Bayesian decision theory

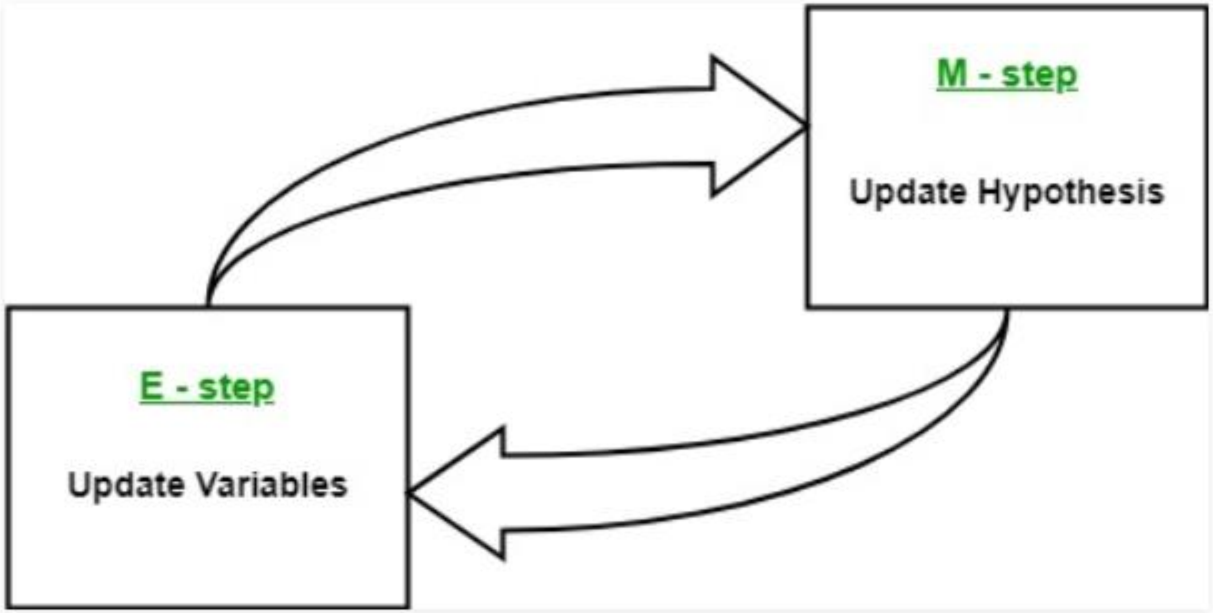
INT354

Expectation-Maximization Algorithm

- In the real-world applications of machine learning, it is very common that there are many relevant features available for learning but only a small subset of them are observable.
- ***Expectation-Maximization algorithm*** can be used for the latent variables (variables that are not directly observable and are actually inferred from the values of the other observed variables) too in order to predict their values with the condition that the general form of probability distribution governing those latent variables is known to us.
- This algorithm is actually at the base of many unsupervised clustering algorithms in the field of machine learning.
- It is used to find the *local maximum likelihood parameters* of a statistical model in the cases where latent variables are involved and the data is missing or incomplete.

Algorithm

- Given a set of incomplete data, consider a set of starting parameters.
- **Expectation step (E – step):** Using the observed available data of the dataset, estimate (guess) the values of the missing data.
- **Maximization step (M – step):** Complete data generated after the expectation (E) step is used in order to update the parameters.
- Repeat step 2 and step 3 until convergence.



- The essence of Expectation-Maximization algorithm is to use the available observed data of the dataset to estimate the missing data and then using that data to update the values of the parameters.
- Let us understand the EM algorithm in detail.
  - Initially, a set of initial values of the parameters are considered. A set of incomplete observed data is given to the system with the assumption that the observed data comes from a specific model.
  - The next step is known as “Expectation” – step or *E-step*. In this step, we use the observed data in order to estimate or guess the values of the missing or incomplete data. It is basically used to update the variables.
  - The next step is known as “Maximization”-step or *M-step*. In this step, we use the complete data generated in the preceding “Expectation” – step in order to update the values of the parameters. It is basically used to update the hypothesis
  - Now, in the fourth step, it is checked whether the values are converging or not, if yes, then stop otherwise repeat *step-2* and *step-3* i.e. “Expectation” – step and “Maximization” – step until the convergence occurs.

Usage of EM algorithm

- It can be used to fill the missing data in a sample.
- It can be used as the basis of unsupervised learning of clusters.
- It can be used for the purpose of estimating the parameters of Hidden Markov Model (HMM).
- It can be used for discovering the values of latent variables.

Advantages of EM algorithm

- It is always guaranteed that likelihood will increase with each iteration.
- The E-step and M-step are often pretty easy for many problems in terms of implementation.
- Solutions to the M-steps often exist in the closed form.

Disadvantages of EM algorithm

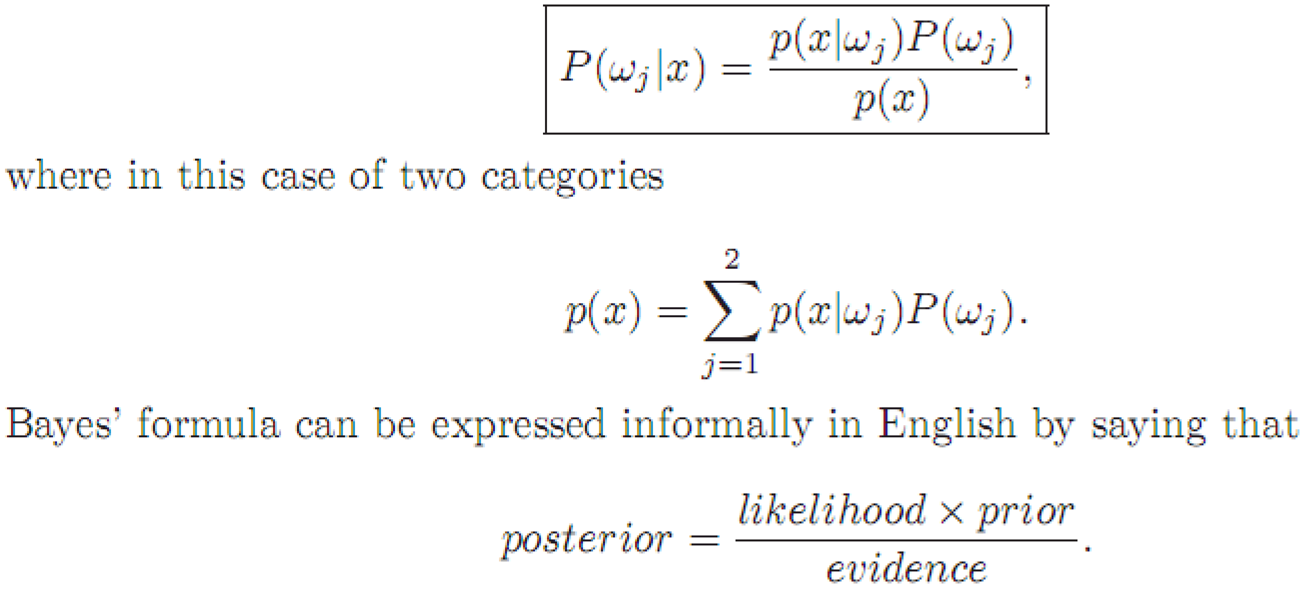
- It has slow convergence.
- It makes convergence to the local optima only.
- It requires both the probabilities, forward and backward (numerical optimization requires only forward probability).

Bayesian Decision theory

- Bayesian decision theory is a fundamental statistical approach to the problem of pattern classification.
- It makes the assumption that the decision problem is posed in ***probabilistic*** terms, and that all of the relevant probability values are known.
- let ω denote the **state of nature**, with $\omega = \omega 1$ for sea bass and $\omega = \omega 2$ for salmon. Because the state of nature is so unpredictable, we consider ***ω to be a variable*** that must be described probabilistically.
- we assume that there is some *a* **priori probability** (or simply *prior*) $P(\omega 1)$ that the next fish is sea bass, and prior some prior probability $P(\omega 2)$ that it is salmon.
  - If we assume there are no other types of fish relevant here, then $P(\omega 1)$ and $P(\omega 2)$ sum to one.

Decision Rule

- If a decision must be made with so little information, it seems logical to use the following
    - ***Decision rule*: decision Decide $\omega 1$ if $P(\omega 1) > P(\omega 2)$; otherwise decide $\omega 2$.**
- we consider *x* to be a continuous random variable whose distribution depends on the state of nature, and is expressed as $p(x/\omega 1)$.This is the ***class-conditional probability density function***.



$$\boxed{P(\omega_j|x) = \frac{p(x|\omega_j)P(\omega_j)}{p(x)},}$$

where in this case of two categories

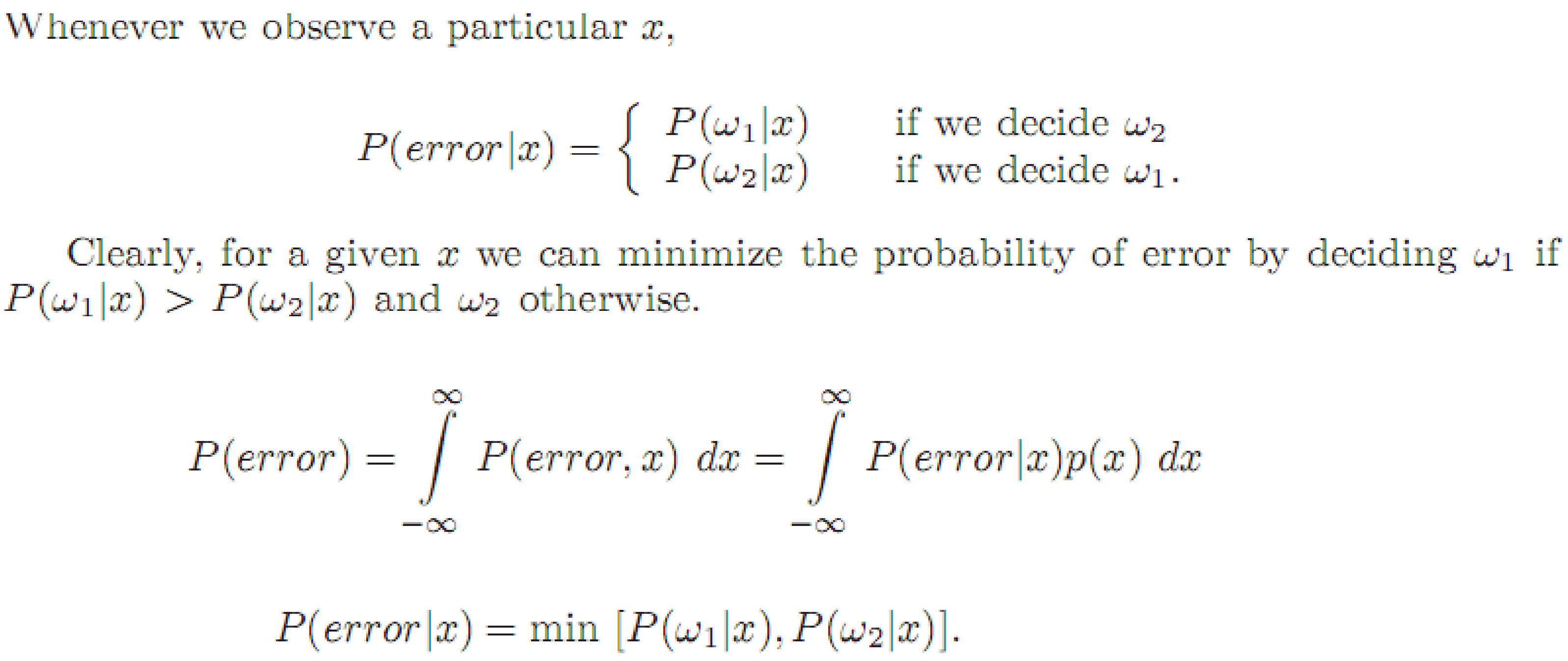
$$p(x) = \sum_{j=1}^{2} p(x|\omega_j)P(\omega_j).$$

Bayes' formula can be expressed informally in English by saying that

$$posterior = \frac{likelihood \times prior}{evidence}.$$

We call $p(x|\omega j)$ the *likelihood* of ωj with respect to x

- Thus we have justified the following *Bayes' decision rule* for minimizing the probability of error:
  - ***Decide ω1 if P(ω1|x) > P(ω2|x); otherwise decide ω2.***
  - **Decide *ω*1 if *p*(*x*|*ω*1)*P*(*ω*1) > *p*(*x*|*ω*2)*P*(*ω*2); otherwise decide *ω2.***



Whenever we observe a particular x,

$$P(error|x) = \begin{cases} P(\omega_1|x) & \text{if we decide } \omega_2 \\ P(\omega_2|x) & \text{if we decide } \omega_1 . \end{cases}$$

Clearly, for a given x we can minimize the probability of error by deciding ω_1 if $P(\omega_1|x) > P(\omega_2|x)$ and ω_2 otherwise.

$$P(error) = \int_{-\infty}^{\infty} P(error, x)\,dx = \int_{-\infty}^{\infty} P(error|x)p(x)\,dx$$

$$P(error|x) = \min\,[P(\omega_1|x), P(\omega_2|x)].$$

Bayesian Decision Theory – Continuous Features

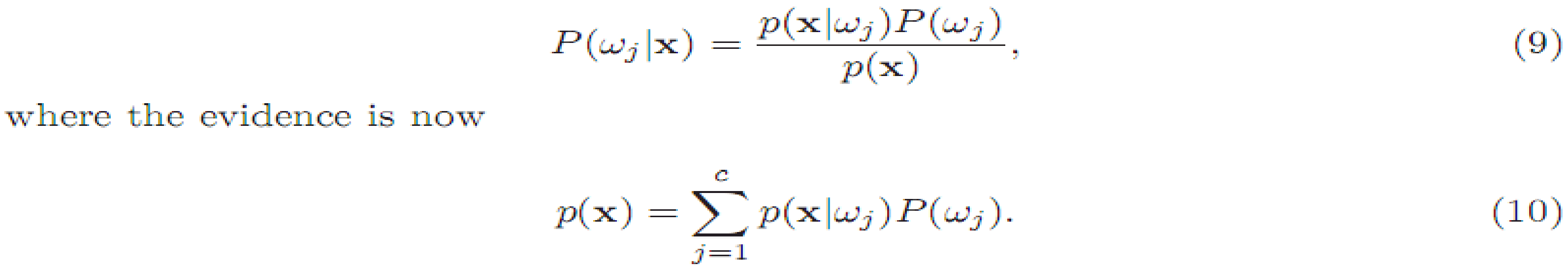
We now formalize the ideas just considered, and generalize them in four ways:

- by allowing the use of more than one feature
- by allowing more than two states of nature
- by allowing actions other than merely deciding the state of nature
- by introducing a loss function more general than the probability of error.

Allowing the use of more than one feature merely requires replacing the scalar *x* by the ***feature vector*** **x**, where **x** is in a *d*-dimensional Euclidean space **R***d*, called the ***feature space***.

the ***loss function*** states exactly how costly each action is, and is used to convert a probability determination into a decision.
Cost function let us treat situations in which some kinds of classification mistakes are more costly than others, although we often discuss the simplest case, where all errors are equally costly.

- Let *ω*1*, ..., ωc* be the finite set of *c* states of nature ("categories")
- *α*1*, ..., αa* be the finite set of *a* possible actions.
- The loss function *λ*(*αi*/*ωj*) describes the loss incurred for taking action *αi* when the state of nature is *ωj* .
- Let the feature vector **x** be a *d*-component vector-valued random variable.
- let *p*(**x**/*ωj*) be the state conditional probability density function for **x** — the probability density function for **x** conditioned on *ωj* being the true state of nature.
- As before, *P*(*ωj*) describes the prior probability that nature is in state *ωj* .
- Then the posterior probability *P*(*ωj* /**x**) can be computed from *p*(**x**/*ωj*) by Bayes' formula:



$$P(\omega_j|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_j)P(\omega_j)}{p(\mathbf{x})}, \qquad (9)$$

where the evidence is now

$$p(\mathbf{x}) = \sum_{j=1}^{c} p(\mathbf{x}|\omega_j)P(\omega_j). \qquad (10)$$

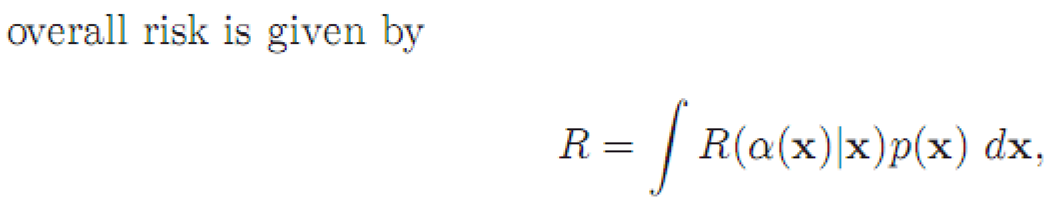
expected loss or Risk

- Suppose that we observe a particular **x** and that we contemplate taking action αi.
- If the true state of nature is ωj , by definition we will incur the loss $\lambda(\alpha i/\omega j\ )$.
- Since $P(\omega j\ /\mathbf{x})$ is the probability that the true state of nature is ωj , the expected loss associated with taking action αi is merely

$$R(\alpha_i|\mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j)P(\omega_j|\mathbf{x}).$$

In decision-theoretic terminology, an expected loss is called a *risk*, and $R(\alpha i\ /\mathbf{x})$ is called the *conditional risk*.

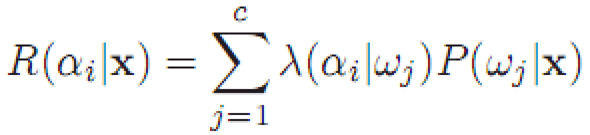
over All risk



overall risk is given by

$$R = \int R(\alpha(\mathbf{x})|\mathbf{x})p(\mathbf{x})\,d\mathbf{x},$$

- This justifies the following statement of the *Bayes decision rule*: To minimize the overall risk, compute the conditional risk



$$R(\alpha_i|\mathbf{x}) = \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j)P(\omega_j|\mathbf{x})$$

for *i* = 1,...,*a* and select the action *αi* for which *R*(*αi*/**x**) is minimum.

Two-Category Classification

- Here action *α*1 corresponds to deciding that the true state of nature is *ω*1, and action *α*2 corresponds to deciding that it is *ω*2.
- For notational simplicity, let *λij* = *λ*(*αi*/*ωj*) be the loss incurred for deciding *ωi* when the true state of nature is *ωj* .
- If we write out the conditional risk

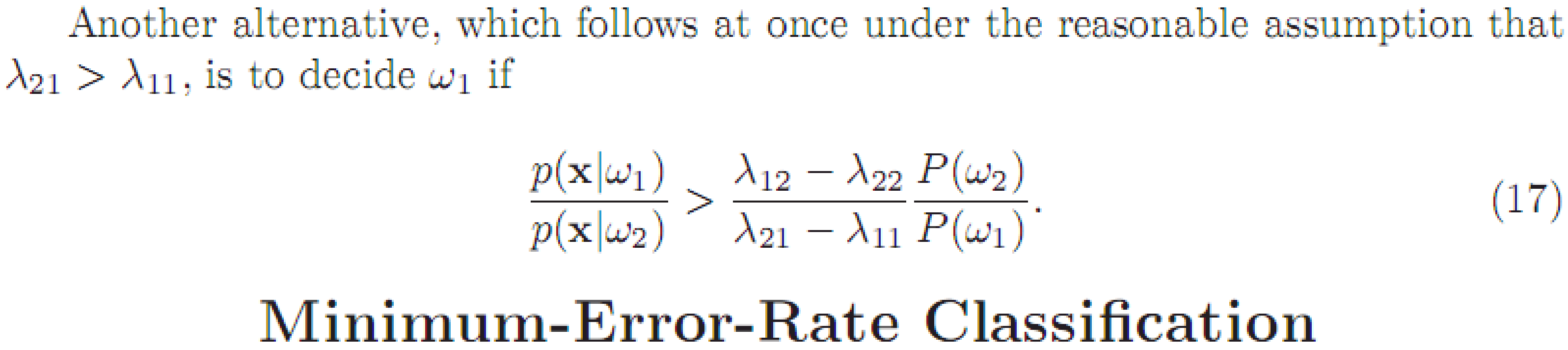
$$\begin{aligned} R(\alpha_1|\mathbf{x}) &= \lambda_{11}P(\omega_1|\mathbf{x}) + \lambda_{12}P(\omega_2|\mathbf{x}) \\ R(\alpha_2|\mathbf{x}) &= \lambda_{21}P(\omega_1|\mathbf{x}) + \lambda_{22}P(\omega_2|\mathbf{x}). \end{aligned}$$

The fundamental rule is to decide ***ω*1 if *R*(*α*1/x) < *R*(*α*2/x)**.
In terms of the posterior probabilities, we decide *ω*1 if

$$(\lambda_{21} - \lambda_{11})P(\omega_1|\mathbf{x}) > (\lambda_{12} - \lambda_{22})P(\omega_2|\mathbf{x}).$$

$$(\lambda_{21} - \lambda_{11})p(\mathbf{x}|\omega_1)P(\omega_1) > (\lambda_{12} - \lambda_{22})p(\mathbf{x}|\omega_2)P(\omega_2),$$

Two-Category Classification



Another alternative, which follows at once under the reasonable assumption that $\lambda_{21} > \lambda_{11}$, is to decide ω_1 if

$$\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} > \frac{\lambda_{12} - \lambda_{22}}{\lambda_{21} - \lambda_{11}} \frac{P(\omega_2)}{P(\omega_1)}. \qquad (17)$$

Minimum-Error-Rate Classification

Each state of nature is usually associated with a different one of the *c* classes, and the action *αi* is usually interpreted as the decision that the true state of nature is *ωi*.

If action *αi* is taken and the true state of nature is *ωj* , then the decision is correct if *i* = *j*, and in error if *i* ≠ *j*.

The loss function of interest for this case is hence the so-called *symmetrical* or *zero-one* loss function,

$$\lambda(\alpha_i|\omega_j) = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases} \qquad i, j = 1, ..., c.$$

since the conditional risk is

$$\begin{aligned} R(\alpha_i|\mathbf{x}) &= \sum_{j=1}^{c} \lambda(\alpha_i|\omega_j) P(\omega_j|\mathbf{x}) \\ &= \sum_{j \neq i} P(\omega_j|\mathbf{x}) \\ &= 1 - P(\omega_i|\mathbf{x}) \end{aligned}$$

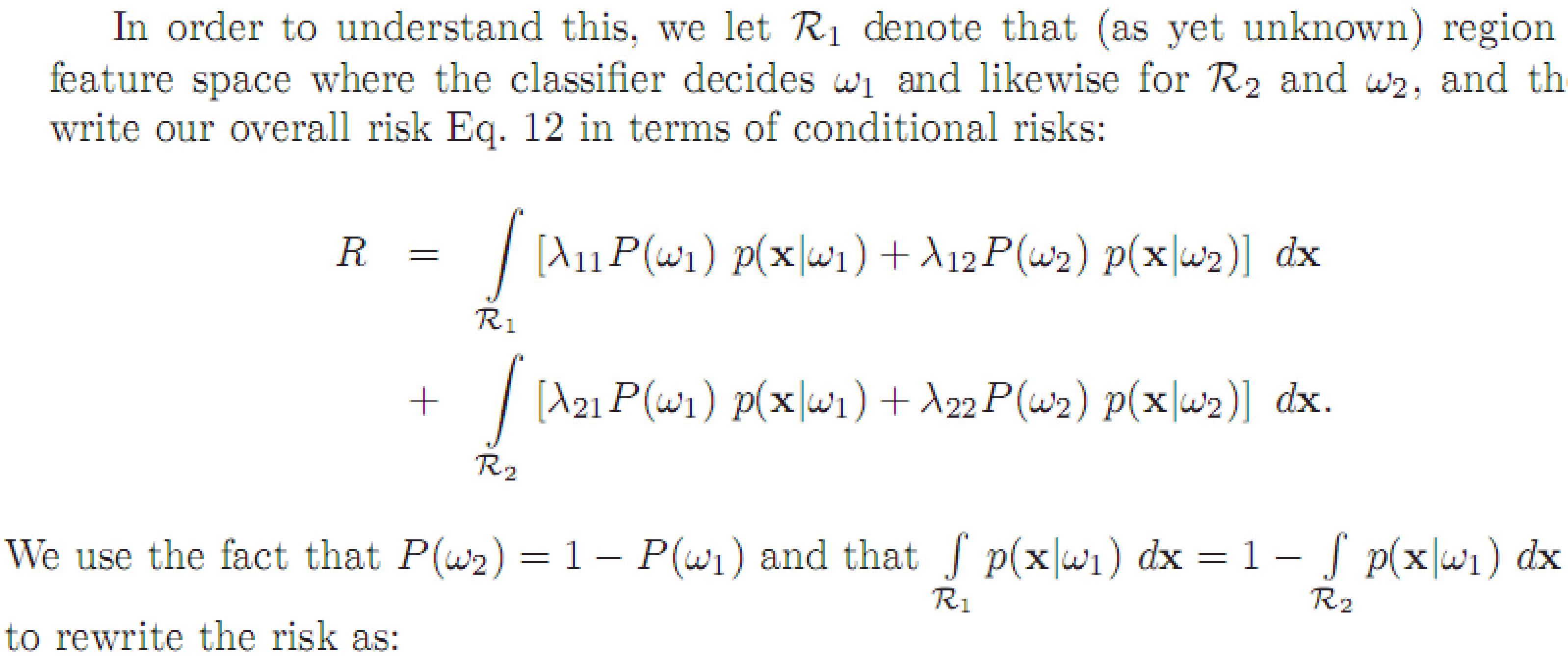
P(*ωi*|**x**) is the conditional probability that action *αi* is correct.

In other words, for *minimum error rate*:

Decide ωi if P(ωi|x) > P(ωj |x) for all j ≠ i.

Minimax Criterion

- minimize the maximum possible overall risk.



In order to understand this, we let $\mathcal{R}_1$ denote that (as yet unknown) region in feature space where the classifier decides ω_1 and likewise for $\mathcal{R}_2$ and ω_2, and then write our overall risk Eq. 12 in terms of conditional risks:

$$\begin{aligned} R &= \int\limits_{\mathcal{R}_1} [\lambda_{11} P(\omega_1)\ p(\mathbf{x}|\omega_1) + \lambda_{12} P(\omega_2)\ p(\mathbf{x}|\omega_2)]\ d\mathbf{x} \\ &+ \int\limits_{\mathcal{R}_2} [\lambda_{21} P(\omega_1)\ p(\mathbf{x}|\omega_1) + \lambda_{22} P(\omega_2)\ p(\mathbf{x}|\omega_2)]\ d\mathbf{x}. \end{aligned}$$

We use the fact that $P(\omega_2) = 1 - P(\omega_1)$ and that $\int\limits_{\mathcal{R}_1} p(\mathbf{x}|\omega_1)\ d\mathbf{x} = 1 - \int\limits_{\mathcal{R}_2} p(\mathbf{x}|\omega_1)\ d\mathbf{x}$ to rewrite the risk as:

Minimax Criterion

$$R(P(\omega_1)) = \overbrace{\lambda_{22} + (\lambda_{12} - \lambda_{22}) \int_{\mathcal{R}_1} p(\mathbf{x}|\omega_2)\, d\mathbf{x}}^{= R_{mm},\ \text{minimax risk}} \tag{22}$$

$$+ \underbrace{P(\omega_1) \left[(\lambda_{11} - \lambda_{22}) - (\lambda_{21} - \lambda_{11}) \int_{\mathcal{R}_2} p(\mathbf{x}|\omega_1)\, d\mathbf{x} - (\lambda_{12} - \lambda_{22}) \int_{\mathcal{R}_1} p(\mathbf{x}|\omega_2)\, d\mathbf{x} \right]}_{= 0 \text{ for minimax solution}} .$$

minimax risk, R_{mm},

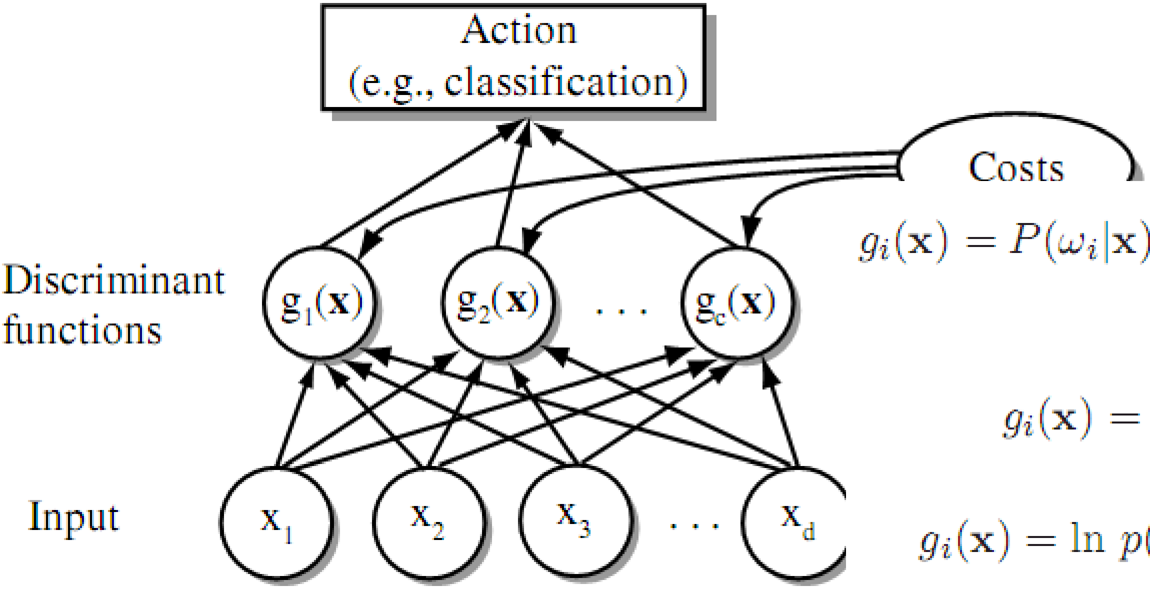
$$\begin{aligned} R_{mm} &= \lambda_{22} + (\lambda_{12} - \lambda_{22}) \int_{\mathcal{R}_1} p(\mathbf{x}|\omega_2)\, d\mathbf{x} \\ &= \lambda_{11} + (\lambda_{21} - \lambda_{11}) \int_{\mathcal{R}_2} p(\mathbf{x}|\omega_1)\, d\mathbf{x}. \end{aligned}$$

Classifiers, Discriminant Functions and Decision Surfaces

The Multi-Category Case

There are many different ways to represent pattern classifiers. One of the most useful is in terms of a set of *discriminant functions* $g_i(\mathbf{x})$, $i = 1, ..., c$. The classifier is said to assign a feature vector $\mathbf{x}$ to class ω_i if

(24)



$$g_i(\mathbf{x}) = P(\omega_i|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_i)P(\omega_i)}{\sum_{j=1}^{c} p(\mathbf{x}|\omega_j)P(\omega_j)}$$

$$g_i(\mathbf{x}) = p(\mathbf{x}|\omega_i)P(\omega_i)$$

$$g_i(\mathbf{x}) = \ln\, p(\mathbf{x}|\omega_i) + \ln\, P(\omega_i),$$

The Two-Category Case

- Indeed, a classifier that places a pattern in one of only two categories has a special name — a *dichotomizer*. Instead of using two discriminant functions *g*1 and *g*2 and *assigning* ***x*** *to ω1 if g1 > g2*, it is more common to define a single discriminant function

$$g(\mathbf{x}) \equiv g_1(\mathbf{x}) - g_2(\mathbf{x}), \tag{28}$$

and to use the following decision rule: Decide ω_1 if $g(\mathbf{x}) > 0$; otherwise decide ω_2.

$$g(\mathbf{x}) = P(\omega_1|\mathbf{x}) - P(\omega_2|\mathbf{x})$$

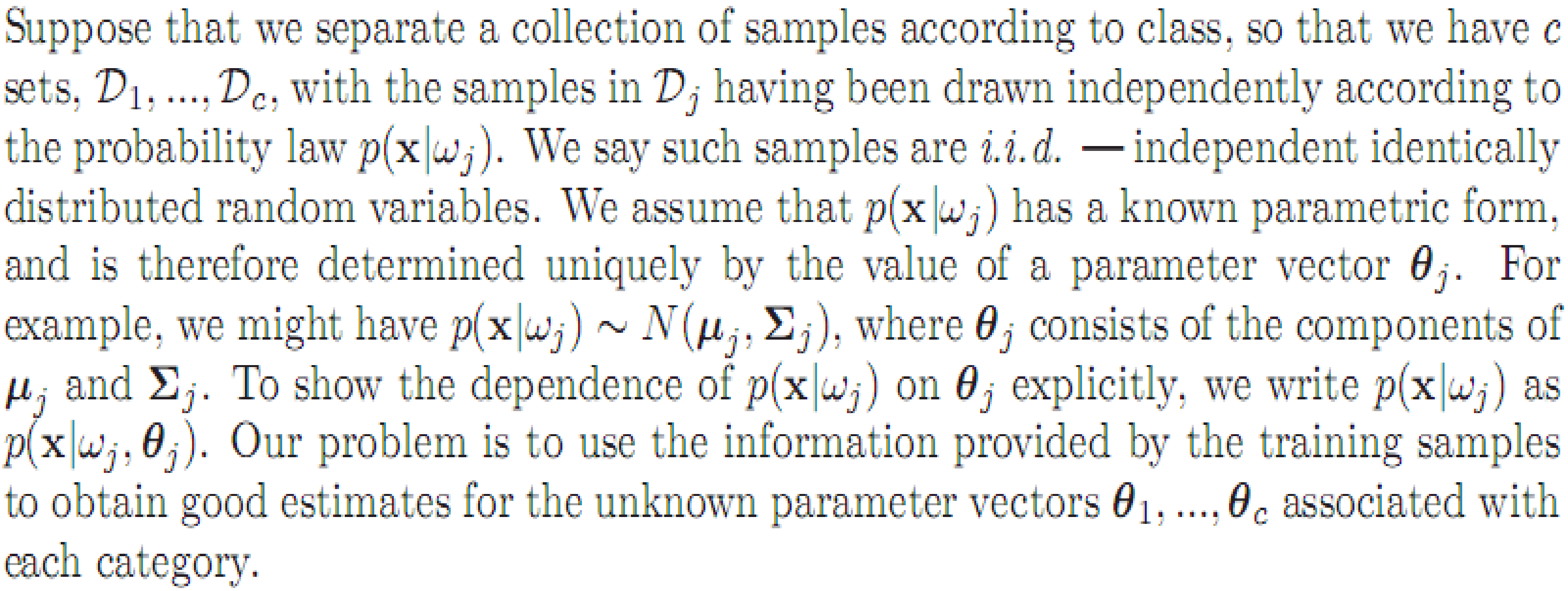
$$g(\mathbf{x}) = \ln\frac{p(\mathbf{x}|\omega_1)}{p(\mathbf{x}|\omega_2)} + \ln\frac{P(\omega_1)}{P(\omega_2)}.$$

Maximum Likelihood Estimation

1. Have good convergence properties as number of training sample increases.

2. Often can be simpler than alternate methods, such as Bayesian techniques.

- **The General Principle**



Suppose that we separate a collection of samples according to class, so that we have c sets, $\mathcal{D}_1, ..., \mathcal{D}_c$, with the samples in $\mathcal{D}_j$ having been drawn independently according to the probability law $p(\mathbf{x}|\omega_j)$. We say such samples are *i.i.d.* — independent identically distributed random variables. We assume that $p(\mathbf{x}|\omega_j)$ has a known parametric form, and is therefore determined uniquely by the value of a parameter vector $\boldsymbol{\theta}_j$. For example, we might have $p(\mathbf{x}|\omega_j) \sim N(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)$, where $\boldsymbol{\theta}_j$ consists of the components of $\boldsymbol{\mu}_j$ and $\boldsymbol{\Sigma}_j$. To show the dependence of $p(\mathbf{x}|\omega_j)$ on $\boldsymbol{\theta}_j$ explicitly, we write $p(\mathbf{x}|\omega_j)$ as $p(\mathbf{x}|\omega_j, \boldsymbol{\theta}_j)$. Our problem is to use the information provided by the training samples to obtain good estimates for the unknown parameter vectors $\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_c$ associated with each category.

Suppose that $\mathcal{D}$ contains n samples, $\mathbf{x}_1, ..., \mathbf{x}_n$. Then, since the samples were drawn independently, we have

$$p(\mathcal{D}|\boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k|\boldsymbol{\theta}).$$

$p(\mathcal{D}|\boldsymbol{\theta})$ is called the *likelihood* of $\boldsymbol{\theta}$ with respect to the set of samples. The *maximum likelihood estimate* of $\boldsymbol{\theta}$ is, by definition, the value $\hat{\boldsymbol{\theta}}$ that maximizes $p(\mathcal{D}|\boldsymbol{\theta})$.

If the number of parameters to be set is p, then we let $\boldsymbol{\theta}$ denote

the p-component vector $\boldsymbol{\theta} = (\theta_1, ..., \theta_p)^t$, and $\nabla_{\boldsymbol{\theta}}$ be the gradient operator

$$\nabla_{\boldsymbol{\theta}} \equiv \begin{bmatrix} \frac{\partial}{\partial \theta_1} \\ \vdots \\ \frac{\partial}{\partial \theta_p} \end{bmatrix}.$$

We define $l(\boldsymbol{\theta})$ as the *log-likelihood* function*

$$l(\boldsymbol{\theta}) \equiv \ln p(\mathcal{D}|\boldsymbol{\theta}).$$

We can then write our solution formally as the argument $\boldsymbol{\theta}$ that maximizes the log-likelihood, i.e.,

$$\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} l(\boldsymbol{\theta}), \tag{4}$$

where the dependence on the data set $\mathcal{D}$ is implicit. Thus we have from Eq. 1

$$l(\boldsymbol{\theta}) = \sum_{k=1}^{n} \ln p(\mathbf{x}_k|\boldsymbol{\theta}) \tag{5}$$

and

$$\nabla_{\boldsymbol{\theta}} l = \sum_{k=1}^{n} \nabla_{\boldsymbol{\theta}} \ln p(\mathbf{x}_k|\boldsymbol{\theta}). \tag{6}$$

Thus, a set of necessary conditions for the maximum likelihood estimate for $\boldsymbol{\theta}$ can be obtained from the set of p equations

$$\boxed{\nabla_{\boldsymbol{\theta}} l = \mathbf{0}.} \tag{7}$$

We note in passing that a related class of estimators — *maximum a posteriori* or MAP estimators — find the value of $\boldsymbol{\theta}$ that maximizes $l(\boldsymbol{\theta})p(\boldsymbol{\theta})$. Thus a maximum likelihood estimator is a MAP estimator for the uniform or "flat" prior. As such, a MAP estimator finds the peak, or *mode* of a posterior density.

The Gaussian Case: Unknown $\boldsymbol{\mu}$

For simplicity, consider first the case where only the mean is unknown. Under this condition, we consider a sample point $\mathbf{x}_k$ and find

$$\ln p(\mathbf{x}_k|\boldsymbol{\mu}) = -\frac{1}{2}\ln\left[(2\pi)^d|\boldsymbol{\Sigma}|\right] - \frac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu})^t\boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu})$$

and

$$\nabla_{\boldsymbol{\theta}} \ln p(\mathbf{x}_k|\boldsymbol{\mu}) = \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}).$$

Identifying $\boldsymbol{\theta}$ with $\boldsymbol{\mu}$, we see from Eq. 9 that the maximum likelihood estimate for $\boldsymbol{\mu}$ must satisfy

$$\sum_{k=1}^{n} \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \hat{\boldsymbol{\mu}}) = \mathbf{0},$$

that is, each of the d components of $\hat{\boldsymbol{\mu}}$ must vanish. Multiplying by $\boldsymbol{\Sigma}$ and rearranging, we obtain

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k. \qquad (11)$$

The Gaussian Case: Unknown μ and Σ

In the more general (and more typical) multivariate normal case, neither the mean $\boldsymbol{\mu}$ nor the covariance matrix $\boldsymbol{\Sigma}$ is known. Thus, these unknown parameters constitute the components of the parameter vector $\boldsymbol{\theta}$. Consider first the univariate case with $\theta_1 = \mu$ and $\theta_2 = \sigma^2$. Here the log-likelihood of a single point is

$$\ln p(x_k|\boldsymbol{\theta}) = -\frac{1}{2} \ln 2\pi\theta_2 - \frac{1}{2\theta_2}(x_k - \theta_1)^2 \tag{12}$$

and its derivative is

$$\nabla_{\boldsymbol{\theta}} l = \nabla_{\boldsymbol{\theta}} \ln p(x_k|\boldsymbol{\theta}) = \begin{bmatrix} \frac{1}{\theta_2}(x_k - \theta_1) \\ -\frac{1}{2\theta_2} + \frac{(x_k-\theta_1)^2}{2\theta_2^2} \end{bmatrix}. \tag{13}$$

Applying Eq. 7 to the full log-likelihood leads to the conditions

$$\sum_{k=1}^{n} \frac{1}{\hat{\theta}_2}(x_k - \hat{\theta}_1) = 0 \tag{14}$$

$$-\sum_{k=1}^{n} \frac{1}{\hat{\theta}_2} + \sum_{k=1}^{n} \frac{(x_k - \hat{\theta}_1)^2}{\hat{\theta}_2^2} = 0,$$

where $\hat{\theta}_1$ and $\hat{\theta}_2$ are the maximum likelihood estimates for θ_1 and θ_2, respectively. By substituting $\hat{\mu} = \hat{\theta}_1$, $\hat{\sigma}^2 = \hat{\theta}_2$ and doing a little rearranging, we obtain the following maximum likelihood estimates for μ and σ^2:

$$\hat{\mu} = \frac{1}{n} \sum_{k=1}^{n} x_k \tag{16}$$

and

$$\hat{\sigma}^2 = \frac{1}{n} \sum_{k=1}^{n} (x_k - \hat{\mu})^2. \tag{17}$$

3.2.4 Bias

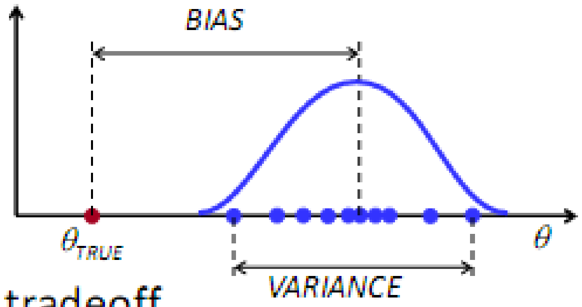
The maximum likelihood estimate for the variance σ^2 is *biased*; that is, the expected value over all data sets of size n of the sample variance is not equal to the true variance:*

$$\mathcal{E}\left[\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2\right] = \frac{n-1}{n}\sigma^2 \neq \sigma^2. \tag{20}$$

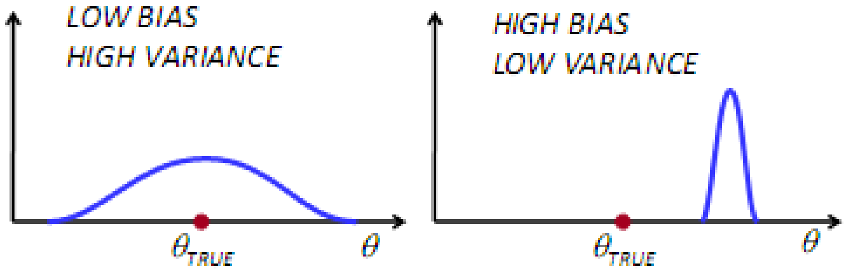
Bias and variance

How good are these estimates?

- Two measures of "goodness" are used for statistical estimates
- **BIAS**: how close is the estimate to the true value?
- **VARIANCE**: how much does it change for different datasets?



- The bias-variance tradeoff
  - In most cases, you can only decrease one of them at the expense of the other



Thank You !!!