

# Overview of Pattern Recognition Methods for Speech & Speaker Recognition

Professor Marie Roch  
San Diego State University

Huang et al.: 4-4.2.3, 4.4



## Bayes Decision Theory

- Foundation of statistical pattern recognition
- Attempts to build a function which minimizes the classification error rate.



## Priors

- Suppose we want to know the probability that an observation will come from one of  $s$  classes:  $\omega_1, \omega_2, \dots, \omega_s$ .
- Assuming that we know the distribution of the classes, our guess with *prior* knowledge would be  $P(\omega_j)$ .

## Posteriors

- However, if the event has already occurred, i.e. we have an instance  $x$  of random variable  $X$ ,
- then we should use that information and compute the *posterior* probability, the probability of a class *after* having observed the data :  
$$P(\omega_i | x)$$

## Bayes rule

- In practice, it is difficult to estimate the posterior directly.
- Bayes rule can be used to transform this into a problem which is easier to work with:

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

## Rewriting the posterior

$$\begin{aligned} P(\omega_i | x) &= \frac{P(\omega_i \cap x)}{P(x)} && \text{as } P(A|B) = \frac{P(A \cap B)}{P(B)} \\ &= \frac{P(x|\omega_i)P(\omega_i)}{P(x)} && \text{as } P(A \cap B) = P(A|B)P(B) \end{aligned}$$

- $\frac{P(x|\omega_i)P(\omega_i)}{P(x)}$  can be more easily estimated.

## Pieces of the puzzle

- The class-conditional pdf or *likelihood function*:  $P(x | \omega_i)$
- The prior probability:  $P(\omega_i)$
- and the probability of the observation:

$$P(x) = \sum_{i=1}^s P(x | \omega_i) P(\omega_i) \quad \text{where } s \text{ is the number of classes}$$

## Bayes decision rule

- Bayes decision rule states that we select the class whose posterior probability is highest:

$$\begin{aligned} \text{class} &= \arg \max_{i \in \text{all classes}} P(\omega_i | x) \\ &= \arg \max_{i \in \text{all classes}} \frac{P(x | \omega_i) P(\omega_i)}{P(x)} \quad \text{Bayes rule} \\ &= \arg \max_{i \in \text{all classes}} P(x | \omega_i) P(\omega_i) \quad \text{as } P(x) \text{ independent of } i \end{aligned}$$

## Is our decision rule any good?

- Let  $\delta_i$  denote a decision of class  $\omega_i$ .
- To analyze this, we must determine the *conditional risk* of our decision rule.
- Conditional risk is the expected loss of  $\delta_i$ .

## Loss

- To determine conditional risk, we will need a loss function. Define a 0-1 *loss* function:

$$l(\delta_i | \omega_j) = \begin{cases} 0 & i = j \\ 1 & i \neq j \end{cases} \quad i, j \in \{1, \dots, s\}$$

- This function returns 0 when the decision matches the true class  $\omega_j$  of the data, and 1 when it does not.

## Conditional risk

- The *conditional risk* is the expected loss of the decision with respect to the class conditioned upon the observation  $x$ :

$$\begin{aligned}R_{\Omega|X=x}(\delta_i | x) &= E_{\Omega|X=x}[l(\delta_i | \omega)] \\ &= \sum_{j=1}^s l(\delta_i | \omega_j)P(\omega_j | x)\end{aligned}$$

## Overall risk

- Let  $\delta(x)$  denote a decision rule from  $\Lambda = \{\delta_1, \dots, \delta_s\}$  associated with observation  $x$ .
- The *overall risk* is the expected value of the conditional risk with respect to all observations.

## Overall risk

$$\begin{aligned}R_X &= E_X [R_{\Omega|X=x}(\delta(x) | x)] \\&= \int_{-\infty}^{\infty} R_{\Omega|X=x}(\delta(x) | x)P(x)dx \\&= \int_{-\infty}^{\infty} R_{\Omega|X=x}(\delta(x) | x) \sum_{i=1}^s P(x | \omega_i)P(\omega_i)dx\end{aligned}$$

## Bayes decision rule

- If  $\delta(x)$  is chosen to minimize the conditional risk  $R(\delta_i|x)$  for every  $x$ , then the overall risk is minimized.
- To minimize conditional risk
  - compute the conditional risk  $R(\delta_i|x)$  for each  $\delta_i$
  - select the  $\delta_i$  which results smallest conditional risk:

$$R_{\Omega|X=x}(\delta_i | x) = \sum_{j=1}^s l(\delta_i | \omega_j)P(\omega_j | x)$$

## Conditional risk in another form

- Rewriting will make it easier to minimize

$$\begin{aligned}
 R_{\Omega|X=x}(\delta_i | x) &= E_{\Omega|X=x} [l(\delta_i | x)] && \text{defn. conditional risk} \\
 &= \sum_{j=1}^s l(\delta_i | \omega_j) P(\omega_j | x) \\
 &= \sum_{j \neq i} P(\omega_j | x) && \text{as } l(\delta_i | \omega_j) = 1 \text{ when } i \neq j \\
 &= \sum_{j=1}^s P(\omega_j | x) - P(\omega_i | x) && \text{as } \pm P(\omega_i | x) = 0 \\
 &= 1 - P(\omega_i | x) && \text{as } \sum_{j=1}^s P(\omega_j | x) = 1
 \end{aligned}$$

## Bayes decision rule

- Minimizing  $R(\delta_i | x) = 1 - P(\omega_i | x)$
- can be done maximizing  $P(\omega_i | x)$  and the Bayes decision rule is:

$$\begin{aligned}
 \delta(x) &= \arg \max_{1 \leq i \leq s} P(\omega_i | x) \\
 &= \arg \max_{1 \leq i \leq s} \frac{P(x | \omega_i) P(\omega_i)}{P(x)} \\
 &= \arg \max_{1 \leq i \leq s} P(x | \omega_i) P(\omega_i)
 \end{aligned}$$

## Bayes decision rule

- Also called:
  - minimum-error-rate decision rule
  - maximum *a-posteriori* probability (MAP) decision rule
  - plug in MAP decision rule

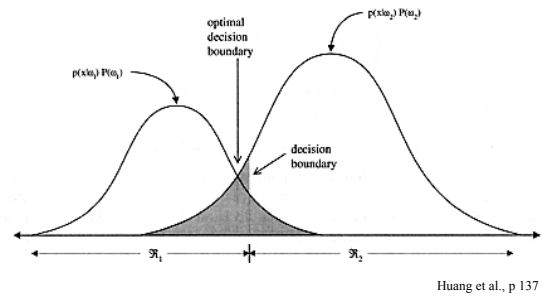
## An illustration

- Suppose there are two classes
- Bayes/MAP/minimum-error-rate decision rule will be:

$$\delta(x) = \begin{cases} 1 & P(x | \omega_1)P(\omega_1) > P(x | \omega_2)P(\omega_2) \\ 2 & P(x | \omega_1)P(\omega_1) < P(x | \omega_2)P(\omega_2) \end{cases}$$

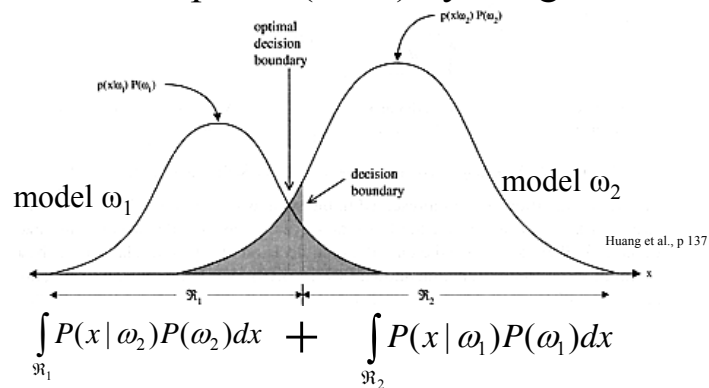
## An illustration

- The optimal decision boundary is where the *a-posteriori* boundaries cross:



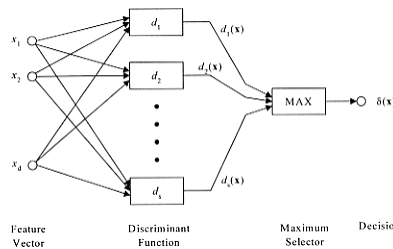
## Probability of error

- We can compute  $P(\text{error})$  by integration:



## Discriminant functions

- A discriminant function  $d_i(x)$  measures the similarity between class  $\omega_i$  and  $x$ .
- $d_i(x)$  constructed for each  $\omega_i$ :



Huang et al., p 139

## Bayes Decision & Discriminant Fns

- Bayes decision rule can be formulated as a discriminant function:

$$\begin{aligned}
 d_i(x) &= -R(\delta_i | x) \\
 &= \frac{\Pr(x | \omega_i)P(\omega_i)}{P(x)} \text{ with the min error rate classifier} \\
 &\propto \Pr(x | \omega_i)P(\omega_i)
 \end{aligned}$$

- Other discriminant functions possible

## Likelihood ratios

- In hypothesis tests where we decide between two classes, likelihood ratios are common:

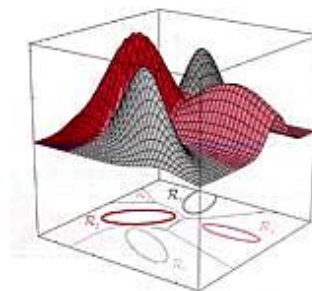
$$\ell(x) = \frac{P(x | \omega_1)}{P(x | \omega_2)} \underset{\omega_2}{\overset{\omega_1}{>}} \frac{P(\omega_2)}{P(\omega_1)}$$

- as are log likelihood ratios:

$$\log \ell(x) = \log P(x | \omega_1) - \log P(x | \omega_2) \underset{\omega_2}{\overset{\omega_1}{>}} \log P(\omega_2) - \log P(\omega_1)$$

## Decision boundaries

- When the two or more discriminants are tied for the maximal value, either decision is valid.
- This induces a partitioning of the feature space. The crossing points are called decision boundaries.



Duda et al, 2001 p. 45

## Building classifiers

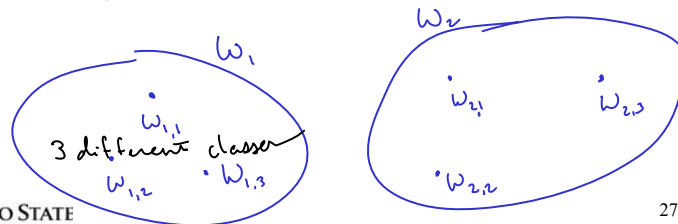
- The prior  $P(\omega_i)$  and class-conditional  $P(x|\omega_i)$  pdfs are rarely known.
- Consequently, we must estimate them:
  - Prior: Reasonably easy.
  - Class-conditional: Much more difficult.

## What type of classifier?

- Supervised vs. unsupervised learning
- Parametric vs. non-parametric

## Vector Quantization (VQ)

- Unsupervised learner – We learn a set of codewords. Each codeword can be seen as a class.
  - Frequently we speak about the class of the codebook, this is different:

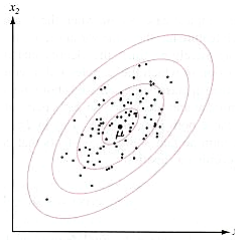


## Other VQ issues

- Distortion measure
  - What's appropriate?
  - Euclidean distortion makes sense if the variance of each component is proportional to the importance of the component in identifying the class.

## Distortion - Mahalanobis

- Mahalanobis distance
  - Accounts for the variance and covariance ( $\Sigma$ )
  - Removes assumption of equal scaling



$$d_{Mahalanobis}(\bar{x}, \bar{z}) = (\bar{x} - \bar{z})^t \Sigma^{-1} (\bar{x} - \bar{z})$$

## Overall average distortion

$$D = E[d(\bar{x}, \bar{z})]$$

examine expectation with respect to each of K code words

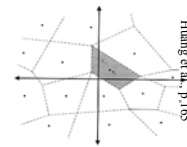
$$= \sum_{i=1}^K p(\bar{x} \in \omega_i) E[d(\bar{x}, \bar{z}_i) | \bar{x} \in \omega_i]$$

expected value of  $\bar{x}$  for each codeword

$$= \sum_{i=1}^K p(\bar{x} \in \omega_i) \int_{\bar{x} \in \omega_i} d(\bar{x}, \bar{z}_i) p(\bar{x} | \bar{x} \in \omega_i) dx$$

$$\text{Let } D_i = p(\bar{x} \in \omega_i) \int_{\bar{x} \in \omega_i} d(\bar{x}, \bar{z}_i) p(\bar{x} | \bar{x} \in \omega_i) dx$$

$$= \sum_{i=1}^K D_i$$



## Min overall average distortion

- Minimizing each partition or *cell*:

$$D_i = p(\bar{x} \in \omega_i) \int_{\bar{x} \in \omega_i} d(\bar{x}, \bar{z}_i) p(\bar{x} | \bar{x} \in \omega_i) dx$$

minimizes the overall average distortion.

- Select  $\bar{z}_i$  such that  $D_i$  is minimized.
- Results in an optimal classifier.

## K-Means Revisited

- Unfortunately, no general solution exists.
- K-Means, or the generalized Lloyd algorithm, produces a reasonable approximation which produces a *local* minimum.

## Selecting initial codewords

- Random selection frequently not good enough.
- The LBG algorithm (Linde, Buzo, Gray) was proposed as an alternative method.

## LBG algorithm

$M = 1$

Find centroid (average) of training data

while  $M < \text{desired size}$

    split each codeword  $w$  into  $w \pm \epsilon$

    Use k-means to determine new codewords

$M = 2M$

end

## VQ summary

- VQ is a non-parametric, unsupervised classifier
- It is possible to use *multiple* VQ codebooks to build a supervised learning VQ recognizer.

## Gaussian classifiers

- Multidimensional Gaussian

$$f_i(\bar{x}) = \frac{1}{(2\pi)^{d/2} |\Sigma_i|^{1/2}} e^{\left( \frac{-1}{2} (\bar{x} - \mu_i)^t \Sigma_i^{-1} (\bar{x} - \mu_i) \right)}$$

where  $d$  = # dimensions,  $i$  = classifier index

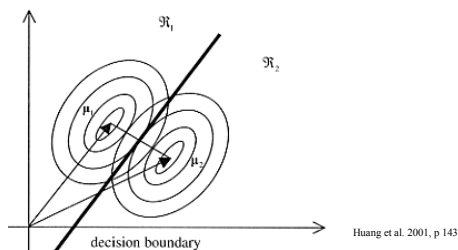
- Maximum likelihood estimators are simply sample mean & variance

## Gaussian discriminant functions

$$\begin{aligned}d_i(\bar{x}) &= \log(\Pr(\bar{x} | \omega_i)P(\omega_i)) \\ &= \log\left(\frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} e^{\left(-\frac{1}{2}(\bar{x}-\mu)^t \Sigma_i^{-1}(\bar{x}-\mu)\right)}\right) + \log(P(\omega_i)) \\ &= -\frac{d}{2}\log(2\pi) - \frac{1}{2}\log(|\Sigma_i|) - \frac{1}{2}(\bar{x}-\mu)^t \Sigma_i^{-1}(\bar{x}-\mu) + \log(P(\omega_i))\end{aligned}$$

## Gaussian decision boundaries

- In a uniform prior two class system:



- In non-uniform priors, boundary shifts towards less likely class.

## Estimating error rate

- Empirical error rate of N trials

$$\text{error rate} = \frac{\# \text{errors}}{N}$$

- A confidence interval would be desirable
  - The number of errors has a binomial distribution
  - Confidence intervals for binomials are difficult
  - Normal distribution approximation can be used when  $N > 50$

## Normal test

- Let
  - $p$  = error rate
  - $q$  = accuracy =  $1-p$
  - $1-\alpha$  = desired confidence interval (i.e. .95)
- Inverse cdf of  $(1-\alpha/2)$  of a normal distribution with variance  $n/(pq)$  gives the distance from  $p$  for a  $1-\alpha$  confidence interval.

## Example

- $n=100, p = .14, q = .86$
- Select 95% confidence interval
- $\alpha=1-.95=.05$
- $\sigma^2 = pq/n = .001204$
- $\text{cdf}^{-1}(1 - \alpha/2)=1.96$
- confidence interval:  
 $.14 \pm 1.96 \sigma = (.072, .208)$

## Normal test in Matlab

- With the statistics toolbox, `norminv(x,  $\mu$ ,  $\sigma$ )` returns the inverse cdf for a normal  $N(\mu, \sigma^2)$  for  $x\%$  of the probability.
- To compute the confidence interval:  
`p + [-1 1] .* norminv(1- $\alpha$ /2, 0, sqrt(p*q/n))`

## Train/test partitions

- Problem: Providing enough training data may reduce the test set size to that of impracticality.
- Solution:
  - Collect more data (expensive)
  - N-fold cross validation

## N-fold cross validation

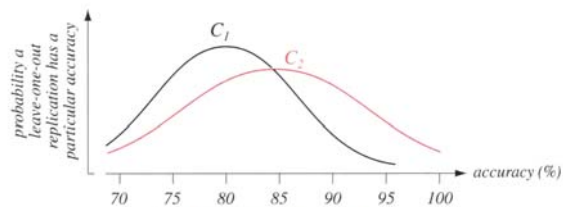
- Split the data into N partitions
- Use N-1 partitions for training and 1 partition for testing.
- This can be done N times.
- Permits all of the data to be used for testing.

## Jackknife testing

- N-fold cross validation to the extreme
- Given D data points, use D-1 for training and test on one point.
- D different classifiers can be constructed in this manner

## Jackknife testing

- Useful for comparing classifier performance.



Duda et al. 2001, p. 486

- Can use hypothesis testing to determine if performance different with N% confidence

## The *curse* of dimensionality

- It can be shown that adding more independent features will reduce error rate.
- *Caveat*: You must have more data to estimate the additional parameters

## Why more data?

- If a system is not overdetermined, a high order fit is not necessarily the right one.
- 1<sup>st</sup>, 2<sup>nd</sup>, and 10<sup>th</sup> order polynomial fits of 10 data points:

