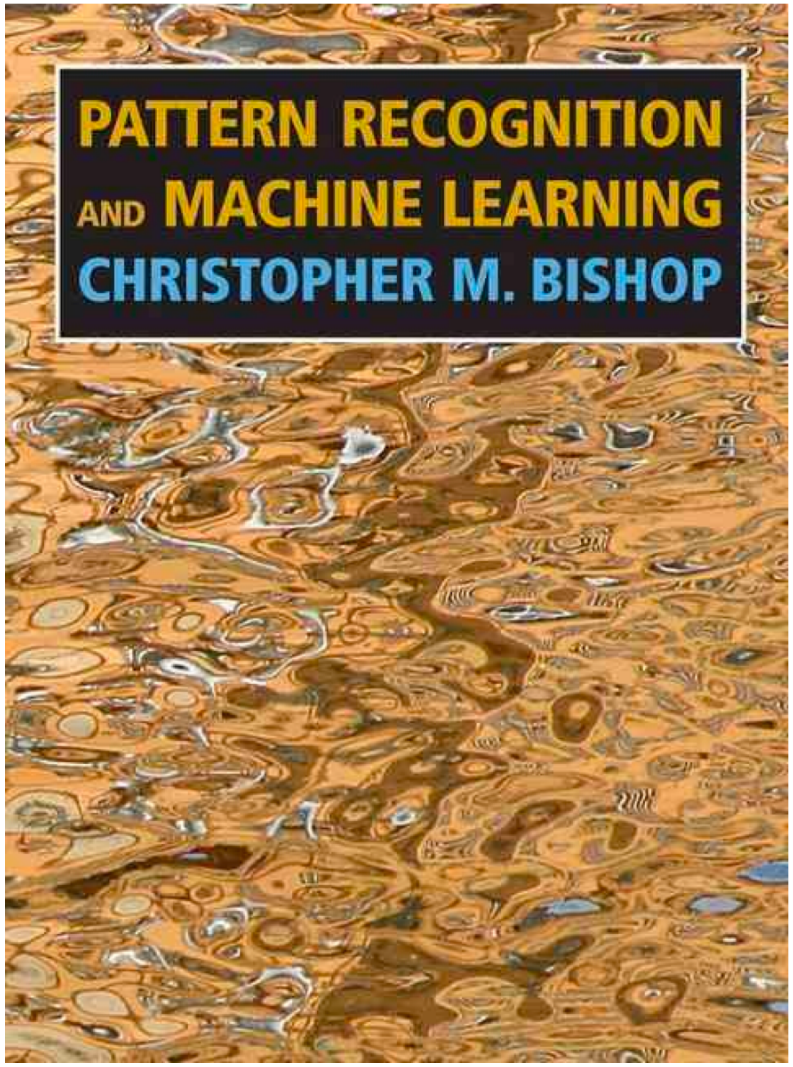
國立交通大學
National Chiao Tung University

Machine Learning
機器學習
Spring 2018

Introduction
(Chapter 1.1)

Prof. Chia-Han Lee
李佳翰 副教授

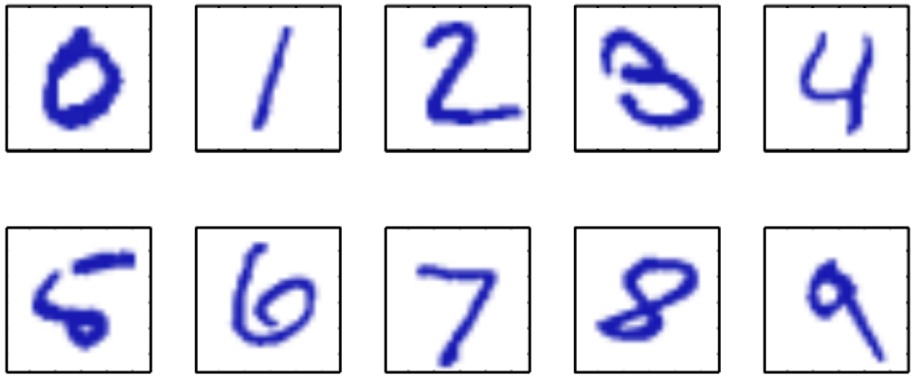


Pattern recognition

- **The field of pattern recognition is concerned with the automatic discovery of regularities in data through the use of computer algorithms and with the use of these regularities to take actions such as classifying the data into different categories.**

Hand-written digits

- Consider the example of recognizing handwritten digits. Each digit corresponds to a 28×28 pixel image and so can be represented by a vector $\mathbf{x}$ comprising 784 real numbers.

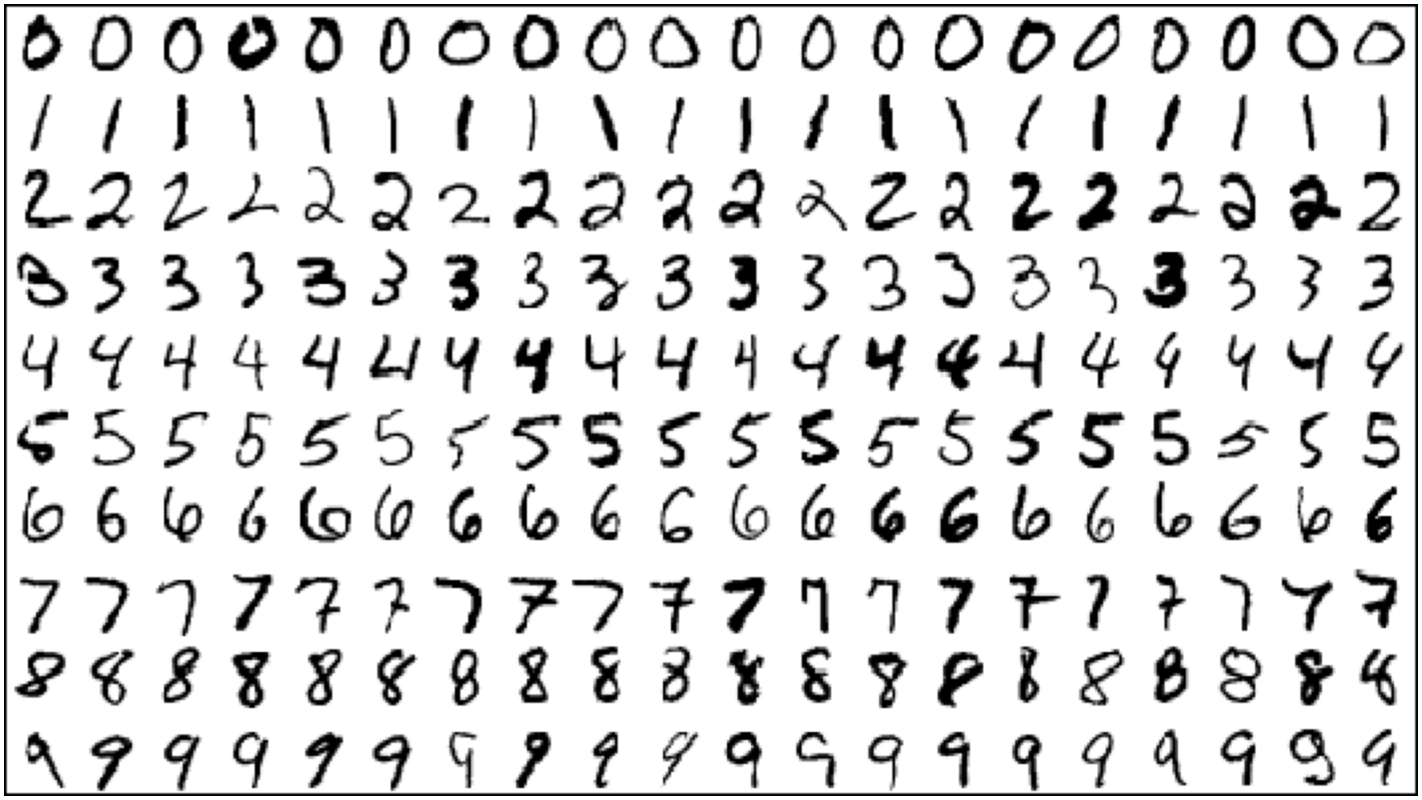


- The goal is to build a machine that will take such a vector $\mathbf{x}$ as input and that will produce the identity of the digit 0,…,9 as the output.

Machine learning approach

- A large set of N digits $\{\mathbf{x_1}, \ldots, \mathbf{x_N}\}$ called a **training set** is used to tune the parameters of an adaptive model.
- The categories of the digits in the training set are known in advance, typically by inspecting them individually and hand-labelling them.
- Express the category of a digit using **target vector** $\mathbf{t}$, which represents the identity of the corresponding digit.
- There is one such target vector $\mathbf{t}$ for each digit image $\mathbf{x}$.

Machine learning approach



http://myselph.de/mnistExamples.png

Machine learning approach

- The result of running the machine learning algorithm can be expressed as a function $\mathbf{y}(\mathbf{x})$ which takes a new digit image $\mathbf{x}$ as input and that generates an output vector $\mathbf{y}$, encoded in the same way as the target vectors.

- The precise form of the function $\mathbf{y}(\mathbf{x})$ is determined during the training phase, also known as the learning phase, on the basis of the training data.

- Once the model is trained it can then determine the identity of new digit images, which are said to comprise a test set.

Generalization

- The ability to categorize correctly new examples that differ from those used for training is known as generalization.

- In practical applications, the variability of the input vectors will be such that the training data can comprise only a tiny fraction of all possible input vectors, and so generalization is a central goal in pattern recognition.

Pre-processing

- For most practical applications, the original input variables are typically preprocessed to transform them into some new space of variables where the pattern recognition problem will be easier to solve.

- For instance, in the digit recognition problem, the images of the digits are typically translated and scaled so that each digit is contained within a box of a fixed size. This greatly reduces the variability within each digit class, because the location and scale of all the digits are now the same, which makes it much easier for a subsequent pattern recognition algorithm to distinguish between the different classes.

Feature extraction

- The aim of feature extraction is to find useful features that are fast to compute, and yet that also preserve useful discriminatory information.
- Because the number of such features is smaller than the number of pixels, this kind of pre-processing represents a form of dimensionality reduction.
- Care must be taken during pre-processing because often information is discarded, and if this information is important to the solution of the problem then the overall accuracy of the system can suffer.

Supervised learning

- Applications in which the training data comprises examples of the input vectors along with their corresponding target vectors are known as **supervised learning** problems.

- Cases such as the digit recognition example, in which the aim is to assign each input vector to one of a finite number of discrete categories, are called **classification** problems.

- If the desired output consists of one or more continuous variables, then the task is called **regression**.

Unsupervised learning

- In other pattern recognition problems, the training data consists of a set of input vectors $\mathbf{x}$ without any corresponding target values.

- The goal in such unsupervised learning problems may be
  - to discover groups of similar examples within the data, where it is called clustering,
  - or to determine the distribution of data within the input space, known as density estimation,
  - or to project the data from a high-dimensional space down to two or three dimensions for the purpose of visualization.

Reinforcement learning

- The technique of reinforcement learning is concerned with the problem of finding suitable actions to take in a given situation in order to maximize a reward.

- The learning algorithm is not given examples of optimal outputs, in contrast to supervised learning, but must instead discover them by a process of trial and error.

- Typically there is a sequence of states and actions in which the learning algorithm is interacting with its environment. In many cases, the current action not only affects the immediate reward but also has an impact on the reward at all subsequent time steps.

Reinforcement learning

- A general feature of reinforcement learning is the trade-off between exploration, in which the system tries out new kinds of actions to see how effective they are, and exploitation, in which the system makes use of actions that are known to yield a high reward.

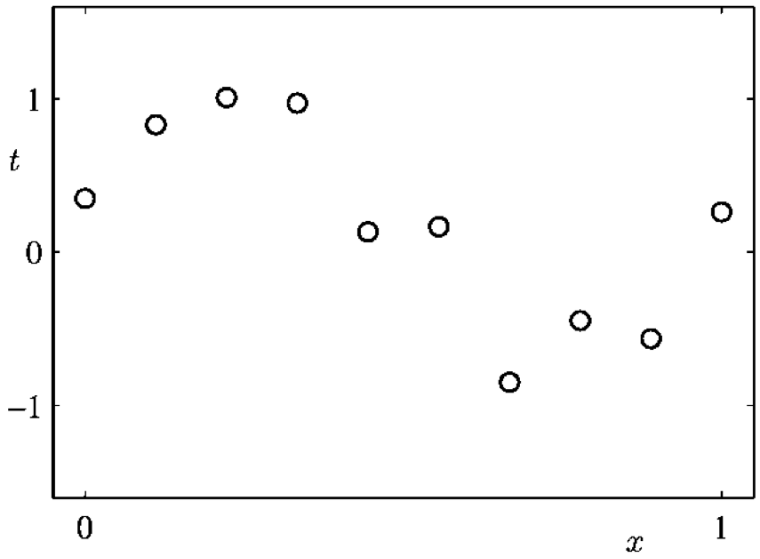
- Too strong a focus on either exploration or exploitation will yield poor results.

Polynomial curve fitting

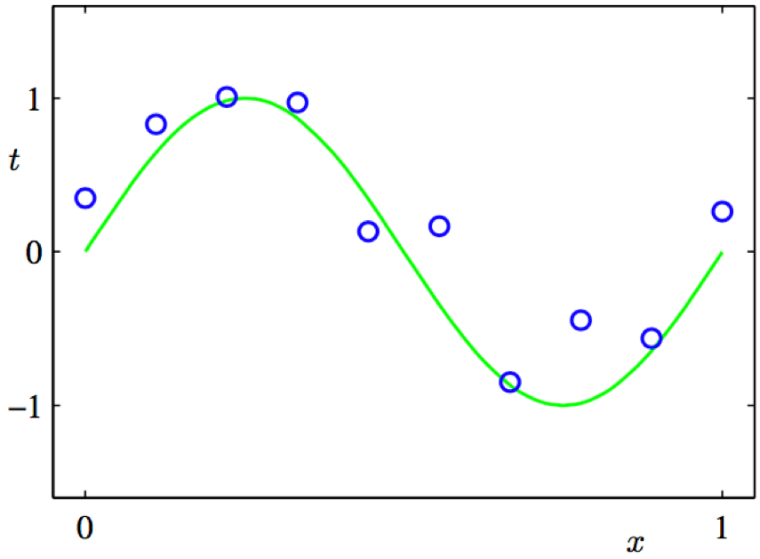
- **Suppose we observe a real-valued input variable x and we wish to use this observation to predict the value of a real-valued target variable t.**

- **Suppose that we are given a training set comprising N observations of x, written $\mathbf{x} \equiv (x_1, \ldots, x_N)^{\mathrm{T}}$, together with corresponding observations of the values of t, denoted $\mathbf{t} \equiv (t_1, \ldots, t_N)^{\mathrm{T}}$.**

Polynomial curve fitting



Polynomial curve fitting



- **The input data set x was generated by choosing values of x_n, $n = 1, \ldots, N$, spaced uniformly in range [0,1], and the target data set t was obtained by first computing the corresponding values of the function $\sin(2\pi x)$ and then adding a small level of random noise having a Gaussian distribution to each such point in order to obtain the corresponding value t_n.**

Polynomial curve fitting

- Our goal is to exploit this training set in order to make predictions of the value $\hat{t}$ of the target variable for some new value $\hat{x}$ of the input variable.

- This involves implicitly trying to discover the underlying function $\sin(2\pi x)$.

- This is intrinsically a difficult problem as we have to generalize from a finite data set. Furthermore the observed data are corrupted with noise, and so for a given $\hat{x}$ there is uncertainty as to the appropriate value for $\hat{t}$.

Polynomial curve fitting

- We fit the data using a polynomial function of the form

$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j \qquad (1.1)$$

where M is the order of the polynomial, and x^j denotes x raised to the power of j.

- The polynomial coefficients $\omega_0, \ldots, \omega_M$ are collectively denoted by the vector w. Note that, although the polynomial function $y(x, \mathbf{w})$ is a nonlinear function of x, it is a linear function of the coefficients w. Functions, which are linear in the unknown parameters, have important properties and are called linear models.

Polynomial curve fitting

- **The values of the coefficients will be determined by fitting the polynomial to the training data.**

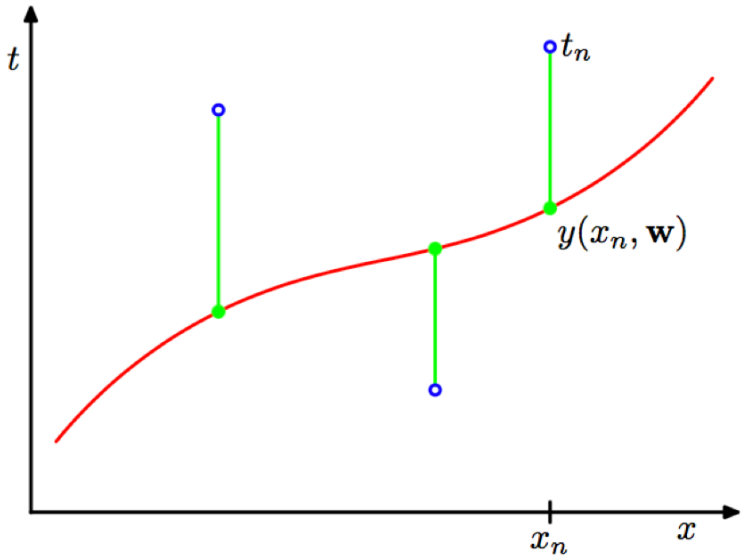
- **This can be done by minimizing an error function that measures the misfit between the function $y(x, \mathbf{w})$, for any given value of $\mathbf{w}$, and the training set data points.**

Polynomial curve fitting

- One simple choice of error function is given by the sum of the squares of the errors between the predictions $y(x_n, \mathbf{w})$ for each data point x_n and the corresponding target values t_n, so that we minimize

$$E(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 \qquad (1.2)$$

where the factor of 1/2 is included for convenience.



Polynomial curve fitting

- We can solve the curve fitting problem by choosing the value of $\mathbf{w}$ for which $E(\mathbf{w})$ is as small as possible.

- Because the error function is a quadratic function of the coefficients $\mathbf{w}$, its derivatives with respect to the coefficients will be linear in the elements of $\mathbf{w}$, and so the minimization of the error function has a unique solution, denoted by $\mathbf{w}^\star$, which can be found in closed form. The resulting polynomial is given by the function $y(x, \mathbf{w}^\star)$.

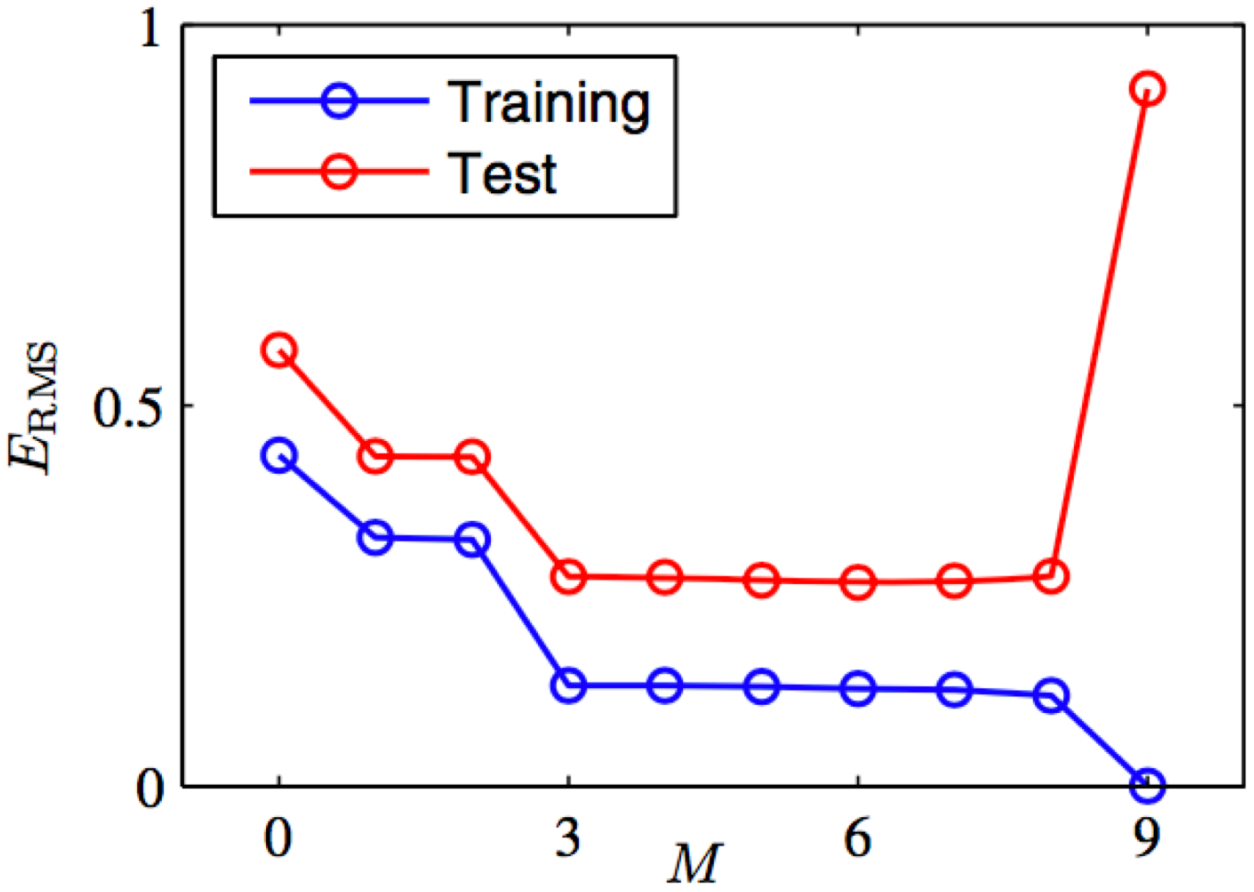
Model selection

- There remains the problem of choosing the order M of the polynomial, called **model comparison** or **model selection**.
- For each choice of M, we can then evaluate the residual value of $E(\mathbf{w}^\star)$ given by (1.2) for the training data, and we can also evaluate $E(\mathbf{w}^\star)$ for the test data set. It is sometimes more convenient to use the root-mean-square (RMS) error defined by

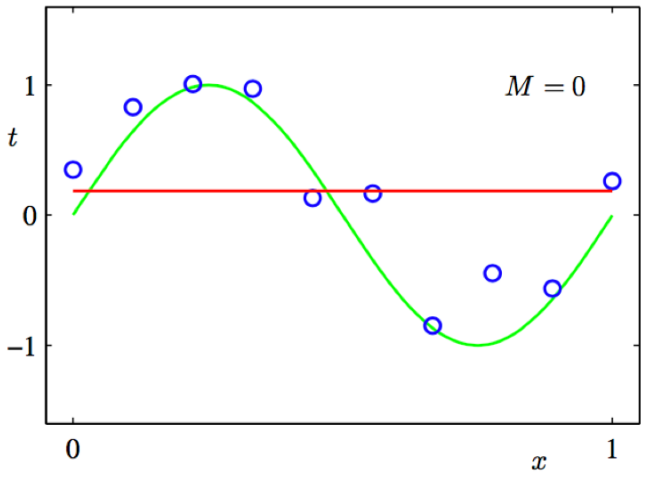
$$E_{\mathrm{RMS}} = \sqrt{2E(\mathbf{w}^\star)/N} \tag{1.3}$$

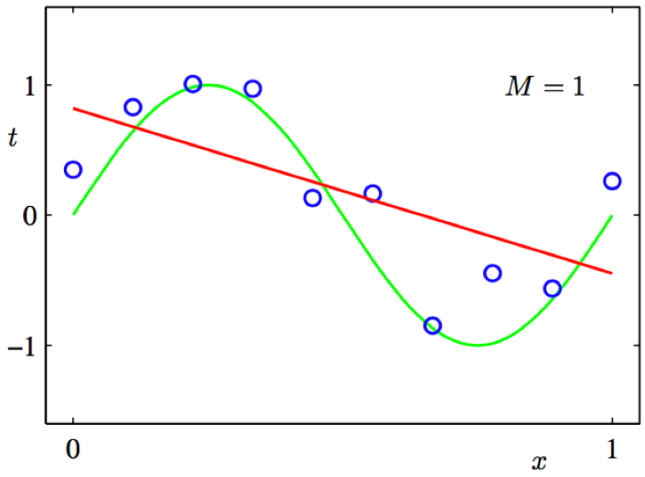
- The division by N allows us to compare different sizes of data sets on an equal footing, and the square root ensures that E_{RMS} is measured on the same scale (and in the same units) as the target variable t.

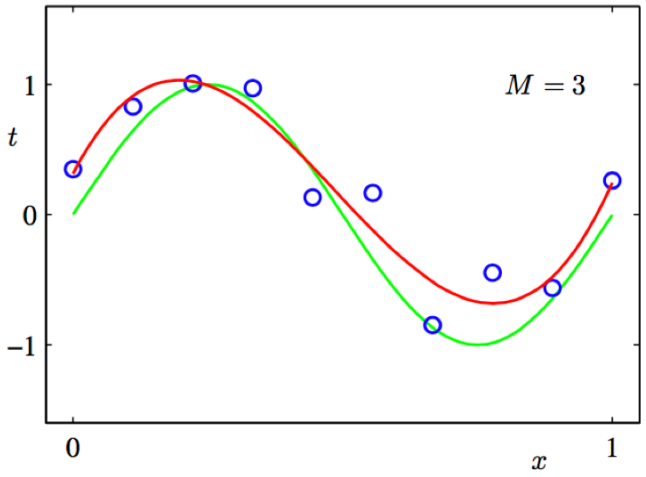
Model selection

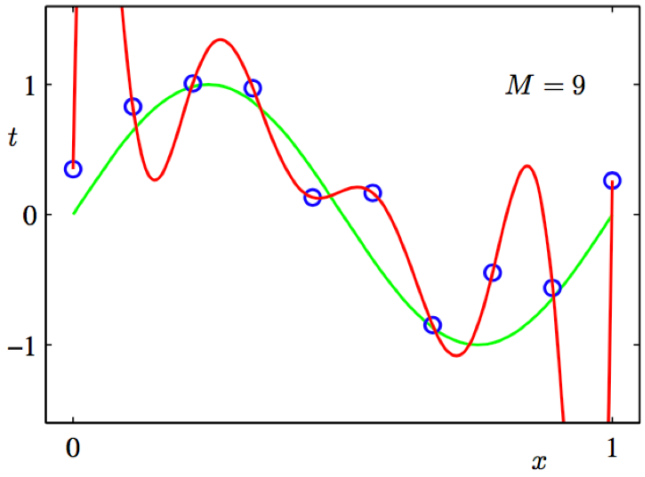


Model selection









Over-fitting

Over-fitting

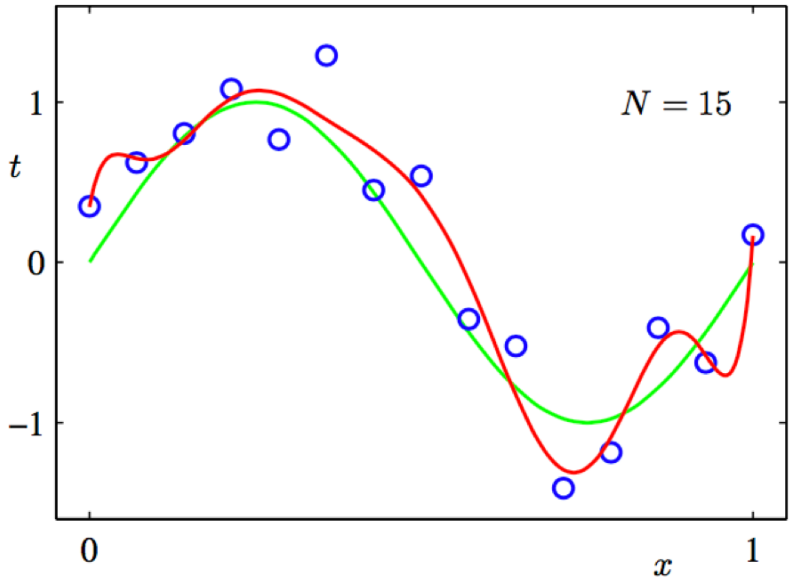
- As M increases, the magnitude of the coefficients typically gets larger. In particular for the M = 9 polynomial, the coefficients have become finely tuned to the data by developing large positive and negative values so that the corresponding polynomial function matches each of the data points exactly, but between data points the function exhibits the large oscillations.

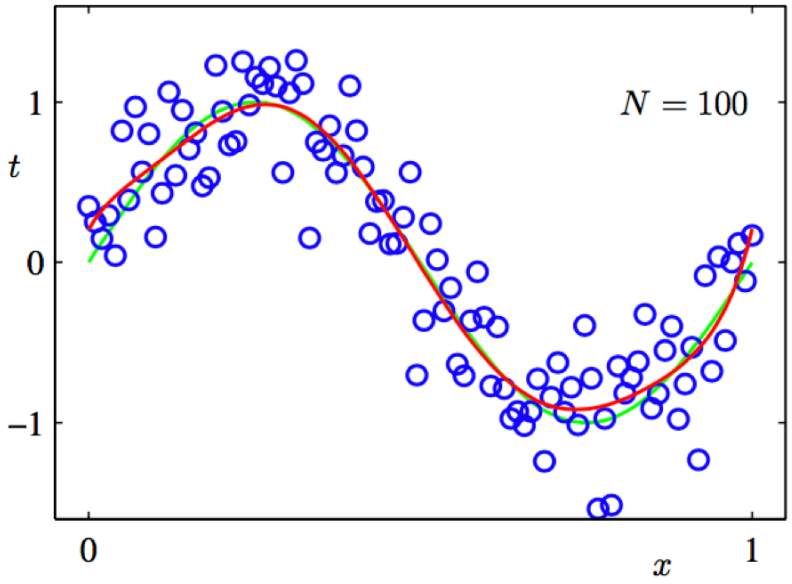
	$M=0$	$M=1$	$M=6$	$M=9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

- The more flexible polynomials with larger values of M are becoming increasingly tuned to the random noise on the target values.

Over-fitting

- For a given model complexity, the over-fitting problem become less severe as the size of the data set increases.
- Another way to say this is that the larger the data set, the more complex (more flexible) the model that we can afford to fit to the data.





Regularization

- One technique that is often used to control the over-fitting phenomenon in such cases is that of regularization, which involves adding a penalty term to the error function in order to discourage the coefficients from reaching large values.

- The simplest such penalty term takes the form of a sum of squares of all of the coefficients:

$$\widetilde{E}(\mathbf{w}) = \frac{1}{2}\sum_{n=1}^{N}\{y(x_n,\mathbf{w}) - t_n\}^2 + \frac{\lambda}{2}\|\mathbf{w}\|^2 \qquad (1.4)$$

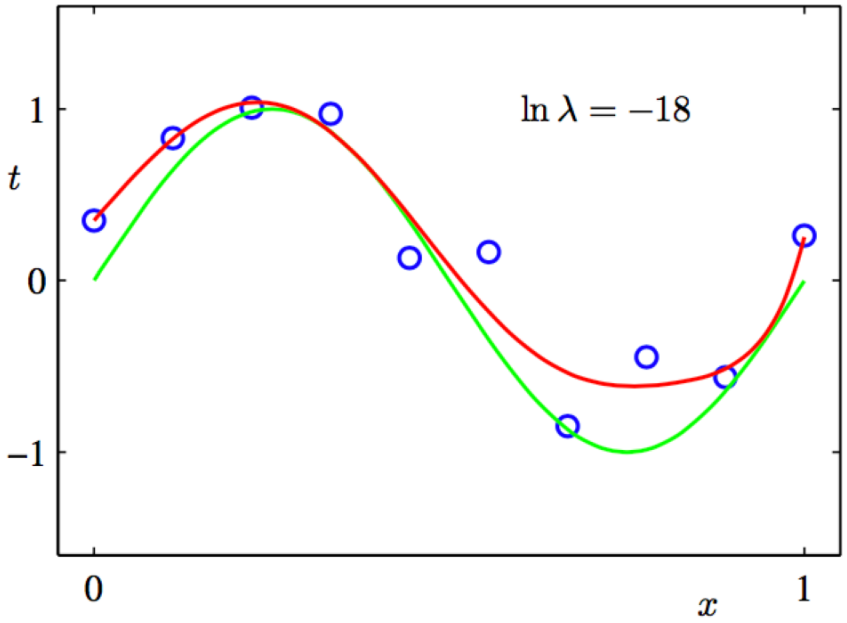
where $\|\mathbf{w}\|^2 \equiv \mathbf{w}^{\mathrm{T}}\mathbf{w} = w_0^2 + w_1^2 + \ldots + w_M^2$, and the coefficient λ governs the relative importance of the regularization term compared with the sum-of-squares error term.

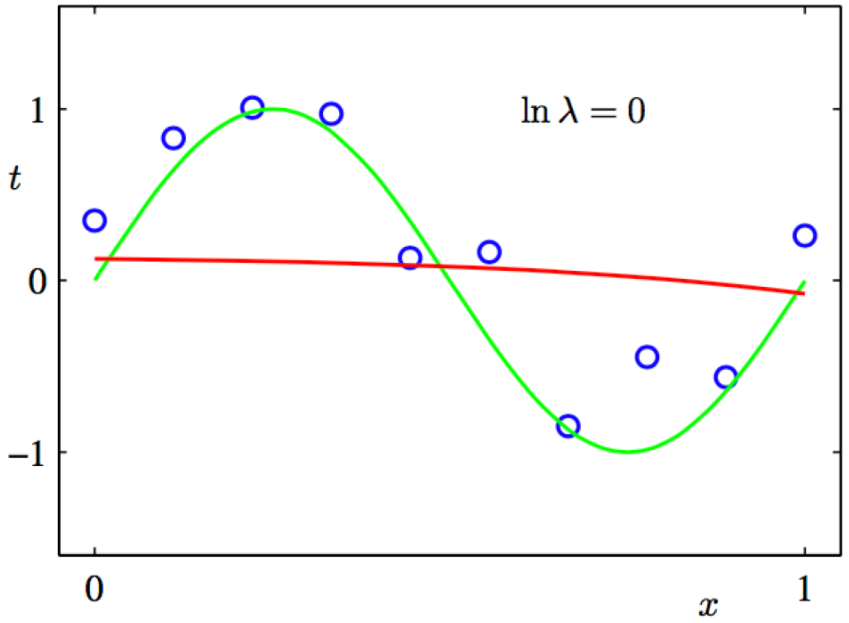
Regularization

- Note that often the coefficient ω_0 is omitted from the regularizer because its inclusion causes the results to depend on the choice of origin for the target variable, or it may be included but with its own regularization coefficient.

- Techniques such as this are known in the statistics literature as shrinkage methods because they reduce the value of the coefficients. The particular case of a quadratic regularizer is called ridge regression. In the context of neural networks, this approach is known as weight decay.

Regularization

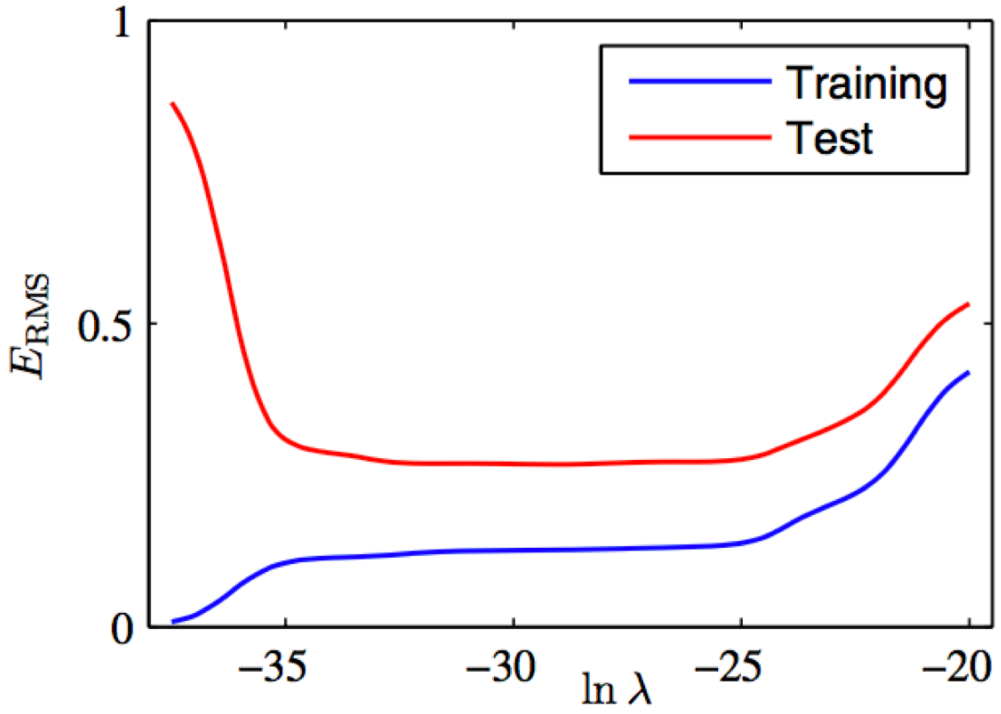




	$\ln \lambda = -\infty$	$\ln \lambda = -18$	$\ln \lambda = 0$
$w_0^\star$	0.35	0.35	0.13
$w_1^\star$	232.37	4.74	-0.05
$w_2^\star$	-5321.83	-0.77	-0.06
$w_3^\star$	48568.31	-31.97	-0.05
$w_4^\star$	-231639.30	-3.89	-0.03
$w_5^\star$	640042.26	55.28	-0.02
$w_6^\star$	-1061800.52	41.32	-0.01
$w_7^\star$	1042400.18	-45.95	-0.00
$w_8^\star$	-557682.99	-91.53	0.00
$w_9^\star$	125201.43	72.68	0.01

Regularization

- **We see that in effect λ now controls the effective complexity of the model and hence determines the degree of over-fitting.**



Validation

- If we were trying to solve a practical application using this approach of minimizing an error function, we would have to find a way to determine a suitable value for the model complexity.

- The results above suggest a simple way of achieving this, namely by taking the available data and partitioning it into a training set, used to determine the coefficients $\mathbf{w}$, and a separate **validation set**, also called a **hold-out** set, used to optimize the model complexity (either M or λ).

- In many cases, however, this will prove to be too wasteful of valuable training data, and we have to seek more sophisticated approaches.