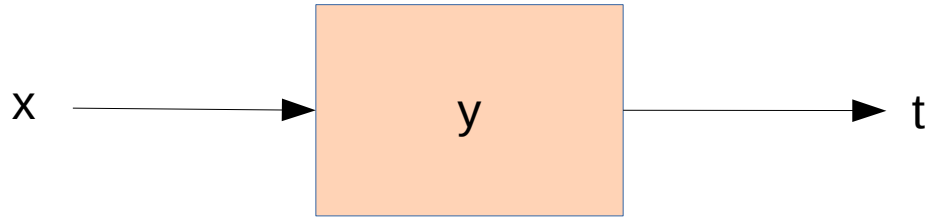
Introduction to Machine Learning

Lecture 2
Fundamentals I

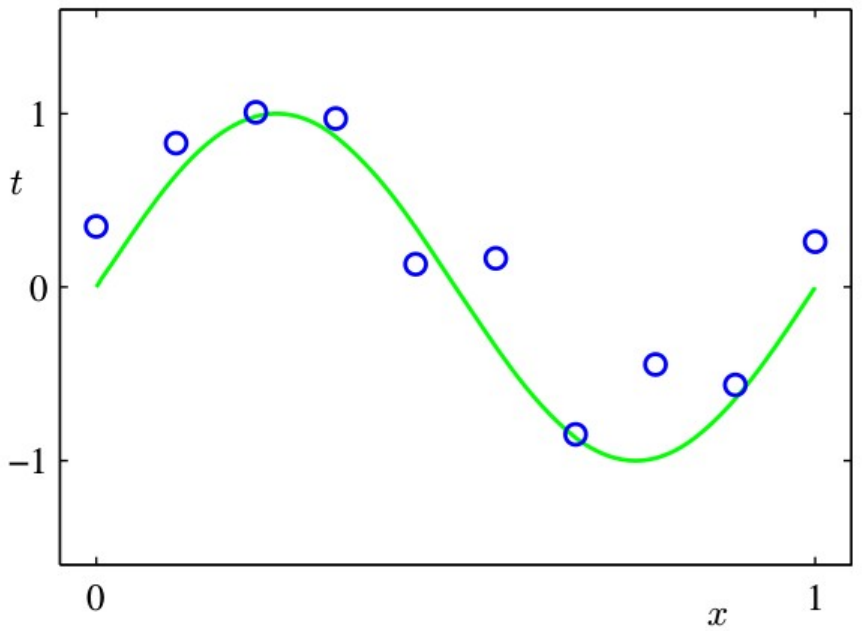
Goker Erdogan
26 – 30 November 2018
Pontificia Universidad Javeriana

Setup



- Given N samples (training set) of {x, t}, learn function y
  - So we can predict t for a new x

Plot of a training data set of $N = 10$ points, shown as blue circles, each comprising an observation of the input variable x along with the corresponding target variable t. The green curve shows the function $\sin(2\pi x)$ used to generate the data. Our goal is to predict the value of t for some new value of x, without knowledge of the green curve.



A motivating example: Polynomial curve fitting[1]

- Need to assume some (parametric) form for function y
  - Assume it is a polynomial of order M

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

  - where w are the parameters (to learn from data)

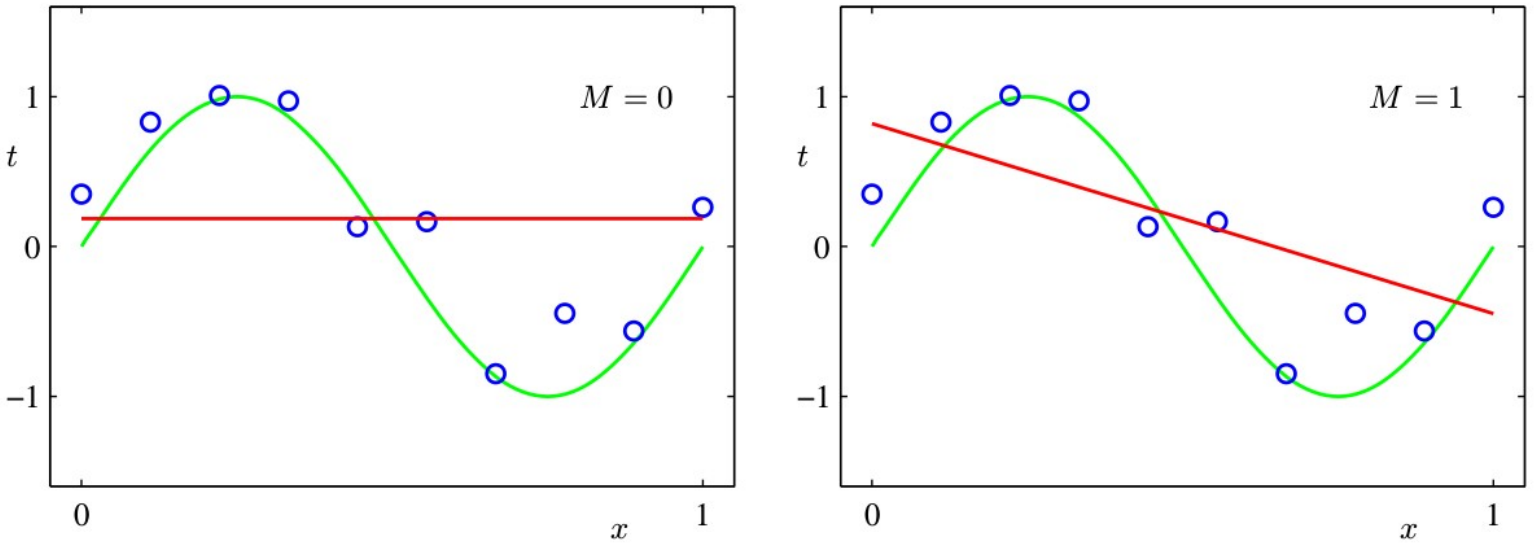
- How do we choose w? What makes a set of w better than another?
  - Define an *error function*

$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

  - Find w* that minimizes the error E(w)
  - (there is a closed form solution in this case)

A motivating example contd.

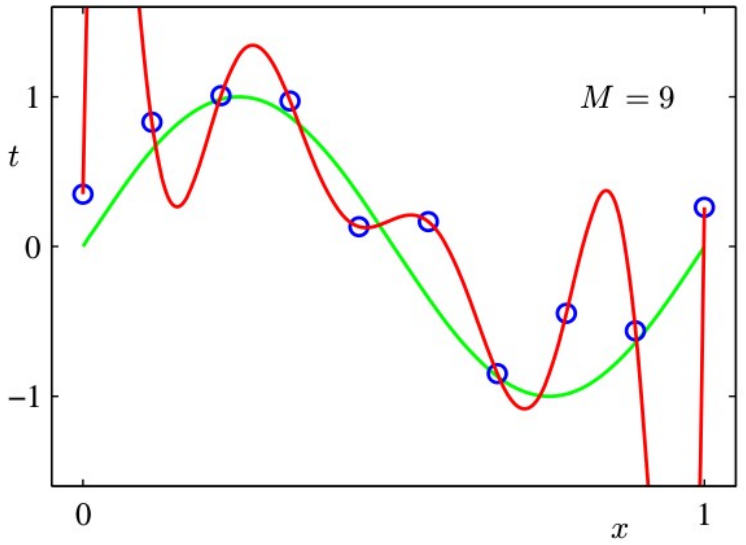
- How do we choose M (the order of polynomial)?
  - An important problem in ML: *model selection*



  - M=0 and M=1 are probably too small
    - Large training error
    - Cannot capture the pattern in data: *underfitting*

Overfitting and underfitting

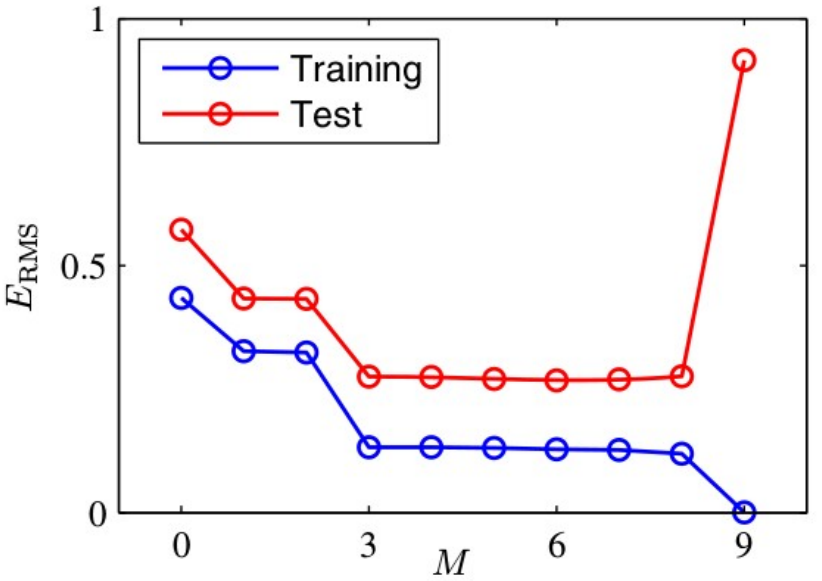
- How about M=9?
  - Zero training error!
  - However, does not generalize well to test data
  - Model with M=9 is too flexible and it learned to capture the noise
  - This is known as *overfitting*

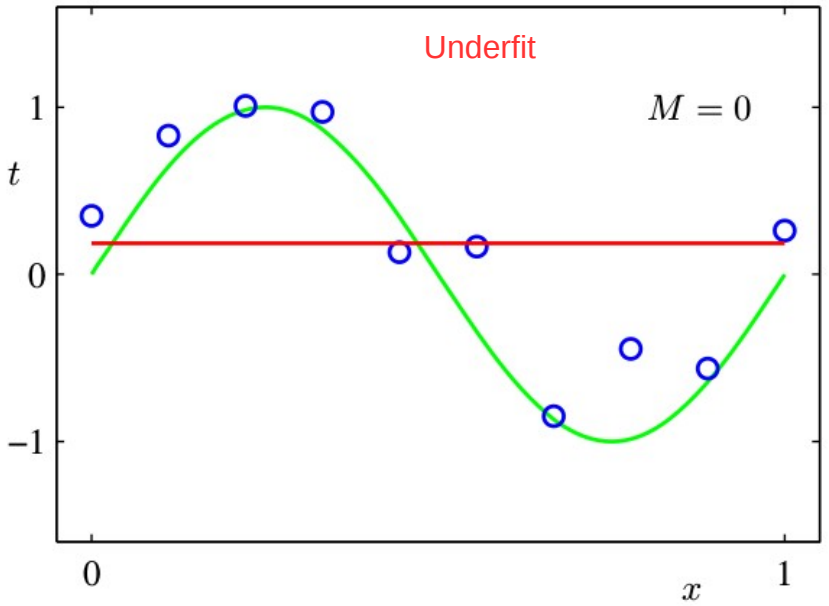


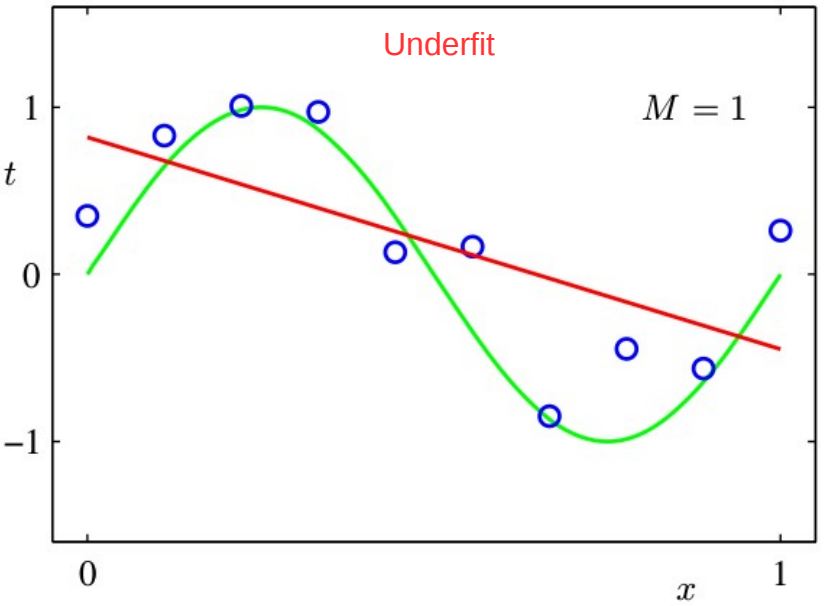
Overfitting and underfitting contd.

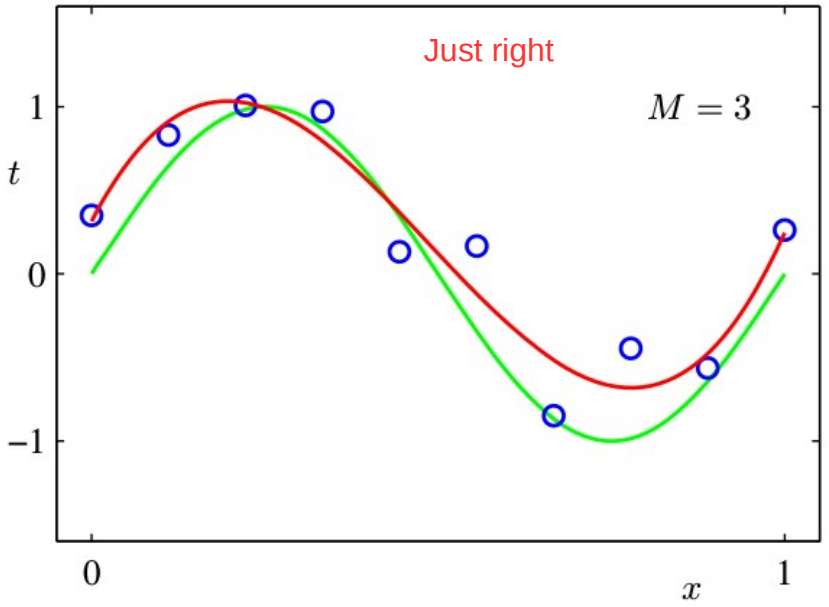
- Let's look at generalization error as we increase M
  - Assume we have a separate test set of {x, t}
  - Evaluate model on training and test sets and plot $E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N}$
  - This is a simple form of *model validation*

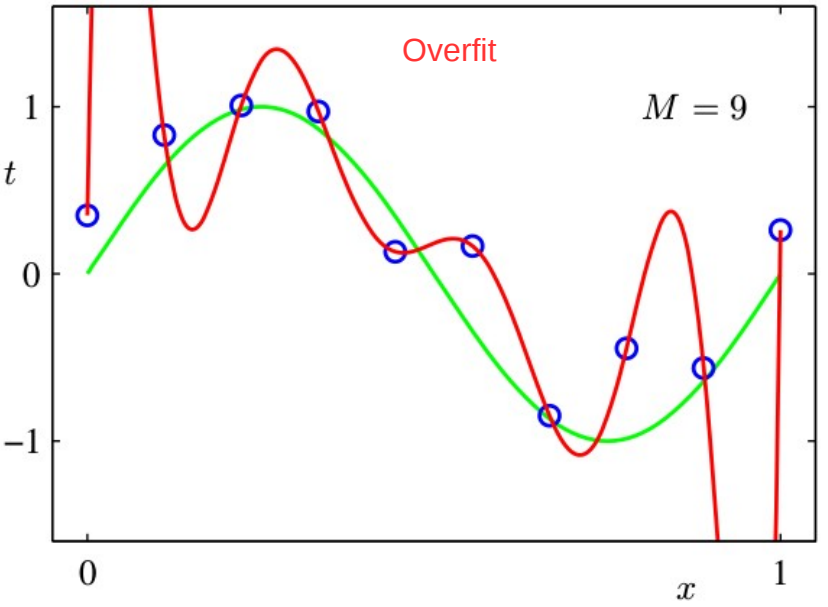
Graphs of the root-mean-square error, defined by (1.3), evaluated on the training set and on an independent test set for various values of M.




Underfit
$M = 0$
1
0
−1
t
0
x
1


Underfit
$M = 1$
1
0
−1
t
0
x
1


Just right
$M = 3$
1
0
−1
t
0
x
1


Overfit
$M = 9$
1
0
−1
t
0
x
1

What is going on?

- There are variations in the data that you don't want to learn (e.g., noise)
- More complex model → Captures training data better
  - Training error will go down as the model gets more complex
  - But this is not a good measure of the generalization performance
    - Your model might be fitting the noise in the data
  - There is a sweet spot in terms of model complexity

Table of the coefficients $\mathbf{w}^\star$ for polynomials of various order. Observe how the typical magnitude of the coefficients increases dramatically as the order of the polynomial increases.

	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

More data always helps

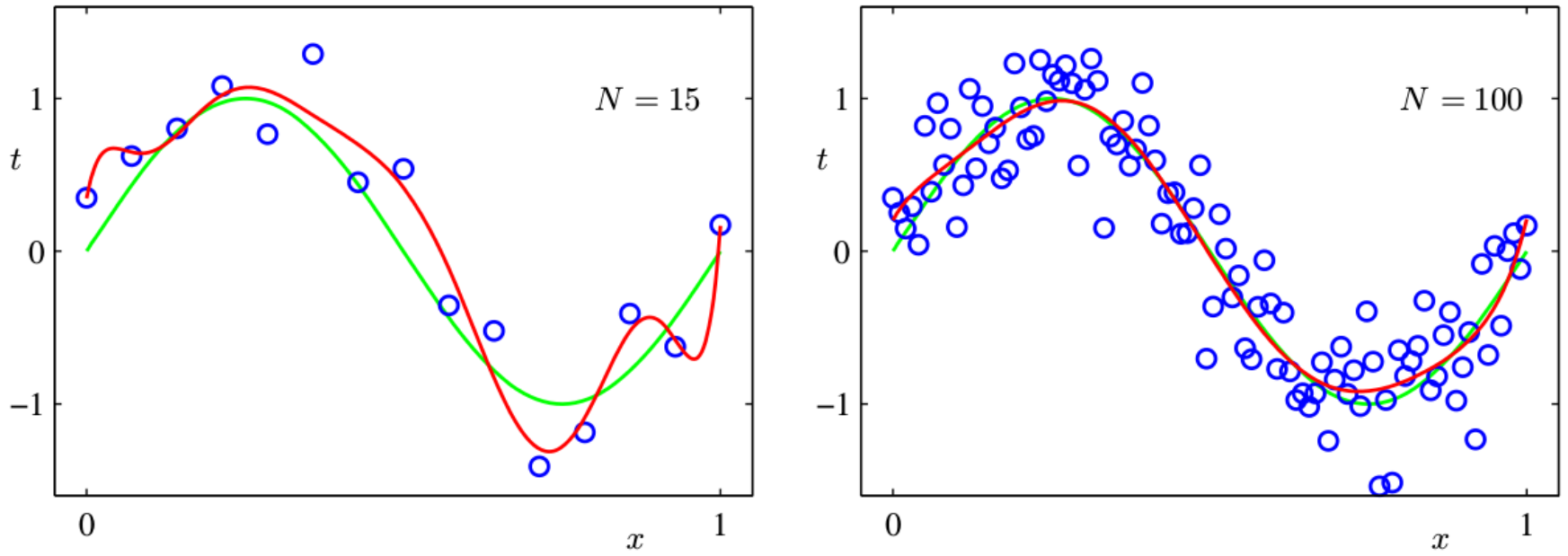


Figure 1.6 Plots of the solutions obtained by minimizing the sum-of-squares error function using the $M = 9$ polynomial for $N = 15$ data points (left plot) and $N = 100$ data points (right plot). We see that increasing the size of the data set reduces the over-fitting problem.

Bias-variance tradeoff[2,3]

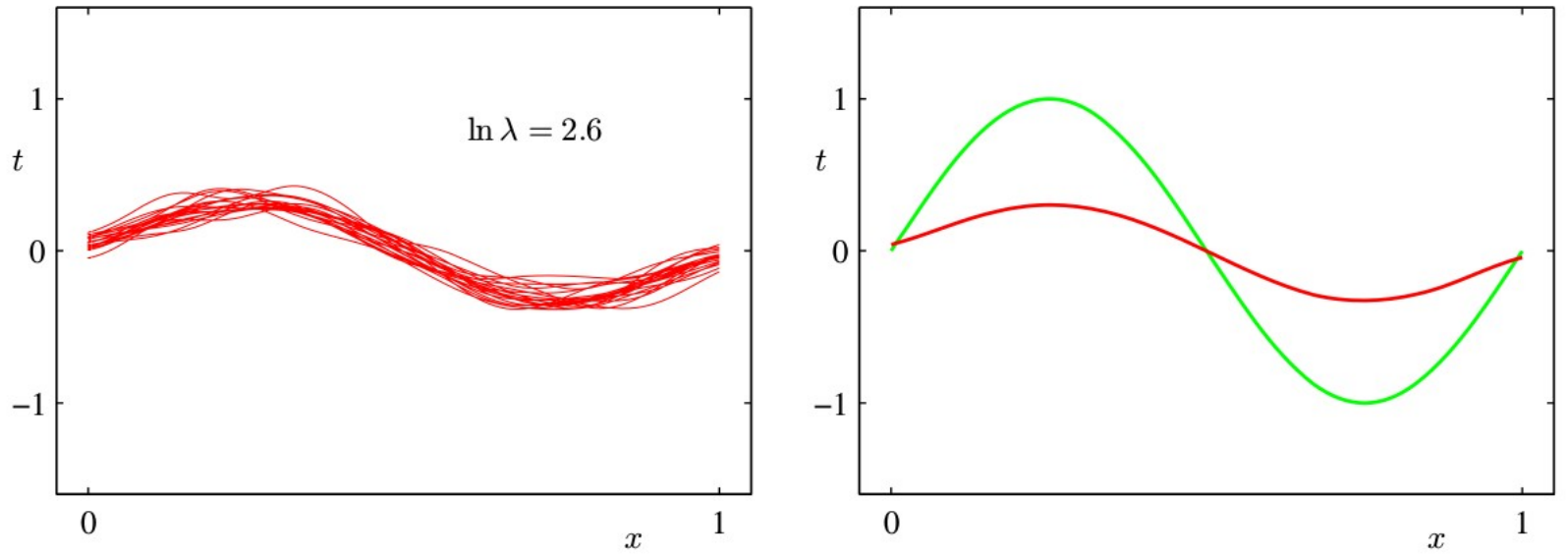
- One way to understand the overfitting problem
- Error = Bias^2 + Variance

$$\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - h(\mathbf{x})\}^2\right] = \underbrace{\{\mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})] - h(\mathbf{x})\}^2}_{(\text{bias})^2} + \underbrace{\mathbb{E}_{\mathcal{D}}\left[\{y(\mathbf{x};\mathcal{D}) - \mathbb{E}_{\mathcal{D}}[y(\mathbf{x};\mathcal{D})]\}^2\right]}_{\text{variance}}.$$

- Here D denotes a random training dataset (h: target function)
  - Imagine learning the model for each random dataset
  - For each dataset, we will have a slightly different model that makes slightly different predictions
- Bias: how much the average prediction over all data sets differs from the desired regression function
  - How well can y model the training data?
- Variance: how much the solutions for individual data sets vary around their average
  - How much the function y(x; D) is sensitive to the particular choice of data set

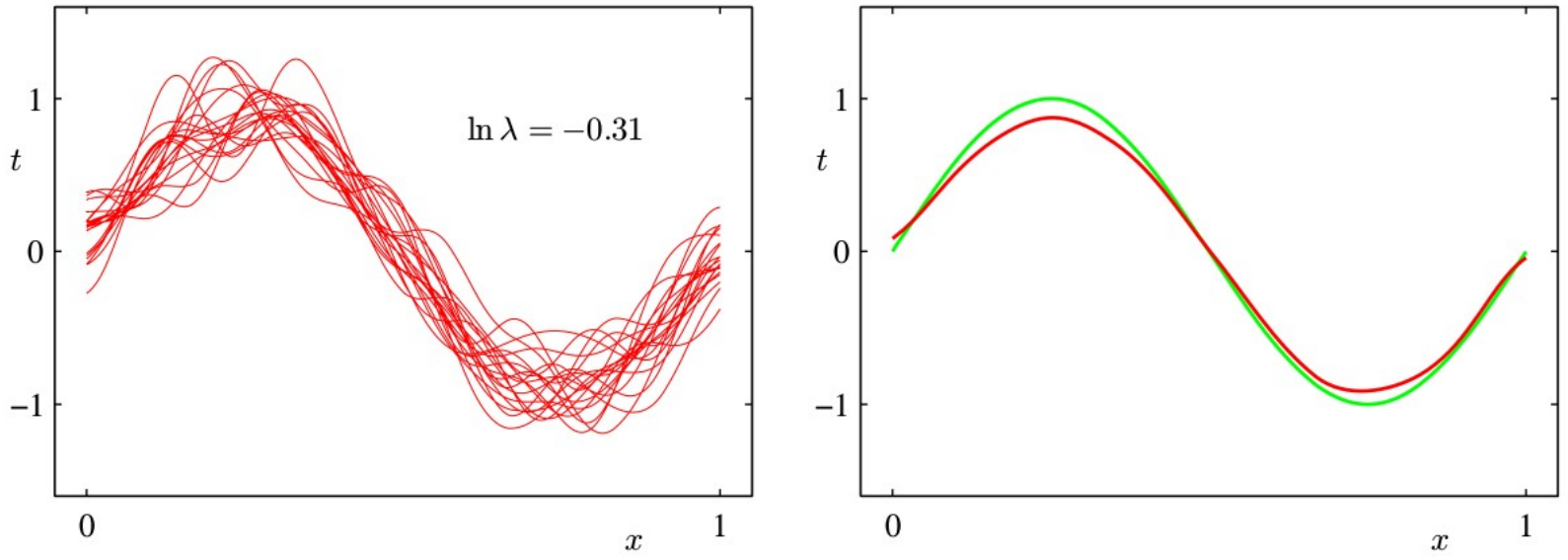
Bias-variance vs model complexity

- In general, as a model gets more complex
  - Bias decreases
  - Variance increases



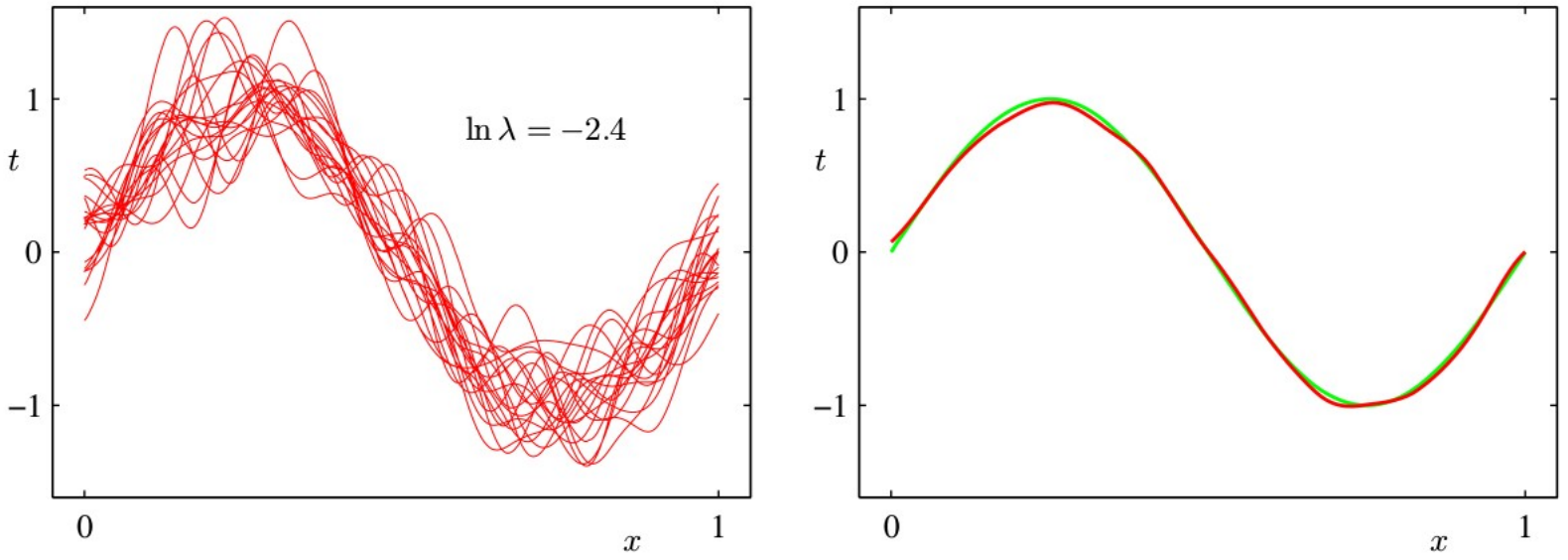
Bias-variance vs model complexity

- In general, as a model gets more complex
  - Bias decreases
  - Variance increases

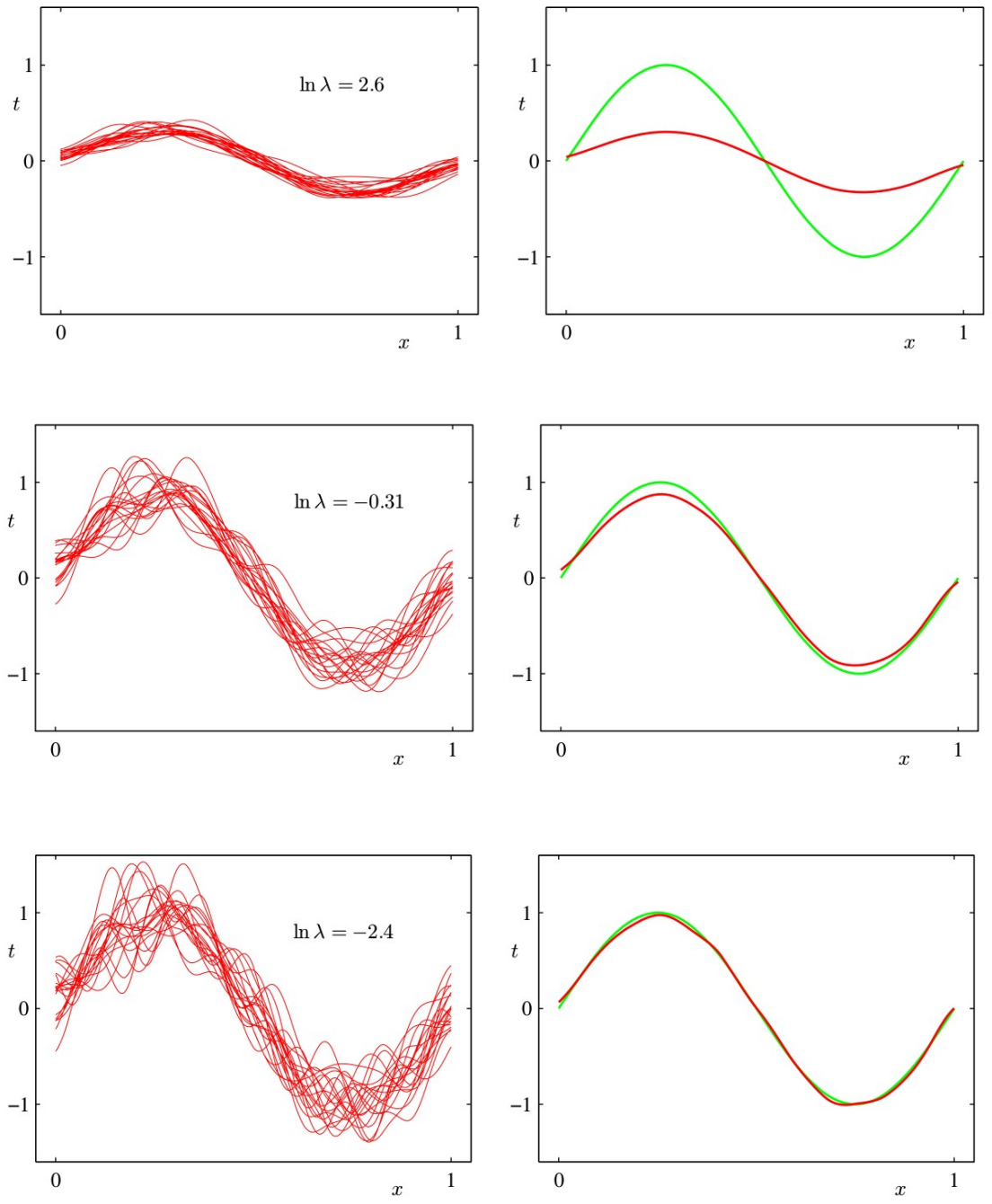


Bias-variance vs model complexity

- In general, as a model gets more complex
  - Bias decreases
  - Variance increases

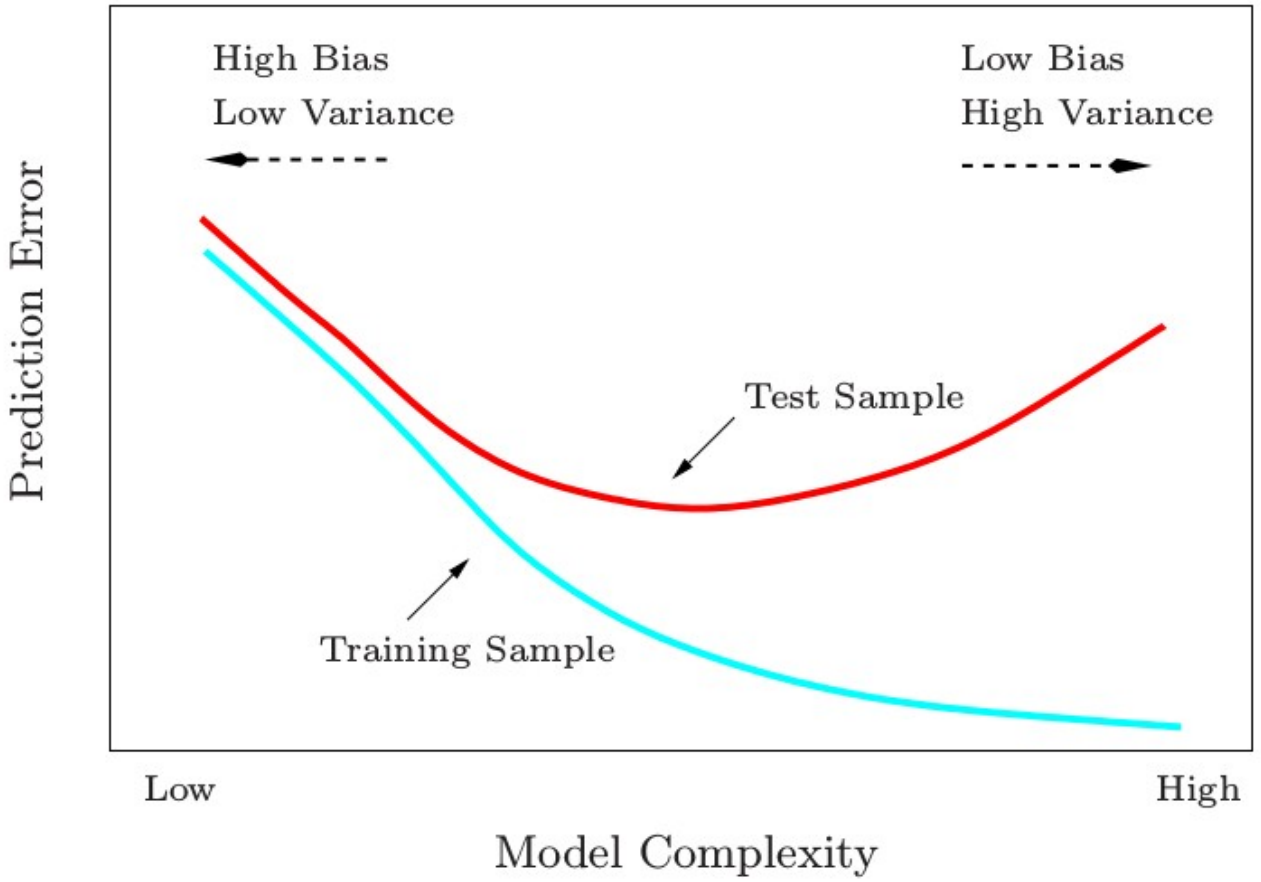


More complex



Bias-variance tradeoff and generalization

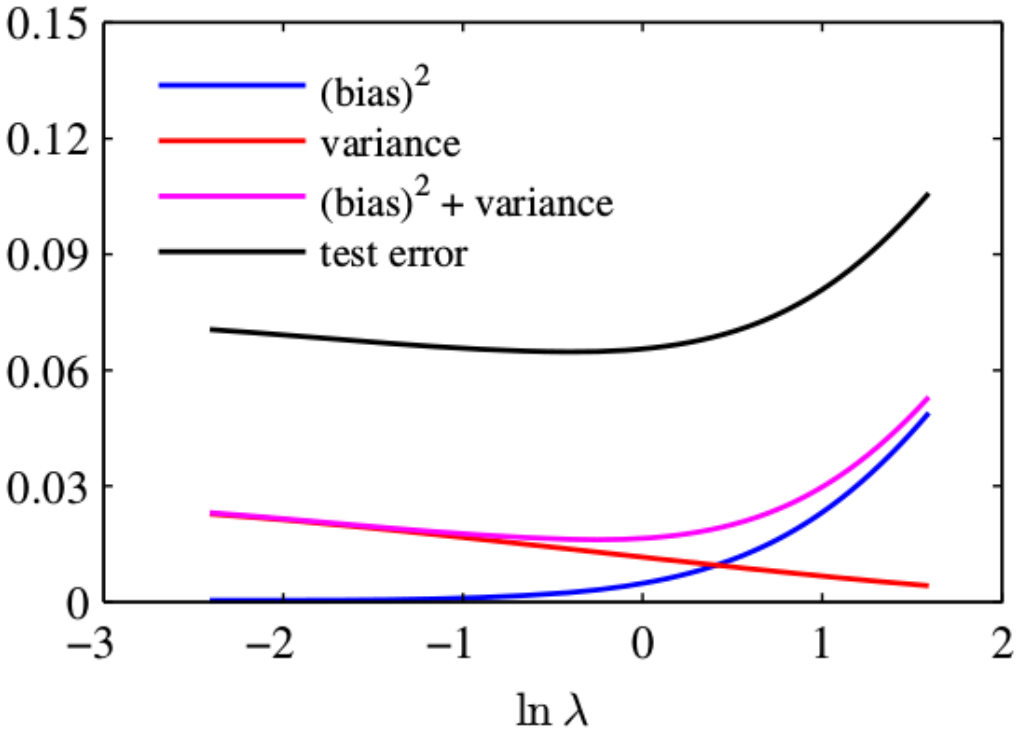
- As model gets more complex
  - Bias decreases and training error decreases
  - Variance increases and dominates test error after a point



Bias-variance tradeoff and generalization contd.

- Looking at the polynomial curve fitting example

Plot of squared bias and variance, together with their sum, corresponding to the results shown in Figure 3.5. Also shown is the average test set error for a test data set size of 1000 points. The minimum value of $(\text{bias})^2 + \text{variance}$ occurs around $\ln \lambda = -0.31$, which is close to the value that gives the minimum error on the test data.



More complex

Summary

- General setup
  - Given some training data
  - Find the function that minimizes some measure of error
- Overfitting and underfitting
  - Model too simple, does not capture patterns in the data: underfitting
  - Model too complex, fit the noise in data: overfitting
- Bias-variance tradeoff
  - We can talk about the bias and variance of a model
  - Bias: How well the model can model the training data
  - Variance: How sensitive model predictions are to training dataset
  - Error = Bias^2 + Variance

References

[1] Bishop C. Pattern Recognition and Machine Learning. Section 1.1

[2] Bishop C. Pattern Recognition and Machine Learning. Section 3.2

[3] Hastie, Tibshirani, and Friedman. The Elements of Statistical Learning. Section 2.9