Introduction

Chia-Ping Chen

Professor
National Sun Yat-sen University
Department of Computer Science and Engineering

Machine Learning

Topics

- Example: Curve Fitting
- Probability Theory
- Model Selection
- Curse of Dimensionality
- Decision Theory
- Information Theory

Pattern Recognition

Patterns are structures within data.

- shapes
- objects
- words/phrases
- phonemes
- stock market

Recognition of patterns leads to scientific breakthroughs.

Brahe	$\rightarrow$	Kepler	$\rightarrow$	Newton
(data)		(description)		(explanation)

In this course, we are interested in learning patterns from a collection of data, and then recognizing patterns in unseen data.

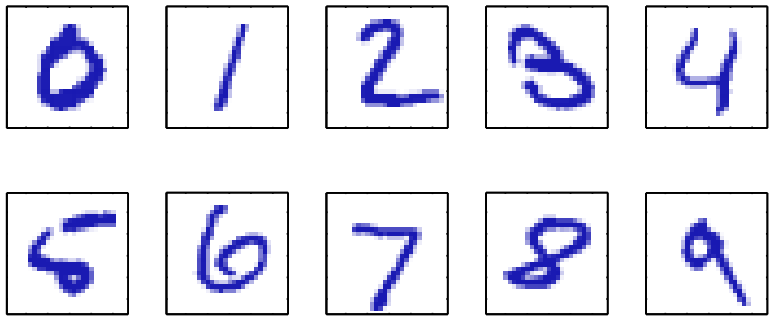
The use of computers (machines) to learn from data.

- classification
- regression
- transcription
- detection
- translation
- synthesis
- much more

A machine learning system is characterized by **task**, **data**, **model**, **algorithm**, and **evaluation**.

- A task is specified in terms of input and output.
- Task-related data is used for learning or inference.
- A model characterizes the mathematical relationship between input and output.
- An algorithm learns model parameters from training data or does inference on test data.
- Evaluation measures system performance.

Example: Hand-written Digit Recognition



- task: to recognize the digit in an image
- data: images of hand-written digits
- model: a probability model $P(C_k|\boldsymbol{x})$
- algorithm: learns model parameters or decides class
- evaluation: recognition accuracy

Example: Polynomial Curve Fitting

Polynomial Curve Fitting

- Task: to output $y(x)$ as the prediction for input x
- Data: $\mathcal{D} = \{(x_n, t_n)\}_{n=1}^{N}$ and $\mathcal{D}' = \{(x'_n, t'_n)\}_{n=1}^{N'}$
- Model: a **polynomial function** mapping input x to output y

$$y(x, \boldsymbol{w}) = w_0 + w_1 x + \cdots + w_M x^M$$

- Algorithm: to decide $\boldsymbol{w}$ by minimizing a **cost function**

$$\boldsymbol{w}^*(\mathcal{D}) = \arg\min_{\boldsymbol{w}} l(\mathcal{D}; \boldsymbol{w})$$

- Evaluation: mean squared error

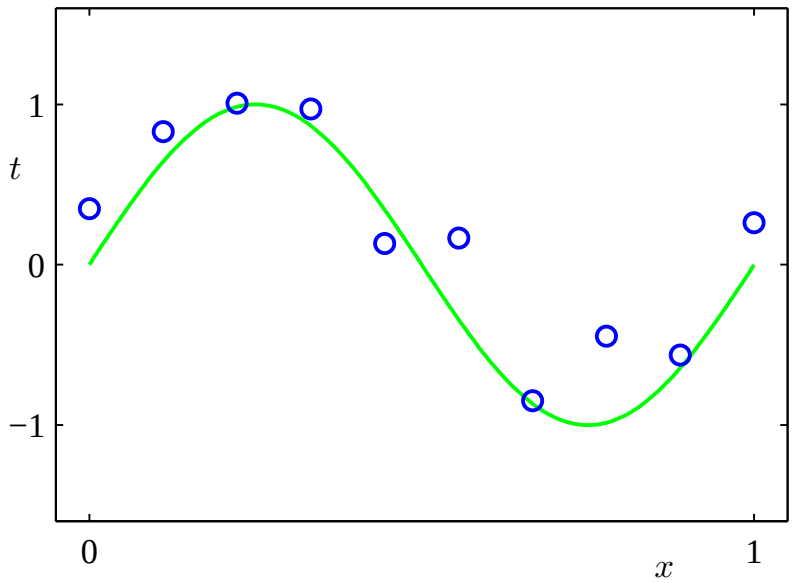
$$\text{MSE}(\mathcal{D}') = \frac{1}{N'} \sum_{n=1}^{N'} (y(x'_n, \boldsymbol{w}^*) - t'_n)^2$$

Training Data

Let the training data $\mathcal{D} = \{(x_n, t_n)\}_{n=1}^{N}$ be generated by

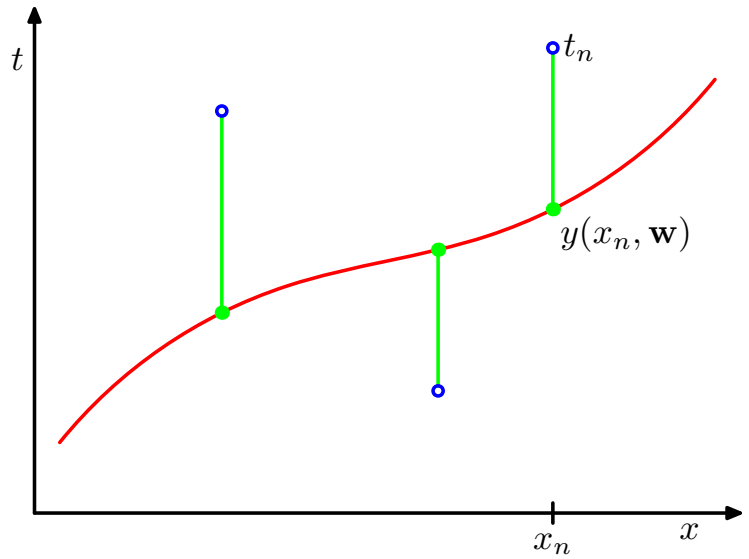
$$t_n = \sin(2\pi x_n) + \epsilon_n$$

The points $x_1, \ldots, x_{10}$ are equally spaced between 0 and 1, and the noises $\epsilon_1, \ldots, \epsilon_{10}$ are **i.i.d.** zero-mean Gaussian random variables.

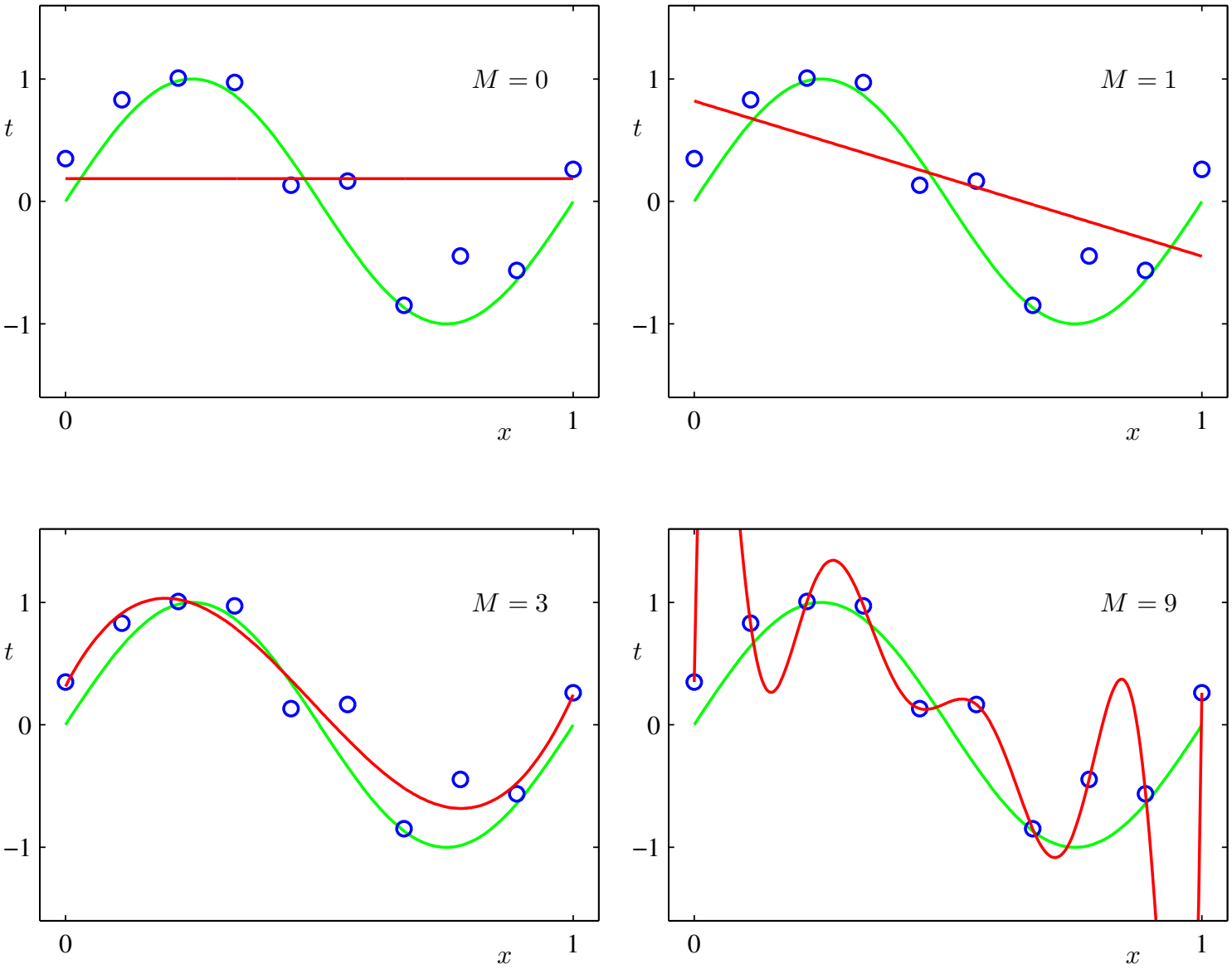


A simple cost function for learning a prediction function is **the sum of squared errors** on the training set.

$$\boldsymbol{w}^*(\mathcal{D}) = \arg\min_{\boldsymbol{w}} \overbrace{\frac{1}{2}\sum_{n=1}^{N}\{y(x_n, \boldsymbol{w}) - t_n\}^2}^{l(\mathcal{D};\boldsymbol{w})}$$



Results



Over-Fitting

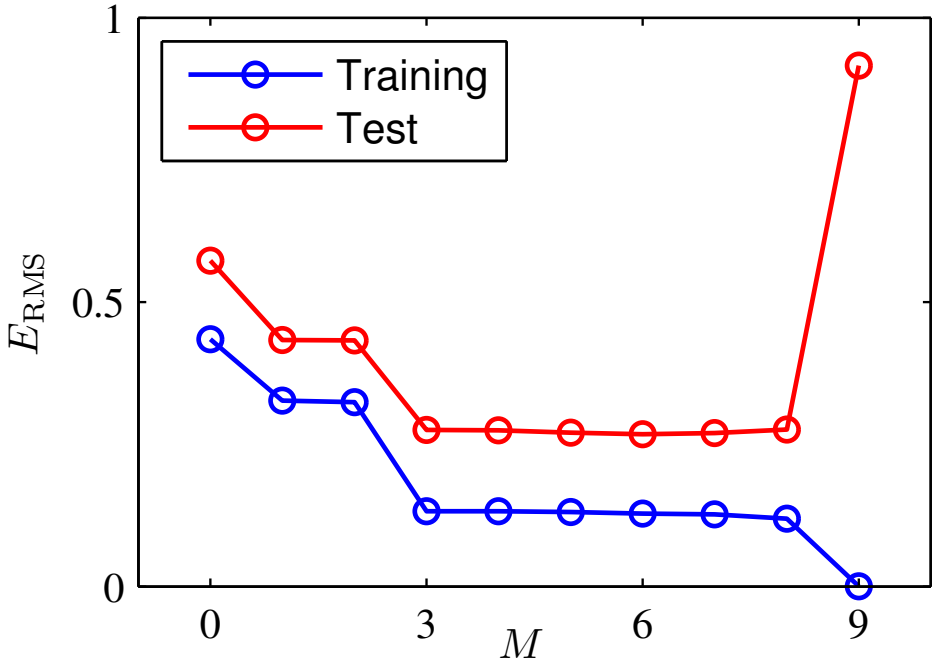
For $M = 9$, the learned output function fits the training data well, but fits unseen test data poorly.

- M controls the number of parameters in $y(x, \boldsymbol{w})$.
- For $M = 9$, the learned output function $y(x, \boldsymbol{w}^*)$ passes all 10 points exactly.
- However, $y(x, \boldsymbol{w}^*)$ is a very poor approximation to the true function $\sin(2\pi x)$ used to generate data.
- What happens here is called **over-fitting**.

A simple performance measure for curve fitting is the **root-mean-square error**

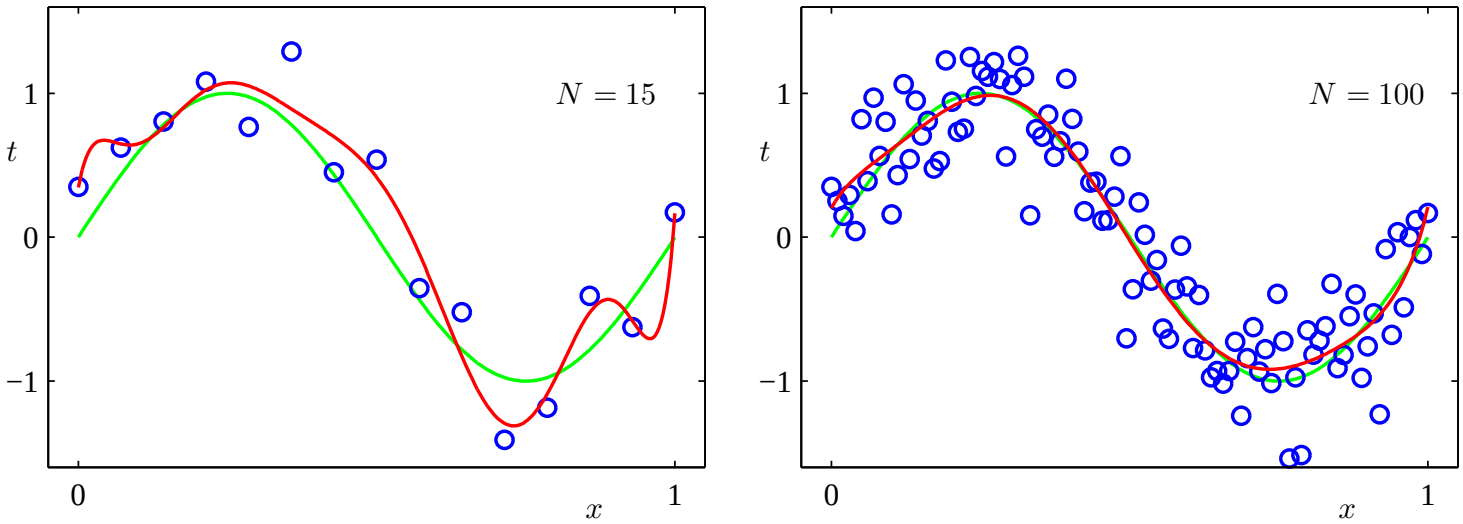
$$E_{\text{RMS}} = \sqrt{\frac{2E(\boldsymbol{w}^*)}{N}}$$

The RMS errors on the training set and test set are different.



More Data

One way to deal with over-fitting is to add data points to the training set.



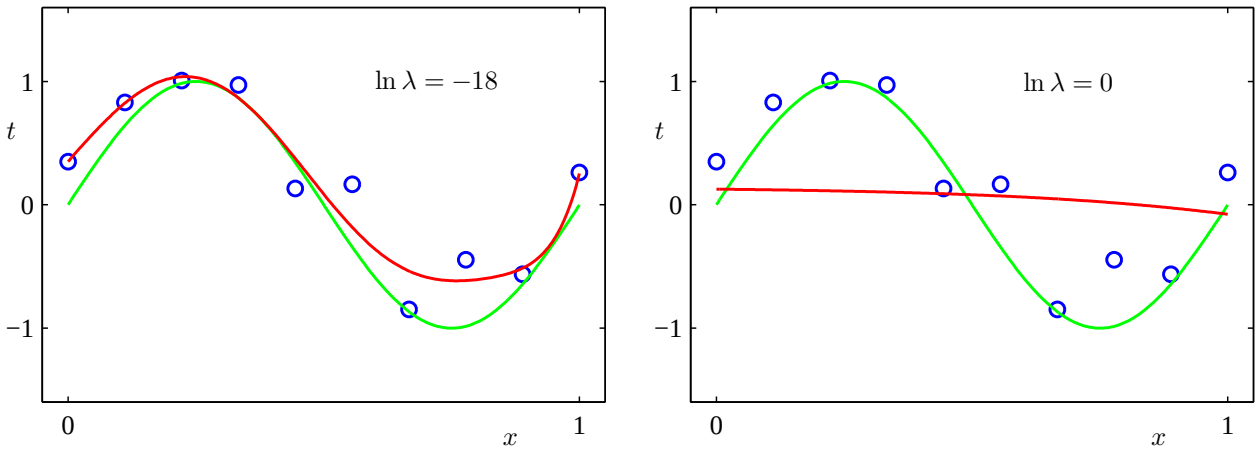
For $M = 9$, increasing N overcomes over-fitting.

Norm Penalty

Another way to deal with over-fitting is to add **norm penalty** to the cost function

$$\tilde{E}(\boldsymbol{w}) = E(\boldsymbol{w}) + \frac{\lambda}{2}\|\boldsymbol{w}\|^2$$

Norm penalty is an example of **regularization**. It favors parameters with small magnitudes. λ controls the degree of regularization.



The results for $M = 9$ with different λs.

Probability Theory (Quick Review)

Probability theory is fundamental to AI.

- input completion

$$w^*_{\geq n} = \arg\max_{w_{\geq n}} P(w_{\geq n}|w_{<n})$$

- speech recognition

$$W^* = \arg\max_{W} P(W|A)$$

- machine translation

$$e^* = \arg\max_{e} P(e|f)$$

- information retrieval

$$d^* = \arg\max_{d} P(d|q)$$

The **sum rule** and **product rule** of probability are fundamental relationships between the joint probability, the marginal probability, and the conditional probability of two (groups of) random variables.

Let X and Y be random variables.

- sum rule

$$P(X) = \sum_{Y} P(X, Y)$$

$$P(Y) = \sum_{X} P(X, Y)$$

- product rule

$$P(X, Y) = P(Y|X)P(X)$$

$$P(X, Y) = P(X|Y)P(Y)$$

Bayes' Rule

A conditional probability can be derived from the conditional probability in the other direction.

Bayes' rule. Let X and Y be random variables.

$$P(Y|X) = \frac{P(X|Y)P(Y)}{\sum_{Y'} P(X|Y')P(Y')}$$

Proof.

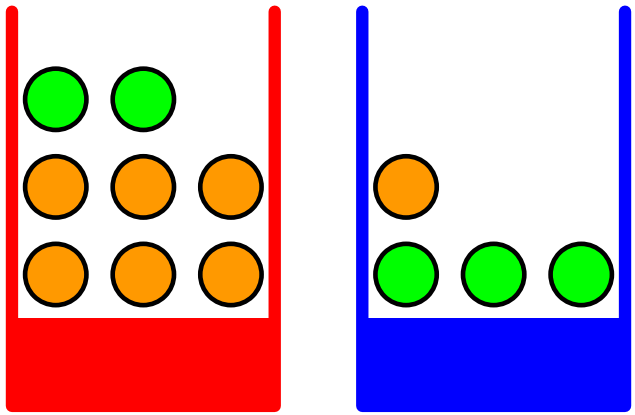
$$P(Y|X) = \frac{P(X,Y)}{P(X)} = \frac{P(X|Y)P(Y)}{P(X)} = \frac{P(X|Y)P(Y)}{\sum_{Y'} P(X,Y')}$$

$$= \frac{P(X|Y)P(Y)}{\sum_{Y'} P(X|Y')P(Y')}$$

Example

There are 2 apples and 6 oranges in a red box. There are 3 apples and 1 orange in a blue box. Choose a box at random (red chosen with probability 0.4) and then choose a fruit from the chosen box.

- What is the probability that an orange is chosen?
- Given an orange is chosen, what is the probability that the red box has been chosen?



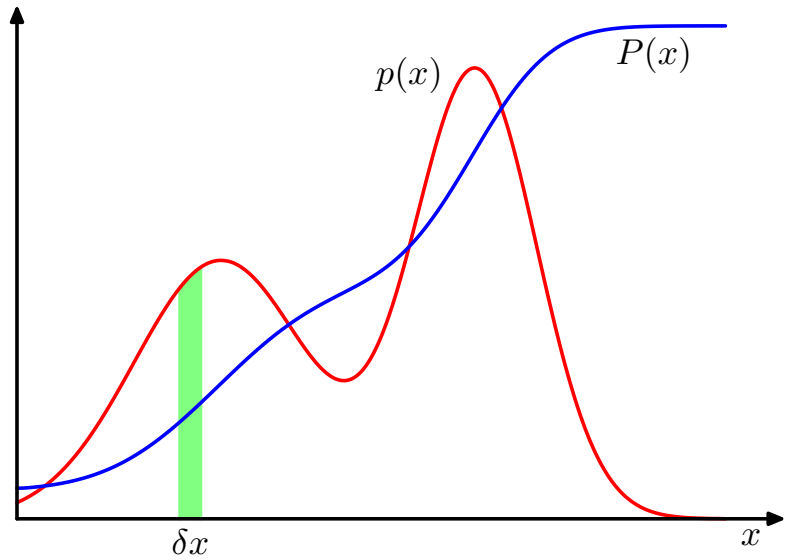
Let $Y \in \{r, b\}$ be the chosen box, and $X \in \{a, o\}$ be the chosen fruit.

Probability Density Function

Definition. Let X be a continuous random variable. The probability of event $X \in (x, x+\delta)$ is

$$p_X(x)\delta + o(\delta)$$

$p_X(x)$ is the **probability density function** (PDF) of X.



Properties of PDF

- density

$$P(X \in (a,b)) = \int_a^b p_X(x)dx$$

- non-negativity

$$p_X(x) \geq 0$$

- normalization

$$\int p_X(x)dx = 1$$

- marginal probability

$$p_X(x) = \int p_{XY}(x,y)dy$$

- conditional probability

$$p_{Y|X}(y|x) = \frac{p_{XY}(x,y)}{p_X(x)}$$

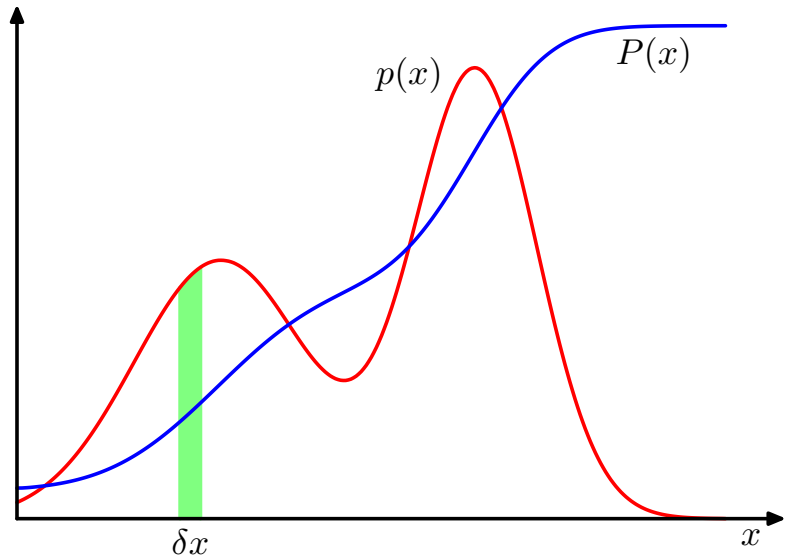
Cumulative Distribution Function

Definition. The probability of a random variable can also be specified by a **cumulative distribution function** (CDF)

$$F_X(x) = P(X \leq x)$$

The CDF and the PDF of X are related by

$$p_X(x) = \frac{d}{dx} F_X(x)$$



Inverse Function Method

Let X be a continuous random variable with CDF F_X. A sample of X can be generated by applying F_X^{-1} to a sample drawn from the uniform distribution.

- Draw an instance y from

$$Y \sim \textbf{uniform}(0, 1)$$

- Transform y by

$$x = F_X^{-1}(y)$$

Let X be a continuous random variable with PDF $p_X(x)$ and CDF $F_X(x)$, and $Y = F_X(X)$. Then $Y \sim$ **uniform**$(0, 1)$.

We have $0 \leq Y \leq 1$. Also, event $X \in (x, x + \delta_x)$ is equivalent to $Y \in (y, y + \delta_y)$, where $y = F_X(x)$. So

$$p_X(x)|\delta_x| + o(\delta_x) = p_Y(y)|\delta_y| + o(\delta_y)$$

Consider $\delta_x \to 0^+$.

$$p_Y(y) = p_X(x) \left|\frac{\delta_x}{\delta_y}\right| = p_X(x) \left|\frac{dy}{dx}\right|^{-1} = p_X(x) \left(p_X(x)\right)^{-1} = 1$$

Thus

$$Y \sim \textbf{uniform}(0, 1)$$

Example: Exponential Random Variable

Consider $X \sim$ **exponential**(λ). Since the CDF of X is

$$F_X(x) = 1 - e^{-\lambda x}$$

we have $Y = F_X(X) = 1 - e^{-\lambda X}$ is **uniform**$(0, 1)$.

An instance x of X can be obtained by drawing an instance y of Y and transforming y to x by

$$x = F_X^{-1}(y) = -\frac{1}{\lambda} \log(1 - y)$$

Definition. Let X be a random variable. The **expectation** of X is

$$\mathbb{E}[X] = \sum_x x\, P(X = x)$$

or

$$\mathbb{E}[X] = \int x\, p_X(x) dx$$

Expectation can be taken with respect to conditional probability, called **conditional expectation**. That is

$$\mathbb{E}[X|Y = y] = \sum_x x\, P(X = x|Y = y)$$

or

$$\mathbb{E}[X|Y = y] = \int x\, p_{X|Y}(x|y) dx$$

Let X be a random variable and $f(\cdot)$ be a function. Then $f(X)$ is a random variable.

The expectation of $f(X)$ is

$$\mathbb{E}[f] = \int f(x)p_X(x)dx$$

or

$$\mathbb{E}[f] = \sum_x f(x)P(X = x)$$

Sample Mean

Definition. The **sample mean** of $f(X)$ based on sample $\{x_1, \ldots, x_N\}$ of X is

$$\mathbb{E}[f] \approx \frac{1}{N} \sum_{n=1}^{N} f(x_n)$$

Law of large numbers. A sequence of the sample means of a random variable converges to the expectation of the random variable.

Definition. Let X be a random variable. The **variance** of X is

$$\text{var}[X] = \mathbb{E}\left[(X - \mathbb{E}[X])^2\right]$$

Let $f(\cdot)$ be a function. The variance of $f(X)$ is

$$\text{var}[f] = \mathbb{E}\left[(f(X) - \mathbb{E}[f(X)])^2\right]$$

Definition. Let X and Y be random variables. The **covariance** of X and Y is

$$\text{cov}[X, Y] = \mathbb{E}\left[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])\right]$$

It can be shown that

$$\text{cov}[X, Y] = \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]$$

Covariance Matrix

Definition. Let $\mathbf{x}$ and $\mathbf{y}$ be random vectors. The **covariance matrix** of $\mathbf{x}$ and $\mathbf{y}$ is

$$\text{cov}[\mathbf{x}, \mathbf{y}] = \mathbf{\Sigma} = \mathbb{E}\left[(\mathbf{x} - \mathbb{E}[\mathbf{x}])(\mathbf{y} - \mathbb{E}[\mathbf{y}])^{\mathbf{T}}\right]$$

Furthermore, the **covariance matrix** of $\mathbf{x}$ is

$$\text{cov}[\mathbf{x}] = \text{cov}[\mathbf{x}, \mathbf{x}]$$

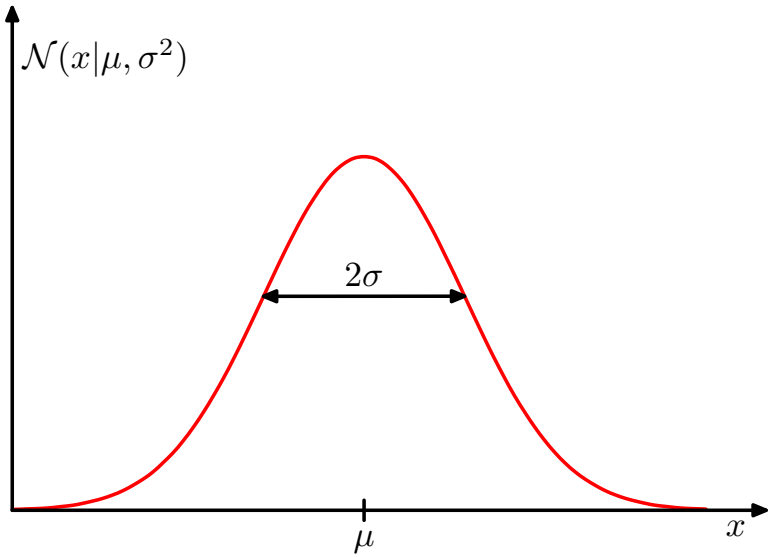
The covariance matrix of $\mathbf{x}$ and $\mathbf{y}$ consists of pairwise covariances

$$\sigma_{ij} = \mathbb{E}\left[(x_i - \mathbb{E}[x_i])(y_j - \mathbb{E}[y_j])\right]$$

Definition. A **Gaussian PDF** is

$$\mathcal{N}(x|\mu,\sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x-\mu)^2\right\}$$

A **Gaussian random variable** has a Gaussian PDF.



Normalization

$$\int \mathcal{N}(x|\mu, \sigma^2)dx = 1$$

Expectation

$$\mathbb{E}[x] = \mu$$

Variance

$$\text{var}[x] = \sigma^2$$

Definition. Let $\mathbf{x}$ be a random vector. $\mathbf{x}$ is a **Gaussian random vector** if it has PDF

$$\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}$$

- The parameters are in $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$.
- $\boldsymbol{\mu}$ is the mean vector of $\mathbf{x}$ and $\boldsymbol{\Sigma}$ is the covariance matrix of $\mathbf{x}$.

The logarithm of a Gaussian PDF is

$$\begin{aligned}
&\log\left[\frac{1}{(2\pi)^{D/2}}\frac{1}{|\boldsymbol{\Sigma}|^{1/2}}\exp\left\{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})\right\}\right] \\
&= -\frac{D}{2}\log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}| - \frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu}) \\
&= -\frac{1}{2}\boldsymbol{x}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \boldsymbol{\mu}^T\boldsymbol{\Sigma}^{-1}\boldsymbol{x} + \text{const}
\end{aligned}$$

It is a quadratic function of $\boldsymbol{x}$.

- The second-order term depends on the covariance matrix.
- The first-order term depends on the covariance matrix and the mean vector.

Curve-Fitting with Probability Model

- We want to fit a data set $\mathcal{D} = \{(x_n, t_n)\}_{n=1}^N$.
- Suppose that the dataset is generated according to

$$t_n = u(x_n) + \epsilon_n, \ n = 1, \ldots, N$$

  where $\epsilon_1, \ldots, \epsilon_N$ are **i.i.d.** random variables.
- The function $u(x)$ is unknown to us. The distribution of ϵ_n is also unknown.
- Hence, we assume a **parametric prediction function** $y(x, \boldsymbol{w})$ to approximate $u(x)$ and a **parametric distribution** for ϵ_n.
- After setting up a probability model, we can learn the parameters in the model from $\mathcal{D}$ via **maximum likelihood** or **Bayesian learning**.

Learning Schemes

- **Maximum likelihood.** Treat the model parameters as unknowns. Learning is the process of deciding optimal values.
- **Bayesian learning.** Treat the model parameters as random variables (with parametric distribution). Learning is the process of updating their (parametric) distribution.

Earlier, we found optimal parameters $\boldsymbol{w}^*$ by minimizing a cost function

$$\boldsymbol{w}^* = \arg\max_{\boldsymbol{w}} E(\boldsymbol{w})$$

For example, with norm penalty, we minimized

$$\begin{aligned} E(\boldsymbol{w}) &= E_D(\boldsymbol{w}) + E_W(\boldsymbol{w}) \\ &= \sum_{n=1}^{N} E(y(x_n, \boldsymbol{w}), t_n) + \lambda R(\boldsymbol{w}) \end{aligned}$$

This learning method has nothing to do with probability. Nonetheless, it can be derived from probabilistic models.

Gaussian Noise Model

We often assume a data point is corrupted by a Gaussian noise. More specifically, we assume

$$t = u(x) + \epsilon$$

where $u(x)$ is a function and $\epsilon \sim \mathcal{N}(e|0, \beta^{-1})$ is Gaussian.

It follows that the conditional distribution of t given x is Gaussian

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

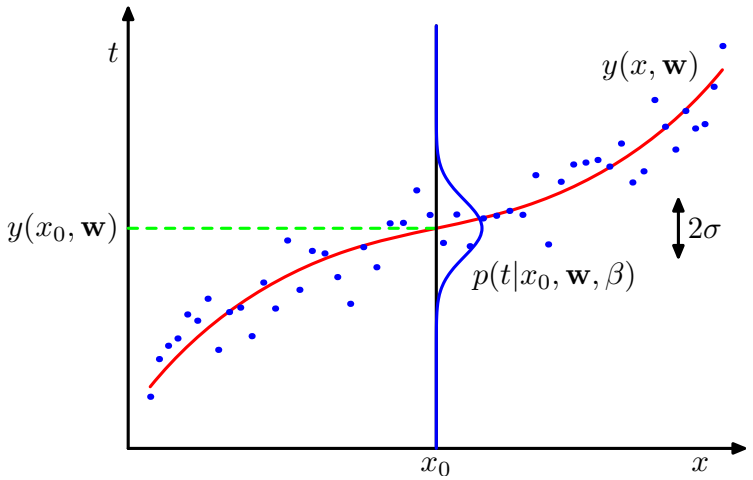
Approximate the True Function

As $u(x)$ is unknown to us, we use a parametric function $y(x, \boldsymbol{w})$ to approximate $u(x)$, i.e.

$$u(x) \approx y(x, \boldsymbol{w})$$

With this approximation and Gaussian noise assumption, we have

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



Data Likelihood

The likelihood of a data point (x_n, t_n) is

$$p(t_n|x_n) \approx \mathcal{N}(t_n|y(x_n, \boldsymbol{w}), \beta^{-1})$$

The data likelihood of $\mathcal{D} = \{(x_n, t_n)\}_{n=1}^N$ is

$$p(\mathcal{D}|\boldsymbol{w}, \beta) = \prod_{n=1}^{N} \mathcal{N}(t_n|y(x_n, \boldsymbol{w}), \beta^{-1})$$

and the log data-likelihood of $\mathcal{D}$ is

$$\log p(\mathcal{D}|\boldsymbol{w}, \beta) = -\frac{\beta}{2} \sum_{n=1}^{N} \{y(x_n, \boldsymbol{w}) - t_n\}^2 + \frac{N}{2} \log \beta - \frac{N}{2} \log(2\pi)$$

Definition. The parameter values maximizing data likelihood are **maximum-likelihood** (ML) estimates.

For the above Gaussian noise model, the maximum-likelihood estimate is

$$\boldsymbol{w}_{\mathsf{ML}} = \arg\min_{\boldsymbol{w}} \sum_{n=1}^{N} \{y(x_n, \boldsymbol{w}) - t_n\}^2$$

$$\frac{1}{\beta_{\mathsf{ML}}} = \frac{1}{N} \sum_{n=1}^{N} (y(x_n, \boldsymbol{w}_{\mathsf{ML}}) - t_n)^2$$

Note $\boldsymbol{w}_{\mathsf{ML}}$ is the same as $\boldsymbol{w}^*$ which minimizes the sum of squared errors. Here we see that the results based on squared-error cost function can be re-derived based on a Gaussian noise model.

Definition. In machine learning with probability models, the negative log-likelihood function is the **error function**.

Following this definition, we have

$$\text{maximum likelihood} = \text{minimum error}$$

In Bayesian learning, we treat model parameters as random variables and update their distribution with data.

Specifically

- Assume a **prior distribution** $p(\boldsymbol{w})$ for parameters $\mathbf{w}$.
- Update the distribution of $\mathbf{w}$ to **posterior distribution** $p(\boldsymbol{w}|\mathcal{D})$ by the Bayes' rule

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

Let us assume a Gaussian prior distribution for $\mathbf{w}$

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

For the posterior distribution of $\mathbf{w}$

$$p(\boldsymbol{w}|\mathcal{D}, \alpha, \beta) \propto p(\boldsymbol{w}, \mathcal{D}|\alpha, \beta) = p(\mathcal{D}|\boldsymbol{w}, \alpha, \beta)p(\boldsymbol{w}|\alpha, \beta)$$
$$= p(\mathcal{D}|\boldsymbol{w}, \beta)p(\boldsymbol{w}|\alpha)$$

Taking logarithm, we get

$$\log p(\boldsymbol{w}|\mathcal{D}, \alpha, \beta)$$
$$= -\frac{\beta}{2}\sum\{y(x_n, \boldsymbol{w}) - t_n\}^2 - \frac{\alpha}{2}\boldsymbol{w}^T\boldsymbol{w} + \text{ (terms independent of } \boldsymbol{w})$$

Maximum a Posteriori (MAP)

Definition. Suppose model parameters are treated as random variables. The parameter values maximizing posterior distribution are **maximum a posteriori** (MAP) estimates.

In the current example, we have

$$\begin{aligned}
\boldsymbol{w}_{\text{MAP}} &= \arg\max_{\boldsymbol{w}} \log p(\boldsymbol{w}|\mathcal{D}, \alpha, \beta) \\
&= \arg\max_{\boldsymbol{w}} -\frac{\beta}{2}\sum\{y(x_n, \boldsymbol{w}) - t_n\}^2 - \frac{\alpha}{2}\boldsymbol{w}^T\boldsymbol{w} \\
&= \arg\min_{\boldsymbol{w}} \frac{\beta}{2}\sum\{y(x_n, \boldsymbol{w}) - t_n\}^2 + \frac{\alpha}{2}\boldsymbol{w}^T\boldsymbol{w}
\end{aligned}$$

So $\boldsymbol{w}_{\text{MAP}}$ is the same as $\boldsymbol{w}^*$ which minimizes the sum of squared errors with norm penalty. Again, we see that the results based on cost function can be re-derived based on a probability model.

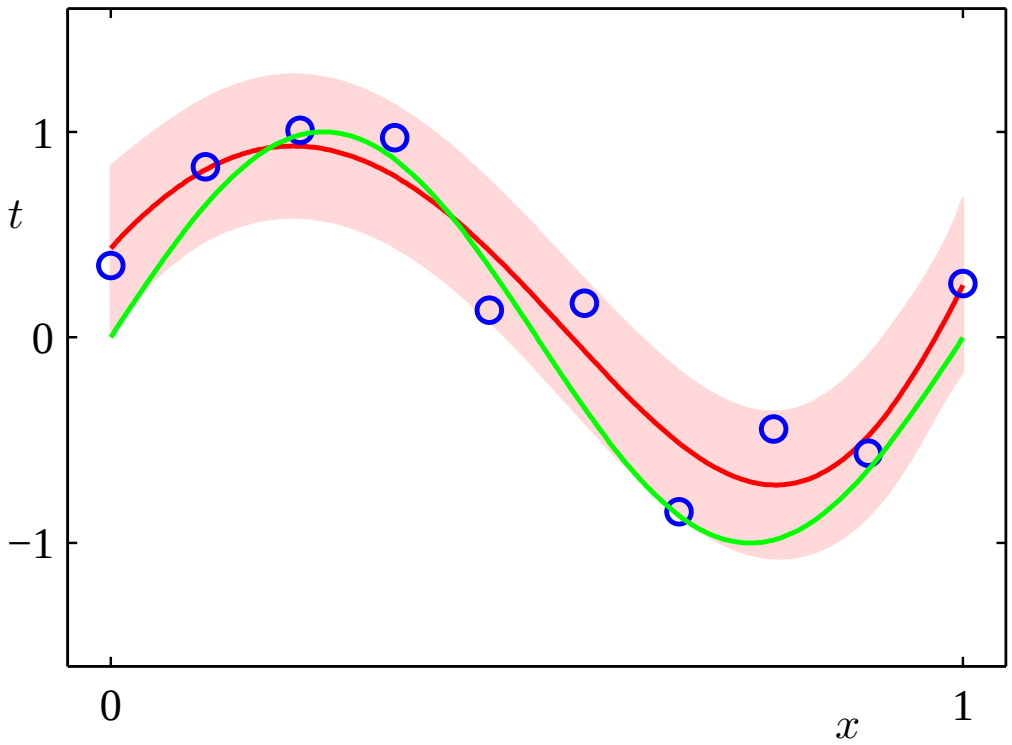
It is unnecessary to settle for $\boldsymbol{w}_{\text{MAP}}$ or $\boldsymbol{w}_{\text{ML}}$, which are merely **point estimates**. With the posterior distribution of $\mathbf{w}$, the conditional distribution of the target variable t given x is

$$\begin{aligned} p(t|x, \mathcal{D}, \alpha, \beta) &= \int p(t, \boldsymbol{w}|x, \mathcal{D}, \alpha, \beta) d\boldsymbol{w} \\ &= \int p(t|x, \boldsymbol{w}, \mathcal{D}, \alpha, \beta) p(\boldsymbol{w}|x, \mathcal{D}, \alpha, \beta) d\boldsymbol{w} \\ &= \int p(t|y(x, \boldsymbol{w}), \beta) p(\boldsymbol{w}|\mathcal{D}, \alpha) d\boldsymbol{w} \end{aligned}$$

It will become clear later that the distribution is Gaussian, i.e.

$$p(t|x, \mathcal{D}, \alpha, \beta) = \mathcal{N}(t|m(x, \mathcal{D}, \alpha, \beta), s^2(x, \mathcal{D}, \alpha, \beta))$$

Full Bayesian Learning Result



$M = 9$. Green curve: $f(x)$. Red curve: $\mathbb{E}[t|x]$. Shade: $\pm\text{var}(t|x)$.

Model Selection

Hyper-parameters

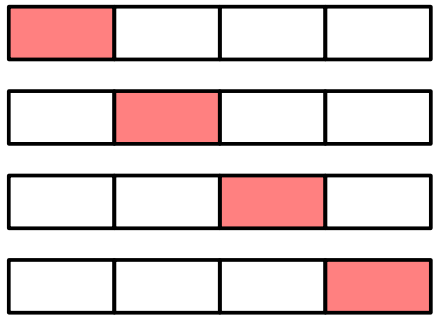
Hyper-parameters are the parameters that are pre-determined before the training algorithm begins to run, such as

- parameters related to model complexity, e.g. M
- parameters in the distribution of model parameters, e.g. α
- parameters in the learning algorithm, e.g. learning rate and batch size

Validation Set and Cross Validation

Suppose we have a set of candidate models. How do we choose from them?

- One way for model selection is to use a **validation set**.
  - Let each candidate model be trained with the training set.
  - Choose the candidate model with the best performance on the validation set.
- When data is limited, we use **cross-validation**.
  - Training data is partitioned into groups, each of which serves as a **held-out set**, i.e. not used for training but for validation.
  - Choose the model with the best average performance.



We want to fit the data well without over-fitting.

- **Information criteria.** The over-fitting issue can be handled by the addition of a penalty term for complex models. For example, we can choose the model that maximizes

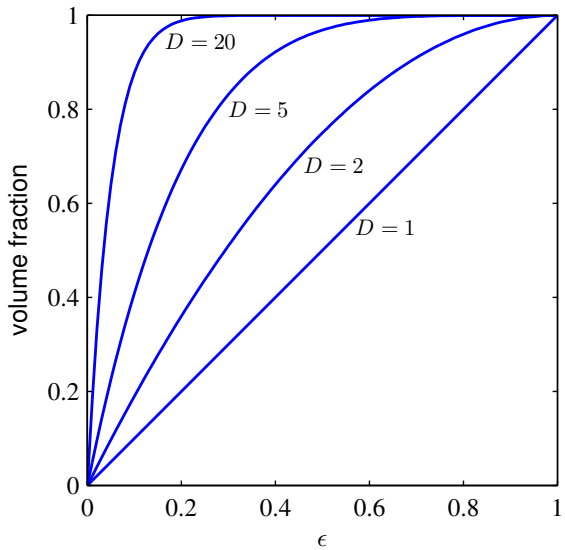
$$\log p(\mathcal{D}|\boldsymbol{w}_{\mathsf{ML}}) - M$$

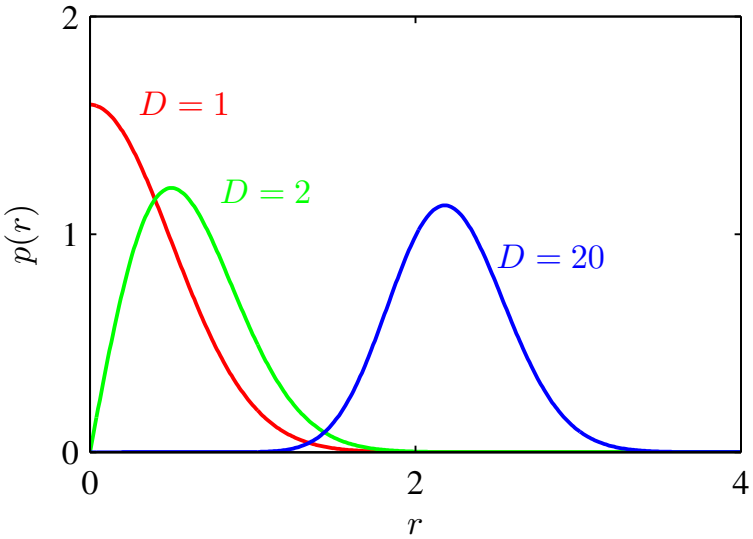
  where M is the number of parameters in a candidate model and $\boldsymbol{w}_{\mathsf{ML}}$ is the maximum-likelihood estimate.
- Compute model evidence (to be shown later)

The Curse of Dimensionality

High-Dimensional Space

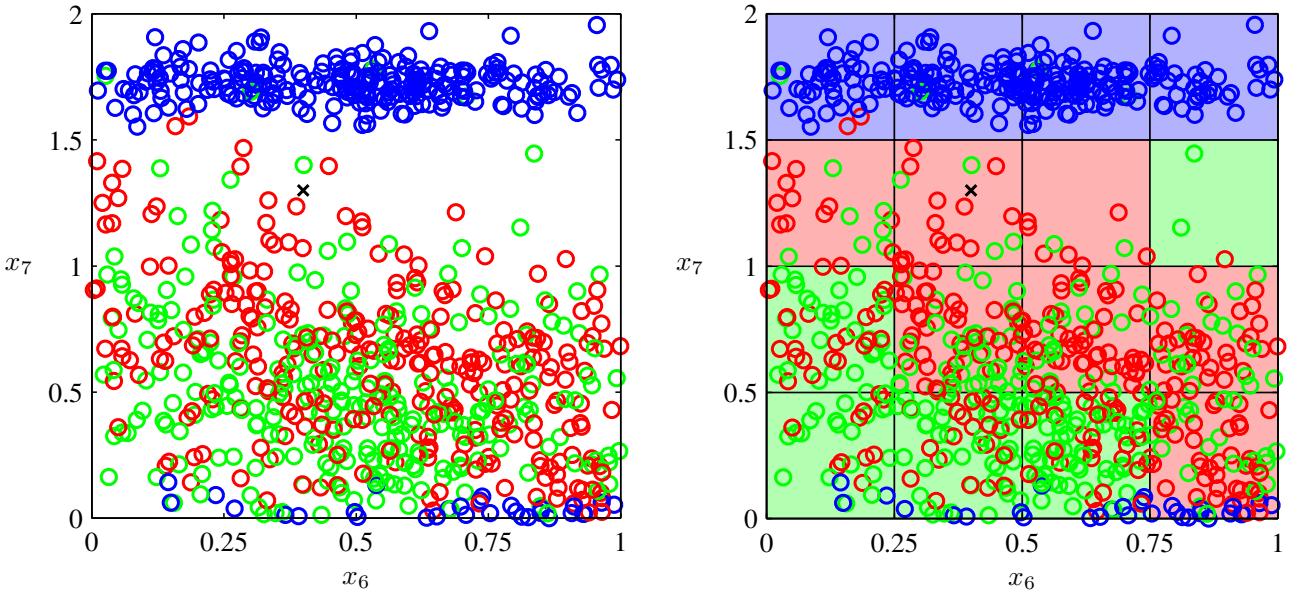
- In a real problem, we often has many input variables.
- It means we often have to work with a high-dimensional space.
- High-dimension spaces are different from low-dimension spaces in many ways.





Left: the volume of a shell of fraction ϵ of the radius of a sphere of dimension D. Right: the probability density with respect to r of an isotropic Gaussian vector of dimension D.

Data in High-Dimensional Space



Left: 3 classes of data points in 3 colors.

Right: partition into meshes for classification.

Issue: The number of meshes grows exponentially with dimension.

Decision Theory

Suppose we have a data set $\mathcal{D} = \{\boldsymbol{x}_n, \boldsymbol{t}_n\}_{n=1}^N$. Based on $\mathcal{D}$, our goal is to make an optimal decision for any input vector $\boldsymbol{x}$. We can break down this problem into two stages.

- **Inference.** Learn the joint distribution $p(\boldsymbol{x}, \boldsymbol{t})$ from $\mathcal{D}$. This is the difficult part.
- **Decision.** With $p(\boldsymbol{x}, \boldsymbol{t})$, make an optimal decision for any input vector $\boldsymbol{x}$. This part is relatively simple.

The **decision theory** concerns with the decision stage.

Classification: An Example

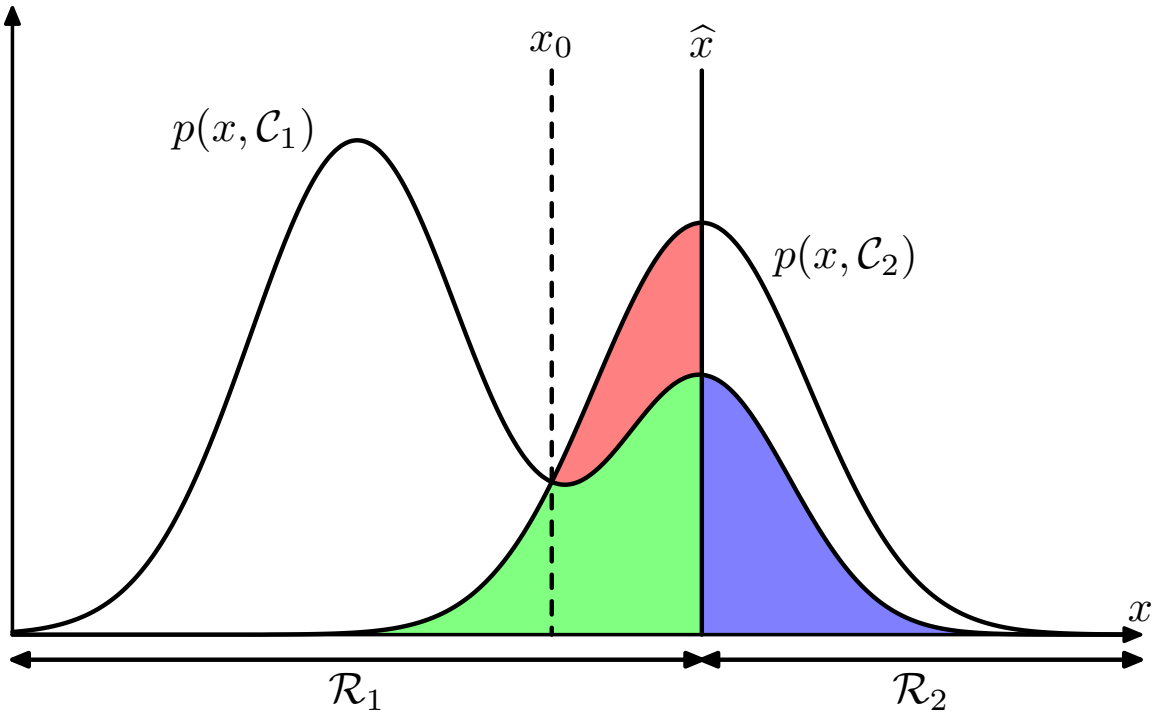
Consider a medical diagnosis problem in which we have taken an x-ray image of a patient, and we wish to decide whether the patient has a certain disease or not.

The input $\boldsymbol{x}$ is the set of pixel intensities in the image, and the output indicates a class: the presence (class $\mathcal{C}_1$) or non-presence (class $\mathcal{C}_2$) of the disease. What is the optimal output function $y(\boldsymbol{x})$?

Optimizing Decision Regions

Given $y(\boldsymbol{x})$, the input space is partitioned into decision regions, i.e. $\mathcal{R}_1$ (where $\mathcal{C}_1$ is the decision) and $\mathcal{R}_2$ (where $\mathcal{C}_2$ is the decision). Optimizing $y(\boldsymbol{x})$ is the same as optimizing decision regions.

Varying Decision Regions



A decision error occurs if an example of $\mathcal{C}_2$ (resp. $\mathcal{C}_1$) lies in $\mathcal{R}_1$ (resp. $\mathcal{R}_2$). Moving the decision boundary from $\hat{x}$ to x_0 reduces the probability of error by the area of the red region.

The error event is

$$E = (\{\mathbf{x} \in \mathcal{R}_1\} \cap \{\mathbf{t} = \mathcal{C}_2\}) \cup (\{\mathbf{x} \in \mathcal{R}_2\} \cap \{\mathbf{t} = \mathcal{C}_1\})$$

The probability of error is

$$\begin{aligned} P(E) &= P(\mathbf{x} \in \mathcal{R}_1, \mathbf{t} = \mathcal{C}_2) + P(\mathbf{x} \in \mathcal{R}_2, \mathbf{t} = \mathcal{C}_1) \\ &= \int_{\mathcal{R}_1} p(\boldsymbol{x}, \mathcal{C}_2)\, d\boldsymbol{x} + \int_{\mathcal{R}_2} p(\boldsymbol{x}, \mathcal{C}_1)\, d\boldsymbol{x} \end{aligned}$$

Consider a neighborhood $\delta\boldsymbol{x}$ of $\boldsymbol{x}$.

- For $\boldsymbol{x} \in \mathcal{R}_1$, the contribution of $\delta\boldsymbol{x}$ to $P(E)$ is $p(\boldsymbol{x}, \mathcal{C}_2)\delta\boldsymbol{x}$.
- For $\boldsymbol{x} \in \mathcal{R}_2$, the contribution of $\delta\boldsymbol{x}$ to $P(E)$ is $p(\boldsymbol{x}, \mathcal{C}_1)\delta\boldsymbol{x}$.

Hence, to minimize $P(E)$, we should let

$$p(\boldsymbol{x}, \mathcal{C}_1) > p(\boldsymbol{x}, \mathcal{C}_2) \;\Leftrightarrow\; \boldsymbol{x} \in \mathcal{C}_1$$

For K classes, the probability of correct decision is

$$P(C) = \sum_{k=1}^{K} P(\mathbf{x} \in \mathcal{R}_k, \mathbf{t} = \mathcal{C}_k)$$
$$= \sum_{k=1}^{K} \int_{\mathcal{R}_k} p(\boldsymbol{x}, \mathcal{C}_k)\, d\boldsymbol{x}$$

Hence, to maximize $P(C)$, we should let

$$\boldsymbol{x} \in \mathcal{C}_k \;\Leftrightarrow\; \mathcal{C}_k = \arg\max_{\mathcal{C}_j} p(\boldsymbol{x}, \mathcal{C}_j)$$

Decision Based on Posterior Probability

For 2 classes, the decision rule

$$p(\boldsymbol{x}, \mathcal{C}_1) > p(\boldsymbol{x}, \mathcal{C}_2) \Leftrightarrow \boldsymbol{x} \in \mathcal{C}_1$$

is equivalent to (dividing by $p(\boldsymbol{x})$)

$$P(\mathcal{C}_1|\boldsymbol{x}) > P(\mathcal{C}_2|\boldsymbol{x}) \Leftrightarrow \boldsymbol{x} \in \mathcal{C}_1$$

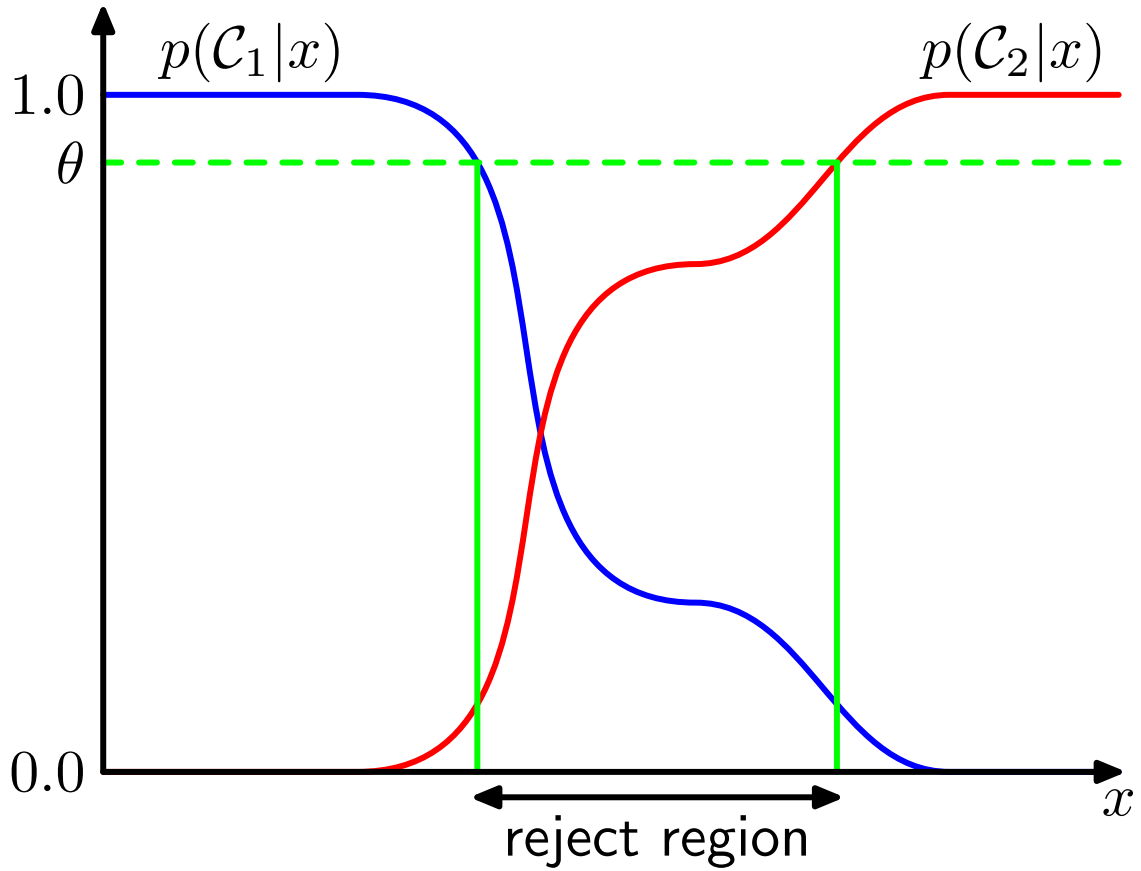
For K classes, the classification rule

$$\boldsymbol{x} \in \mathcal{C}_k \Leftrightarrow \mathcal{C}_k = \arg\max_{\mathcal{C}_j} p(\boldsymbol{x}, \mathcal{C}_j)$$

is equivalent to

$$\boldsymbol{x} \in \mathcal{C}_k \Leftrightarrow \mathcal{C}_k = \arg\max_{\mathcal{C}_j} p(\mathcal{C}_j|\boldsymbol{x})$$

Rejection Option



Costs of Decision Errors

Miss vs. False alarm

- (expensive) a patient with a disease is diagnosed as healthy
- (cheap) a healthy patient is diagnosed as having a disease

A **loss matrix** specifies the costs of different types of errors.

$$\boldsymbol{L} = \{l_{ij}\}$$

Specifically, a loss of l_{ij} is incurred if an example of class $\mathcal{C}_i$ is assigned to class $\mathcal{C}_j$ (lies in $\mathcal{R}_j$).

Expected Loss

Let $\mathcal{R}_1, \ldots, \mathcal{R}_K$ be the decision regions. The expected loss is

$$\begin{aligned}\mathbb{E}[L] &= \sum_i \sum_j l_{ij}\ P(\mathbf{x} \in \mathcal{R}_j, \mathbf{t} = \mathcal{C}_i) \\ &= \sum_j \int_{\mathcal{R}_j} \left(\sum_i l_{ij}\ p(\boldsymbol{x}, \mathcal{C}_i) \right) d\boldsymbol{x}\end{aligned}$$

To minimize $\mathbb{E}[L]$, we should let

$$\boldsymbol{x} \in \mathcal{C}_k \ \Leftrightarrow\ \mathcal{C}_k = \arg\max_{\mathcal{C}_j} \sum_{i=1}^{K} l_{ij}\, p(\boldsymbol{x}, \mathcal{C}_i)$$

We can apply decision theory to regression problems.

The **squared loss** between a target variable t and output $y(\mathbf{x})$ is

$$L = (y(\mathbf{x}) - t)^2$$

The expected loss given $\mathbf{x} = \boldsymbol{x}$ is

$$\mathbb{E}[L|\mathbf{x} = \boldsymbol{x}] = \int (y(\boldsymbol{x}) - t)^2 p(t|\boldsymbol{x}) dt$$

The total expected loss is

$$\begin{aligned}\mathbb{E}[L] = \mathbb{E}[\mathbb{E}[L|\mathbf{x}]] &= \int p(\boldsymbol{x}) d\boldsymbol{x} \int (y(\boldsymbol{x}) - t)^2 p(t|\boldsymbol{x}) dt \\ &= \int \int (y(\boldsymbol{x}) - t)^2 p(\boldsymbol{x}, t) d\boldsymbol{x} dt\end{aligned}$$

Minimization of Expected Squared Loss

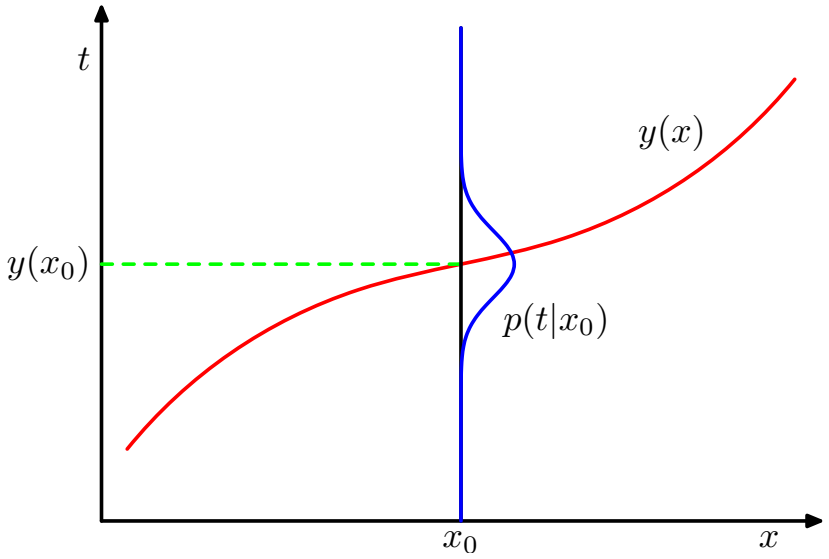
The prediction function that minimizes the expected loss can be determined by **calculus of variation**.

For the squared loss, the optimal prediction function satisfies

$$\frac{\delta \mathbb{E}[L]}{\delta y(\boldsymbol{x})} = 2 \int (y(\boldsymbol{x}) - t) p(\boldsymbol{x}, t) dt = 0$$

That is

$$y^*(\boldsymbol{x}) = \frac{\int t\, p(\boldsymbol{x}, t) dt}{p(\boldsymbol{x})} = \int t\, p(t|\boldsymbol{x}) dt = \mathbb{E}[t|\boldsymbol{x}]$$


t
$y(x)$
$y(x_0)$
$p(t|x_0)$
x_0
x

Information Theory

- The occurrence of a sure event (an event with probability 1) conveys no information.
- Information is related to probability: the less likely an event, the more information is conveyed when it occurs.
- Let A and B be independent events. The occurrence of A and B in sequence should convey an amount of information that is the sum of the information conveyed by the occurrence of A and the occurrence of B.

Thus, the information of the occurrence of an event is

$$I(A) = -\log P(A)$$

The unit is bit (resp. nat) when the base of logarithm is 2 (resp e).

Definition. Let X be a discrete random variable with distribution $p(x)$. The **entropy** of X is

$$H[X] = -\sum_x p(x) \log p(x)$$

It is also denoted by $H(p)$.

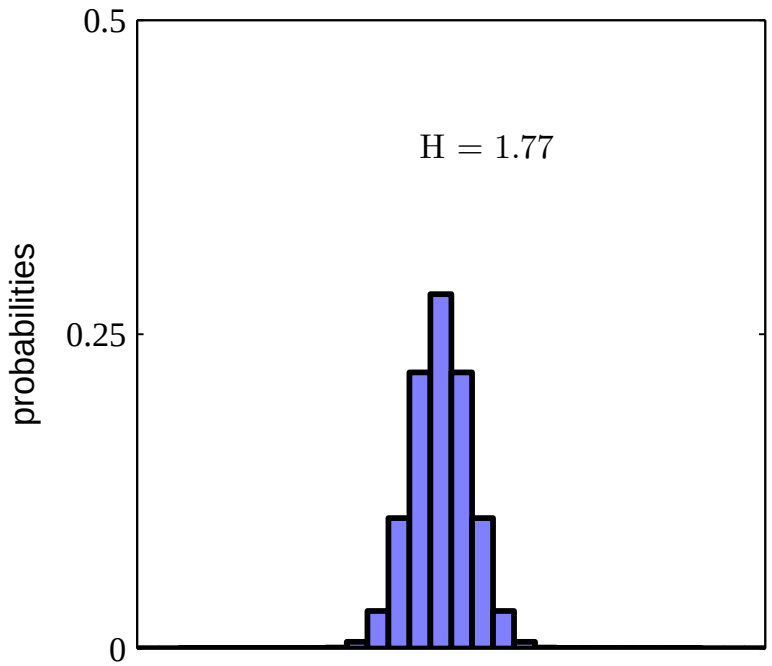
- It is the average of information when X takes a value.
- It is non-negative.
- It measures the disorder of X.

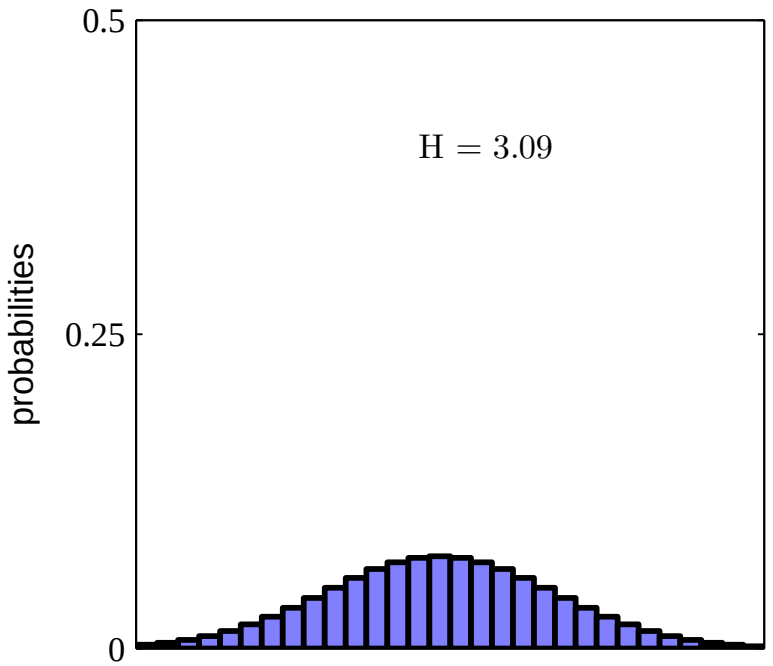
Example

The entropy of a Bernoulli random variable is

$$H[X] = -q \log_2 q - (1-q) \log_2 (1-q)$$

where $q = P(X = 0)$.


0.5
0.25
0
probabilities
H = 1.77


0.5
0.25
0
probabilities
H = 3.09

Relative Entropy or KL Divergence

Definition. Let X be a discrete random variable. The **relative entropy** or **KL divergence** (or KL distance) between distribution $p(x)$ and distribution $q(x)$ is

$$\mathsf{KL}(p\|q) = \sum_x p(x) \log \frac{p(x)}{q(x)}$$

For a continuous random variable X

$$\mathsf{KL}(p\|q) = \int p(x) \log \frac{p(x)}{q(x)} dx$$

It can be shown that

$$\mathsf{KL}(p\|q) \geq 0$$

It is important to note

$$\mathsf{KL}(p\|q) \neq \mathsf{KL}(q\|p)$$

Let $\mathbf{x}$ be a random vector with distribution $p(\boldsymbol{x})$. Suppose $p(\boldsymbol{x})$ is unknown, yet we have data points $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$ drawn from $p(\boldsymbol{x})$. We can approximate $p(\boldsymbol{x})$ via a parametric distribution $q(\boldsymbol{x}|\boldsymbol{\theta})$ with

$$q(\boldsymbol{x}|\boldsymbol{\theta}^*) = \arg\min_{\boldsymbol{\theta}} \mathsf{KL}(p(\boldsymbol{x})\|q(\boldsymbol{x}|\boldsymbol{\theta}))$$

Note

$$\begin{aligned}
\mathsf{KL}(p(\boldsymbol{x})\|q(\boldsymbol{x}|\boldsymbol{\theta})) &= \mathbb{E}\left\{\log \frac{p(\mathbf{x})}{q(\mathbf{x}|\boldsymbol{\theta})}\right\} \\
&= \mathbb{E}\left\{\log p(\mathbf{x})\right\} - \mathbb{E}\left\{\log q(\mathbf{x}|\boldsymbol{\theta})\right\} \\
&\approx -H(p) - \frac{1}{N}\sum_{n=1}^{N}\{\log q(\boldsymbol{x}_n|\boldsymbol{\theta})\}
\end{aligned}$$

So minimizing $\mathsf{KL}(p\|q)$ is equivalent to maximizing the data likelihood under q.

Cross Entropy

Definition. Let X be a discrete random variable. The **cross entropy** between distribution $p(x)$ and distribution $q(x)$ is

$$E(p,q) = -\sum_x p(x)\log q(x)$$

For a continuous random variable X

$$E(p,q) = -\int p(x)\log q(x)dx$$

The KL divergence between distribution $p(x)$ and distribution $q(x)$ can be written as

$$\mathrm{KL}(p\|q) = \sum_x p(x)\log\frac{p(x)}{q(x)} = E(p,q) - H(p)$$

Hence minimizing $\mathrm{KL}(p\|q)$ over q is equivalent to minimizing $E(p,q)$.

In a classification model, we use a parametric function $\boldsymbol{y}(\boldsymbol{x}, \boldsymbol{w})$ to approximate the class posterior probabilities. For a data point $(\boldsymbol{x}, \boldsymbol{t})$ from class $\mathcal{C}_j$, the cross entropy between $\boldsymbol{t}$ and $\boldsymbol{y}(\boldsymbol{x}, \boldsymbol{w})$ is

$$\begin{aligned} E(\boldsymbol{t}, \boldsymbol{y}(\boldsymbol{x}, \boldsymbol{w})) &= -\sum_k t_k \log y_k(\boldsymbol{x}, \boldsymbol{w}) \\ &= -\left(\sum_{k \neq j} \delta_{kj} \log y_k(\boldsymbol{x}, \boldsymbol{w}) + \delta_{jj} \log y_j(\boldsymbol{x}, \boldsymbol{w}) \right) \\ &= -\log\ y_j(\boldsymbol{x}, \boldsymbol{w}) \\ &\approx -\log\ p(\mathcal{C}_j | \boldsymbol{x}) \end{aligned}$$

This is the negative log likelihood.

Definition. Let X and Y be discrete random variables with joint distribution $p(\boldsymbol{x}, \boldsymbol{y})$. The **conditional entropy** of Y given X is

$$\begin{aligned} H[Y|X] &= -\sum_x p(x) \sum_y p(y|x) \log p(y|x) \\ &= -\sum_x \sum_y p(x,y) \log p(y|x) \end{aligned}$$

It can be shown that

$$H[X,Y] = H[X] + H[Y|X]$$

Definition. Let X and Y be discrete random variables with joint distribution $p(\boldsymbol{x}, \boldsymbol{y})$. The **mutual information** between X and Y is

$$I[X, Y] = \mathsf{KL}(p(x, y) \| p(x)p(y))$$

It can be shown that

$$\begin{aligned} I[X, Y] &= H[X] - H[X|Y] \\ &= H[Y] - H[Y|X] \end{aligned}$$