Applied Machine Learning

Linear Models (1)

Computer Science, Fall 2022

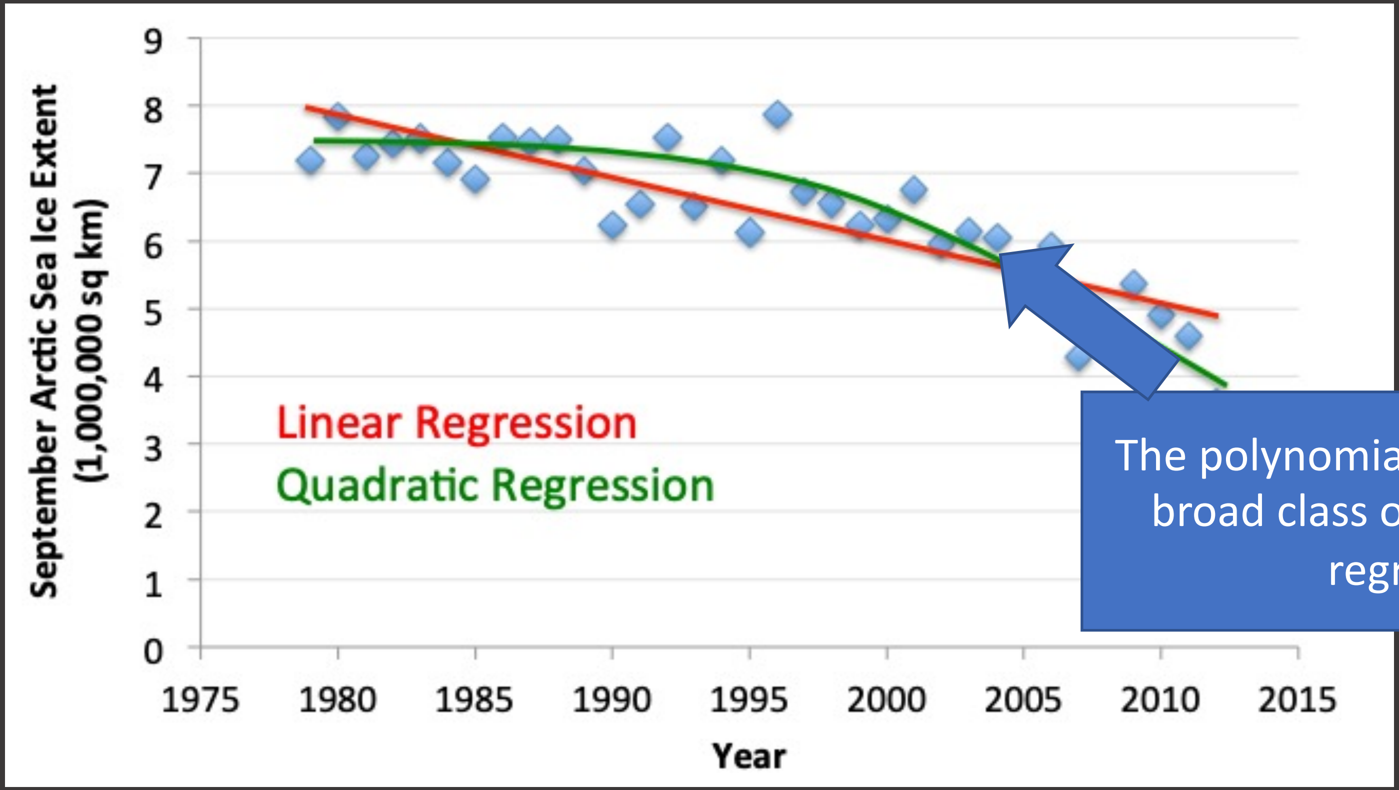
Instructor: Xuhong Zhang

Some Basic Philosophy: Simplicity First

- Simple algorithms often work very well !
- There are many kinds of simple structures, e.g.:
  - One attribute does all the work
  - All attributes contribute equally & independently
  - Logical structure with a few attributes suitable for tree
  - A set of simple logical rules
  - Relationships between groups of attributes
  - A weighted linear combination of the attributes
  - Strong neighborhood relationships based on distance
  - Clusters of data in unlabeled data
  - Bags of instances that can be aggregated
- Success of method depends on the domain

Linear Models: Regression

- Given data $X = \{x_1, \dots, x_n\}$, where $x_i \in \mathbb{R}^d$
- Corresponding labels $y = \{y_1, \dots, y_n\}$, where $y_i \in \mathbb{R}$



The polynomial is a special example of a broad class of functions called linear regression models

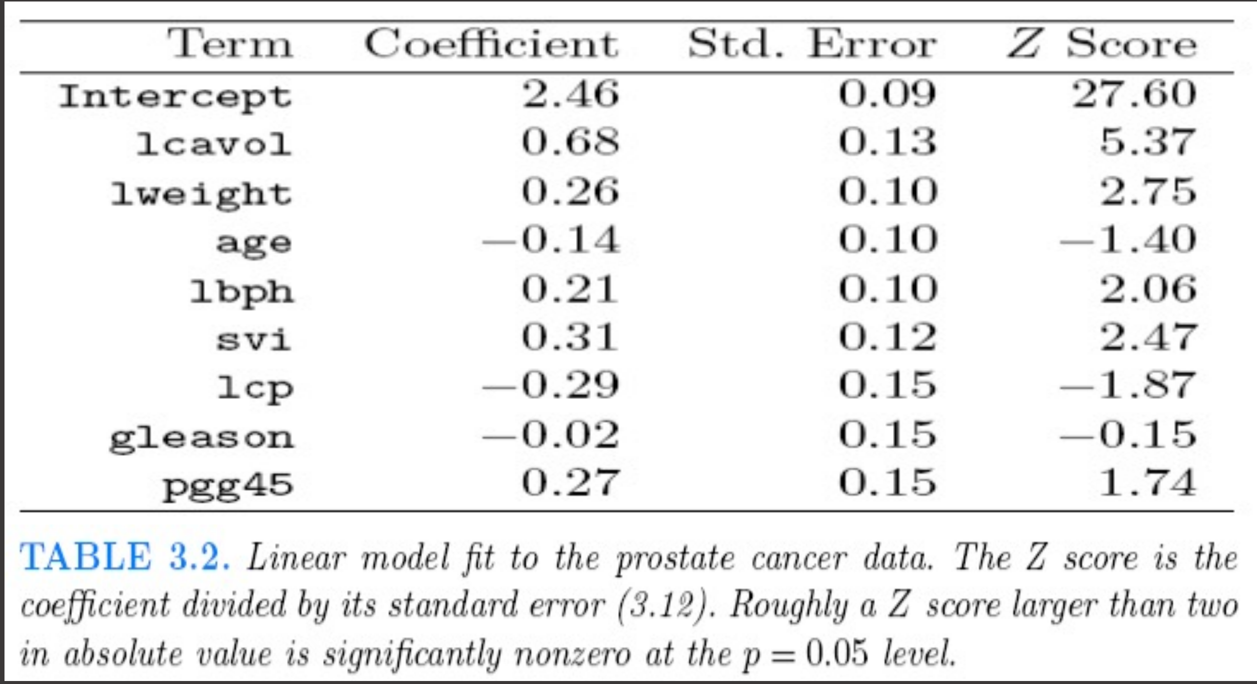
Data from G. Witt. Journal of Statistics Education, Volume 21, Number 1 (2013)

Goal

- The goal of regression is to predict the value of one or more continuous target variables t given the value of a D-dimensional vector x of input variables.
- The simplest form of linear regression models are also linear functions of the input variables.
- However, we can obtain a much more useful class of functions by taking linear combinations of a fixed set of nonlinear functions of the input variables, known as *basis functions.*

Prostate Cancer Dataset

- 97 samples, partitioned into 67 train / 30 test
- 8 predictors / features/ covariates
  - 6 continuous (4 log transforms), 1 binary, 1 ordinal
- Continuous outcome variable
  - lpsa: log(prostate specific antigen level)



Term	Coefficient	Std. Error	Z Score
Intercept	2.46	0.09	27.60
lcavol	0.68	0.13	5.37
lweight	0.26	0.10	2.75
age	−0.14	0.10	−1.40
lbph	0.21	0.10	2.06
svi	0.31	0.12	2.47
lcp	−0.29	0.15	−1.87
gleason	−0.02	0.15	−0.15
pgg45	0.27	0.15	1.74

TABLE 3.2. *Linear model fit to the prostate cancer data. The Z score is the coefficient divided by its standard error (3.12). Roughly a Z score larger than two in absolute value is significantly nonzero at the $p = 0.05$ level.*

Slide credit: Jeff Howbert

Linear Basis Function Models

- The simplest linear model for regression is one that involves a linear combination of the input variables:

$$y(x, w) = w_0 + w_1 x_1 + \cdots + w_D x_D$$

where $x = (x_1, \ldots, x_D)^T$

Imposes significant limitations

- The **KEY** property: it is a linear function of the parameters $w_0, \ldots, w_D$.

- It is also a linear function of the input variables x_i.

Linear Basis Function Models

- Extend the model: linear combinations of fixed nonlinear functions of the input variables

$\phi_j(x)$ is the basis functions

$$y(x, w) = w_0 + \sum_{j=1}^{M-1} w_j \phi_j(x)$$

bias

Linear Basis Function Models

- It is often convenient to define an additional dummy 'basis function', $\phi_0(x) = 1$ so that

$$y(x, w) = \sum_{j=0}^{M-1} w_j \phi_j(x) = w^T \phi(x)$$

where w = $(w_0, \dots, w_{M-1})^T$ and ϕ = $(\phi_0, \dots, \phi_{M-1})^T$

- In many practical applications of pattern recognition, we will apply some form of fixed pre-processing, or feature extraction, to the original data variables.

Linear Basis Function Models

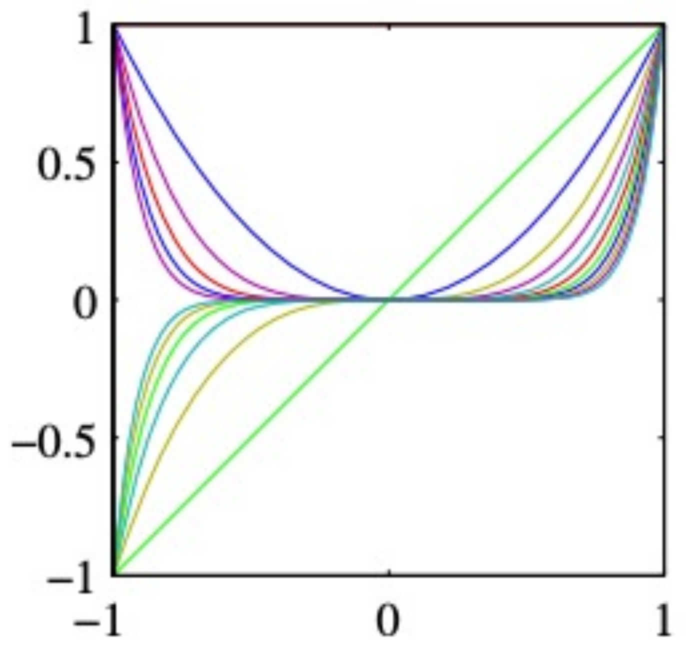
- By using nonlinear basis functions, we allow the function $y(x, w)$ to be non-linear function of the input vector x.

- The reason it is still called a linear model is because this function is linear in w.

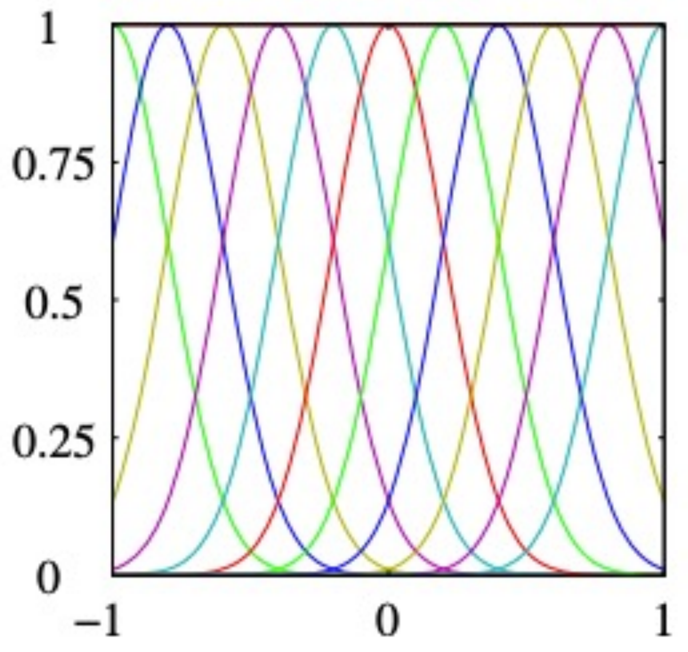
- Examples of the basis functions:
  - $\phi_j(x) = x^j$
  - $\phi_j(x) = \exp\{-\frac{(x-\mu_j)^2}{2s^2}\}$
  - $\phi_j(x) = \sigma(\frac{x-\mu_j}{s})$, where $\sigma(a) = \frac{1}{1+\exp(-a)}$

Logistic sigmoid function

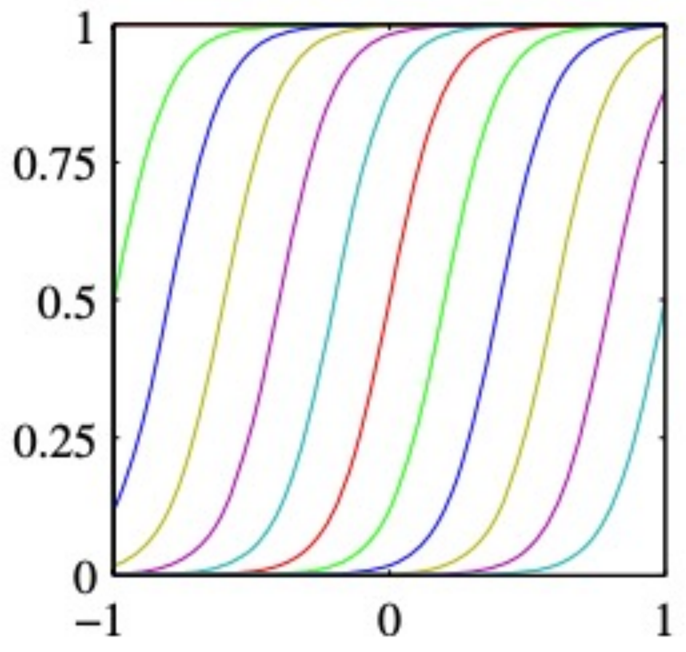
Linear Basis Function Models



Polynomial basis



Gaussian basis



Sigmoidal basis

Maximum Likelihood and Least Squares

- Assumption: the target variable t is given by a deterministic function $y(x, w)$ with additive Gaussian noise

$$t = y(x, w) + \epsilon$$

where ϵ is a zero mean Gaussian random variable with precision (inverse variance) β

$$p(t|x, w, \beta) = \mathcal{N}(t|y(x, w), \beta^{-1})$$

Bayes' Theorem

- In probability theory and statistics, Bayes' theorem (alternatively Bayes' law or Bayes' rule), named after Tomas Bayes, describes the probability of an event, based on the prior knowledge of conditions that might be related to the event.

Portrait purportedly of Bayes used in a 1936 book

Statement of Theorem

- Bayes’ theorem is stated mathematically as the following:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

  - $P(A|B)$ is a conditional probability; it is also called posterior probability.
  - $P(B|A)$ is also a conditional probability; it is also called the likelihood.
  - $P(A)$ is the probability of observing A; it is also called prior probability.
  - $P(B)$ is the probability of observing B; it is known as the marginal probability.

Proof for events

- Bayes' theorem may be derived from the definition of conditional probability:

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, \text{ if } P(B) \neq 0$$

- Similarly,

$$P(B|A) = \frac{P(B \cap A)}{P(A)}, \text{ if } P(A) \neq 0$$

- Solving for $P(A \cap B)$ and substituting into the above expression for $P(A|B)$ yields Bayes' theorem:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, \text{ if } P(B) \neq 0$$

Maximum Likelihood and Least Squares

- Suppose the parameter for the model assumption is w and before we see the data, the prior probability distribution about w is $p(w)$.
- The effect of the observed data $\mathcal{D} = \{d_1, \dots, d_n\}$ is expressed through the conditional probability $p(\mathcal{D}|w)$.
- Based on Bayes' theorem, we have

$$p(w|\mathcal{D}) = \frac{p(\mathcal{D}|w)p(w)}{p(\mathcal{D})}$$

which allows us to evaluate the uncertainty in w <u>after</u> we have observed dataset $\mathcal{D}$.

Likelihood function

Maximum Likelihood and Least Squares

- The likelihood function expresses how probable the observed data set is for different settings of the parameter vector w.
- Given this definition of likelihood, we can state Bayes' theorem in words as

$$Posterior \propto likelihood \times prior$$

- The denominator $p(\mathcal{D})$ can be expressed in terms of the prior and the likelihood

$$p(\mathcal{D}) = \int p(\mathcal{D}|w)p(w)dw$$

Normalization constant

Maximum Likelihood and Least Squares

- *Maximum Likelihood: w is set to the value that maximizes the likelihood function $p(\mathcal{D}|w)$.*
    - *This corresponds to choosing the value of w for which the probability of the observed data set is maximized.*
    - *In the machine learning literature, the negative log of the likelihood function is called an error function.*
    - *Because the negative logarithm is a monotonically decreasing function, maximizing the likelihood is equivalent to minimizing the error.*

Advantage of Bayes' view

- Suppose, for instance, that a fair-looking coin is tossed three times and lands heads each time.
  - A classical maximum likelihood estimate of the probability of landing heads would be 1, implying that all future tosses will land heads!
  - By contrast, a Bayesian approach with any reasonable prior will lead to a much less extreme conclusion.

Maximum Likelihood and Least Squares

- Now consider a dataset inputs $X = \{x_1, \dots, x_N\}$ with corresponding target values $t_1, \dots, t_N$, and we use **t** to denote the column vector of $\boldsymbol{t} = \{t_n\}$.
- Assumption: the data points $\boldsymbol{t}$ are drawn independently from the distribution. We then get the likelihood function, with parameters w and β :

$$p(t|X, w, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n | w^T \phi(x_n), \beta^{-1})$$

  - Note that in supervised learning problems such as regression (and classification), we are not seeking to model the distribution of the input variables. Thus, x will always appear in the set of conditioning variables.

Maximum Likelihood and Least Squares

- Taking the logarithm of the likelihood function, and making use of the univariate Gaussian, we have:

$$\ln p(t|w,\beta) = \sum_{n=1}^{N} \ln \mathcal{N}(t_n|w^T\phi(x_m),\beta^{-1})$$

$$= \frac{N}{2}\ln\beta - \frac{N}{2}\ln 2\pi - \beta E_D(w)$$

Where the sum-of-squares error function is defined by

$$E_D(w) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - w^T\phi(x_n)\}^2$$

Maximum Likelihood and Least Squares

- With the likelihood function, we can use maximum likelihood to determine w and β.
  - First consider the maximization with respect to w.

$$\nabla \ln p(\boldsymbol{t}|w,\beta) = \sum_{n=1}^{N} \{t_n - w^T\phi(x_n)\}\phi(x_n)^T$$

  - Setting the gradient to zero gives

$$0 = \sum_{n=1}^{N} t_n\phi(x_n)^T - w^T(\sum_{n=1}^{N} \phi(x_n)\phi(x_n)^T)$$

Maximum Likelihood and Least Squares

Moore-Penrose pseudo-inverse of the matrix Φ, we label it as $\phi^{\dagger}$

- Solving for w we obtain

$$w_{ML} = \boxed{(\Phi^T\Phi)^{-1}\Phi^T}\boldsymbol{t}$$

Which are known as the normal equations for the **least squares** problem.

- Here Φ is an $N\times M$ matrix, called the *design matrix*, whose elements are given by $\Phi_{nj} = \phi_j(x_n)$, so that

$$\boldsymbol{\Phi} = \begin{pmatrix} \phi_0(\boldsymbol{x_1}) & \cdots & \phi_{M-1}(\boldsymbol{x_1}) \\ \vdots & \ddots & \vdots \\ \phi_0(\boldsymbol{x_N}) & \cdots & \phi_{M-1}(\boldsymbol{x_N}) \end{pmatrix}$$

- If Φ is square and invertible, then using the property $(AB)^{-1} = B^{-1}A^{-1}$ we then have $\phi^{\dagger} = \phi^{-1}$.

Maximum Likelihood and Least Squares

- At this point, we can gain some insight into the role of the bias parameter w_0. If we make the bias parameter explicit, then the error function becomes

$$E_D(w) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - w_0 - \sum_{j=1}^{M-1} w_j\,\phi(x_n)\}^2$$

Setting the derivative with respect to w_0 equal to zero, and solving for w_0, we obtain

$$w_0 = \bar{t} - \sum_{j-1}^{M-1} w_j\overline{\phi_j}$$

Where we have defined $\bar{t} = \frac{1}{N}\sum_{n=1}^{N} t_n$, and $\overline{\phi_j} = \frac{1}{N}\sum_{n=1}^{N}\phi_j(x_n)$

The difference between the averages (over the training set) of the target values

The weighted sum of the averages of the basis function values

Maximum Likelihood and Least Squares

- We can also maximize the log likelihood function with respect to the noise precision parameter β:

$$\frac{1}{\beta_{ML}} = \frac{1}{N}\sum_{n=1}^{N}\{t_n - w_{ML}^T \phi(x_n)\}^2$$

Multiple Outputs

- In some applications, we may wish to predict $K > 1$ target variables, which we denote collectively by the target vector $\boldsymbol{t}$.
- This could be done by introducing a different set of basis functions for each component of $\boldsymbol{t}$, leading to multiple, independent regression problems.
- More common, approach is to use the same set of basis functions to model all of the components of the target vector so that

$$y(x, w) = \boldsymbol{W}^T \phi(x)$$

Multiple Outputs

$$y(x, w) = \boldsymbol{W}^T \phi(x)$$

Where y is a K-dimensional column vector, W is an $M \times K$ matrix of parameters, and $\phi(x)$ is an M-dimensional column vector with elements $\phi_j(x)$, with $\phi_0(x) = 1$ as before.

- Suppose we take the conditional distribution of the target vector to be an isotropic Gaussian of the form
$$p(t|x, W, \beta) = \mathcal{N}(t|\boldsymbol{W}^T \phi(x), \beta^{-1}\mathbf{I})$$

Multiple Outputs

- Hint: if we have a set of observations $t_1, \ldots, t_N$, we can combine these into a matrix T of size $N \times K$ such that the n^{th} row is given by t_n^T.
- Similarly, we can combine the input vectors $x_1, \ldots, x_N$ into a matrix $\mathbf{X}$.
- The log likelihood function is then given by

$$\ln p(T|X, W, \beta) = \sum_{n=1}^{N} \ln \mathcal{N}(t_n | \boldsymbol{W}^T \phi(x_n), \beta^{-1}\mathbf{I})$$

$$= \frac{NK}{2} \ln\left(\frac{\beta}{2\pi}\right) - \frac{\beta}{2} \sum_{n=1}^{N} ||t_n - W^T \phi(x_n)||^2$$

Multiple Outputs

- As before, we can maximize this function with respect to W, giving

$$\mathbf{W}_{ML} = (\Phi^T \Phi)^{-1} \Phi^T \mathbf{T}$$

If we examine this result for each target variable t_k, we have

$$\boldsymbol{w}_k = (\Phi^T \Phi)^{-1} \Phi^T t_k = \Phi^{\dagger} t_k$$

where t_k is an N-dimensional column vector with components t_{nk} for $n = 1, \dots, N$. Thus the solution to the regression decouples between the different target variables, and we need only compute a single pseudo-inverse matrix $\Phi^{\dagger}$, which is shared by all the vectors $\boldsymbol{w}_k$.

Sequential Learning

- In real world, processing the entire training set in one go, can be computationally costly and even prohibitive.
- If the dataset is sufficiently large, it may be worthwhile to use *sequential* algorithms, also known as *on-line* algorithms, in which the data points are considered one at a time, and the model parameters updated after each such presentation.
- Appropriate for real-time applications in which the data observations are arriving in a continuous stream and predictions must be made before all of the data points are seen.

Sequential Learning

- Stochastic gradient descent (also known as sequential gradient descent):
  - If the error function comprises a sum over data points $E = \sum_n E_n$, then after presentation of pattern n, the stochastic gradient descent algorithm updates the parameter vector w using

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E_n$$

Where τ denotes the iteration number, and η is the learning rate parameter.

Sequential Learning

- The value of w is initialized to some starting vector w.
- For the case of the sum-of-squares error function

$$E_D(w) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - w^T\phi(x_n)\}^2$$

This gives

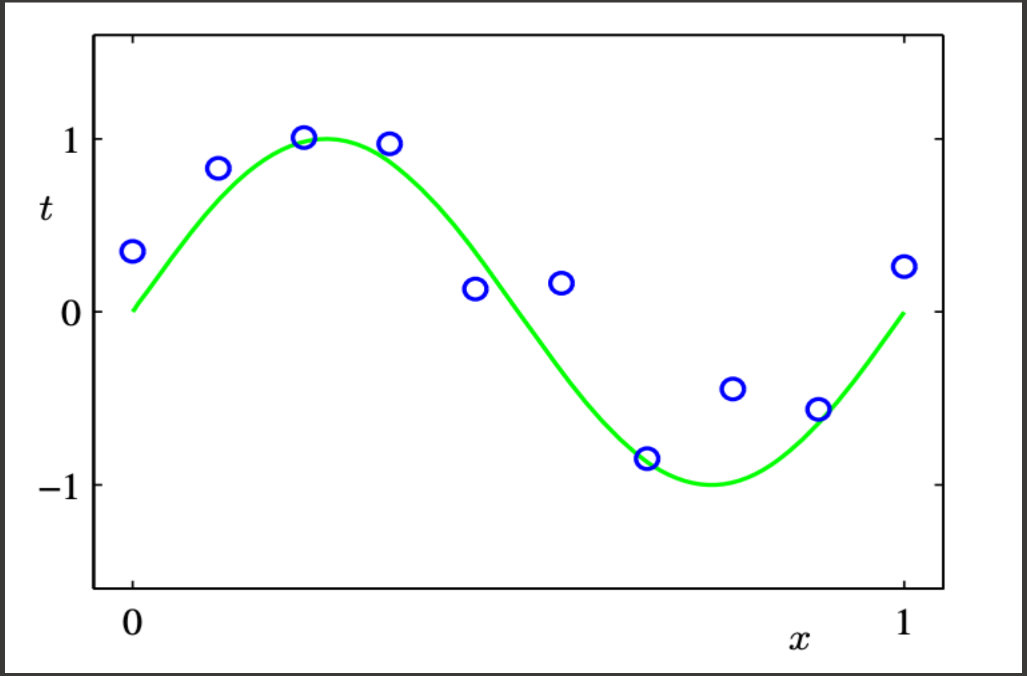
$$w^{(\tau+1)} = w^{(\tau)} + \eta(t_n - w^{(\tau)T}\phi_n)\phi_n$$

where $\phi_n = \phi(x_n)$

- The value η needs to be chosen with care to ensure that the algorithm converges.

Overfitting: Polynomial Curve Fitting

- Recap setting: we are given a training set comprising N observations of x, written as $\mathrm{x} = \{x_1, \dots, x_N\}$, together with observation values of t, denoted as $t = \{t_1, \dots, t_N\}$.
- Our input data looks like this:



$N = 10$ points, shown as blue circles, each comprising an observation of the input value x along with the corresponding target variable t.

The green curve shows the function $\sin(2\pi x)$ used to generate the data. The goal is to predict the value of t for some new value x, without knowledge of the green curve.

Overfitting: Polynomial Curve Fitting

- We first try to fit the data with a polynomial curve

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

- $y(x, w)$ is a nonlinear function of x, but linear in the unknown parameters, and are still called linear models.

- Again, we can try to minimize the error function

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

Overfitting: Polynomial Curve Fitting

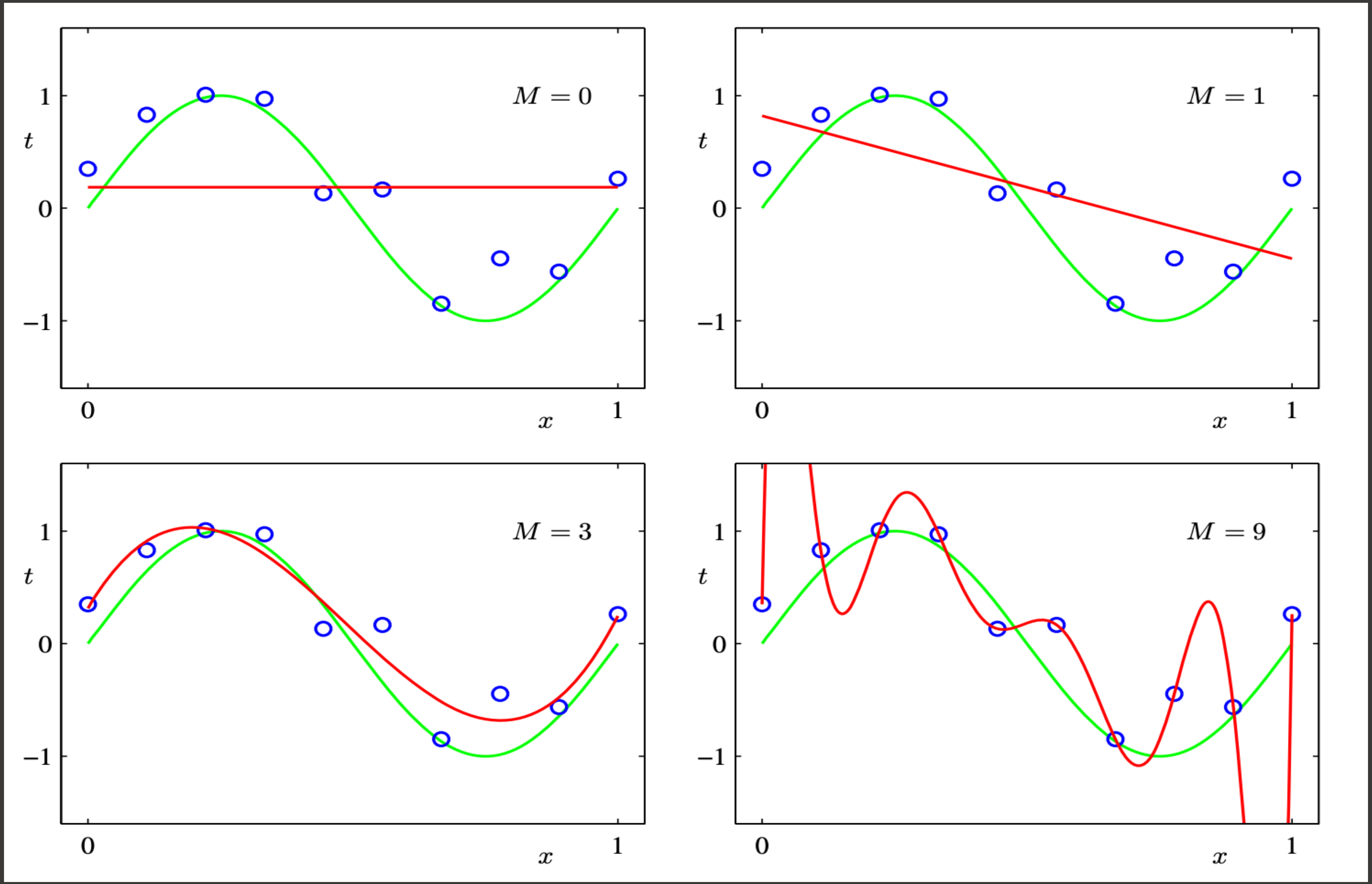
- The error function is a quadratic function of the coefficients w, its derivatives with respect to the coefficients will be linear in the elements of w, and so the minimization of the error function has a unique solution, denoted by w^*, which can be found in closed form. The resulting polynomial is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

- Now the remaining problem is <u>choosing the order M.</u>

Model Selection or Model Comparison

Overfitting: Polynomial Curve Fitting

Q: What's the Problem with Each curve?



Overfitting: Polynomial Curve Fitting

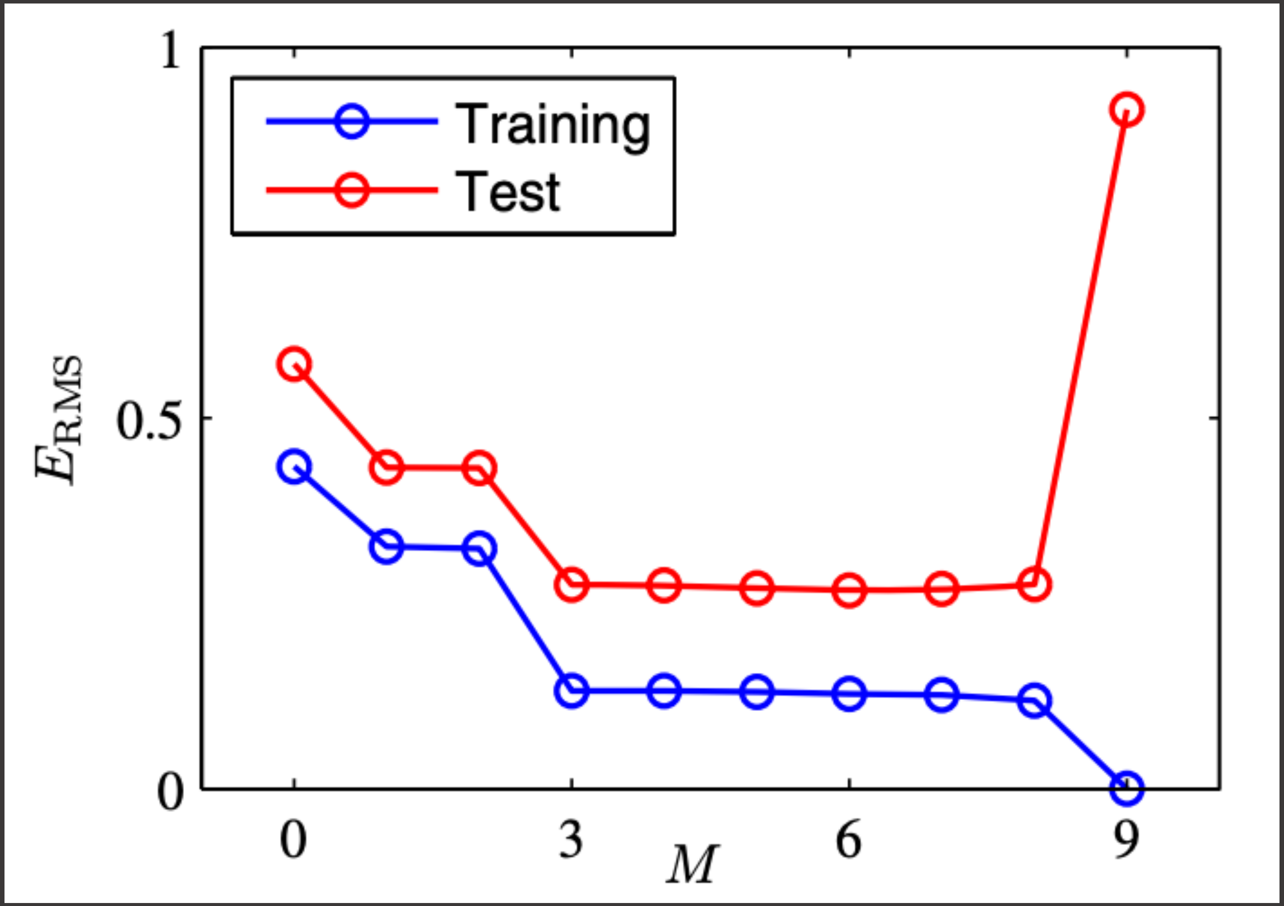
- When we go to a much higher order polynomial ($M = 9$), we can obtain an excellent fit to the training data with $E(w^*) = 0$.

- However, the fitted curve oscillates wildly and gives a very poor representation of the $\sin$ function. This behavior is called *over-fitting*.

Overfitting: Polynomial Curve Fitting

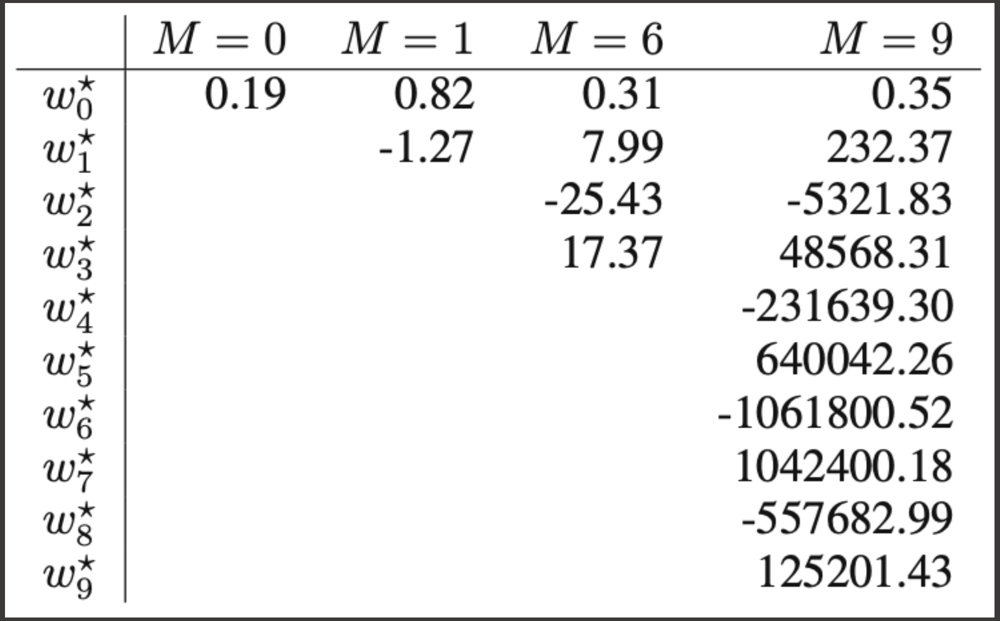
- **Remember that, our goal is <u>always</u> making accurate predictions for new data.**
  - We can evaluate $\mathrm{E}(\mathrm{w}^*)$ for the training data
  - We can also evaluate $\mathrm{E}(\mathrm{w}^*)$ for the test data
  - Sometimes it is more convenient to use the root-mean-square (RMS): $E_{RMS} = \sqrt{2E(w^*)/N}$

Overfitting: Polynomial Curve Fitting



Overfitting: Polynomial Curve Fitting

- We can also gain some insight into the problem by examining the values of the coefficients w^* obtained from various orders.

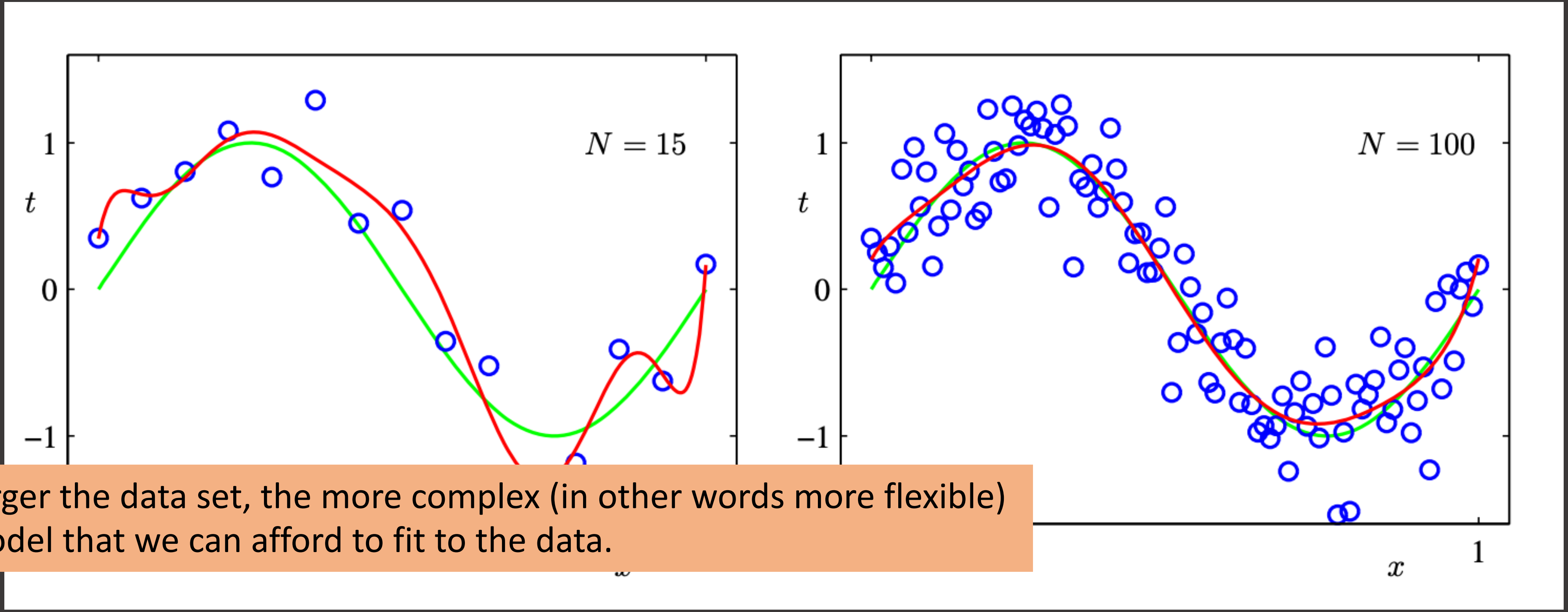


	$M = 0$	$M = 1$	$M = 6$	$M = 9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

Observe how the typical magnitude of the coefficients increases dramatically as the order of the polynomial increases.

Overfitting: Polynomial Curve Fitting

- It is also interesting to examine the behaviour of a given model as the size of the data set is varied. Both has M=9.



The larger the data set, the more complex (in other words more flexible) the model that we can afford to fit to the data.

Overfitting: Polynomial Curve Fitting

- One rough heuristic that is sometimes advocated is that the number of data points should be no less than some multiple (say 5 or 10) of the number of adaptive parameters in the model.

- There is something rather unsatisfying about having to limit the number of parameters in a model according to the size of the available training set. It would be more reasonable to choose the complexity of the model according to the complexity of the problem.

Regularization

- One technique that is often used to control the over-fitting phenomenon is that of regularization.
- It involves adding a penalty term to the error function in order to discourage the coefficients from reaching large values.
- A simple penalty term:

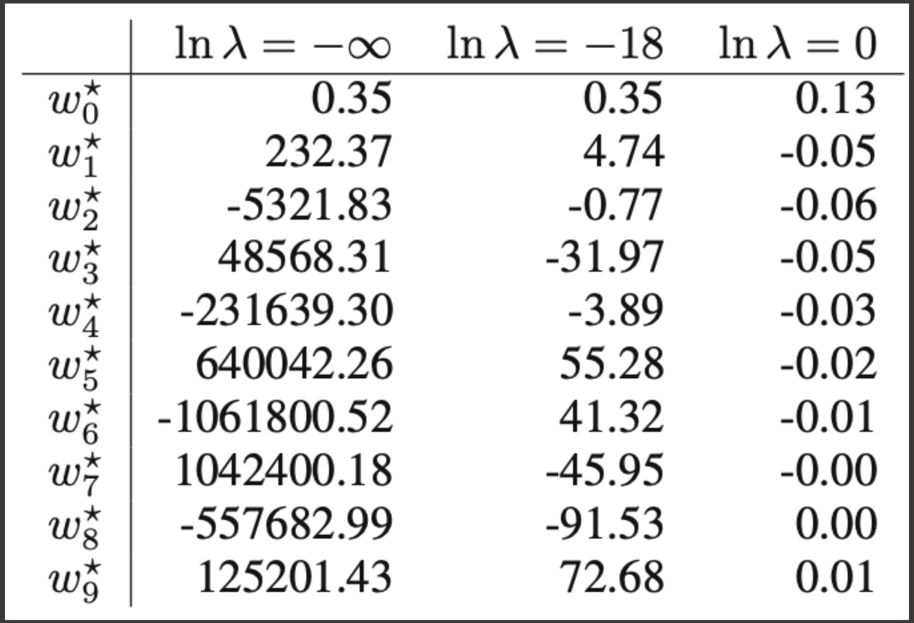
$$E_D(w) = \frac{1}{2}\sum_{n=1}^{N}\{t_n - y(x_n, w)\}^2 + \frac{\lambda}{2}||w||^2$$

where $||w||^2 = w^T w = w_0^2 + w_1^2 + \cdots + w_M^2$ and λ governs the ratio between this term and the sum error.

Regularization

- Notes:
  - w_0 is often omitted because its inclusion causes the results to depend on the choice of origin for the target variable.
  - This new error function can be minimized exactly in closed form.
  - In statistics, it is known 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

$\lambda = -\infty$ corresponds to a model with no regularization. As the value increases, the typical magnitude of the coefficients gets smaller.