"Statistical models" for ASR

So far, we've seen "min. dist" classific^n.

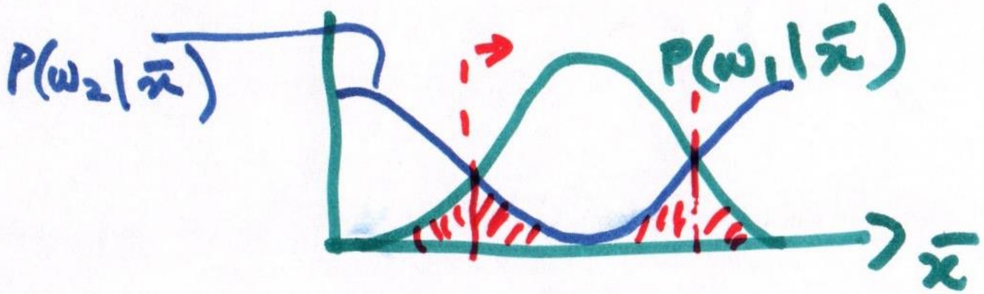
Instead, we consider now minimising P(error)

$\Rightarrow$ min. classific. error

For a test vector $\bar{x}$ belonging to one of the classes $\{\omega_i\}$. Bayes' Decision rule $\Rightarrow$

choose the class that maximises $P(\omega_i | \bar{x})$

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



CDEEP
IIT Bombay

By Bayes Theorem :

$$P(\omega_i \mid \bar{x}) = \frac{P(\bar{x} \mid \omega_i)\, P(\omega_i)}{P(\bar{x})}$$

$P(\bar{x}) = \sum_{j=1}^{N} P(\bar{x} \mid \omega_j)\, P(\omega_j)$ N = # classes

$P(\omega_i)$ = Prior probab (of class ω_i)

$P(\omega_i \mid \bar{x})$ = Posterior probab (of class ω_i given obs. $\bar{x}$)

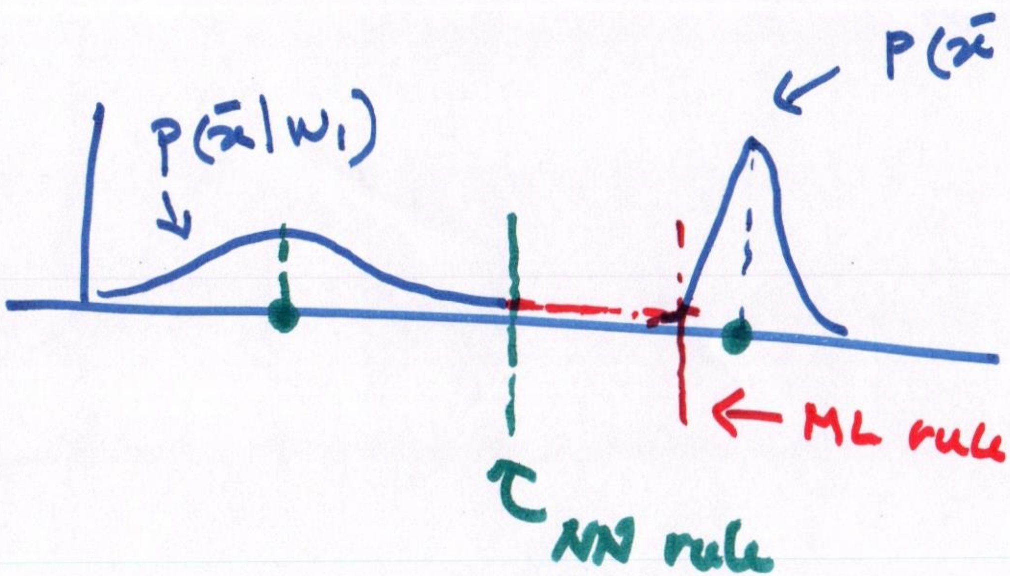
$P(\bar{x} \mid \omega_i)$ = Likelihood (cond. probab. of $\bar{x}$ given ω_i)

$P(\bar{x}) \equiv$ normalis^n const., does not affect decision.

Bayes Decision Rule is therefore MAP (max. a posteriori)

MAP reduces ML when $P(\omega_i)$ = const.

Ex. using ML ~~rule~~ rule → maximise $P(\bar{x}|w_i)$



Classification of a single spectral vector

One observ^n frame at time "t"

Assume that the possible phoneme hypotheses are

λ_i $\quad\quad \lambda_1 = /i/ \quad , \quad \lambda_2 = /a/$

Then, "best" hypothesis according to the MAP rule :

$$\hat{\lambda} = \arg\max_{\lambda_i} p(\lambda_i | O_t)$$

We have $p(\lambda_i | O_t) = \dfrac{p(O_t | \lambda_i) \cdot p(\lambda_i)}{p(O_t)}$

$$\hat{\lambda} = \arg\max_{\lambda_i} p(O_t | \lambda_i) \cdot \underbrace{p(\lambda_i)}_{\text{Lang. model}}$$

Gaussian probability models

$$p(O \mid \lambda_i) = \mathcal{N}(O; \mu_i; \Sigma_i)$$

L-dim spectral vector

$$= \frac{1}{\sqrt{(2\pi)^L \, |\Sigma_i|}} \exp\left[-\frac{1}{2} (O-\mu_i)^T \Sigma_i^{-1} (O-\mu_i) \right]$$

where

$$\mu_i = \frac{1}{N} \sum_{n=1}^{N} O_n = \mathcal{E}\{O_n\}$$

N = # training vectors

$$\Sigma_i = \mathcal{E}\{(O_n - \mu_i)(O_n - \mu_i)^T\}$$

$$= \frac{1}{N} \sum_{n=1}^{N} (O_n - \mu_i)(O_n - \mu_i)^T$$

In computations, we use "negative log likelihood"

$-\log(p(O \mid \lambda_i))$

$= \frac{1}{2}\log((2\pi)^L |\Sigma_i|) + 0.5(O-\mu_i)^T \Sigma_i^{-1}(O-\mu_i)$

Minimizing the negative log-likelihood $\equiv$ MLE

If all class covariances (Σ_i) are assumed to be equal, we have: minimise $(O-\mu_i)^T \Sigma^{-1}(O-\mu_i)$ Mahalanobis dist

Further if $\Sigma = I$,

$\Rightarrow$ minimise $(O-\mu_i)^T(O-\mu_i) \Leftarrow$ E.D.!

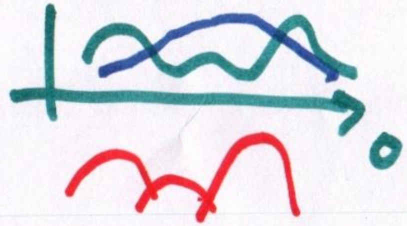
Gaussian Mixture models

j^{th} class

k^{th} Gau of j^{th} class

$$p(o|\lambda_j) = \sum_{k=1}^{K} p(o|G_{jk}) \cdot p(G_{jk}|\lambda_j)$$

$$= \sum_{k=1}^{K} c_{jk} \, N(o; \mu_{jk}; \Sigma_{jk})$$



For large enough K, a GMM can represent any cont. prob. distrib^n with arbitrarily good precision.

The mixture distrib^n parameters for class j $\equiv \{c_{jk}; \mu_{jk}; \Sigma_{jk}\}$ are "trained" on labeled data of j.

Training a GMM via EM algo

EM attempts to estimate the params of the distrib^n (model) that maximize the $\log(P(X|\underbrace{\theta}_{\text{params of the model}}))$

X: all the training data

where $P(X|\theta) = \prod_{n=1}^{N} p(x_n|\theta)$, $X = \{x_1, x_2, \ldots, x_N\}$

Steps:

Initialization: Set K cluster parameters $\{\mu_k, \sigma_k^2, c_k\}$ by guessing.

<u>Assignment</u> step : take x_n

$$p_{kn} = \frac{p(x_n | k) \cdot P(k)}{p(x_n)}$$

$\equiv$ "degree of belonging" to cluster k

<u>Update</u> step : Adjust params $\{\mu_k, \sigma_k^2, c_k\}$ based on the data assignments

$$\hat{c}_k = \frac{1}{N} \sum_{n=1}^{N} p_{kn} \; ; \quad \hat{\mu}_k = \frac{\sum_N p_{kn} x_n}{\sum_N p_{kn}} \, , \text{ etc.}$$

<u>Repeat</u>