Lectures 4 and 5: Plug-in Classifiers, PNN, KNN, Logistic Regression and 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

Spring 2021
February 16, 2021 & February 23, 2021

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

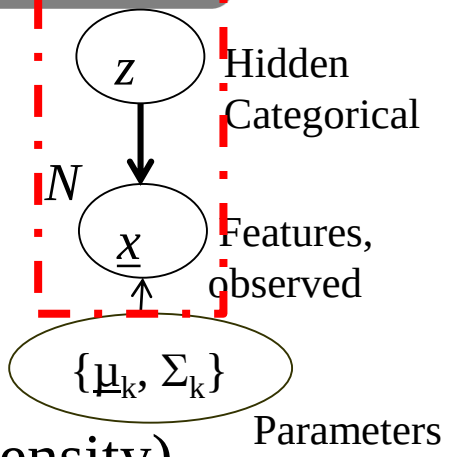
- Estimating Parameters of Densities From Data
  - Maximum Likelihood Methods
  - Bayesian Learning

"Plug-in" Classifiers

- Estimating Probability Densities (Nonparametric)
  - Histogram Methods
  - Parzen Windows
  - Probabilistic Neural Network
  - *k*-nearest Neighbor Approach
- Discriminant Learning: Logistic Regression
- 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 (Later Lectures)
    - 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.

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!

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 to track "non-stationary" means

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

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

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

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

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

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

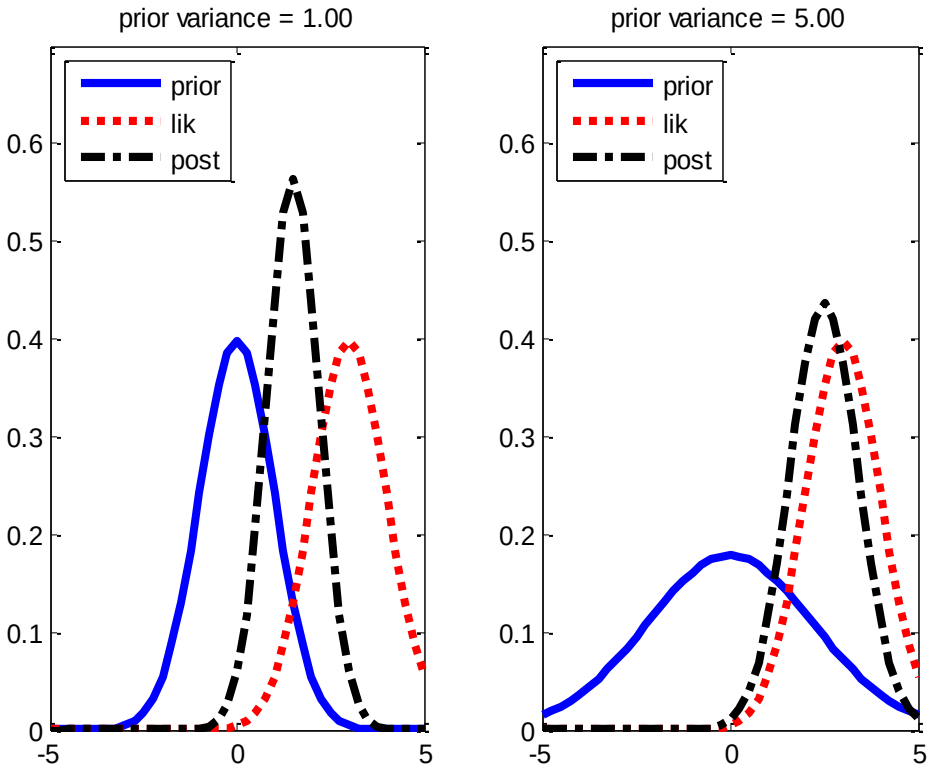
Tough to compute unless reproducing density. Need Simulation.

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

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

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

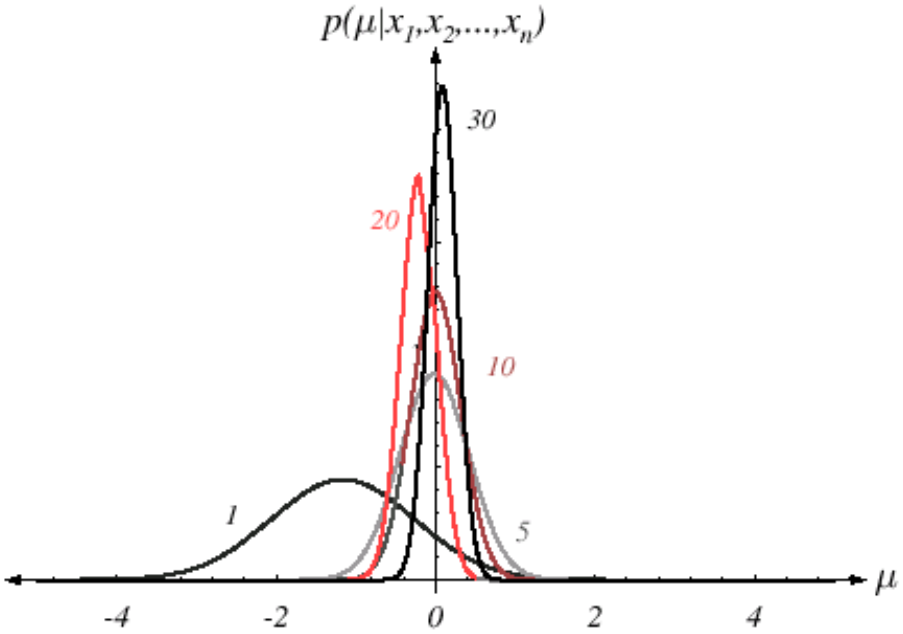
Illustration of Bayesian Learning - 1

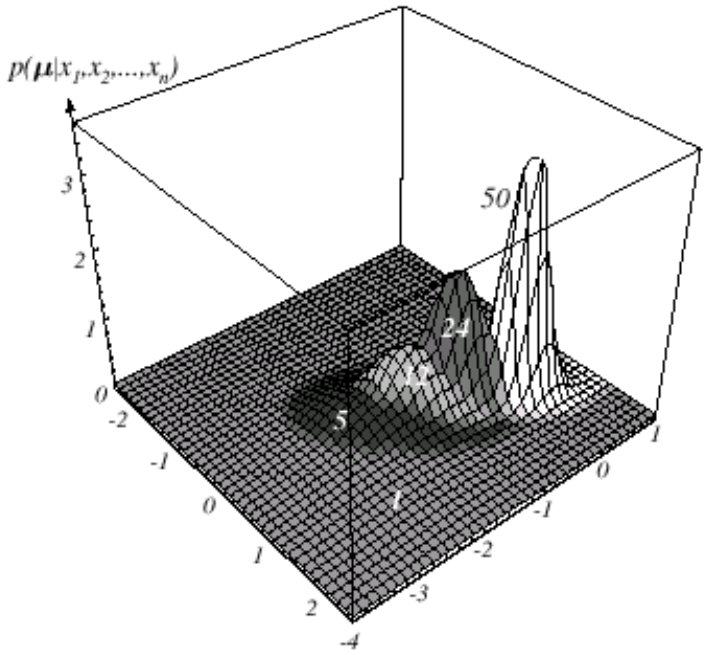


gaussInferParamsMean1d from Murphy, Page 121

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

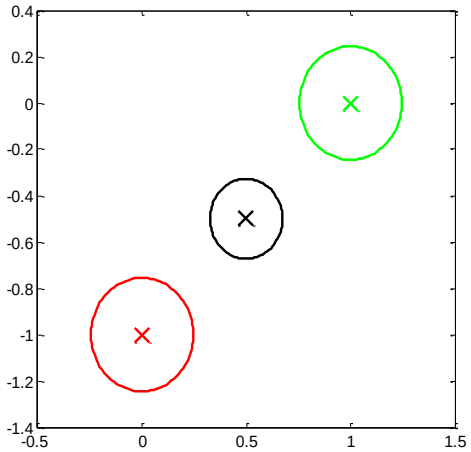
Illustration of Bayesian Learning - 2



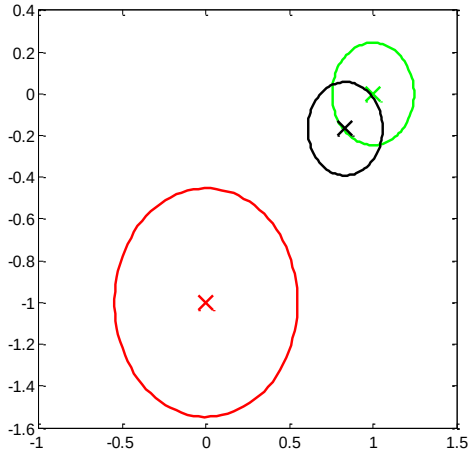


Bayesian learning of the mean of Gaussian distributions in one and two dimensions. As the number of samples increase, the posterior density peaks at the true value.

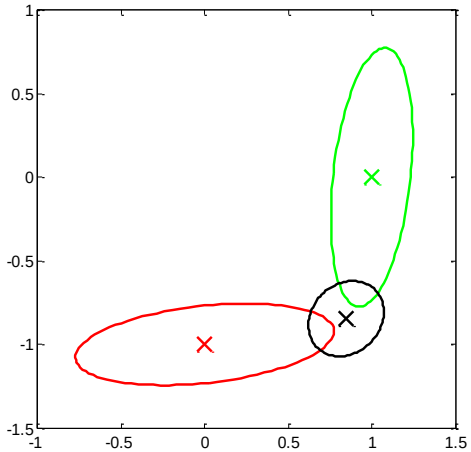
Sensor Fusion



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

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

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

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

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

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

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

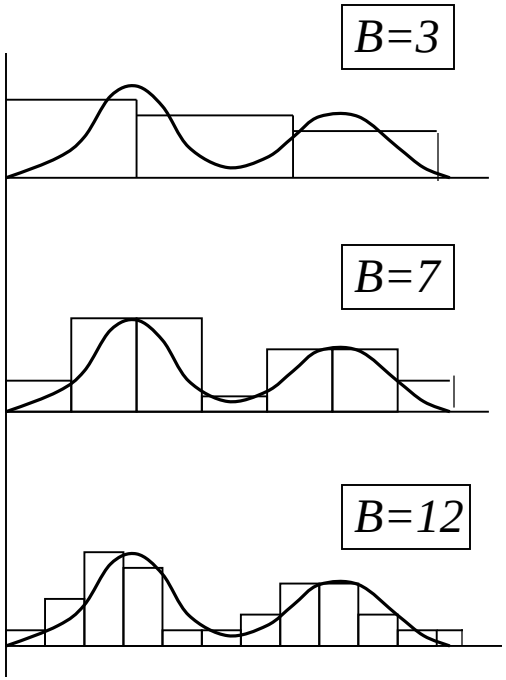
- Assume x was already standardized, then to construct a histogram, divide the range into a set of B evenly spread bins

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 $\downarrow 0$

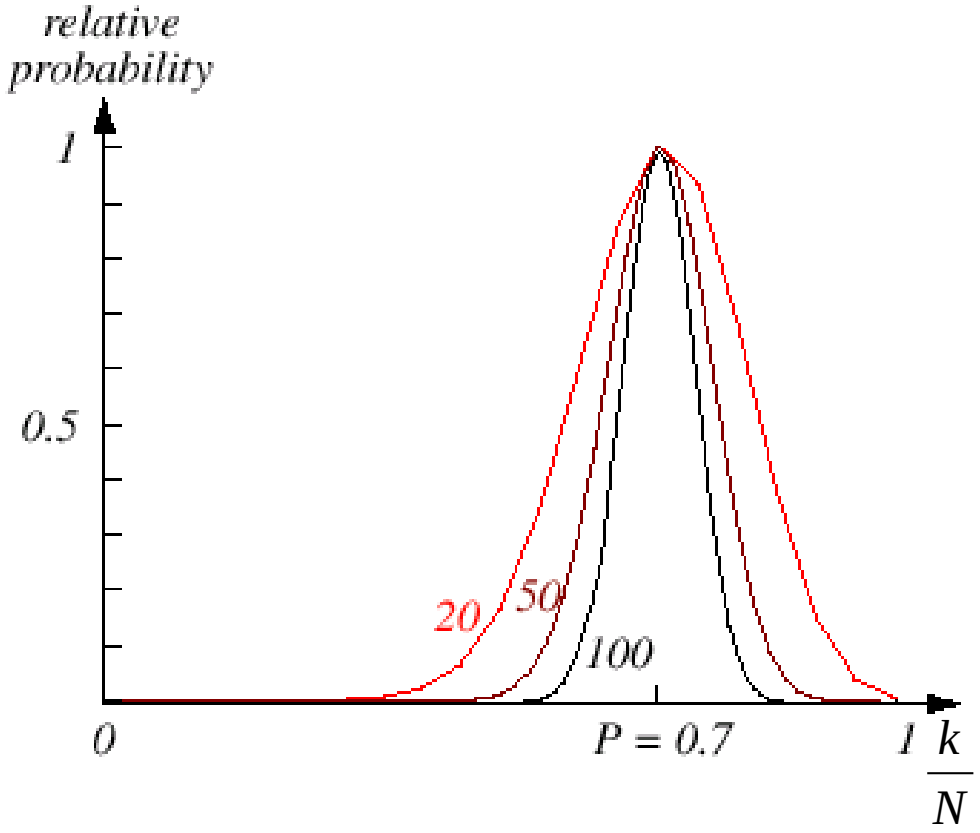
$$\Rightarrow P \approx \frac{k}{N} \text{ is a good estimate} \qquad \ldots\ldots(1)$$

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

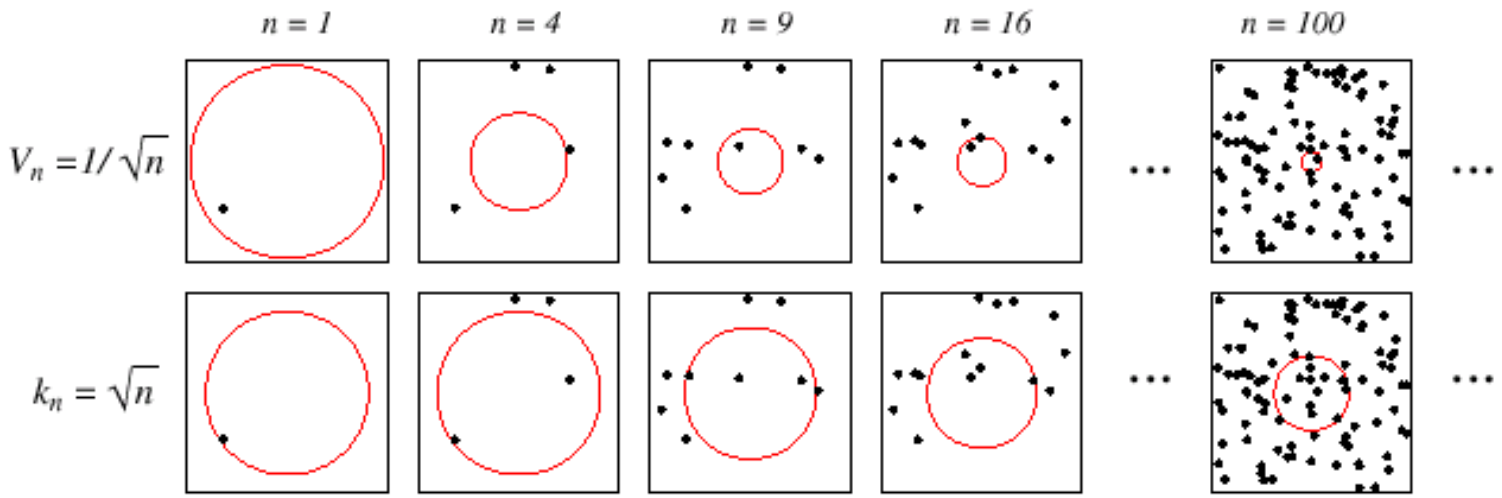


FIGURE 4.2. There are two leading methods for estimating the density at a point, here at the center of each square. The one shown in the top row is to start with a large volume centered on the test point and shrink it according to a function such as $V_n = 1/\sqrt{n}$. The other method, shown in the 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**

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

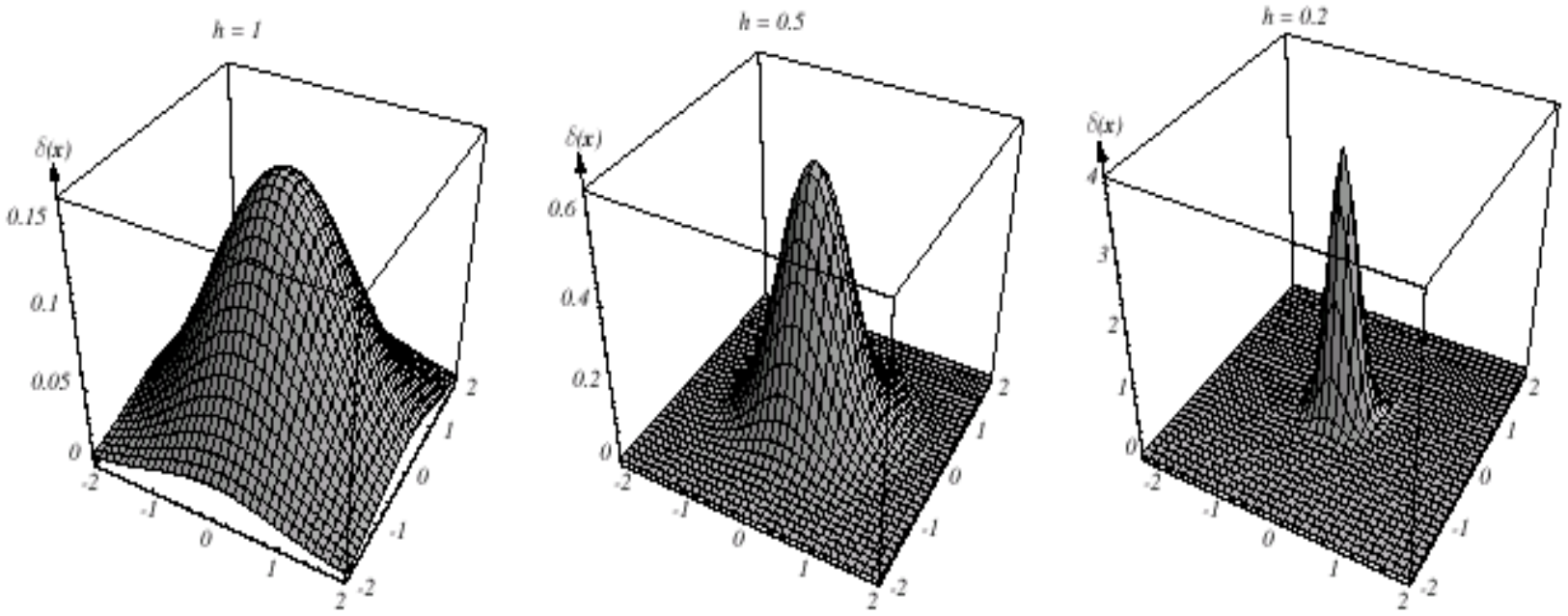
$H(\underline{u})$ is a unit hypercube centered at the origin. Gaussian windows could be used as well.

$\Rightarrow$ 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*

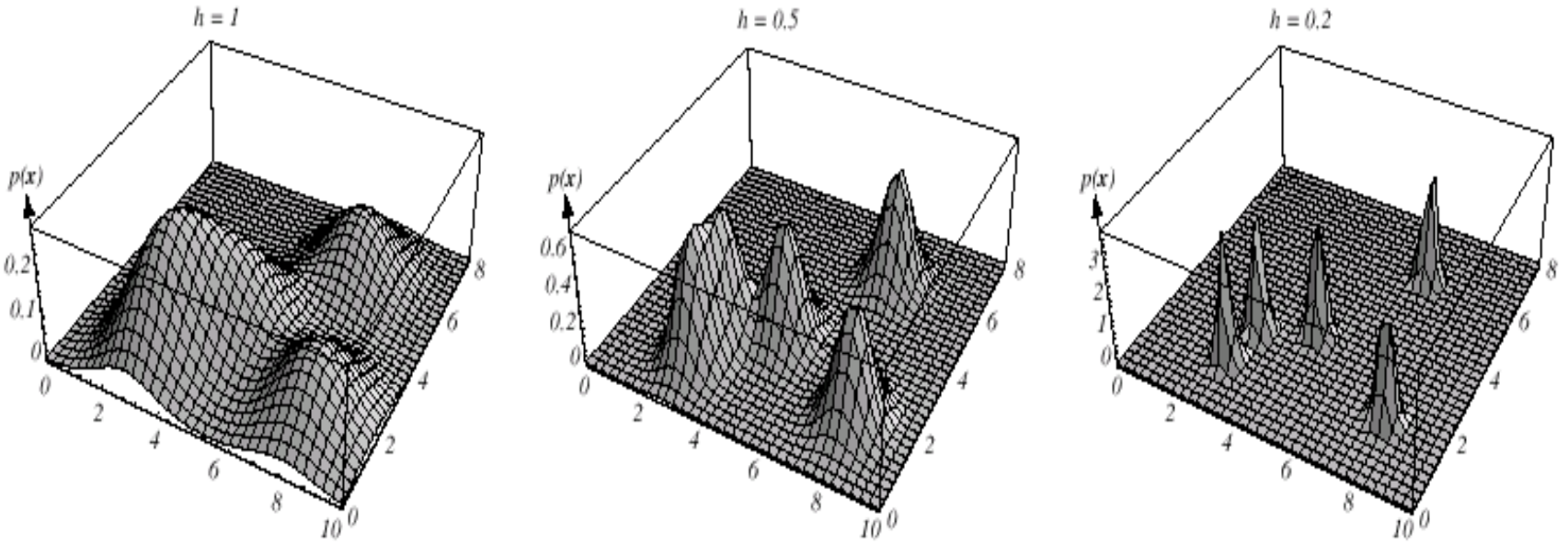
$$E[\hat{p}(\underline{x})] = E\left[\frac{1}{N}\frac{1}{h^p}\sum_{j=1}^{N} H\left(\frac{\underline{x}-\underline{x}^j}{h}\right)\right] = E\left[\frac{1}{h^p} H\left(\frac{\underline{x}-\underline{v}}{h}\right)\right]$$

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

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

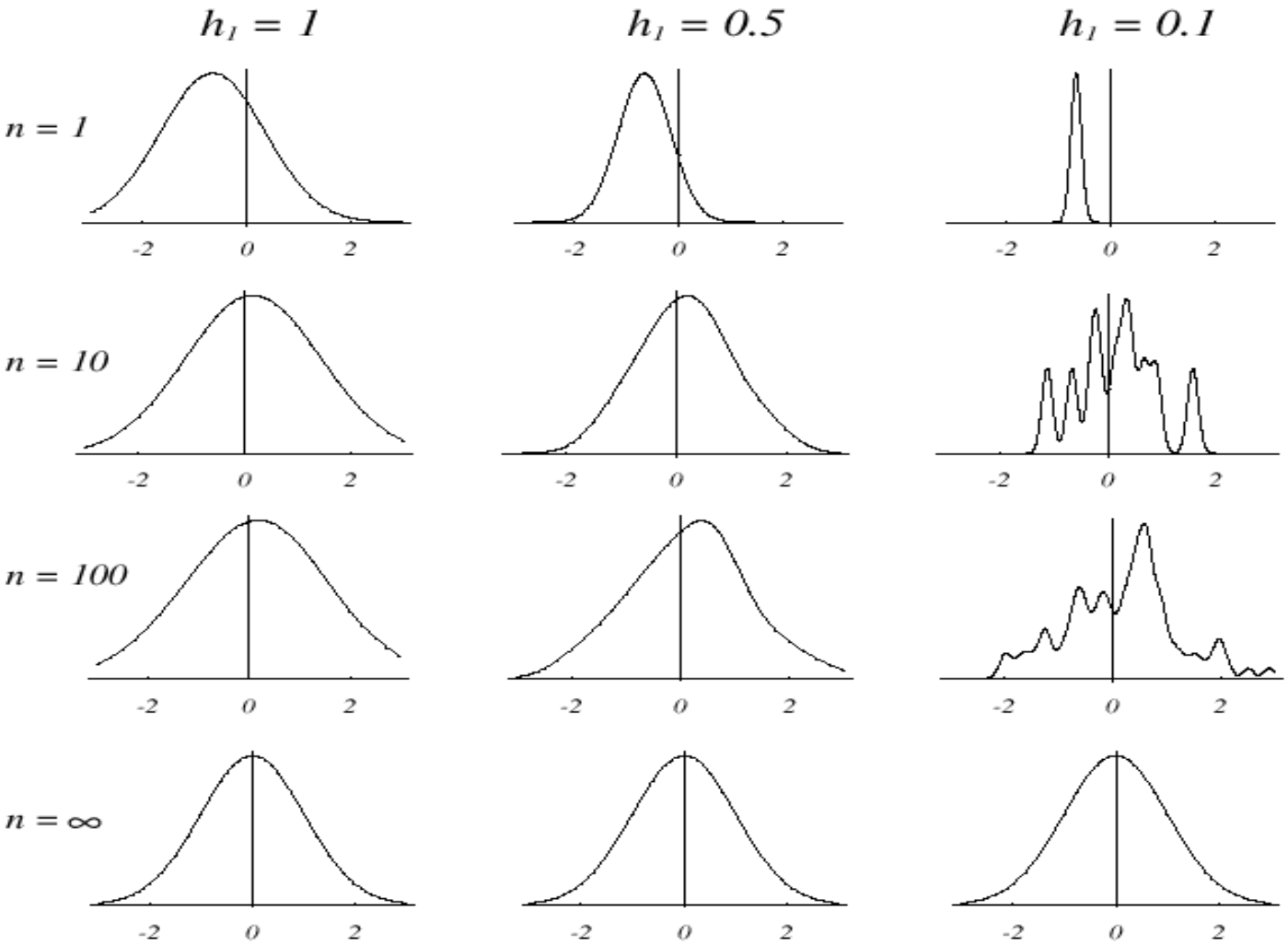
Large $N \Rightarrow$ Good estimate for $h \downarrow 0$
Small $N \Rightarrow$ Need to select h properly
For small N, h small $\Rightarrow$ noisy estimate

Illustration of Density Estimation: Effect of *h*



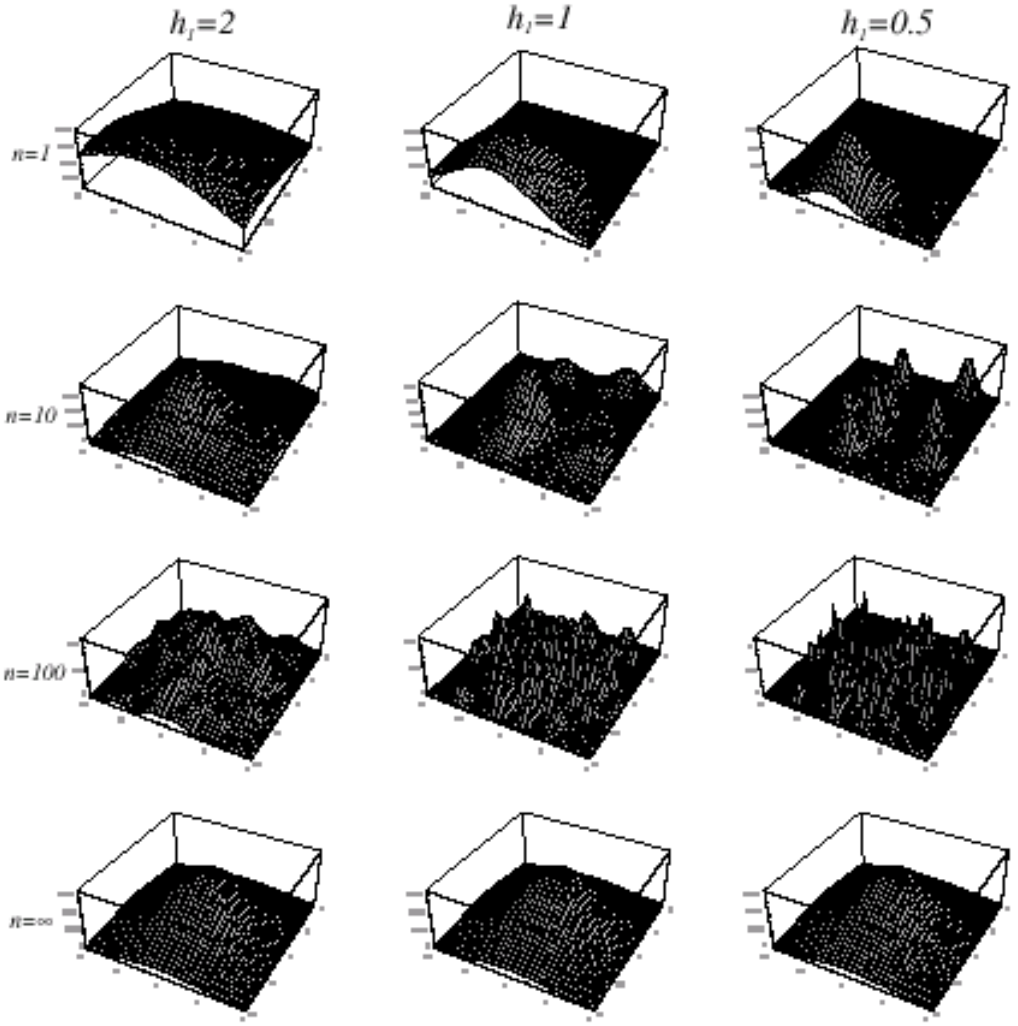
Three Parzen-window density estimates based on the same set of five samples, using the Gaussian Parzen window functions

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



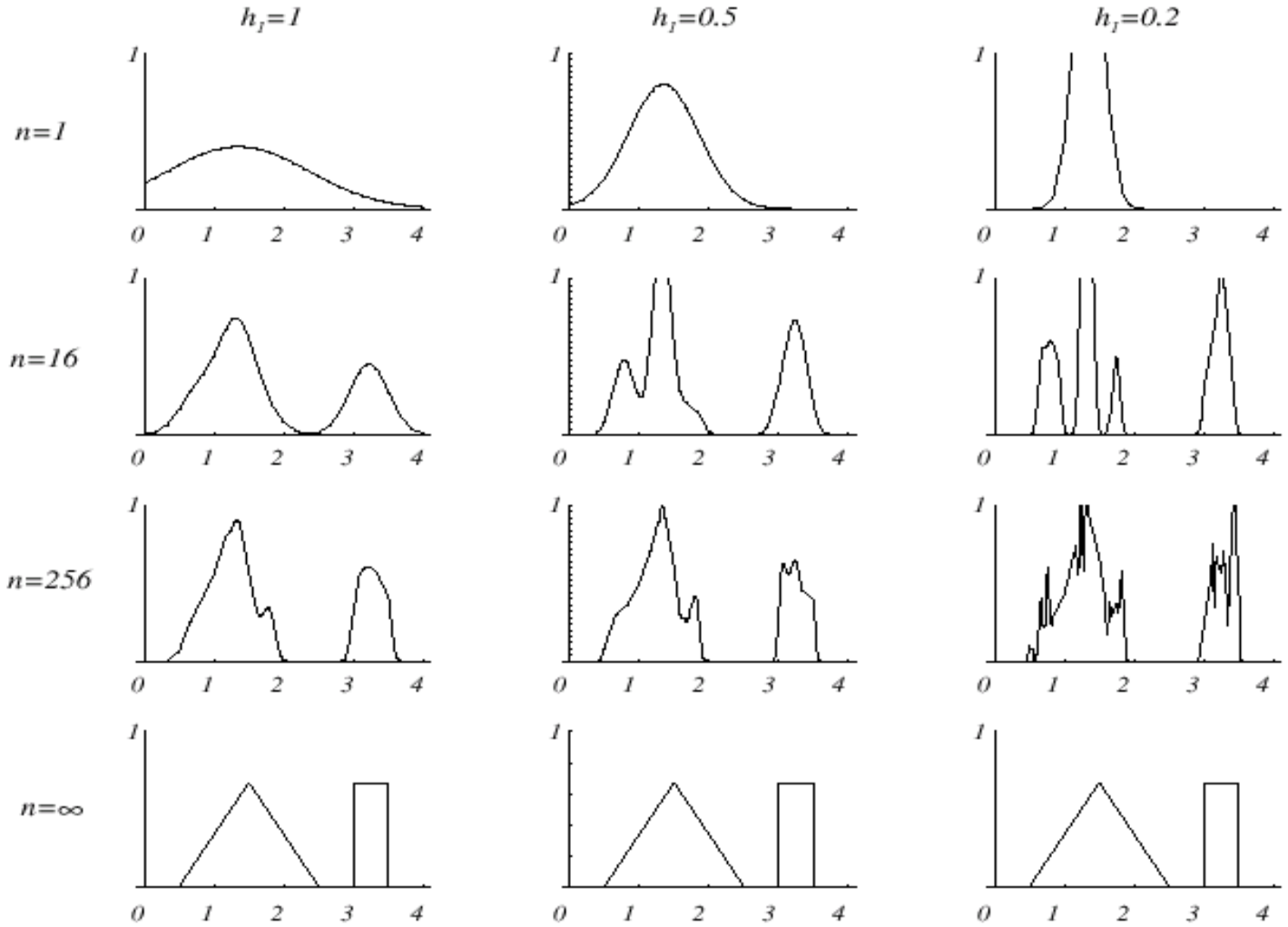
Parzen-window estimates of a univariate normal density using different window widths and numbers of samples

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)

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-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**

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

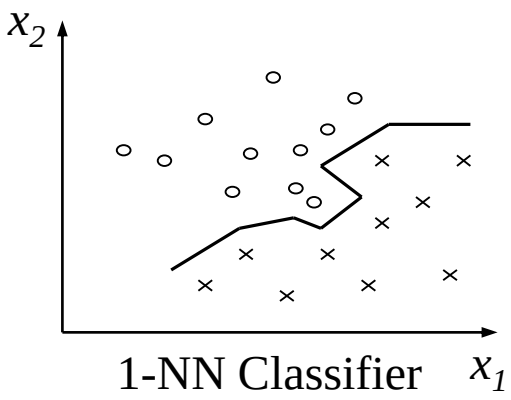
$$p(\underline{x}) = \frac{k}{NV}$$

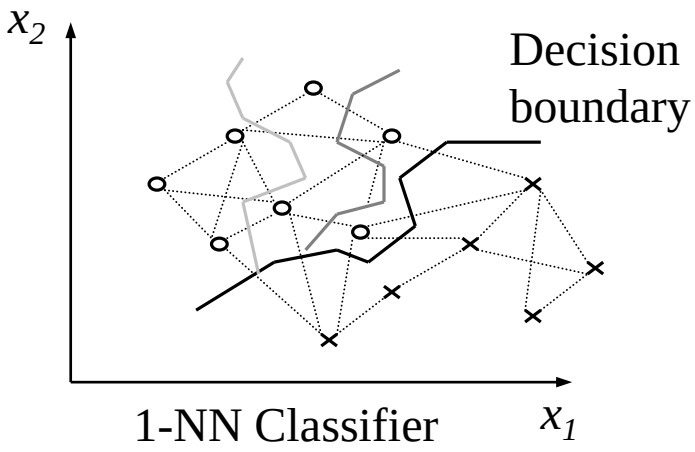
$$P(z = i \mid \underline{x}) = \frac{p(\underline{x} \mid z = i) P(z = i)}{p(\underline{x})} = \frac{k_i}{n_i V} \frac{n_i}{N} \frac{1}{\frac{k}{NV}} = \frac{k_i}{k}$$

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.

Why not Estimate Posterior Probability Directly?

- Minimum distance versus Maximum a Posteriori decisions

Select class i if $\|\underline{P} - \underline{z}_k\|^2$ is a min

$$d_k = (\underline{P} - \underline{z}_k)^T (\underline{P} - \underline{z}_k) = \|\underline{P}\|^2 - 2\underline{P}^T \underline{z}_k + \underline{z}_k{}^T \underline{z}_k = 1 - 2P(z = k \mid \underline{x}) + \|\underline{P}\|^2$$

$$Alternate: d_k = (\underline{y}_k - \underline{z}_k)^T (\underline{y}_k - \underline{z}_k) = \frac{C}{C-1}[1 - P(z = k \mid \underline{x}) + C\|\underline{P}\|^2]$$

$$\min_k d_k \Rightarrow \max_k P(z = k \mid \underline{x})$$

- Learning posterior probabilities directly … **Discriminative Learning**

$P(z = k \mid \underline{x}) \sim$ **posterior probability of class** k **given** $\underline{x}$

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

MAP Decisions: Gaussian Case

- *Gaussian case*

$$P(z=k \mid \underline{x}) = \frac{\frac{1}{|\Sigma_k|^{1/2}} e^{-\frac{1}{2}(\underline{x}-\underline{\mu}_k)^T \Sigma_k^{-1}(\underline{x}-\underline{\mu}_k)} P(z=k)}{\sum_{i=1}^{C} \frac{1}{|\Sigma_i|^{1/2}} e^{-\frac{1}{2}(\underline{x}-\underline{\mu}_i)^T \Sigma_i^{-1}(\underline{x}-\underline{\mu}_i)} P(z=i)}$$

$$= \frac{e^{[-\frac{1}{2}(\underline{x}-\underline{\mu}_k)^T \Sigma_k^{-1}(\underline{x}-\underline{\mu}_k) - \frac{1}{2}\ln|\Sigma_k| + \ln P(z=k)]}}{\sum_{i=1}^{C} e^{[-\frac{1}{2}(\underline{x}-\underline{\mu}_i)^T \Sigma_i^{-1}(\underline{x}-\underline{\mu}_i) - \frac{1}{2}\ln|\Sigma_i| + \ln P(z=i)]}}$$

$$= \frac{e^{y(\underline{x},\underline{\theta}_k)}}{\sum_{i=1}^{C} e^{y(\underline{x},\underline{\theta}_i)}}$$

Why learn $\underline{\theta}$? $y(\underline{x}) = \underline{x}^T A_i \underline{x} + \underline{b}_i^T \underline{x} + c_i$

Why not learn weights directly?

$$\underline{\theta} = \Big[\underbrace{\{\underline{\mu}_k\}, \{\Sigma_k\}, P\{z=k\}}_{\underline{\theta}_k}\Big]$$

"soft max"
"multiple logistic model"
"logistic discrimination"

$y(\underline{x}, \underline{\theta}_k) = g_k(\underline{x})$
discriminant function

MAP Decisions : Gaussian with Equal Covariances

- *Equal covariance case*

$$P(z=k \mid \underline{x}) = \frac{e^{\underline{\mu}_k{}^T \Sigma^{-1} x - \frac{1}{2}\underline{\mu}_k{}^T \Sigma^{-1} \underline{\mu}_k + \ln P(z=k)}}{\sum_{i=1}^{C} e^{\underline{\mu}_i{}^T \Sigma^{-1} x - \frac{1}{2}\underline{\mu}_i{}^T \Sigma^{-1} \underline{\mu}_i + \ln P(z=i)}}$$

$$= \frac{e^{y(\underline{x}, \underline{w}_k)}}{\sum_{i=1}^{C} e^{y(\underline{x}, \underline{w}_i)}}$$

where $y(\underline{x}, \underline{w}_k) = \underline{w}_k{}^T \underline{x} - w_{k0}$

$$\underline{w}_k = \Sigma^{-1} \underline{\mu}_k \qquad w_{k0} = \frac{1}{2} \underline{\mu}_k{}^T \Sigma^{-1} \underline{\mu}_k - \ln P(z=k)$$

MAP Decisions: Gaussian, Equal Σ, Two Class Case

- *Two class case*

  Let the classes be $z \in \{0,1\}$

Note : If $z \in \{-1,1\}$,

$$P(z \mid \underline{x}, \underline{w}) = \frac{1}{1+e^{-zy}} = g(y,z)$$

$$\mu = P(z=1 \mid \underline{x}, \underline{w}_1, \underline{w}_0) = \frac{e^{y(\underline{x},\underline{w}_1)}}{e^{y(\underline{x},\underline{w}_1)} + e^{y(\underline{x},\underline{w}_0)}} = \frac{1}{1+e^{-(y(\underline{x},\underline{w}_1)-y(\underline{x},\underline{w}_0))}}$$

$$= \frac{1}{1+e^{-y}} = g(y) = \mu \qquad \frac{dg}{dy} = g(1-g)$$

where $y = y(\underline{x},\underline{w}_1) - y(\underline{x},\underline{w}_0) = \underline{w}_1^{\,T}\underline{x} - w_{10} - \underline{w}_0^{\,T}\underline{x} + w_{00}$

$$\frac{\mu}{1-\mu} = e^{y}$$

$$= (\underline{w}_1 - \underline{w}_0)^T \underline{x} - (w_{10} - w_{00}) = \tilde{\underline{w}}^T \underline{x} - \tilde{w}_0 = \underline{w}^T \underline{x}_a \qquad \underline{x}_a = \begin{bmatrix} -1 \\ \underline{x} \end{bmatrix}$$

$$P(z \mid \underline{x}, \underline{w}) = \mu^z (1-\mu)^{1-z} = \left(\frac{1}{1+e^{-y}}\right)^z \left(\frac{1}{1+e^{y}}\right)^{1-z} = \frac{1}{1+e^{-(2z-1)y}} = e^{yz} g(-y)$$

One of the classes can be taken as a reference and set its weights to 0!

The parameters $\tilde{\underline{w}}, \tilde{w}_0$ are directly estimated from the data. Assume $\underline{x}$ is augmented.

Maximum Likelihood Estimation of Weights

- Suppose have data

$$D = \left\{ \{\underline{x}^1, z^1\}\ \{\underline{x}^2, z^2\}, \cdots\cdots, \{\underline{x}^N, z^N\} :\ z^n \in \{0,1\} \right\}$$

- Likelihood

$$L(\underline{w}) = P(\{z^n\}_{n=1}^N \mid \{\underline{x}^n\}_{n=1}^N, \underline{w}) = \prod_{n=1}^N P(z^n \mid \underline{x}^n, \underline{w}) = \prod_{n=1}^N (\mu_n)^{z^n} (1-\mu_n)^{1-z^n}$$

Recall convexity of Cross Entropy, $-J_n$

$$J(\underline{w}) = -\ln L(\underline{w}) = NLL(\underline{w}) = -\sum_{n=1}^N J_n ; J_n = [z^n \ln \mu_n + (1-z^n)\ln(1-\mu_n)] \ldots . - ve\ Cross\ Entropy$$

$$\nabla_{\underline{w}} J = -\sum_{n=1}^N \underbrace{\frac{\partial J_n}{\partial \mu_n}}_{\frac{z^n-\mu_n}{\mu_n(1-\mu_n)}} \underbrace{\frac{\partial \mu_n}{\partial y_n}}_{\mu_n(1-\mu_n)} \underbrace{\nabla_{\underline{w}} y_n}_{\underline{x}^n} = \sum_{n=1}^N \left(\mu_n - z^n\right) \underline{x}^n = \underbrace{X^T}_{(p+1)\text{x}N} \underbrace{\left(\underline{\mu} - \underline{z}\right)}_{N\text{x}1}$$

$$X = \begin{bmatrix} (\underline{x}^1)^T \\ (\underline{x}^2)^T \\ . \\ (\underline{x}^N)^T \end{bmatrix}; N\text{x}(p+1)$$

$\nabla^2_{\underline{w}} J = X^T \Sigma X > 0$ *where* $\Sigma = Diag[\mu_n(1-\mu_n)]$

Re*gularize by adding* $\lambda \underline{w}^T \underline{w}$ *or* $\lambda \| \underline{w} \|_1$ *to* $J(\underline{w})$

- Optimization Methods: SD, Newton, Levenberg-Marquardt, CG, QN,...

Iteratively Reweighted Least Squares (IRLS)

- Newton's Method

$$\underline{w}_{i+1} = \underline{w}_i - \left(\nabla^2_{\underline{w}_i} J\right)^{-1} \nabla_{\underline{w}_i} J$$

$$= \underline{w}_i + \left(X^T \Sigma_i X\right)^{-1} X^T (\underline{z} - \underline{\mu}_i)$$

$$= \left(X^T \Sigma_i X\right)^{-1} \left[X^T \Sigma_i X \underline{w}_i + X^T (\underline{z} - \underline{\mu}_i)\right]$$

$$= \left(X^T \Sigma_i X\right)^{-1} X^T \Sigma_i \underbrace{\left[X \underline{w}_i + \Sigma_i^{-1} (\underline{z} - \underline{\mu}_i)\right]}_{\underline{y}_i}$$

$$y_{in} = \underline{w}_i^T \underline{x}^n + \frac{z^n - \mu_{in}}{\mu_{in}(1-\mu_{in})}$$

- This corresponds to the following **weighted least squares** problem at iteration *i*

$$\min J_i = \frac{1}{2} \left(\underline{y}_i - X \underline{w}\right)^T \Sigma_i \left(\underline{y}_i - X \underline{w}\right)$$

- Weighted least squares can be implemented in recursive way…. One sample at a time.

Multi-class Case

- Convert the problem to one versus rest problem (one-hot coding)
- Multiclass problems
  - *M* class problem: *M* two-class tasks or *M(M-1)/2* two-class tasks
  - Obtain a separate classifier for each pair
  - What if more than one decision rule classifies? **Majority rule**
- Error Correcting Output Codes
  - Each class: an *n* bit pattern chosen so that Hamming distance is as large as possible among classes
  - Learn each of the *n* bits via binary classifiers
  - Select nearest class (i.e., one with the least Hamming distance)
- Work Directly with Multi-class formulation
  - Direct extension of the binary case
  - Upper and lower bounds on sigmoid function
- Laplace approximation of posterior to quantify uncertainty in predictions

Learning Weights

- What to do in multiple class case ?

$$Let\ z_k^n = \begin{cases} 1\ if\ \ z^n = k \\ 0;\ otherwise \end{cases} \quad ....\ one\ of\ C\ coding$$

$$P(z_k^n = 1 \mid \underline{x}^n, \{\underline{w}_i\}_{i=1}^C) = \frac{e^{y(\underline{x}^n, \underline{w}_k)}}{\sum_{i=1}^{C} e^{y(\underline{x}^n, \underline{w}_i)}} = \frac{e^{\underline{w}_k^T \underline{x}^n}}{\sum_{i=1}^{C} e^{\underline{w}_i^T \underline{x}^n}} = \mu_{nk} \ ...\ soft\ \max\ function$$

$$Use\ class\ C\ as\ a\ reference.\ So,\ set\ \underline{w}_C = \underline{0} \Rightarrow W^T = \begin{bmatrix} \underline{w}_1^T \\ \underline{w}_2^T \\ . \\ \underline{w}_{C-1}^T \end{bmatrix}; (C-1)\mathrm{x}(p+1)$$

$$L = p(\{z^n\}_{n=1}^N \mid \{\underline{x}^n\}_{n=1}^N, W) = \prod_{n=1}^{N} \left(\prod_{k=1}^{C-1} (\mu_{nk})^{z_k^n} \right) \left(1 - \sum_{i=1}^{C-1} \mu_{ni} \right)^{\left(1 - \sum_{i=1}^{C-1} z_i^n\right)}$$

Gradient and Hessian in Multi-class Case

$$J = -\ln L = \sum_{n=1}^{N}\left[\left\{-\sum_{k=1}^{C-1} z_k^n \ln\left(\frac{\mu_{nk}}{1-\sum_{i=1}^{C-1}\mu_{ni}}\right)\right\} - \ln\left(1-\sum_{i=1}^{C-1}\mu_{ni}\right)\right]$$

$$Note: \frac{\mu_{nk}}{1-\sum_{i=1}^{C-1}\mu_{ni}} = e^{\underline{w}_k^T \underline{x}^n} \text{ and } \left(1-\sum_{i=1}^{C-1}\mu_{ni}\right) = \left(1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n}\right)^{-1}; \text{Recall } \underline{w}_C = \underline{0}$$

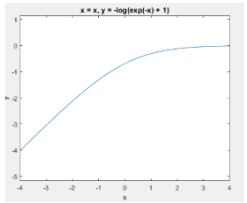
$$\Rightarrow J = \sum_{n=1}^{N}\left[\left(-\sum_{k=1}^{C-1} z_k^n \underline{w}_k^T \underline{x}^n\right) + \ln\left(1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n}\right)\right]$$

$$\nabla_{\underline{w}_k} J = \sum_{n=1}^{N}\left(\mu_{nk} - z_k^n\right)\underline{x}^n \because \nabla_{\underline{w}_k} \ln\left(1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n}\right) = \left(\frac{e^{\underline{w}_k^T \underline{x}^n}}{1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n}}\right)\underline{x}^n = \mu_{nk}\underline{x}^n$$

$$\nabla^2_{\underline{w}_k \underline{w}_k} J = \sum_{n=1}^{N} \mu_{nk}\left(1-\mu_{nk}\right)\underline{x}^n\left(\underline{x}^n\right)^T; \nabla^2_{\underline{w}_k \underline{w}_j} J = -\sum_{n=1}^{N} \mu_{nk}\mu_{nj}\underline{x}^n\left(\underline{x}^n\right)^T; k \neq j$$

Big Matrix

Variational Upper Bound on Sigmoid



- Upper bounds on sigmoid function (helps when C is large)

Consider the problem of maximizing $\ln g(x) = -\ln(1+e^{-x})$ ... *concave in x*

Let $y = e^{-x} \Rightarrow \ln y = -x$

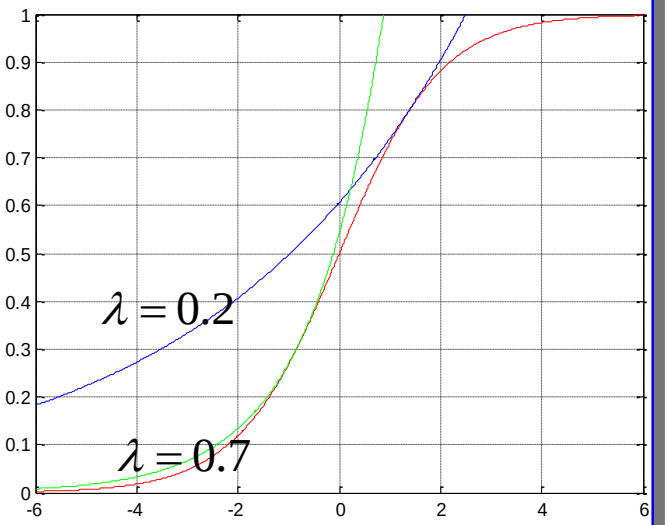
Consider $\max\{-\ln(1+y)\}$ *subject to* $\ln y + x = 0$

$$L(x, y, \lambda) = -\ln(1+y) + \lambda(\ln y + x) \Rightarrow y^* = \frac{\lambda}{1-\lambda}$$

$$L(x, y^*, \lambda) = \underbrace{(1-\lambda)\ln(1-\lambda) + \lambda \ln \lambda}_{-H(\lambda)} + \lambda x$$

$$\Rightarrow \ln g(x) \le \min_{\lambda} \left[\underbrace{(1-\lambda)\ln(1-\lambda) + \lambda \ln \lambda}_{-H(\lambda)} + \lambda x \right] \forall x$$

So, $g(x) \le \exp(\lambda x - H(\lambda)) \; \forall x, \lambda$



This can be used to obtain an upper bound on LL and use primal-dual methods for optimization. We will illustrate its application in the multi-class case.

Log-Sum-Exponent bound & Multi-class Case

- Recall

$$l = \ln p(\{z^n\}_{n=1}^N | \{\underline{x}^n\}_{n=1}^N, W) = \sum_{n=1}^{N}\left[\left(\sum_{k=1}^{C-1} z_k^n \underline{w}_k^T \underline{x}^n\right) - \ln\left(1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n}\right)\right]$$

& $-\ln(1+e^x)$ *is concave in* x

Consider $\max -\{\ln(1+\sum_{i=1}^{C-1} e^{y_i})\} \Rightarrow$ *Lagrangian* $L(\{y_i\},\lambda,v) = -\ln(1+v) + \lambda(v - \sum_{i=1}^{C-1} e^{y_i})$

$$\Rightarrow v = (1-\lambda)/\lambda \Rightarrow -\ln(1+\sum_{i=1}^{C-1} e^{y_i}) \le \ln\lambda + 1 - \lambda(1+\sum_{i=1}^{C-1} e^{y_i})$$

$$so, -\ln\left(1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n}\right) \le \ln\lambda_n + 1 - \lambda_n(1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n})$$

$$So, l \le \sum_{n=1}^{N}\left[\left(\sum_{k=1}^{C-1} z_k^n \underline{w}_k^T \underline{x}^n\right) + \ln\lambda_n + 1 - \lambda_n(1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n})\right] \cdots \text{maximize UB wrt } \{\underline{w}_k\} \text{ and minimize wrt } \lambda$$

The UB is a separable problem for each class given $\{\lambda_n\}$!

$$\tilde{l}_k = \sum_{n=1}^{N}\left[z_k^n \underline{w}_k^T \underline{x}^n - \lambda_n e^{\underline{w}_k^T \underline{x}^n})\right] \text{.....Needs NLP optimizer}$$

$$Next\ \lambda_n : \lambda_n = 1/\left(1+\sum_{i=1}^{C-1} e^{\underline{w}_i^T \underline{x}^n}\right)$$

} Iterate

Laplace (Gaussian) Approximation of Posterior

Represent W by a vector $\underline{w} = \left[\underline{w}_1^T \; \underline{w}_2^T ....., \underline{w}_{C-1}^T \right]^T$, a $(p+1)$x$(C-1)$ vector

Let $\underline{w}^*$ be the optimal solution minimizing the NLL function, J

Let $\nabla^2_{\underline{w}\underline{w}} J = H$ be the Hessian evaluated at $\underline{w}^*$

- Laplace approximation of posterior weight distribution

  $p(\underline{w} \mid D) \sim N(\underline{w}; \underline{w}^*, H^{-1}); H =$ Hessian is the Information matrix

- Posterior predictive distribution of z for a given $\underline{x}$ ($\Rightarrow$testing)

$$P(z \mid \underline{x}, D) = \int_{\underline{w}} P(z \mid \underline{x}, \underline{w}) \, p(\underline{w} \mid D) d\underline{w}$$

(i) Plug-in MAP estimate $\Rightarrow p(\underline{w} \mid D) = \delta(\underline{w} - \underline{w}^*)$: $P(z \mid \underline{x}, D) \approx P(z \mid \underline{x}, \underline{w}^*)$

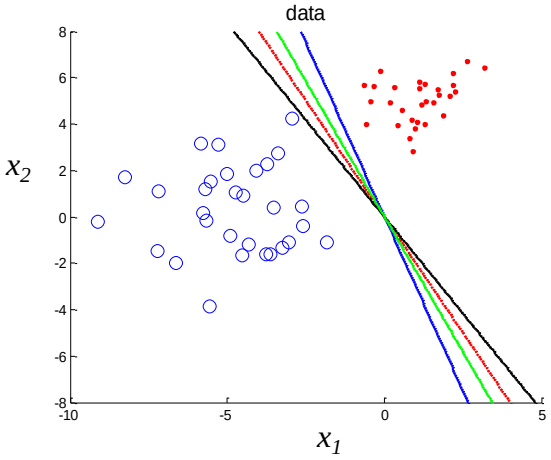
(ii) Monte Carlo approximation: $P(z \mid \underline{x}, D) \approx \frac{1}{M} \sum_{m=1}^{M} P(z \mid \underline{x}, \underline{w}^m)$

$\underline{w}^m$ is drawn from the posterior distribution, $p(\underline{w} \mid D)$

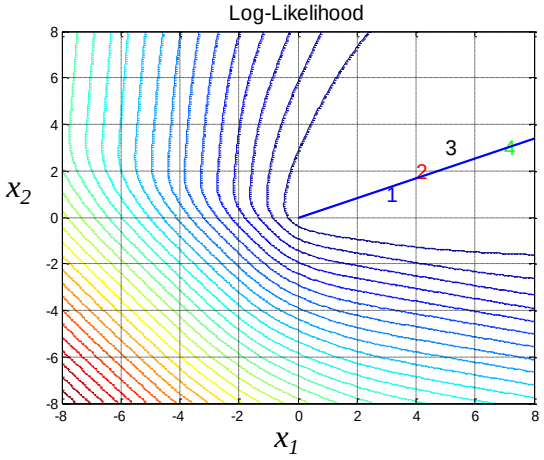
(iii) Probit approximation: $P(z \mid \underline{x}, D) \approx \int_{\underline{w}} P(z \mid \underline{x}, \underline{w}) N(\underline{w}; \underline{w}^*, H^{-1}) d\underline{w}$

Illustration of Logistic Regression in 2D

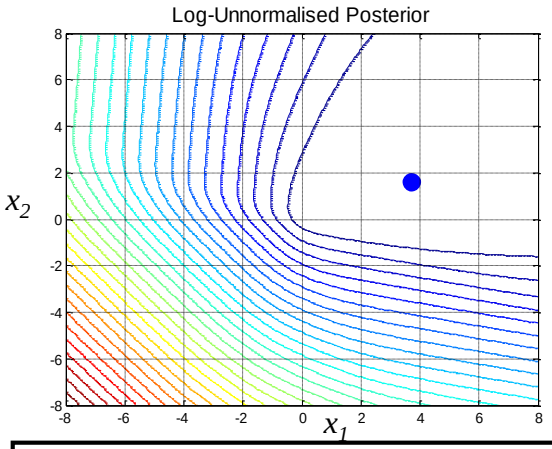
Linearly Separable Data



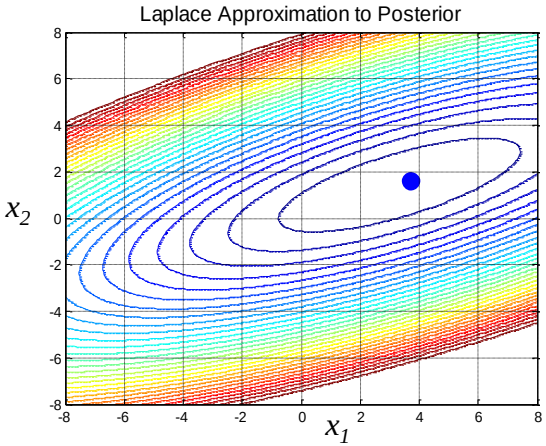
MLE



MAP for Prior $N(\underline{w},\underline{0}, 100I)$



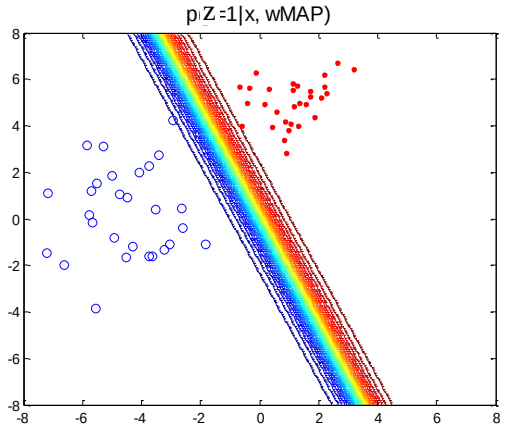
Laplace Approximation of Posterior



Generated using *logredLaplaceGirolamiDemo* from Murphy, Page 257

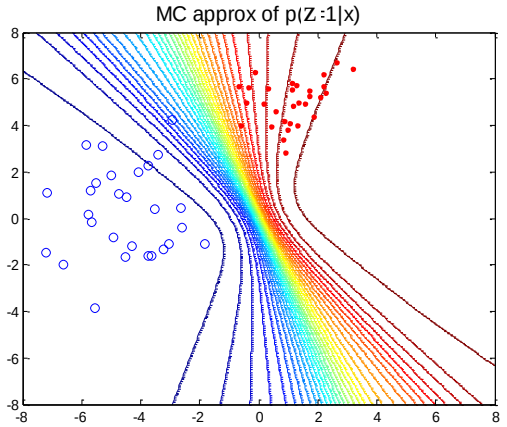
Contours of Posterior Predictive Distribution

Plug-in Approximation

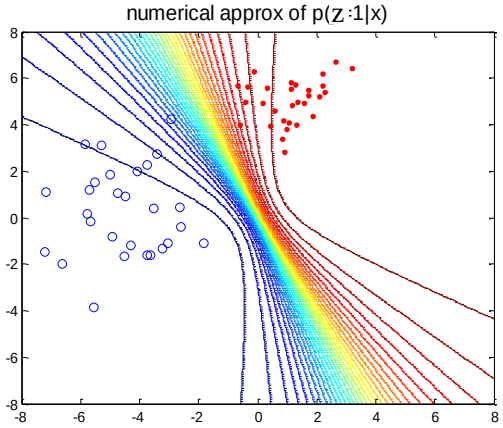


Contours of $P(z=1|\underline{x}, D)$

MC Approximation



Laplace + Probit Approximation of Posterior



- Plug-in underestimates uncertainty
- MC: Posterior mean decision boundary is equally far from both classes (similar to large margin classifiers, e.g., SVM)
- Laplace + probit is similar to MC

Probit Approximation of Sigmoid - 1

- What is probit? CDF of normal distribution

$$\Phi(a) \triangleq \int_{-\infty}^{a} N(x;0,1)dx \quad \textit{cumulative distribution function (CDF)}; \Phi(-\infty)=0; \Phi(\infty)=1$$

- Sigmoid function $g(a)$ approximates probit $\Phi(\lambda a)$ well if slopes are matched at the origin. Note $g(-\infty)=0$; $g(\infty)=1$

$$g'(0) = g(0)[1-g(0)] = \frac{1}{4}; \Phi'(0) = \frac{\lambda}{\sqrt{2\pi}} \Rightarrow \lambda = \sqrt{\frac{\pi}{8}}$$

$$\begin{aligned}
&\textit{Define } x = z - \lambda a; z \sim N(z;0,1) \\
&\Rightarrow p(x) = N(x; -\lambda\mu_a, 1+\lambda^2\sigma_a^2) \\
&p(x \mid a) = N(x; -\lambda a, 1) \\
&P(x \le 0) = \int_a P(x \le 0 \mid a) N(a; \mu_a, \sigma_a^2) da \\
&= \int_a \Phi(\lambda a) N(a; \mu_a, \sigma_a^2) da = \Phi\left(\frac{\mu_a}{\left(\sigma_a^2 + 8/\pi\right)^{1/2}} \right)
\end{aligned}$$

- So,

$$P(z=1 \mid \underline{x}, D) \approx \int_{\underline{w}} P(z=1 \mid \underline{x}, \underline{w}) N(\underline{w}; \underline{w}^*, H^{-1}) d\underline{w}$$

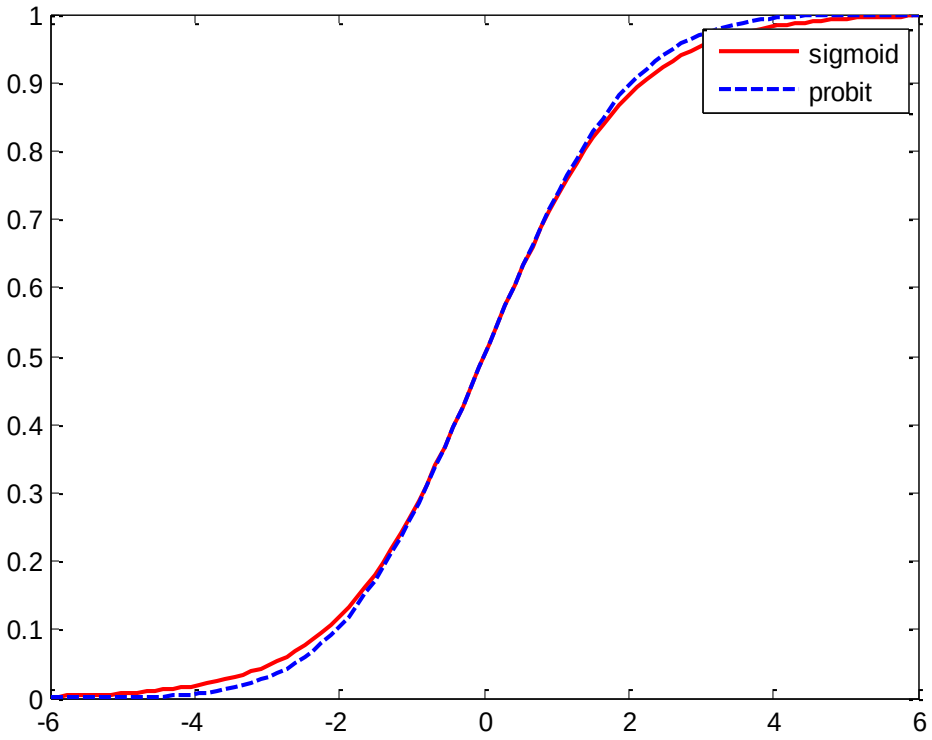
$$= \int_{\underline{w}} \left(\frac{1}{1+e^{-\underline{w}^T \underline{x}}} \right) N(\underline{w}; \underline{w}^*, H^{-1}) d\underline{w}$$

$$\textit{Let } a = \underline{w}^T \underline{x} = \underline{x}^T \underline{w} \Rightarrow a = N(a; \underbrace{\underline{x}^T \underline{w}^*}_{\mu_a}, \underbrace{\underline{x}^T H^{-1} \underline{x}}_{\sigma_a^2}). \textit{ So, } P(z=1 \mid \underline{x}, D) \approx \int_a \underbrace{\left(\frac{1}{1+e^{-a}} \right)}_{g(a)} N(a; \mu_a, \sigma_a^2) da$$

$$\approx \int_a \Phi(\sqrt{\pi/8}\, a) N(a; \mu_a, \sigma_a^2) da = \Phi\left(\frac{\mu_a}{\left(\sigma_a^2 + 8/\pi\right)^{1/2}} \right) \approx g\left(\frac{\mu_a}{\left(1+\pi\sigma_a^2/8\right)^{1/2}} \right)$$

$$\int_{-\infty}^{\infty} \Phi(a+bx)\phi(x)\, dx = \Phi\left(\frac{a}{\sqrt{1+b^2}} \right)$$

Probit Approximation of Sigmoid - 2



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

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

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

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

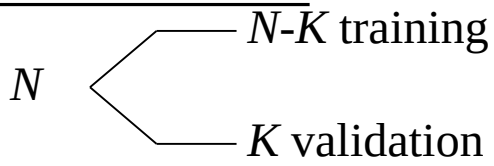
$$So, AIC(H_1) < AIC(H_0)\ if\ \left(\bar{x}\right)^2 > \frac{2}{N} \Rightarrow |\bar{x}| > \sqrt{\frac{2}{N}}$$

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

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

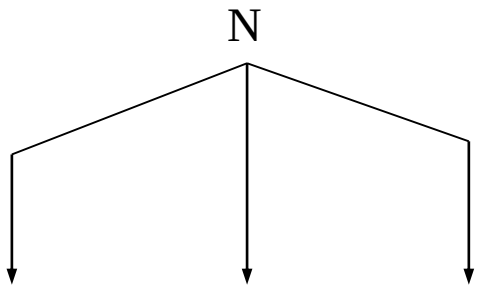
"*Confidence* (α) *is cheap, but accuracy* (ε) *is more* exp *ensive*"

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

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

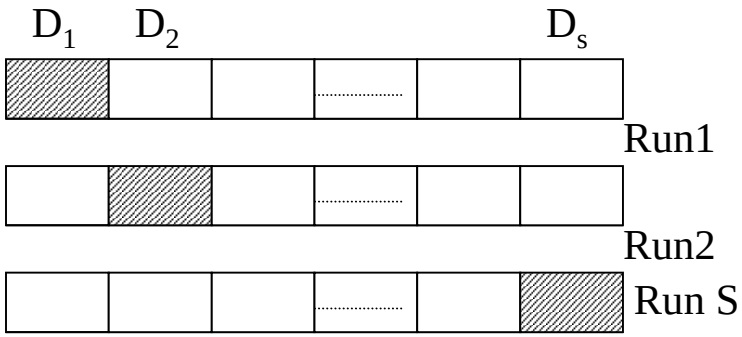
S-fold Cross Validation

- *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

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

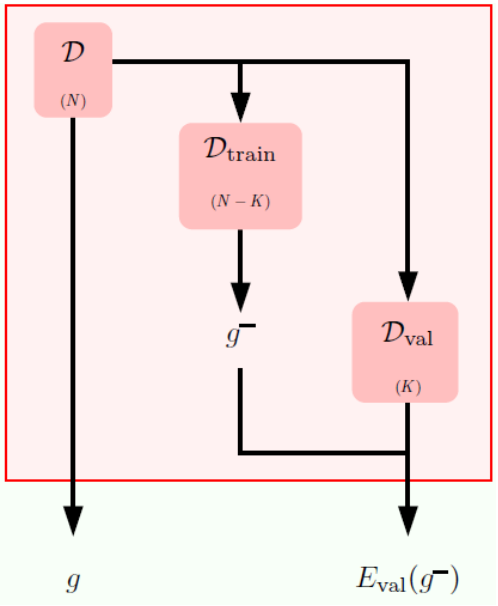
- $S=N \Rightarrow N$-fold cross validation or Leave one-out CV (LOOCV) method

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

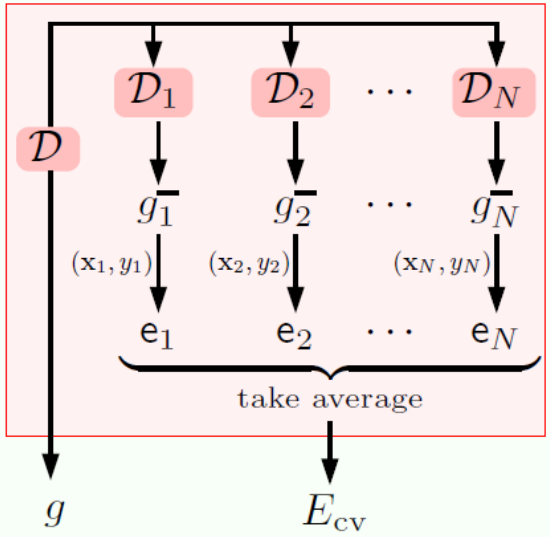
- Practical Scheme: 5x2 Cross-Validation. Variation: 5 repetitions of 2-fold cross-validation on a randomized dataset

Summary of Validation and Model Selection Methods

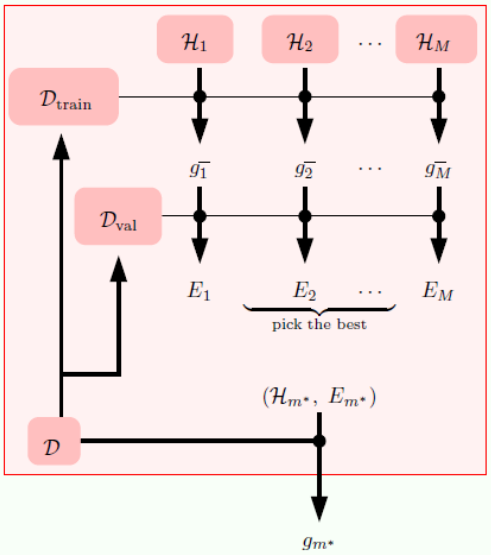
Simple split



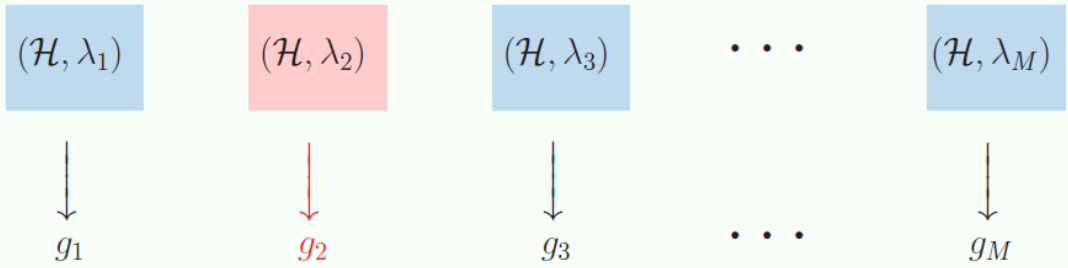
Leave one out CV
N=S for S-fold CV



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)

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

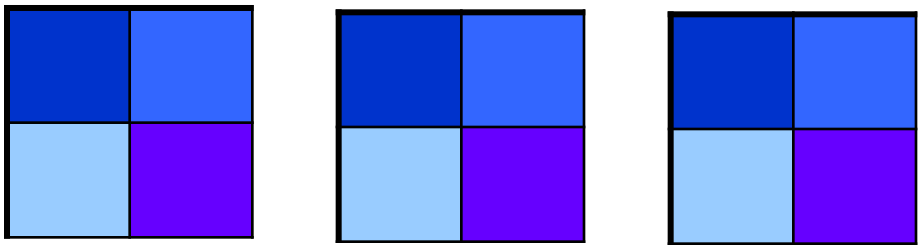
$\text{Pr}\,ecision(P) = 0.666 \qquad P_F = R(1-P)/P = 0.304$

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

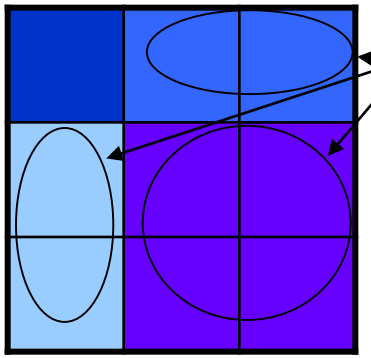
Confusion Matrices for Multiple Classes - 1

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

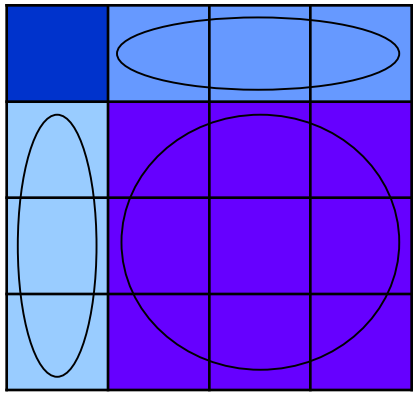


One-versus-All Confusion Matrix

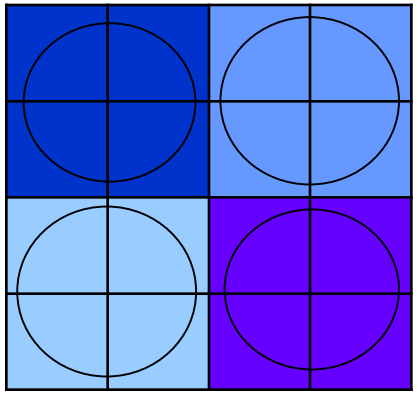
Confusion Matrices for Multiple Classes - 2

- **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



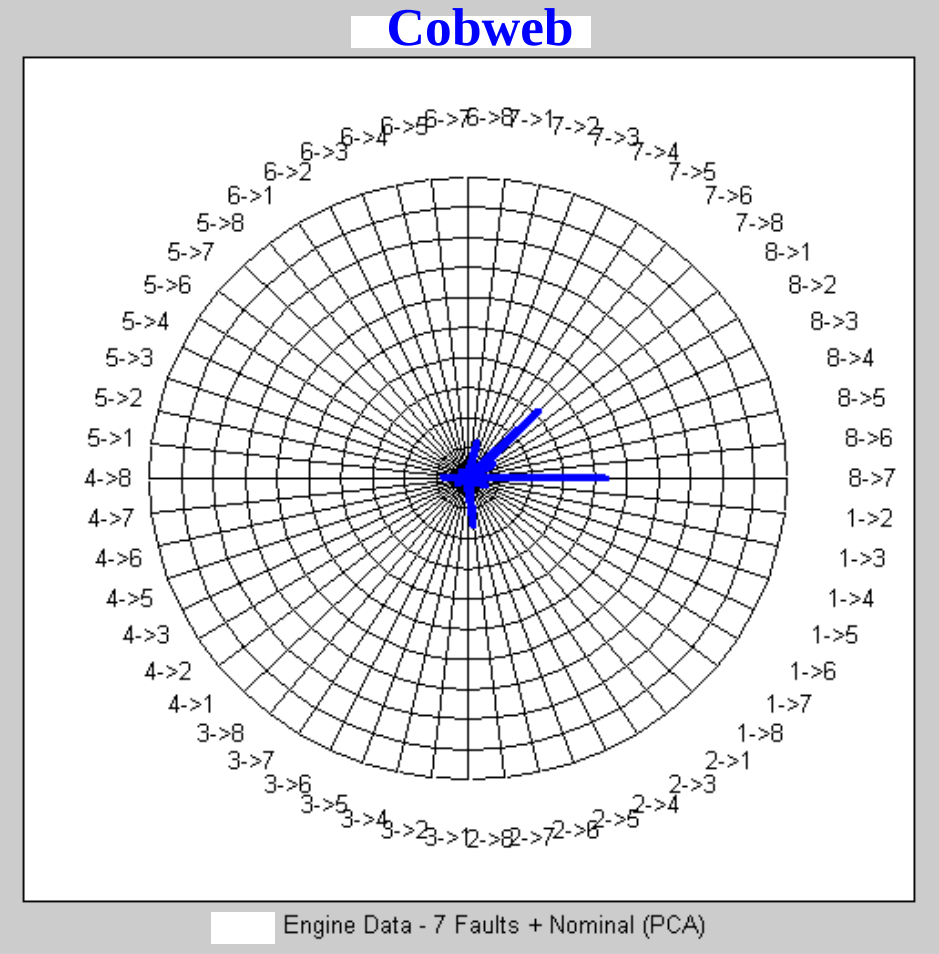
C1 vs. C2 & C3 & C4



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.

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
- Discriminant Learning: Logistic Regression
- Performance Assessment of Classifiers