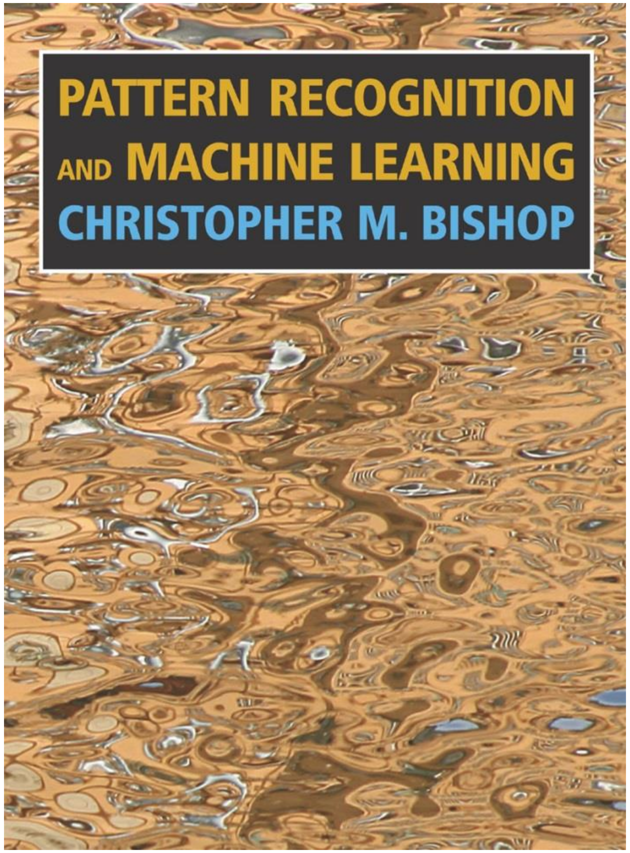
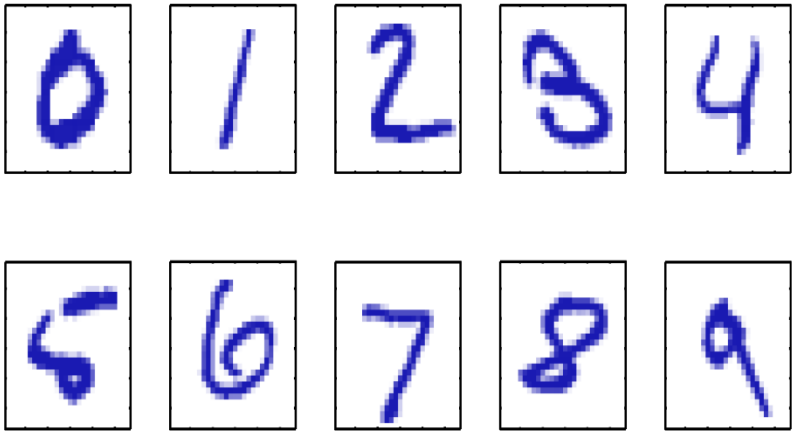
Machine-learning Crash Course

- This is not a course, people are expected to contribute (read, discuss, propose). Please bear in mind that we can be wrong!
- The initial idea is to follow Bishop's book, but there are other important references (e.g. Deep-learning book by Goodfellow).
- We thought of meeting twice a week 1 hour each session.
- Apart from discussing the concepts we (all) deem interesting, we could: do exercises, program, discuss relevant papers… Suggestions?



MNIST dataset



Automatically finding **regularities** in a dataset through the use of computer algorithms

Supervised learning: the training data comprises examples of the input vectors along with their corresponding target:

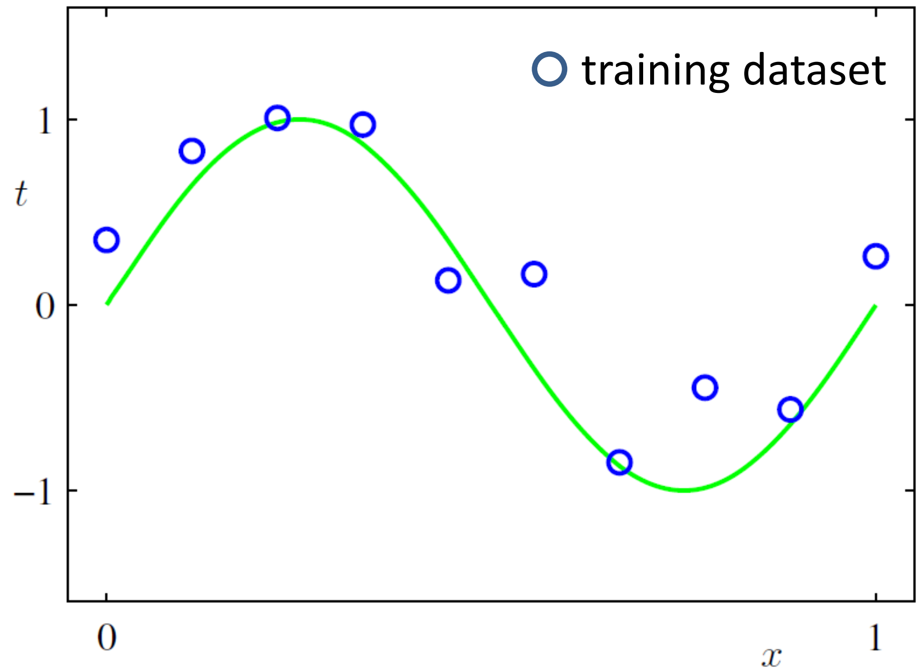
- Digit recognition (classification).
- Predicting price from size of a house (regression).

Unsupervised learning: there is not target (no ground truth), which makes the performance harder to evaluate:

- Clustering.
- Density estimation.
- Visualization.

Reinforcement learning: find the optimal policy (set of actions) to maximize the reward. Not cover in the book (and, probably, not cover by this course).

Example: polynomial curve fitting



parameters

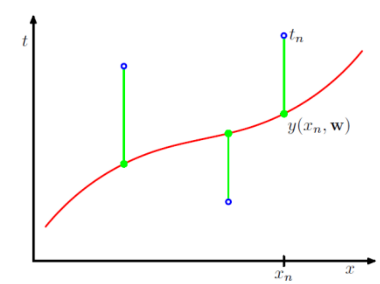
hyperparameter(s)

Our (linear) model:

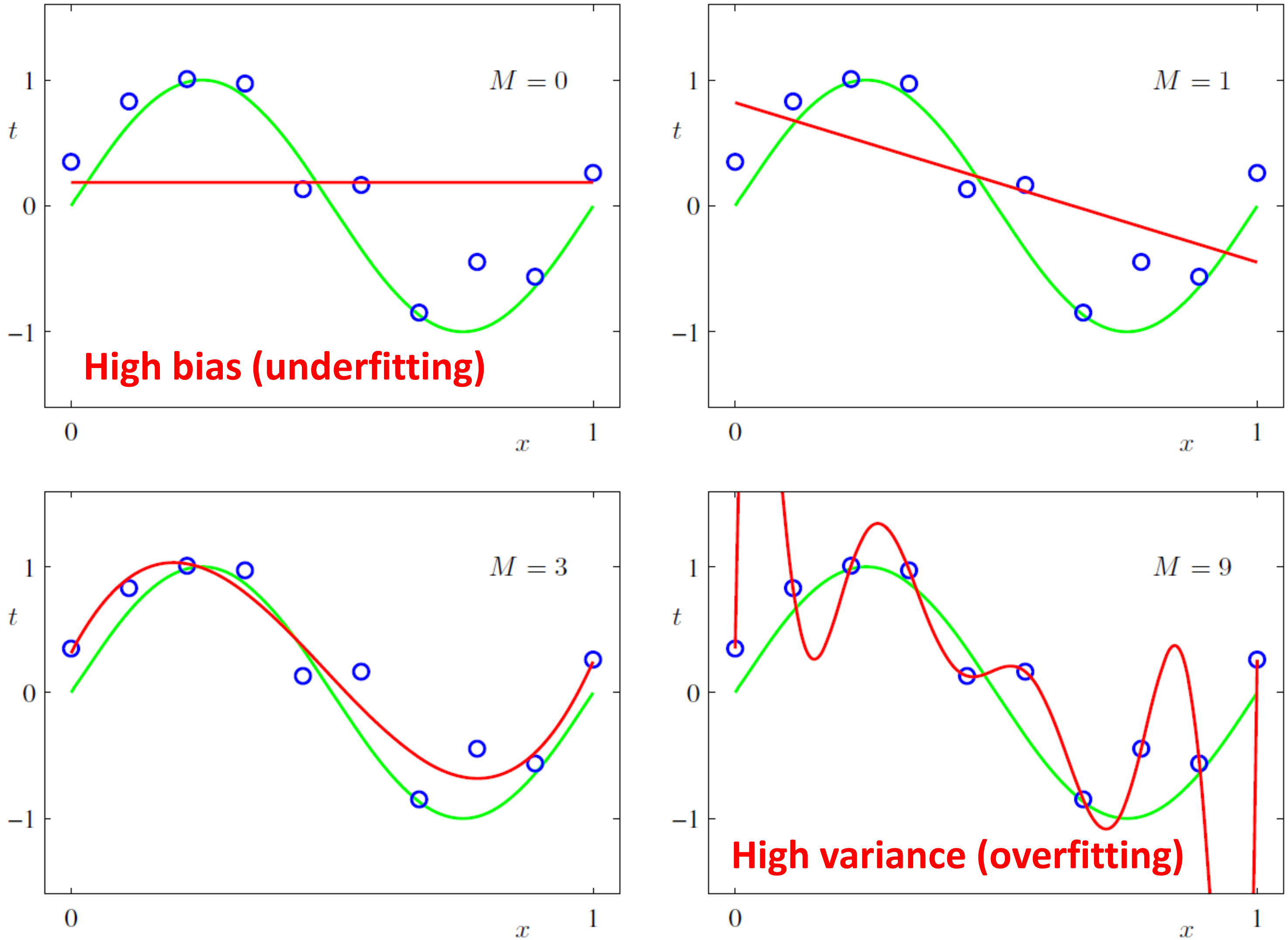
$$y(x, \mathbf{w}) = w_0 + w_1 x + w_2 x^2 + \ldots + w_M x^M = \sum_{j=0}^{M} w_j x^j$$

Our error function (sum of squares):

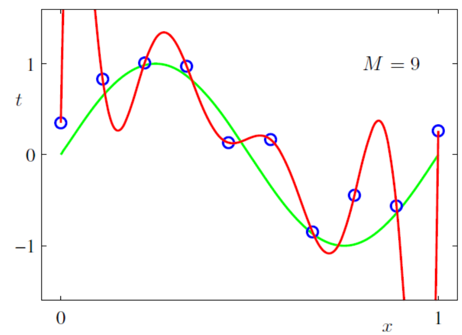
$$E(\mathbf{w}) = \frac{1}{2} \sum_{n=1}^{N} \{y(x_n, \mathbf{w}) - t_n\}^2$$

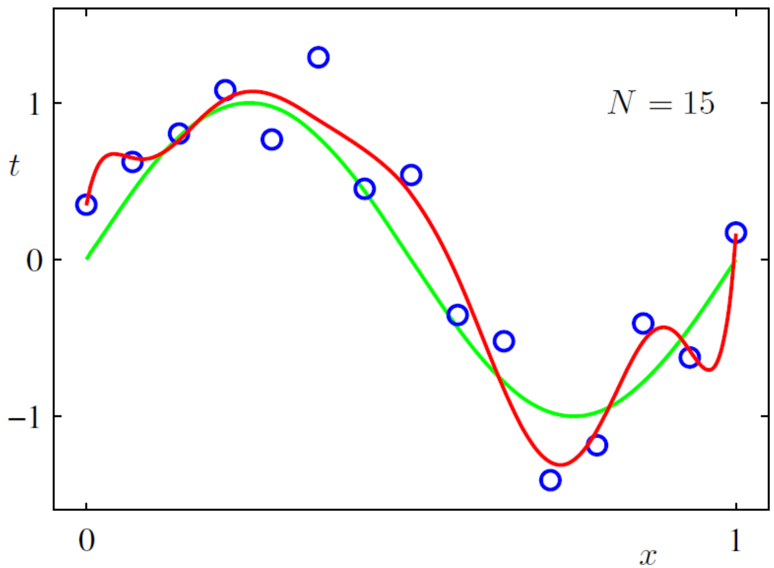


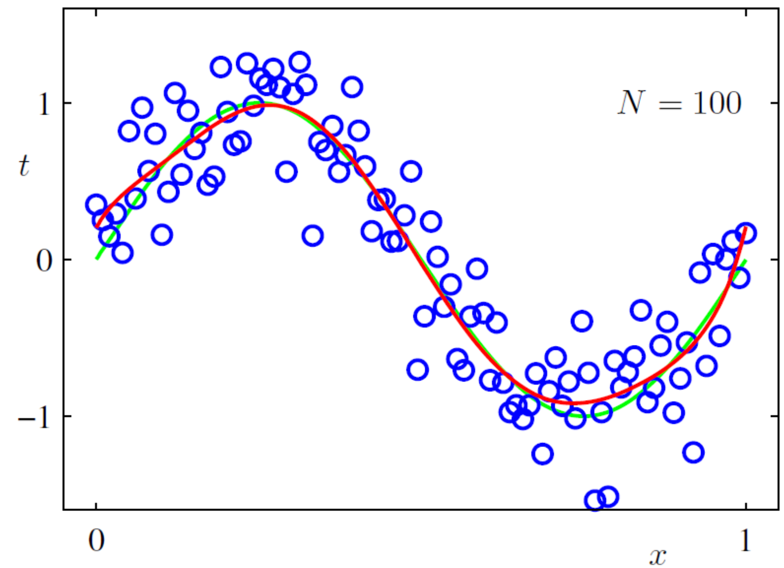
Model Selection (the bias-variance tradeoff)



The bias-variance tradeoff depends (a lot) on the amount of data.







The larger the training dataset, the more flexible the model can be without risking a high variance issue

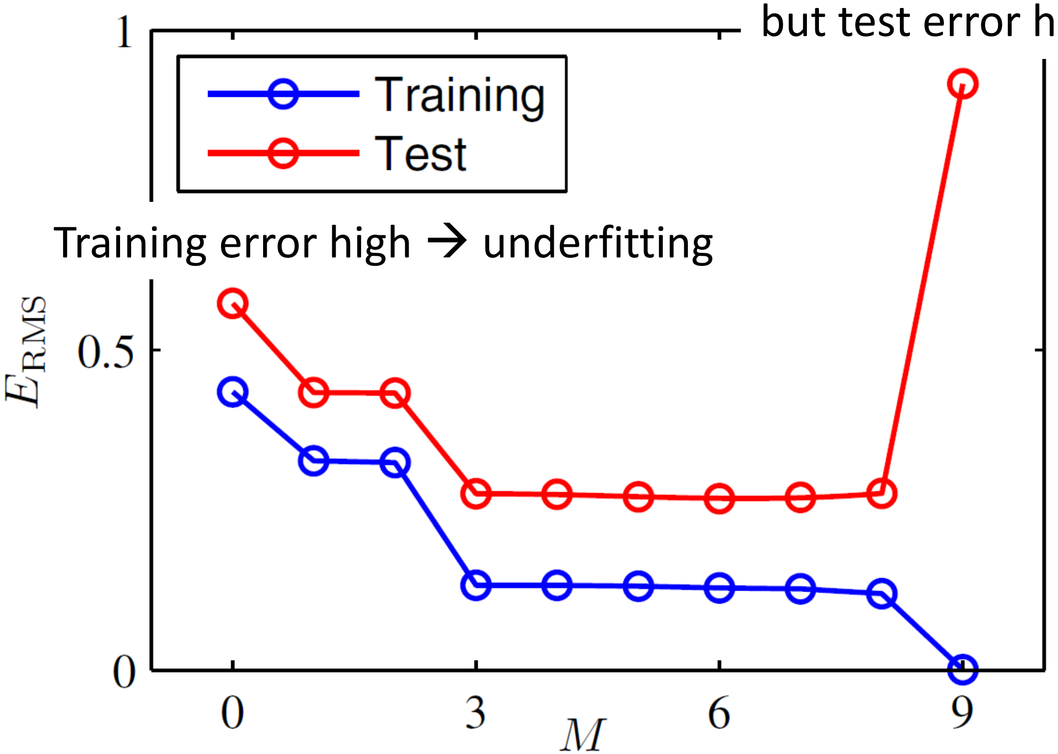
A way to control for overfitting:

1. Divide dataset into *training* and *test* sets.
2. Train the model on the training dataset.
3. Test the model on the test dataset.

(note that this limits the amount of data available to train the model)

Training error low
but test error high → overfitting

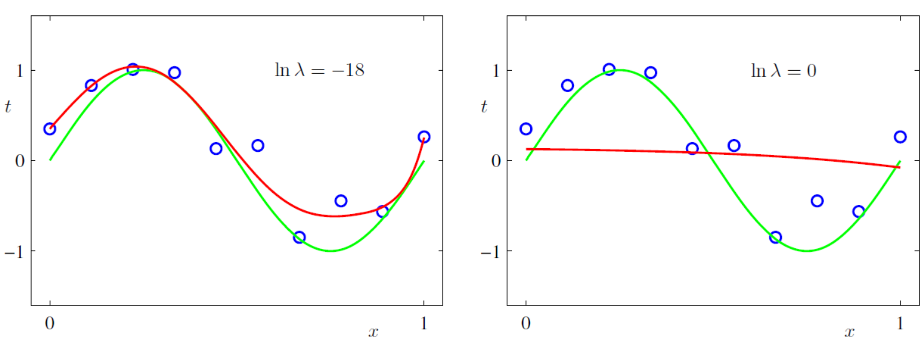
Training error high → underfitting

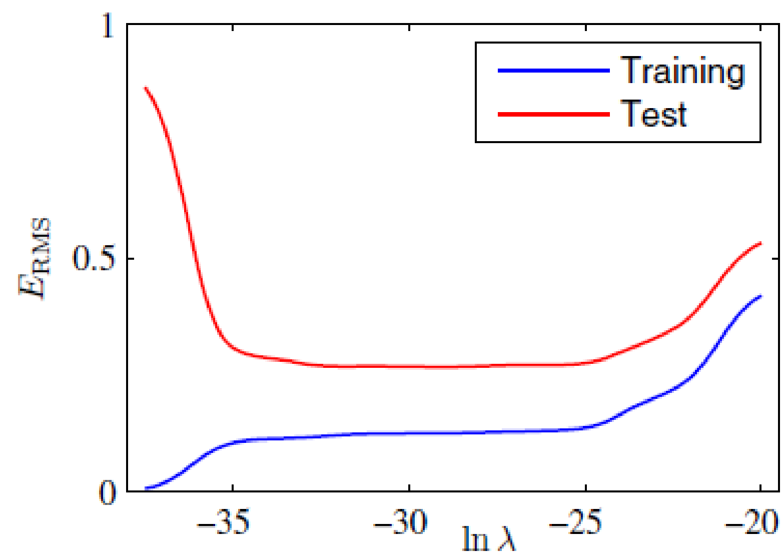


	$M = 0$	$M = 1$	$M = 6$	$M = 9$
$w_0^\star$	0.19	0.82	0.31	0.35
$w_1^\star$		-1.27	7.99	232.37
$w_2^\star$			-25.43	-5321.83
$w_3^\star$			17.37	48568.31
$w_4^\star$				-231639.30
$w_5^\star$				640042.26
$w_6^\star$				-1061800.52
$w_7^\star$				1042400.18
$w_8^\star$				-557682.99
$w_9^\star$				125201.43

A way to avoid overfitting: regularization.

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





Probability Theory

The Rules of Probability

sum rule $$p(X) = \sum_Y p(X, Y)$$

product rule $$p(X, Y) = p(Y|X)p(X).$$

Bayesian approach

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

we would like to address and quantify the **uncertainty** that surrounds the appropriate choice for the model parameters **w**

posterior

likelihood

prior

$$\underbrace{p(\mathbf{w}|\mathcal{D})}_{\text{posterior}} = \frac{\overbrace{p(\mathcal{D}|\mathbf{w})}^{\text{likelihood}}\,\overbrace{p(\mathbf{w})}^{\text{prior}}}{p(\mathcal{D})}$$

$$p(\mathcal{D}) = \int p(\mathcal{D}|\mathbf{w})p(\mathbf{w})\,\mathrm{d}\mathbf{w}$$

$$\text{posterior} \propto \text{likelihood} \times \text{prior}$$

$$\underbrace{p(\mathbf{w}|\mathcal{D})}_{\text{posterior}} = \frac{\overbrace{p(\mathcal{D}|\mathbf{w})}^{\text{likelihood}}\,\overbrace{p(\mathbf{w})}^{\text{prior}}}{p(\mathcal{D})}$$

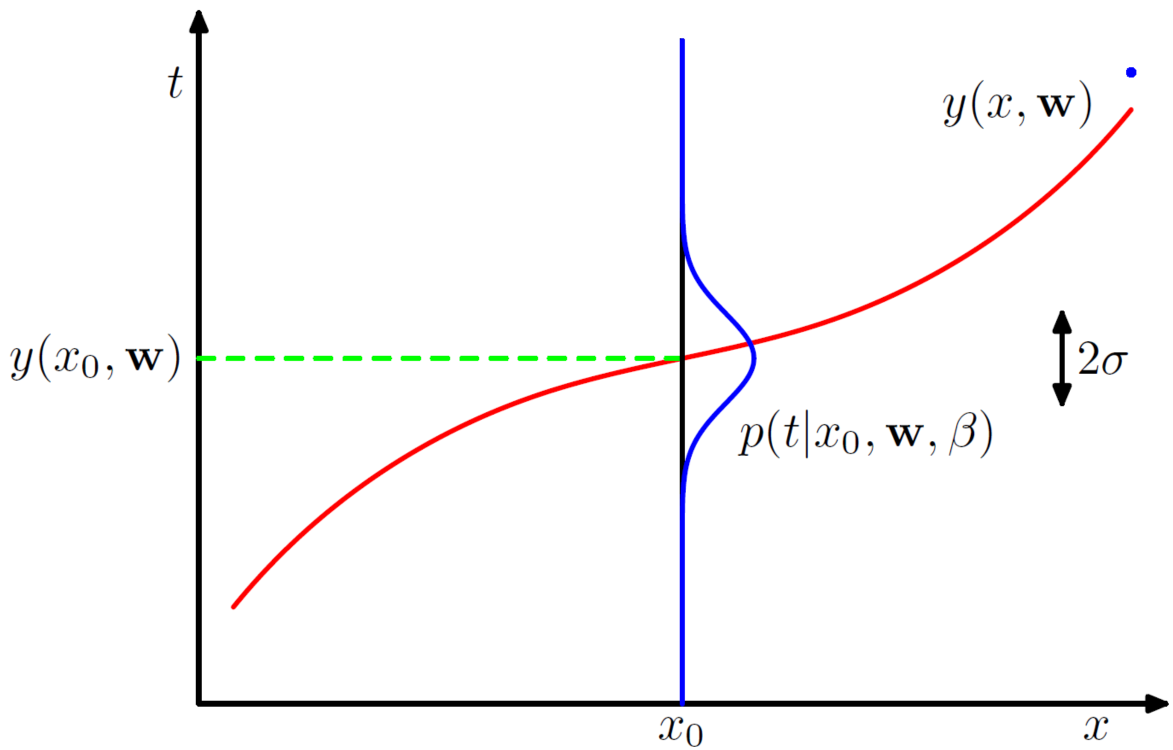
M*aximum likelihood (frequentist approach)*:

- **w** is set to the value that maximizes the likelihood function *p*(*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*.

A more probabilistic approach to curve fitting:

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

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



training data $\{\mathbf{x}, \mathbf{t}\}$

Likelihood function:

$$p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}\left(t_n|y(x_n, \mathbf{w}), \beta^{-1}\right)$$

$$\ln p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta) = -\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \cancel{\frac{N}{2}\ln\beta} - \cancel{\frac{N}{2}\ln(2\pi)}$$

The sum-of-squares error function arises as a consequence of **maximizing likelihood under the assumption of a Gaussian noise distribution.**

$$\boxed{p(t|x, \mathbf{w}_{\mathrm{ML}}, \beta_{\mathrm{ML}}) = \mathcal{N}\left(t|y(x, \mathbf{w}_{\mathrm{ML}}), \beta_{\mathrm{ML}}^{-1}\right)}$$

predictive distribution that gives the probability distribution over *t*, rather than simply a point estimate

Maximum a Posteriori (MAP):

$$\text{posterior} \propto \text{likelihood} \times \text{prior}$$

hyperparameter

$$p(\mathbf{w}|\mathbf{x}, \mathbf{t}, \alpha, \beta) \propto p(\mathbf{t}|\mathbf{x}, \mathbf{w}, \beta)p(\mathbf{w}|\alpha)$$

$$p(\mathbf{w}|\alpha) = \mathcal{N}(\mathbf{w}|\mathbf{0}, \alpha^{-1}\mathbf{I}) = \left(\frac{\alpha}{2\pi}\right)^{(M+1)/2} \exp\left\{-\frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}\right\}$$

$$\boxed{\frac{\beta}{2}\sum_{n=1}^{N}\{y(x_n, \mathbf{w}) - t_n\}^2 + \frac{\alpha}{2}\mathbf{w}^{\mathrm{T}}\mathbf{w}}$$

maximizing the posterior distribution is equivalent to minimizing the regularized sum-of-squares error function

Full Bayesian approach:

$$p(t|x, \mathbf{x}, \mathbf{t}) = \int p(t|x, \mathbf{w})p(\mathbf{w}|\mathbf{x}, \mathbf{t})\,\mathrm{d}\mathbf{w}$$