger function class $\mathcal{H}' \supseteq \mathcal{H}$
  - The first term decreases: lower bias
  - The second term increases as k increases: higher variance

Corollary of sample complexity

- **Corollary.** Let $|\mathcal{H}| = k$, and let any δ, γ be fixed. Then for $\varepsilon(\hat{h}) \leq \min_{h \in \mathcal{H}} \varepsilon(h) + 2\gamma$ to hold with probability at least $1 - \delta$, it suffices that

$$\begin{aligned} n &\geq \frac{1}{2\gamma^2} \log \frac{2k}{\delta} \\ &= O\left(\frac{1}{\gamma^2} \log \frac{k}{\delta}\right), \end{aligned}$$

Extension to infinite $\mathcal{H}$: Intuition

- Usually the hypothesis set is infinite
  - For example, the linear function set contains a infinite number of parameters

- Suppose $\mathcal{H}$ is parameterized by d real numbers
- The computer uses 64 bits to represent a floating point number
- $\mathcal{H}$ contains 2^{64d} different hypotheses
- Existing results show that with fixed γ, δ

$$n \geq O\left(\frac{1}{\gamma^2}\log\frac{2^{64d}}{\delta}\right) = O\left(\frac{d}{\gamma^2}\log\frac{1}{\delta}\right) = O_{\gamma,\delta}(d)$$

VC dimension

- Shatter

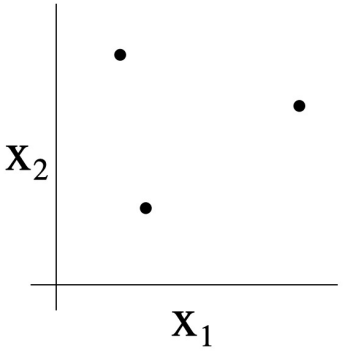
Given a set $S = \{x^{(i)}, \ldots, x^{(\mathbf{D})}\}$ (no relation to the training set) of points $x^{(i)} \in \mathcal{X}$, we say that $\mathcal{H}$ **shatters** S if $\mathcal{H}$ can realize any labeling on S. I.e., if for any set of labels $\{y^{(1)}, \ldots, y^{(\mathbf{D})}\}$, there exists some $h \in \mathcal{H}$ so that $h(x^{(i)}) = y^{(i)}$ for all $i = 1, \ldots \mathbf{D}$.

- VC dimension

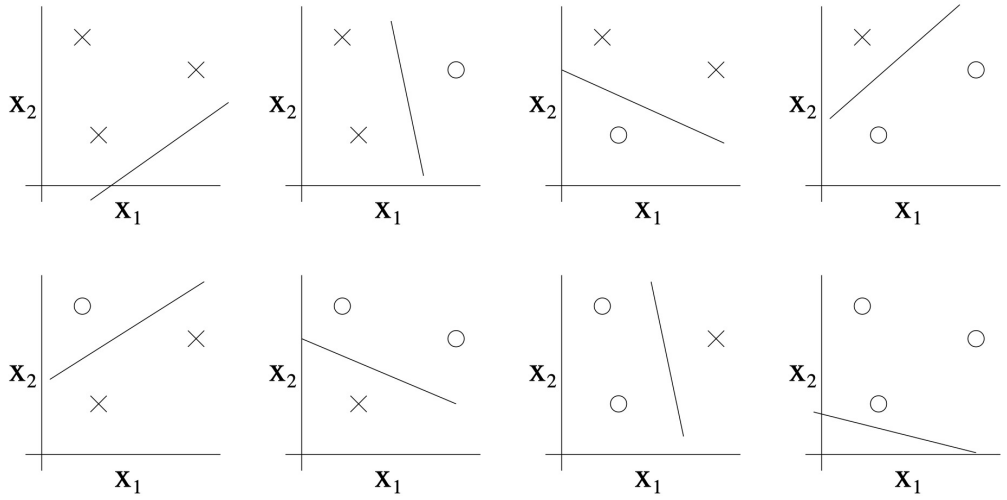
Given a hypothesis class $\mathcal{H}$, we then define its **Vapnik-Chervonenkis dimension**, written $\mathrm{VC}(\mathcal{H})$, to be the size of the largest set that is shattered by $\mathcal{H}$. (If $\mathcal{H}$ can shatter arbitrarily large sets, then $\mathrm{VC}(\mathcal{H}) = \infty$.)

VC dimension: illustration

- Can the set $\mathcal{H}$ of linear classifiers in two dimensions shatter the set below?



- For any labeling, $\mathcal{H}$ can correctly classify



VC dimension: illustration (cont'd)

- How about 4 points?

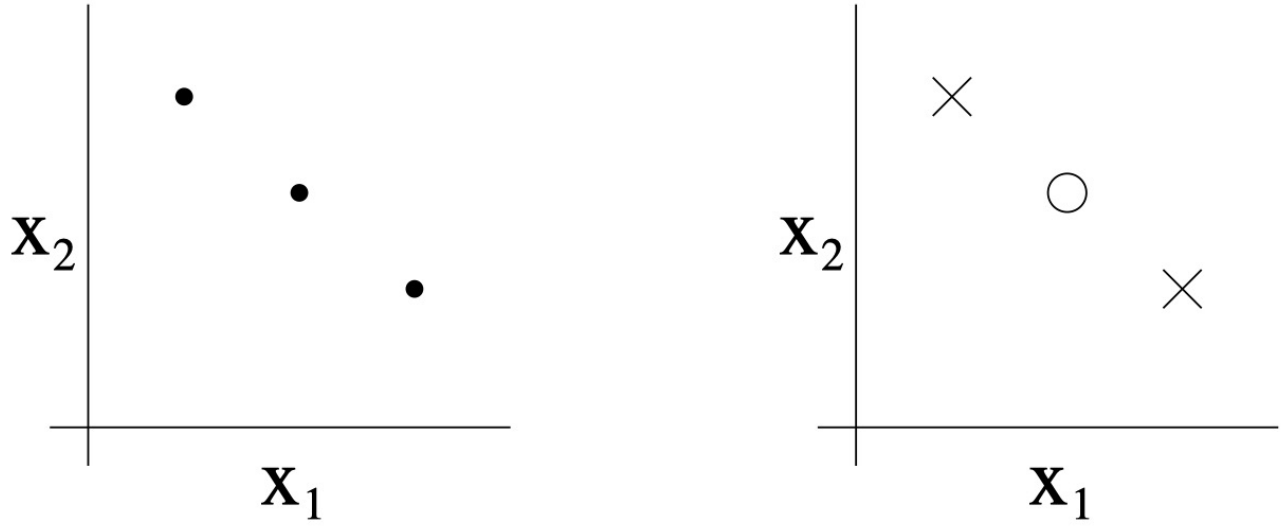
VC dimension: illustration (cont'd)

- How about 4 points?
  - No

- Thus, the largest set that $\mathcal{H}$ can shatter is of size 3, and hence $\text{VC}(\mathcal{H}) = 3$.

VC dimension: illustration (cont'd)

- In order to prove that VC($\mathcal{H}$) is at least D, we need to show only that there's **at least one** set of size D that H can shatter (not every set of size D)



Convergence results

- **Theorem.** Let $\mathcal{H}$ be given, and let $\mathbf{D} = \mathrm{VC}(\mathcal{H})$. Then with probability at least $1 - \delta$, we have that for all $h \in \mathcal{H}$,

$$|\varepsilon(h) - \hat{\varepsilon}(h)| \leq O\left(\sqrt{\frac{\mathbf{D}}{n}\log\frac{n}{\mathbf{D}} + \frac{1}{n}\log\frac{1}{\delta}}\right).$$

  Thus, with probability at least $1 - \delta$, we also have that:

$$\varepsilon(\hat{h}) \leq \varepsilon(h^*) + O\left(\sqrt{\frac{\mathbf{D}}{n}\log\frac{n}{\mathbf{D}} + \frac{1}{n}\log\frac{1}{\delta}}\right).$$

Usually the VC dimension is roughly linear in the number of parameters

- **Corollary.** For $|\varepsilon(h) - \hat{\varepsilon}(h)| \leq \gamma$ to hold for all $h \in \mathcal{H}$ (and hence $\varepsilon(\hat{h}) \leq \varepsilon(h^*) + 2\gamma$) with probability at least $1 - \delta$, it suffices that $n = O_{\gamma,\delta}(\mathbf{D})$.

The double descent phenomenon

Observation

- Previous works show that

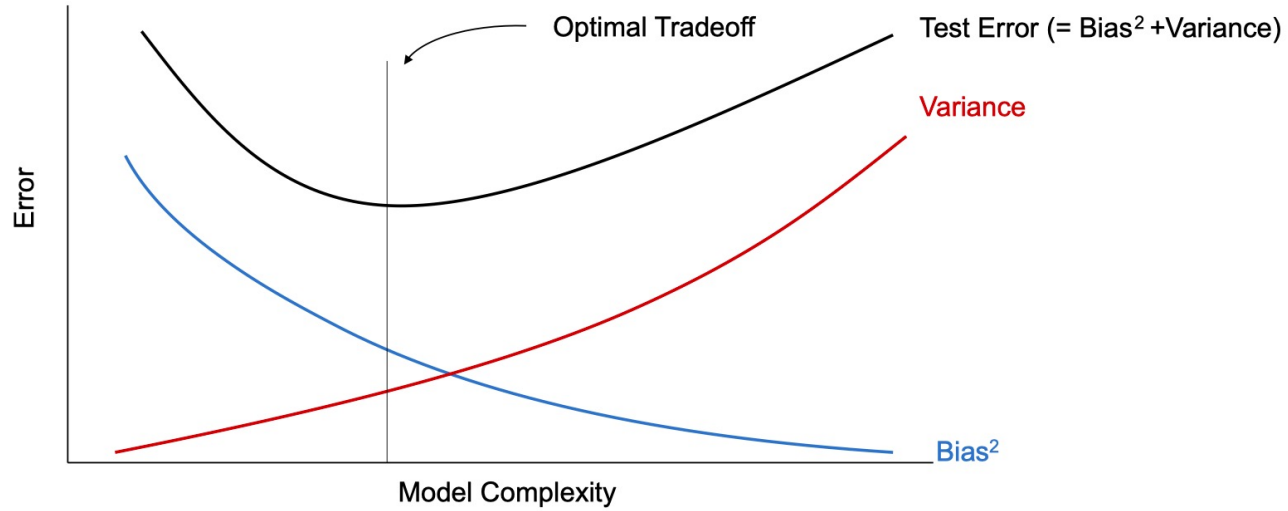
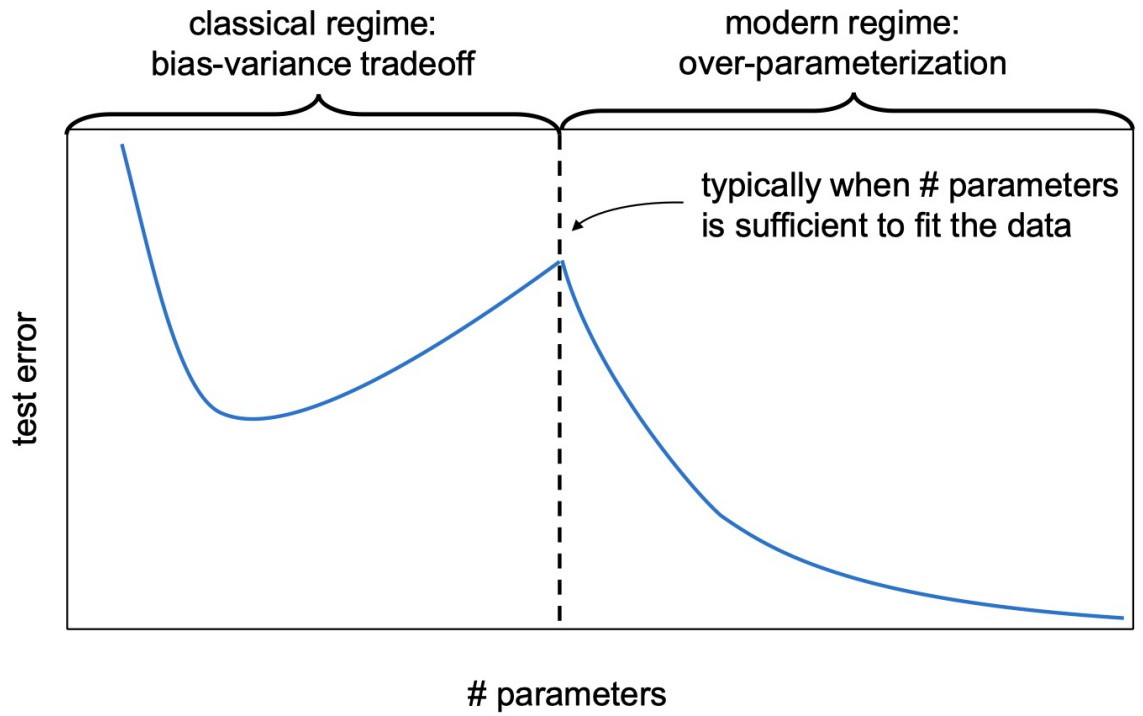


Figure 8.8: An illustration of the typical bias-variance tradeoff.

- Interestingly, the bias-variance tradeoff curves or the test error curves do not universally follow the shape

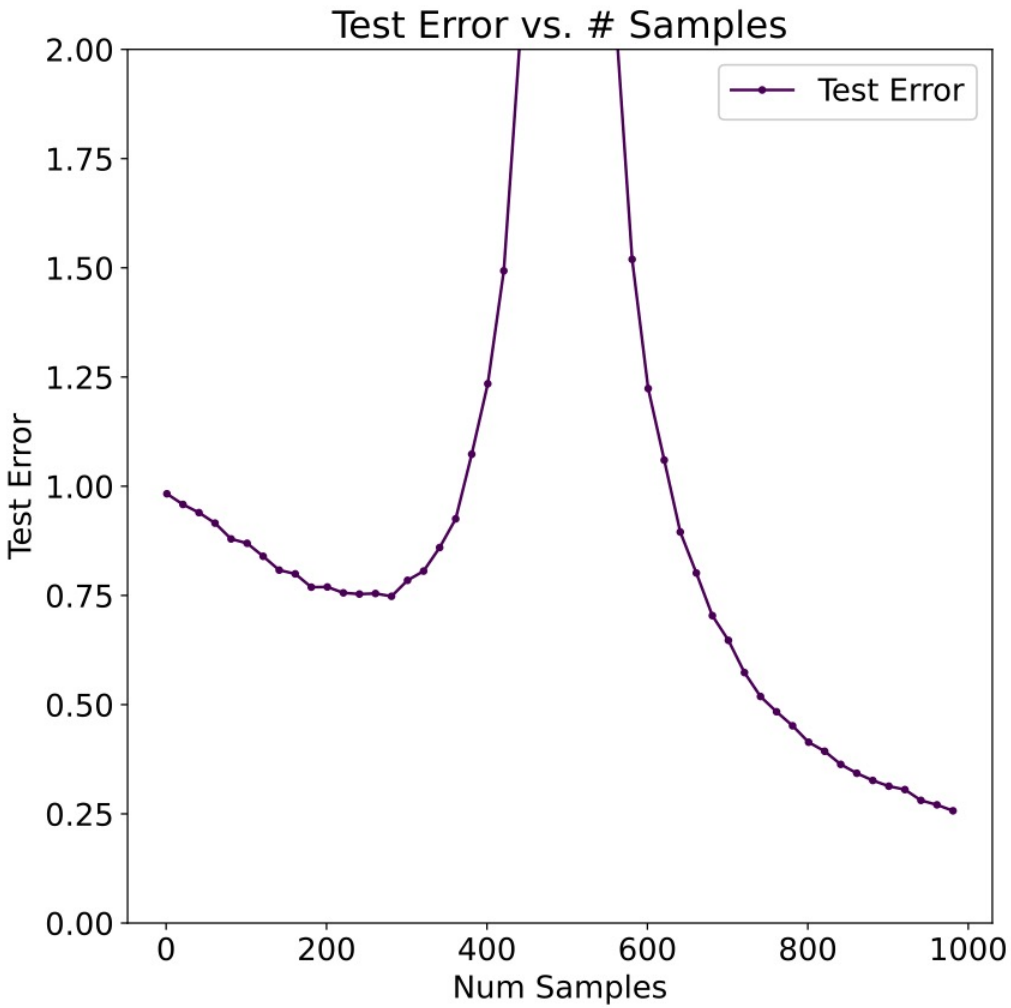
Model-wise double descent

- Recent works demonstrated that the test error can present a "double descent" phenomenon in a range of machine learning models including linear models and deep neural networks



Sample-wise double descent

- Recent work observes that the test error is **not monotonically decreasing** when the sample size increases
  - The test error first decreases
  - Then increases and peaks around when the number of examples is similar to the number of parameters ($n \approx d$)
  - And then decreases again
  - Sample-wise double descent and model-wise double descent are essentially describing similar phenomena—the test error is peaked when $n \approx d$

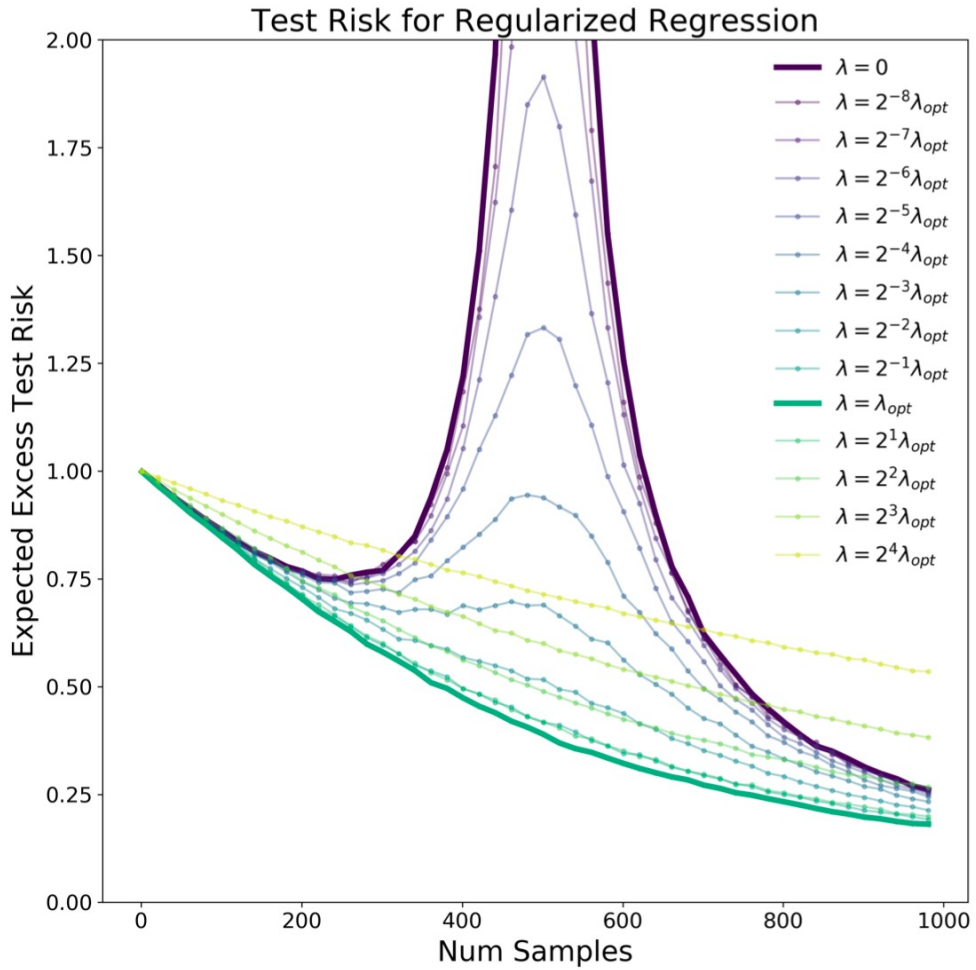


Explanation

- The observation illustrates that
  - Existing training algorithms evaluated in these experiments are far from optimal when $n \approx d$
- How to be better?
  - Tossing away some examples and run the algorithms with a smaller sample size to steer clear of the peak
  - With an optimally-tuned regularization, the test error in the $n \approx d$ regime can be dramatically improved

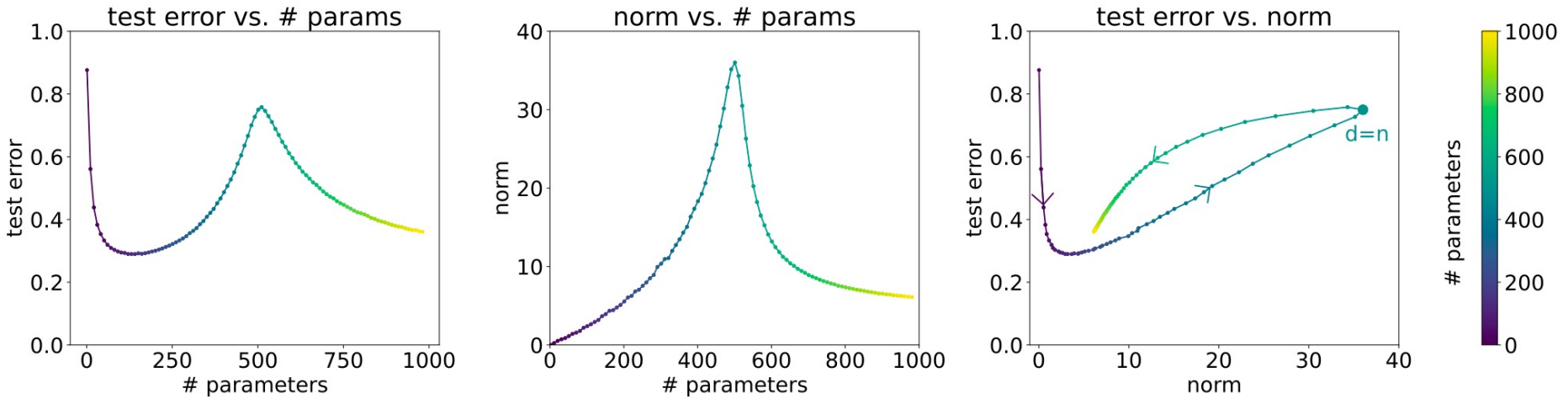
Regularization

- Using the optimal regularization parameter λ (optimally tuned for each n, shown in green solid curve) mitigates double descent



Complexity measure of the model

- The double descent phenomenon has been observed when the model complexity is measured by the number of parameters
- It is unclear if and when the number of parameters is the best complexity measure of a model



Implicit regularization effect

Explanation for overparameterization

- A typical explanation
  - Commonly-used optimizers such as gradient descent provide an implicit regularization effect
  - Intuition: even in the overparameterized regime and with an unregularized loss function, the model is still implicitly regularized, and thus exhibits a better test performance than an arbitrary solution that fits the data.

Intuition of implicit regularization effect

- In most classic settings
  - The optimal solution is unique
  - Any reasonable optimizer should converge to this point
- In deep learning
  - There are usually more than one (approximate) global minimum
  - Different optimizers may converge to different global minima
  - Though they have similar training loss
    - The solution may have dramatically different generalization performance

Illustration

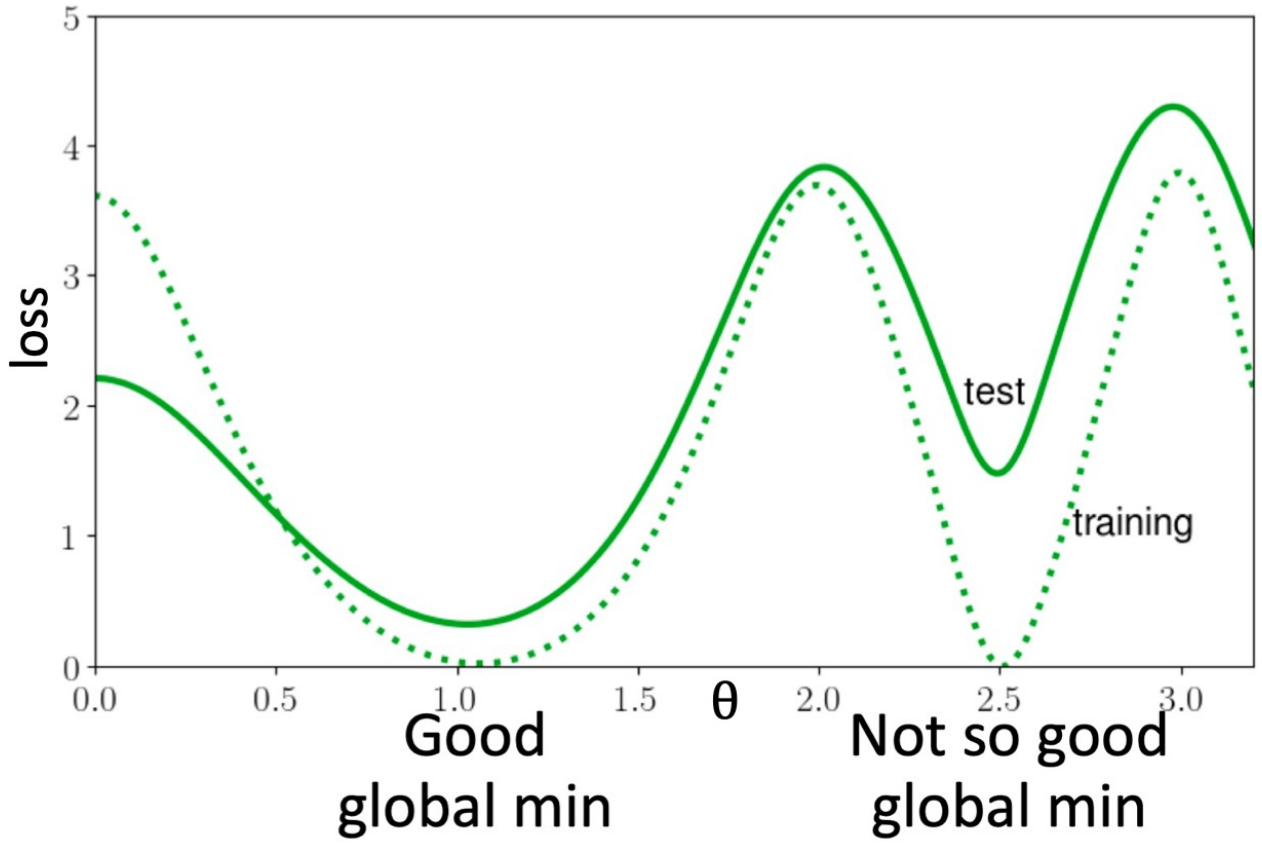


Illustration (cont'd)

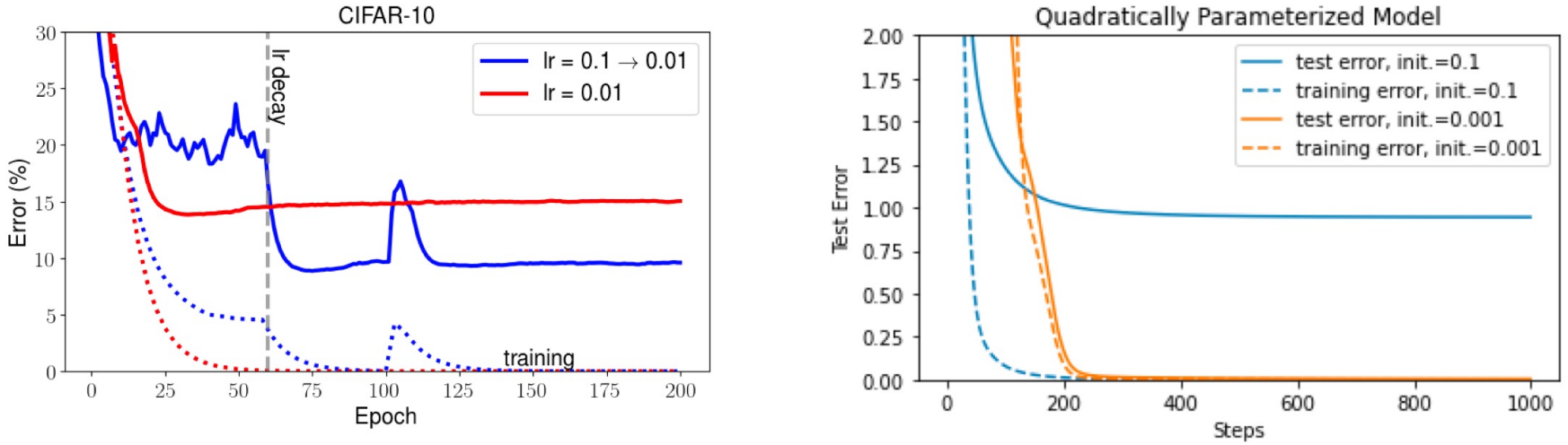


Figure 9.2: **Left:** Performance of neural networks trained by two different learning rates schedules on the CIFAR-10 dataset. Although both experiments used exactly the same regularized losses and the optimizers fit the training data perfectly, the models' generalization performance differ much. **Right:** On a different synthetic dataset, optimizers with different initializations have the same training error but different generalization performance.[4]

What type of global minima may generalize better?

- Still an active research area

- Some heuristics
  - Larger initial learning rate
  - Smaller initialization
  - Smaller batch size
  - Introducing momentum

Implicitly regularization of GD

定理 8.3 (过参数线性回归的隐式正则化). 在过参数线性回归 (*MSE* 损失) 中，如果使用 *0* 初始化的梯度下降法且训练误差降为 *0*，则最终优化到的解 $\hat{\boldsymbol{w}}$ 是最小范数解，即在集合 $\{\boldsymbol{w} : X\boldsymbol{w} = Y\}$ 中，$\hat{\boldsymbol{w}}$ 具有最小的二范数。

- Initialization: $w_0 = 0 = X^\top 0$
- Update: $w_{t+1} = w_t - \eta X^\top (Y - X^\top \mathrm{w}_t)$
- Final convergence: $\widehat{w} = X^\top v$
- Training loss = 0 $\Longrightarrow Y = X\widehat{w} = XX^\top v \Longrightarrow \widehat{w} = X^\top (XX^\top)^{-1} Y$
- For all other solution w' satisfying $Y = Xw'$
  - $X(\mathrm{w}' - \widehat{w}) = 0 \implies \widehat{w}(\mathrm{w}' - \widehat{w}) = 0$
  - It holds that $\|w'\| - \|\widehat{w}\| \geq 0$

https://www.tengjiaye.com/book/TML.pdf#page=60.15

Bayesian statistics and regularization

Frequentist V.S. Bayesian

- Consider θ as the model parameter
- Frequentist view
  - θ is constant-valued but unknown
  - We need to estimate this parameter, such as MLE

$$\theta_{\mathrm{MLE}} = \arg\max_{\theta} \prod_{i=1}^{n} p(y^{(i)}|x^{(i)};\theta).$$

- Bayesian review
  - θ is a random variable with unknown value
  - We can specify a prior distribution $p(\theta)$ on θ that expresses our "prior beliefs" about the parameters

Bayesian view

- Given a training set $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
- Compute the posterior of θ

$$
\begin{aligned}
p(\theta|S) &= \frac{p(S|\theta)p(\theta)}{p(S)} \\
&= \frac{\left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)\right) p(\theta)}{\int_{\theta} \left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)p(\theta)\right) d\theta}
\end{aligned}
$$

- To predict the label of a new data x

$$
p(y|x, S) = \int_{\theta} p(y|x, \theta)p(\theta|S)d\theta \qquad \mathrm{E}[y|x, S] = \int_{y} yp(y|x, S)dy
$$

Bayesian view (cont'd)

- Given a training set $S = \{(x^{(i)}, y^{(i)})\}_{i=1}^{n}$
- Compute the posterior of θ

$$p(\theta|S) = \frac{p(S|\theta)p(\theta)}{p(S)}$$

$$= \frac{\left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)\right) p(\theta)}{\int_{\theta} \left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)p(\theta)\right) d\theta}$$

Computationally difficult!

- To predict the label of a new data x

$$p(y|x, S) = \int_{\theta} p(y|x, \theta)p(\theta|S)d\theta \qquad \mathrm{E}[y|x, S] = \int_{y} yp(y|x, S)dy$$

Maximum a posteriori (MAP)

- Approximate the posterior distribution for θ
- Use single point estimate

$$\theta_{\text{MAP}} = \arg\max_{\theta} \prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta) p(\theta)$$

$$\begin{aligned} p(\theta|S) &= \frac{p(S|\theta)p(\theta)}{p(S)} \\ &= \frac{\left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)\right) p(\theta)}{\int_{\theta} \left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)}, \theta)p(\theta)\right) d\theta} \end{aligned}$$

Maximum a posteriori (MAP)

- Approximate the posterior distribution for θ
- Use single point estimate

$$p(\theta|S) = \frac{p(S|\theta)p(\theta)}{p(S)} = \frac{\left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)\right)p(\theta)}{\int_\theta \left(\prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)p(\theta)\right)d\theta}$$

$$\theta_{\text{MAP}} = \arg\max_\theta \prod_{i=1}^{n} p(y^{(i)}|x^{(i)},\theta)p(\theta)$$

Additional term compared with MLE

- The prior $p(\theta)$ is usually assumed to be $\theta \sim \mathcal{N}(0, \tau^2 I)$
- Parameters with smaller norm are more preferred than MLE
- Less susceptible to overfitting

Summary

- Intuition
- Bias-variance tradeoff
- Sample complexity bounds
  - Finite hypothesis class
  - Infinite hypothesis class
- The double descent phenomenon
  - Model-wise double descent
  - Sample-wise double descent
- Implicit regularization effect
- Bayesian statistics and regularization