# Lecture 10: Radial Basis Functions, Gaussian Processes, Relevance Vector Machines, Feature Selection and Dimensionality Reduction

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

***Spring 2021***
***April 6, 2021***

# Lecture Outline

- Radial Basis Functions
- Gaussian Processes
- Relevant Vector Machines
- Feature Selection
- Dimensionality Reduction
- Summary

# What are Radial Basis Functions ?

- **Radial Basis Functions (RBFs)**

**MLP**

Activation function is determined by the inner product

$$y = \underline{w}^T \underline{x} \quad and \ \hat{z} = g(y) = g(\underline{w}^T \underline{x})$$

**RBF (Recall Kernel Idea of SVM and PNN)**

Activation of hidden units is determined by the distance between the input vector and a prototype vector. Activation is highest at the center and radiates outward with intensity decreasing toward zero as the distance from the center increases, $\phi\left(\left\|\underline{x} - \underline{x}^n\right\|\right)$

## Characterizing RBF Networks

- Direct Connections between inputs and outputs also help a great deal
- *Hyperspheres* versus *Hyperplanes*
- Hidden units in RBF are *local* versus *global* in MLP
- Simple architecture
- Faster to train

$$y_k(\underline{x}) = \sum_{j=1}^{M} w_{kj}\phi_j(\underline{x}) + w_{k0} = \sum_{j=0}^{M} w_{kj}\phi_j(\underline{x})$$

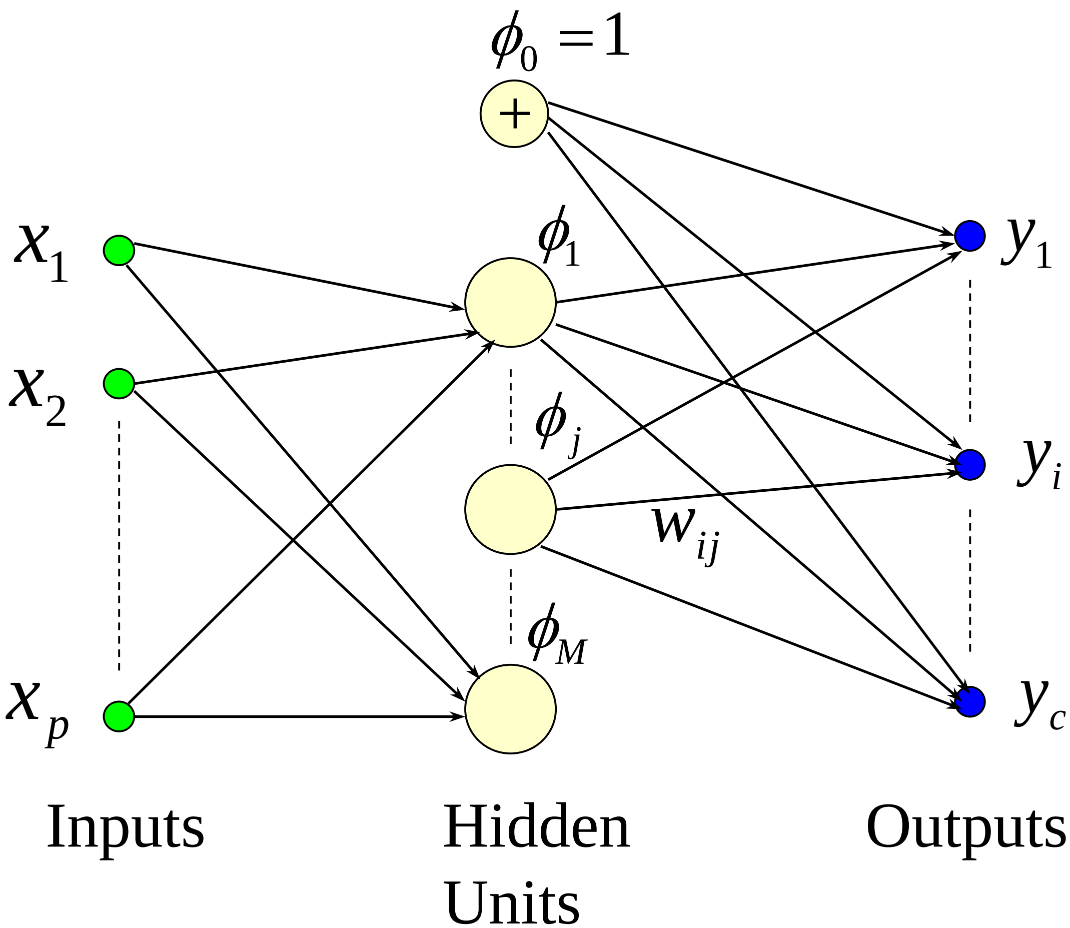
or $\underline{y} = W\,\underline{\phi}(\underline{x})$ $\quad W$ is a $C$ by $(M+1)$ matrix

Typically, the basis functions $\phi_j(\underline{x}),\ j \geq 1$ are given by

$$\varphi_j(\underline{x}) = \exp\left\{-\frac{\left\|\underline{x}-\underline{\mu}_j\right\|^2}{2\sigma_j^2}\right\};\ \varphi_j(\underline{x}) = \exp\left\{-\frac{1}{2}(\underline{x}-\underline{\mu}_j)^T \Sigma_j^{-1}(\underline{x}-\underline{\mu}_j)\right\}$$

# Structure of an RBF Network

- In fact, RBF network is a three layer nonlinear regression system



# Training RBF Weights - 1

- If know $\{\phi_j(\underline{x})\}$ and training data $\{\underline{x}^n, \underline{z}^n\}_{n=1}^{N}$, then the weights that minimize the squared error

$$E = \frac{1}{2}\sum_{n=1}^{N}\left\|\underline{y}(\underline{x}^n, W) - \underline{z}^n\right\|^2 + \underbrace{\frac{\lambda}{2} tr(W^T W)}_{Regularization\ term}$$

can be obtained as follows

$$E = \frac{1}{2}\sum_{n=1}^{N}\left\|W\ \underline{\phi}(\underline{x}^n) - \underline{z}^n\right\|^2 + \frac{\lambda}{2} tr(W^T W)$$

$$E = \frac{1}{2}\sum_{n=1}^{N} tr\left[\left(W\ \underline{\phi}(\underline{x}^n) - \underline{z}^n\right)\left(W\ \underline{\phi}(\underline{x}^n) - \underline{z}^n\right)^T\right] + \frac{\lambda}{2} tr(W^T W)$$

$$= \frac{1}{2}\sum_{n=1}^{N} tr\left[W\ \underline{\phi}(\underline{x}^n)\underline{\phi}^T(\underline{x}^n)W^T - \underline{z}^n\underline{\phi}^T(\underline{x}^n)W^T - W\ \underline{\phi}(\underline{x}^n)\underline{z}^{nT} + \underline{z}^n\underline{z}^{nT}\right]$$

$$+ \frac{\lambda}{2} tr(W^T W)$$

# Training RBF Weights - 2

$$\nabla_{\mathrm{w}} E = W\sum_{n=1}^{N} \underline{\phi}(\underline{x}^n)\underline{\phi}^T(\underline{x}^n) - \sum_{n=1}^{N} \underline{z}^n \underline{\phi}^T(\underline{x}^n) + \lambda W$$

or $$\left[\sum_{n=1}^{N} \underline{\phi}(\underline{x}^n)\underline{\phi}^T(\underline{x}^n)\right] W^T + \lambda W^T = \sum_{n=1}^{N} \underline{\phi}(\underline{x}^n)\underline{z}^{nT}$$

Let

$$\Phi =_N \overset{M+1}{\begin{bmatrix} \underline{\phi}^T(\underline{x}^1) \\ \underline{\phi}^T(\underline{x}^2) \\ \vdots \\ \underline{\phi}^T(\underline{x}^n) \end{bmatrix}}; \qquad Z =_N \overset{C}{\begin{bmatrix} \underline{z}^{1T} \\ \underline{z}^{2T} \\ \vdots \\ \underline{z}^{NT} \end{bmatrix}}$$

$$(\Phi^T\Phi + \lambda I_{M+1})W^T = \Phi^T Z$$

$$\underline{y}(\underline{x}) = W\underline{\phi}(\underline{x}) = Z^T \underbrace{(\Phi\Phi^T + \lambda I_N)^{-1}\Phi\underline{\phi}(\underline{x})}_{\text{all inner products}} = D^T \underbrace{\Phi\underline{\phi}(\underline{x})}_{\text{kernel vector } \underline{g}(\underline{x})}$$

M+1 by M+1  M+1 by C  M+1 by N  N by C

$$W^T = (\Phi^T\Phi + \lambda I_{M+1})^{-1}\Phi^T Z = \Phi^T \underbrace{\underbrace{(\Phi\Phi^T + \lambda I_N)}_{\text{Kernel Matrix } G+\lambda I_N}{}^{-1} Z}_{D \text{ NxC matrix}} = \Phi^T D$$

Solution via: G-S, SVD, RLS, LMS, . . .

# Optimal Weights of RBF - 1

Thus, we can obtain each column of $D$ independently

$$y_k(\underline{x}) = \sum_{n=1}^{N} d_{n,k}\, g(\underline{x}, \underline{x}^n) = \underline{d}_k^T \underline{g}(\underline{x});\ \underline{g}(\underline{x}) = \begin{bmatrix} g(\underline{x}, \underline{x}^1) \\ g(\underline{x}, \underline{x}^2) \\ . \\ g(\underline{x}, \underline{x}^N) \end{bmatrix} = \begin{bmatrix} \underline{\phi}^T(\underline{x}^1)\underline{\phi}(\underline{x}) \\ \underline{\phi}^T(\underline{x}^2)\underline{\phi}(\underline{x}) \\ . \\ \underline{\phi}^T(\underline{x}^N)\underline{\phi}(\underline{x}) \end{bmatrix}$$

$$\begin{bmatrix} g(\underline{x}^1, \underline{x}^1) + \lambda & g(\underline{x}^1, \underline{x}^2) & \cdots & g(\underline{x}^1, \underline{x}^N) \\ g(\underline{x}^2, \underline{x}^1) & g(\underline{x}^2, \underline{x}^2) + \lambda & \cdots & g(\underline{x}^2, \underline{x}^N) \\ \vdots & \vdots & \ddots & \vdots \\ g(\underline{x}^N, \underline{x}^1) & g(\underline{x}^N, \underline{x}^2) & \cdots & g(\underline{x}^N, \underline{x}^N) + \lambda \end{bmatrix} \underbrace{\begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_N \end{bmatrix}}_{\text{A column of } D} = \underbrace{\begin{bmatrix} z^1 \\ z^2 \\ \vdots \\ z^N \end{bmatrix}}_{\text{Corresponding column of } Z}$$

$$\underline{d}_k = \left(G + \lambda I_N\right)^{-1} \underline{z}_k = \left(\Phi\Phi^T + \lambda I_N\right)^{-1} \underline{z}_k;\ k = 1, 2, .., C$$

$$y_k(\underline{x}) = \underline{d}_k^T \underline{g}(\underline{x}) = \underline{z}_k^T \left(G + \lambda I_N\right)^{-1} \underline{g}(\underline{x})$$

$$= \underline{g}^T(\underline{x})\left(G + \lambda I\right)^{-1} \underline{z}_k$$

**Kernel Ridge Regression**

# Selecting Basis Functions

- Key questions:
  - Why Gaussian basis functions?
  - How to choose $M$?
  - How to choose the basis function parameters?
- **Why Gaussian basis functions?**

  The motivation comes from regularization theory. Consider a single output problem for simplicity and consider the error function

$$J = \frac{1}{2}\sum_{n=1}^{N}(y(\underline{x}^n) - z^n)^2 + \frac{\lambda}{2}\int_{\underline{x}} |Py|^2 \, d\underline{x} \qquad \cdots\cdots(1)$$

where $P$ is some linear operator (e.g., differential operator) and $\lambda$ is a regularization parameter. $\lambda$ controls the smoothness of the fit.

# Smoothing Functions

- Some examples of *P*

  – $$\int_{\underline{x}} |Py|^2 d\underline{x} = \int_{\underline{x}} \sum_{i=1}^{p} \left( \frac{\partial^2 y}{\partial x_i^{\,2}} \right)^2 d\underline{x} \quad \Rightarrow \text{reduce curvature}$$

  – $$\int_{\underline{x}} |Py|^2 d\underline{x} = \int_{\underline{x}} \sum_{i=1}^{p} \sum_{m=0}^{\infty} a_m \left( \frac{\partial^m y}{\partial x_i^{\,m}} \right)^2 d\underline{x}$$

  It can be shown that if $a_m = \dfrac{\sigma^{2m}}{m!\,2^m}$, then $y(\underline{x}^n)$ is a sum of Gaussian RBFs.

  – $$\int_{\underline{x}} |Py|^2 d\underline{x} = \int_{\underline{x}} \sum_{i=1}^{p} \left( \frac{\partial y}{\partial x_i} \right)^2 d\underline{x} \quad \Rightarrow \textit{to reduce sensitivity} \text{ w.r.t. } x_i$$

# Generalized Regression Network

- As in PNN, we can select $M = N$. In practice, take $M < N$ basis functions (e.g., via K-means, GMM,… clustering)
- Have different $\sigma_i^{\,2}$ for each unit.

$$\Rightarrow \; y(\underline{x}) = \sum_{n=1}^{M} w_n e^{-\left\|\underline{x}-\underline{\mu}_n\right\|^2/2\sigma_n^{\,2}} \quad \Rightarrow \text{Generalized RBF}$$

- ***Relationships to GRNN and Regression***
  - Suppose have training data $\left\{\underline{x}^n, \underline{z}^n\right\}$
  - Joint density estimate via Gaussian kernels (recall PNN)

$$\hat{p}(\underline{x}, \underline{z}) = \frac{1}{N}\sum_{n=1}^{N} \frac{1}{\left(2\pi\sigma_x^{\,2}\right)^{p/2}\left(2\pi\sigma_z^{\,2}\right)^{c/2}} \exp\left\{\frac{-\left\|\underline{x}-\underline{x}^n\right\|^2}{2\sigma_x^{\,2}} - \frac{\left\|\underline{z}-\underline{z}^n\right\|^2}{2\sigma_z^{\,2}}\right\}$$

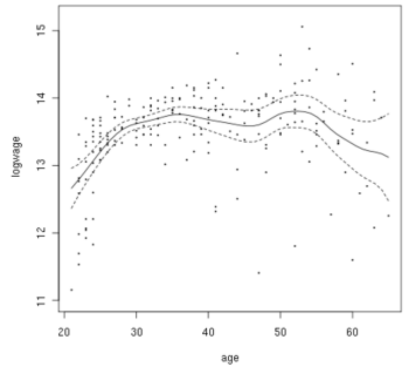
# Nadaraya – Watson Estimator

- **MMSE estimate = Conditional mean**

$$y(\underline{x}) = E\{\underline{z}/\underline{x}\} = \int \underline{z}\, p(\underline{z}/\underline{x})\, d\underline{z}$$

$$= \frac{\int_{\underline{z}} \underline{z}\, p(\underline{x}, \underline{z})\, d\underline{z}}{\int_{\underline{z}} p(\underline{x}, \underline{z})\, d\underline{z}}$$

$$= \frac{\sum_{n=1}^{N} \underline{z}^n \exp\left\{-\left\|\underline{x} - \underline{x}^n\right\|^2 / 2\sigma_x^{\ 2}\right\}}{\sum_{n=1}^{N} \exp\left\{-\left\|\underline{x} - \underline{x}^n\right\|^2 / 2\sigma_x^{\ 2}\right\}} = \sum_{n=1}^{N} k(\underline{x}, \underline{x}^n)\, \underline{z}^n$$



Estimated Regression Function.

$$k(\underline{x}, \underline{x}^n) = \frac{\exp\left\{-\left\|\underline{x} - \underline{x}^n\right\|^2 / 2\sigma_x^{\ 2}\right\}}{\sum_{i=1}^{N} \exp\left\{-\left\|\underline{x} - \underline{x}^i\right\|^2 / 2\sigma_x^{\ 2}\right\}}$$

**Nadaraya-Watson Estimator**

# RBF Networks for Classification

- RBF Networks for classification

Let $$p(\underline{x} \mid z=k) = \sum_{j=1}^{M} p(\underline{x} \mid j) P(j \mid z=k)$$

$$p(z=k \mid \underline{x}) = \frac{p(\underline{x} \mid z=k) P(z=k)}{p(\underline{x})}$$

$$= \frac{\sum_{j=1}^{M} P(j \mid z=k)\, p(\underline{x} \mid j) \frac{P_j}{P_j} P(z=k)}{\sum_{l=1}^{M} p(\underline{x} \mid l) P_l}$$

$$= \sum_{j=1}^{M} \underbrace{\left[ \frac{p(\underline{x} \mid j) P_j}{\sum_{l=1}^{M} p(\underline{x} \mid l) P_l} \right]}_{\phi_j(\underline{x}) = P(j/\underline{x})} \underbrace{\left[ \frac{P(j \mid z=k) P(z=k)}{P_j} \right]}_{w_{kj} = P(z=k \mid j)} = \sum_{j=1}^{M} \phi_j(\underline{x}) w_{kj}$$

Normalized RBF $\phi_j(\underline{x}) = P(j/\underline{x})$ RBF center given $\underline{x}$

$w_{kj} = P(z=k \mid j)$ Output class given RBF center

# Probabilistic Interpretation of Normalized RBF

- Note that
  - $\phi_j(\underline{x}) = P(j \mid \underline{x})$ $\Rightarrow$ posterior probability of mixture component $j$ given $\underline{x}$
  - $w_{kj} = P(z = k \mid j)$ $\Rightarrow$ posterior probability of class $k$ given mixture component $j$

Alternately,

$$P(z = k \mid \underline{x}) = \sum_{j=1}^{M} P(z = k, j \mid \underline{x}) = \sum_{j=1}^{M} P(z = k \mid j, \underline{x}) P(j \mid \underline{x})$$

$$= \sum_{j=1}^{M} P(z = k \mid j) P(j \mid \underline{x}) = \sum_{j=1}^{M} w_{kj} \varphi_j(\underline{x})$$

# Optimizing RBF Parameters - 1

- **How to choose the basis function parameters ?**
  - Supervised training . . . Batch mode

$$y_k(\underline{x}) = \sum_{j=0}^{M} w_{kj}\phi_j(\underline{x})$$

where

$$\phi_j(\underline{x}) = \sum_{j=1}^{M} \exp\left\{-\frac{\left\|\underline{x}-\underline{\mu}_j\right\|^2}{2\sigma_j^2}\right\} \qquad j \geq 1$$

$$\phi_0(\underline{x}) = 1$$

Criterion: $$\min_{\{W,\{\underline{\mu}_j\}\{\sigma_j\}\}} J = \frac{1}{2}\sum_{n=1}^{N}\sum_{k=1}^{C}\underbrace{\left[z_k^n - \sum_{j=0}^{M} w_{kj}\varphi_j(\underline{x}^n)\right]}_{e_k^n}{}^2$$

# Selecting Basis Function Parameters - 2

$$\frac{\partial J}{\partial w_{kj}} = -\sum_{n=1}^{N} e_k{}^n \varphi_j(\underline{x}^n) \qquad \text{or,} \;\; \nabla_w J = -\underline{e}^n \underline{\varphi}^{n^T} \qquad \cdots(1)$$

where

$$\underline{e}^n = \begin{bmatrix} e_1{}^n \\ e_2{}^n \\ \vdots \\ e_C{}^n \end{bmatrix}; \qquad \underline{\varphi}^n = \begin{bmatrix} \varphi_0 \\ \varphi_1 \\ \vdots \\ \varphi_M \end{bmatrix}$$

$$\nabla_{\underline{\mu}_j} J = -\sum_{n=1}^{N}\sum_{k=1}^{C} e_k{}^n w_{kj} \varphi_j(\underline{x}^n) \frac{(\underline{x}^n - \underline{\mu}_j)}{\sigma_j{}^2} \qquad \cdots(2)$$

$$\nabla_{\sigma_j} J = -\sum_{n=1}^{N}\sum_{k=1}^{C} e_k{}^n w_{kj} \varphi_j(\underline{x}^n) \frac{\left\| \underline{x}^n - \underline{\mu}_j \right\|^2}{\sigma_j{}^3} \qquad \cdots(3)$$

# Optimization Algorithms

- A variety of implementations are possible:

i) NLP techniques using (1), (2) and (3)

ii) Coordinate descent (alternating optimization) algorithms:

a) For a given $\{\underline{\mu}_j\},\{\sigma_j\}$, find $W$ via least squares

b) For a given $W$, find $\{\underline{\mu}_j\},\{\sigma_j\}$ via NLP techniques or by setting gradients to zero for $\{\underline{\mu}_j\}$ and by grid search over $\{\sigma_j\}$

iii) Same as i) or ii) with $M$ and initial $\{\underline{\mu}_j\},\{\sigma_j\}$ via unsupervised techniques… K-means, GMM,.. Lecture 11

iv) Extended Kalman Filter (treat parameters as states with identity transition matrix and process noise, and observations $(\underline{x}, \underline{z})$ as training data).

# Gaussian Processes (GP) for Regression

Regression:

Let $y(\underline{x}) = \underline{w}^T \underline{\phi}(\underline{x}) = \underline{\phi}^T(\underline{x})\underline{w}; \quad p(\underline{w}) = N(\underline{0}, \sigma_w^2 I) \Rightarrow p(y(\underline{x}) | \underline{x}) = N(0, \sigma_w^2 \underline{\phi}^T(\underline{x})\underline{\phi}(\underline{x}))$...*note inner product*

Let $\underline{y}^N = [y(\underline{x}^1), y(\underline{x}^2), \ldots, y(\underline{x}^N)]^T; \Phi_N = \begin{bmatrix} \underline{\phi}^T(\underline{x}^1) \\ \underline{\phi}^T(\underline{x}^2) \\ . \\ \underline{\phi}^T(\underline{x}^N) \end{bmatrix}; \underline{X}^N = [\underline{x}^1, \underline{x}^2, \ldots, \underline{x}^N]$

$$\underline{y}^N = \Phi_N \underline{w} \Rightarrow p(\underline{y}^N | \underline{X}^N) = N(\underline{0}, \sigma_w^2 \Phi_N \Phi_N^T) = N(0, \Sigma_{yN})$$

$\sigma_w^2 \Phi \Phi^T = K \Rightarrow k_{ij} = k(\underline{x}^i, \underline{x}^j) = \sigma_w^2 \underline{\phi}^T(\underline{x}^i)\underline{\phi}(\underline{x}^j) = Cov(y(\underline{x}^i), y(\underline{x}^j)) \Rightarrow$ it is a Kernel

$$Typical\ k(\underline{x}_i, \underline{x}_j) = \theta_0 \exp[-(\underline{x}^i - \underline{x}^j)^T Diag(\sigma_l^{-2})(\underline{x}^i - \underline{x}^j)] + \theta_1 + \theta_2 \underline{x}^{i^T} \underline{x}^j$$

Measured Targets: $z^n = y(\underline{x}^n) + v^n;\ p(v^n) = N(0, \sigma_v^2);\ \underline{z}^N = [z^1, z^2 \ldots, z^N]^T$

$$\boxed{\underline{z}^N = \underline{y}^N + \underline{v}^N = \Phi^N \underline{w} + \underline{v}^N}$$

$$\underline{z}^N = \underline{y}^N + \underline{v}^N \Rightarrow p(\underline{z}^N | \underline{y}^N) = N(\underline{y}^N, \sigma_v^2 I_N)$$

$$\Rightarrow p(\underline{z}^N | \underline{X}^N) = \int_{\underline{y}^N} p(\underline{z}^N | \underline{y}^N) p(\underline{y}^N | \underline{X}^N) d\underline{y}^N = N(\underline{0}, \Sigma_y^N + \sigma_v^2 I_N) = N(\underline{0}, \Sigma_z^N)$$

$$Note: p(\underline{z}^{N+1} | \underline{X}^{N+1}) = N(\Sigma_y^{N+1} + \sigma_v^2 I_{N+1}) = N(\underline{0}, \Sigma_z^{N+1}) = \begin{bmatrix} \overbrace{\Sigma_{yN} + \sigma_v^2 I_N}^{\Sigma_z^N} & \overbrace{\sigma_w^2 \Phi_N \underline{\phi}(\underline{x}^{N+1})}^{\underline{k}} \\ \underbrace{\sigma_w^2 \underline{\phi}^T(\underline{x}^{N+1}) \Phi_N^T}_{\underline{k}^T} & \sigma_v^2 + \sigma_w^2 \underline{\phi}^T(\underline{x}^{N+1}) \underline{\phi}(\underline{x}^{N+1}) \end{bmatrix}$$

# GP-based Prediction

$\Pr ediction \Rightarrow$ Conditional mean and variance

$$p(z^{N+1} \mid \underline{z}^N, \underline{X}^{N+1}) = \frac{p(\underline{z}^{N+1}, \underline{X}^{N+1})}{p(\underline{z}^N, \underline{X}^{N+1})} = \frac{p(\underline{z}^{N+1} \mid \underline{X}^{N+1})}{\int_{z^{N+1}} p(\underline{z}^{N+1} \mid \underline{X}^{N+1}) dz^{N+1}}$$

$$p(z^{N+1} \mid \underline{z}^N, \underline{X}^{N+1}) = N(\overbrace{\underline{k}^T \underbrace{\left(\Sigma_z^N\right)^{-1} \underline{z}^N}_{\underline{\alpha}}}^{E\{z^{N+1} \mid \underline{z}^N, \underline{X}^{N+1}\}}, \overbrace{\sigma_v^2 + \sigma_w^2 \underline{\phi}^T(\underline{x}^{N+1}) \underline{\phi}(\underline{x}^{N+1}) - \underline{k}^T \left(\Sigma_z^N\right)^{-1} \underline{k}}^{Var(z^{N+1} \mid \underline{z}^N, \underline{X}^{N+1})})$$

$$Note: E\{z^{N+1} \mid \underline{z}^N, \underline{X}^{N+1}\} = \underbrace{\sigma_w^2 \underline{\phi}^T(\underline{x}^{N+1}) \Phi_N^T}_{\underline{k}^T(\underline{x}^N, \underline{x}^{N+1})} \left(\Sigma_z^N\right)^{-1} \underline{z}^N = \underline{k}^T(\underline{x}^N, \underline{x}^{N+1}) \underline{\alpha}$$

$$where\ \underline{\alpha} = \left(\Sigma_z^N\right)^{-1} \underline{z}^N$$

$$\underline{k}^T(\underline{x}^N, \underline{x}^{N+1}) = \sigma_w^2 [\underbrace{\underline{\phi}^T(\underline{x}^{N+1}) \underline{\phi}(\underline{x}^1), \underline{\phi}^T(\underline{x}^{N+1}) \underline{\phi}(\underline{x}^2), \ldots\ldots, \underline{\phi}^T(\underline{x}^{N+1}) \underline{\phi}(\underline{x}^N)}_{inner\ products}]$$

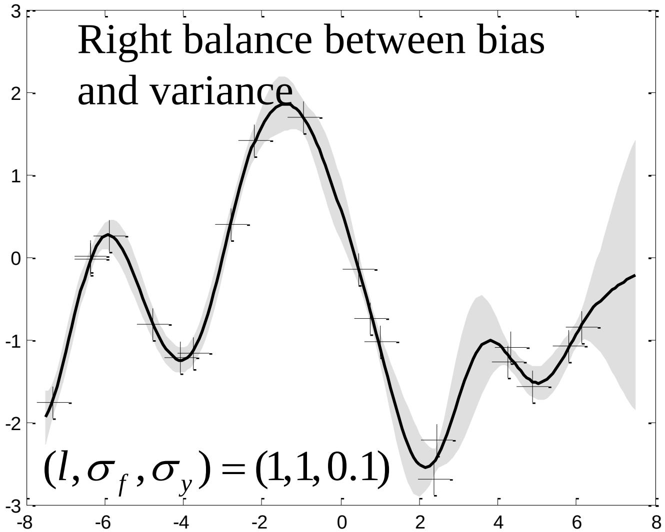
$$\underline{\phi}^T(\underline{x}^{N+1}) \underline{\phi}(\underline{x}^j) = \sigma_f^2 \exp\left(-\frac{1}{2}(\underline{x}^{N+1} - \underline{x}^j)^T diag(l_i^{-2})(\underline{x}^{N+1} - \underline{x}^j)\right) + \sigma_y^2 I_{p+1}$$

$l_i =$ *horizontal scale over which the function changes along coordinate i*

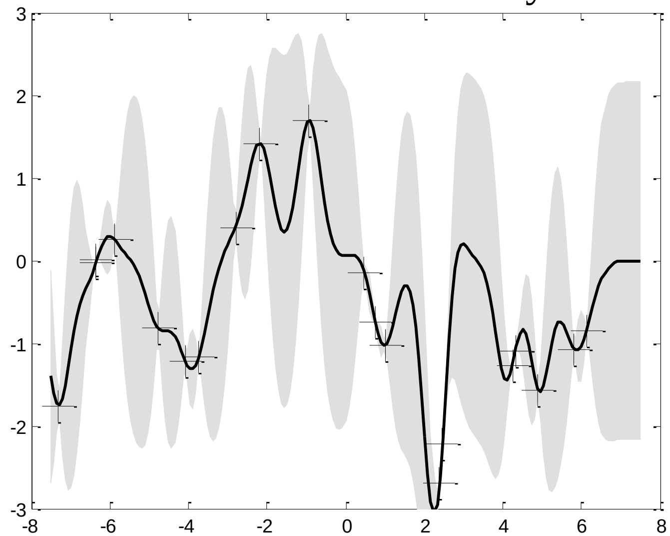
$\sigma_f^2 =$ *controls the vertical scale of the function*

$\sigma_y^2 =$ *noise* var*iance*

# GP-based Prediction

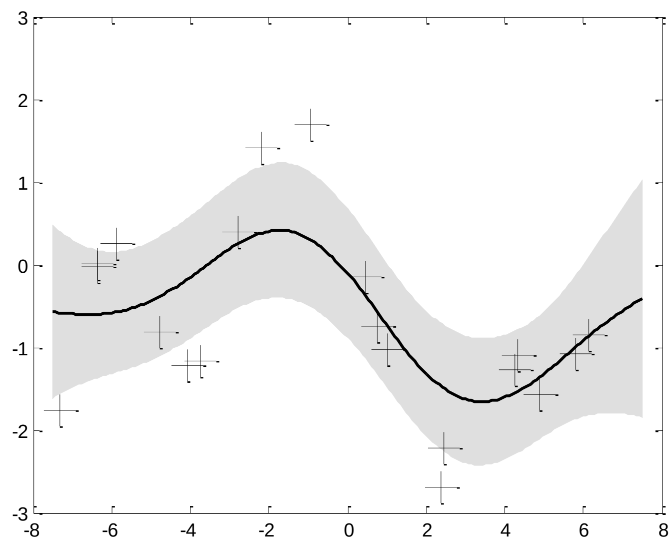



Too much variability



$(l, \sigma_f, \sigma_y) = (0.3, 0.108, 0.00005)$

Too much bias



$(l, \sigma_f, \sigma_y) = (3.0, 1.16, 0.89)$

# GP-based Learning

*Learning Kernel Parameters* $\{l_i, \sigma_f^2, \sigma_y^2\}$ *via ML* :

$$\ln p(\underline{z}^N, \underline{X}^N \mid \underline{\theta}) = -\frac{N}{2}\ln 2\pi - \frac{1}{2}\ln|\Sigma_z^N| - \frac{1}{2}\underline{z}^{N^T}\left(\Sigma_z^N\right)^{-1}\underline{z}^N$$

$$\frac{\partial}{\partial\theta_j}\ln p(\underline{z}^N, X^N \mid \underline{\theta}) = -\frac{1}{2}tr[\left(\Sigma_z^N\right)^{-1}\frac{\partial\Sigma_z^N}{\partial\theta_j}] + \frac{1}{2}\underline{z}^{N^T}\left(\Sigma_z^N\right)^{-1}\frac{\partial\Sigma_z^N}{\partial\theta_j}\left(\Sigma_z^N\right)^{-1}\underline{z}^N$$

$$= \frac{1}{2}tr[\left(\underline{\alpha}\underline{\alpha}^T - \left(\Sigma_z^N\right)^{-1}\right)\frac{\partial\Sigma_z^N}{\partial\theta_j}]$$

*where* $\underline{\alpha} = \left(\Sigma_z^N\right)^{-1}\underline{z}^N$

*Rank ordering features* :

**Large scale** $l_i \Rightarrow$ **feature** $i$ **is not relevant**

# GP Implementation via Kalman Filter

$$\underline{w}^{n+1} = \underline{w}^{n}; \underline{w}^{0} \sim N(\underline{0}, \sigma_w^2 I)$$
$$z^n = \underline{\phi}^T(\underline{x}^n)\underline{w}^n + v^n; v^n \sim N(0, \sigma_v^2)$$

*Measurement Update:*

$$\hat{\underline{w}}^{N/N} = \Sigma^{N|N}\left(\Sigma^{N-1|N-1}\right)^{-1}\hat{\underline{w}}^{N-1/N-1} + \frac{\Sigma^{N|N}\underline{\phi}(\underline{x}^n)}{\sigma_v^2} z^N$$

$$= \left(I + \Sigma^{N-1|N-1}\frac{\underline{\phi}(\underline{x}^N)\underline{\phi}^T(\underline{x}^N)}{\sigma_v^2}\right)^{-1}\hat{\underline{w}}^{N-/N-1} + \frac{\Sigma^{N|N}\underline{\phi}(\underline{x}^n)}{\sigma_v^2} z^N$$

$$= \hat{\underline{w}}^{N-/N-1} + K_n\left(z^N - \underline{\phi}^T(\underline{x}^N)\hat{\underline{w}}^{N-/N-1}\right)$$

**where** $K_n = \dfrac{\Sigma^{N-1|N-1}\underline{\phi}(\underline{x}^N)}{\sigma_v^2 + \underline{\phi}^T(\underline{x}^N)\Sigma^{N-1|N-1}\underline{\phi}(\underline{x}^N)} = \dfrac{\Sigma^{N|N}\underline{\phi}(\underline{x}^N)}{\sigma_v^2} =$ **Kalman Gain**

$$So, E\{z^{N+1} \mid \underline{z}^N, \underline{X}^{N+1}\} = \underline{\phi}^T(\underline{x}^{N+1})\hat{\underline{w}}^{N/N}$$

$Var\{z^{N+1} \mid \underline{z}^N, \underline{X}^{N+1}\} = \underline{\phi}^T(\underline{x}^{N+1})\Sigma^{N|N}\underline{\phi}(\underline{x}^{N+1}) + \sigma_v^2 =$ **Innovation variance**

Note:This is recursive, but lose Kernel advantage (requires $\phi(\underline{x})$)

# GP for Classification

**Classification : Remember continuous output goes through a logistic**

$$p(z\,|\,y)=\sigma(y)^{z}[1-\sigma(y)]^{1-z};\ \sigma(y)=[1+\exp(-y)]^{-1}=\hat{z};\ y(\underline{x})=\underline{w}^{T}\phi(\underline{x})$$

$$p(\underline{y}^{N}\,|\,\underline{X}^{N})=N(\underline{0},\sigma_w^2\Phi\Phi^T+\varepsilon I_N)=N(\underline{0},\Sigma_y^N);\ \varepsilon\ \textbf{for numerical stability}$$

$$p(z^{N+1}=1\,|\,\underline{z}^{N},\underline{X}^{N+1})=\int_{y(\underline{x}^{N+1})} p(z^{N+1}=1|y(\underline{x}^{N+1}))\,p(y(\underline{x}^{N+1})\,|\,\underline{z}^{N},\underline{X}^{N+1})dy(\underline{x}^{N+1})$$

$$=\int_{y(\underline{x}^{N+1})}\sigma(y(\underline{x}^{N+1}))\,p(y(\underline{x}^{N+1})\,|\,\underline{z}^{N},\underline{X}^{N+1})dy(\underline{x}^{N+1})$$

$\Rightarrow$ **integration of logistic**

$$\boxed{\begin{array}{l}\underline{w}^{n+1}=\underline{w}^{n};\ \underline{w}^{0}\sim N(\underline{0},\sigma_w^2 I)\\ z^n=\operatorname{sgn}\left[\underbrace{\underline{\phi}^T(\underline{x}^n)\underline{w}^n+v^n}_{y^n}\right];v^n\sim N(0,\varepsilon)\end{array}}$$

**Laplace Approximation: Assume** $p(y(\underline{x}^{N+1})\,|\,\underline{z}^{N},\underline{X}^{N+1})$ **is Gaussian**

$$p(y(\underline{x}^{N+1})\,|\,\underline{z}^{N},\underline{X}^{N+1})=\int_{\underline{y}^{N}} p(y(\underline{x}^{N+1})\,|\,\underline{y}^{N},\underline{x}^{N+1})\,p(\underline{y}^{N}\,|\,\underline{z}^{N},\underline{X}^{N})d\underline{y}^{N}$$

$$p(y(\underline{x}^{N+1})\,|\,\underline{y}^{N})=N(\underline{k}^{T}\left(\Sigma_y^N\right)^{-1}\underline{y}^{N},\sigma_w^2\underline{\phi}^T(\underline{x}^{N+1})\underline{\phi}(\underline{x}^{N+1})+\varepsilon-\underline{k}^{T}\left(\Sigma_y^N\right)^{-1}\underline{k})$$

$$\underline{k}^{T}=[k(\underline{x}^{1},\underline{x}^{N+1}),k(\underline{x}^{2},\underline{x}^{N+1}),\ldots,k(\underline{x}^{N},\underline{x}^{N+1})]$$

$$p(\underline{y}^{N}\,|\,\underline{z}^{N},\underline{X}^{N})\propto p(\underline{z}^{N}\,|\,\underline{y}^{N})\,p(\underline{y}^{N}\,|\,\underline{X}^{N})=\prod_{n=1}^{N}[1-\sigma(y(\underline{x}^{n}))][\frac{\sigma(y(\underline{x}^{n}))}{1-\sigma(y(\underline{x}^{n}))}]^{z^n}N(\underline{0},\Sigma_y^N)$$

# GP and Probit Approximation

$Use\ the\ \text{mode}\ of\ \ \ln p(\underline{y}^N \mid \underline{z}^N, \underline{X}^N)\ as\ mean\ and\ (\text{-}Hessian)^{-1}\ as\ \text{cov}ariance\ :$

$$\ln p(\underline{y}^N \mid \underline{z}^N, \underline{X}^N) \propto -\sum_{n=1}^{N} \ln(1+e^{y(\underline{x}^n)}) + \left(\underline{z}^N\right)^T \underline{y}^N - \frac{1}{2}\ln|\Sigma_y^N| - \frac{1}{2}\underline{y}^{N^T}(\Sigma_y^N)^{-1}\underline{y}^N$$

$$\nabla_{\underline{y}^N} \ln p(\underline{y}^N \mid \underline{z}^N, \underline{X}^N) = -\underline{\sigma}(\underline{y}^N) + \underline{z}^N - (\Sigma_y^N)^{-1}\underline{y}^N$$

$$where\ \ \underline{\sigma}(\underline{y}^N) = [\sigma(y(\underline{x}^1)), \sigma(y(\underline{x}^2)), \ldots., \sigma(y(\underline{x}^N))^T$$

$$\nabla^2_{\underline{y}\underline{y}} \ln p(\underline{y}^N \mid \underline{z}^N, \underline{X}^{N+1}) = -Diag[\sigma(y(\underline{x}^n))(1-\sigma(y(\underline{x}^n))] - (\Sigma_y^N)^{-1} = -D_N - (\Sigma_y^N)^{-1}$$

$$p(\underline{y}^N \mid \underline{z}^N, \underline{X}^{N+1}) \approx N(\underline{y}^N; \underline{\hat{y}}^N, [D_N + (\Sigma_y^N)^{-1}]^{-1});\ \underline{\hat{y}}^N = \Sigma_y^N[\underline{z}^N - \underline{\sigma}(\underline{\hat{y}}^N)]$$

$$so,\ p(y(\underline{x}^{N+1}) \mid \underline{z}^N, \underline{X}^{N+1}) \approx N(y(\underline{x}^{N+1}); \hat{y}(\underline{x}^{N+1}), \Sigma_{\hat{\underline{y}}})$$

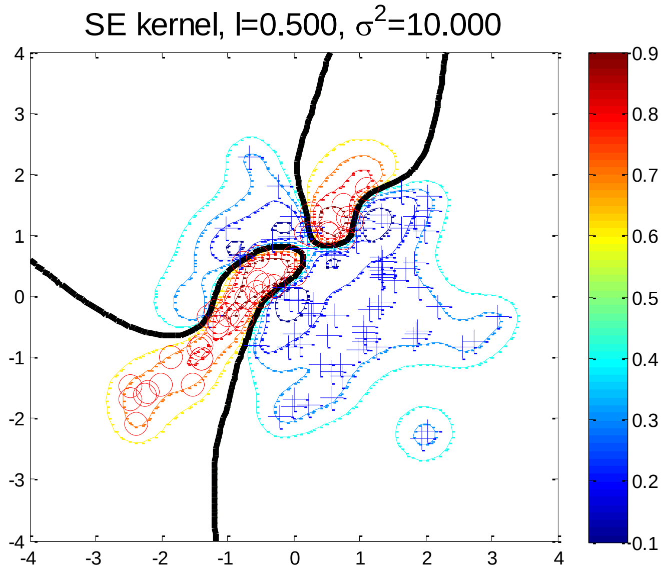
$$\hat{y}(\underline{x}^{N+1}) = \underline{k}^T(\Sigma_y^N)^{-1}\Sigma_y^N(\underline{z}^N - \underline{\sigma}(\underline{\hat{y}}^N)) = \underline{k}^T(\underline{z}^N - \underline{\sigma}(\underline{\hat{y}}^N))$$

$$\Sigma_{\hat{y}} = \sigma_w^2 \underline{\phi}^T(\underline{x}^{N+1})\underline{\phi}(\underline{x}^{N+1}) + \varepsilon - \underline{k}^T[D_N + (\Sigma_y^N)^{-1}]^{-1}\underline{k}$$

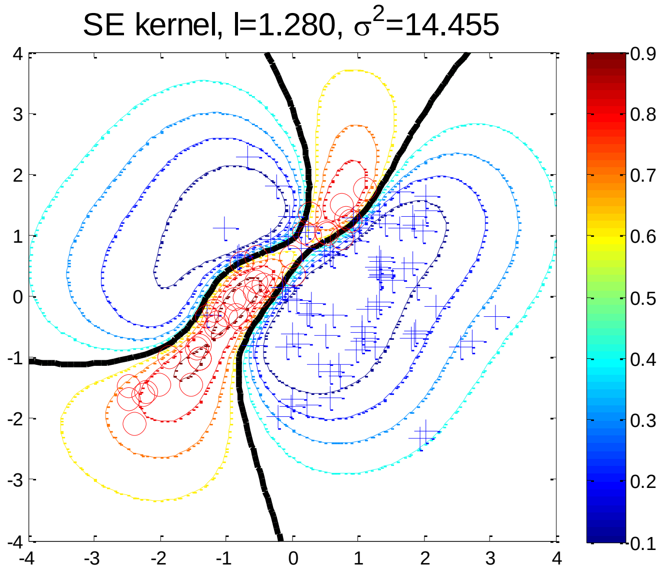
$$Finally,\ p(z^{N+1}=1 \mid \underline{z}^N, \underline{X}^{N+1}) = \int_a \sigma(a) N(a \mid \hat{y}(\underline{x}^{N+1}), \Sigma_{\hat{\underline{y}}}) da = \Phi\left(\frac{\hat{y}(\underline{x}^{N+1})}{\sqrt{1+\pi\Sigma_{\hat{y}}/8}}\right)$$

$$Approximate\ \sigma(a) \approx \Phi\left(\sqrt{\frac{\pi}{8}}a\right)\ by\ requring\ \text{log}istic\ and\ probit\ to\ have\ same\ slope\ at\ a=0.$$

# GP for Classification




$$(l, \sigma_f, \sigma_y) = (0.3, 1, 3.16)$$




$$(l, \sigma_f, \sigma_y) = (1.280, 1, 3.8)$$

SE: Squared Exponential or Exponentiated Quadratic

# Relevance Vector Machine

**Classification : Remember continuous output goes through a logistic**

$$p(z\,|\,y)=\sigma(y)^{z}[1-\sigma(y)]^{1-z};\ \sigma(y)=[1+\exp(-y)]^{-1}=\hat{z};\ y(\underline{x})=\underline{w}^{T}\underline{\phi}(\underline{x})$$

$$\ln p(\underline{w}\,|\,\underline{z}^{N},\underline{X}^{N})=\ln p(\underline{z}^{N}\,|\,\underline{w},\underline{X}^{N})+\ln p(\underline{w})+const.$$

$$\nabla_{\underline{w}}\left[z^{n}\ln\sigma(y^{n})+(1-z^{n})\ln(1-\sigma(y^{n}))\right]$$
$$=\left(z^{n}\left[1-\sigma(y^{n})\right]-(1-z^{n})\sigma(y^{n})\right)\underline{\phi}(\underline{x}^{n})$$
$$=\left(z^{n}-\sigma(y^{n})\right)\underline{\phi}(\underline{x}^{n})$$

$$p(\underline{w})=N(\underline{0},\Sigma_{w});\Sigma_{w}=Diag(\sigma_{w_i}^{2})\ldots need\ to\ be\ learned$$

$$p(\underline{z}^{N}\,|\,\underline{w},\underline{X}^{N})=\prod_{n=1}^{N}\sigma(y^{n})^{z^{n}}[1-\sigma(y^{n})]^{1-z^{n}}=\prod_{n=1}^{N}[\hat{z}^{n}]^{z^{n}}[1-\hat{z}^{n}]^{1-z^{n}}$$

$$\ln p(\underline{w}\,|\,\underline{z}^{N},\underline{X}^{N})=\sum_{n=1}^{N}z^{n}\underbrace{\ln\sigma(y^{n})+(1-z^{n})\ln(1-\sigma(y^{n}))}_{negative\ cross\ entropy}-\frac{1}{2}\underline{w}^{T}\Sigma_{w}^{-1}\underline{w}+const.$$

$$\nabla_{\underline{w}}\ln p(\underline{w}\,|\,\underline{z}^{N},\underline{X}^{N})=\Phi^{T}(\underline{z}^{N}-\underline{\hat{z}}^{N})-\Sigma_{w}^{-1}\underline{w}\Rightarrow\underline{w}^{*}=\Sigma_{w}\Phi^{T}(\underline{z}^{N}-\underline{\hat{z}}^{*N})$$

$$\nabla_{\underline{w}}^{2}\ln p(\underline{w}\,|\,\underline{z}^{N},\underline{X}^{N})=-(\Phi^{T}D_{N}\Phi+\Sigma_{w}^{-1})\Rightarrow\Sigma_{w}^{*}=(\Phi^{T}D_{N}^{*}\Phi+\Sigma_{w}^{-1})^{-1};D_{N}=diag[\hat{z}^{n}(1-\hat{z}^{n})]$$

# Learning RVM Classifiers

$$\nabla_{\underline{w}} \ln p(\underline{w} \mid \underline{z}^N, \underline{X}^N) = \Phi^T(\underline{z}^N - \hat{\underline{z}}^N) - \Sigma_w^{-1}\underline{w} \Rightarrow \underline{w}^* = \Sigma_w \Phi^T(\underline{z}^N - \hat{\underline{z}}^{*N})$$

$$\nabla_{\underline{w}}^2 \ln p(\underline{w} \mid \underline{z}^N, \underline{X}^N) = -(\Phi^T D_N \Phi + \Sigma_w^{-1}) \Rightarrow \Sigma_w^* = (\Phi^T D_N^* \Phi + \Sigma_w^{-1})^{-1}; D_N = diag[\hat{z}^n(1-\hat{z}^n)]$$

$EM\ or\ IRLS - like\ Algorithm\ between\ steps\ 1\ and\ 2:$

1. **For given** $\underline{w}$ **and** $\Sigma_w$**, obtain** $\underline{w}^*$ *and* $\Sigma_w^*$

2. $p(\underline{z}^N \mid \underline{X}^N) = \int_{\underline{w}} p(\underline{z}^N \mid \underline{w}, \underline{X}^N) p(\underline{w} \mid \underline{X}^N) d\underline{w} \approx (2\pi)^{M/2} |\Sigma_w^*|^{1/2}\ p(\underline{z}^N \mid \underline{w}^*, \underline{X}^N) p(\underline{w}^*) \ldots Laplace$

$$\frac{\partial \ln p(\underline{z}^N \mid \underline{X}^N)}{\partial \sigma_{w_i}} = \frac{\partial}{\partial \sigma_{w_i}}\left[\frac{1}{2}\ln|\Sigma_w^*| - \frac{w_i^{*2}}{2\sigma_{w_i}^2} - \ln \sigma_{w_i}\right] = 0$$

$$\Rightarrow \frac{1}{\sigma_{w_i}^3}[(\Phi^T D_N^* \Phi + \Sigma_w^{-1})]^{-1}_{ii} + \frac{w_i^{*2}}{\sigma_{w_i}^3} - \frac{1}{\sigma_{w_i}} = 0 \Rightarrow \sigma_{w_i}^2 = w_i^{*2} + [\Sigma_w^*]_{ii}$$

Logistic regression with $L_2$, $L_1$ criteria, RVM and SVM have similar performance

# RVM for Regression

$$\underline{w}^{n+1} = \underline{w}^{n} = \underline{w}; \underline{w} \sim N(\underline{0}, Diag(\sigma_{w_i}^2))$$
$$z^n = \underline{\phi}^T(\underline{x}^n)\underline{w}^n + v^n; v^n \sim N(0, \beta^{-1})$$

**Estimate** : $y(\underline{x}) = \underline{w}^T \underline{\phi}(\underline{x})$

**Likelihood** : $p(z \mid \underline{x}, \underline{w}, \beta) = N(z; y(\underline{x}), \beta^{-1})$

**Prior** : $p(\underline{w} \mid \Sigma_w) = N(\underline{0}, \Sigma_w); \Sigma_w = Diag(\sigma_{w_i}^2)$

Like estimation with unknown noise variances

$$p(\underline{w} \mid \underline{z}^N, \underline{X}^N, \beta, \Sigma_w) \approx N(\underline{w}, \underline{w}_{MAP}, H^{-1}); \underline{w}_{MAP} = \beta H^{-1} \Phi^T \underline{z}^N; H = \left(\Sigma_w\right)^{-1} + \beta\, \Phi^T \Phi; \Phi = \begin{bmatrix} \underline{\phi}^T(\underline{x}^1) \\ \underline{\phi}^T(\underline{x}^2) \\ \cdot \\ \underline{\phi}^T(\underline{x}^N) \end{bmatrix}; N \text{ by } M \text{ vector}$$

$$p(\underline{z}^N \mid \underline{X}^N, \beta, \Sigma_w) = \int_{\underline{w}} p(\underline{z}^N \mid \underline{X}^N, \beta, \underline{w}) p(\underline{w} \mid \Sigma_w) d\underline{w} = N(\underline{z}^N; \underline{0}, \Sigma_z^N); \Sigma_z^N = \beta^{-1} I_N + \Phi \Sigma_w \Phi^T$$

$$NLL = -\ln p(\underline{z}^N \mid \underline{X}^N, \beta, \Sigma_w) = \frac{1}{2}\left\{N \ln 2\pi + \ln |\Sigma_z^N| + \left(\underline{z}^N\right)^T \left(\Sigma_z^N\right)^{-1} \underline{z}^N\right\}$$
$$= \frac{1}{2}\left\{N \ln 2\pi - N \ln \beta + \beta\left(\underline{z}^N - \Phi \underline{w}_{MAP}\right)^T (\underline{z}^N - \Phi \underline{w}_{MAP}) + \ln |H| + \underline{w}_{MAP}^T \Sigma_w^{-1} \underline{w}_{MAP}\right\}$$

$$\underline{z}^N = \Phi \underline{w} + v^n; v^n \sim N(0, \beta^{-1})$$

$$\left(\sigma_{w_i}^2\right)^{new} = \frac{w_{MAP,i}^2 \sigma_{w_i}^2}{\sigma_{w_i}^2 - \left(H^{-1}\right)_{ii}} \Rightarrow$$ **feature that does not reduce posterior variance will be removed.**

$$\left(\beta^{new}\right)^{-1} = \frac{\left(\underline{z}^N - \Phi \underline{w}_{MAP}\right)^T \left(\underline{z}^N - \Phi \underline{w}_{MAP}\right) \sigma_{w_i}^2}{(N - M)\sigma_{w_i}^2 + \sum_{n=1}^{N} \left(H^{-1}\right)_{ii}}$$

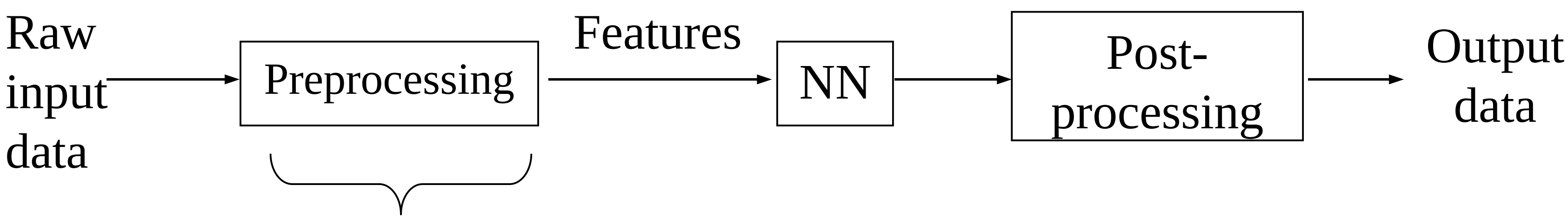
See Bishop's book, pp. 168-170 and 345-353

# Preprocessing and Feature Extraction

- Preprocessing and Feature Extraction

  "Good measurements are half the success" $\Rightarrow$ Finding Good pattern Features to make classification/Regression problem easier.

- Features: Transformation from real world objects to a set of ordinal or cardinal attributes




- Input normalization (Lecture 1)
- Feature extraction and selection
- Feature learning (e.g., PCA)

# Forward and Backward Selection

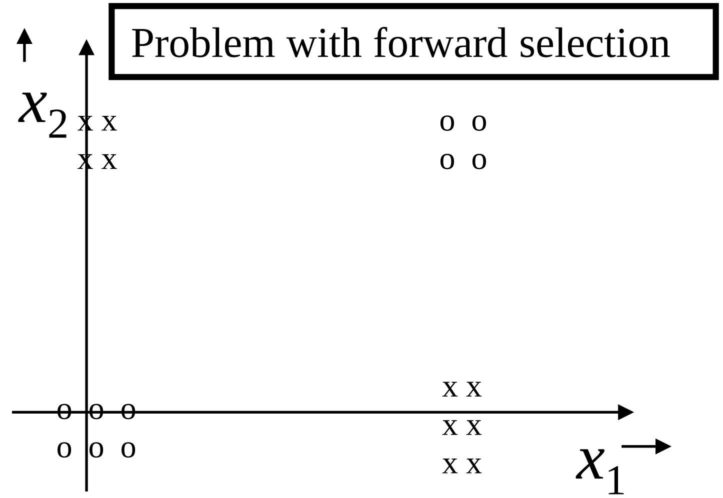
{1} {2} {3} {4}

{1, 2} {3, 2} {4, 2}

{1, 3, 2} {4, 3, 2}

{4, 1, 3, 2}

$$\boxed{\frac{p(p+1)}{2}\ evaluations}$$




Individually, $x_1$ and $x_2$ do not provide discrimination but together they do.

Another strategy is *sequential backward elimination*. In the strategy, one feature is deleted from the set, chosen from among the candidates as the one which gives the *smallest* decrement $\Delta S_k^{\ 2}$. This overcomes the problem of sequential forward selection methods, but at a higher computational cost.

# Orthogonal Matching Pursuit (OMP)

OMP: Greedy Forward Selection

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

$$X=\begin{bmatrix}(\underline{x}^1)^T\\(\underline{x}^2)^T\\ \cdot\\(\underline{x}^N)^T\end{bmatrix}=\begin{bmatrix}\underline{\tilde{x}}_1 & \underline{\tilde{x}}_2 & \cdot & \underline{\tilde{x}}_p\end{bmatrix};Nxp;\underline{z}=\begin{bmatrix}z^1\\z^2\\ \cdot\\z^N\end{bmatrix};N\text{x}1$$

Start with set of features, $S_o=\varnothing$. At iteration $k$, select a feature $j$ such that

$$j_k=\arg\min_j\min_{\underline{w}}\|X_{S_{k-1}\cup\{j\}}\underline{w}-\underline{z}\|^2=\arg\max_j\frac{|\underline{\tilde{x}}_j^T\underline{r}_{k-1}|^2}{\underline{\tilde{x}}_j^T\underline{\tilde{x}}_j};\text{residual }\underline{r}_{k-1}=\underline{z}-X_{S_{k-1}}\underline{w}_{k-1}$$

$$\underline{w}_k=\arg\min_{\underline{w}}\|X_{S_k}\underline{w}-\underline{z}\|^2$$

- Implement using SVD
- $L_1$ regression (LASSO, compressed sensing) is an alternate method for feature selection
- Adaboost is a form of forward selection procedure

$$L_M(\underline{\alpha})=\sum_{n=1}^{N}\exp\{-z^n\sum_{m=1}^{M}\alpha_m g_m(\underline{x}^n)\};g_m(\underline{x}^n)=discri\min ant\ (MLP,SVM,LDA,QDA,Logistic,..)$$

# Alternative Feature Selection Methods

- Alternate criterion: Mutual Information
  - Select feature (or feature subsets) with the maximum mutual information

$$k = \arg \max_{x_i} \left( H(\underline{z}) - H(\underline{z} \mid x_i) \right)$$

  - Applicable to both classification and regression
- Optimization: Select features via
  - Local search
  - Tabu search
  - Genetic Algorithms
  - JMI and CMMI

$T_k$ = Current Test set used on the path leading to node $k$

Joint Mutual Information (JMI) Criterion: (works well)

$$j_k = \arg \max_j \sum_{l \in T_k} I(\{t_l, t_j\}; z) = \arg \max_j \sum_{l \in T_k} [I(t_l; z) + I(t_j; z \mid t_l)]$$

$$= \arg \max_j \sum_{l \in T_k} I(t_j; z \mid t_l)$$

Conditional Mutual Information Maximization (CMIM) Criterion:

$$j_k = \arg \max_j \min_{l \in T_k} \left[ I(t_j; z \mid t_l) \right]$$

- References


  - Guyon, I., and A. Elisseff,"An Introduction to Variable and Feature Selection," *JMLR*, Vol. 3, 2003, pp. 1157-1182.
  - Brown, J., A. Pocock, M-J. Zhao and M. Lujan, "Conditional Likelihood Maximization: A Unifying Framework for Information Theoretic Feature Selection," *JMLR*, Vol. 13, 2012, pp. 27-66.

# More on Information-theoretic Feature Selection

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

$\underline{\theta} \sim$ binary vector, $\theta_i = 1$ if feature $i$ is selected

$\underline{x}_\theta$ ~ selected features; $\underline{x}_{\tilde{\theta}}$ ~ not selected features

$p(z \mid \underline{x}) = p(z \mid \underline{x}_{\theta^*}); \underline{x}_{\theta^*}$ ~ optimal set of features

$q(z \mid \underline{x}_\theta, \underline{w})$ approximates $p(z \mid \underline{x}_\theta)$

$$J(\underline{\theta}, \underline{w}) = -\frac{1}{N} \ln L(\underline{\theta}, \underline{w} \mid D) = -\frac{1}{N} \left( \sum_{n=1}^{N} \ln [q(z^n \mid \underline{x}_\theta^n, \underline{w})] \right)$$

$$= -\frac{1}{N} \left( \begin{array}{c} \sum_{n=1}^{N} \ln [q(z^n \mid \underline{x}_\theta^n, \underline{w})] - \ln[p(z^n \mid \underline{x}_\theta^n) + \ln[p(z^n \mid \underline{x}_\theta^n)] \\ - \ln[p(z^n \mid \underline{x}^n)] + \ln[p(z^n \mid \underline{x}^n)] \end{array} \right)$$

$$= \underbrace{E_{\underline{x}z} \left\{ \ln \frac{p(z \mid \underline{x}_\theta)}{q(z \mid \underline{x}_\theta, \underline{w})} \right\}}_{KL(p(z \mid \underline{x}_\theta) \| q(z \mid \underline{x}_\theta, \underline{w}))} + I(\underline{x}_{\tilde{\theta}}; z \mid \underline{x}_\theta) + \underbrace{H(z \mid \underline{x})}_{irreducible}$$

$$= KL(p(z \mid \underline{x}_\theta) \| q(z \mid \underline{x}_\theta, \underline{w})) + I(\underline{x}; z) - I(\underline{x}_\theta; z) + H(z \mid \underline{x})$$

$so$, $\min I(\underline{x}_{\tilde{\theta}}; z \mid \underline{x}_\theta) \Rightarrow \max I(\underline{x}_\theta; z)$

# JMI & CMIM Criteria

$S_k$ = **Current Feature set**

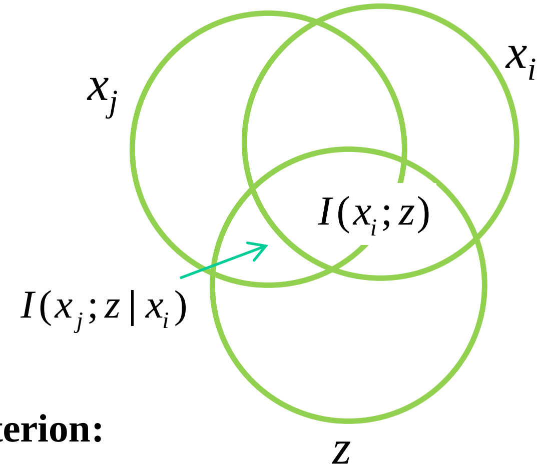
**Joint Mutual Information (JMI) Criterion:**

$$j_{k+1} = \arg\max_j \sum_{i \in S_k} I(\{x_i, x_j\}; z) = \arg\max_j \sum_{i \in S_k} [I(x_i; z) + I(x_j; z \mid x_i)] \Rightarrow \arg\max_j \sum_{i \in S_k} I(x_j; z \mid x_i)$$

$$= \arg\max_j \sum_{i \in S_k} [I(x_j; z) - I(x_j; x_i) + I(x_j; x_i \mid z)]$$

$$= \arg\max_j \left[ |S_k| I(x_j; z) - \sum_{i \in S_k} [I(x_j; x_i) - I(x_j; x_i \mid z)] \right]$$

$$= \arg\max_j \left[ I(x_j; z) - \frac{1}{|S_k|} \sum_{i \in S_k} [I(x_j; x_i) - I(x_j; x_i \mid z)] \right]$$




**Conditional Mutual Information Maximization (CMIM) Criterion:**

$$j_{k+1} = \arg\max_j \min_{i \in S_k} \left[ I(x_j; z \mid x_i) \right]$$

$$= \arg\max_j \min_{i \in S_k} \left[ I(x_j; z) - I(x_j; x_i) + I(x_j; x_i \mid z) \right]$$

$$= \arg\max_j \left\{ I(x_j; z) + \min_{i \in S_k} \left[ I(x_j; x_i \mid z) - I(x_j; x_i) \right] \right\}$$

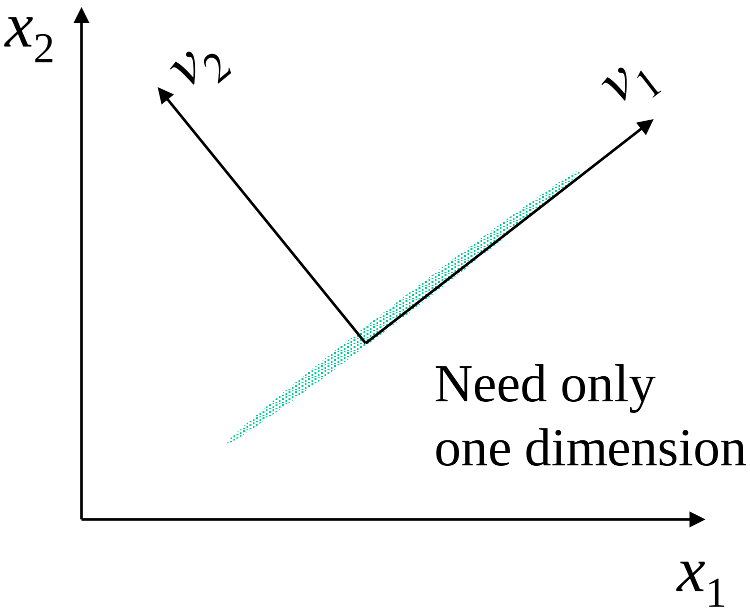
$$= \arg\max_j \left[ I(x_j; z) - \max_{i \in S_k} \left( I(x_j; x_i) - I(x_j; x_i \mid z) \right) \right]$$

# Dimensionality Reduction Methods

- Principal Component Analysis (PCA) ~ SVD

  Consider the input data matrix (mean removed)

$$X = \begin{bmatrix} (\underline{x}_1 - \bar{x})^T \\ (\underline{x}_2 - \bar{x})^T \\ . \\ . \\ . \\ (\underline{x}_N - \bar{x})^T \end{bmatrix} \quad an \;\; N \times P \;\; matrix$$




$$X = U\Sigma V^T \qquad \Sigma \ni \quad \sigma_1 > \sigma_2 > \cdots > \sigma_P$$

$$X^T X = \sum_{i=1}^{N} (\underline{x}_i - \bar{x})(\underline{x}_i - \bar{x})^T = V\Sigma^2 V^T = V\Lambda V^T$$

If $\sigma_1 > \sigma_2 > \cdots > \sigma_r >>> \sigma_{r+1}$, the best *r* feature vectors are

$$\boxed{z_i = \underline{v}_i^T \underline{x}; i = 1, 2, .., r \sim \textit{first } \underline{r \;\; \textit{principal components}} \textit{ define the new features}}$$

# Model Selection in PCA

- Applications of PCA: Pre-processing; Visualization; Data compression
- "Scree" plot: Sum of discarded eigenvalues of $X^T X$. If pick L < $r$ components

$$J(L) = \sum_{k=L+1}^{p} \lambda_k$$

"plot of $J(L)$ looks like the side of a mountain and "scree" refers to the debris fallen at the base"

- Fraction of variance explained

$$F(L) = \frac{\sum_{k=1}^{L} \lambda_k}{\sum_{k=1}^{p} \lambda_k}$$

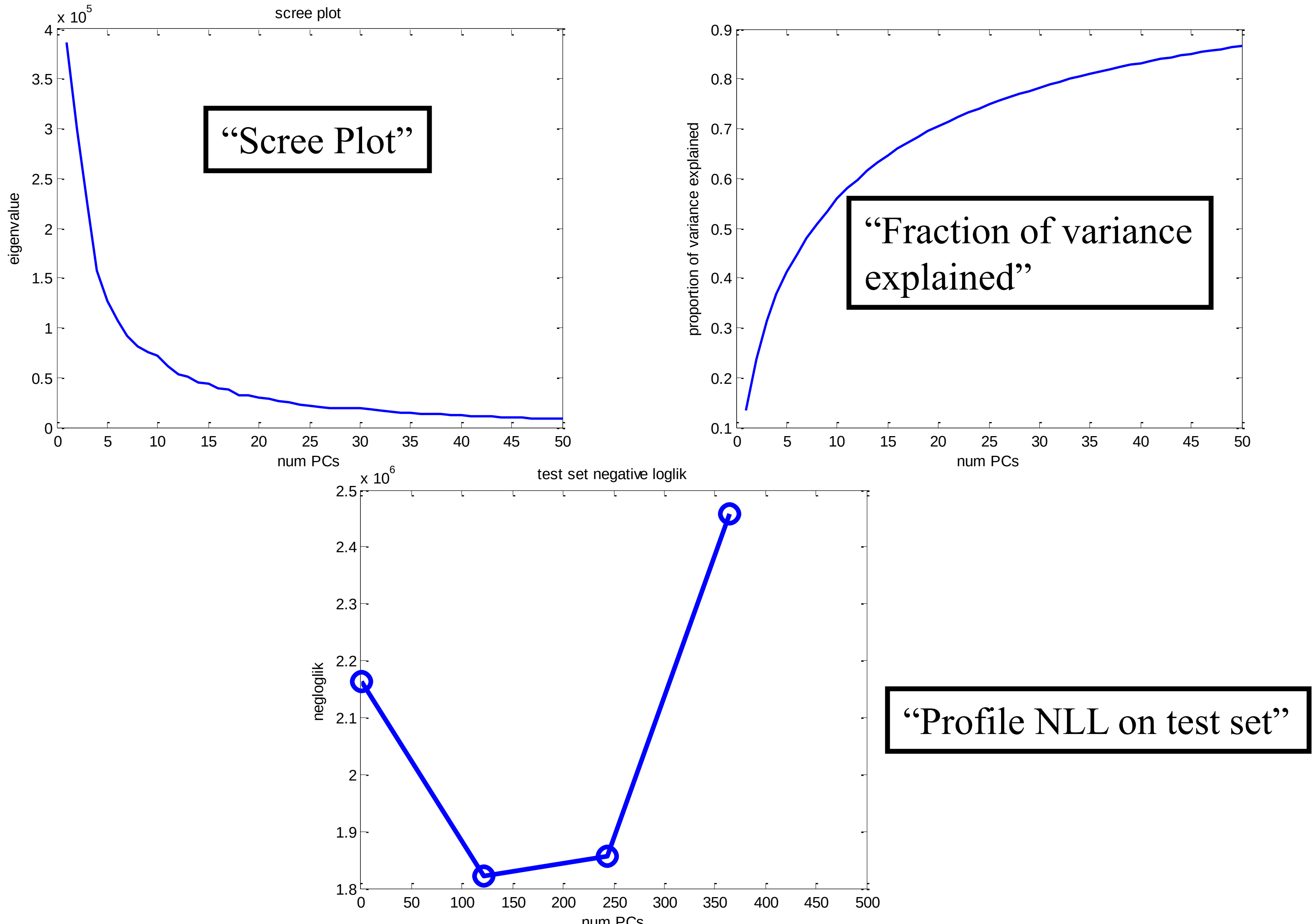
"Ideas are applicable to K-means as well"

- Profile likelihood

$$NLL(L) = \left( \sum_{k=1}^{L} \left( \ln \sigma(L) + \frac{1}{2} \left( \frac{\lambda_k - \mu_1(L)}{\sigma(L)} \right)^2 \right) + \sum_{k=L+1}^{p} \left( \ln \sigma(L) + \frac{1}{2} \left( \frac{\lambda_k - \mu_2(L)}{\sigma(L)} \right)^2 \right) \right)$$

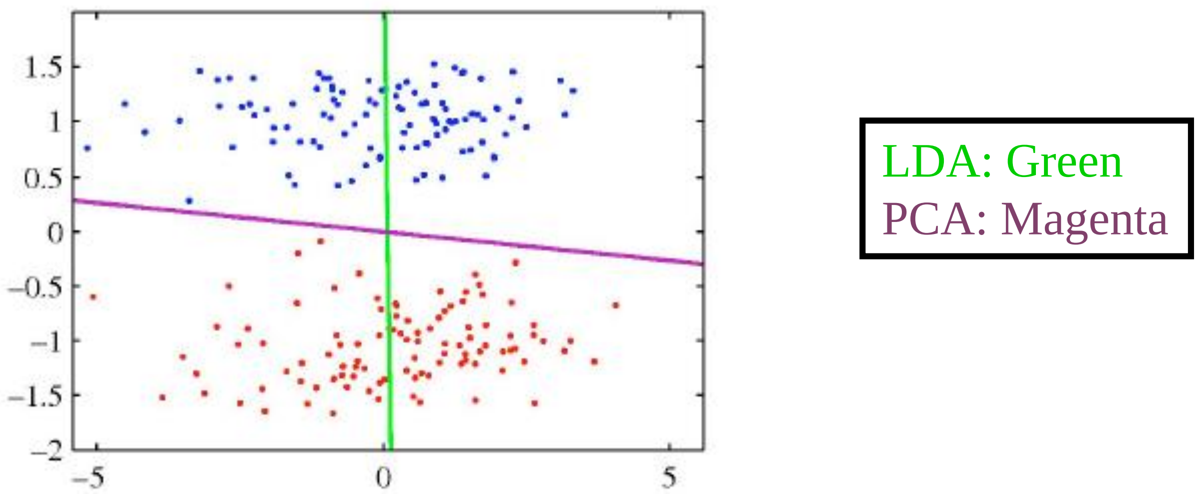
$$\mu_1(L) = \left( \frac{\sum_{k=1}^{L} \lambda_k}{L} \right); \mu_2(L) = \left( \frac{\sum_{k=L+1}^{p} \lambda_k}{p-L} \right); \sigma^2(L) = \frac{\sum_{k=1}^{L} (\lambda_k - \mu_1(L))^2 + \sum_{k=L+1}^{p} (\lambda_k - \mu_2(L))^2}{p}$$

# Illustration of Model Selection Metrics



# PCA versus Fisher's Discriminant

- PCA versus Fisher's Linear Discriminant Analysis
  - Unsupervised (PCA) versus supervised (LDA)



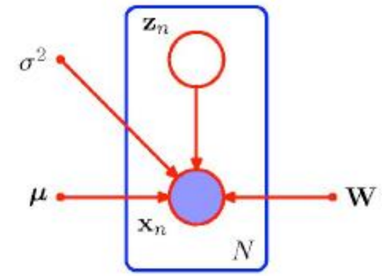

- PCA chooses direction of maximum variance leading to strong class overlap (unsupervised)
- LDA takes into account the class labels (supervised), leading to a projection into the green curve

# Probabilistic PCA - 1

$Latent\ (Hidden)\ Variables: \underline{z} \sim N(\underline{0}, I_z)$

$Measurements: p(\underline{x}\,|\,\underline{z}) = N(W\underline{z} + \underline{\mu}, \sigma^2 I_x); W$ **a** $p$ **x** $r$ **matrix**

$Example: \underline{x} = W\underline{z} + \underline{\mu} + \underline{v}; \underline{v} = N(\underline{0}, \sigma^2 I_x)$




$$\Rightarrow p(\underline{x}) = \int_{\underline{z}} N(W\underline{z} + \underline{\mu}, \sigma^2 I_x) N(\underline{0}, I_z) d\underline{z} = N(\underline{\mu}, WW^T + \sigma^2 I_x) = N(\underline{\mu}, \Sigma_{xx})$$

$$\Rightarrow p(\underline{z}\,|\,\underline{x}) = N[E(\underline{z}\,|\,\underline{x}), \Sigma_z]$$

$$where\ E(\underline{z}\,|\,\underline{x}) = \Sigma_{zx}\Sigma_{xx}^{-1}(\underline{x} - \underline{\mu}) = W^T\Sigma_{xx}^{-1}(\underline{x} - \underline{\mu}) = \frac{M^{-1}W^T}{\sigma^2}(\underline{x} - \underline{\mu})$$

$$\Sigma_z = I_z - \Sigma_{zx}\Sigma_{xx}^{-1}\Sigma_{zx}^T = I_z - W^T\Sigma_{xx}^{-1}W = (I_z + \frac{W^TW}{\sigma^2})^{-1} = M^{-1}$$

$$\boxed{\begin{aligned} & I_z - W^T\Sigma_{xx}^{-1}W \\ & = I_z - W^T(\sigma^2 I_x + WW^T)^{-1}W \\ & = (I_z + \frac{W^TW}{\sigma^2})^{-1} = M^{-1} \end{aligned}}$$

$W^T\Sigma_{xx}^{-1} = \dfrac{M^{-1}W^T}{\sigma^2}$ **... Recall Kalman gain can be written in terms of predicted or updated covariance**

$$\boxed{\begin{aligned} & \begin{bmatrix} \underline{x} \\ \underline{z} \end{bmatrix} \sim N(\begin{bmatrix} \underline{\mu} \\ \underline{0} \end{bmatrix}; \begin{bmatrix} WW^T + \sigma^2 I_x & W \\ W^T & I_z \end{bmatrix}) \\ & K = W^T(\sigma^2 I_x + WW^T)^{-1} \\ & \quad = \frac{W^T}{\sigma^2}\left(I_x + \frac{WW^T}{\sigma^2}\right)^{-1} \\ & \quad = \frac{1}{\sigma^2}(W^T - \frac{W^TW}{\sigma^2}(I_z + \frac{W^TW}{\sigma^2})^{-1}W^T) \\ & \quad = \frac{(I_z + \frac{W^TW}{\sigma^2})^{-1}W^T}{\sigma^2} = \frac{M^{-1}W^T}{\sigma^2} \end{aligned}}$$

$$ML\ PCA: \ln(p(\underline{X}^N\,|\,\underline{\mu}, W, \sigma^2) = -\frac{N}{2}\{p\ln(2\pi) + \ln|\Sigma_{xx}| + Tr(\Sigma_{xx}^{-1}S)\}$$

$$where\ S = \sum_{i=1}^{N}(\underline{x}^i - \underline{\mu})(\underline{x}^i - \underline{\mu})^T = U\Lambda_s U^T$$

Recall Kernel Trick

$Solution: \underline{\mu} = \bar{\underline{x}}; W = U_r(\Lambda_r - \sigma^2 I)^{1/2}R; \sigma^2 = \dfrac{1}{p-r}\displaystyle\sum_{i=r+1}^{p}\lambda_i; R$ **orthogonal**

$Kernel\ PCA:$ Re $place\ \underline{x}_i\ by\ \phi(\underline{x}_i)$

# Probabilistic PCA - 2

$$J = -\ln(p(\underline{X}^N \mid \underline{\mu}, W, \sigma^2) \quad = \frac{N}{2}\{p\ln(2\pi) + \ln|WW^T + \sigma^2 I_p| + Tr((WW^T + \sigma^2 I_p)^{-1} S)\}$$

$$\nabla_{\underline{\mu}} J = -(WW^T + \sigma^2 I_p)^{-1}\left(\sum_{i=1}^{N}(\underline{x}_i - \underline{\mu})\right) = \underline{0} \Rightarrow \underline{\mu} = \frac{1}{N}\sum_{i=1}^{N}\underline{x}^i$$

$$\nabla_W J = (WW^T + \sigma^2 I_p)^{-1}\left(I_p - S(WW^T + \sigma^2 I_p)^{-1}\right)W = 0$$

$$\Rightarrow W \in N\left(I_p - S(WW^T + \sigma^2 I_p)^{-1}\right) = N\left((WW^T + \sigma^2 I_p) - S\right)$$

$$Let\ S = U\Lambda_s U^T \Rightarrow W \in N\left((WW^T + \sigma^2 I_p) - U\Lambda_s U^T\right)$$

$$\Rightarrow W \in N\left((U^T WW^T U + \sigma^2 I_p) - \Lambda_s\right)$$

$$\Rightarrow W \in N\left(U^T WW^T U - (\Lambda_s - \sigma^2 I_p)\right)$$

$$\Rightarrow W \in N\left((\Lambda_s - \sigma^2 I_p)^{-1/2} U^T WW^T U (\Lambda_s - \sigma^2 I_p)^{-1/2} - I_p\right)$$

*so*, set $p$ x $r$ matrix $W = U_r(\Lambda_{sr} - \sigma^2 I_r)^{1/2} R$; $R$ is $r$ by $r$ orthogonal (e.g., $I_r$).

$U_r$ = first $r$ columns of $U$

# Probabilistic PCA - 3

$$W = U_r(\Lambda_{sr} - \sigma^2 I_r)^{1/2} R$$

$$WW^T + \sigma^2 I = U_r(\Lambda_{sr} - \sigma^2 I_r)U_r^T + \sigma^2 UU^T = U\ Diag\begin{bmatrix}\Lambda_{sr} & 0\\ 0 & \sigma^2 I_{p-r}\end{bmatrix}U^T\ and$$

$$Noting\ that\ |U| = \pm 1,$$

$$\ln|WW^T + \sigma^2 I| = \sum_{i=1}^{r} \ln\lambda_{si} + (p-r)\ln\sigma^2$$

$$Tr\left[(WW^T + \sigma^2 I)^{-1} U \Lambda_s U^T\right] = Tr\left[\begin{bmatrix}\Lambda_{sr} & 0\\ 0 & \sigma^2 I_{p-r}\end{bmatrix}^{-1}\Lambda_s\right] = r + \sum_{i=r+1}^{p}\frac{\lambda_{si}}{\sigma^2}$$

$$\nabla_\sigma J = \frac{2(p-r)}{\sigma} - \sum_{i=r+1}^{p}\frac{2\lambda_{si}}{\sigma^3} = 0 \Rightarrow \sigma^2 = \frac{1}{p-r}\sum_{i=r+1}^{p}\lambda_{si}$$

This model correctly captures the variance of the data along principal axes, and approximates the variance in all remaining directions with a single average value $\sigma^2$.

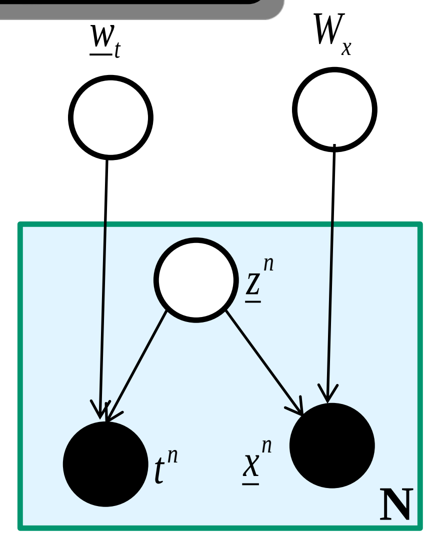
Reminiscent of Array Signal Processing Methods: Direct ML, MUSIC, Minimum Norm, Minimum Variance Distortionless Methods, AR Spatial Spectral Estimation, ……

# Supervised PCA (Bayesian Factor Regression)

*Latent (Hidden) Variables* : $\underline{z} \sim N(\underline{0}, I)$

*Measurements* : $p(\underline{x} \mid \underline{z}) = N(W_x \underline{z} + \underline{\mu}_x, \sigma_x^2 I_p); W_x$ **a** $p$ **x** $r$ **matrix**

$p(t \mid \underline{z}) = N(\underline{w}_t^T \underline{z} + \mu_t, \sigma_t^2); \underline{w}_t$ **a** $r$ **x** $1$ **column vector**




$(\underline{x}, t)$ **is jointly Gaussian with mean** $\begin{bmatrix} \underline{\mu}_x \\ \mu_t \end{bmatrix}$

**and covariance matrix** $\begin{bmatrix} W_x W_x^T + \sigma_x^2 I_p & W_x \underline{w}_t \\ \underline{w}_t^T W_x^T & \underline{w}_t^T \underline{w}_t + \sigma_t^2 \end{bmatrix}$

$$E(t \mid \underline{x}) = \mu_t + \Sigma_{tx}\Sigma_{xx}^{-1}(\underline{x} - \underline{\mu}_x) = \mu_t + \underline{w}_t^T W_x^T \left(W_x W_x^T + \sigma_x^2 I_p\right)^{-1} (\underline{x} - \underline{\mu}_x)$$

$$= \mu_t + \frac{\underline{w}_t^T W_x^T}{\sigma_x^2}\left(\frac{W_x W_x^T}{\sigma_x^2} + I_p\right)^{-1} (\underline{x} - \underline{\mu}) = \mu_t + \underline{w}_t^T \underbrace{\left(\frac{W_x^T W_x}{\sigma_x^2} + I_r\right)^{-1}}_{M^{-1}} \frac{W_x^T}{\sigma_x^2} (\underline{x} - \underline{\mu}_x)$$

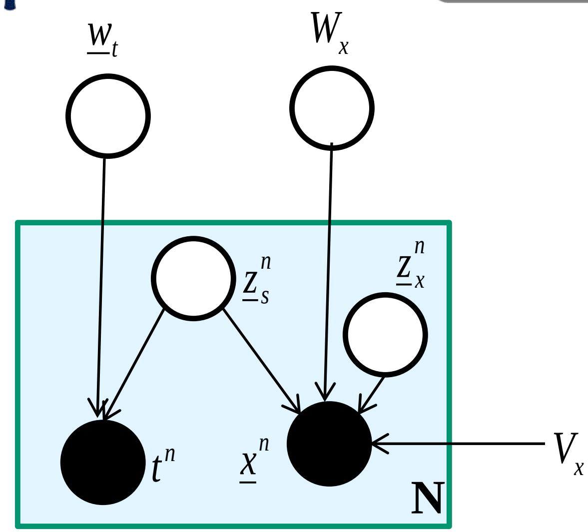
$$\hat{\sigma}_t^2 = \sigma_t^2 + \underline{w}_t^T \underline{w}_t - \Sigma_{tx}\Sigma_{xx}^{-1}\Sigma_{tx}^T = \sigma_t^2 + \underline{w}_t^T \underline{w}_t - \underline{w}_t^T W_x^T \left(W_x W_x^T + \sigma_x^2 I_p\right)^{-1} W_x \underline{w}_t$$

$$= \sigma_t^2 + \underline{w}_t^T \underline{w}_t - \frac{\underline{w}_t^T W_x^T}{\sigma_x^2}\left(\frac{W_x W_x^T}{\sigma_x^2} + I_p\right)^{-1} W_x \underline{w}_t = \sigma_t^2 + \underline{w}_t^T \underbrace{\left(\frac{W_x^T W_x}{\sigma_x^2} + I_r\right)^{-1}}_{M^{-1}} \underline{w}_t$$

Want to estimate $t$ based on $\underline{x}$ and both $\underline{x}$ and $t$ are related to latent(state) variable $\underline{z}$

*Kernel PCA* : Re *place* $\underline{x}$ *by* $\phi(\underline{x})$

# Partial Least Squares



PLS

$$Latent\ (Hidden)\ Variables: \underline{z}_s \sim N(\underline{0}, I_s);\ \ \underline{z}_x \sim N(\underline{0}, I_x)$$

$$Measurements: p(\underline{x}\,|\,\underline{z}) = N(W_x\underline{z}_s + V_x\underline{z}_x + \underline{\mu}_x, \sigma_x^2 I_p)$$

$$p(t\,|\,\underline{z}) = N(\underline{w}_t^T\underline{z}_s + \mu_t, \sigma_t^2)$$

$(\underline{x}, t)$ **is jointly Gaussian with mean** $\begin{bmatrix} \underline{\mu}_x \\ \mu_t \end{bmatrix}$

**and covariance matrix** $\begin{bmatrix} W_xW_x^T + V_xV_x^T + \sigma_x^2 I_p & W_x\underline{w}_t \\ \underline{w}_t^T W_x^T & \underline{w}_t^T\underline{w}_t + \sigma_t^2 \end{bmatrix}$

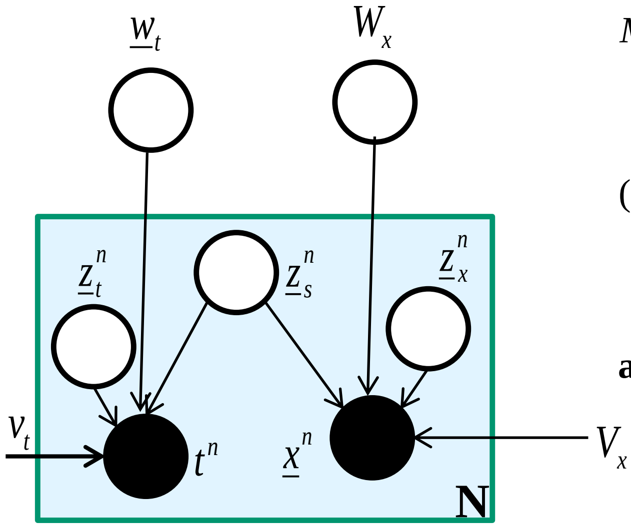
$Kernel\ PLS$ : Re$place\ \underline{x}\ by\ \phi(\underline{x})$

$$E(t\,|\,\underline{x}) = \mu_t + \Sigma_{tx}\Sigma_{xx}^{-1}(\underline{x} - \underline{\mu}_x) = \mu_t + \underline{w}_t^T W_x^T\left(W_xW_x^T + V_xV_x^T + \sigma_x^2 I_p\right)^{-1}(\underline{x} - \underline{\mu}_x)$$

$$\hat{\sigma}_t^2 = \sigma_t^2 - \Sigma_{tx}\Sigma_{xx}^{-1}\Sigma_{tx}^T = \sigma_t^2 + \underline{w}_t^T[1 - W_x^T\left(W_xW_x^T + V_xV_x^T + \sigma_x^2 I_p\right)^{-1}W_x]\underline{w}_t$$

Good when both features and targets are noisy.

# Canonical Correlation Analysis




CCA

$Latent\ (Hidden)\ Variables: \underline{z}_s \sim N(\underline{0}, I_s);\ \underline{z}_x \sim N(\underline{0}, I_x); \underline{z}_t \sim N(\underline{0}, I_t)$

$Measurements: p(\underline{x}\,|\,\underline{z}) = N(W_x \underline{z}_s + V_x \underline{z}_x + \underline{\mu}_x, \sigma_x^2 I_p)$

$$p(t\,|\,\underline{z}) = N(\underline{w}_t^T \underline{z}_s + \underline{v}_t^T \underline{z}_t + \mu_t, \sigma_t^2)$$

$(\underline{x}, t)$ **is jointly Gaussian with mean** $\begin{bmatrix} \underline{\mu}_x \\ \mu_t \end{bmatrix}$

**and covariance matrix** $\begin{bmatrix} \overbrace{W_xW_x^T + V_xV_x^T + \sigma_x^2 I_p}^{C_{xx}} & \overbrace{W_x \underline{w}_t}^{C_{xt}} \\ \underbrace{\underline{w}_t^T W_x^T}_{C_{tx}} & \underbrace{\underline{w}_t^T \underline{w}_t + \underline{v}_t^T \underline{v}_t + \sigma_t^2}_{C_{tt}} \end{bmatrix}$

$Kernel\ CCA: \mathrm{Re}\,place\ \underline{x}\ by\ \phi(\underline{x})$

$$E(t\,|\,\underline{x}) = \mu_t + \Sigma_{tx}\Sigma_{xx}^{-1}(\underline{x} - \underline{\mu}_x) = \mu_t + \underline{w}_t^T W_x^T \left(W_xW_x^T + V_xV_x^T + \sigma_x^2 I_p\right)^{-1}(\underline{x} - \underline{\mu}_x) = \mu_t + C_{tx}C_{xx}^{-1}(\underline{x} - \underline{\mu}_x)$$

$$\hat{\sigma}_t^2 = \underline{v}_t^T \underline{v}_t + \sigma_t^2 \sigma_t^2 - \Sigma_{tx}\Sigma_{xx}^{-1}\Sigma_{tx}^T$$

$$= \sigma_t^2 + \underline{v}_t^T \underline{v}_t + \underline{w}_t^T [1 - W_x^T \left(W_xW_x^T + V_xV_x^T + \sigma_x^2 I_p\right)^{-1} W_x]\underline{w}_t = C_{tt} - C_{tx}C_{xx}^{-1}C_{xt}$$

Canonical correlation analysis (CCA) is a way of measuring the linear relationship between two sets of variables. It finds two bases, one for each variable, that are optimal with respect to correlations and, at the same time, it finds the corresponding correlations.

$$\rho^2 = 1 - \frac{\hat{\sigma}_t^2}{C_{tt}} = \frac{\underline{w}_t^T W_x^T \left(W_xW_x^T + V_xV_x^T + \sigma_x^2 I_p\right)^{-1} W_x \underline{w}_t}{\sigma_t^2 + \underline{v}_t^T \underline{v}_t + \underline{w}_t^T \underline{w}_t} = \frac{C_{tx}C_{xx}^{-1}C_{xt}}{C_{tt}}$$

# Summary

- Radial Basis Functions
- Gaussian Processes
- Relevant Vector Machines
- Feature Selection
- Dimensionality Reduction
- Summary