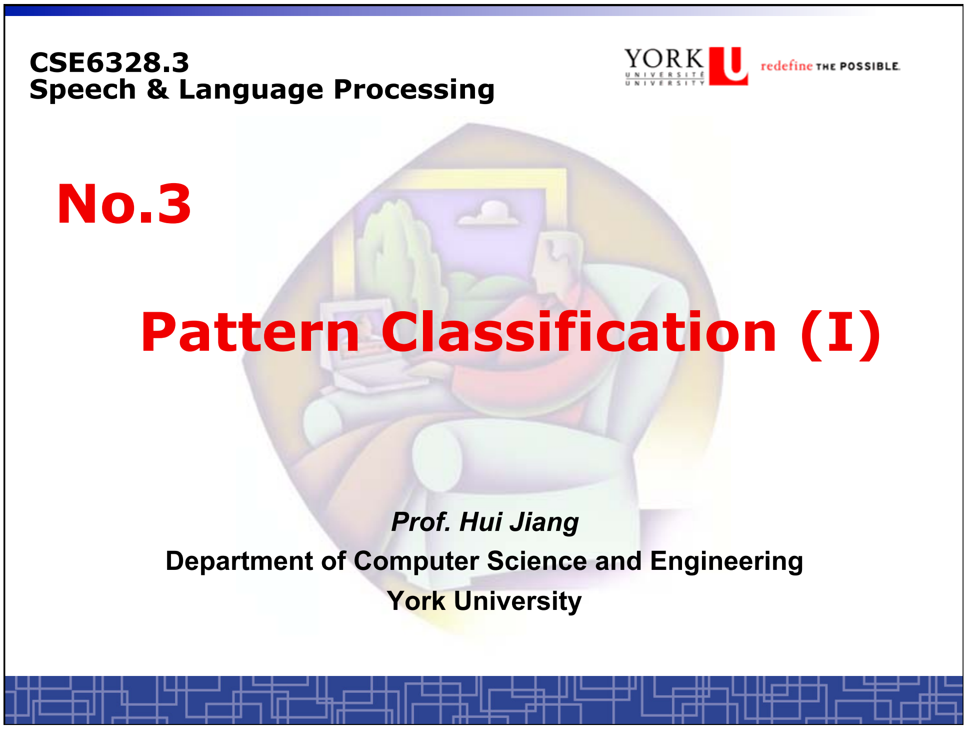


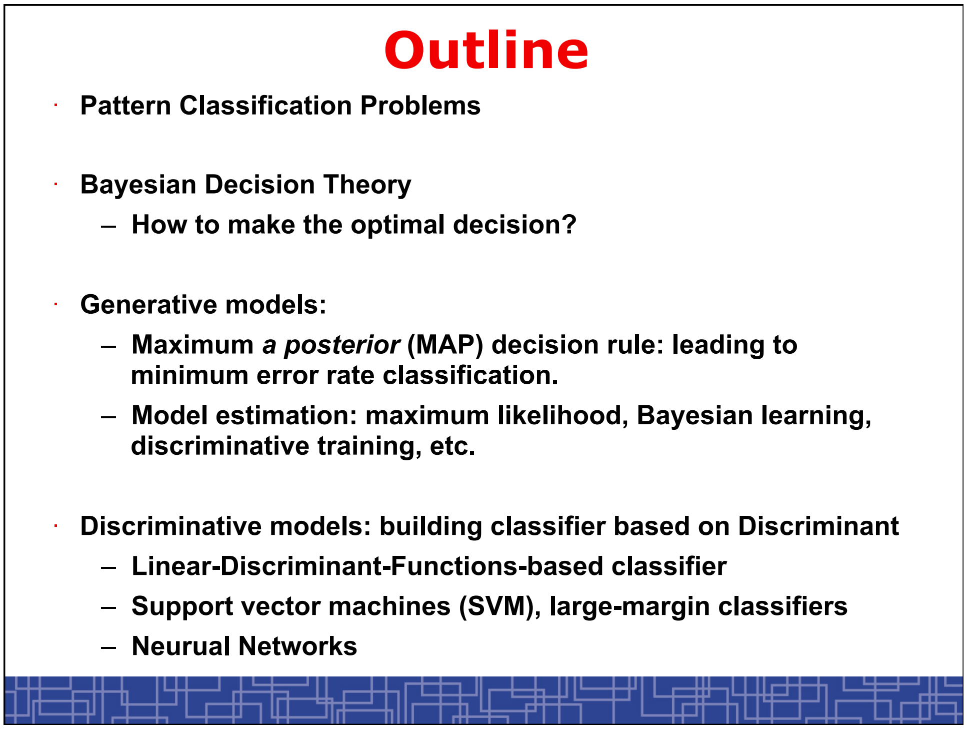
CSE6328.3
Speech & Language Processing

YORK U UNIVERSITÉ UNIVERSITY redefine THE POSSIBLE.

No.3

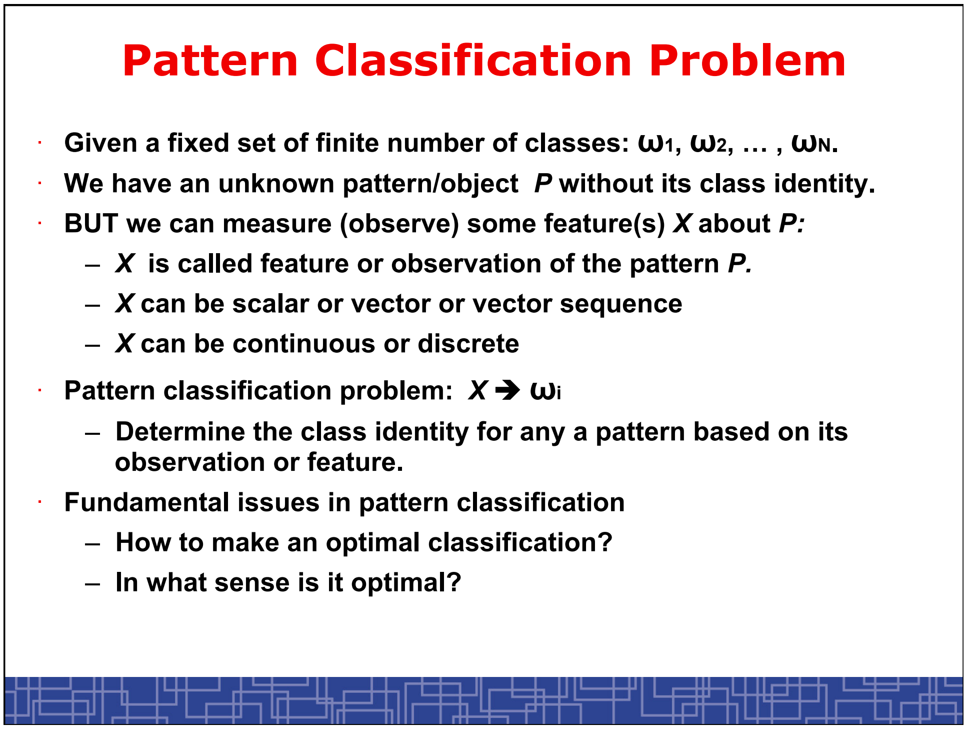
Pattern Classification (I)

Prof. Hui Jiang
Department of Computer Science and Engineering
York University



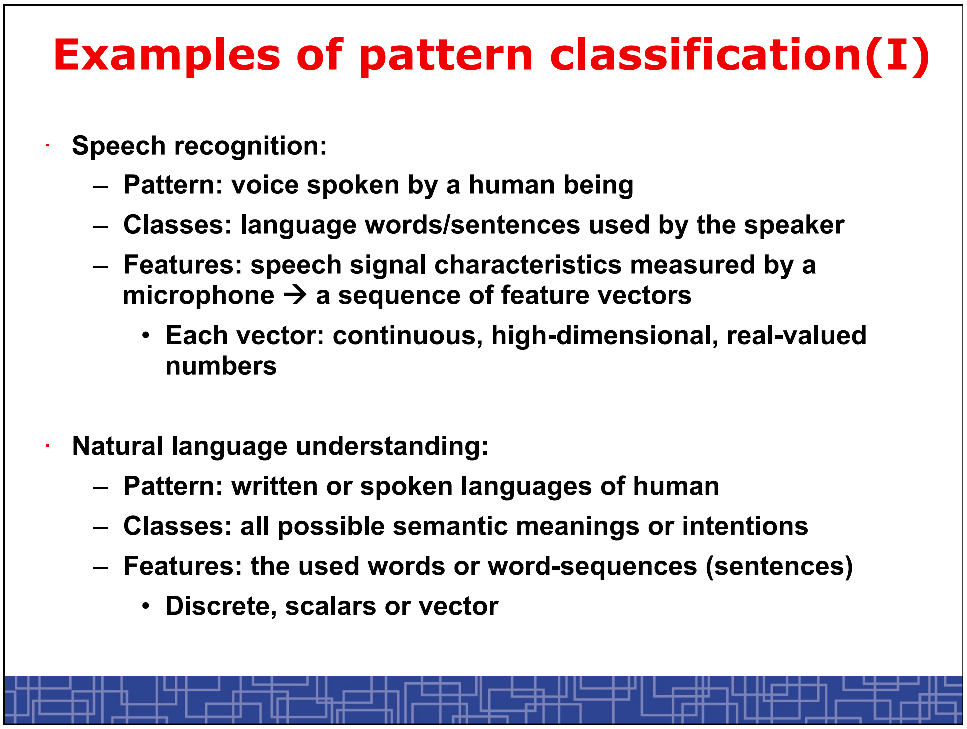
Outline

- Pattern Classification Problems
- Bayesian Decision Theory
  - How to make the optimal decision?
- Generative models:
  - Maximum *a posterior* (MAP) decision rule: leading to minimum error rate classification.
  - Model estimation: maximum likelihood, Bayesian learning, discriminative training, etc.
- Discriminative models: building classifier based on Discriminant
  - Linear-Discriminant-Functions-based classifier
  - Support vector machines (SVM), large-margin classifiers
  - Neurual Networks



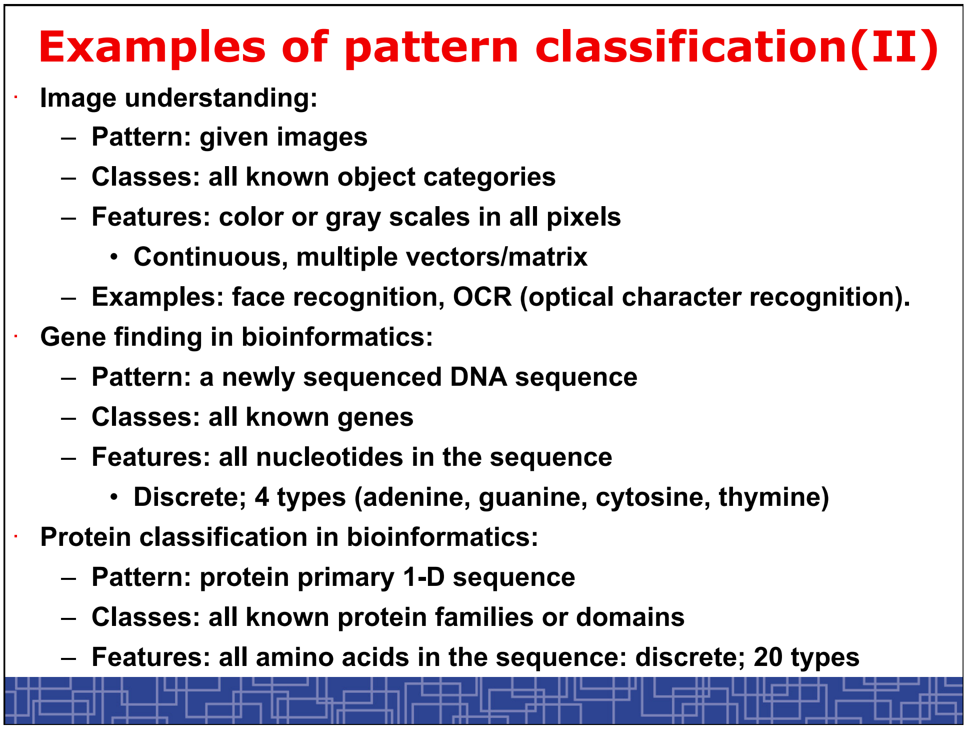
Pattern Classification Problem

- Given a fixed set of finite number of classes: $\omega_1, \omega_2, \ldots, \omega_N$.
- We have an unknown pattern/object *P* without its class identity.
- BUT we can measure (observe) some feature(s) *X* about *P:*
  - *X* is called feature or observation of the pattern *P.*
  - *X* can be scalar or vector or vector sequence
  - *X* can be continuous or discrete
- Pattern classification problem: $X \rightarrow \omega_i$
  - Determine the class identity for any a pattern based on its observation or feature.
- Fundamental issues in pattern classification
  - How to make an optimal classification?
  - In what sense is it optimal?



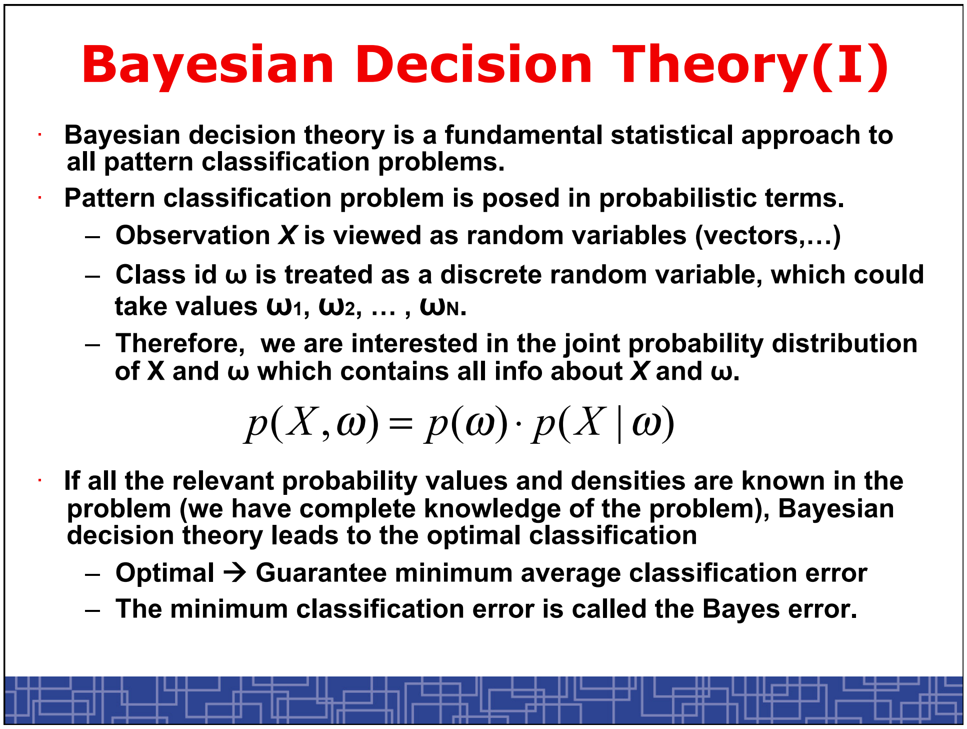
Examples of pattern classification(I)

- Speech recognition:
  - Pattern: voice spoken by a human being
  - Classes: language words/sentences used by the speaker
  - Features: speech signal characteristics measured by a microphone → a sequence of feature vectors
    - Each vector: continuous, high-dimensional, real-valued numbers
- Natural language understanding:
  - Pattern: written or spoken languages of human
  - Classes: all possible semantic meanings or intentions
  - Features: the used words or word-sequences (sentences)
    - Discrete, scalars or vector



Examples of pattern classification(II)

- Image understanding:
  - Pattern: given images
  - Classes: all known object categories
  - Features: color or gray scales in all pixels
    - Continuous, multiple vectors/matrix
  - Examples: face recognition, OCR (optical character recognition).
- Gene finding in bioinformatics:
  - Pattern: a newly sequenced DNA sequence
  - Classes: all known genes
  - Features: all nucleotides in the sequence
    - Discrete; 4 types (adenine, guanine, cytosine, thymine)
- Protein classification in bioinformatics:
  - Pattern: protein primary 1-D sequence
  - Classes: all known protein families or domains
  - Features: all amino acids in the sequence: discrete; 20 types

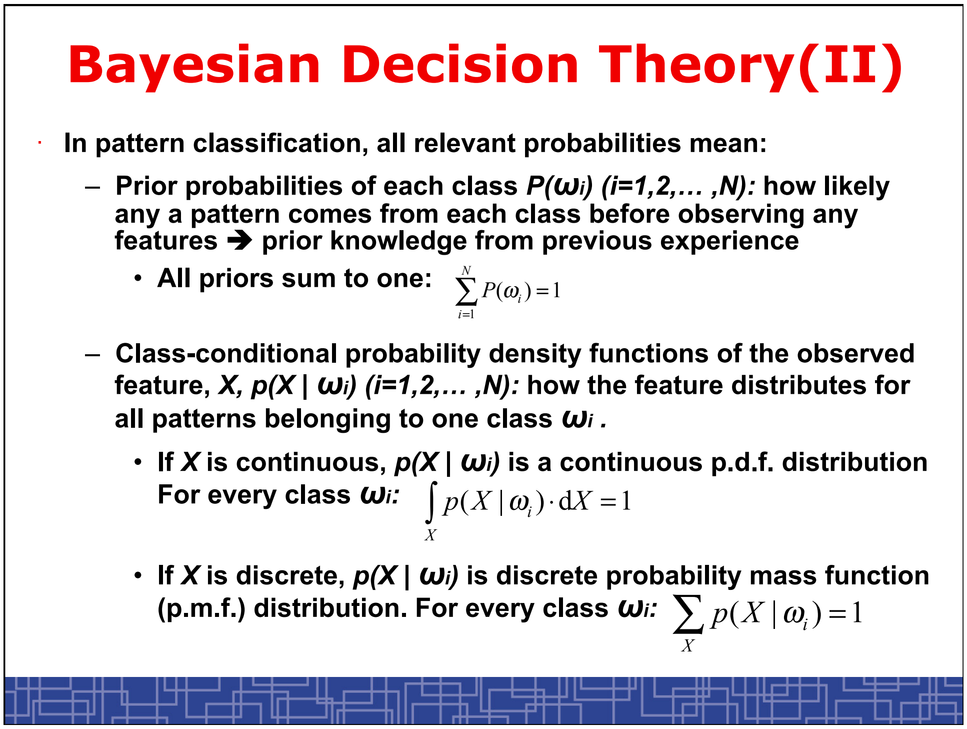


Bayesian Decision Theory(I)

- Bayesian decision theory is a fundamental statistical approach to all pattern classification problems.
- Pattern classification problem is posed in probabilistic terms.
  - Observation *X* is viewed as random variables (vectors,…)
  - Class id ω is treated as a discrete random variable, which could take values $\omega_1, \omega_2, \ldots, \omega_N$.
  - Therefore, we are interested in the joint probability distribution of X and ω which contains all info about *X* and ω.

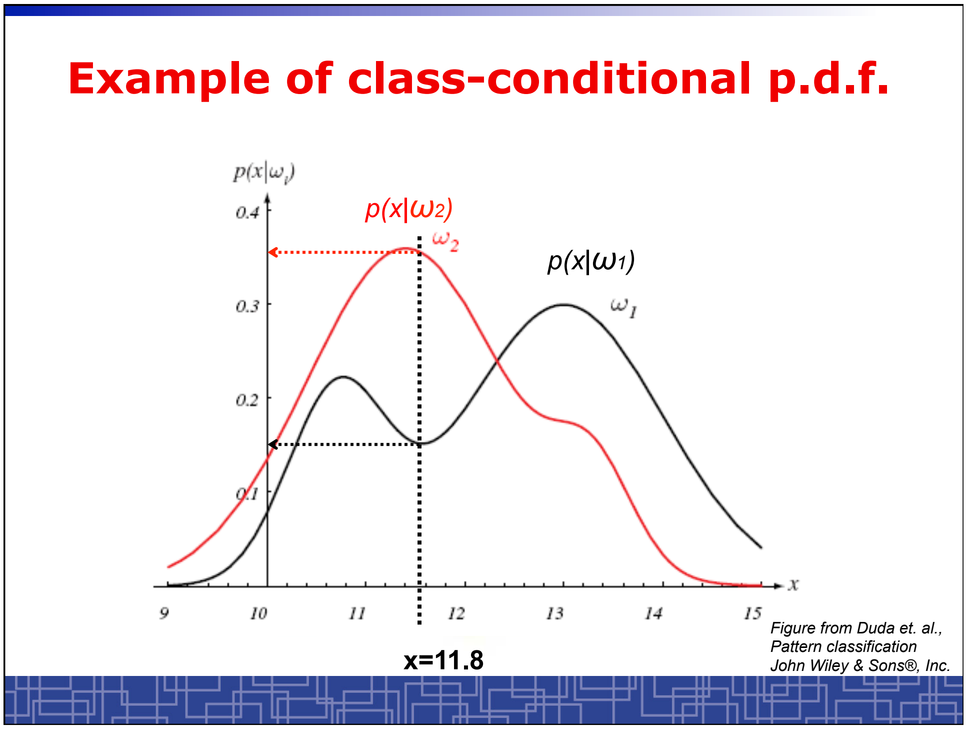
$$p(X, \omega) = p(\omega) \cdot p(X \mid \omega)$$

- If all the relevant probability values and densities are known in the problem (we have complete knowledge of the problem), Bayesian decision theory leads to the optimal classification
  - Optimal → Guarantee minimum average classification error
  - The minimum classification error is called the Bayes error.



Bayesian Decision Theory(II)

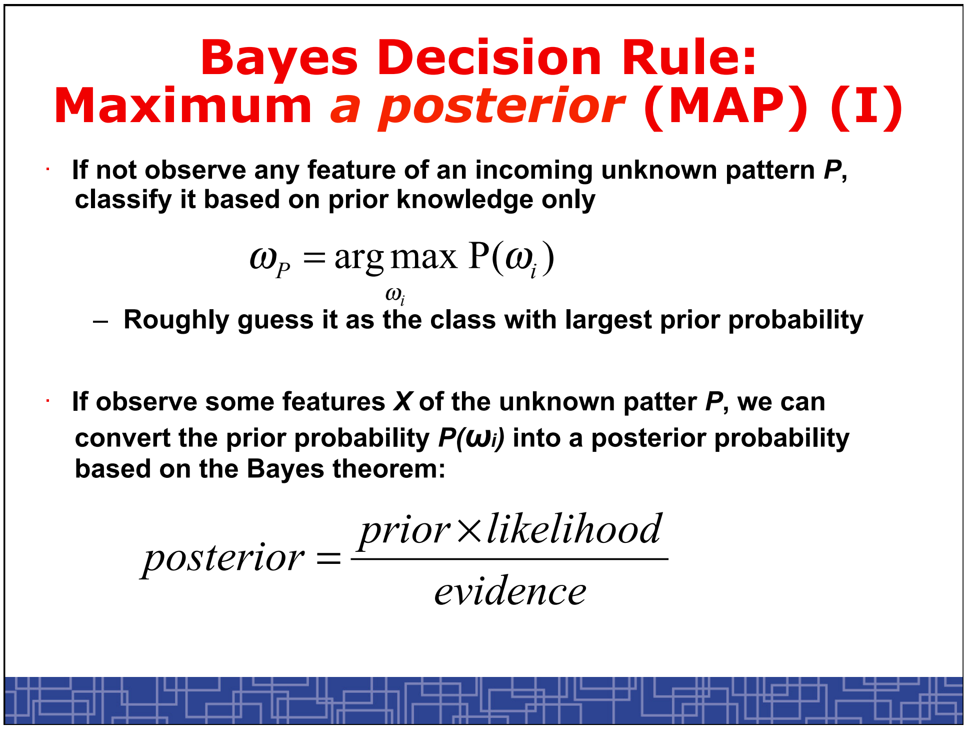
- In pattern classification, all relevant probabilities mean:
  - Prior probabilities of each class *P(ω_i) (i=1,2,… ,N):* how likely any a pattern comes from each class before observing any features ➔ prior knowledge from previous experience
    - All priors sum to one: $\sum_{i=1}^{N} P(\omega_i) = 1$
  - Class-conditional probability density functions of the observed feature, *X, p(X | ω_i) (i=1,2,… ,N):* how the feature distributes for all patterns belonging to one class *ω_i .*
    - If *X* is continuous, *p(X | ω_i)* is a continuous p.d.f. distribution For every class *ω_i:* $\int_X p(X \mid \omega_i) \cdot \mathrm{d}X = 1$
    - If *X* is discrete, *p(X | ω_i)* is discrete probability mass function (p.m.f.) distribution. For every class *ω_i:* $\sum_X p(X \mid \omega_i) = 1$



Example of class-conditional p.d.f.

x=11.8

Figure from Duda et. al., Pattern classification John Wiley & Sons®, Inc.



Bayes Decision Rule: Maximum *a posterior* (MAP) (I)

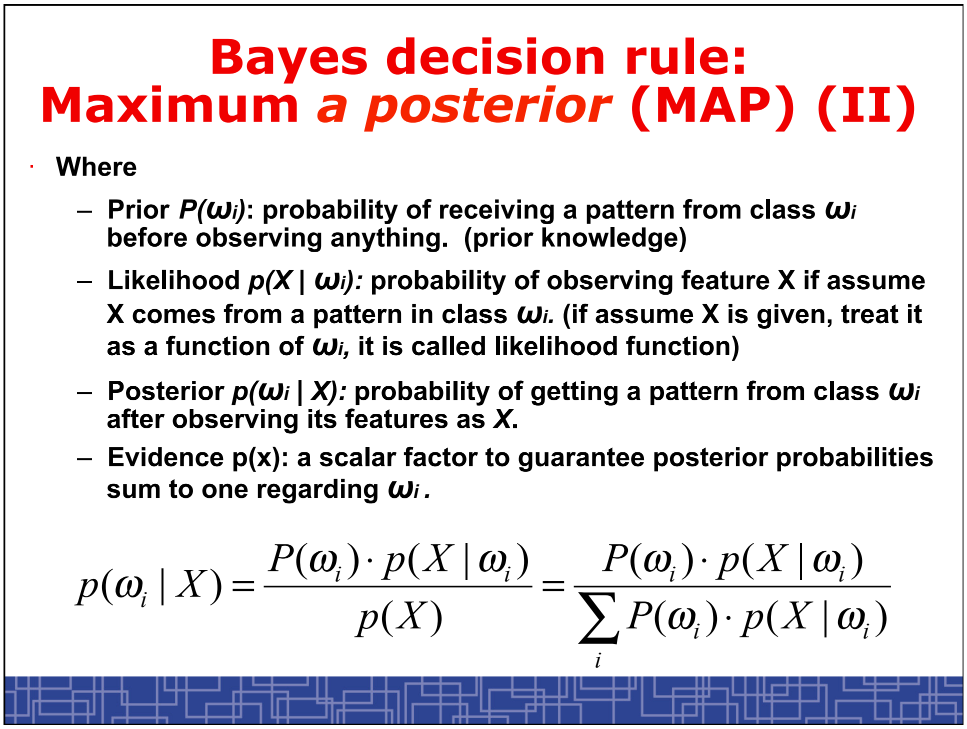
- If not observe any feature of an incoming unknown pattern ***P***, classify it based on prior knowledge only

$$\omega_P = \arg\max_{\omega_i} \mathrm{P}(\omega_i)$$

  - Roughly guess it as the class with largest prior probability

- If observe some features ***X*** of the unknown patter ***P***, we can convert the prior probability ***P(ωi)*** into a posterior probability based on the Bayes theorem:

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



Bayes decision rule: Maximum *a posterior* (MAP) (II)

- Where
  - Prior ***P(ωi)***: probability of receiving a pattern from class ***ωi*** before observing anything. (prior knowledge)
  - Likelihood ***p(X | ωi):*** probability of observing feature X if assume X comes from a pattern in class ***ωi.*** (if assume X is given, treat it as a function of ***ωi,*** it is called likelihood function)
  - Posterior ***p(ωi | X):*** probability of getting a pattern from class ***ωi*** after observing its features as ***X***.
  - Evidence p(x): a scalar factor to guarantee posterior probabilities sum to one regarding ***ωi .***

$$p(\omega_i \mid X) = \frac{P(\omega_i) \cdot p(X \mid \omega_i)}{p(X)} = \frac{P(\omega_i) \cdot p(X \mid \omega_i)}{\sum_i P(\omega_i) \cdot p(X \mid \omega_i)}$$

Bayes decision rule: Maximum *a posterior* (MAP) (II)

- If we observe some features *X* of an unknown pattern, the observation can convert the prior into posterior. Intuitively, we can class the pattern based on the posterior probabilities, resulting in *the maximum a posterior (MAP) decision rule*, also called *Bayes decision rule*.
- For an unknown pattern *P*, after observing some features *X*, we classify it into the class with the largest posterior probability:

$$\omega_P = \arg\max_{\omega_i} p(\omega_i \mid X) = \arg\max_{\omega_i} \frac{P(\omega_i)\cdot p(X \mid \omega_i)}{p(X)}$$

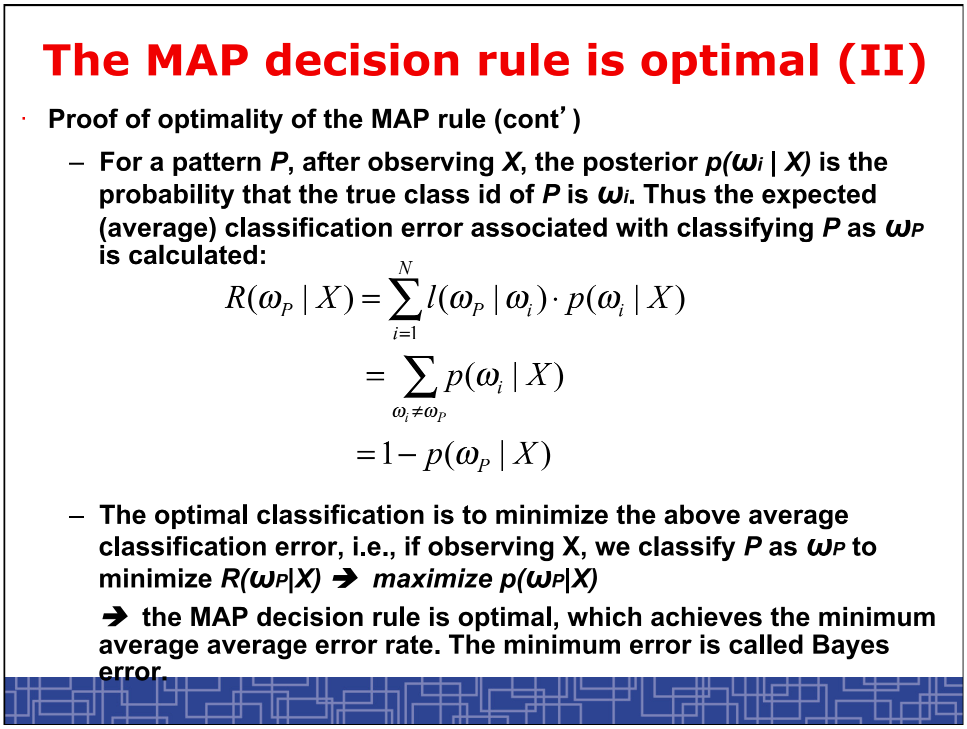
$$= \arg\max_{\omega_i} P(\omega_i)\cdot p(X \mid \omega_i)$$

The MAP decision rule is optimal (I)

- How well the MAP decision rule behaves??
- Optimality: assume we have complete knowledge, including *$P(\omega_i)$* and *$p(X \mid \omega_i)$ (i=1,2,… ,N),* the MAP decision rule is optimal to classify patterns, which means it will achieve the lowest average classification error rate.
- Proof of optimality of the MAP rule:
  - Given a pattern *P*, if its true class id is *ω_i,* but we classify it as *ω_P, then the classification error is counted as $l(\omega_P \mid \omega_i)$:*

$$l(\omega_P \mid \omega_i) = \begin{cases} 0 & (\omega_P = \omega_i) \\ 1 & (\omega_P \neq \omega_i) \end{cases}$$

which is also known as 0-1 loss function.



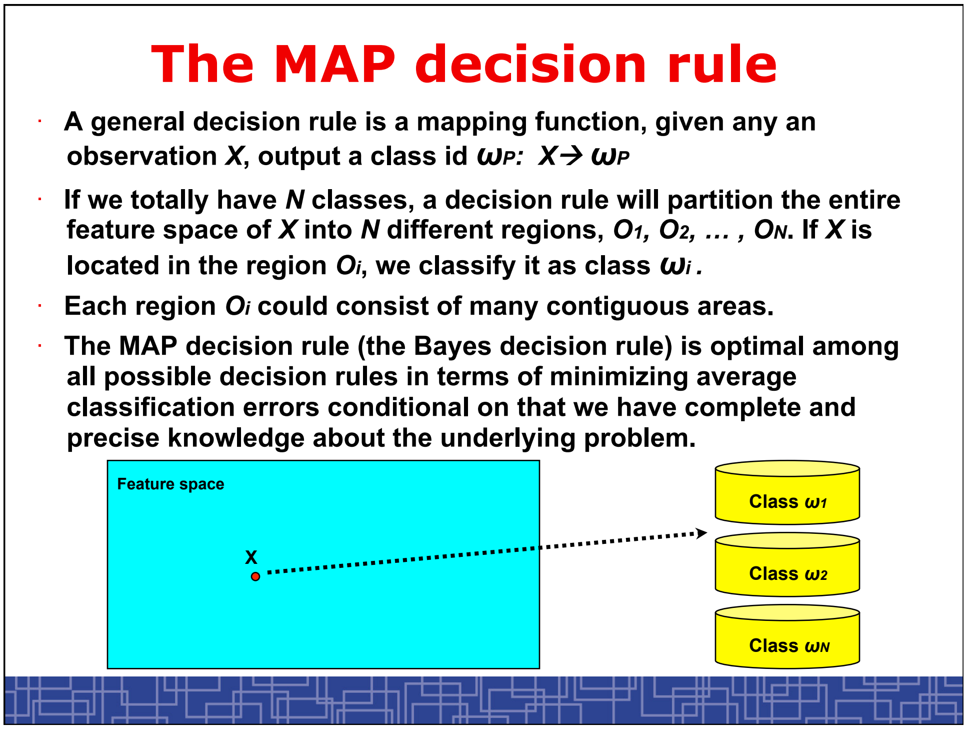
The MAP decision rule is optimal (II)

- Proof of optimality of the MAP rule (cont')
  - For a pattern *P*, after observing *X*, the posterior *p(ω*$_i$ *| X)* is the probability that the true class id of *P* is *ω*$_i$. Thus the expected (average) classification error associated with classifying *P* as *ω*$_P$ is calculated:

$$
\begin{aligned}
R(\omega_P \mid X) &= \sum_{i=1}^{N} l(\omega_P \mid \omega_i) \cdot p(\omega_i \mid X) \\
&= \sum_{\omega_i \neq \omega_P} p(\omega_i \mid X) \\
&= 1 - p(\omega_P \mid X)
\end{aligned}
$$

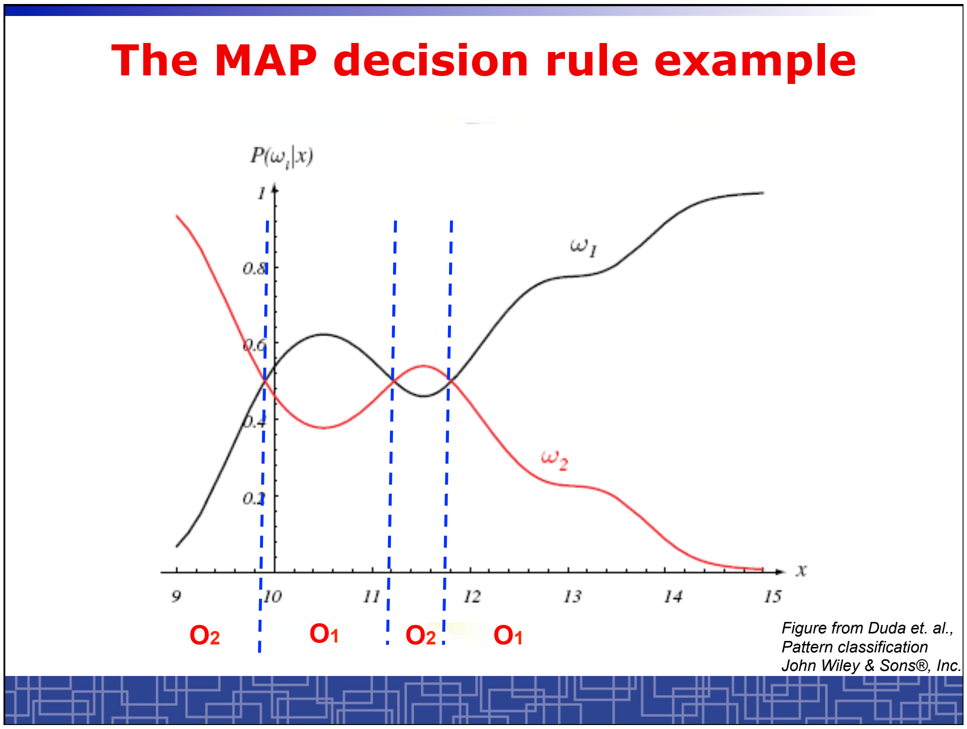
  - The optimal classification is to minimize the above average classification error, i.e., if observing X, we classify *P* as *ω*$_P$ to minimize *R(ω*$_P$*|X)* ➔ *maximize p(ω*$_P$*|X)*

    ➔ the MAP decision rule is optimal, which achieves the minimum average average error rate. The minimum error is called Bayes error.



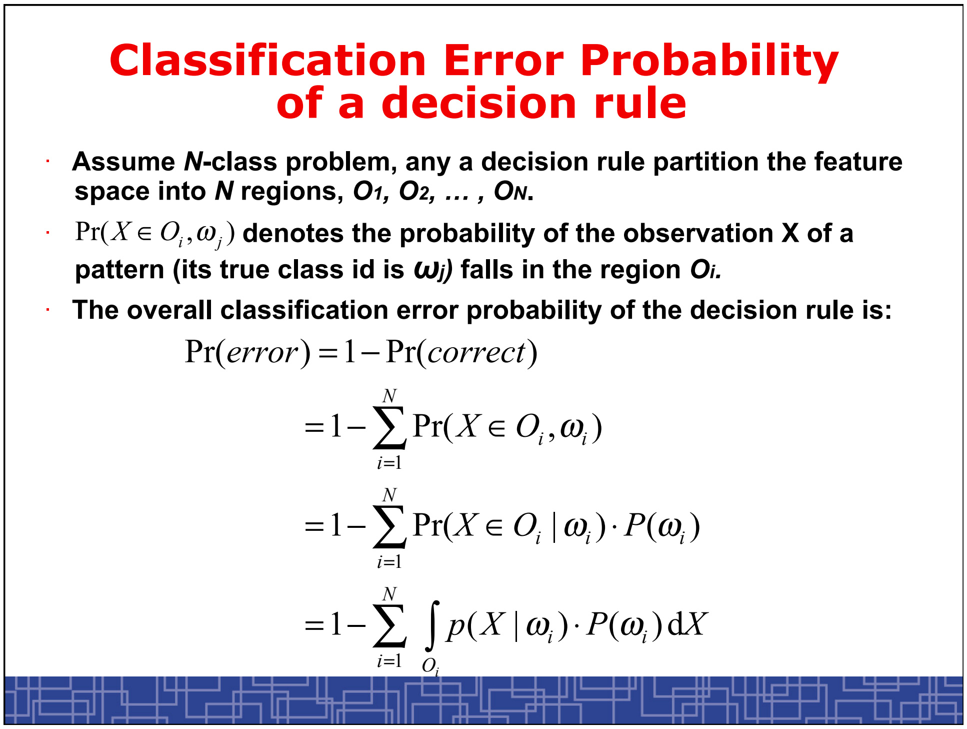
The MAP decision rule

- A general decision rule is a mapping function, given any an observation *X*, output a class id *ω*$_P$*: X*→ *ω*$_P$
- If we totally have *N* classes, a decision rule will partition the entire feature space of *X* into *N* different regions, *O*$_1$*, O*$_2$*, … , O*$_N$. If *X* is located in the region *O*$_i$, we classify it as class *ω*$_i$ *.*
- Each region *O*$_i$ could consist of many contiguous areas.
- The MAP decision rule (the Bayes decision rule) is optimal among all possible decision rules in terms of minimizing average classification errors conditional on that we have complete and precise knowledge about the underlying problem.



The MAP decision rule example

Figure from Duda et. al., Pattern classification John Wiley & Sons®, Inc.



Classification Error Probability of a decision rule

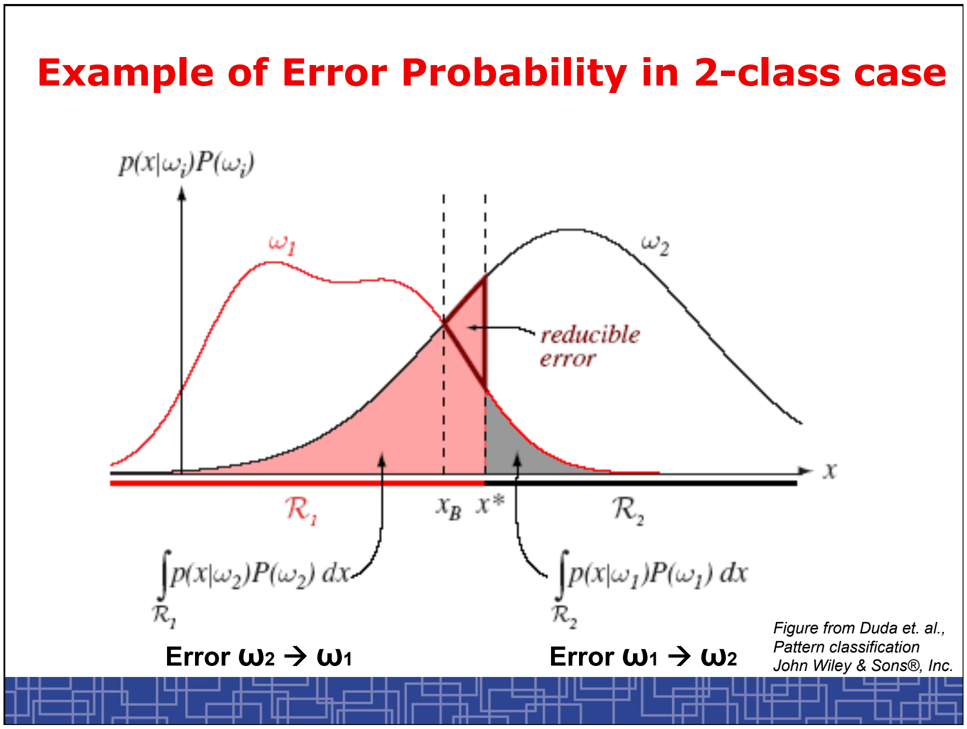
- Assume ***N***-class problem, any a decision rule partition the feature space into ***N*** regions, ***O_1, O_2, … , O_N***.
- $\Pr(X \in O_i, \omega_j)$ **denotes the probability of the observation X of a pattern (its true class id is ω_j) falls in the region O_i.**
- **The overall classification error probability of the decision rule is:**

$$\Pr(error) = 1 - \Pr(correct)$$

$$= 1 - \sum_{i=1}^{N} \Pr(X \in O_i, \omega_i)$$

$$= 1 - \sum_{i=1}^{N} \Pr(X \in O_i \mid \omega_i) \cdot P(\omega_i)$$

$$= 1 - \sum_{i=1}^{N} \int_{O_i} p(X \mid \omega_i) \cdot P(\omega_i) \, \mathrm{d}X$$

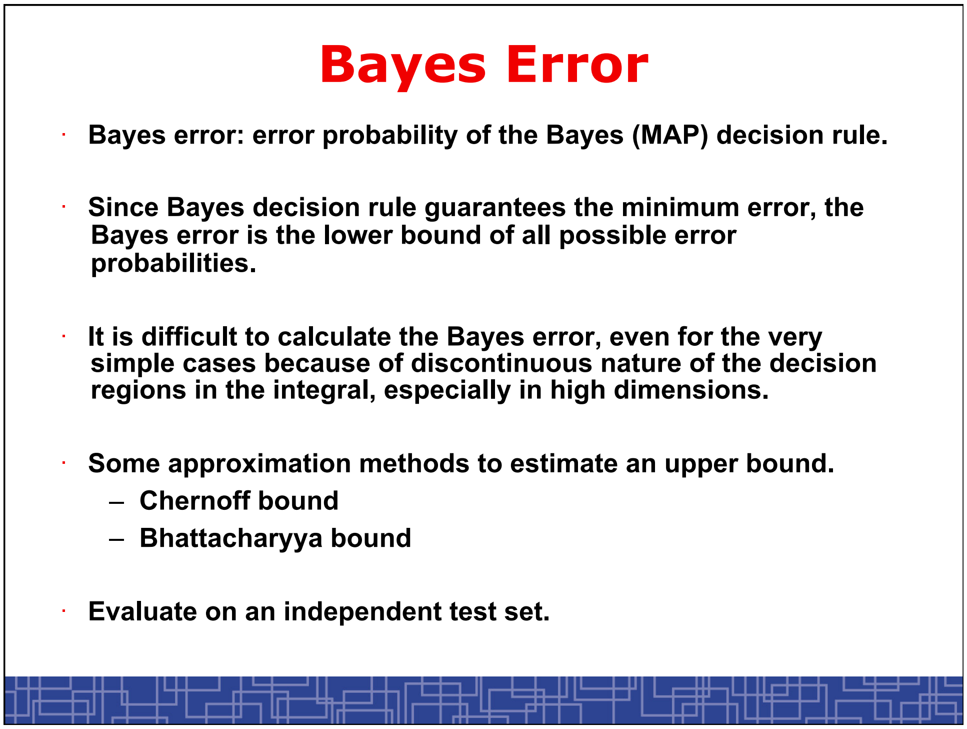


Example of Error Probability in 2-class case

Error ω2 → ω1

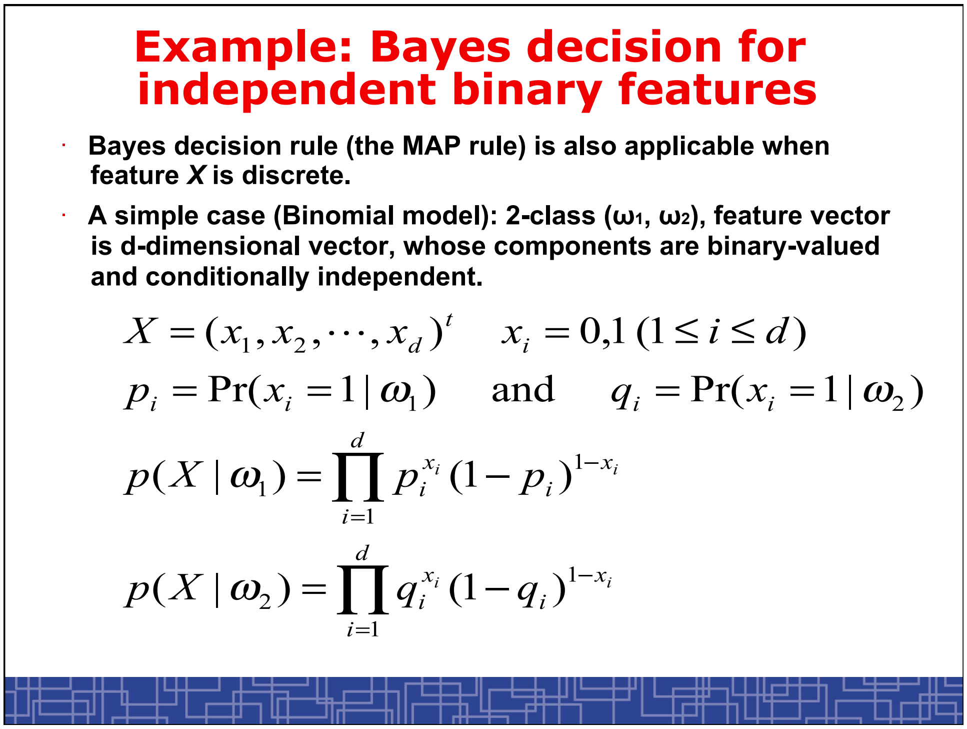
Error ω1 → ω2

Figure from Duda et. al., Pattern classification John Wiley & Sons®, Inc.



Bayes Error

- Bayes error: error probability of the Bayes (MAP) decision rule.
- Since Bayes decision rule guarantees the minimum error, the Bayes error is the lower bound of all possible error probabilities.
- It is difficult to calculate the Bayes error, even for the very simple cases because of discontinuous nature of the decision regions in the integral, especially in high dimensions.
- Some approximation methods to estimate an upper bound.
  - Chernoff bound
  - Bhattacharyya bound
- Evaluate on an independent test set.



Example: Bayes decision for independent binary features

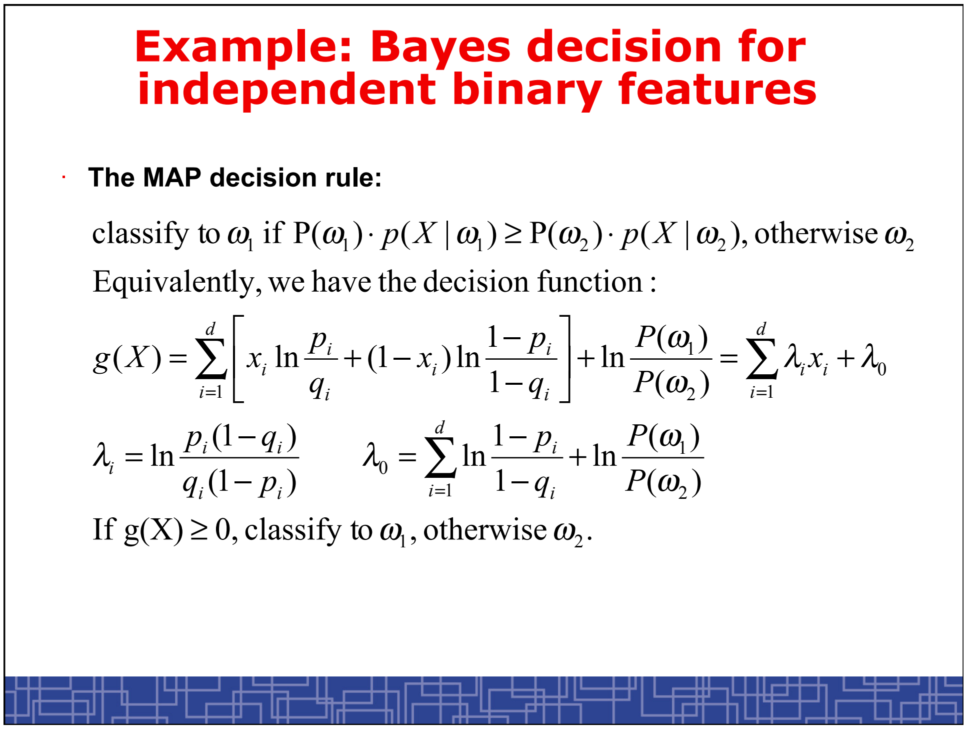
- Bayes decision rule (the MAP rule) is also applicable when feature *X* is discrete.
- A simple case (Binomial model): 2-class (ω1, ω2), feature vector is d-dimensional vector, whose components are binary-valued and conditionally independent.

$$X = (x_1, x_2, \cdots, x_d)^t \quad x_i = 0,1 \; (1 \le i \le d)$$

$$p_i = \Pr(x_i = 1 \mid \omega_1) \quad \text{and} \quad q_i = \Pr(x_i = 1 \mid \omega_2)$$

$$p(X \mid \omega_1) = \prod_{i=1}^{d} p_i^{x_i} (1 - p_i)^{1 - x_i}$$

$$p(X \mid \omega_2) = \prod_{i=1}^{d} q_i^{x_i} (1 - q_i)^{1 - x_i}$$



Example: Bayes decision for independent binary features

- The MAP decision rule:

$$\text{classify to } \omega_1 \text{ if } \mathrm{P}(\omega_1) \cdot p(X \mid \omega_1) \ge \mathrm{P}(\omega_2) \cdot p(X \mid \omega_2), \text{otherwise } \omega_2$$

Equivalently, we have the decision function :

$$g(X) = \sum_{i=1}^{d} \left[x_i \ln \frac{p_i}{q_i} + (1 - x_i) \ln \frac{1 - p_i}{1 - q_i} \right] + \ln \frac{P(\omega_1)}{P(\omega_2)} = \sum_{i=1}^{d} \lambda_i x_i + \lambda_0$$

$$\lambda_i = \ln \frac{p_i (1 - q_i)}{q_i (1 - p_i)} \qquad \lambda_0 = \sum_{i=1}^{d} \ln \frac{1 - p_i}{1 - q_i} + \ln \frac{P(\omega_1)}{P(\omega_2)}$$

$$\text{If } \mathrm{g(X)} \ge 0, \text{classify to } \omega_1, \text{otherwise } \omega_2.$$

Missing features/data (I)

- If we know the full probability structure of a problem, we can construct the optimal Bayes decision rule.
- In some practical situations, for some patterns, we can’t observe the full feature vector described in the probability structure. Only partial information of the feature vector is observed, but some components are missing.
- How to classify such corrupted inputs to obtain minimum average error?

- Let the full feature vector ***X=[X$_g$,X$_b$]***, ***X$_g$*** represents the observed or good features, ***X$_b$*** represents the missing or bad ones.
- In this case, the optimal decision rule is constructed based on the posterior, ***p(ω$_i$ |X$_g$)***, as follows:

$$\omega_P = \arg\max_{\omega_i} \; p(\omega_i \mid X_g)$$

Missing features/data(II)

- Where

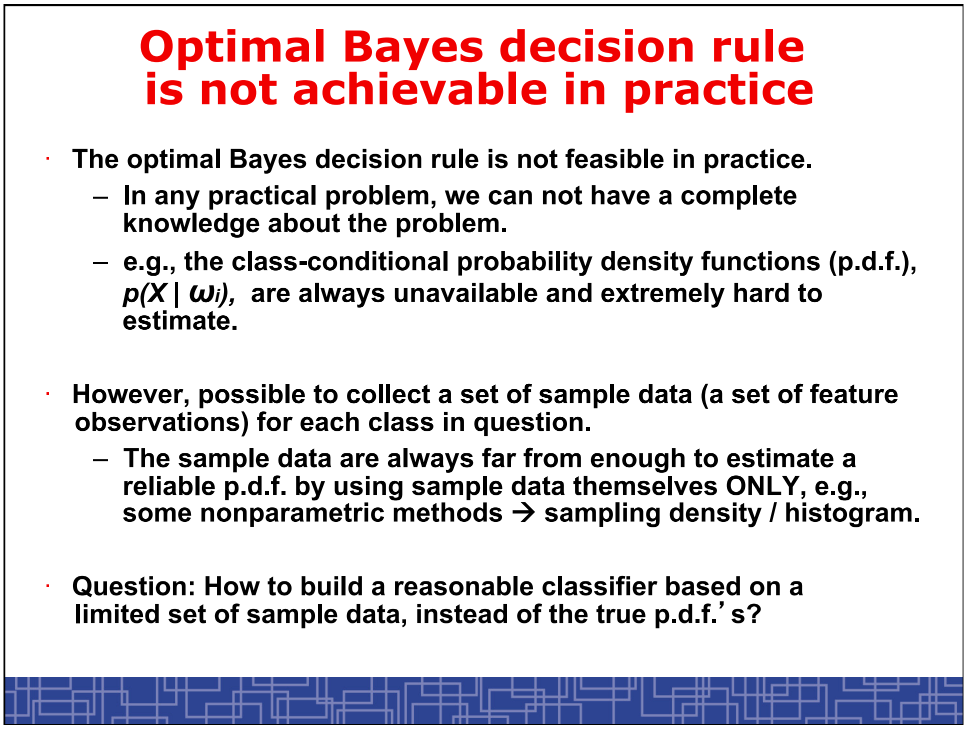
$$p(\omega_i \mid X_g) = \frac{p(\omega_i, X_g)}{p(X_g)} = \frac{\int p(\omega_i, X_g, X_b)\,\mathrm{d}X_b}{p(X_g)}$$

$$(1) \quad = \frac{\int p(\omega_i \mid X_g, X_b)\cdot p(X_g, X_b)\mathrm{d}X_b}{p(X_g)}$$

$$= \frac{\int p(\omega_i \mid X)\cdot p(X)\,\mathrm{d}X_b}{\int p(X)\,\mathrm{d}X_b}$$

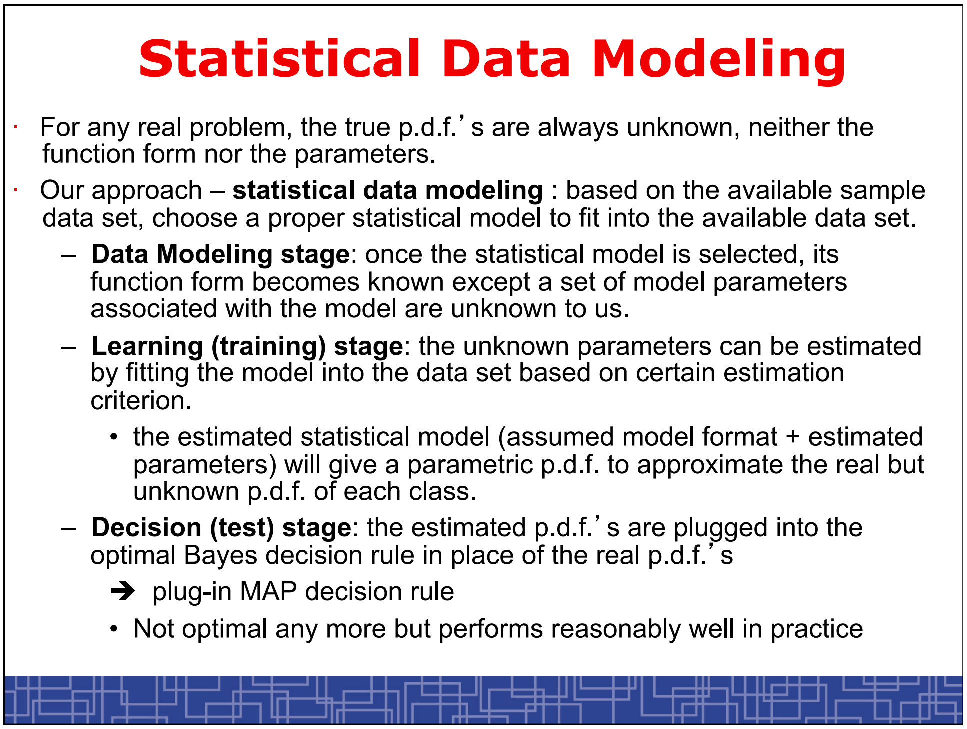
$$(2) \quad = \frac{\int P(\omega_i)\cdot p(X_g, X_b \mid \omega_i)\,\mathrm{d}X_b}{\sum_{\omega_i}\int P(\omega_i)\cdot p(X_g, X_b \mid \omega_i)\,\mathrm{d}X_b}$$

$$= \frac{\int P(\omega_i)\cdot p(X \mid \omega_i)\,\mathrm{d}X_b}{\sum_{\omega_i}\int P(\omega_i)\cdot p(X \mid \omega_i)\,\mathrm{d}X_b}$$



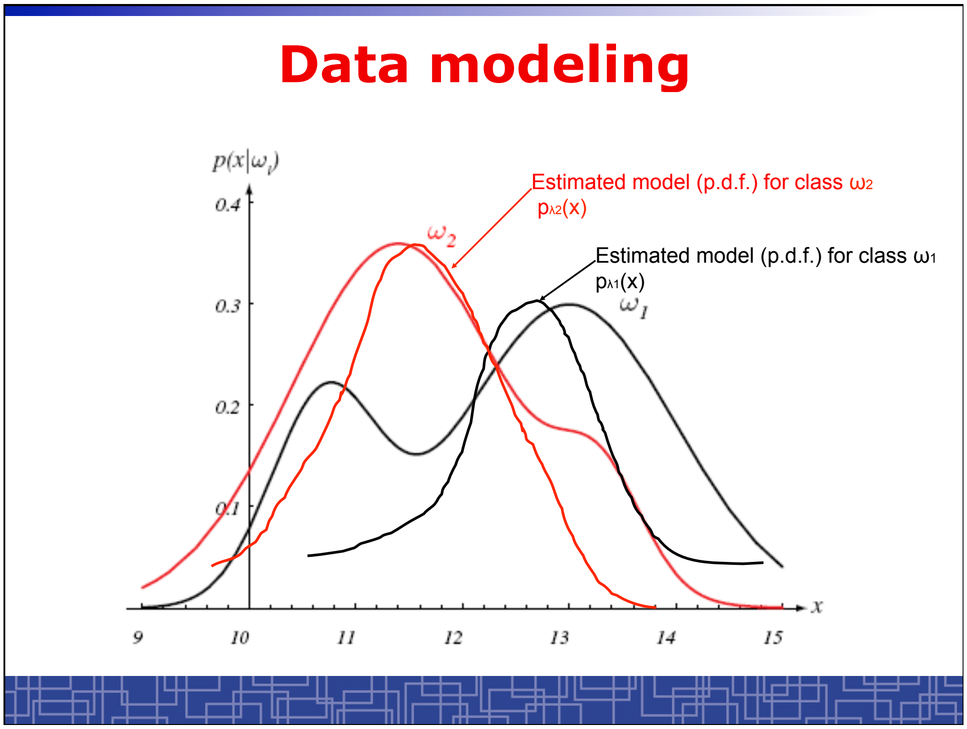
Optimal Bayes decision rule is not achievable in practice

- The optimal Bayes decision rule is not feasible in practice.
  - In any practical problem, we can not have a complete knowledge about the problem.
  - e.g., the class-conditional probability density functions (p.d.f.), $p(X \mid \omega_i)$, are always unavailable and extremely hard to estimate.
- However, possible to collect a set of sample data (a set of feature observations) for each class in question.
  - The sample data are always far from enough to estimate a reliable p.d.f. by using sample data themselves ONLY, e.g., some nonparametric methods → sampling density / histogram.
- Question: How to build a reasonable classifier based on a limited set of sample data, instead of the true p.d.f.'s?



Statistical Data Modeling

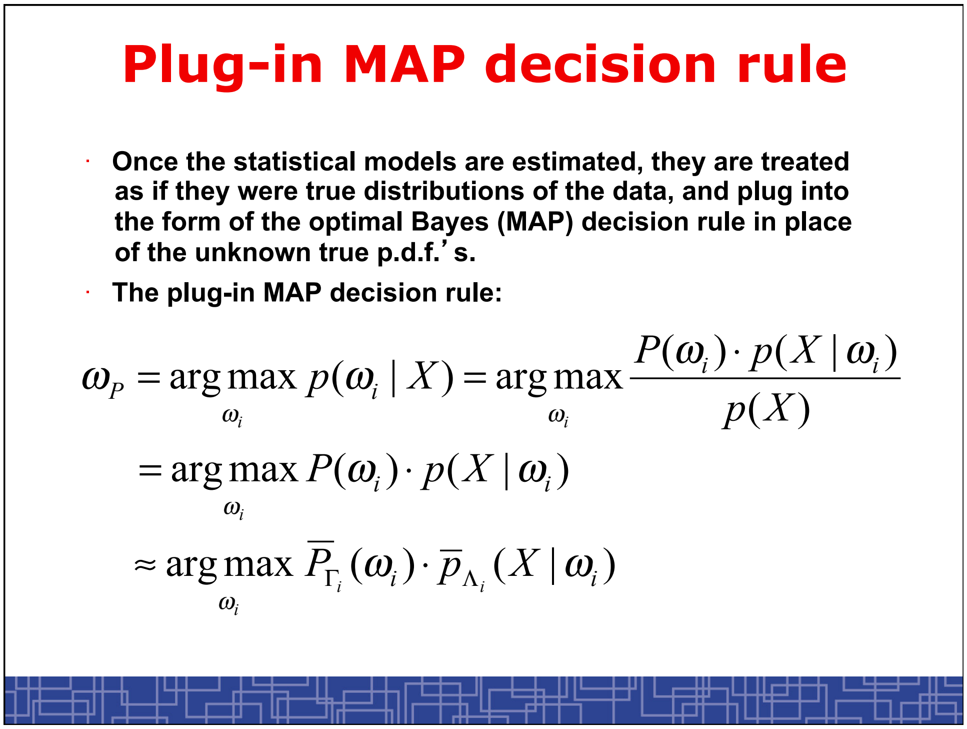
- For any real problem, the true p.d.f.'s are always unknown, neither the function form nor the parameters.
- Our approach – **statistical data modeling** : based on the available sample data set, choose a proper statistical model to fit into the available data set.
  - **Data Modeling stage**: once the statistical model is selected, its function form becomes known except a set of model parameters associated with the model are unknown to us.
  - **Learning (training) stage**: the unknown parameters can be estimated by fitting the model into the data set based on certain estimation criterion.
    - the estimated statistical model (assumed model format + estimated parameters) will give a parametric p.d.f. to approximate the real but unknown p.d.f. of each class.
  - **Decision (test) stage**: the estimated p.d.f.'s are plugged into the optimal Bayes decision rule in place of the real p.d.f.'s
    - ➔ plug-in MAP decision rule
    - Not optimal any more but performs reasonably well in practice



Data modeling

Estimated model (p.d.f.) for class ω2 $p_{\Lambda_2}(x)$

Estimated model (p.d.f.) for class ω1 $p_{\Lambda_1}(x)$



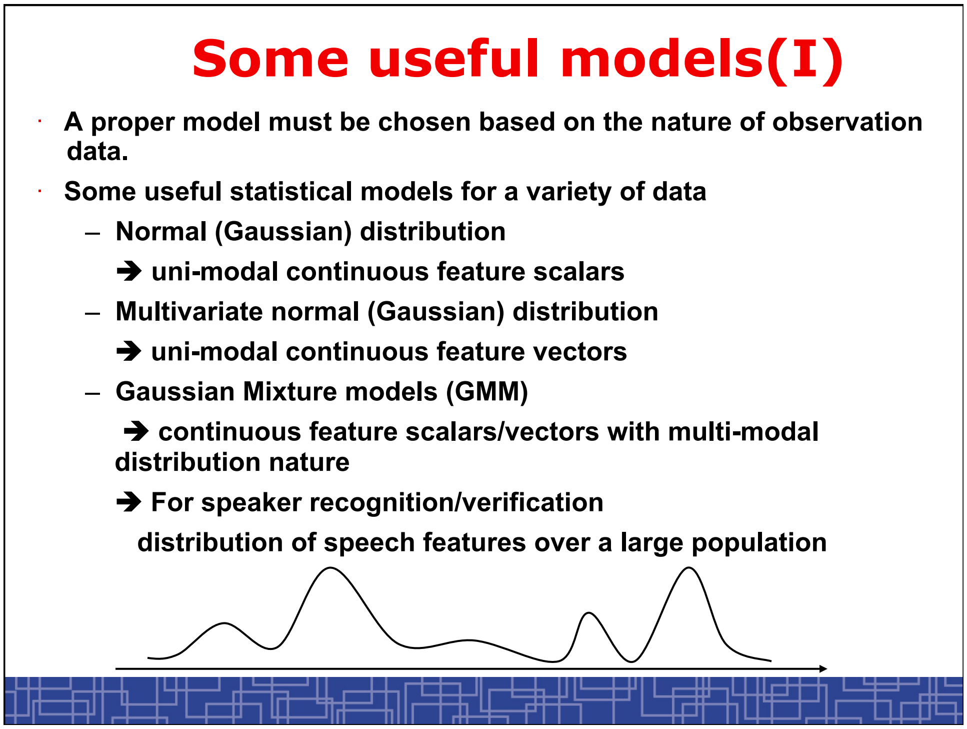
Plug-in MAP decision rule

- Once the statistical models are estimated, they are treated as if they were true distributions of the data, and plug into the form of the optimal Bayes (MAP) decision rule in place of the unknown true p.d.f.’s.
- The plug-in MAP decision rule:

$$\omega_P = \arg\max_{\omega_i} p(\omega_i \mid X) = \arg\max_{\omega_i} \frac{P(\omega_i)\cdot p(X \mid \omega_i)}{p(X)}$$

$$= \arg\max_{\omega_i} P(\omega_i)\cdot p(X \mid \omega_i)$$

$$\approx \arg\max_{\omega_i} \overline{P}_{\Gamma_i}(\omega_i)\cdot \overline{p}_{\Lambda_i}(X \mid \omega_i)$$



Some useful models(I)

- A proper model must be chosen based on the nature of observation data.
- Some useful statistical models for a variety of data
  - Normal (Gaussian) distribution

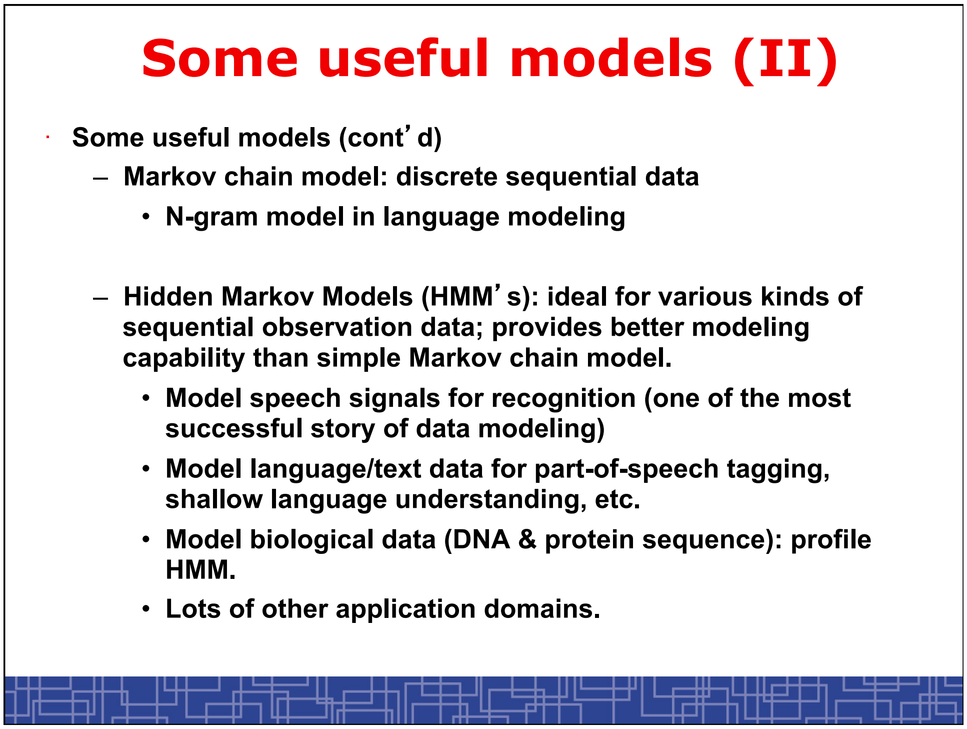
    ➔ uni-modal continuous feature scalars
  - Multivariate normal (Gaussian) distribution

    ➔ uni-modal continuous feature vectors
  - Gaussian Mixture models (GMM)

    ➔ continuous feature scalars/vectors with multi-modal distribution nature

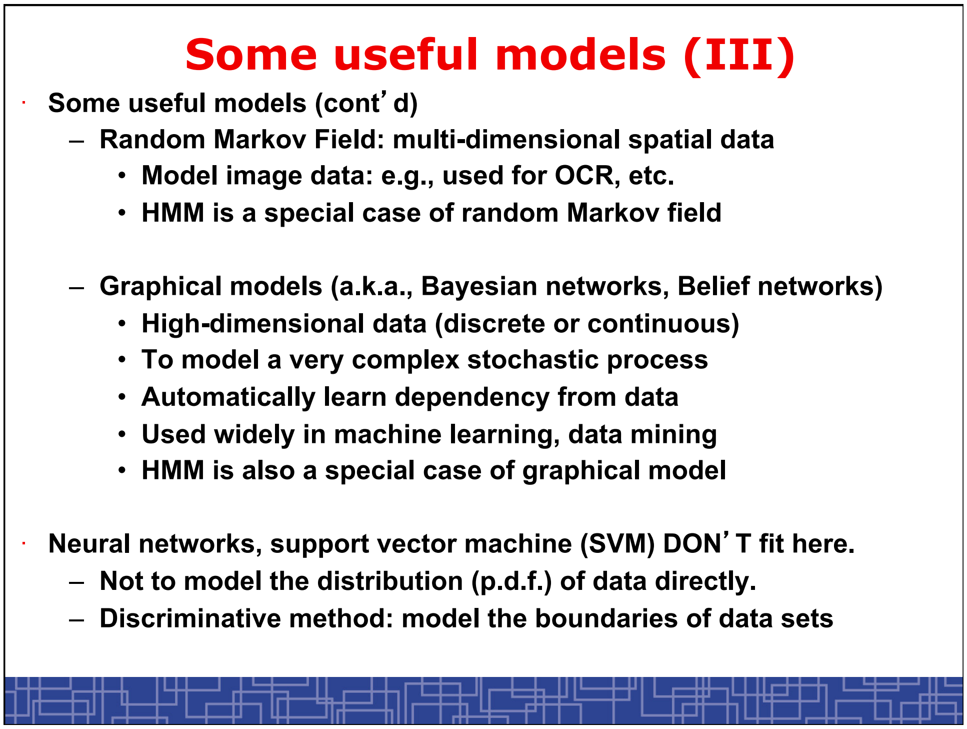
    ➔ For speaker recognition/verification

    distribution of speech features over a large population



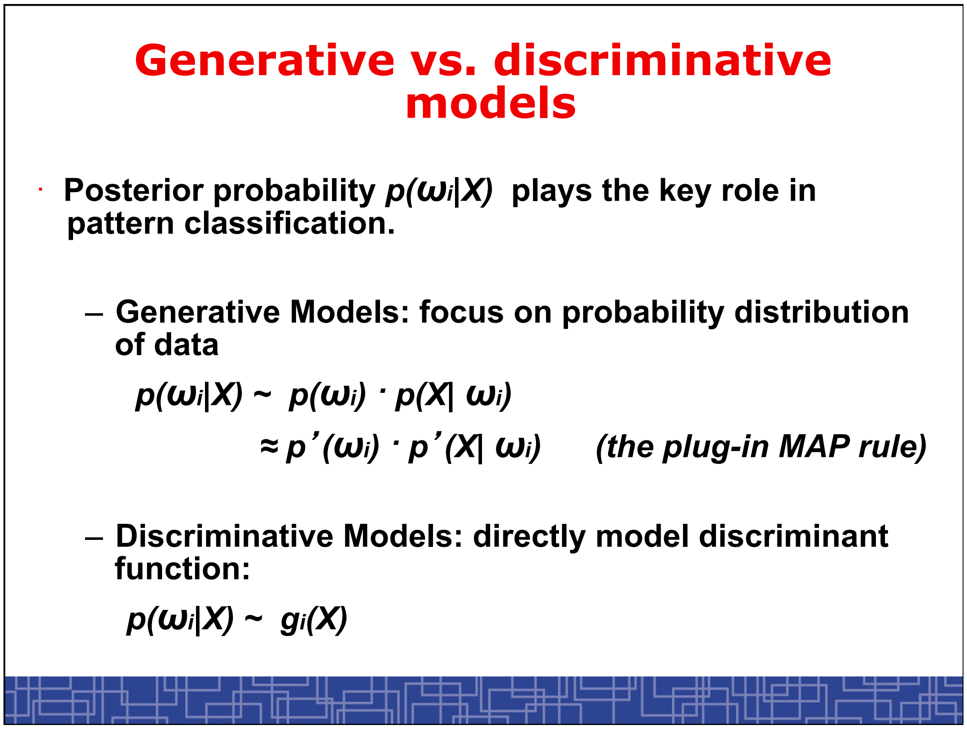
Some useful models (II)

- Some useful models (cont'd)
  - Markov chain model: discrete sequential data
    - N-gram model in language modeling
  - Hidden Markov Models (HMM's): ideal for various kinds of sequential observation data; provides better modeling capability than simple Markov chain model.
    - Model speech signals for recognition (one of the most successful story of data modeling)
    - Model language/text data for part-of-speech tagging, shallow language understanding, etc.
    - Model biological data (DNA & protein sequence): profile HMM.
    - Lots of other application domains.



Some useful models (III)

- Some useful models (cont’d)
  - Random Markov Field: multi-dimensional spatial data
    - Model image data: e.g., used for OCR, etc.
    - HMM is a special case of random Markov field
  - Graphical models (a.k.a., Bayesian networks, Belief networks)
    - High-dimensional data (discrete or continuous)
    - To model a very complex stochastic process
    - Automatically learn dependency from data
    - Used widely in machine learning, data mining
    - HMM is also a special case of graphical model
- Neural networks, support vector machine (SVM) DON’T fit here.
  - Not to model the distribution (p.d.f.) of data directly.
  - Discriminative method: model the boundaries of data sets



Generative vs. discriminative models

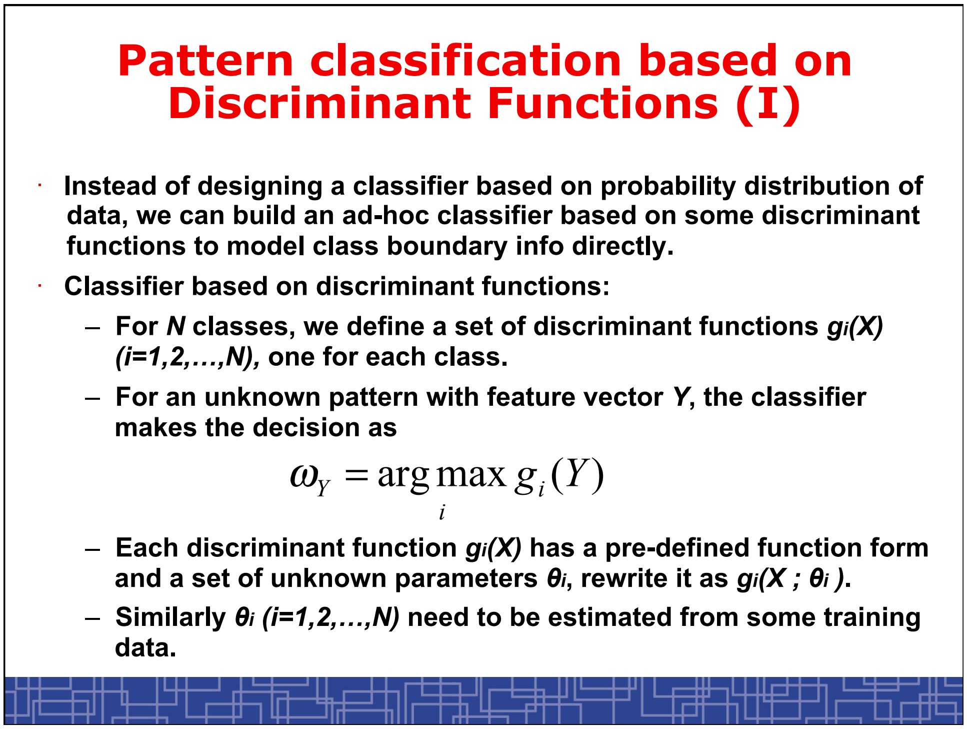
- Posterior probability $p(\omega_i|X)$ plays the key role in pattern classification.
  - Generative Models: focus on probability distribution of data

    $$p(\omega_i|X) \sim p(\omega_i) \cdot p(X|\,\omega_i)$$

    $$\approx p'(\omega_i) \cdot p'(X|\,\omega_i) \qquad \textit{(the plug-in MAP rule)}$$

  - Discriminative Models: directly model discriminant function:

    $$p(\omega_i|X) \sim g_i(X)$$

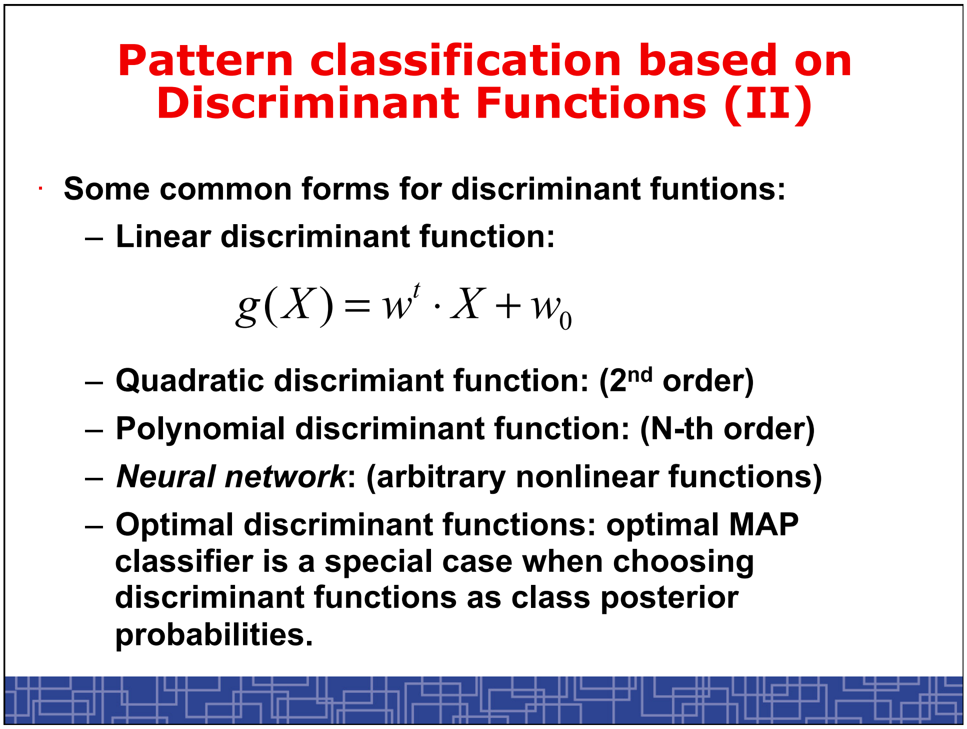


Pattern classification based on Discriminant Functions (I)

- Instead of designing a classifier based on probability distribution of data, we can build an ad-hoc classifier based on some discriminant functions to model class boundary info directly.
- Classifier based on discriminant functions:
  - For *N* classes, we define a set of discriminant functions $g_i(X)$ *$(i=1,2,\ldots,N)$,* one for each class.
  - For an unknown pattern with feature vector *Y*, the classifier makes the decision as

$$\omega_Y = \arg\max_i g_i(Y)$$

  - Each discriminant function $g_i(X)$ has a pre-defined function form and a set of unknown parameters θ_i, rewrite it as $g_i(X\ ;\ \theta_i\ )$.
  - Similarly θ_i $(i=1,2,\ldots,N)$ need to be estimated from some training data.



Pattern classification based on Discriminant Functions (II)

- Some common forms for discriminant funtions:
  - Linear discriminant function:

$$g(X) = w^t \cdot X + w_0$$

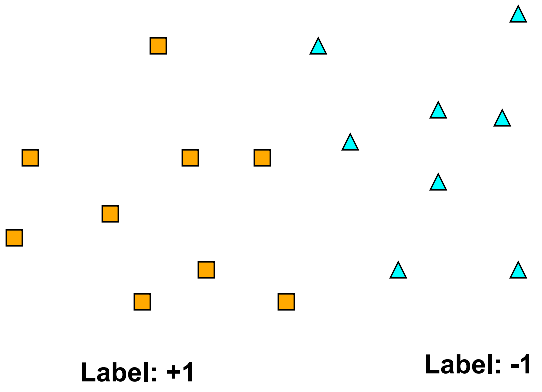
  - Quadratic discrimiant function: (2nd order)
  - Polynomial discriminant function: (N-th order)
  - *Neural network*: (arbitrary nonlinear functions)
  - Optimal discriminant functions: optimal MAP classifier is a special case when choosing discriminant functions as class posterior probabilities.

Pattern classification based on Discriminant Functions (III)

- Unknown parameters of discriminant functions are estimated by some gradient descent method to optimize an objective function:
  - Linear regression: Achieving a good mapping.
  - Logistic regression: Minimizing empirical classification errors.
  - support vector machine (SVM): Maximizing separation margin:

Linear Regression

- Find a good mapping from X to y



Linear Regression

- **Find a good mapping from X to y:**

$$\mathrm{X} = \begin{bmatrix} \mathrm{x}_1 \\ \mathrm{x}_2 \\ \vdots \\ \mathrm{x}_T \end{bmatrix} \xrightarrow{\mathrm{Y} = \mathrm{X}\mathrm{w}^T} \mathrm{Y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_T \end{bmatrix} = \begin{bmatrix} +1 \\ -1 \\ \vdots \\ +1 \end{bmatrix}$$

$$w^* = \arg\min_w \sum_i \left(X_i w^T - y_i \right)^2 = \arg\min_w \sum_i \left(y_i X_i w^T - 1 \right)^2$$

$$\mathrm{w}^* = \left(X^T X \right)^{-1} X^T Y$$

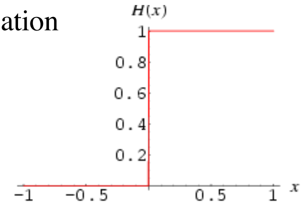
- **Linear regression does NOT work well for classification**

Logistic Regression

- **Counting errors in training samples.**

$$(\mathrm{x}_i, y_i) \Rightarrow \begin{cases} g_i = -y_i \mathrm{x}_i \mathrm{w}^T < 0 & \text{correct classification} \\ g_i = -y_i \mathrm{x}_i \mathrm{w}^T > 0 & \text{wrong classification} \end{cases}$$

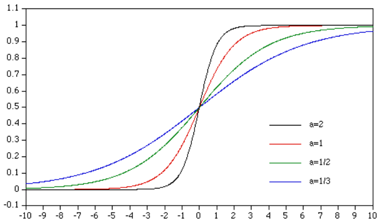
$$w^* = \arg\min_w \sum_i H(g_i) = \arg\min_w \sum_i H(y_i \mathrm{x}_i \mathrm{w}^T)$$



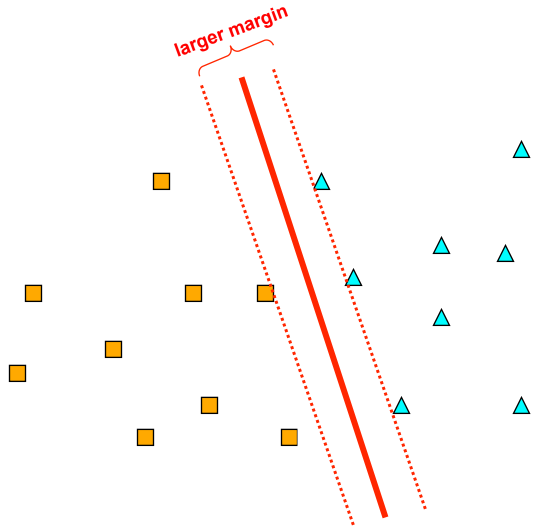
$$w^* = \arg\min_w \sum_i l(g_i) = \arg\min_w \sum_i l(y_i \mathrm{x}_i \mathrm{w}^T)$$

$$l(x) = \frac{1}{1 + e^{-\sigma x}}$$

logistic sigmoid function

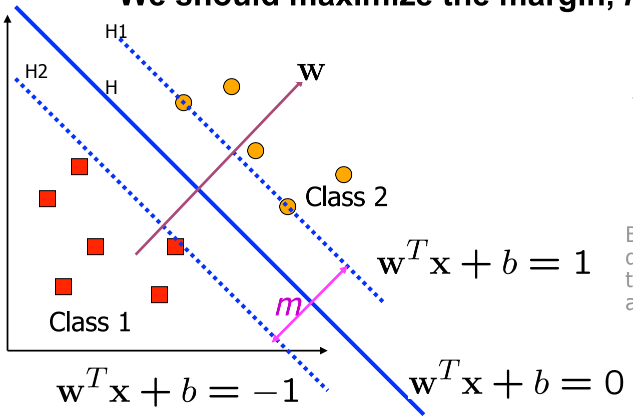


Large-Margin Classifier: Support Vector Machine (SVM)



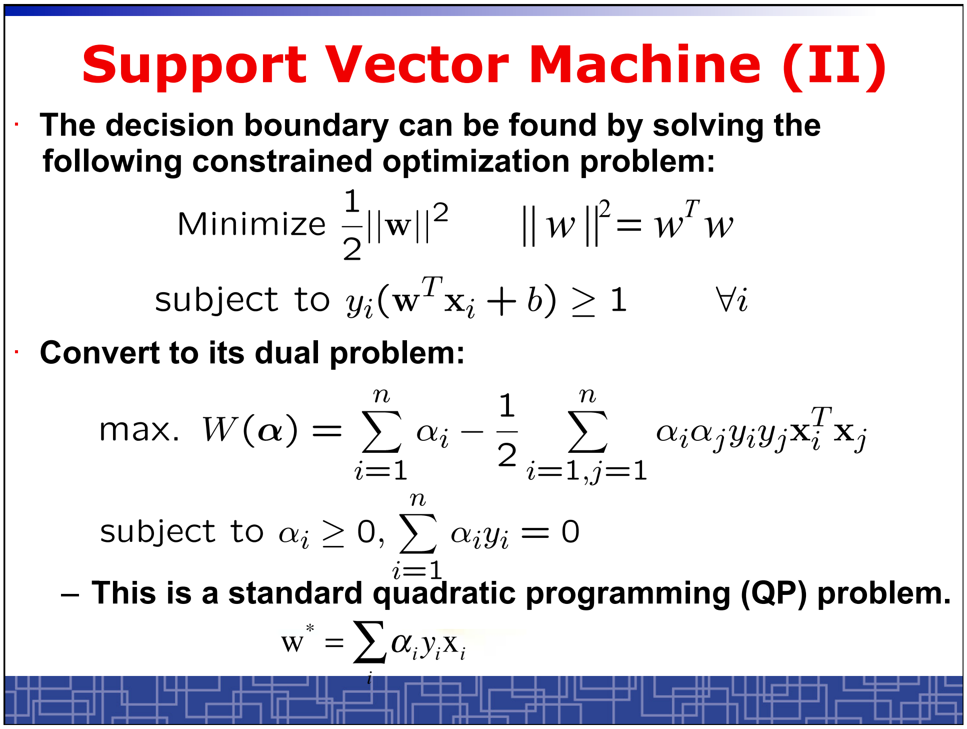
Support Vector Machine (I)

- The decision boundary H should be as far away from the data of both classes as possible
  - We should maximize the margin, *m*


H1
H2
H
w
Class 2
Class 1
m
$\mathbf{w}^T\mathbf{x} + b = 1$
$\mathbf{w}^T\mathbf{x} + b = -1$
$\mathbf{w}^T\mathbf{x} + b = 0$

$$m = \frac{2}{||\mathbf{w}||}$$

Because the perpendicular distance from H1, H2 to the origin is |1-b|/||w|| and |-1-b|/||w||



Support Vector Machine (II)

- The decision boundary can be found by solving the following constrained optimization problem:

$$\text{Minimize } \frac{1}{2}\|\mathbf{w}\|^2 \qquad \| w \|^2 = w^T w$$

$$\text{subject to } y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1 \qquad \forall i$$

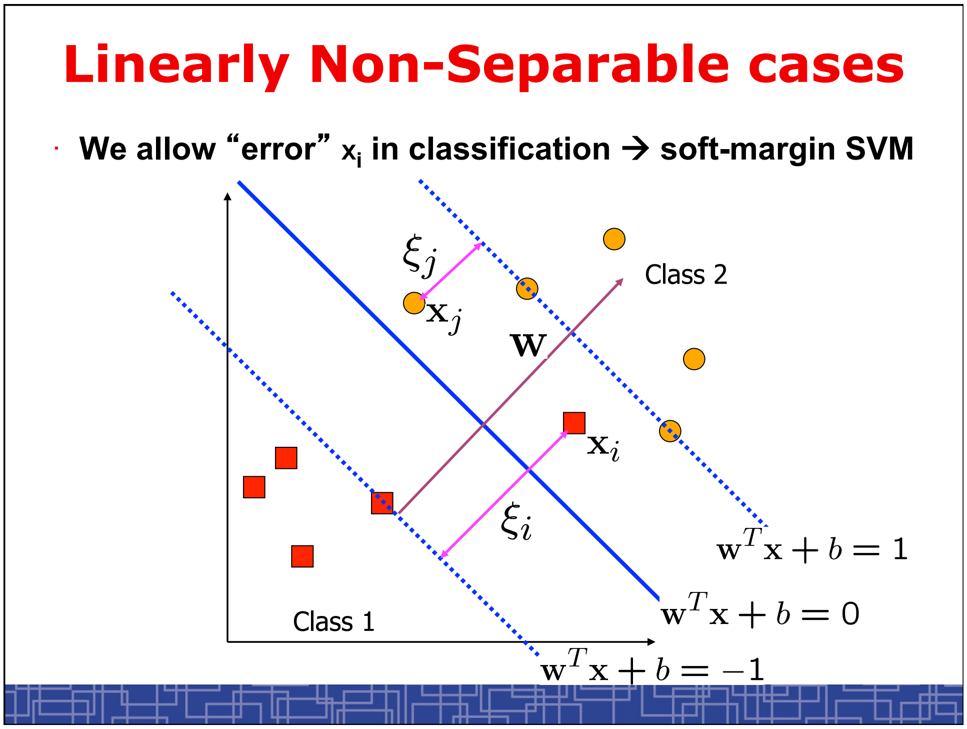
- Convert to its dual problem:

$$\text{max. } W(\boldsymbol{\alpha}) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1,j=1}^{n} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j$$

$$\text{subject to } \alpha_i \geq 0, \sum_{i=1}^{n} \alpha_i y_i = 0$$

– This is a standard quadratic programming (QP) problem.

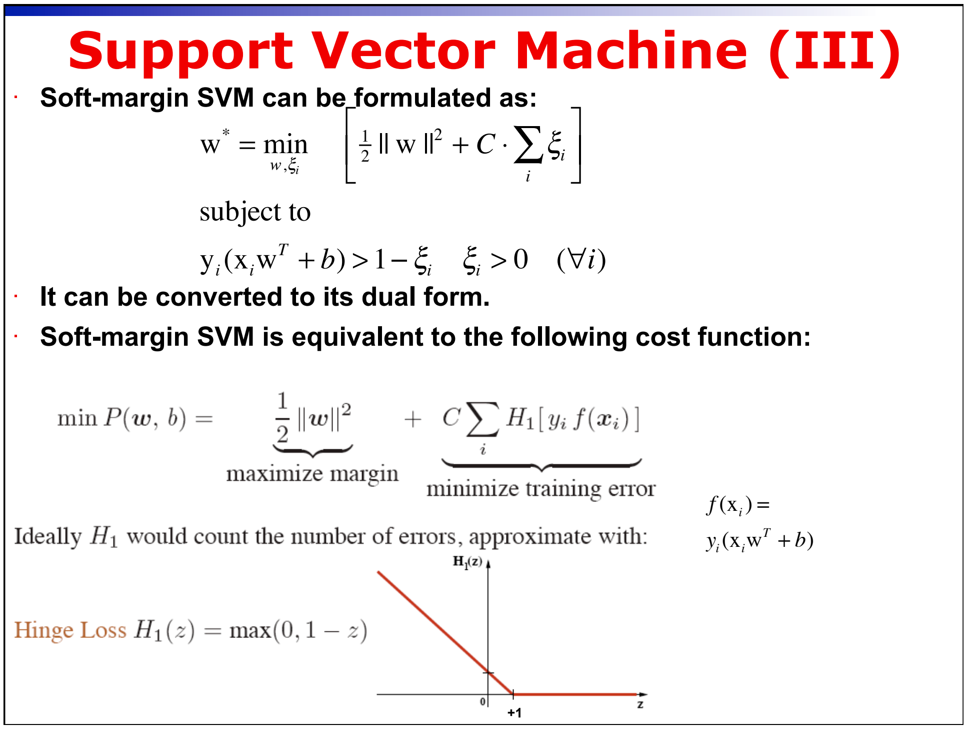
$$\mathrm{w}^* = \sum_i \alpha_i y_i \mathrm{x}_i$$



Linearly Non-Separable cases

- We allow "error" x_i in classification → soft-margin SVM

ξ_j
$\mathbf{x}_j$
$\mathbf{w}$
Class 2
$\mathbf{x}_i$
ξ_i
$\mathrm{w}^T\mathrm{x} + b = 1$
$\mathrm{w}^T\mathrm{x} + b = 0$
Class 1
$\mathrm{w}^T\mathrm{x} + b = -1$



Support Vector Machine (III)

- Soft-margin SVM can be formulated as:

$$\mathrm{w}^* = \min_{w,\xi_i} \left[\tfrac{1}{2} \| \mathrm{w} \|^2 + C \cdot \sum_i \xi_i \right]$$

subject to

$$\mathrm{y}_i(\mathrm{x}_i \mathrm{w}^T + b) > 1 - \xi_i \quad \xi_i > 0 \quad (\forall i)$$

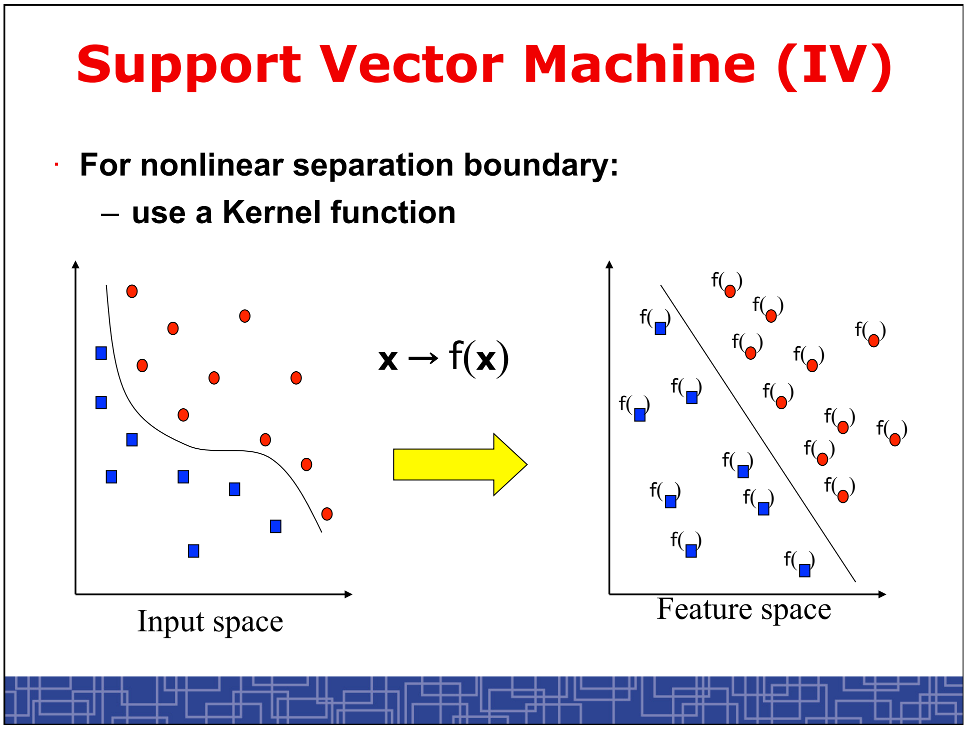
- It can be converted to its dual form.
- Soft-margin SVM is equivalent to the following cost function:

$$\min P(\boldsymbol{w}, b) = \underbrace{\frac{1}{2}\|\boldsymbol{w}\|^2}_{\text{maximize margin}} + \underbrace{C \sum_i H_1[\, y_i \, f(\boldsymbol{x}_i) \,]}_{\text{minimize training error}}$$

$f(\mathrm{x}_i) =$
$y_i(\mathrm{x}_i \mathrm{w}^T + b)$

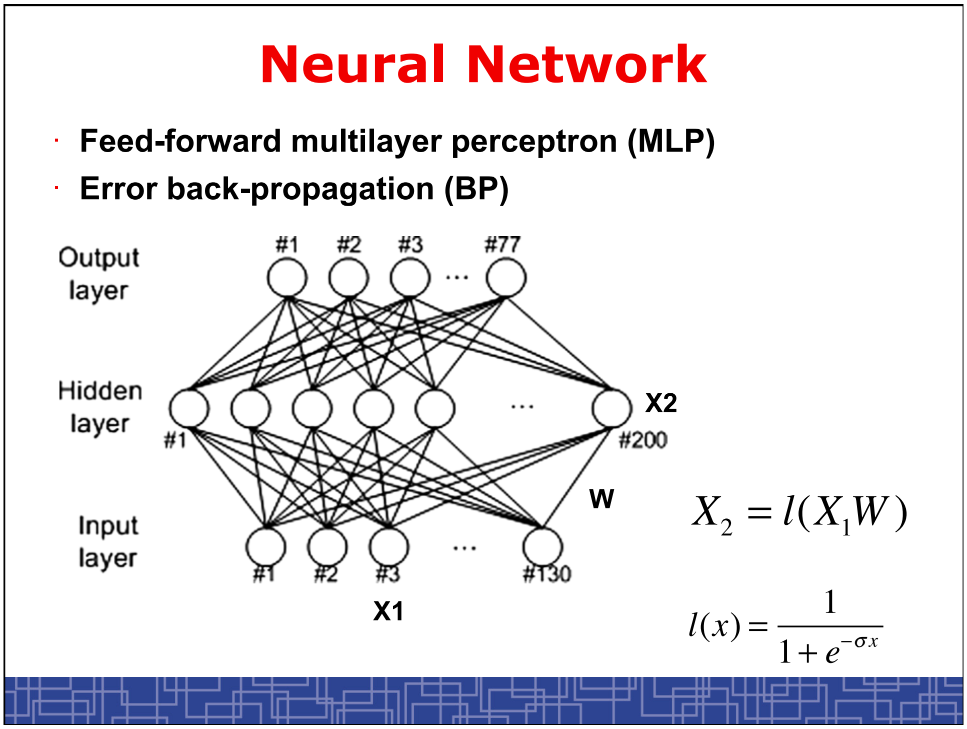
Ideally H_1 would count the number of errors, approximate with:

Hinge Loss $H_1(z) = \max(0, 1 - z)$



Support Vector Machine (IV)

- For nonlinear separation boundary:
  - use a Kernel function



Neural Network

- Feed-forward multilayer perceptron (MLP)
- Error back-propagation (BP)

$$X_2 = l(X_1 W)$$

$$l(x) = \frac{1}{1 + e^{-\sigma x}}$$