Lectures 4 and 5: ML and Bayesian Learning, Density Estimation, Gaussian Mixtures and EM and Variational Bayesian Inference, Performance Assessment of Classifiers

Prof. Krishna R. Pattipati
Dept. of Electrical and Computer Engineering
University of Connecticut
Contact: krishna@engr.uconn.edu (860) 486-2890

Fall 2018
October 1, 8 & 15, 2018

Reading List

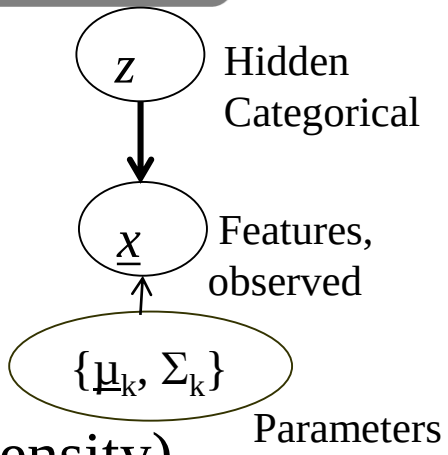
- Duda, Hart and Stork, Sections 3.1-3.6, Chapter 4
- Bishop, Section 2.5, Chapter 9, Sections 10.1 , 10.2
- Murphy, Chapter 4, Chapter 11, Section 21.6
- Theodoridis, Chapter 7, Chapter 12

Lecture Outline

- Estimating Parameters of Densities From Data
  - Maximum Likelihood Methods
  - Bayesian Learning
- Estimating Probability Densities (Nonparametric)
  - Histogram Methods
  - Parzen Windows
  - Probabilistic Neural Network
  - *k*-nearest Neighbor Approach
- Mixture Models and EM
- Performance Assessment of Classifiers

Recall Bayesian Classifiers

- Need to know $\{P(z = j), p(\underline{x} \mid z = j), \lambda_{ij}\}$



- We usually estimate them from data
  - Parametric Methods (assume a form for density)
  - Nonparametric Methods
    - Estimate density via Parzen windows, PNN, RCE,NN, k-NN,..
  - Mixture Models
    - Mixture of Gaussians, Multinoullis, Multinomials, student,…

Estimating Parameters of Densities from Data

- Estimating parameters of densities from data:

$$\boxed{\{P(z=k)\}_{k=1}^{C} \quad \{p(\underline{x},\underline{\theta}\,|\,z=k)\}_{k=1}^{C}}$$

For example, in the Gaussian case

$\underline{\theta}=\{(\underline{\mu}_k,\Sigma_k)\}$ **General Case or** $(\{\underline{\mu}_k\},\Sigma)$ **Hyperellipsoid Case or** $(\{\underline{\mu}_k\},\sigma^2 I_p)$ **Hypersphere case**

Data: $D=\{\underline{x}_k^1\ \underline{x}_k^2\ \underline{x}_k^3 \ldots\ldots\ldots \underline{x}_k^{n_k} :\ k=1,2,3,\ \ldots C\}$

n_k samples from class k. Let $\sum_{k=1}^{C} n_k = N$

Assuming samples are independent,

$$L(\underline{\theta}) = p(D|\underline{\theta}) = \prod_{k=1}^{C}\prod_{j=1}^{n_k} p(\underline{x}_k^j \,|\, z=k,\underline{\theta})P(z=k)$$

$$l(\underline{\theta}) = \ln L(\underline{\theta}) = \ln\, p(D\,|\,\underline{\theta})$$

$$= \sum_{k=1}^{C}\sum_{j=1}^{n_k} \ln\, p(\underline{x}_k^j \,|\, z=k,\underline{\theta}) + \sum_{k=1}^{c} n_k \ln \pi_k ; P(z=k)=\pi_k$$

Optimal π_k

- Optimal ML Estimate of π_k : $\hat{\pi}_k$

$$\max \sum_{k=1}^{C} n_k \ln \pi_k \qquad \text{s.t.} \quad \sum_{k=1}^{C} \pi_k = 1$$

Recall:
ln *x* is concave
-ln *x* is convex

$$Lagrangian: \max_{\{\pi_k\}} [\sum_{k=1}^{C} n_k \ln \pi_k + \lambda(\sum_{k=1}^{C} \pi_k - 1)]$$

$$\frac{n_k}{\hat{\pi}_k} = -\lambda \quad \Rightarrow \quad \hat{\pi}_k = -\frac{n_k}{\lambda} \quad \Rightarrow \quad \lambda = -N$$

$$\hat{\pi}_k = \frac{n_k}{N}$$

Fraction of samples of class *k*.
Intuitively appealing.

- What if some classes did not have samples in training data?
  - **Zero count** or **sparse data** or **black swan problem**

$$\hat{\pi}_k = \frac{n_k + 1}{N + C}$$

Laplace rule of succession or add-one smoothing

Optimal $\underline{\theta}$: Hyperellipsoid Case

- Optimal ML estimate of $\underline{\theta}$: (when covariance matrices for all classes are taken to be equal, i.e., $\Sigma_i = \Sigma$)

$$l(\underline{\theta}) = \sum_{k=1}^{C} \sum_{j=1}^{n_k} \ln\, p(\underline{x}_k^j \mid z = k, \underline{\theta}) \qquad \underline{\theta} = \left\{ \{\underline{\mu}_k\}, \Sigma \right\}$$

$$\nabla_{\underline{\theta}} \left(\underline{\theta}^T A \underline{\theta} \right) = \nabla_{\underline{\theta}} \left(tr\left[A \underline{\theta} \underline{\theta}^T \right] \right) = 2A\underline{\theta}$$

$$l(\{\underline{\mu}_k\}_{k=1}^{C}, \Sigma) = -\frac{N}{2} \ln|\Sigma| - \frac{1}{2} \sum_{k=1}^{C} \sum_{j=1}^{n_k} (\underline{x}_k^j - \underline{\mu}_k)^T \Sigma^{-1} (\underline{x}_k^j - \underline{\mu}_k)$$

$$\nabla_{\underline{\mu}_k} l = \underline{0} \quad \Rightarrow \quad \sum_{j=1}^{n_k} \hat{\Sigma}^{-1} (\underline{x}_k^j - \hat{\underline{\mu}}_k) = 0 \Rightarrow \hat{\underline{\mu}}_k = \frac{1}{n_k} \sum_{j=1}^{n_k} \underline{x}_k^j ; k = 1,2,..,C$$

ML estimate for mean is just the sample mean!

Optimal Σ

$\nabla_{\Sigma} l = \underline{0}$

we know, $\nabla_{\Sigma}\left[\ln|\Sigma|\right] = \Sigma^{-1}$

Aside : *A is PD*

$$|A| = \exp(\ln|A|) = \exp(\sum_{i=1}^{n} \ln \lambda_i) = \exp(trace[\ln A])$$

$$\Rightarrow \ln|A| = trace[\ln A]$$

$$\nabla_A\left[\ln|A|\right] = A^{-1}$$

$$\nabla_{\Sigma} l = -\frac{1}{2} N\hat{\Sigma}^{-1} + \frac{1}{2}\hat{\Sigma}^{-1}\left[\sum_{k=1}^{C}\sum_{j=1}^{n_k}(\underline{x}_k^j - \underline{\hat{\mu}}_k)(\underline{x}_k^j - \underline{\hat{\mu}}_k)^T\right]\hat{\Sigma}^{-1} = \underline{0}$$

$$\hat{\Sigma} = \frac{1}{N}\sum_{k=1}^{C}\sum_{j=1}^{n_k}(\underline{x}_k^j - \underline{\hat{\mu}}_k)(\underline{x}_k^j - \underline{\hat{\mu}}_k)^T$$

ML estimate of covariance Matrix is the arithmetic average of $(\underline{x}_k^j - \underline{\hat{\mu}}_k)(\underline{x}_k^j - \underline{\hat{\mu}}_k)^T$ over all classes

We can show that, $E\left[\hat{\Sigma}\right] = \frac{N-C}{N}\Sigma$ so use

$$\hat{\Sigma} = \frac{1}{N-C}\sum_{k=1}^{C}\sum_{j=1}^{n_k}(\underline{x}_k^j - \underline{\hat{\mu}}_k)(\underline{x}_k^j - \underline{\hat{\mu}}_k)^T$$

$$\nabla_A\left[\ln|A|\right] = A^{-1}$$

Proof of $E[\hat{\Sigma}] = \frac{N-C}{N}\Sigma$

$$\hat{\Sigma} = \frac{1}{N}\sum_{k=1}^{C}\sum_{j=1}^{n_k}(\underline{x}_k^j - \underline{\hat{\mu}}_k)(\underline{x}_k^j - \underline{\hat{\mu}}_k)^T \quad know\ n_k\,\underline{\hat{\mu}}_k = \sum_{j=1}^{n_k}\underline{x}_k^j$$

$$= \frac{1}{N}\sum_{k=1}^{C}\sum_{j=1}^{n_k}[\underline{x}_k^j\,\underline{x}_k^{jT} - \underline{x}_k^j\,\underline{\hat{\mu}}_k^{\ T} - \underline{\hat{\mu}}_k\,\underline{x}_k^{jT} + \underline{\hat{\mu}}_k\,\underline{\hat{\mu}}_k^{\ T}]$$

$$= \frac{1}{N}\sum_{k=1}^{C}\sum_{j=1}^{n_k}[\underline{x}_k^j\,\underline{x}_k^{jT} - \underline{\hat{\mu}}_k\,\underline{\hat{\mu}}_k^{\ T}]$$

$$E(\hat{\Sigma}) = \Sigma + \frac{1}{N}\sum_{k=1}^{C} n_k\,\underline{\mu}_k\,\underline{\mu}_k^{\ T} - \frac{1}{N}E\{\sum_{k=1}^{C}\frac{n_k}{n_k^2}\sum_{q=1}^{n_k}\sum_{r=1}^{n_k}\underline{x}_k^q\,\underline{x}_k^{r\,T}\}$$

$$= \Sigma + \frac{1}{N}\sum_{k=1}^{C} n_k\,\underline{\mu}_k\,\underline{\mu}_k^{\ T} - \frac{1}{N}\sum_{k=1}^{C} n_k\,\underline{\mu}_k\,\underline{\mu}_k^{\ T} - \frac{C}{N}\Sigma$$

$$= (\frac{N-C}{N})\Sigma$$

$$\boxed{E\left[\hat{\Sigma}\right] = \frac{N-C}{N}\Sigma}$$

Shrinkage Methods

- *Linear discriminants may outperform quadratic discriminants when training data is small*
- *Shrink the covariance matrices towards common value $\Rightarrow$ possibly biased, but results in a less variable estimator*

$$\hat{\Sigma}_k(\alpha) = \frac{(1-\alpha)n_k\hat{\Sigma}_k + \alpha N\hat{\Sigma}}{(1-\alpha)n_k + \alpha N}$$

(convex combination of $\hat{\Sigma}_k$ and $\hat{\Sigma}$)

$$\hat{\Sigma}(\gamma) = (1-\gamma)\hat{\Sigma} + \gamma I; \quad 0 < \gamma < 1$$

If variables are scaled to have zero mean and unit variance

$$\hat{\Sigma}(\gamma) = \gamma\, diag(\hat{\Sigma}) + (1-\gamma)\hat{\Sigma}$$

This has Bayesian (MAP) interpretation

$$\hat{\Sigma}_k(\alpha,\gamma) = (1-\gamma)\hat{\Sigma}_k(\alpha) + \frac{\gamma}{p} tr(\hat{\Sigma}_k(\alpha))I$$

Regularized Discriminant Analysis

Select α and γ to minimize error rate via cross validation

ML-based Discriminants

- ML-estimated Best Linear Rule ($C+Cp+p(p+1)/2$ parameters)

$$j = \arg \max_{k \in \{1,2,\ldots,C\}} \left\{ \hat{\underline{\mu}}_k^T \hat{\Sigma}^{-1} \underline{x} - [\frac{1}{2} \hat{\underline{\mu}}_k^T \hat{\Sigma}^{-1} \hat{\underline{\mu}}_k - \ln \hat{\pi}_k] \right\}$$

- Unequal Covariance Case

$$\hat{\underline{\mu}}_k = \frac{1}{n_k} \sum_{j=1}^{n_k} \underline{x}_k^j$$

$$\hat{\Sigma}_k = \frac{1}{n_k - 1} \sum_{j=1}^{n_k} (\underline{x}_k^j - \hat{\underline{\mu}}_k)(\underline{x}_k^j - \hat{\underline{\mu}}_k)^T$$

Note that we need $n_k \geq p$ for each class. If not, use shrinkage methods.

- ML-estimated quadratic rule ($C+Cp+Cp(p+1)/2$ parameters)

$$j = \arg \max_{k \in \{1,2,..,C\}} \left\{ -\frac{1}{2} \ln |\hat{\Sigma}_k| - \frac{1}{2} (\underline{x} - \hat{\underline{\mu}}_k)^T \hat{\Sigma}_k^{-1} (\underline{x} - \hat{\underline{\mu}}_k) + \ln \hat{\pi}_k \right\}$$

As $N \to \infty$, they approach the performance of optimal Bayesian Classifier. However, for finite N, a linear rule may outperform a quadratic one!

Recursive Estimation of Parameters-1

- What if "big streaming data"? **Recursive estimation of mean**

$$\hat{\underline{\mu}}_k^n = (1-\frac{1}{n})\hat{\underline{\mu}}_k^{n-1} + \frac{1}{n}\underline{x}_k^n$$

- In general, we can use SA algorithm:

$$\frac{1}{n_k}\frac{\partial}{\partial \underline{\theta}}\sum_{j=1}^{n_k}\ln \mathrm{p}(\underline{x}_k^j \mid z=k,\underline{\theta}) = 0$$

$$\lim_{n_k\to\infty} \Rightarrow E\left\{\frac{\partial}{\partial \underline{\theta}}\ln \mathrm{p}(\underline{x}_k^j \mid z=k,\underline{\theta})\right\} = 0$$

$$\hat{\underline{\mu}}_k^n = \hat{\underline{\mu}}_k^{n-1} + \alpha_n \frac{\partial}{\partial \underline{\theta}}\ln \mathrm{p}(\underline{x}_k^n \mid z=k,\underline{\theta})\Big|_{\hat{\underline{\mu}}_k^{n-1}}$$

$$= \hat{\underline{\mu}}_k^{n-1} + \alpha_n \hat{\Sigma}^{-1}(\underline{x}_k^n - \hat{\underline{\mu}}_k^{n-1})$$

$\alpha_n = \frac{1}{n}, \frac{1}{(n+m)}, \alpha_0 \frac{(n/K+1)}{(n/K)^2+1}$ *satisfy SA conditions*

Idea of Stochastic Approximation:

$f(\theta) = E[g \mid \theta]$

roots of $f(\theta) = u$

assume $E[(g-f)^2|\theta] < \infty$

$$\theta_{n+1} = \theta_n + \alpha_n[u - g(\theta_n)]$$

$$\lim_{n\to\infty}\alpha_n = 0$$

$$\sum_{n=1}^{\infty}\alpha_n = \infty \qquad \sum_{n=1}^{\infty}\alpha_n{}^2 < \infty$$

Recursive Estimation of Parameters-2

- Unequal Covariance Case

$$\hat{\underline{\mu}}_k = \frac{1}{n_k}\sum_{j=1}^{n_k} \underline{x}_k^j \qquad \hat{\Sigma}_k = \frac{1}{n_k - 1}\sum_{j=1}^{n_k} (\underline{x}_k^j - \hat{\underline{\mu}}_k)(\underline{x}_k^j - \hat{\underline{\mu}}_k)^T$$

Class k

- Covariance Recursion for Class k: $\alpha_n = \frac{1}{n}$

$$\hat{\Sigma}_k^n = \frac{1}{n-1}\sum_{j=1}^{n} (\underline{x}_k^j - \hat{\underline{\mu}}_k^n)(\underline{x}_k^j - \hat{\underline{\mu}}_k^n)^T \qquad \hat{\underline{\mu}}_k^n = \hat{\underline{\mu}}_k^{n-1} + \frac{1}{n}(\underline{x}_k^n - \hat{\underline{\mu}}_k^{n-1})$$

$$= (\frac{n-2}{n-1})\frac{1}{n-2}\{\sum_{j=1}^{n} (\underline{x}_k^j - \hat{\underline{\mu}}_k^{n-1} + \hat{\underline{\mu}}_k^{n-1} - \hat{\underline{\mu}}_k^n)(\underline{x}_k^j - \hat{\underline{\mu}}_k^{n-1} + \hat{\underline{\mu}}_k^{n-1} - \hat{\underline{\mu}}_k^n)^T\}$$

$$= (\frac{n-2}{n-1})\hat{\Sigma}_k^{n-1} + (\frac{1}{n-1})(1 - \frac{1}{n} - \frac{1}{n} + \frac{n}{n^2})(\underline{x}_k^n - \hat{\underline{\mu}}_k^{n-1})(\underline{x}_k^n - \hat{\underline{\mu}}_k^{n-1})^T$$

$$= (\frac{n-2}{n-1})\hat{\Sigma}_k^{n-1} + \frac{1}{n}(\underline{x}_k^n - \hat{\underline{\mu}}_k^{n-1})(\underline{x}_k^n - \hat{\underline{\mu}}_k^{n-1})^T$$

Recursive Estimation of Parameters-3

- Covariance Recursion in Hyperellipsoid Case
  - Update means via

$$\underline{\hat{\mu}}_i^{n_i} = (1-\frac{1}{n})\underline{\hat{\mu}}_i^{n_i-1} + \frac{1}{n}\underline{x}_i^{n_i}; i = 1,2,..,C$$

- For covariance, suppose n^{th} sample is from class l. Then

$$\hat{\Sigma}^n = \frac{1}{n-C}\{\sum_{k=1}^{C}\sum_{j=1}^{n_k}(\underline{x}_k^j - \underline{\hat{\mu}}_k^{n_k})(\underline{x}_k^j - \underline{\hat{\mu}}_k^{n_k})^T\}$$

$$= (\frac{n-C-1}{n-C})\frac{1}{n-C-1}\{[\sum_{\substack{k=1 \\ k\neq l}}^{C}\sum_{j=1}^{n_k}(\underline{x}_k^j - \underline{\hat{\mu}}_k^{n_k})(\underline{x}_k^j - \underline{\hat{\mu}}_k^{n_k})^T] + \sum_{j=1}^{n_l}(\underline{x}_l^j - \underline{\hat{\mu}}_l^{n_l})(\underline{x}_l^j - \underline{\hat{\mu}}_l^{n_l})^T\}$$

$$= (\frac{n-C-1}{n-C})\hat{\Sigma}^{n-1} + (\frac{1}{n-C})(1-\frac{1}{n_l}-\frac{1}{n_l}+\frac{n_l}{n_l^2})(\underline{x}_l^{n_l} - \underline{\hat{\mu}}_l^{n_l-1})(\underline{x}_l^{n_l} - \underline{\hat{\mu}}_l^{n_l-1})^T$$

$$= (\frac{n-C-1}{n-C})\hat{\Sigma}^{n-1} + \frac{(n_l-1)}{(n-C)n_l}(\underline{x}_l^{n_l} - \underline{\hat{\mu}}_l^{n_l-1})(\underline{x}_l^{n_l} - \underline{\hat{\mu}}_l^{n_l-1})^T$$

Bayesian Learning

- Posterior probability $P(z=i \mid \underline{x})$ is the key
- Have data: $D = \left\{ \underline{x}_k^1 \quad \underline{x}_k^2 \quad \underline{x}_k^3 ..... \quad \underline{x}_k^{n_k} : \quad k = 1, 2, 3, \ . \ . \ . \ . \ C \right\} = \{D_1, D_2, ..., D_C\}$

  $\Rightarrow$ we can only get $P(z=k \mid \underline{x}, D_k)$

$$P(z=k \mid \underline{x}, D_k) = \frac{p(\underline{x} \mid z=k, D_k) P(z=k)}{\sum_{i=1}^{C} p(\underline{x} \mid z=i, D_i) P(z=i)}$$

- Class conditional density and posterior density of parameters

$$p(\underline{x} \mid z=k, D_k) = \int_{\underline{\theta}} p(\underline{x} \mid \underline{\theta}, z=k)\, p(\underline{\theta} \mid z=k, D_k)\, d\underline{\theta}$$

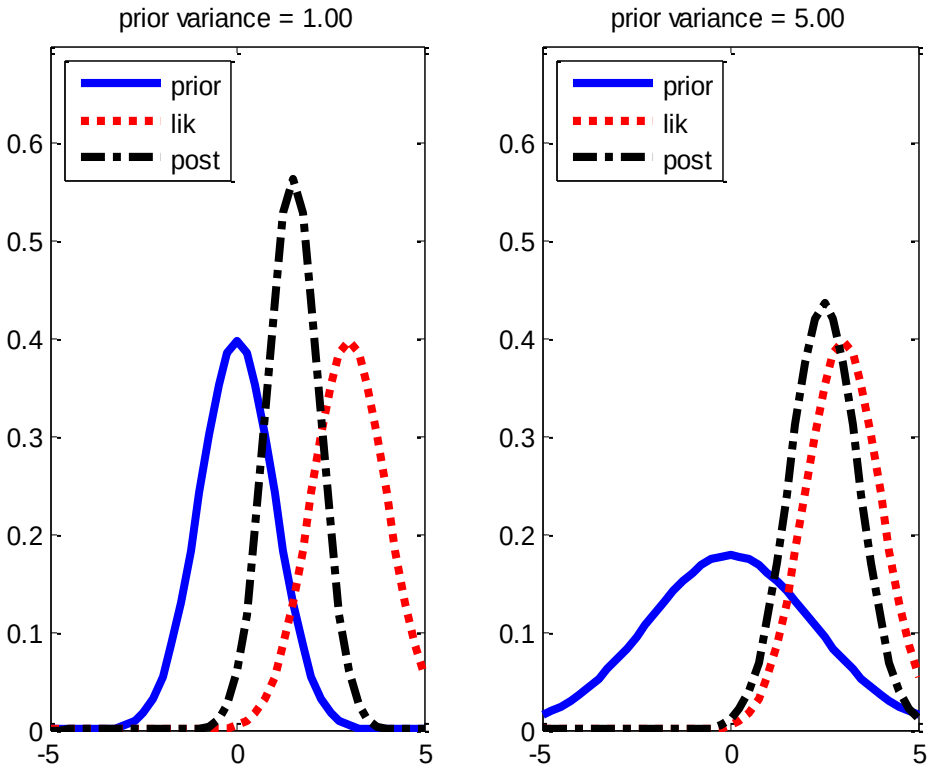
Tough to compute unless reproducing density. Need Simulation.

$$p(\underline{\theta} \mid z=k, D_k) = \frac{p(D_k \mid z=k, \underline{\theta})\, p(\underline{\theta} \mid z=k)}{\int_{\underline{\theta}} p(D_k \mid z=k, \underline{\theta})\, p(\underline{\theta} \mid z=k)\, d\underline{\theta}}$$

$$= \frac{\prod_{j=1}^{n_k} p(\underline{x}_k^j \mid z=k, \underline{\theta})\, p(\underline{\theta} \mid z=k)}{\int_{\underline{\theta}} \prod_{j=1}^{n_k} p(\underline{x}_k^j \mid z=k, \underline{\theta})\, p(\underline{\theta} \mid z=k)\, d\underline{\theta}}$$

If $p(D_k \mid z=k, \underline{\theta})$ has sharp peak at $\hat{\underline{\theta}}$ then so does $p(\underline{\theta} \mid z=k, D_k)$

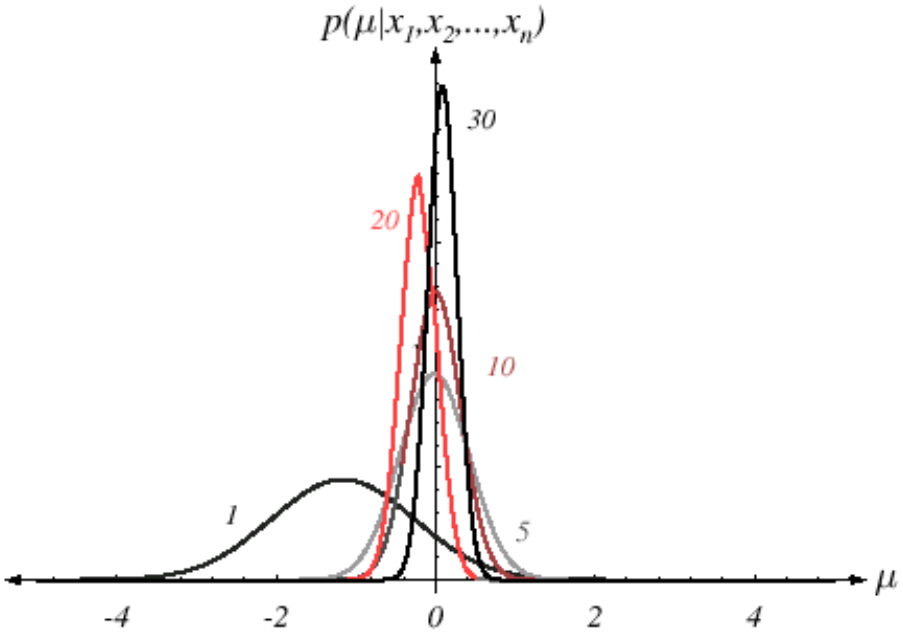
Illustration of Bayesian Learning - 1

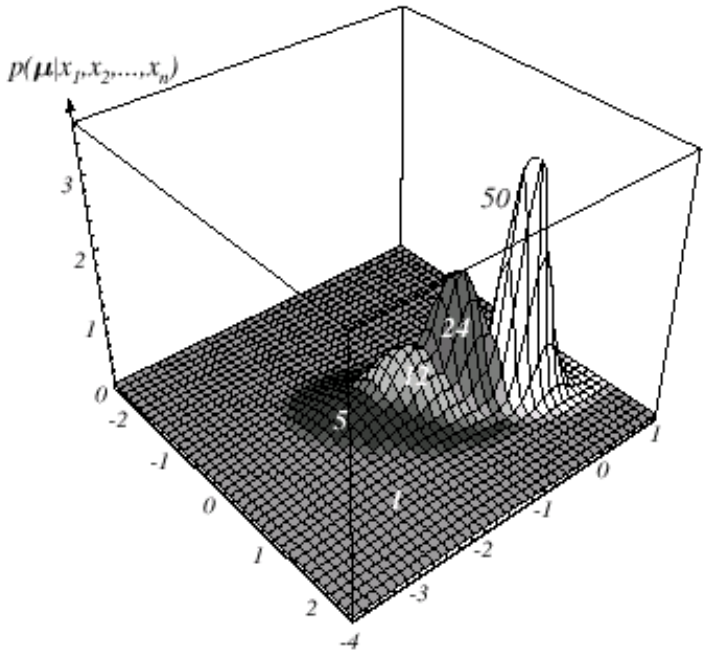


gaussInferParamsMean1d from Murphy, Page 121

Bayesian learning of the mean of Gaussian distributions in one dimension.
Strong prior (small variance) $\Rightarrow$ posterior mean "shrinks" towards the prior mean.
Weak prior (large variance) $\Rightarrow$ posterior mean is similar to the MLE

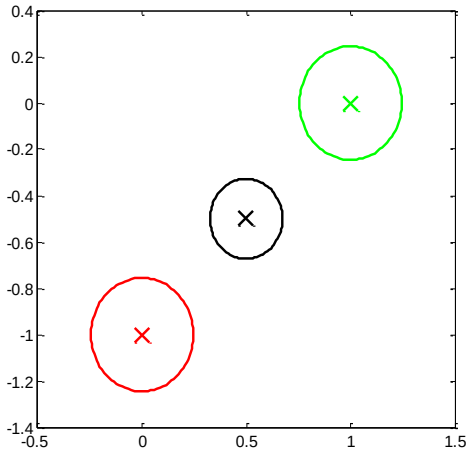
Illustration of Bayesian Learning - 2



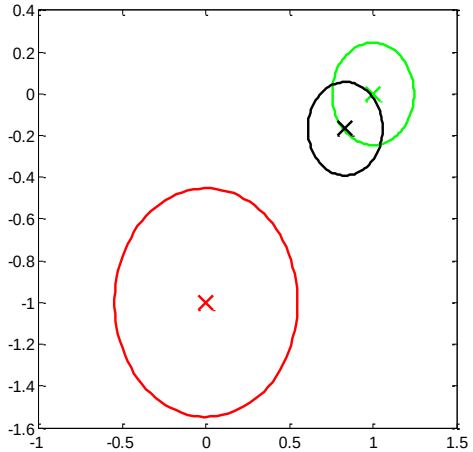


Bayesian learning of the mean of Gaussian distributions in one and two dimensions. As the number of samples increase, the posterior density peaks at the true value.

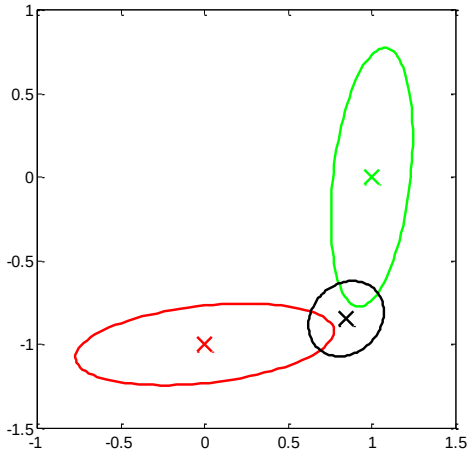
Sensor Fusion



Equally reliable Sensors (R,G)
Fused Estimate: Black



G is more reliable than R
Fused Estimate: Black



R is more reliable in *y*
G is more reliable in *x*
Fused Estimate: Black

sensorFusion2d from Murphy, Page 123

Bayesian sensor fusion appropriately combines measurements from multiple sensors based on uncertainty of the sensor measurements. Larger uncertainty $\Rightarrow$ less weight.

Recursive Bayesian Learning

- Likelihood Recursion

$$Let\ D_k^{n_k} = \{\underline{x}_k^1, \underline{x}_k^2, ..., \underline{x}_k^{n_k-1}, \underline{x}_k^{n_k}\} = \{D_k^{n_k-1}, \underline{x}_k^{n_k}\}$$

$$p(D_k^{n_k} \mid z=k, \underline{\theta}) = p(\underline{x}_k^{n_k} \mid z=k, \underline{\theta})\, p(D_k^{n_k-1} \mid z=k, \underline{\theta})$$

- Recursion for Posterior Density

$$p(\underline{\theta} \mid z=k, D_k^{n_k}) = \frac{p(\underline{x}_k^{n_k} \mid z=k, \underline{\theta})\, p(\underline{\theta} \mid z=k, D_k^{n_k-1})}{\int p(\underline{x}_k^{n_k} \mid z=k, \underline{\theta})\, p(\underline{\theta} \mid z=k, D_k^{n_k-1}) d\underline{\theta}}$$

$$where\ p(\underline{\theta} \mid z=k, D_k^0) = p(\underline{\theta} \mid z=k)$$

Problem: Need to store all training samples $D_k^{n_k-1}$ to calculate $p(\underline{\theta} \mid z=k, D_k^{n_k-1})$.

For exponential family (e.g., Gaussian, exponential, Rayleigh, Gamma, Beta, Poisson, Bernoulli, Binomial, Multinomial) need only few parameters to characterize $p(\underline{\theta} \mid z=k, D_k^{n_k-1})$. They are called *sufficient statistics*.

Recursive Learning of Mean and Covariance

- Want to learn both the mean vector $\underline{\mu}$ & covariance Σ

$$p(\underline{x}^n \mid \underline{\mu}, \Sigma_v) = N(\underline{\mu}, \Sigma_v) \text{ and } p(\underline{\mu}, \Sigma_v) = \overbrace{\underbrace{p(\underline{\mu} \mid \underline{m}^0, \frac{1}{k^0}\Sigma_v)}_{Gaussian}\, \underbrace{p(\Sigma_v \mid \Sigma^0, \nu^0)}_{inverse-Wishart}}^{NIW\,(Normal-Inverse-Wishart)}$$

$$p(\underline{\mu} \mid m^0, \frac{1}{k^0}\Sigma_v) = N(\underline{\mu}; \underline{m}^0, \frac{1}{k^0}\Sigma_v)$$

Wishart is a generalization of Gamma and chi-squared

$$IW(\Sigma_v \mid \Sigma^0, \nu^0) = \frac{|\Sigma^0|^{\frac{\nu^0}{2}}}{2^{\frac{\nu^0 p}{2}}\Gamma_p(\frac{\nu^0}{2})} |\Sigma_v|^{-\frac{\nu^0+p+1}{2}} e^{-\frac{1}{2}tr(\Sigma^0\Sigma_v^{-1})}; \Gamma_p(\frac{\nu^0}{2}) = \prod_{i=1}^{p}\Gamma(\frac{\nu^0+1-i}{2})$$

$$Given\ D^n = \{\underline{x}^1, \underline{x}^2, ..., \underline{x}^n\},\ p(\underline{\mu}, \Sigma_v \mid D^n) = NIW(\underline{\mu}, \Sigma_v \mid \underline{m}^n, k^n, \nu^n, \Sigma^n)$$

$$where\ \underline{m}^n = \frac{k^0}{k^0+n}\underline{m}^0 + \frac{n}{k^0+n}\bar{\underline{x}}^n = \frac{k^{n-1}}{k^n}\underline{m}^{n-1} + \frac{1}{k^n}\underline{x}^n$$

$$k^n = k^0 + n = k^{n-1} + 1;\ \nu^n = \nu^0 + n = \nu^{n-1} + 1$$

$$\Sigma^n = \Sigma^0 + \sum_{i=1}^{n}(\underline{x}^i - \bar{\underline{x}}^n)(\underline{x}^i - \bar{\underline{x}}^n)^T + \frac{\nu^0 n}{\nu^n}(\bar{\underline{x}}^n - \underline{m}^0)(\bar{\underline{x}}^n - \underline{m}^0)^T \text{ (HW: Get recursive expression)}$$

Bayesian Generalization of Laplacian Smoothing

- Recall $L(\{\pi_k\}) = \prod_{k=1}^{C} \pi_k^{n_k}$

- Conjugate prior: Dirichlet distribution

$$p(\underline{\pi} \mid \underline{\alpha}) = Dir(\underline{\pi} \mid \underline{\alpha}) = \frac{\Gamma(\alpha_0)}{\Gamma(\alpha_1)....\Gamma(\alpha_C)} \prod_{k=1}^{C} \pi_k^{\alpha_k - 1}; \alpha_0 = \sum_{k=1}^{C} \alpha_k$$

- Posterior $p(\underline{\pi} \mid D^N, \underline{\alpha}); N = \sum_{k=1}^{C} n_k$

$$p(\underline{\pi} \mid D^N, \underline{\alpha}) = \frac{p(D^N \mid \underline{\pi}).p(\underline{\pi} \mid \underline{\alpha})}{p(D^N \mid \underline{\alpha})} = \frac{\Gamma(\alpha_0 + N)}{\Gamma(\alpha_1 + n_1).....\Gamma(\alpha_C + n_C)} \prod_{k=1}^{C} \pi_k^{\alpha_k + n_k - 1}$$

$$= \mathrm{Dir}(\underline{\pi} \mid \underline{\alpha} + \underline{n})$$

- MAP estimate

$$\hat{\pi}_k^{MAP} = \frac{n_k + \alpha_k - 1}{N + \alpha_0 - C}$$

Conditional mean (MMSE estimate)

$$\hat{\pi}_k^{MMSE} = \frac{n_k + \alpha_k}{N + \alpha_0}$$

Mode and Mean are not the same here!

Operations on Gaussians - 1

- Sum of Gaussian vectors is Gaussian.

$$\underline{x}_1 \sim N(\underline{\mu}_1, \Sigma_{11}); \underline{x}_2 \sim N(\underline{\mu}_2, \Sigma_{22}); \mathrm{cov}(\underline{x}_1, \underline{x}_2) = \Sigma_{12}$$

$$\underline{x} = \underline{x}_1 + \underline{x}_2 \sim N(\underline{\mu}_1 + \underline{\mu}_2, \Sigma_{11} + \Sigma_{12} + \Sigma_{12}^T + \Sigma_{22})$$

- Linear transformations of Gaussians is Gaussian

$$\underline{x} \sim N(\underline{\mu}, \Sigma); \underline{y} = A\underline{x} \sim N(A\underline{\mu}, A\Sigma A^T)$$

$$J_{11} = \left(\Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{12}^T\right)^{-1}; J_{22} = \left(\Sigma_{22} - \Sigma_{12}^T\Sigma_{11}^{-1}\Sigma_{12}\right)^{-1}$$
$$J_{12} = -\Sigma_{11}^{-1}\Sigma_{12}J_{22} = -J_{11}\Sigma_{12}\Sigma_{22}^{-1}$$

- Marginal and conditional densities

$$\underline{x} = \begin{bmatrix} \underline{x}_1 \\ \underline{x}_2 \end{bmatrix} \text{ and } \underline{x} \sim N(\underline{\mu}, \Sigma); \underline{\mu} = \begin{bmatrix} \underline{\mu}_1 \\ \underline{\mu}_2 \end{bmatrix}; \Sigma = \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{12}^T & \Sigma_{22} \end{bmatrix}; J = \Sigma^{-1} = \begin{bmatrix} J_{11} & J_{12} \\ J_{12}^T & J_{22} \end{bmatrix}$$

Then, the marginal & conditional densities are also Gaussian

$$p(\underline{x}_2) = N(\underline{\mu}_2, \Sigma_{22})$$

$$p(\underline{x}_1 \mid \underline{x}_2) = N(\underline{\mu}_1 + \Sigma_{12}\Sigma_{22}^{-1}(\underline{x}_2 - \underline{\mu}_2), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{12}^T)$$

$$= \mathrm{N}(\underline{\mu}_1 - J_{11}^{-1}J_{12}(\underline{x}_2 - \underline{\mu}_2), J_{11}^{-1})$$

If x_1 is scalar (e.g., Gibbs sampling), Information form could save computation.

This is what happens in Least Squares & Kalman filtering

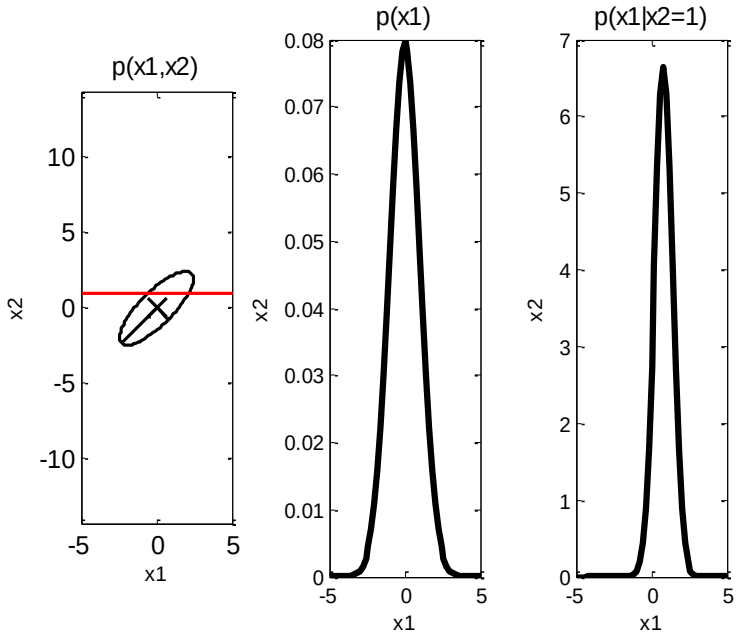
Marginals & Conditionals of 2D Gaussian

- Marginal and conditional densities

$$\underline{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \text{ and } \underline{x} \sim N(\underline{\mu}, \Sigma); \underline{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix} = \underline{0}; \Sigma = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_2^2 \end{bmatrix} = \begin{bmatrix} 1 & 0.8 \\ 0.8 & 1 \end{bmatrix}$$

$$p(x_1) = N(\mu_1, \sigma_1^2) = N(0,1)$$

$$p(x_1 \mid x_2) = N(\mu_1 + \frac{\rho\sigma_1}{\sigma_2}(x_2 - \mu_2), \sigma_1^2(1-\rho^2)). \; If \; x_2 = 1 \Rightarrow p(x_1 \mid x_2) = N(0.8, 0.36)$$



$$J = \begin{bmatrix} \frac{1}{\sigma_1^2(1-\rho^2)} & \frac{-\rho}{\sigma_1\sigma_2(1-\rho^2)} \\ \frac{-\rho}{\sigma_1\sigma_2(1-\rho^2)} & \frac{1}{\sigma_2^2(1-\rho^2)} \end{bmatrix} = \begin{bmatrix} 2.778 & -2.222 \\ -2.222 & 2.778 \end{bmatrix}$$

gaussCondition2Ddemo2
from Murphy, Page 112

Operations on Gaussians - 2

- Covariance matrix captures **marginal** independencies between variables

$$\Sigma_{ij} = 0 \Leftrightarrow x_i \;\&\; x_j \; \textit{are independent} \; (or) \; x_i \perp x_j$$

- Information matrix $J = \Sigma^{-1}$ captures **conditional** independencies

$$J_{ij} = 0 \Leftrightarrow x_i \perp x_j \mid \left\{ \{x_1, x_2, ..., x_n\} - \{x_i, x_j\} \right\}$$

  - Non-zero entries in J correspond to edges in the dependency network

- Product of Gaussian PDFs is proportional to a Gaussian

$$p_i(\underline{x}; \underline{\mu}_i, J_i^{-1}) = \frac{|J_i|^{1/2}}{(2\pi)^{p/2}} \exp\{-\frac{1}{2}(\underline{x} - \underline{\mu}_i)^T J_i (\underline{x} - \underline{\mu}_i)\} = \exp(A_i + \underline{\eta}_i^T \underline{x} - \frac{1}{2} \underline{x}^T J_i \underline{x})$$

$$J_i = \Sigma_i^{-1}; \; \underline{\eta}_i = \Sigma_i^{-1} \underline{\mu}_i = J_i \underline{\mu}_i; \; A_i = -\frac{1}{2}\left(p \ln 2\pi - \ln |J_i| + \underline{\eta}_i^T J_i^{-1} \underline{\eta}_i \right)$$

$$\prod_{i=1}^{n} p_i(\underline{x}; \underline{\mu}_i, J_i^{-1}) \propto p(\underline{x}; \underline{\mu}, J^{-1}) \; \textit{where} \; J = \Sigma^{-1} = \sum_{i=1}^{n} J_i; \; \underline{\mu} = J^{-1}\left(\sum_{i=1}^{n} J_i \underline{\mu}_i \right)$$

- Valid for division also

Sampling From Multivariate Gaussian

$\underline{x} \sim N(\underline{\mu}, \Sigma) = N(\underline{\mu}, J^{-1}); J = \Sigma^{-1}$ (Precision or Information Matrix)

Method 1: $\Sigma = Q\Lambda Q^T; \underline{z} = N(\underline{0}, I); \underline{x} = \underline{\mu} + Q\Lambda^{1/2}\underline{z}$

Method 2: Cholesky Decomposition of $\Sigma = LL^T; \underline{z} = N(\underline{0}, I); \underline{x} = \underline{\mu} + L\underline{z}$

Method 3: Given Cholesky Decomposition of Precision Matrix, $J = BB^T$,

$$\underline{z} = N(\underline{0}, I); \text{Solve } B^T \underline{y} = \underline{z}; \underline{x} = \underline{\mu} + \underline{y}$$

Method 4: Gibbs sampler using Covariance Matrix/Precision Matrix

$$\text{Let } \underline{y} = \begin{bmatrix} x_i \\ \underline{x}_{-i} \end{bmatrix} \sim N\left(\begin{bmatrix} \mu_i \\ \underline{\mu}_{-i} \end{bmatrix}, \begin{bmatrix} \sigma_i^2 & \Sigma_{i,-i} \\ \Sigma^T_{i,-i} & \Sigma_{-i,-i} \end{bmatrix}\right) = N\left(\begin{bmatrix} \mu_i \\ \underline{\mu}_{-i} \end{bmatrix}, \begin{bmatrix} J_{ii} & J_{i,-i} \\ J^T_{i,-i} & J_{-i,-i} \end{bmatrix}^{-1}\right)$$

$$\Rightarrow p(x_i \mid \underline{x}_{-i}) \sim N(\mu_i + \Sigma_{i,-i}\Sigma^{-1}_{-i,-i}(\underline{x}_{-i} - \underline{\mu}_{-i}), \sigma_i^2 - \Sigma_{i,-i}\Sigma^{-1}_{-i,-i}\Sigma^T_{i,-i})$$

$$= N\left(\mu_i - \frac{J_{i,-i}}{J_{ii}}(\underline{x}_{-i} - \underline{\mu}_{-i}), \frac{1}{J_{ii}}\right)$$

Methods 1 and 2 require $O(n^3)$ computation. Method 3 can exploit sparsity of J. Information form of Gibbs sampler does not require matrix inversion! Relation to Gausee-Seidel, successive over-relaxation, etc. https://arxiv.org/pdf/1505.03512.pdf

ML versus Bayesian Learning

ML	Bayesian
Computationally Simpler (calculus, optimization)	Complex Numerical Integration (unless a reproducing density)
Single Best Model. Easier to Interpret $p(\underline{x} \mid z=k, D_k) = p(\underline{x} \mid z=k, \hat{\underline{\theta}}_{ML})$	Weighted Av. of Models $p(\underline{x} \mid z=k, D_k) = \int p(\underline{x} \mid \underline{\theta}, z=k)\, p(\underline{\theta} \mid z=k, D_k)\, d\underline{\theta}$
Does not use a priori information on $\underline{\theta}$	Uses a priori information on $\underline{\theta}$

Density Estimation: Histograms

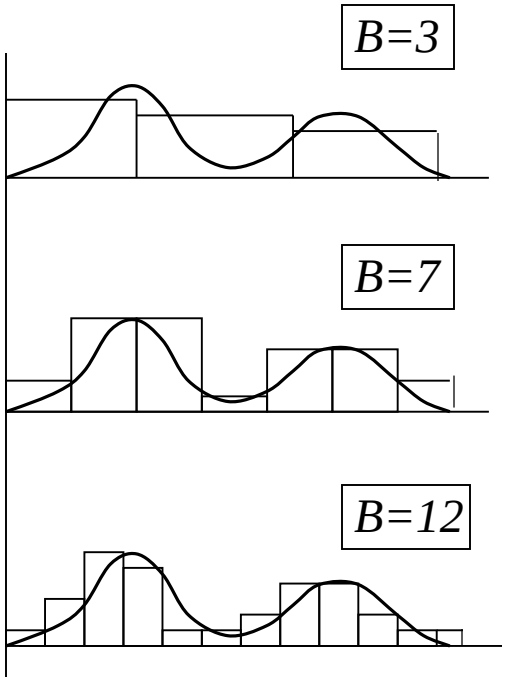
- Assume x was already standardized, then to construct a histogram, divide the range into a set of B evenly spread bins

Then,

$$h(b) = \frac{K_b}{N}, \qquad b = 1, 2, \ldots, B$$

where K_b = number of points which fall inside the bin

- B is crucial
- B large $\Rightarrow$ Est. density is spiky ("noisy")
- B small $\Rightarrow$ Smoothed density
- $\Rightarrow$ There is an optimum choice for B



We can construct the histogram sequentially considering data one at a time

Major Problems with Histogram Algorithm

- Estimated density is not smooth (has discontinuities at the boundaries of the bins)
- In high dimensions, we need B^p bins ($\Rightarrow$ *curse of dimensionality* $\Rightarrow$ requires a huge number of data points to estimate density)
- In high dimensions
  - Density is concentrated in a small part of the space
  - Most of the bins will be empty $\Rightarrow$ estimated density = 0
  - As the number of dimensions grows, a shell of thin, constant thickness on the interior of the sphere ends up containing almost all of its volume $\Rightarrow$ most of the volume is near the surface!

Kernel and K-nearest Neighbor Methods-1

General idea

Suppose we want to find $p(\underline{x})$

$$\Pr ob\{\underline{x} \in R\} = P = \int_{\underline{x} \in R} p(\underline{x}) d\underline{x} \Rightarrow \Pr ob\{\underline{x} \notin R\} = 1 - P$$

Suppose draw N points from $p(\underline{x})$

$$\Pr ob\{ k \ of \ \text{these fall in } R\} = \binom{N}{k} P^k (1-P)^{N-k} = B(k; N, P)$$

Expected number falling in region ***R***

$$E(k) = \sum_{k=0}^{N} k \binom{N}{k} P^k (1-P)^{N-k} = \sum_{k=1}^{N} k \binom{N}{k} P^k (1-P)^{N-k} = NP \sum_{k=1}^{N} \binom{N-1}{k-1} P^{k-1} (1-P)^{N-k}$$

$$= NP$$

So, $\boxed{E[k] = NP}$

Kernel and K-nearest Neighbor Methods-2

Expected fraction of points falling in region $\boldsymbol{R}$ $= \dfrac{E[k]}{N} = P$

Variance of $\dfrac{k}{N}$ $\Rightarrow E\left[\left(\dfrac{k}{N} - P\right)^2\right] = \sum_{k=1}^{N}\left(\dfrac{k}{N} - P\right)^2 \binom{N}{k} P^k (1-P)^{N-k} = \dfrac{P(1-P)}{N}$

As, $N \to \infty$, variance $\downarrow 0$

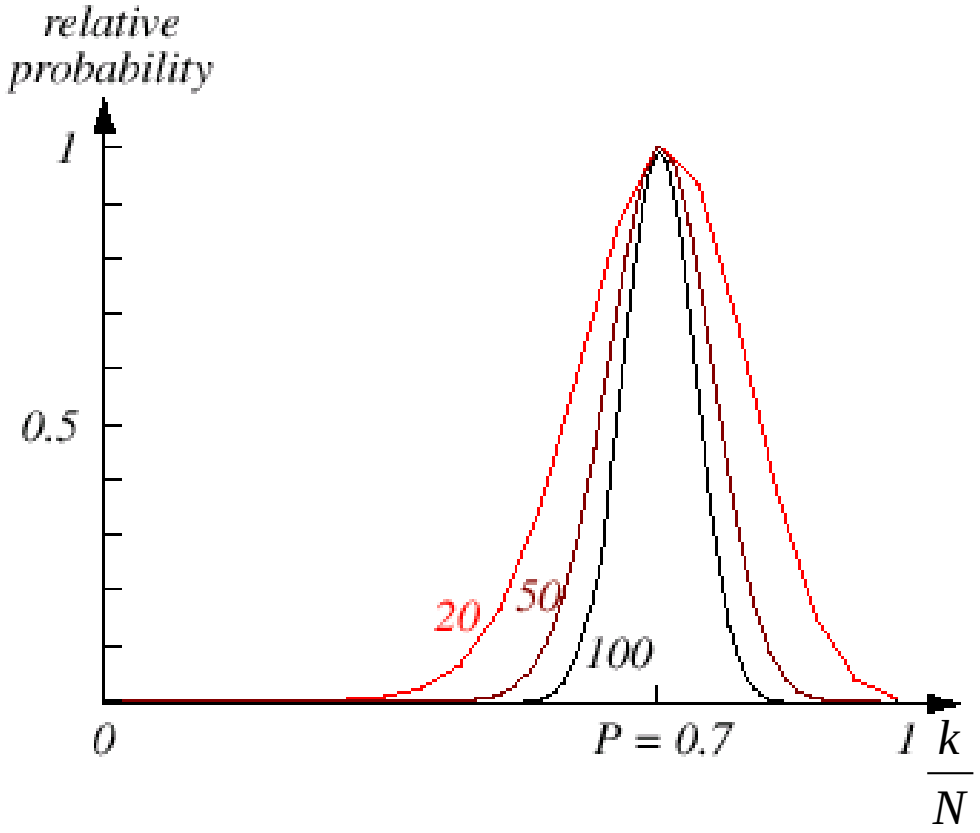
$$\Rightarrow P \approx \frac{k}{N} \text{ is a good estimate} \qquad \ldots\ldots(1)$$

Also, $$P = \int_{\mathbf{R}} p(\underline{x})dx \approx p(\underline{x})V \qquad \ldots\ldots(2)$$

So, $\boxed{p(\underline{x})V = \dfrac{k}{N}}$ or, $\boxed{p(\underline{x}) \cong \dfrac{k}{NV}}$ $\qquad \ldots\ldots(3)$

Note that $\boldsymbol{R}$ should be large for (1) to hold. However, $\boldsymbol{R}$ should be small for (2) to hold $\Rightarrow \exists$ an optimal choice for $\boldsymbol{R}$.

Effect of N on Probability Estimates



We are trying to estimate P via $p(x) \cong \frac{k}{NV}$. As N increases, the estimate peaks at the true value (it becomes a delta function)

Kernel and K-nearest Neighbor Methods-3

- There are basically two approaches to use Eq. (3) for density estimation
  - Fix V and determine k from the data $\Rightarrow$ Kernel Based Estimation
  - Fix k and determine the corresponding volume from the data $\Rightarrow$ k-nearest neighbor approach.

  - Major Disadvantage: Needs all data

Both of these methods converge to true densities as $N \to \infty$, provided

- $V \downarrow$ as $N \uparrow \Rightarrow V \to 0$ as $N \to \infty$
- $k \uparrow$ as $N \uparrow \Rightarrow k \to \infty$ as $N \to \infty$ and $k/N \to 0$

$$Typically\ select\ V = \frac{1}{\sqrt{N}}\ for\ \text{ker}\,nel - based\ methods$$

$$Select\ k = \sqrt{N}\ for\ k - nearest\ neighbor\ methods$$

Selection of V and k

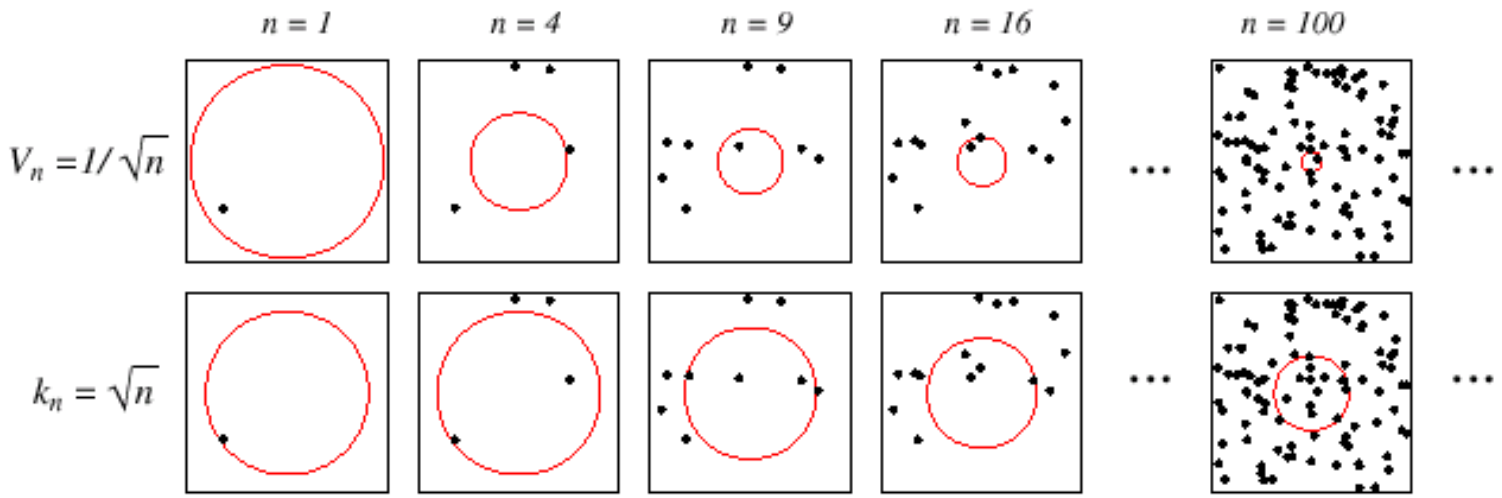


FIGURE 4.2. There are two leading methods for estimating the density at a point, here at the center of each square. The one shown in the top row is to start with a large volume centered on the test point and shrink it according to a function such as $V_n = 1/\sqrt{n}$. The other method, shown in the bottom row, is to decrease the volume in a data-dependent way, for instance letting the volume enclose some number $k_n = \sqrt{n}$ of sample points. The sequences in both cases represent random variables that generally converge and allow the true density at the test point to be calculated. From: Richard O. Duda, Peter E. Hart, and David G. Stork, *Pattern Classification*. Copyright © 2001 by John Wiley & Sons, Inc.

Kernel Estimators-1

❑ Kernel Estimators

- Suppose we take the region ***R*** to be a hypercube with sides of length h centered on the point $\underline{x}$. Its volume is $V = h^P$
- We can find an expression for k, the number of points which fall within this region, by defining a Kernel Function, $H(\underline{u})$. It is also known as the **Parzen Window**

$$H(\underline{u}) = \begin{cases} 1 \ for \ |u_i| < \frac{1}{2} & i = 1, 2, \ ...., p \\ 0 & otherwise \end{cases}$$

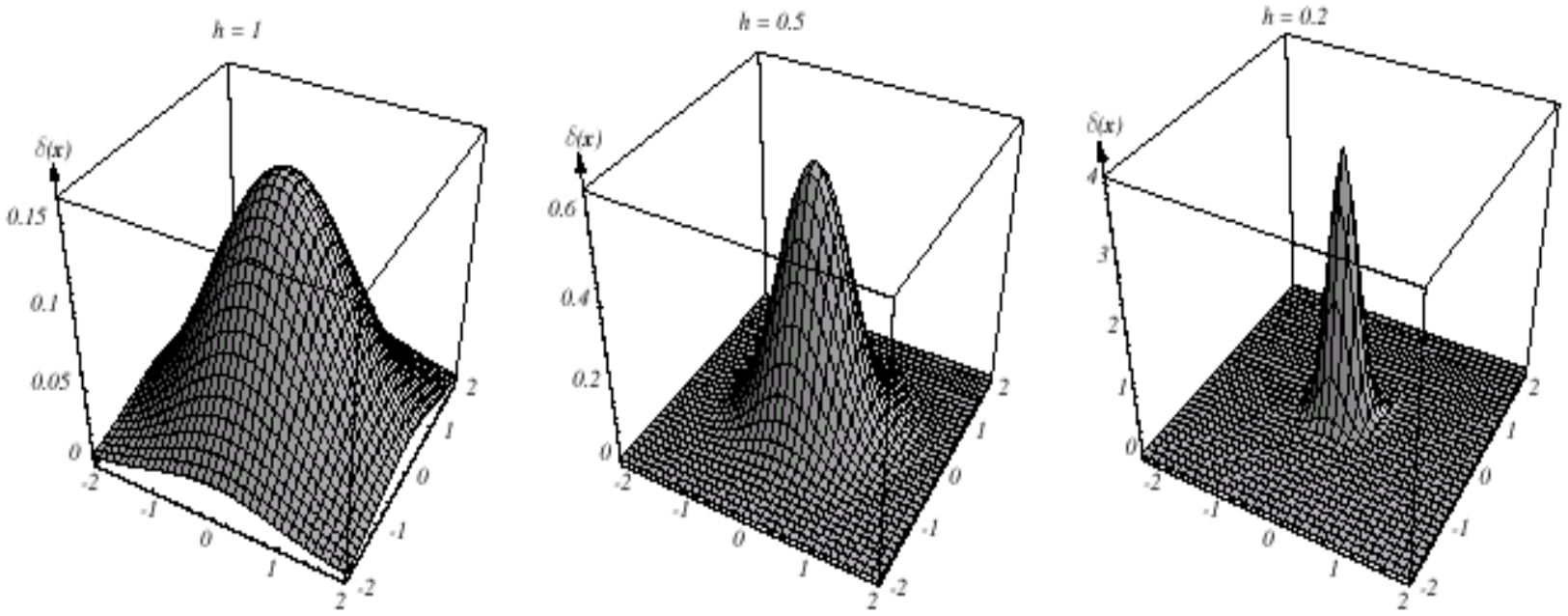
$H(\underline{u})$ is a unit hypercube centered at the origin. Gaussian windows could be used as well.

$\Rightarrow$ For all data points $\underline{x}^j$, the quantity $H(\frac{\underline{x} - \underline{x}^j}{h})$ is equal to unity (*1*) if the point $\underline{x}^j$ falls inside the hypercube of side h at $\underline{x}$ and 0 otherwise.

$\Rightarrow$ Total number of points falling inside the hypercube is $k = \sum_{j=1}^{N} H(\frac{\underline{x} - \underline{x}^j}{h})$

$$\Rightarrow \hat{p}(\underline{x}) \cong \frac{k}{Nh^p} = \frac{1}{Nh^p} \sum_{j=1}^{N} H(\frac{\underline{x} - \underline{x}^j}{h})$$

Gaussian Parzen Windows



Example of two dimensional circularly symmetric normal Parzen windows for three different values of *h*

$$H(\frac{\underline{x}}{h}) = \frac{1}{(2\pi)^{p/2} h^p} \exp[-\frac{1}{2} \| \frac{\underline{x}}{h} \|^2]$$

Kernel Estimators-2

$$\hat{p}(\underline{x}) = \frac{k}{NV} = \frac{1}{Nh^p}\sum_{j=1}^{N} H\left(\frac{\underline{x}-\underline{x}^j}{h}\right)$$

$\Rightarrow \hat{p}(\underline{x})$ = superposition of N cubes of side h, with each cube centered on one of the data points

$\Rightarrow$ We can smooth out this estimate by choosing different forms for the kernel function $H(\underline{u})$. *Example: Gaussian Parzen Windows*

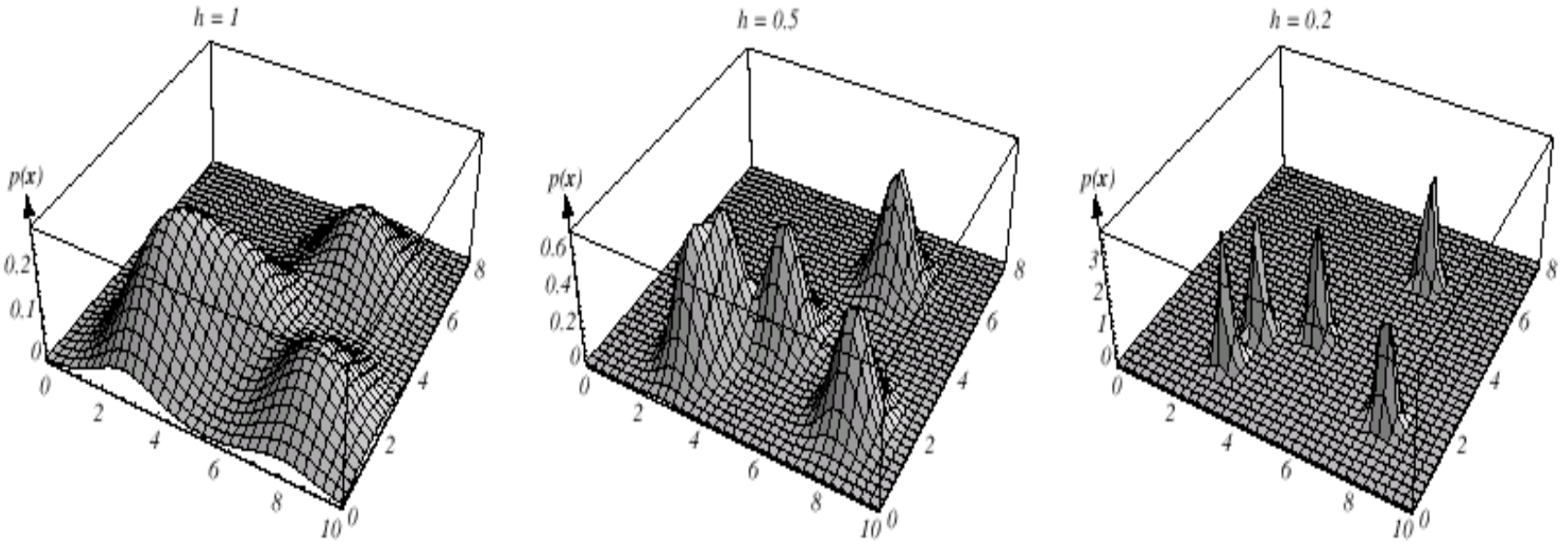
$$E[\hat{p}(\underline{x})] = E\left[\frac{1}{N}\frac{1}{h^p}\sum_{j=1}^{N} H\left(\frac{\underline{x}-\underline{x}^j}{h}\right)\right] = E\left[\frac{1}{h^p} H\left(\frac{\underline{x}-\underline{v}}{h}\right)\right]$$

$$= \frac{1}{h^p}\int_{\underline{v}} H(\underline{x}-\underline{v})\,p(\underline{v})\,d\underline{v}$$

Convolution of H and p. Blurred version of $p(\underline{x})$ as seen through the window

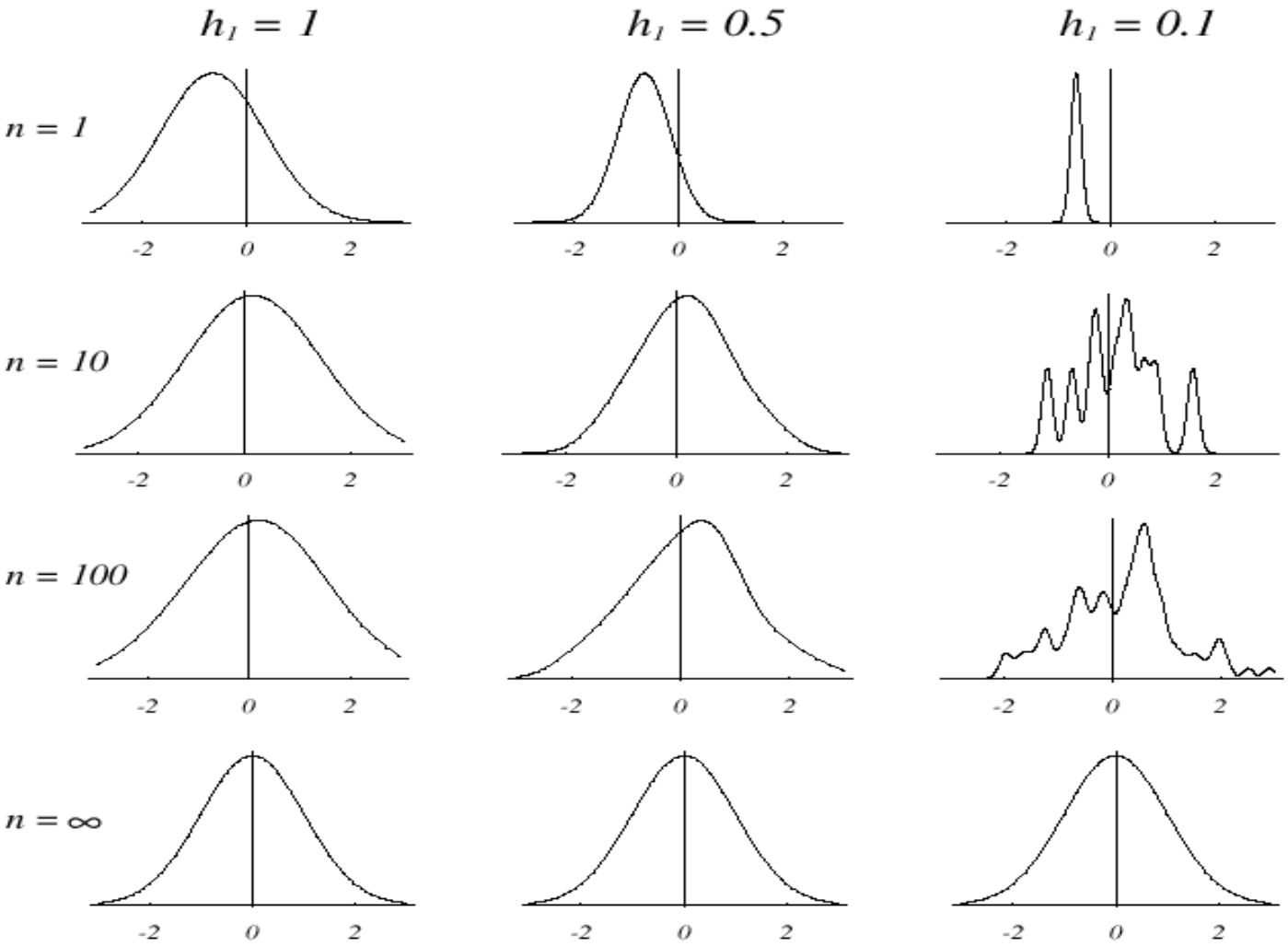
Large $N \Rightarrow$ Good estimate for $h \downarrow 0$
Small $N \Rightarrow$ Need to select h properly
For small N, h small $\Rightarrow$ noisy estimate

Illustration of Density Estimation: Effect of *h*



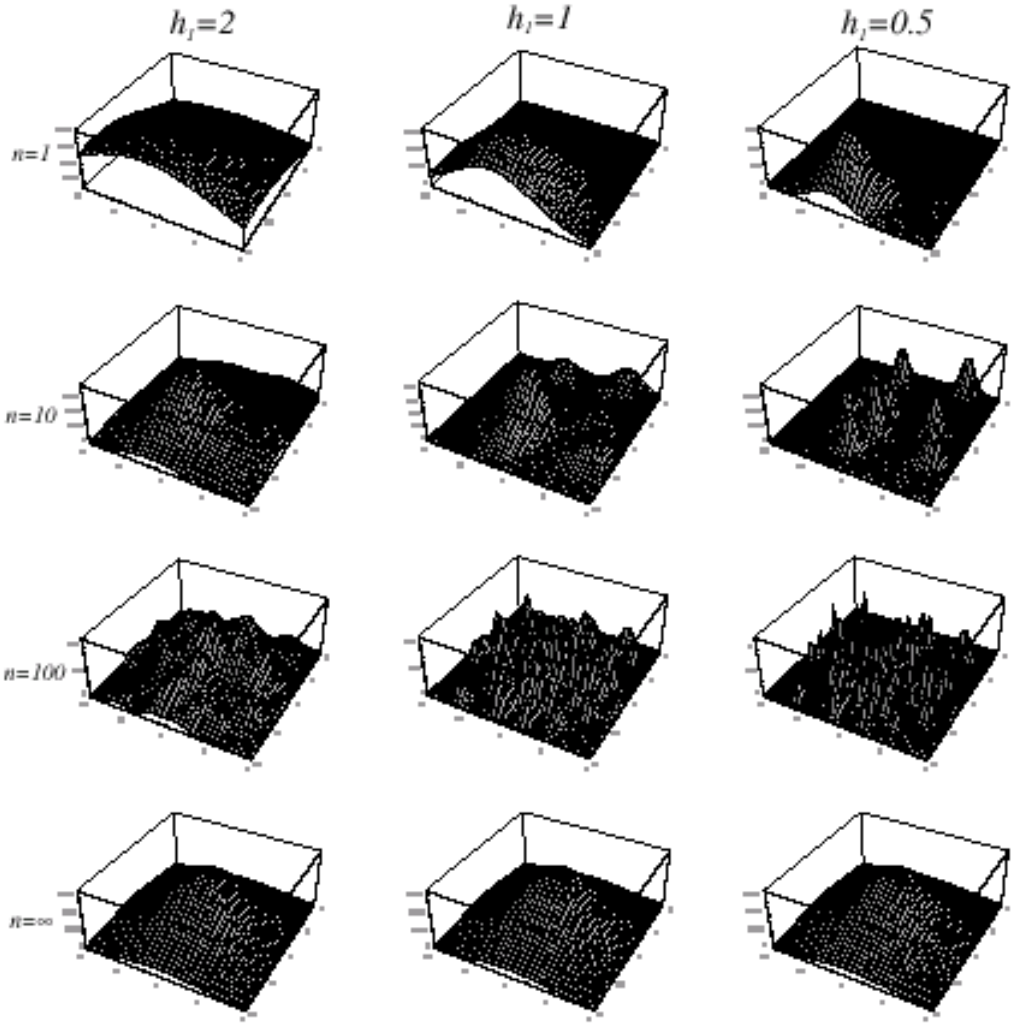
Three Parzen-window density estimates based on the same set of five samples, using the Gaussian Parzen window functions

Density Estimation: Effects of *h* and *N*



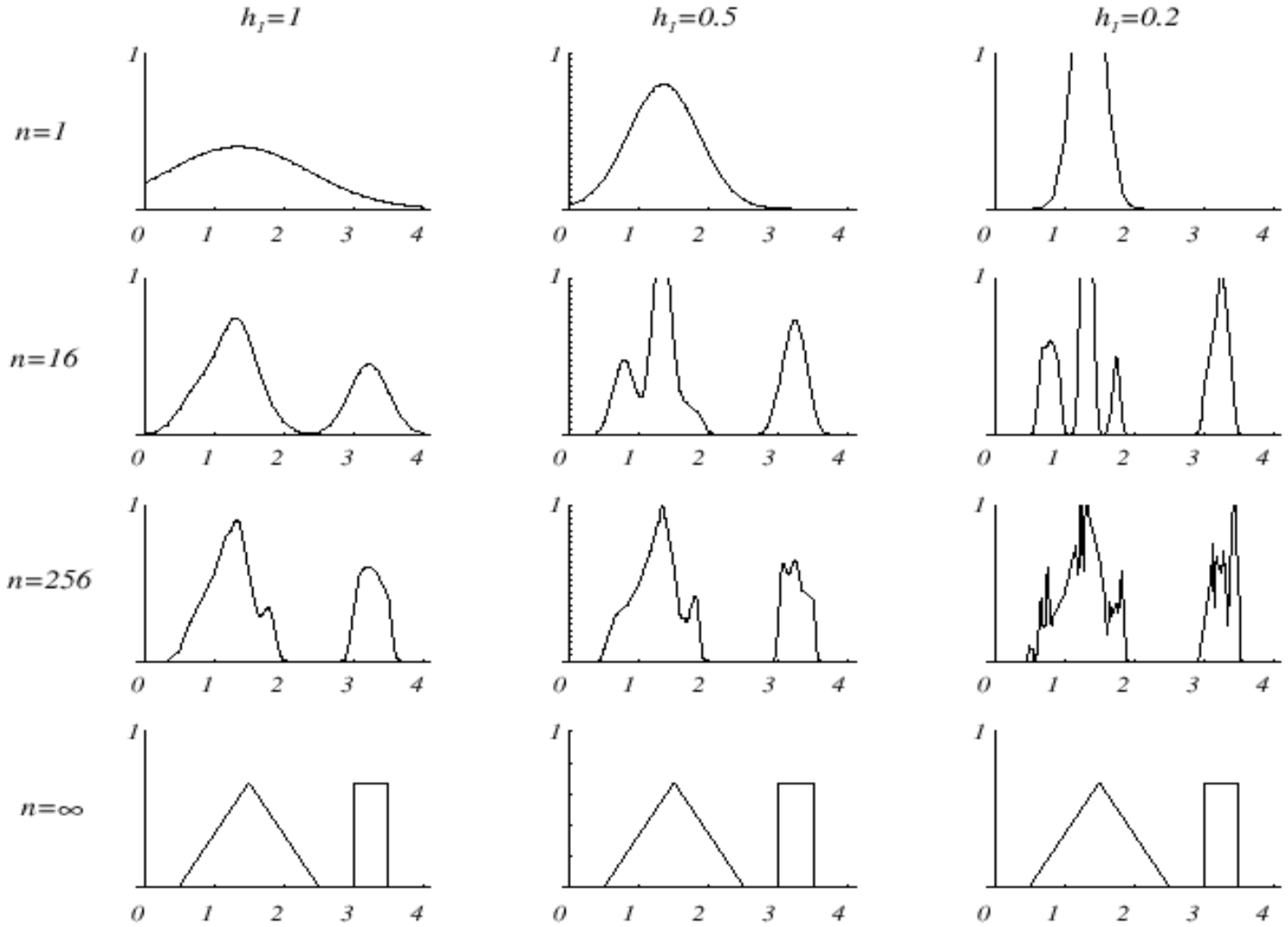
Parzen-window estimates of a univariate normal density using different window widths and numbers of samples

Bivariate Density Estimation



Parzen-window estimates of a bivariate normal density using different window widths and numbers of samples

Bimodal Density Estimation



Parzen-window estimates of a bimodal distribution using different window widths and numbers of samples. Note that when $n \to \infty$, estimates are the same and match the true distribution.

Gaussian Windows:PNN

Volume of a hypersphere in p dimensions of radius $\sigma = \frac{2(\pi)^{p/2}}{\Gamma(p/2)} \frac{\sigma^p}{p}$

$$\Gamma(a) = \int_0^\infty u^{a-1} e^{-u} du$$

$$\hat{p}(\underline{x}) = \frac{1}{(2\pi)^{p/2}\sigma^p} \frac{1}{N} \sum_{j=1}^{N} \exp\left\{ -\frac{(\underline{x} - \underline{x}^j)^T (\underline{x} - \underline{x}^j)}{2\sigma^2} \right\} \qquad \sigma = h$$

= sum of multivariate Gaussian distributions centered at each training sample.

(can also do for each class $\hat{p}(\underline{x} \mid z = k) \Rightarrow \begin{matrix} N \to n_k \\ \underline{x}^j \to \underline{x}_k^j \end{matrix}$)

→ Also called "Probabilistic Neural Network (PNN)"

Small σ ⇒ density estimation will have discontinuities
Larger σ ⇒ causes greater degree of interpolation (smoothness)

Probabilistic Neural Network-1

- Note that each component of $\underline{x}$ has the same σ

⇒ Must scale x_i to have the same range

Scale

↓

Compute $\hat{\Sigma}_k$

↓

$$\hat{\sigma}_k^2 = \frac{1}{p}\mathrm{tr}(\hat{\Sigma}_k)$$

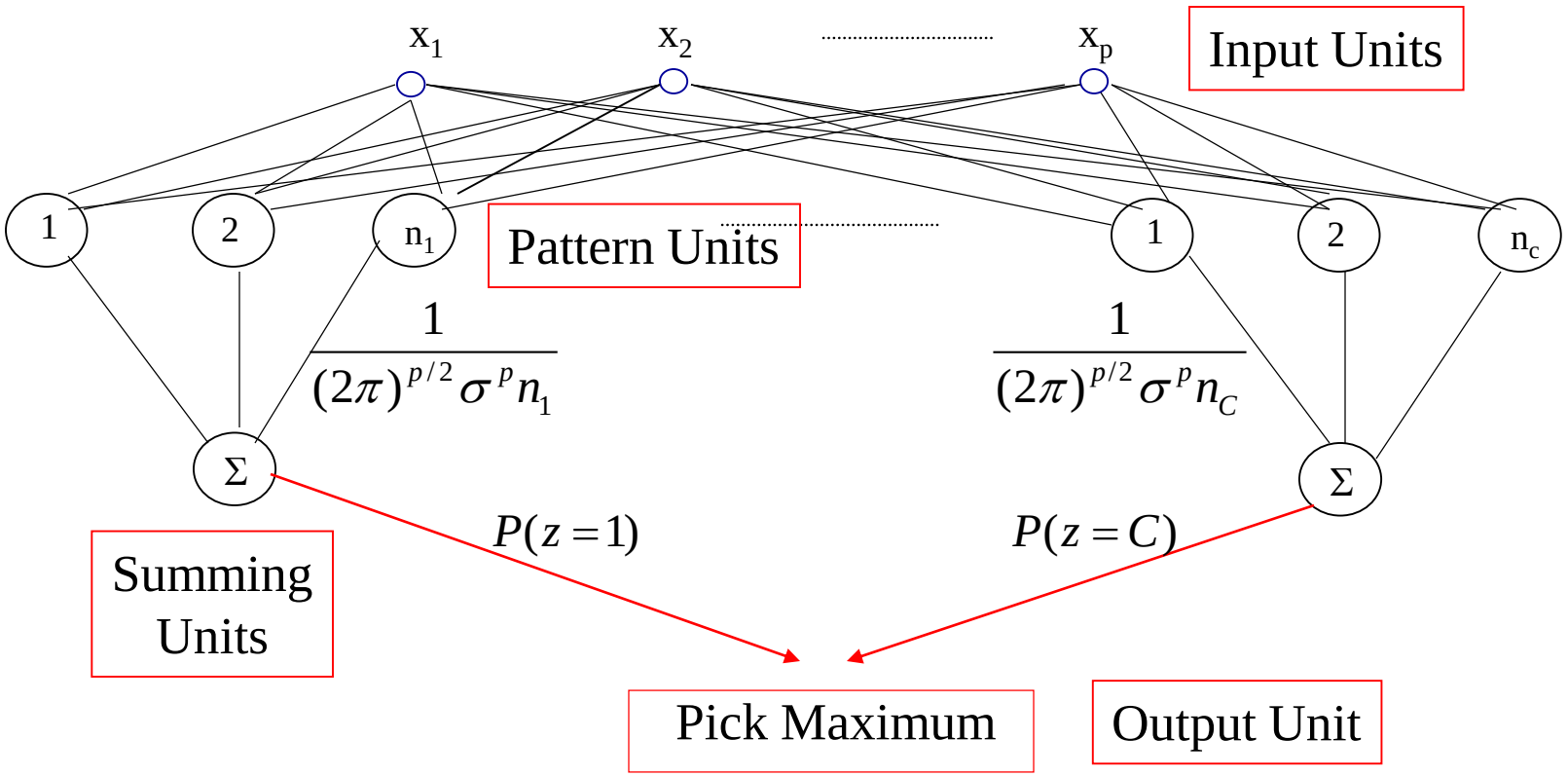
↓

$$\sigma^2 = \frac{1}{N}\sum_{k=1}^{C} n_k \hat{\sigma}_k^2$$

PNN is fairly insensitive to σ over a wide range. Sometimes, it is easier to experiment with σ than compute it this way.

Probabilistic Neural Network-2

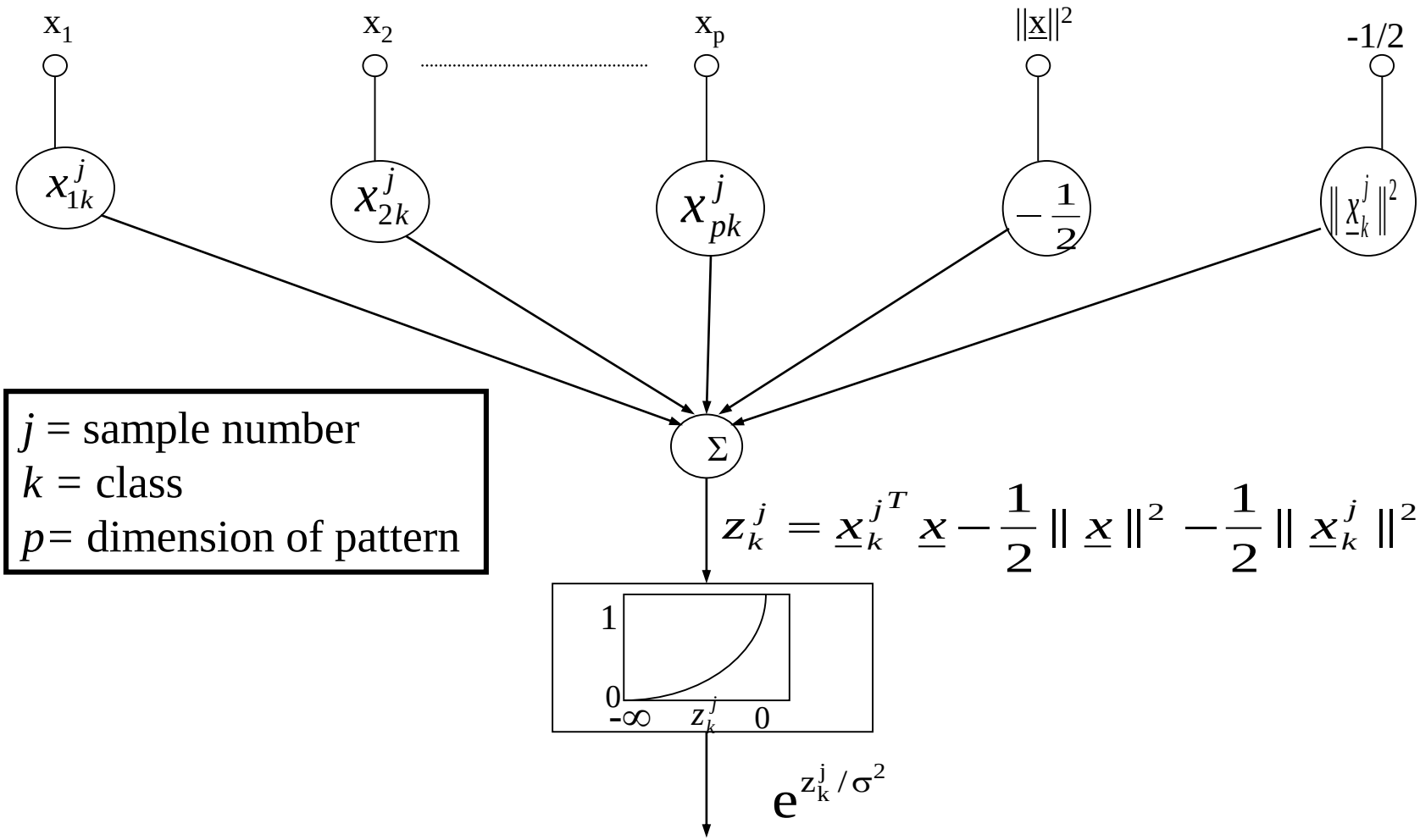
PNN structure



- Speed up PNN using cluster centers as representative patterns
  - Cluster using *K*-means, LVQ,….

Probabilistic Neural Network-2

One form of Pattern Unit structure

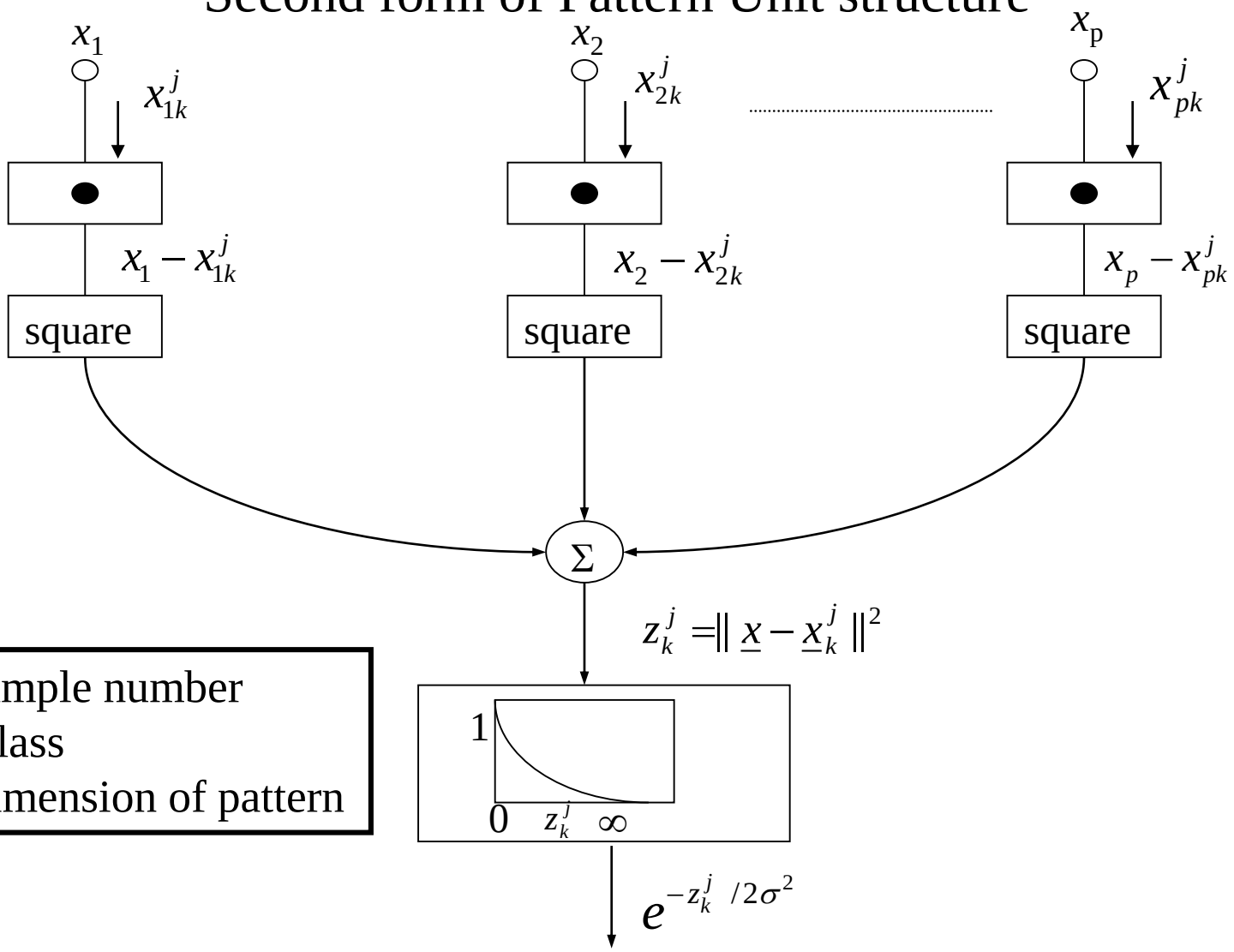


j = sample number
k = class
p= dimension of pattern

$$z_k^j = \underline{x}_k^{jT}\,\underline{x} - \frac{1}{2}\|\underline{x}\|^2 - \frac{1}{2}\|\underline{x}_k^j\|^2$$

Probabilistic Neural Network-3

Second form of Pattern Unit structure



j = sample number
k = class
p= dimension of pattern

Alternative forms of Kernels

❑ Alternate forms of Kernels:

$$p(x) = \frac{1}{Nh^p} \sum_{j=1}^{N} \exp\left(-\underbrace{\frac{2}{h} \sum_{i=1}^{p} | x_i - x_i^j |}_{\text{Manhattan Distance, 1-norm}} \right)$$

$$p(\underline{x}) = \frac{1}{N(\pi h)^p} \sum_{j=1}^{N} \prod_{i=1}^{p} \left[1 + \frac{(x_i - x_i^j)^2}{h^2} \right]^{-1} \quad \text{Cauchy Distribution}$$

$$p(x) = \frac{1}{N(2\pi\sigma)^p} \sum_{j=1}^{N} \prod_{i=1}^{p} \left[\sin c\left(\frac{(x_i - x_i^j)}{\sigma} \right) \right]^2 \quad \text{where} \quad \sin c(x) = \frac{\sin x}{x}$$

$$p(\underline{x}) = \frac{1}{N} \left(\frac{3}{4} \right)^p \sum_{j=1}^{N} \prod_{i=1}^{p} \left(1 - x_i^{\,2} \right); | x_i | \le 1 \quad \textit{Epanechnikov Kernel}$$

$$p(\underline{x}) = \frac{1}{N} \left(\frac{70}{81} \right)^p \sum_{j=1}^{N} \prod_{i=1}^{p} \left(1 - | x_i |^3 \right); | x_i | \le 1 \quad \textit{tri-cube Kernel}$$

K – Means Algorithm or *K* – Centers Algorithm

- Decide on *K*, the number of clusters in advance.
- Suppose there are N data points, $D = \left\{ \underline{x}^1 \; \underline{x}^2 \ldots\ldots, \underline{x}^N \right\}$
- Want to find *K* representative vectors $\left\{ \underline{\mu}_1 \; \underline{\mu}_2 \ldots\ldots \; \underline{\mu}_K \right\}$
- *K*-means algorithm seeks to partition the data into *K* disjoint subsets $\{C_j\}$ containing $\{n_j\}$ points, in such a way as to minimize the sum of squares clustering function

$$J = \sum_{j=1}^{K} \sum_{i \in C_j} \| \underline{x}^i - \underline{\mu}_j \|^2$$

clearly if $\{C_j\}$ are known, then

$$\underline{\mu}_j = \frac{1}{n_j} \sum_{i \in C_j} \underline{x}^i$$

The means of clusters
"Cluster Centers"
"Codebook Vectors"

Batch Version of *K*–Means Algorithm

❑ *K*–Means Algorithm-Unsupervised Learning

- 1) Start with any K random feature vectors as K centers. Alternately, assign the N points into K sets randomly and compute their means; these means serve as initial centers.

2) For $i = 1, 2, \ldots\ldots, N$

Assign pattern i to cluster C_j if $j = \arg \min_{1 \le k \le K} \| \underline{x}_i - \underline{\mu}_k \|_2$

3) Recompute means via: $\underline{\mu}_j = \frac{1}{n_j} \sum_{i \in C_j} \underline{x}^i$

4) If centers have changed, go to step 2. Else, stop.

Covariance of each cluster: $\hat{\Sigma}_j = \frac{1}{n_j - 1} \sum_{i \in \mathbf{C_j}} (\underline{x}_i - \underline{\mu}_j)(\underline{x}_i - \underline{\mu}_j)^T$ etc.

Initialization

- Initialization (Recall David Arthur and Sergei Vassilvitskii's "k-means++: The Advantages of Careful Seeding", *Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*)

a. Choose initial center at random. Let n_1 be the data point.

Let $\underline{\mu}_1 = \underline{x}^{n_1}$.

b. For $k = 2,..,K$

For $n=1,2,..,\ N$ & $n \neq n_i$, $i=1,2,..,k-1$

$$D_n = \min_{1 \le i \le k-1} \left\| \underline{x}^n - \underline{\mu}_i \right\|_2^2 \quad or \quad D_n = \exp\left(\min_{1 \le i \le k-1} \left\| \underline{x}^n - \underline{\mu}_i \right\|_2^2 \right)$$

End

Select $\underline{\mu}_k = \underline{x}^{n_k}$ probabilistically $p(\underline{x}^{n_k}) = \dfrac{D(\underline{x}^{n_k})}{\sum_{\substack{n=1 \\ n \neq n^i; i=1,2,..,k-1}}^{N} D(\underline{x}^n)}$

Store n_k

End

Selection of *K*

- Pick *K* to minimize the sum of errors (training error) + model complexity term. The sum is called **prediction error**.

$$PE = \frac{2}{N}\sum_{j=1}^{K}\sum_{n\in C_j}\left\|\underline{x}^n - \underline{\mu}_j\right\|^2 + \frac{2Kp}{N}\sigma^2$$

where σ^2 is the variance of noise in the data.

- *Kurtosis based measure*
  - Kurtosis for normalized Gaussian $E\left[\left(\frac{x-\mu}{\sigma}\right)^4\right] = 3$
  - Find Excess Kurtosis for each cluster

$$K_{ji} = \frac{1}{|C_j|}\sum_{n\in \mathbf{C}_j}\left(\frac{x_i^n - \mu_{ji}}{\sigma_{ij}}\right)^4 - 3 \qquad i = 1, 2, \cdots\cdots, p$$

$$K_T = \frac{1}{Kp}\sum_{j=1}^{K}\sum_{i=1}^{p}K_{ji} = \frac{1}{K}\sum_{j=1}^{K}K_j;\; K_j = \frac{1}{p}\left[\sum_{i=1}^{p}K_{ji}\right]$$

  - Plot K_T vs. K and pick K that gives minimum K_T
  - Can also use this idea in a dynamic cluster splitting scheme (see Vlasis and Likos, *IEEE Trans.- SMCA*, July 1999, pp-393-399).

Online Version of K-Means

❑ Online Version

- Start with K randomly chosen centers $\rightarrow \{\mu_j\}_{j=1}^{K}$
- For each data point, update the nearest $\underline{\mu}^j$ via

$$\underline{\mu}^{j^{new}} = \underline{\mu}^{j^{old}} + \eta(\underline{x}^i - \underline{\mu}^{j^{old}})$$

Vector Quantization via Stochastic Approximation

Leave all others the same.

Can use it for dynamic data.

For static data, need to go through data multiple times!

Supervised Algorithm

❑ **Supervised Algorithm** ($\Rightarrow$Learning Vector Quantization {know class labels!})

- Start with K codebook vectors.
- For each data point $\underline{x}^i \Rightarrow$ find the closest codebook (or center) $\underline{\mu}^j$.

$$\underline{\mu}^{j^{new}} = \underline{\mu}^{j^{old}} + \alpha(\underline{x}^i - \underline{\mu}^{j^{old}}) \quad \text{if } \underline{x}^i \text{ is classified correctly.}$$

$$\underline{\mu}^{j^{new}} = \underline{\mu}^{j^{old}} - \alpha(\underline{x}^i - \underline{\mu}^{j^{old}}) \quad \text{if } \underline{x}^i \text{ is classified incorrectly.}$$

It is a version of reinforcement learning

We will take up the variants of the algorithm in Lecture 8.

k-Nearest Neighbor Approach

❑ *k*-Nearest Neighbor Approach

- Fix k and allow the volume V to vary

$$p(\underline{x}) \cong \frac{k}{NV}$$

V is the volume of a hypersphere centered at point $\underline{x}$ and contains exactly k points.

Again, Small $k \Rightarrow$ noisy

Large $k \Rightarrow$ smooth

Major use of *k*- Nearest Neighbor Technique is not in probability estimation, but in classification.

- *Assign a point to class i if class i has the largest number of points among the k-nearest neighbors* $\Rightarrow$ **MAP rule**

$$p(\underline{x} \mid z = i) = \frac{k_i}{n_i V} \qquad P(z = i) = \frac{n_i}{N}$$

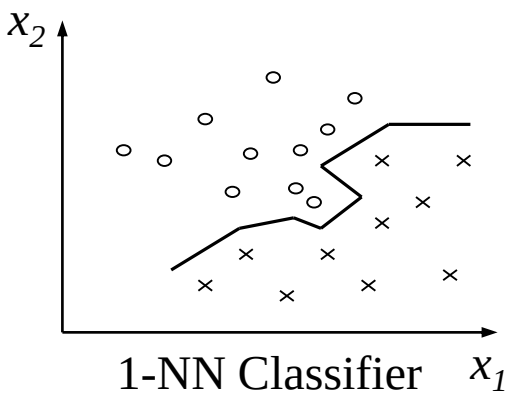
$$p(\underline{x}) = \frac{k}{NV}$$

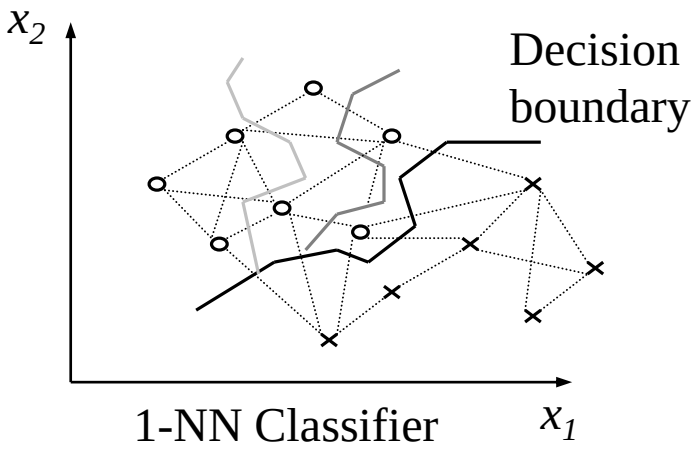
$$P(z = i \mid \underline{x}) = \frac{p(\underline{x} \mid z = i) P(z = i)}{p(\underline{x})} = \frac{k_i}{n_i V} \frac{n_i}{N} \frac{1}{\frac{k}{NV}} = \frac{k_i}{k}$$

1-Nearest Neighbor Classifier

1-NN Classifier

1-NN classifier has interesting Geometric Interpretation





- Data points become corner points of triangles spanned by each reference point and two of its neighbors. The network of triangles is called Delauny Triangulation (or Delauny Net).

- Perpendiculars to triangle's edges run borders of polygonal patches delimiting local neighborhood of each point. Resulting network of polygons is called VORONOI Diagram or VORONOI Nets.

- Decision Boundary follows sections of VORONOI Nets.

- 1-NN & K-NN adapt to any data; high variability & low bias.

Gaussian Mixtures, Maximum Likelihood and Expectation Maximization Algorithm - 1

❑ Why Gaussian Mixtures?

- Parametric → fast but limited
- Non Parametric → general but slow (require lot of data)
- Mixture Models
  - → RBF
  - → Conditional Density Estimation (function approx.)
  - → Mixture of experts models

$$p(\underline{x}) = \sum_{j=1}^{M} p(\underline{x} \mid j) P_j$$

$$\sum_{j=1}^{M} P_j = 1 \quad ; \; 0 \le P_j \le 1$$

$$\sum_{j=1}^{M} \int_{\underline{x}} p(\underline{x} \mid j) P_j \, d\underline{x} = 1$$

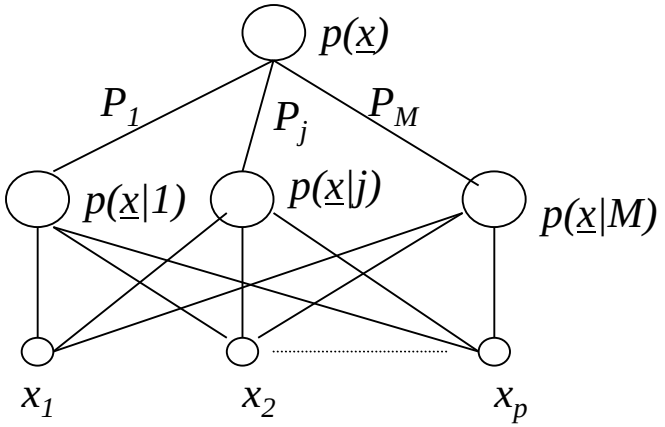
Similar technique for $p(.) = p(\underline{x} \mid k)$ $k = 1, 2, \ldots, C$

Gaussian Mixtures, Maximum Likelihood and Expectation Maximization Algorithm - 2

$$p(\underline{x} \mid j) = N(\underline{\mu}_j, \Sigma_j)$$

$$= N(\underline{\mu}_j, \sigma_j{}^2 I) \quad \text{typically}$$

$$= \frac{1}{(2\pi\sigma_j{}^2)^{p/2}} e^{\left(\frac{-\|(\underline{x}-\underline{\mu}_j)\|_2^2}{2\sigma_j{}^2} \right)}$$



Problem: Given data,

$$D = \left\{ \underline{x}^1 \ \underline{x}^2 \ \ldots . \underline{x}^N \right\}, \ \textit{find the ML estimates of} \ \left\{ P_j, \underline{\mu}_j, \sigma_j \right\}_{j=1}^{M}$$

Let $\underline{\theta} = \left\{ P_j, \underline{\mu}_j, \sigma_j \right\}$

$$L = \max_{\underline{\theta}} p(D|\underline{\theta}) \quad \Rightarrow \max_{\underline{\theta}} l = \left[\ln p(D \mid \underline{\theta}) \right] \quad \Rightarrow \min_{\underline{\theta}} \left[-\ln p(D \mid \underline{\theta}) \right] = J$$

Gaussian Mixtures, Maximum Likelihood and Expectation Maximization Algorithm - 3

$$J = -\sum_{i=1}^{N} \ln\ p(\underline{x}^i, \underline{\theta}) \ = -\sum_{i=1}^{N} \ln \left(\sum_{j=1}^{M} p(\underline{x}^i \mid j) P_j \right)$$

$$\min \quad J$$

$$s.t. \sum_{j=1}^{M} P_j = 1; \ \ 0 \le P_j \le 1$$

Lagrangian:

$$l = J + \lambda \sum_{i=1}^{M} P_j - \lambda$$

$$\frac{\partial J}{\partial \underline{\mu}_j} = -\sum_{i=1}^{N} \frac{1}{\sum_{k=1}^{M} p(\underline{x}^i \mid k) P_k} P_j \frac{\partial p(\underline{x}^i \mid j)}{\partial \underline{\mu}_j}$$

$$\frac{\partial p(\underline{x}^i \mid j)}{\partial \underline{\mu}_j} = -\frac{1}{(2\pi\sigma_j^{\ 2})^{p/2}} \mathrm{e}^{\left(\frac{-\|(\underline{x}^i - \underline{\mu}_j)\|^2}{2\sigma_j^{\ 2}} \right)} \frac{\underline{\mu}_j - \underline{x}^i}{\sigma_j^{\ 2}} = -p(\underline{x}^i \mid j) \frac{\underline{\mu}_j - \underline{x}^i}{\sigma_j^2}$$

$$So, \quad \frac{\partial J}{\partial \underline{\mu}_j} = \sum_{i=1}^{N} P(j \mid \underline{x}^i) \frac{\underline{\mu}_j - \underline{x}^i}{\sigma_j^{\ 2}} \qquad \ldots\ldots\ldots\ldots\ldots\ldots (1)$$

posterior

Note the Simplicity of Gradient

Gaussian Mixtures, Maximum Likelihood and Expectation Maximization Algorithm - 4

$$\frac{\partial J}{\partial \underline{\sigma}_j} = -\sum_{i=1}^{N} \frac{1}{\sum_{k=1}^{M} p(\underline{x}^i|k)P_k} P_j \frac{\partial p(\underline{x}^i|j)}{\partial \sigma_j}$$

Dimension of feature vector

$$= \sum_{i=1}^{N} P(j|\underline{x}^i) \left\{ \frac{p}{\sigma_j} - \frac{||\underline{x}^i - \underline{\mu}_j||^2}{\sigma_j^3} \right\} \quad \text{. . . . .(2)}$$

$$\frac{\partial l}{\partial P_j} = -\sum_{i=1}^{N} \frac{1}{\sum_{k=1}^{M} p(\underline{x}^i \mid k)P_k} p(\underline{x}^i \mid j) + \lambda$$

$$= -\sum_{i=1}^{N} \frac{P(j|\underline{x}^i)}{P_j} + \lambda \qquad \Rightarrow \frac{1}{P_j}\left[-\sum_{i=1}^{N} P(j|\underline{x}^i) + \lambda P_j \right] \quad \text{. . . . . .(3)}$$

Gaussian Mixtures, Maximum Likelihood and Expectation Maximization Algorithm - 5

From (1),

$$\hat{\underline{\mu}}_j = \frac{\sum_{i=1}^{N} P(j \mid \underline{x}^i)\underline{x}^i}{\sum_{i=1}^{N} P(j \mid \underline{x}^i)}$$

Necessary Conditions of Optimality:
Set Gradients Equal to Zero

$$\hat{\underline{\sigma}}_j{}^2 = \frac{1}{p} \frac{\sum_{i=1}^{N} P(j \mid \underline{x}^i) \| \underline{x}^i - \hat{\underline{\mu}}_j \|^2}{\sum_{i=1}^{N} P(j \mid \underline{x}^i)}$$

General Case:

$$\Sigma_j = \frac{\sum_{i=1}^{N} P(j \mid \underline{x}^i)(\underline{x}^i - \hat{\underline{\mu}}_j)(\underline{x}^i - \hat{\underline{\mu}}_j)^T}{\sum_{i=1}^{N} P(j \mid \underline{x}^i)}$$

noting that, $\sum_{j=1}^{M} P(j \mid \underline{x}^i) = 1$ and $\sum_{j=1}^{M} P_j = 1$ we have $\lambda = \mathrm{N}$

$$\Rightarrow \hat{P}_j = \frac{1}{N} \sum_{i=1}^{N} P(j \mid \underline{x}^i)$$

Responsibility

These are coupled non-linear equations

Methods of Solution: NLP

- Nonlinear Programming (NLP) Techniques

$$\underline{\theta}_0 \rightarrow \underline{\theta}_1 \rightarrow \ldots\ldots\ldots\ldots \underline{\theta}^*$$

$$\underline{\theta}_{k+1} \rightarrow \underline{\theta}_k - \eta H \nabla_{\underline{\theta}} l$$

$$H = \begin{cases} I & \Rightarrow \text{SD or Gradient Method} \\ \left[\nabla^2 J\right]^{-1} & \Rightarrow \text{Newton's Method} \\ \left[\nabla^2 J + \varepsilon I\right]^{-1} & \Rightarrow \text{Levenberg-Marquardt Method} \\ \left[\nabla_{\underline{\theta}} J \nabla_{\underline{\theta}} J^T + \varepsilon I\right]^{-1} & \Rightarrow \text{Levenberg-Marquardt version of Gauss Newton Method} \\ & \text{Various versions of Quasi-Newton Method} \\ & \text{Various versions of Conjugate Gradient method} \end{cases}$$

Best to compute Hessian using finite Difference method

EM Algorithm

❑ EM Algorithm

Gauss-Seidel view of EM

M-step

$$\hat{\mu}_j^{new} = \frac{\sum_{i=1}^{N} \hat{P}^{old}(j \mid \underline{x}^i)\underline{x}^i}{\sum_{i=1}^{N} \hat{P}^{old}(j \mid \underline{x}^i)}$$

How did we get these equations and Why?…. Later

- By setting gradient to zero (M-step)
- Evaluating posterior Probabilities/Responsibilities (E-step)

$$\hat{\sigma}_j^{new^2} = \frac{1}{p} \frac{\sum_{i=1}^{N} \hat{P}^{old}(j \mid \underline{x}^i) \| \underline{x}^i - \underline{\hat{\mu}}_j^{new} \|^2}{\sum_{i=1}^{N} \hat{P}^{old}(j \mid \underline{x}^i)}$$

E-step

$$\hat{P}^{new}(j \mid \underline{x}^i) = \frac{p(\underline{x}^i \mid j)\hat{P}_j^{new}}{\sum_{m=1}^{M} p(\underline{x}^i \mid m)\hat{P}_m^{new}}$$

$$\hat{P}_j^{new} = \frac{1}{N} \sum_{i=1}^{N} \hat{P}^{old}(j \mid \underline{x}^i)$$

Sequential Estimation -1

Sequential Estimation ~ Stochastic Approximation

$$\hat{\underline{\mu}}_j^{n+1} = \frac{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)\underline{x}^i}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)}$$

$$= \frac{\sum_{i=1}^{n} P(j \mid \underline{x}^i)}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)} \hat{\underline{\mu}}_j^{n} + \frac{P(j \mid \underline{x}^{n+1})}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)} \underline{x}^{n+1}$$

$$= \hat{\underline{\mu}}_j^{n} + \frac{P(j \mid \underline{x}^{n+1})}{\underbrace{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)}_{\eta_j^{n+1}}} \left[\underline{x}^{n+1} - \hat{\underline{\mu}}_j^{n} \right]$$

Note:

$$\frac{1}{\eta_j^{n+1}} = \frac{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)}{P(j \mid \underline{x}^{n+1})} = 1 + \frac{\sum_{i=1}^{n} P(j \mid \underline{x}^i)}{P(j \mid \underline{x}^{n+1})}$$

$$= 1 + \frac{\sum_{i=1}^{n} P(j \mid \underline{x}^i)}{P(j \mid \underline{x}^{n+1})} \cdot \frac{P(j \mid \underline{x}^n)}{P(j \mid \underline{x}^n)}$$

$$= 1 + \frac{P(j \mid \underline{x}^n)}{P(j \mid \underline{x}^{n+1})} \cdot \frac{1}{\eta_j^n}$$

Sequential Estimation ~ Stochastic Approximation -2

- Sometimes replace, $\underline{\eta}_j^{n+1} = \dfrac{P(j \mid \underline{x}^{n+1})}{(n+1)\hat{P}_j^{n+1}}$ or, $\dfrac{1}{\underline{\eta}_j^{n+1}} = \dfrac{P(j \mid \underline{x}^{n})}{P(j \mid \underline{x}^{n+1})}\dfrac{1}{\underline{\eta}_j^{n}} + 1$

Similarly, $\hat{\underline{\sigma}}_j^{2\,n} = \dfrac{1}{p}\dfrac{\sum_{i=1}^{n} P(j \mid \underline{x}^i)\,\| \underline{x}^i - \underline{\hat{\mu}}_j^{n} \|^2}{\sum_{i=1}^{n} P(j \mid \underline{x}^i)}$

$$\boxed{\underline{\eta}_j^{n+1} = \frac{\underline{\eta}_j^{n} P(j \mid \underline{x}^{n+1})}{\underline{\eta}_j^{n} P(j \mid \underline{x}^{n+1}) + P(j \mid \underline{x}^{n})}}$$

$$\hat{\underline{\sigma}}_j^{2\,n+1} = \frac{1}{p}\frac{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)\,\| \underline{x}^i - \underline{\hat{\mu}}_j^{n+1} \|^2}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)} = \frac{1}{p}\frac{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)\,\| \underline{x}^i + \underline{\hat{\mu}}_j^{n} - \underline{\hat{\mu}}_j^{n} - \underline{\hat{\mu}}_j^{n+1} \|^2}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)}$$

$$= \frac{1}{p}\frac{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)\left\{ \| \underline{x}^i - \underline{\hat{\mu}}_j^{n} \|^2 + 2\left(\underline{x}^i - \underline{\hat{\mu}}_j^{n} \right)^T \left(\underline{\hat{\mu}}_j^{n} - \underline{\hat{\mu}}_j^{n+1} \right) + \| \underline{\hat{\mu}}_j^{n} - \underline{\hat{\mu}}_j^{n+1} \|^2 \right\}}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)}$$

$$= \hat{\underline{\sigma}}_j^{2\,n} \frac{\sum_{i=1}^{n} P(j \mid \underline{x}^i)}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)} + \frac{1}{p}\frac{P(j \mid \underline{x}^{n+1})}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)}\left(\| \underline{x}^{n+1} - \underline{\hat{\mu}}_j^{n} \|^2 \right) - \frac{1}{p}\left(\| \underline{\hat{\mu}}_j^{n} - \underline{\hat{\mu}}_j^{n+1} \|^2 \right)$$

Sequential Estimation ~ Stochastic Approximation -3

$$= \hat{\underline{\sigma}}_j^{2^n} + \frac{P(j \mid \underline{x}^{n+1})}{\sum_{i=1}^{n+1} P(j \mid \underline{x}^i)} \left(\frac{\| \underline{x}^{n+1} - \hat{\underline{\mu}}_j^n \|^2}{p} - \hat{\underline{\sigma}}_j^{2^n} \right) - \frac{1}{p} \left(\| \hat{\underline{\mu}}_j^n - \hat{\underline{\mu}}_j^{n+1} \|^2 \right)$$

$$\hat{\underline{\sigma}}_j^{2^{n+1}} = \hat{\underline{\sigma}}_j^{2^n} + \eta_j^{n+1} \left[\frac{1}{p} \| \underline{x}^{n+1} - \hat{\underline{\mu}}_j^n \|^2 - \hat{\underline{\sigma}}_j^{2^n} \right] - \frac{1}{p} \left(\| \hat{\underline{\mu}}_j^n - \hat{\underline{\mu}}_j^{n+1} \|^2 \right)$$

$$= \hat{\underline{\sigma}}_j^{2^n} + \eta_j^{n+1} \left[\frac{1}{p} (1 - \eta_j^{n+1}) \| \underline{x}^{n+1} - \hat{\underline{\mu}}_j^n \|^2 - \hat{\underline{\sigma}}_j^{2^n} \right]$$

- Similarly,

Recall

$$\hat{\underline{\mu}}_j^{n+1} - \hat{\underline{\mu}}_j^n = \eta_j^{n+1} \left[\underline{x}^{n+1} - \hat{\underline{\mu}}_j^n \right]$$

$$\hat{P}_j^{n+1} = \hat{P}_j^n + \frac{1}{n+1} \left[P(j \mid \underline{x}^{n+1}) - \hat{P}_j^n \right]$$

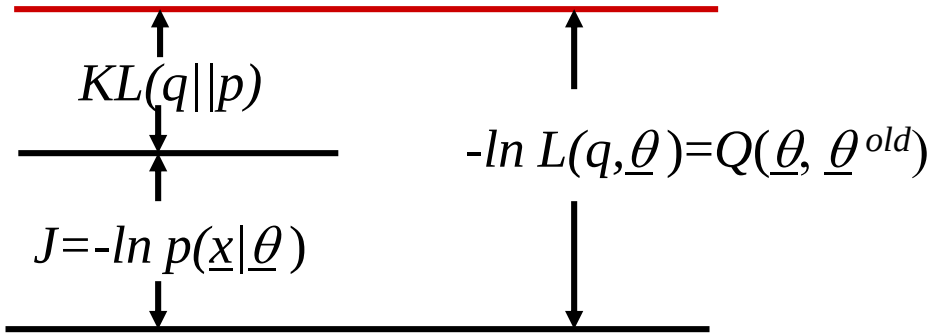
EM Algorithm for Gaussian Mixture Problem-1

❑ Key ideas of EM as applied to Gaussian Mixture Problem

$$J = -\sum_{i=1}^{N} \ln\, p(\underline{x}^i) \quad = -\sum_{i=1}^{N} \ln \left(\sum_{j=1}^{M} p(\underline{x}^i \mid j) P_j \right)$$

$$J^{new} - J^{old} = -\sum_{i=1}^{N} \ln \left[\frac{p^{new}(\underline{x}^i)}{p^{old}(\underline{x}^i)} \right]$$

$$= -\sum_{i=1}^{N} \ln \left[\frac{\sum_{j=1}^{M} P_j^{new} p^{new}(\underline{x}^i | j) \frac{P^{old}(j|\underline{x}^i)}{P^{old}(j|\underline{x}^i)}}{p^{old}(\underline{x}^i)} \right]$$



Idea:

$\underline{x}$: *data*

$\underline{z}$: *hidden* var *iables* (*mixture*)

$\underline{\theta}$: *parameters*

$q(\underline{z})$ = *any arbitrary distribution*

$$-\ln p(\underline{x}, \underline{z} \mid \underline{\theta}) = -\ln p(\underline{z} \mid \underline{x}, \underline{\theta}) - \ln p(\underline{x} \mid \underline{\theta})$$

$$-\ln p(\underline{x} \mid \underline{\theta}) = \underbrace{-E_q[\ln \frac{p(\underline{x}, \underline{z} \mid \underline{\theta})}{q(\underline{z})}]}_{\ln L(q, \underline{\theta})} + \underbrace{E_q[\ln \frac{p(\underline{z} \mid \underline{x}, \underline{\theta})}{q(\underline{z})}]}_{-KL(q(\underline{z}) \| p(\underline{z} \mid \underline{x}, \underline{\theta}))}$$

$$\Rightarrow J = -\ln L(q, \underline{\theta}) - KL(q(\underline{z}) \,||\, p(\underline{z} \mid \underline{x}, \underline{\theta}))$$

$$J \le -\ln L(q, \underline{\theta}) \because KL(q(\underline{z}) \,||\, p(\underline{z} \mid \underline{x}, \underline{\theta})) \ge 0$$

$$E - step : q(\underline{z}) = p(\underline{z} \mid \underline{x}, \underline{\theta}^{old})$$

$$M - step : \underline{\theta}^{new} = \min_{\underline{\theta}}[-\ln L(q, \underline{\theta})]$$

$$= \min_{\underline{\theta}} -E_q[\ln p(\underline{x}, \underline{z} \mid \underline{\theta})]$$

$$= \min_{\underline{\theta}} \tilde{Q}(\underline{\theta}, \underline{\theta}^{old})$$

$$Note : -\ln L(q, \underline{\theta}) = Q(\underline{\theta}, \underline{\theta}^{old}) = \tilde{Q}(\underline{\theta}, \underline{\theta}^{old}) - H_q(\underline{z}, \underline{\theta}^{old})$$

EM Algorithm for Gaussian Mixture Problem-2

- For convex functions,

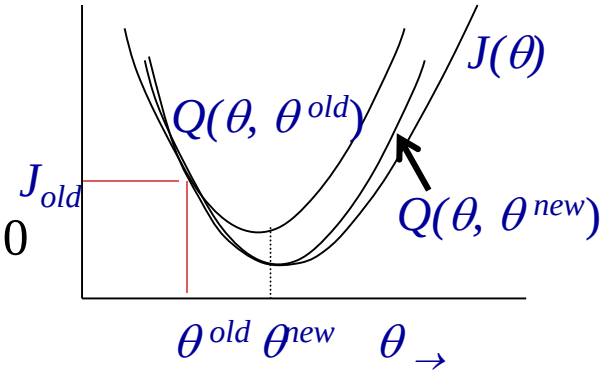
$$-\ln\left[\sum \lambda_i x_i\right] \le -\sum \lambda_i \ln x_i \qquad \text{where } \sum \lambda_i = 1,\ \lambda_i \ge 0$$

$$\Rightarrow J^{new} - J^{old} \le -\sum_{i=1}^{N}\sum_{j=1}^{M} P^{old}(j \mid \underline{x}^i) \ln\left[\frac{P_j^{new} p^{new}(\underline{x}^i \mid j)}{p^{old}(\underline{x}^i)\ P^{old}(j \mid \underline{x}^i)}\right]$$

$$= -\sum_{i=1}^{N}\sum_{j=1}^{M} P^{old}(j \mid \underline{x}^i) \ln\left[\frac{p^{new}(\underline{x}^i, j)}{p^{old}(\underline{x}^i, j)}\right]$$

$$\Rightarrow J^{new} \le -\sum_{i=1}^{N}\sum_{j=1}^{M} \underbrace{P^{old}(j \mid \underline{x}^i)}_{q(\underline{z}, \theta^{old})} \ln p^{new}(\underline{x}^i, j) = Q(\theta, \theta^{old}) = -\ln L(q, \theta)$$

$\Rightarrow$ Minimizing l will lead to a decrease in $J(\theta)$



$$Note: at\ \theta^{old},\ J^{old}(\theta^{old}) = Q(\theta^{old}, \theta^{old}) \Rightarrow Force\ KL = 0$$

$$J^{new}(\theta^{new}) \le Q(\theta^{new}, \theta^{old}) \Rightarrow KL \ge 0$$

$Q(\theta, \theta^{old})$ and $J(\theta)$ have the same gradient at θ^{old}

EM Algorithm for Gaussian Mixture Problem-3

Dropping terms that depend on old parameters, we get

$$Q = -\sum_{i=1}^{N}\sum_{j=1}^{M} P^{old}(j \mid \underline{x}^i) \ln\left[P_j^{new} p^{new}(\underline{x}^i \mid j)\right] = -\sum_{i=1}^{N}\sum_{j=1}^{M} P^{old}(j \mid \underline{x}^i) \ln\left[p^{new}(\underline{x}^i, j)\right]$$

For Gaussian conditional probability density functions

$$Q = -\sum_{i=1}^{N}\sum_{j=1}^{M} P^{old}(j \mid \underline{x}^i) \left\{ \ln P_j^{new} - p \ln \sigma_j^{new} - \frac{\| \underline{x}^i - \underline{\mu}_j^{new} \|^2}{2\sigma_j^{2new}} \right\}$$

- Optimization problem:
  - min Q
  - s.t. $\sum_{j=1}^{M} P_j^{new} = 1; \qquad P_j^{new} \geq 0; \qquad j = 1,2,\ldots\ldots\ldots M$

EM Algorithm for Gaussian Mixture Problem-4

$$\underline{\mu}_j^{new} = \frac{\sum_{i=1}^{N} P^{old}(j|\underline{x}^i)\underline{x}^i}{\sum_{i=1}^{N} P^{old}(j|\underline{x}^i)}$$

$$\sigma_j^{2\,new} = \frac{1}{p}\frac{\sum_{i=1}^{N} P^{old}(j \mid \underline{x}^i) \| \underline{x}^i - \underline{\mu}_j^{new} \|^2}{\sum_{i=1}^{N} P^{old}(j \mid \underline{x}^i)}$$

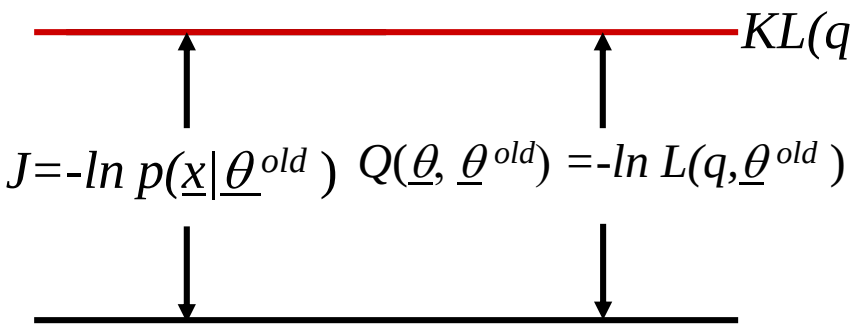
General Case:

$$\Sigma_j^{new} = \frac{\sum_{i=1}^{N} P^{old}(j \mid \underline{x}^i)(\underline{x}^i - \underline{\hat{\mu}}_j^{new})(\underline{x}^i - \underline{\hat{\mu}}_j^{new})^T}{\sum_{i=1}^{N} P^{old}(j \mid \underline{x}^i)}$$

$$P_j^{new} = \frac{1}{N}\sum_{i=1}^{N} P^{old}(j \mid \underline{x}^i)$$

Graphical Illustration of E and M Steps

- **E-step**



$$J = -\ln L(q,\underline{\theta}) - KL(q(\underline{z}) \,||\, p(\underline{z} \,|\, \underline{x},\underline{\theta}))$$

$$J \le -\ln L(q,\underline{\theta}) \because KL(q(\underline{z}) \,||\, p(\underline{z} \,|\, \underline{x},\underline{\theta})) \ge 0$$

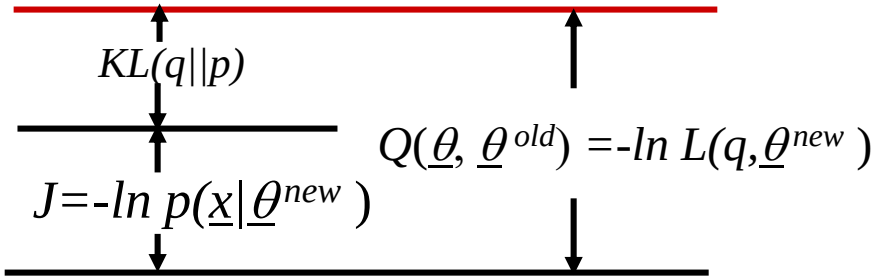
$$E-step: q(\underline{z}) = p(\underline{z} \,|\, \underline{x},\underline{\theta}^{old})$$

$$\Rightarrow -\ln L(q,\underline{\theta}^{old}) = -\ln p(\underline{x} \,|\, \underline{\theta}^{old})$$

Why?

$$\because KL(q(\underline{z}) \,||\, p(\underline{z} \,|\, \underline{x},\underline{\theta}^{old})) = 0$$

- **M-step**



$$M-step: \underline{\theta}^{new} = \arg\min_{\underline{\theta}}[-\ln L(q,\underline{\theta})]$$

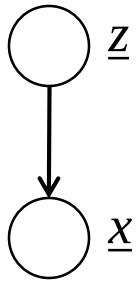
$$\Rightarrow -\ln L(q,\underline{\theta}^{new}) \ge -\ln p(\underline{x} \,|\, \underline{\theta}^{new})$$

why?

$$\because KL(q(\underline{z}) = p(\underline{z} \,|\, \underline{x},\underline{\theta}^{old}) \,||\, p(\underline{z} \,|\, \underline{x},\underline{\theta}^{new})) \ge 0$$

Note: EM is a Maximum Likelihood Algorithm. Is there a Bayesian Version? Yes: If you assume priors on $(\{\mu_j, \sigma_j^2, P_j\})$ called *Variational Bayesian Inference.*

An Alternate View of EM for Gaussian Mixtures - 1



Hidden (Latent) Variables

Observation

- $\underline{z}$ is a M-dimensional binary random vector such that

$$z_j \in \{0,1\} \text{ and } \sum_{j=1}^{M} z_j = 1$$

$$P(z_j = 1) = P_j \Rightarrow P(\underline{z}) = \prod_{j=1}^{M} P_j^{z_j}$$

- $\underline{x}$ is a p-dimensional random vector such that

$$p(\underline{x} \mid \underline{z}) = \prod_{j=1}^{M} [N(\underline{x}; \underline{\mu}_j, \Sigma_j)]^{z_j}$$

$$\Rightarrow p(\underline{x}) = \sum_{\underline{z}} p(\underline{x}, \underline{z}) = \sum_{\underline{z}} P(\underline{z})\, p(\underline{x} \mid \underline{z})$$

$$= \sum_{\underline{z}} \prod_{j=1}^{M} [P_j N(\underline{x}; \underline{\mu}_j, \Sigma_j)]^{z_j} = \sum_{j=1}^{M} P_j N(\underline{x}; \underline{\mu}_j, \Sigma_j)$$

pdf of $\underline{x}$ is a Gaussian Mixture

only possible $\underline{z}$ vectors:

$\underline{z} \in \{\underline{e}_i : i = 1, 2, .., M\}$

$\underline{e}_i = i^{th}$ unit vector

An Alternate View of EM for Gaussian Mixtures - 2

- If have several observations $\{\underline{x}^n: n=1,2,..,N\}$, each data point will have a corresponding latent vector $\underline{z}_n$.

Note the generality

Problem: Given incomplete (partial) data,

$$D = \left\{ \underline{x}^1 \ \underline{x}^2 \ \ldots . \underline{x}^N \right\}, \ \textit{find the ML estimates of} \ \left\{ P_j, \underline{\mu}_j, \Sigma_j \right\}_{j=1}^{M}$$

$$\text{Let } \underline{\theta} = \left\{ P_j, \underline{\mu}_j, \Sigma_j \right\}_{j=1}^{M}$$

$$\min_{\underline{\theta}} J \quad \textit{where} \quad J = -\ln p(D \mid \underline{\theta})$$

Complete Data:

$$D_c = \left\{ (\underline{x}^1, \underline{z}^1), \ (\underline{x}^2, \underline{z}^2) \ldots . (\underline{x}^N, \underline{z}^N) \right\}$$

$$\Rightarrow -\ln p(D_c \mid \underline{\theta}) = \sum_{n=1}^{N} \sum_{j=1}^{M} z_j^n \{ -\ln P_j + \frac{p}{2} \ln 2\pi + \frac{1}{2} \ln |\Sigma_j| + \frac{1}{2} \| \underline{x}^n - \underline{\mu}_j \|_{\Sigma_j^{-1}}^2 \}$$

An Alternate View of EM for Gaussian Mixtures - 3

- If had complete data, estimation is trivial. Similar to Gaussian case, except that we estimate with subsets of data that are assigned to each mixture component
- In EM, replace each latent variable by its expectation *with respect to the posterior density* during the **E-step**

$$z_j^n \rightarrow E(z_j^n \mid \underline{x}^n, \underline{\theta}) = P(z_j^n = 1 \mid \underline{x}^n, \underline{\theta}) = \gamma_j^n$$

$$P(z_j^n = 1 \mid \underline{x}^n, \underline{\theta}) = \frac{P_j N(\underline{x}^n; \underline{\mu}_j, \Sigma_j)}{\sum_{k=1}^{M} P_k N(\underline{x}^n; \underline{\mu}_k, \Sigma_k)} = \gamma_j^n$$

Responsibilities

- In EM, minimize the *expected value of the negative complete-data log likelihood* during the **M-step**

$$E_{\underline{Z}}\{-\ln p(D_c \mid \underline{\theta})\} = \sum_{n=1}^{N}\sum_{j=1}^{M} \gamma_j^n \{-\ln P_j + \frac{p}{2}\ln 2\pi + \frac{1}{2}\ln|\Sigma_j| + \frac{1}{2}\| \underline{x}^n - \underline{\mu}_j \|^2_{\Sigma_j^{-1}}\}$$

$Q(\underline{\theta}, \underline{\theta}^{old})$

EM Algorithm for Gaussian Mixtures -4

1. Initialize the means $\{\underline{\mu}_j\}_{j=1}^M$, covariances $\{\Sigma_j\}_{j=1}^M$, and mixing coefficients $\{P_j\}_{j=1}^M$.

Evaluate $J = -\ln\ p(\underline{x}\,|\,\underline{\theta}) = -\sum_{n=1}^{N} \ln\{\sum_{j=1}^{M} P_j N(\underline{x}^n; \underline{\mu}_j, \Sigma_j)\}$

2. E-step: Evaluate the responsibilities using the current parameter values

$$\gamma_j^n = \frac{P_j N(\underline{x}^n; \underline{\mu}_j, \Sigma_j)}{\sum_{k=1}^{M} P_k N(\underline{x}^n; \underline{\mu}_k, \Sigma_k)}; \; j = 1, 2, .., M; n = 1, 2, .., N$$

$$N_j = \sum_{n=1}^{N} \gamma_j^n; \; j = 1, 2, .., M$$

3. M-step: Re-estimate the parameters using the current responsibilities

$$\underline{\mu}_j^{new} = \frac{1}{N_j} \sum_{n=1}^{N} \gamma_j^n \underline{x}^n$$

$$\Sigma_j^{new} = \frac{1}{N_j} \sum_{n=1}^{N} \gamma_j^n (\underline{x}^n - \underline{\mu}_j^{new})(\underline{x}^n - \underline{\mu}_j^{new})^T$$

$$P_j^{new} = \frac{N_j}{N}$$

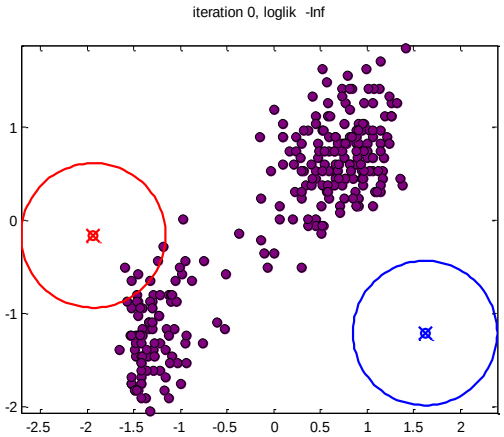
For unbiased estimate of covariance,

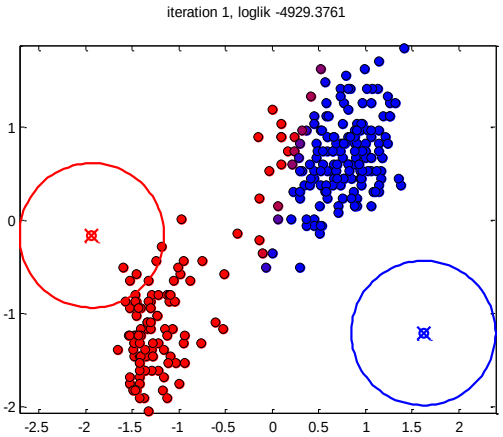
Divide by $\dfrac{1}{(N_j - \dfrac{\sum_{j=1}^{N} (\gamma_j^n)^2}{N_j})}$

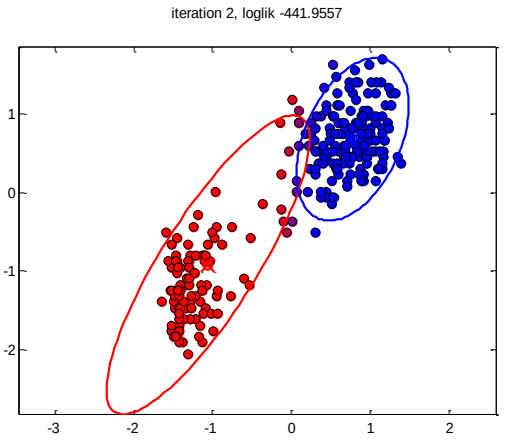
Goes to $1/(N_j - 1)$ for (0-1) case

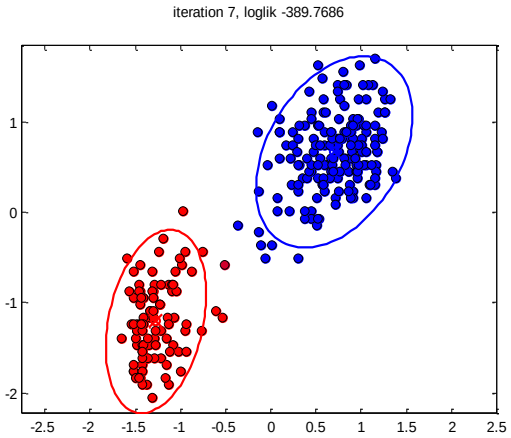
4. Evaluate the negative log likelihood and check for convergence of parameters or the likelihood. If not converged, go to step 2.

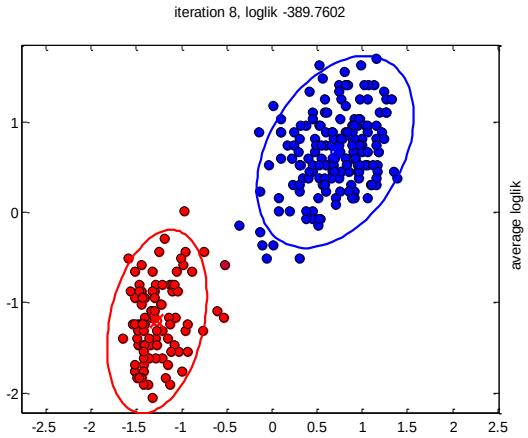
Illustration of EM Algorithm for Gaussian Mixtures

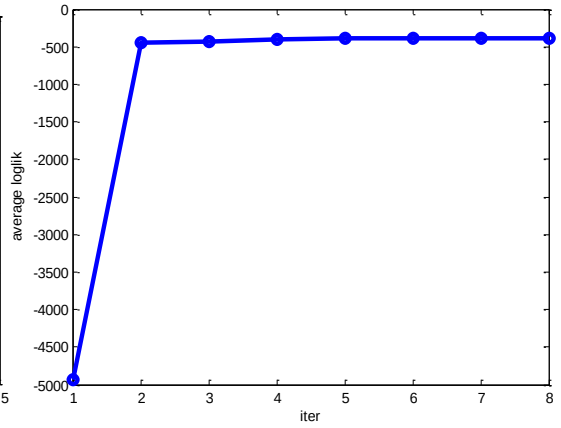












Murphy, Page 353, mixGaussDemoFaithful

Relation of Gaussian Mixtures to K - means

Suppose $\Sigma_j = \varepsilon I$ for $j = 1, 2, .., M$

Then

$$\gamma_j^n = \frac{P_j N(\underline{x}^n; \underline{\mu}_j, \varepsilon I)}{\sum_{k=1}^{M} P_k N(\underline{x}^n; \underline{\mu}_k, \varepsilon I)}; \; j = 1, 2, .., M; n = 1, 2, .., N$$

$$\Rightarrow \qquad \gamma_j^n = \frac{P_j e^{-\|\underline{x}^n - \underline{\mu}_j\|^2 / 2\varepsilon}}{\sum_{k=1}^{M} P_k e^{-\|\underline{x}^n - \underline{\mu}_k\|^2 / 2\varepsilon}}$$

As $\varepsilon \to 0$

$\gamma_j^n \to 1 if \;\; j = \arg\min_k \| \underline{x}^n - \underline{\mu}_k \|$; the rest go to zero as long as none of the P_j is zero.

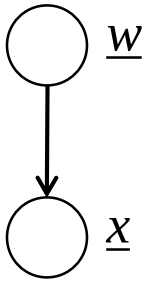
The expected value of negative log likelihood of complete-data is

$$E_{\underline{Z}}\{-\ln p(D_c \mid \underline{\theta})\} = \frac{1}{2\varepsilon} \sum_{n=1}^{N} \sum_{j=1}^{M} \gamma_j^n \| \underline{x}^n - \underline{\mu}_j \|^2 + \text{constant}$$

So, K-means minimizes $\frac{1}{2} \sum_{n=1}^{N} \sum_{j=1}^{M} \gamma_j^n \| \underline{x}^n - \underline{\mu}_j \|^2$

Variational Bayesian Inference - 1

- $\underline{w}$ is a latent vector (continuous or discrete)
  - Mixture vector (discrete), $\underline{z}$
  - Parameters ($\{\mu_j, \Sigma_j, P_j\}$)
- $\underline{x}$ is a p-dimensional random vector
- Recall



Hidden (Latent) Variables

Observation

$$J = -\ln p(\underline{x}) = -\ln L(q(\underline{w})) - KL(q(\underline{w}) \,||\, p(w \,|\, \underline{x}))$$

$$-\ln L(q(\underline{w})) = -\int q(\underline{w}) \ln \left\{ \frac{p(\underline{x}, \underline{w})}{q(\underline{w})} \right\} d\underline{w} = -E_{q(w)} \left(\ln p(\underline{x}, \underline{w}) \right) - H_q(\underline{w})$$

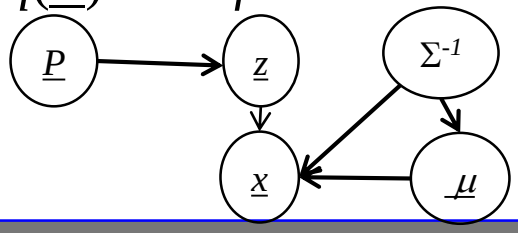
$$KL(q(\underline{w}) \,||\, p(\underline{w} \,|\, \underline{x})) = -\int q(\underline{w}) \ln \left\{ \frac{p(\underline{w} \,|\, \underline{x})}{q(\underline{w})} \right\} d\underline{w} = -E_{q(\underline{w})} \left(\ln p(\underline{w} \,|\, \underline{x}) \right) - H_q(\underline{w})$$

$$J = -\ln p(\underline{x}) \le -\ln L(q(\underline{w})) \because KL(q(\underline{w}) \,||\, p(\underline{w} \,|\, \underline{x})) \ge 0$$

- Variational inference typically assumes $q(\underline{w})$ to be *factorized*

$$q(\underline{w}) = \prod_{j=1}^{K} q_j(\underline{w}_j); \{\underline{w}_j\} \text{ are disjoint groups}$$

Example: $q(\underline{w}) = q(\underline{z})\, q(\{\underline{\mu}_j, \Sigma_j, P_j\})$



Variational Bayesian Inference - 2

- Minimize the upper bound $-\ln L(q(\underline{w}))$ with respect to $q_j(\underline{w}_j)$ **while keeping $\{q_i(\underline{w}_i) : i \neq j\}$ constant** (*a la* Gauss-Seidel)

$$-\ln L(q(\underline{w})) = -\int q(\underline{w}) \ln \left\{ \frac{p(\underline{x}, \underline{w})}{q(\underline{w})} \right\} d\underline{w} = -\int \prod_{i=1}^{K} q_i(\underline{w}_i) \{\ln p(\underline{x}, \underline{w})\} d\underline{w} - \sum_{i=1}^{K} H_{q_i}(\underline{w}_i)$$

$$= -\int q_j(\underline{w}_j) \underbrace{\left\{ \ln p(\underline{x}, \underline{w}) \prod_{\substack{i=1 \\ i \neq j}}^{K} q_i(\underline{w}_i) d\underline{w}_i \right\}}_{E_{i \neq j}[\ln p(\underline{x}, \underline{w})]} d\underline{w}_j - H_{q_j}(\underline{w}_j) - \sum_{\substack{i=1 \\ i \neq j}}^{K} H_{q_i}(\underline{w}_i)$$

$$\frac{\partial[-\ln L(q(\underline{w}))]}{\partial q_j(\underline{w}_j)} = -E_{i \neq j}[\ln p(\underline{x}, \underline{w})] + 1 + \ln\left[q_j(\underline{w}_j)\right] = 0$$

$$\ln\left[q_j(\underline{w}_j)\right] \propto E_{i \neq j}[\ln p(\underline{x}, \underline{w})]$$

$$\Rightarrow q_j(\underline{w}_j) = \frac{e^{E_{i \neq j}[\ln p(\underline{x}, \underline{w})]}}{\int e^{E_{i \neq j}[\ln p(\underline{x}, \underline{w})]} d\underline{w}_j}$$

Log of the optimal q_j is the expectation of the log of joint distribution with respect to all of the other factors $\{q_i(\underline{w}_i) : i \neq j\}$. This idea is used in loopy belief propagation and expectation propagation also.

- Iterative algorithm for finding the factors $\{q_j(\underline{w}_j)\}$

Application to Gaussian Mixtures - 1

- Here $\underline{w}$ involves mixture variables and component parameters

$$q(\underline{w}) = q(\underline{z})\, q(\{\underline{\mu}_j, \Sigma_j, P_j\}_{j=1}^{M})$$

$\underline{z}$ is a binary random vector of dimension M

- Model assumptions

Mixture Distribution: $p(\{\underline{z}^n\}_{n=1}^{N} \mid \{P_j\}_{j=1}^{M}) = \prod_{n=1}^{N} \prod_{j=1}^{M} P_j^{z_j^n}$

Data Likelihood given latent variables:

$$p(\{\underline{x}^n\}_{n=1}^{N} \mid \{\underline{z}^n\}_{n=1}^{N}, \{\underline{\mu}_j, \Sigma_j\}_{j=1}^{M}) = \prod_{n=1}^{N} \prod_{j=1}^{M} [N(\underline{x}^n; \underline{\mu}_j, \Sigma_j)]^{z_j^n}$$

We also assume *priors* on $\{P_j, \underline{\mu}_j, \Sigma_j\}_{j=1}^{M} \Rightarrow$ Bayesian approach

$$p(\underline{P}) = Dirichlet(\underline{P} \mid \underline{\alpha}) = \frac{\Gamma(M\alpha_0)}{\left(\Gamma(\alpha_0)\right)^M} \prod_{j=1}^{M} P_j^{\alpha_0 - 1};$$ conjugate prior to multinomial

$$\Gamma(\alpha) = \int_0^\infty e^{-t} t^{\alpha-1} dt;\ \Gamma(\alpha+1) = \alpha\Gamma(\alpha);\ \Gamma(n) = (n-1)!\ \text{for integers}$$

Application to Gaussian Mixtures - 2

- Model assumptions (continued)

$$p(\{\underline{\mu}_j, \Sigma_j\}_{j=1}^M) = p(\{\underline{\mu}_j\}_{j=1}^M | \{\Sigma_j\}_{j=1}^M) p(\{\Sigma_j\}_{j=1}^M)$$

In one dimension, Gamma
See Bishop's book

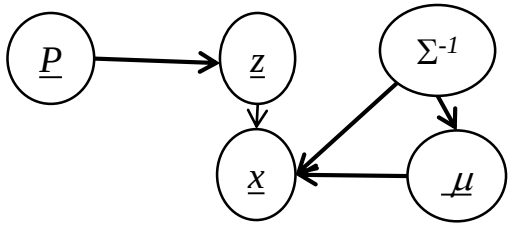
$$= \prod_{j=1}^M N(\underline{\mu}_j; \underline{m}_0, \frac{1}{\beta_0}\Sigma_j). \; Wishart(\Sigma_j^{-1}; \nu_0, W_0)$$

- Joint distribution of all random variables decomposes as follows:

$$p(\{\underline{x}^n\}_{n=1}^N, \{\underline{z}^n\}_{n=1}^N, \{P_j, \underline{\mu}_j, \Sigma_j\}_{j=1}^M) = p(\{\underline{x}^n\}_{n=1}^N | \{\underline{z}^n\}_{n=1}^N, \{\underline{\mu}_j, \Sigma_j\}_{j=1}^M).$$

$$p(\{\underline{z}^n\}_{n=1}^N | \{P_j\}_{j=1}^M). \; p(\underline{P}). p(\{\underline{\mu}_j\}_{j=1}^M | \{\Sigma_j^{-1}\}_{j=1}^M). p(\{\Sigma_j^{-1}\}_{j=1}^M)$$

$$= \left(\prod_{n=1}^N \prod_{j=1}^M [P_j N(\underline{x}^n; \underline{\mu}_j, \Sigma_j)]^{z_j^n} \right).$$



$$\frac{\Gamma(M\alpha_0)}{(\Gamma(\alpha_0))^M} \left(\prod_{j=1}^M P_j^{\alpha_0 - 1}. N(\underline{\mu}_j; \underline{m}_0, \frac{1}{\beta_0}\Sigma_j). \; Wishart(\Sigma_j^{-1}; \nu_0, W_0) \right)$$

Application to Gaussian Mixtures - 3

- Variational Bayes M-step (VBM-step)…. It is easier to see M-step first

$$\ln q(\{P_j, \underline{\mu}_j, \Sigma_j\}_{j=1}^M) = E_{q(\{\underline{z}^n\}_{n=1}^N)} \left(\ln p(\{\underline{x}^n\}_{n=1}^N, \{\underline{z}^n\}_{n=1}^N, \{P_j, \underline{\mu}_j, \Sigma_j\}_{j=1}^M) \right)$$

$$= E_{q(\{\underline{z}^n\}_{n=1}^N)} \left(\begin{array}{l} \ln \prod_{n=1}^N \prod_{j=1}^M [P_j N(\underline{x}^n; \underline{\mu}_j, \Sigma_j)]^{z_j^n} \cdot \\ \frac{\Gamma(M\alpha_0)}{(\Gamma(\alpha_0))^M} \left(\prod_{j=1}^M P_j^{\alpha_0 - 1} . N(\underline{\mu}_j; \underline{m}_0, \frac{1}{\beta_0} \Sigma_j) . \, Wishart(\Sigma_j^{-1}; \nu_0, W_0) \right) \end{array} \right) + cons\,t.$$

$$= \left\{ \sum_{J=1}^M [(\alpha_0 - 1) + \underbrace{\sum_{n=1}^N E\left[z_j^n \right]}_{N_j}] \ln P_j \right\} + \sum_{j=1}^M \left\{ \begin{array}{l} \ln N(\underline{\mu}_j; \underline{m}_0, \frac{1}{\beta_0} \Sigma_j) \\ + \ln Wishart(\Sigma_j^{-1}; \nu_0, W_0) \end{array} \right\}$$

$$+ \sum_{n=1}^N \sum_{j=1}^M \underbrace{E\left[z_j^n \right]}_{\gamma_j^n} \ln N(\underline{x}^n; \underline{\mu}_j, \Sigma_j) + cons\tan t$$

$$q(\{P_j, \underline{\mu}_j, \Sigma_j\}_{j=1}^M) = q(\underline{P})\, q(\{\underline{\mu}_j, \Sigma_j\}_{j=1}^M)$$

$$q(\underline{P}) = Dirichlet(\underline{P}; \{\alpha_0 + N_j = \alpha_j\}_{j=1}^M)$$

$$q(\underline{\mu}_j, \Sigma_j) = Gaussian - Wishart$$

Application to Gaussian Mixtures - 4

- Updated factorized distribution after M-step

$$q(\{P_j, \underline{\mu}_j, \Sigma_j\}_{j=1}^M) = q(\underline{P})\, q(\{\underline{\mu}_j, \Sigma_j\}_{j=1}^M)$$

$$q(\underline{P}) = Dirichlet(\underline{P}; \{\alpha_0 + N_j = \alpha_j\}_{j=1}^M) \Rightarrow E(P_j) = \frac{\alpha_j}{\sum_{k=1}^M \alpha_k} = \frac{\alpha_0 + N_j}{M\alpha_0 + N}$$

$$q(\underline{\mu}_j, \Sigma_j) = Gaussian - Wishart$$

$$= N(\underline{\mu}_j; \underline{m}_j, \frac{1}{\beta_j}\Sigma_j).Wishart(\Sigma_j^{-1}; \nu_j, W_j)$$

$$\beta_j = \beta_0 + N_j;\ N_j = \sum_{n=1}^N \gamma_j^n$$

$$\underline{m}_j = \frac{1}{\beta_j}\left(\beta_0 \underline{m}_0 + N_j \underline{\bar{x}}_j\right);\ \underline{\bar{x}}_j = \frac{1}{N_j}\sum_{n=1}^N \gamma_j^n \underline{x}^n$$

$$W_j^{-1} = W_0^{-1} + N_j S_j + \frac{\beta_0 N_j}{\beta_0 + N_j}(\underline{\bar{x}}_j - \underline{m}_0)(\underline{\bar{x}}_j - \underline{m}_0)^T$$

where $S_j = \frac{1}{N_j}\sum_{n=1}^N \gamma_j^n (\underline{x}^n - \underline{\bar{x}}_j)(\underline{x}^n - \underline{\bar{x}}_j)^T$

$$\nu_j = \nu_0 + N_j$$

Updates for $\{N_j, \underline{\bar{x}}_j, S_j\}$ are similar to ML

Sequential VBEM?

Application to Gaussian Mixtures - 5

- Variational Bayes E-step (VBE-step)

$$\ln q(\{\underline{z}^n\}_{n=1}^N) = E_{q(\{P_j,\underline{\mu}_j,\Sigma_j\}_{j=1}^M)}\left(\ln p(\{\underline{x}^n\}_{n=1}^N,\{\underline{z}^n\}_{n=1}^N,\{P_j,\underline{\mu}_j,\Sigma_j\}_{j=1}^M)\right)$$

$$= E_{q(\{P_j,\underline{\mu}_j,\Sigma_j\}_{j=1}^M)}\left(\ln \prod_{n=1}^N\prod_{j=1}^M [P_j N(\underline{x}^n;\underline{\mu}_j,\Sigma_j)]^{z_j^n}\right) + cons\tan t$$

$$= E_{q(\{\underline{\mu}_j,\Sigma_j\}_{j=1}^M)} \sum_{n=1}^N\sum_{j=1}^M z_j^n \left(\ln \underbrace{p(\{\underline{x}^n\}_{n=1}^N \mid \{\underline{z}^n\}_{n=1}^N,\{\underline{\mu}_j,\Sigma_j\}_{j=1}^M)}_{N(\underline{x}^n;\underline{\mu}_j,\Sigma_j)}\right) +$$

$$E_{q\{\underline{P}\}}\left(\ln \underbrace{p(\{\underline{z}^n\}_{n=1}^N \mid \{P_j\}_{j=1}^M)}_{P_j}\right) + cons\tan t$$

$$= \sum_{n=1}^N\sum_{j=1}^M z_j^n \ln \rho_j^n$$

where $\ln\rho_j^n = E_{P_j}\left[\ln P_j\right] + \frac{1}{2}E_{\Sigma_j^{-1}}\left[\ln|\Sigma_j^{-1}|\right] - \frac{p}{2}\ln(2\pi) - \frac{1}{2}E_{\underline{\mu}_j,\Sigma_j^{-1}}\left(\|\underline{x}^n - \underline{\mu}_j\|^2_{\Sigma_j^{-1}}\right)$

$$\Rightarrow q(\{\underline{z}^n\}_{n=1}^N) = \prod_{n=1}^N\prod_{j=1}^M [\gamma_j^n]^{z_j^n} \text{ where } \gamma_j^n = \frac{\rho_j^n}{\sum_{k=1}^M \rho_k^n} \text{.... responsibilities}$$

Application to Gaussian Mixtures - 6

- Variational Bayes E-step (VBE-step) … continued
  - Evaluation of responsibilities
  - Recall

$$\ln \rho_j^n = E_{P_j}\left[\ln P_j\right] + \frac{1}{2} E_{\Sigma_j^{-1}}\left[\ln |\Sigma_j^{-1}|\right] - \frac{p}{2}\ln(2\pi) - \frac{1}{2} E_{\underline{\mu}_j,\Sigma_j^{-1}}\left(\| \underline{x}^n - \underline{\mu}_j \|^2_{\Sigma_j^{-1}}\right)$$

$$E_{P_j}\left[\ln P_j\right] = \psi(\alpha_j) - \psi(\sum_{k=1}^{M}\alpha_k); \psi(\alpha) = \frac{d}{d\alpha}\ln\Gamma(\alpha) \ldots . \textit{digamma function}$$

$$E_{\Sigma_j^{-1}}\left[\ln |\Sigma_j^{-1}|\right] = \sum_{i=1}^{p}\psi(\frac{\nu_j + 1 - i}{2}) + p\ln 2 + \ln |W_j|$$

See Bishop Chapter 10

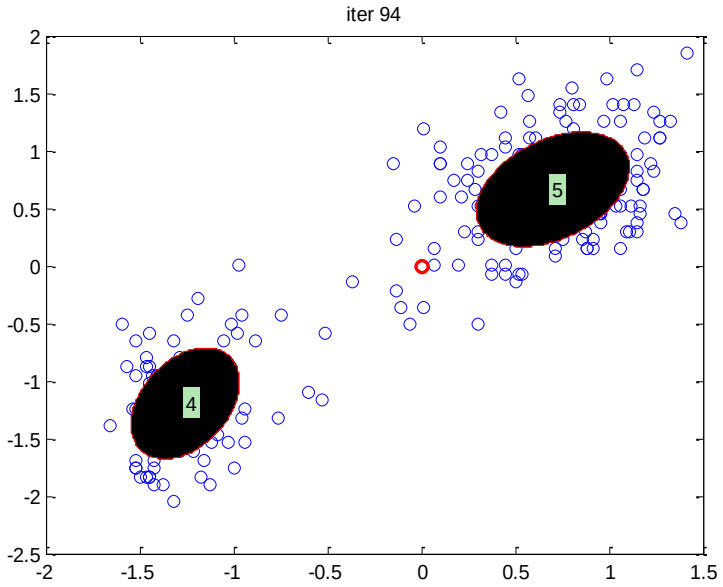
$$E_{\underline{\mu}_j,\Sigma_j^{-1}}\left(\| \underline{x}^n - \underline{\mu}_j \|^2_{\Sigma_j^{-1}}\right) = \frac{p}{\beta_j} + \nu_j (\underline{x}^n - \underline{m}_j)^T W_j (\underline{x}^n - \underline{m}_j)$$

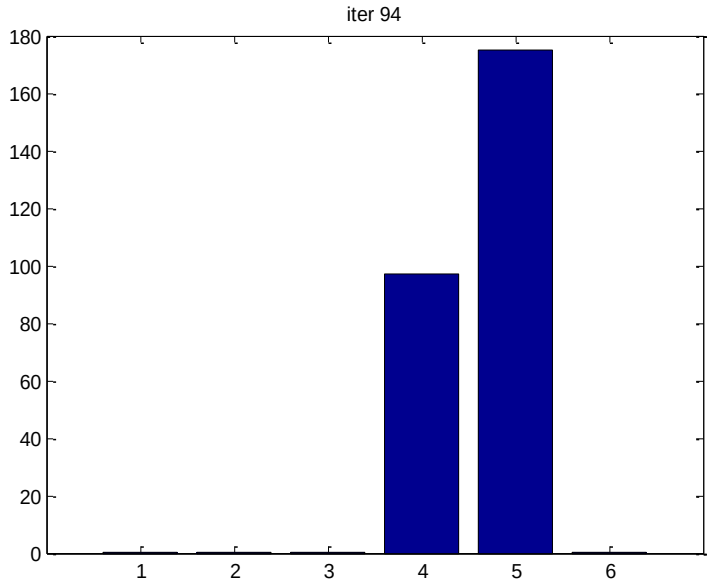
Since $\gamma_j^n \propto \rho_j^n$

$$\gamma_j^n \propto |W_j| \exp\left(\psi(\alpha_j) + \frac{1}{2}\sum_{i=1}^{p}\psi(\frac{\nu_j + 1 - i}{2}) - \frac{p}{2\beta_j} - \frac{\nu_j}{2}(\underline{x}^n - \underline{m}_j)^T W_j (\underline{x}^n - \underline{m}_j)\right)$$

VB Approximation of Gaussian Gamma

- In VBEM start with large M and very small $\alpha_0 << 1$ (≈ 0.001)
- It automatically prunes clusters with very few members ("rich get richer")
- In this example, we start with 6 clusters, but only 2 remain at the end





mixGaussVbDemoFaithful from Murphy, Page 755

Bayesian Model Selection -1

M = Set of models (e.g., linear, quadratic discriminants)

Model m is specified by parameter vector $\underline{\theta}_m$; $m \in M$

- Bayesian Model Selection (max . prob. of model given data)

$$P(m \mid D) = \frac{p(D \mid m)\, p(m)}{\sum_l p(D \mid l)\, p(l)}$$

- Bayes Factors for Comparing Models m and l

$$BF(m,l) \triangleq \frac{p(D \mid m)}{p(D \mid l)} = \frac{p(m \mid D)}{p(l \mid D)} \Big/ \left(\frac{p(m)}{p(l)}\right) = \left(\frac{e^{-\frac{1}{2}BIC(m)}}{e^{-\frac{1}{2}BIC(l)}}\right) \Big/ \left(\frac{p(m)}{p(l)}\right)$$

$BF(m,l) > 1 \Rightarrow$ model m *is preferred over* model l

$BIC =$ *Bayesian Information Criterion* (*using negative* log *likelihood*)

Bayesian Model Selection -2

- Bayesian Information Criterion (BIC)… minimize BIC

$BIC \triangleq -2 \ln p(D \mid \hat{\underline{\theta}}_m) + dof(\hat{\underline{\theta}}_m) \ln N$ *Schwarz criterion*

Valid for regression, classification and density estimation.

For linear and quadratic classifiers

$$dof(\hat{\underline{\theta}}_m) = \begin{cases} C + Cp; \text{ equal \& speherical covaraiance (Note: only } (C\text{-}1) \text{ probabiities)} \\ C + Cp + p - 1; \text{ equal and spherical feature-dependent covaraiance} \\ C + Cp + p(p+1)/2 - 1; \text{ equal and general covaraiance} \\ C + Cp + Cp(p+1)/2 - 1; \text{ unequal and general covaraianc} \end{cases}$$

BIC is closley related to *Minimum Description Length* (*MDL*)

Adjusted BIC : $\ln N \rightarrow \ln[(N+2)/24]$

- Akaike Information Criterion (AIC) …. Minimize AIC

$$AIC \triangleq -2 \ln p(D \mid \hat{\underline{\theta}}_m) + 2dof(\hat{\underline{\theta}}_m)$$

$$AIC_{small\ N\&Gaussian} = AIC + \frac{2dof(\hat{\underline{\theta}}_m)(dof(\hat{\underline{\theta}}_m)+1)}{N - dof(\hat{\underline{\theta}}_m) - 1}$$

Binary Classification, BIC & AIC - 1

- Class $z = 0$: $N(0,1)$;zero mean … Null hypothesis, H_0
- Class z = 1: $N(\mu,1)$; $\mu \neq 0$ …. Alternative hypothesis, H_1

  N *scalar data points* : $D = \{x^1, x^2, ..., x^N\}$

- Under null hypothesis, sample mean $\bar{x} \sim N(0, \frac{1}{N}) \Rightarrow \sqrt{N}\bar{x} \sim N(0,1)$
- Note $\bar{x} = \bar{x} - \mu + \mu$.
- So, under H_1: $\sqrt{N}\left(\bar{x} - \mu\right) \sim N(0,1)$

If $P(\sqrt{N}\,|\bar{x}| > c) > 1-\alpha$ *for a specified* c (*e.g.*, $c = 2$ *for* $\alpha = 0.05$), *we are confident that* $\mu \neq 0$ *with probability* $> 1-\alpha$.

α *is the probability of faslsely rejecting* H_0.

So, test statistic is : $|\bar{x}| > \frac{c}{\sqrt{N}}$

Binary Classification, BIC & AIC - 2

BIC

$$BIC \triangleq -2\ln p(D \mid \underline{\hat{\theta}}_m) + dof(\hat{\theta}) \ln N$$

$$BIC(H_0) = \sum_{i=1}^{N} \left(x^i\right)^2$$

$$BIC(H_1) = \sum_{i=1}^{N} \left(x^i - \bar{x}\right)^2 + \ln N = \sum_{i=1}^{N} \left(x^i\right)^2 - N\left(\bar{x}\right)^2 + \ln N$$

$$So, BIC(H_1) < BIC(H_0)\ if\ \left(\bar{x}\right)^2 > \frac{\ln N}{N} \Rightarrow |\bar{x}| > \sqrt{\frac{\ln N}{N}}$$

sample number-dependent threshold

AIC

$$AIC \triangleq -2\ln p(D \mid \underline{\hat{\theta}}_m) + 2.dof(\hat{\theta})$$

$$AIC(H_0) = \sum_{i=1}^{N} \left(x^i\right)^2$$

$$AIC(H_1) = \sum_{i=1}^{N} \left(x^i - \bar{x}\right)^2 + 2 = \sum_{i=1}^{N} \left(x^i\right)^2 - N\left(\bar{x}\right)^2 + 2$$

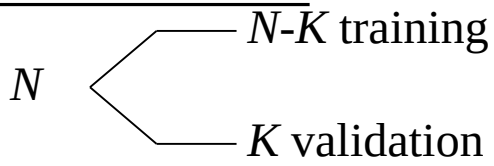
$$So, AIC(H_1) < AIC(H_0)\ if\ \left(\bar{x}\right)^2 > \frac{2}{N} \Rightarrow |\bar{x}| > \sqrt{\frac{2}{N}}$$

$$\Rightarrow Similar\ to\ classical\ hypothesis\ testing; c = \sqrt{2}$$

Holdout Method of Cross Validation

Probability of error, $PE = \mathrm{P}\{\alpha \neq z\}$ misclassification rate.

- Holdout Method:



Typically, $K \approx N/5$

Error Count = R out of $K \Rightarrow PE=R/K$

From Binomial Distribution: $\sigma_{PE} = \sqrt{\frac{PE(1-PE)}{K}}$

To obtain an estimate of PE within 1%: $0.01 = 2\sqrt{\frac{PE(1-PE)}{K}}$

If $PE \approx 0.05$, $K = \frac{4\left(\sqrt{PE(1-PE)}\right)^2}{10^{-4}} = 40000(PE(1-PE)) = 1900$

When $PE \approx ½$, we need lots of samples…. 10,000

Are there better bounds?

We can also estimate Class Conditional Error Rate, PE(z=k). Then

$$PE = \sum_{k=1}^{c} P(z=k).\,\mathrm{PE}(z=k)$$

Markov & Chebyshev Inequalities

- Suppose have a *nonnegative* rv, x (discrete or continuous)
- Assume continuous WLOG
- Markov inequality

$$E(x) = \int_0^\infty xf(x)dx = \int_0^\varepsilon xf(x)dx + \int_\varepsilon^\infty xf(x)dx \geq \int_\varepsilon^\infty xf(x)dx \geq \varepsilon \int_\varepsilon^\infty f(x)dx = \varepsilon P(x \geq \varepsilon)$$

$$\Rightarrow P(x \geq \varepsilon) \leq \frac{E(x)}{\varepsilon}$$

- Chebyshev inequality

$$P(x \geq \varepsilon) \leq \frac{E(x)}{\varepsilon} \Rightarrow P(x^2 \geq \varepsilon^2) \leq \frac{E(x^2)}{\varepsilon^2}$$

$$so,\ P(|x-\mu| \geq \varepsilon) \leq \frac{\sigma^2}{\varepsilon^2}$$

$$Example: x = sample\ mean\ of\ \ n\ numbers, m$$

$$P(|m-\mu| \geq \varepsilon) \leq \frac{\sigma^2}{n\varepsilon^2} \Rightarrow P(\frac{|m-\mu|}{\sigma} \geq \varepsilon) \leq \frac{1}{n\varepsilon^2}$$

$$For\ binary\ classifier\ with\ unknown\ \ probability\ of\ \ error\ P_e\ and\ sample\ error, S_e$$

$$P(|S_e - P_e| \geq \varepsilon) \leq \frac{P_e(1-P_e)}{n\varepsilon^2} \leq \frac{1}{4n\varepsilon^2};\quad n = 100, \varepsilon = 0.2, bound = 0.0625....loose$$

Hoeffding's Inequality

$$Let\ Y = \sum_{i=1}^{n} X_i ; X_i \in [a_i, b_i]; R_i = b_i - a_i$$

$$P\{|Y - E(Y)| \geq n\varepsilon\} \leq 2e^{-2n^2\varepsilon^2 / \sum_{i=1}^{n} R_i^2} ;\ \text{when } R_i = b_i - a_i = 1 \Rightarrow P\{|Y - E(Y)| \geq n\varepsilon\} \leq 2e^{-2n\varepsilon^2}$$

$$Let\ Z_i = X_i - E(X_i) \Rightarrow E(Z_i) = 0\ \&\ Z_i \in [-\frac{R_i}{2}, \frac{R_i}{2}]$$

$$P\{|Y - E(Y)| \geq n\varepsilon\} = P\{|\sum_{i=1}^{n} Z_i| \geq n\varepsilon\} = P\{t\,|\sum_{i=1}^{n} Z_i| \geq tn\varepsilon\} = P\{t\sum_{i=1}^{n} Z_i \geq tn\varepsilon\} + P\{t\sum_{i=1}^{n} Z_i \leq -tn\varepsilon\}$$

$$P\{t\sum_{i=1}^{n} Z_i \geq tn\varepsilon\} = P\{e^{t\sum_{i=1}^{n} Z_i} \geq e^{tn\varepsilon}\} \leq e^{-tn\varepsilon} E\{e^{t\sum_{i=1}^{n} Z_i}\} \ldots Markov\ Inequality$$

$$e^{-tn\varepsilon} E\{e^{t\sum_{i=1}^{n} Z_i}\} = e^{-tn\varepsilon} E\{\prod_{i=1}^{n} e^{tZ_i}\} = e^{-tn\varepsilon} E\{\prod_{i=1}^{n} e^{t[\alpha_i \frac{R_i}{2} - (1-\alpha_i)\frac{R_i}{2}]}\} = e^{-tn\varepsilon} \prod_{i=1}^{n} E\{e^{t[\alpha_i \frac{R_i}{2} - (1-\alpha_i)\frac{R_i}{2}]}\}; \alpha_i = \frac{Z_i + R_i/2}{R_i}$$

$$\leq e^{-tn\varepsilon} \prod_{i=1}^{n} E\{\alpha_i e^{t\frac{R_i}{2}} + (1-\alpha_i) e^{-t\frac{R_i}{2}}\} = e^{-tn\varepsilon} \prod_{i=1}^{n} \frac{1}{2}\left[e^{tR_i/2} + e^{-tR_i/2}\right]$$

Jensen's inequality

$$So, P\{t\sum_{i=1}^{n} Z_i \geq tn\varepsilon\} \leq e^{-tn\varepsilon} \prod_{i=1}^{n} \frac{1}{2}\left[e^{tR_i/2} + e^{-tR_i/2}\right] \leq e^{-tn\varepsilon} \prod_{i=1}^{n} e^{t^2 R_i^2/8} = e^{[t^2 \sum_{i=1}^{n} R_i^2/8 - tn\varepsilon]}$$

https://en.wikipedia.org/wiki/Hoeffding%27s_lemma

$$RHS\ is\ \min imized\ when\ t = 4n\varepsilon / \sum_{i=1}^{n} R_i^2 \Rightarrow P\{|\sum_{i=1}^{n} Z_i| \geq n\varepsilon\} \leq 2e^{-2n^2\varepsilon^2 / \sum_{i=1}^{n} R_i^2} = 2e^{-2n\varepsilon^2}\ when\ R_i = 1$$

Rationale behind Cross Validation

- Suppose we have classifiers/models $M_1, M_2, \ldots, M_C$
- Training Data, D & Validation Data, V
- Samples are assumed to be i.i.d.

Average log *likelihood* $\bar{l}_m \triangleq \frac{1}{|V|}\sum_{\underline{x}^i \in V} \ln p(\underline{x}^i \mid \underline{\hat{\theta}}_m)$

$$P\{\max_m |\bar{l}_m - l_m| > \varepsilon\}$$
$$= P\{\cup_m |\bar{l}_m - l_m| > \varepsilon\}$$
$$\leq \sum_{m=1}^{C} P\{|\bar{l}_m - l_m| > \varepsilon\} \leq 2Ce^{-2|V|\varepsilon^2}$$

Since $\underline{\hat{\theta}}_m$ *does not depend on* V, $E\{\bar{l}_m\} = l_m$

By Hoeffding's inequality $P\{\max_m |\bar{l}_m - l_m| > \varepsilon\} \leq 2Ce^{-2|V|\varepsilon^2}$

So, if $2Ce^{-2|V|\varepsilon^2} = \alpha$, $\varepsilon = \sqrt{\frac{\ln(\frac{2C}{\alpha})}{2|V|}} \Rightarrow \varepsilon \propto \sqrt{\frac{\ln C}{|V|}}$

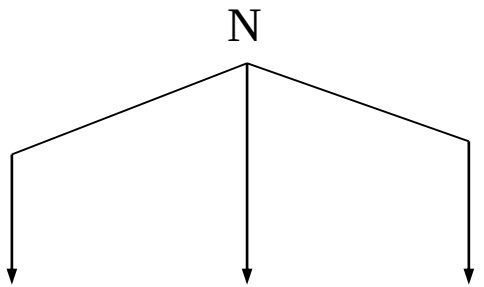
"*Confidence* (α) *is cheap, but accuracy* (ε) *is more* exp *ensive*"

$$\Rightarrow \max_m \bar{l}_m \geq \max_m l_m - 2\sqrt{\frac{\ln(\frac{2C}{\alpha})}{2|V|}} = \max_m l_m - O(\frac{\ln C}{|V|})$$

So, with probability of at least $(1-\alpha)$, *one chooses the best* mod *el.*

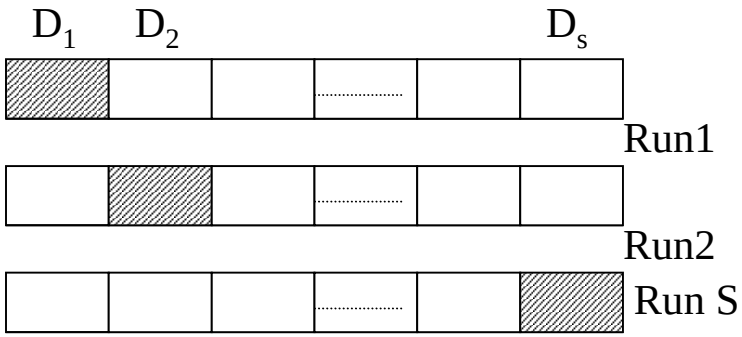
S-fold Cross Validation

- *S*-fold Cross validation:



Testing $\frac{N}{S}$ $\frac{N}{S}$ $\frac{N}{S}$

Validation $\frac{S-1}{S}N$ $\frac{S-1}{S}N$ $\frac{S-1}{S}N$



S is typically 5-10

$$PE = \frac{1}{N}\sum_{s=1}^{S}\sum_{i\in D_s} I(\alpha(\underline{x}_i) \neq z_i) \qquad I(.) = \textit{ indicator function}$$

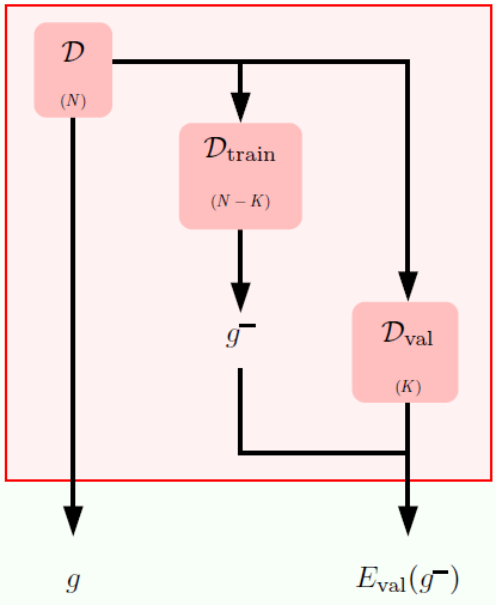
- $S=N \Rightarrow N$-fold cross validation or Leave one-out CV (LOOCV) method

$$PE = \frac{1}{N}\sum_{i=1}^{N} I(\alpha(\underline{x}_i) \neq z_i);\ \alpha(\underline{x}) = f(\underline{x}, D_{-i})$$

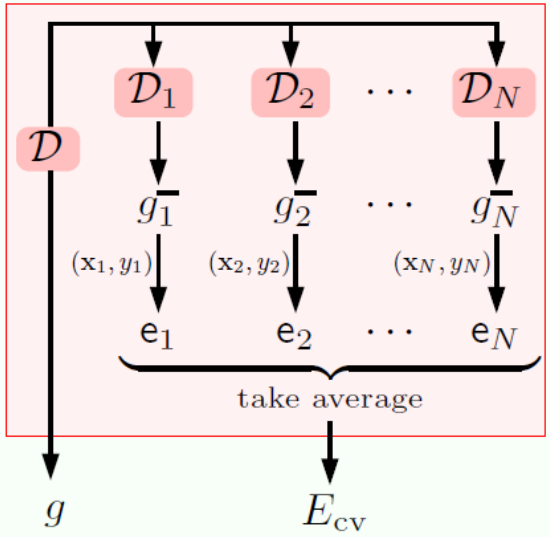
- Practical Scheme: 5x2 Cross-Validation. Variation: 5 repetitions of 2-fold cross-validation on a randomized dataset

Summary of Validation and Model Selection Methods

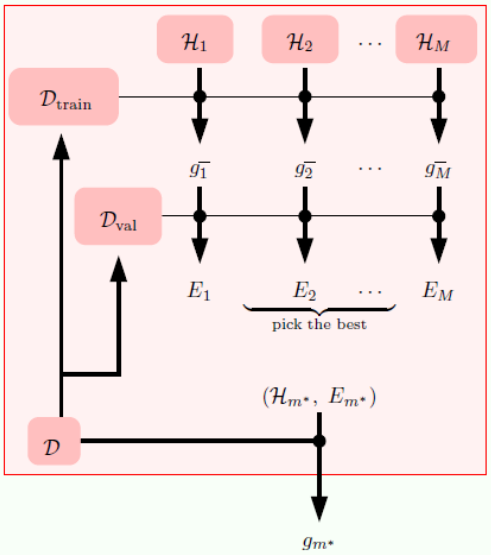
Simple split



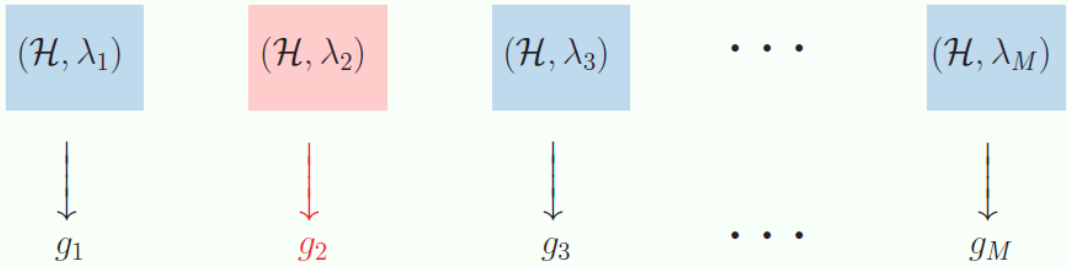
Leave one out CV
N=S for S-fold CV



Model Selection



Regularization and Model Selection



Any combination of CV, Model Selection and Regularization (hyper-parameter selection) is possible.
See AMLbook.com .

Bootstrap Method of Validation

- Bootstrap Method:

$$D = \left\{ \underline{x}^1 \; \underline{x}^2 \ldots\ldots, \underline{x}^N \right\}$$

Sample from D with uniform probability $(1/N)$. Each $\underline{x}_i$ is drawn independently *with replacement*

Let b= bootstrap index, $b=1,2,\ldots,B$

B= number of bootstrap samples (typically 50-200)

Do b=1,2,..,B

Bootstrap Sample $D_b = \left\{ \underline{x}^{(1)} \; \underline{x}^{(2)} \ldots\ldots, \underline{x}^{(N)} \right\}_b \Rightarrow PE_{training}(b)$

$Validation\ Data : A_b = D \setminus D_b \Rightarrow samples\ \underline{not}\ in\ bootstrap \Rightarrow PE_{val}(b)$

End

$$PE = 0.632 * PE_{training}(b) + 0.368 * PE_{val}(b)$$

Effron Estimator

Confusion Matrix

- Bootstrap Method (cont'd):

Why?

$$P\{observation \notin bootstrap\ sample\} = (1 - \frac{1}{N})^N \rightarrow \frac{1}{e}\ as\ N \rightarrow \infty$$

$$\Rightarrow P\{observation \in validation\ samples\} = 0.368$$

$$\Rightarrow P\{observation \in training\ samples\} = 0.632$$

$$\Rightarrow PE = 0.632 * PE_{training}(b) + 0.368 * PE_{val}(b)$$

- Confusion Matrix: $P = [P_{ij}]$

$$P_{ij} = P\{decision\ \alpha = i \mid z = j\} = \frac{N_{ij}}{N_j}; i = 0,1,2,...,C;\ j = 1,2,...,C$$

In words, just count errors from validation set, bootstrap, etc. for class *j* and divide by the number of samples from class *j*

Performance Metrics for Binary Classification - 1

- Contingency table showing the differences between the **true** and **predicted** classes for a set of labeled examples
- The following metrics can be derived from the confusion matrix:
  - P_D (Sensitivity, Recall):
    - TP/(TP+FN)
  - P_F (1-Selectivity):
    - FP/(TN+FP)
  - Positive Prediction Power (Precision)
    - TP/(TP+FP)
  - Negative Prediction Power
    - TN/(TN+FN)
  - Correct Classification Rate (CCR)
    - (TP+TN)/N
  - Misclassification Rate
    - (FP+FN)/N
  - Odds-ratio
    - (TP*TN)/(FP*FN)

No reject option

TRUE

PREDICTED

Outcome	Fault	No-Fault	Total
Positive Detection	Number of detected faults (TP)	Number of false-alarms (FP)	Total number of positive detections
Negative Detection	Number of missed faults (FN)	Number of correct rejections. (TN)	Total number of negative detections
	Total number of faulty samples	Total number of fault-free samples	Total number of samples

- Kappa

$$\frac{CCR - \sum_{i=1}^{2} P_{row(i)} P_{col(i)}}{1 - \sum_{i=1}^{2} P_{row(i)} P_{col(i)}}$$

When doing this,
All four entries should sum to 1

CCR = Correct Classification Rate

$P_{row(i)}$=% entries in row i

$P_{col(i)}$=% entries in column i

Poor: K < 0.4

Good: 0.4 < K < 0.75

Excellent: K > 0.75

Performance Metrics for Binary Classification - 2

Aircraft Engine Data

Outcome	Fault	No-Fault	Total
Positive Detection	3023 (TP)	1518 (FP)	Total number of positive detections
Negative Detection	1977 (FN)	3482 (TN)	Total number of negative detections
	Total number of faulty samples	Total number of fault-free samples	Total number of samples

$$Kappa = \frac{2P_1P_0(P_D - P_F)}{P_1 + (P_0 - P_1)(P_1P_D + P_0P_F)}$$
$$= (P_D - P_F) \text{ if } P_1 = P_0$$

Normalized

0.605	0.304
0.395	0.696

Metrics

- $P_D = 0.605 \Rightarrow$ False Neg. Rate = 0.395
- $P_F = 0.304$ (False Positive Rate)
- Correct Classification Rate = 0.65
- Misclassification Rate = 0.35
- Odds Ratio = 3.51
- Kappa = 0.301$\Rightarrow$Poor

- Positive Prediction Power = 0.666
- Negative Prediction Power = 0.638
- Prevalence = 0.5 (Priors)

$\text{Re}\,call(R) = 0.605 \qquad P_D = R$

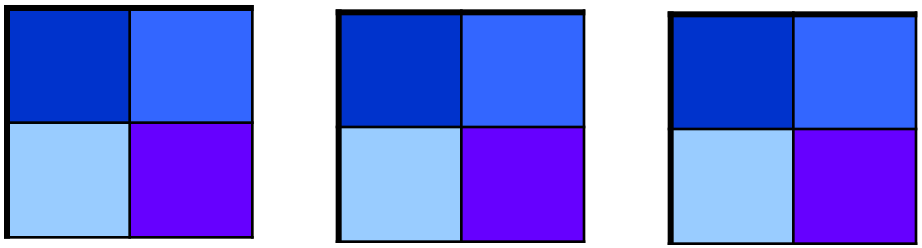
$\text{Pr}\,ecision(P) = 0.666 \qquad P_F = R(1-P)/P = 0.304$

$$F\ score = \frac{2PR}{P+R} = 0.634$$

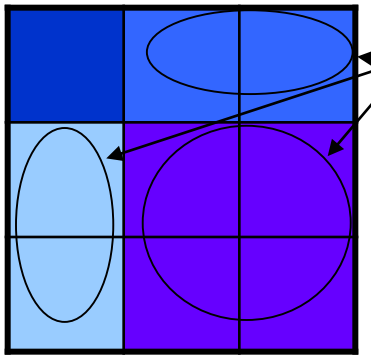
Confusion Matrices for Multiple Classes - 1

Confusion Matrix for C Classes

- **One-versus-One**
  - Generates C(C-1)/2 confusion matrices
    - C1 vs. C2
    - C1 vs. C3
    - C2 vs. C3
- **One-versus-All**
  - Generates C confusion matrices
    - C1 vs. C2 & C3
    - C2 vs. C1 & C3
    - C3 vs. C1 & C2



One-versus-One Confusion Matrices

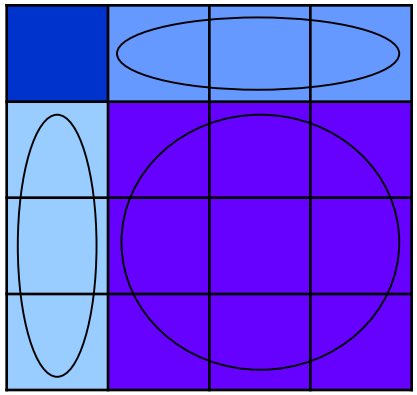


One-versus-All Confusion Matrix

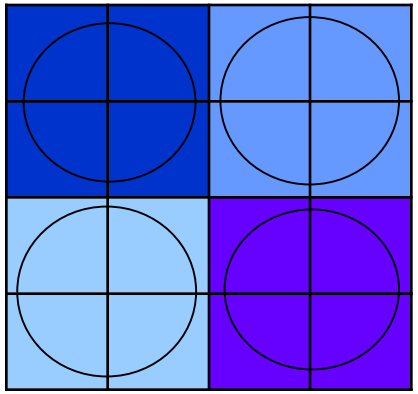
Confusion Matrices for Multiple Classes - 2

- **Some-versus-Rest**
  - Generates $2^{(C-1)}$ - 1 confusion matrices
  - Both true and false classifications may be sums
  - Here is a four class example

C1	vs. C2 & C3 & C4
C2	vs. C1 & C3 & C4
C1 & C2	vs. C3 & C4
C3	vs. C1 & C2 & C4
C1 & C3	vs. C2 & C4
C2 & C3	vs. C1 & C4
C1 & C2 & C3	vs. C4



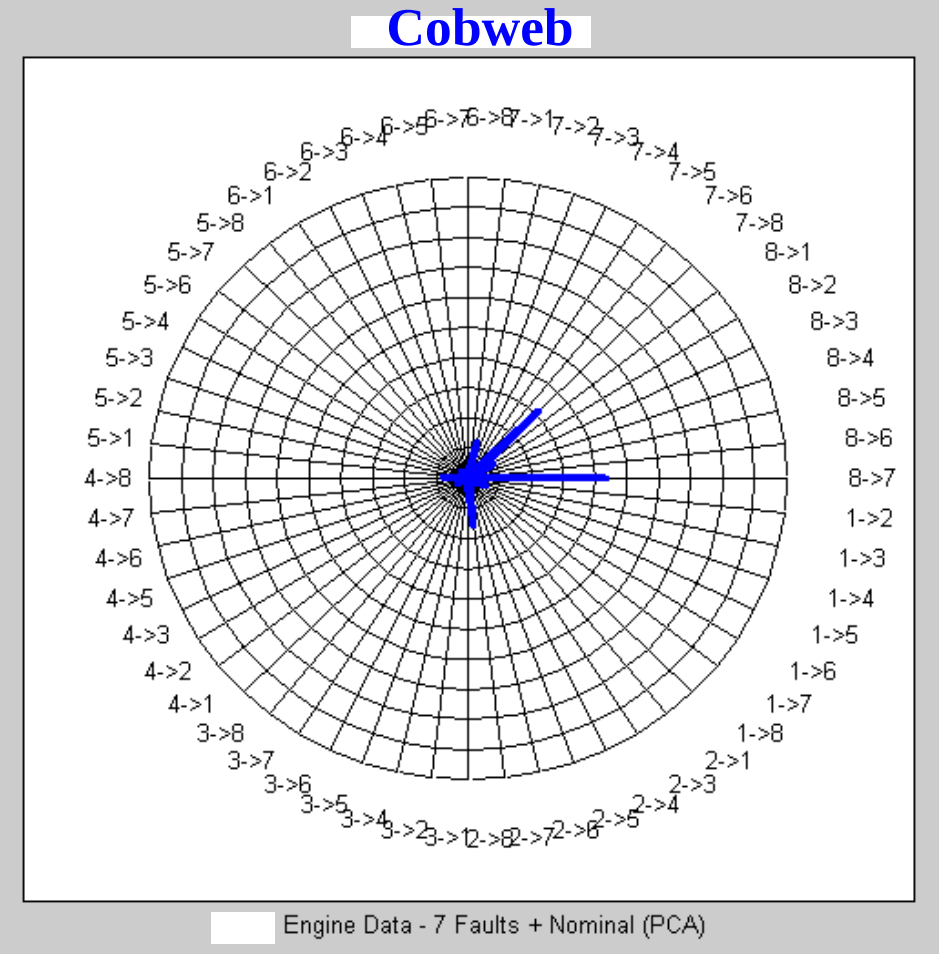
C1 vs. C2 & C3 & C4



C1 & C2 vs. C3 & C4

Can form the basis for code book based classifiers

- **Cobweb**
  - Illustrates the probability that each class will be predicted incorrectly (the off diagonal cells of the confusion matrix)
  - Shows relative performance between classes for each classifier
  - High performance classifiers have poor visibility in cobweb
  - May be difficult to interpret for high numbers of classes
    - c(c-1)/2 rays



Other Metrics of Performance Assessment

- Fawcett's Extension

$$AUCF = \sum_{i=1}^{C} P(z=i)AUC(i, rest); AUC = Area\ under\ the\ ROC\ Curve$$

  - Sums the areas under the curves for **each class** versus **the rest**, multiplied by the probability of that class

- Hand and Till Function

$$HT = \frac{2}{C(C-1)} \sum_{i=1}^{C} \sum_{j=1}^{i-1} AUC(i, j)$$

  - Averages the areas under the curves for each pair of classes

- Macro-Average Modified

$$MAVG_{MOD} = 0.75\left(\sum_{i=1}^{C} P_{ii}\right) + 0.25\left(\sqrt[C]{\prod_{i=1}^{C} P_{ii}}\right)$$

  - Uses both the **geometric mean** and **average correct classification rate**

- **Comparing Classifiers**

Suppose have Classifiers *A* and *B*

n_A= number of *errors made by A but not by B*

n_B= number of *errors made by B but not by A*

McNemar's Test: *Check if* $\frac{|n_A - n_B| - 1}{\sqrt{n_A + n_B}} \approx N(0,1)$

Need $|n_A - n_B| > 5$ *for a significant difference*

To detect 1% difference in error rates, need at least 500 samples

References on Performance Assessment

1. Alpaydin, Ethem. Introduction to Machine Learning. MIT Press, Cambridge. 2004.
2. Kuncheva, Ludmila I. Combining Pattern Classifiers; Methods and Algorithms. John Wiley and Sons: Hoboken, NJ. 2004.
3. Ferri, C. "Volume Under the ROC Surface for Multi-class Problems." Univ. Politecnica de Valencia, Spain. 2003.
4. D. Mossman. "Three-way ROCs", *Medical Decision Making*, 19(1):78–89, 1999.
5. A. Patel, M. K. Markey, "Comparison of three-class classification performance metrics: a case study in breast cancer CAD", *Medical Imaging 2005: Image Processing*.
6. Ferri C., Hernandez J., Salido M. A., "Volume Under the ROC Surface for Multiclass Problems. Exact Computation and Evaluation of Approximations" Technical Report DSOC. Univ. Politec. Valencia. 2003. http://www.dsic.upv.es/users/elp/cferri/VUS.pdf.
7. Mooney, C.Z. and R.D. Duval, 1993, Bootstrapping: A Non-Parametric Approach to Statistical Inference. Newbury Park, CA: Sage Publications.
8. J. K. Martin and D. S. Hirschberg. "Small sample statistics for classification error rates, I: error rate measurements." Technical Report 96-21, Dept. of Information & Computer Science, University of California, Irvine, 1996.

Summary

- Estimating Parameters of Densities From Data
  - Maximum Likelihood Methods
  - Bayesian Learning
- Probability Density Estimation
  - Histogram Methods
  - Parzen Windows
  - Probabilistic Neural Network
  - *k*-nearest Neighbor Approach
- Mixture Models
  - Estimate parameters via EM and VBEM algorithm
  - Various interpretations of EM
- Performance Assessment of Classifiers