Statistical Methods in Artificial Intelligence
CSE471 - Monsoon 2016 : Lecture 18

Avinash Sharma

CVIT, IIIT Hyderabad

Lecture Plan

- Revision from Previous Lecture
- Kernel Trick
- Kernel Methods
  - Kernel PCA (KPCA)
  - Kernel LDA (KLDA)
- Data Clustering (Next Class)

Transductive SVM

$$\arg \min_{z_{n+1},\ldots,z_m} \arg \min_{\boldsymbol{a},\xi,\eta,b} \left(\frac{1}{2} \boldsymbol{a}^T \boldsymbol{a} + C \sum_{i=1}^{n} \xi_i + D \sum_{i=n+1}^{m} \eta_i \right)$$

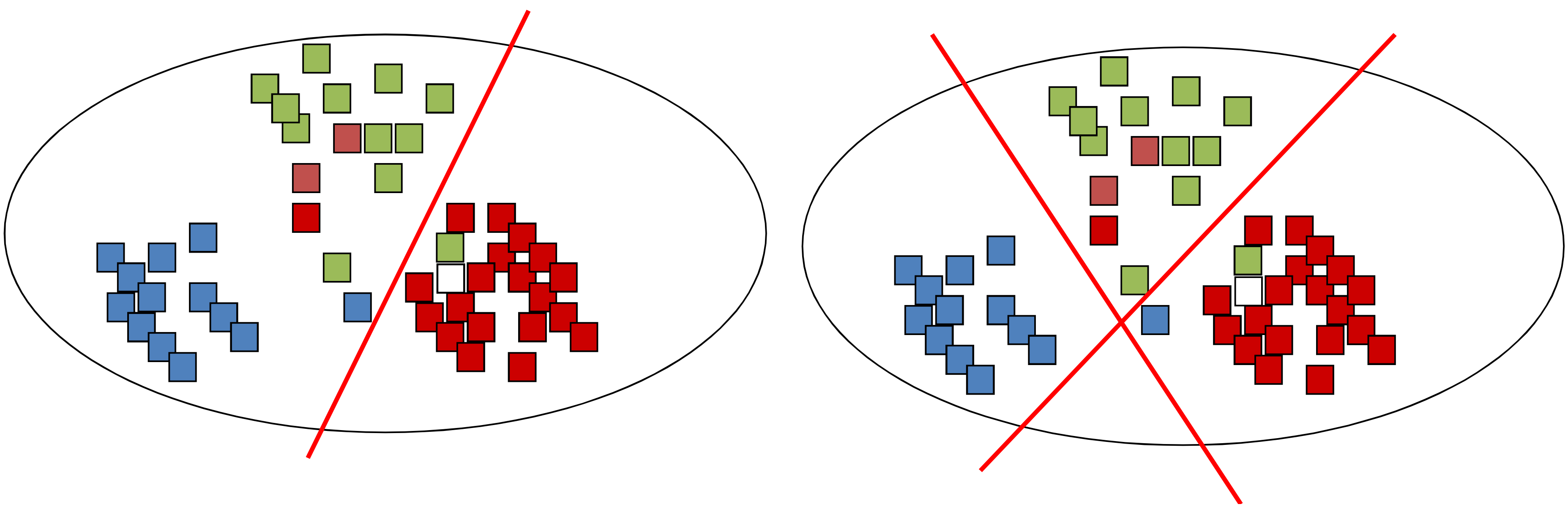
such that $z_i(\boldsymbol{a}^T \boldsymbol{y}_i + b) \geq 1 - \xi_i \;\; \& \;\; \xi_i \geq 0 \quad \forall i \in \{1, \ldots, n\},$

$$z_i(\boldsymbol{a}^T \boldsymbol{y}_i + b) \geq 1 - \eta_i \;\; \& \;\; \eta_i \geq 0 \quad \forall i \in \{n+1, \ldots, m\},$$

- Do Iteratively:
- Step 1: fix $z_{n+1}, \ldots, z_m$, learn weight vector $\boldsymbol{a}$
- Step 2: fix weight vector $\boldsymbol{a}$, try to predict $z_{n+1}, \ldots, z_m$

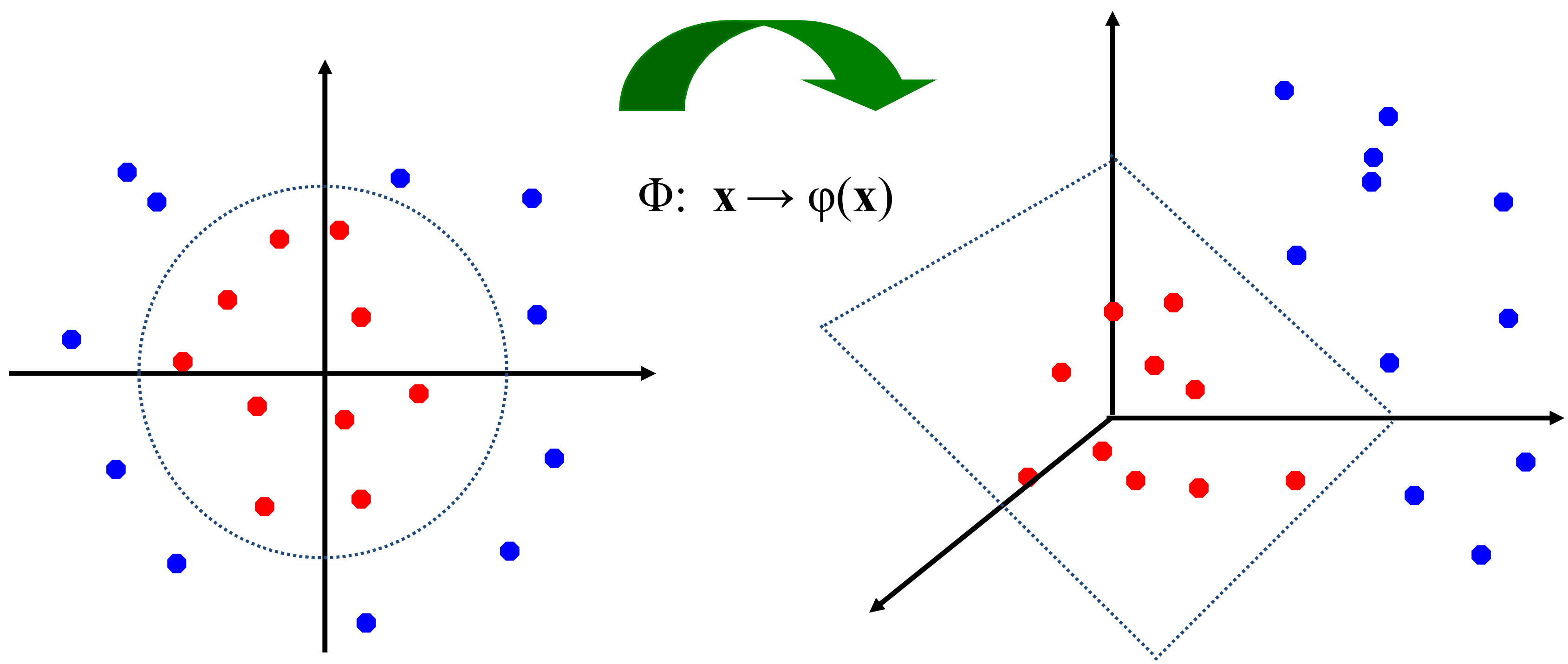
Multi-category SVM

- SVM is a binary classifier.
- Two natural multi-class extensions are:
  - One Class v/s All : Learns C classifiers
  - One Class v/s One Class : Learns C*(C-1) Classifiers



Non-linear SVM

Linear Classification in Non-linear Space



Non-linear SVM

- Non-linear SVM

$$\arg\max_{\alpha_1,\ldots,\alpha_n} \sum_{k=1}^{n} \alpha_k - \frac{1}{2} \sum_{k=1,j=1}^{n} \alpha_k \alpha_j z_k z_j \varphi(\boldsymbol{y}_k)^T \varphi(\boldsymbol{y}_j)$$

$$\arg\max_{\alpha_1,\ldots,\alpha_n} \sum_{k=1}^{n} \alpha_k - \frac{1}{2} \sum_{k=1,j=1}^{n} \alpha_k \alpha_j z_k z_j \boxed{K(k,j)}$$

Kernel Trick

$$f(\boldsymbol{y}) = \sum_{j=1}^{n} \alpha_j z_j \boxed{k(\boldsymbol{y}_j, \boldsymbol{y})} + b$$

Kernel Trick

- Kernels can be defined on general types of data and many classical algorithms can naturally work with general, non-vectorial, data-types !
- Only the inner product (generalization of the dot product in infinite dimensional spaces) matrix. Thus, one can avoid defining an explicit mapping function φ.
- Examples
  - Kernels for Strings: Edit Distance or Number of common substrings
  - Kernel for Documents: BoW, TF-IDF (Term Frequency-Inverse Document Frequency)
  - Kernels for Graphs: Counting Matching Random Walks

Kernelization

- **Kernels** are functions that return inner products between the images of data points in some space.

$$K(k,j) = k(\boldsymbol{y}_k, \boldsymbol{y}_j) = \varphi(\boldsymbol{y}_k)^T \varphi(\boldsymbol{y}_j) = < \varphi(\boldsymbol{y}_k), \varphi(\boldsymbol{y}_j) >$$

- K is $n \times n$ square matrix known as Kernel or Gram matrix.
- K is always a symmetric i.e., $K(k,j) = K(j,k)$
- K is desired to be a positive semi-definite (PSD) matrix (Mercer's Theorem).
- Or, any symmetric & positive semi-definite matrix can be interpreted as kernel matrix.
- Any PSD matrix can have both positive and negative entries. Hence Kernel matrix can have negative entries though mostly positive valued kernel functions are used as similarity functions.

Kernelization

- Commonly used Kernel functions are:
  - Linear Kernel
$$K(k,j) = \boldsymbol{y}_k{}^T \boldsymbol{y}_j$$
  - Polynomial Kernel
$$K(k,j) = (1 + \boldsymbol{y}_k{}^T \boldsymbol{y}_j)^p$$
  - Gaussian /Radial Basis Function (RBF) Kernel
$$K(k,j) = \exp\left(-\frac{\left\|\boldsymbol{y}_k - \boldsymbol{y}_j\right\|^2}{2\sigma^2}\right)$$
  - Sigmoid Kernel (non PSD)
$$K(k,j) = \tanh(\beta_0 \boldsymbol{y}_k{}^T \boldsymbol{y}_j + \beta_1)$$

Properties of Kernels

Given valid kernels $k_1(\mathbf{x}, \mathbf{x}')$ and $k_2(\mathbf{x}, \mathbf{x}')$, the following new kernels will also be valid:

$$\begin{aligned}
k(\mathbf{x}, \mathbf{x}') &= ck_1(\mathbf{x}, \mathbf{x}') & (6.13)\\
k(\mathbf{x}, \mathbf{x}') &= f(\mathbf{x})k_1(\mathbf{x}, \mathbf{x}')f(\mathbf{x}') & (6.14)\\
k(\mathbf{x}, \mathbf{x}') &= q\left(k_1(\mathbf{x}, \mathbf{x}')\right) & (6.15)\\
k(\mathbf{x}, \mathbf{x}') &= \exp\left(k_1(\mathbf{x}, \mathbf{x}')\right) & (6.16)\\
k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}') + k_2(\mathbf{x}, \mathbf{x}') & (6.17)\\
k(\mathbf{x}, \mathbf{x}') &= k_1(\mathbf{x}, \mathbf{x}')k_2(\mathbf{x}, \mathbf{x}') & (6.18)\\
k(\mathbf{x}, \mathbf{x}') &= k_3\left(\boldsymbol{\phi}(\mathbf{x}), \boldsymbol{\phi}(\mathbf{x}')\right) & (6.19)\\
k(\mathbf{x}, \mathbf{x}') &= \mathbf{x}^{\mathrm{T}}\mathbf{A}\mathbf{x}' & (6.20)\\
k(\mathbf{x}, \mathbf{x}') &= k_a(\mathbf{x}_a, \mathbf{x}'_a) + k_b(\mathbf{x}_b, \mathbf{x}'_b) & (6.21)\\
k(\mathbf{x}, \mathbf{x}') &= k_a(\mathbf{x}_a, \mathbf{x}'_a)k_b(\mathbf{x}_b, \mathbf{x}'_b) & (6.22)
\end{aligned}$$

where $c > 0$ is a constant, $f(\cdot)$ is any function, $q(\cdot)$ is a polynomial with nonnegative coefficients, $\boldsymbol{\phi}(\mathbf{x})$ is a function from $\mathbf{x}$ to $\mathbb{R}^M$, $k_3(\cdot, \cdot)$ is a valid kernel in $\mathbb{R}^M$, $\mathbf{A}$ is a symmetric positive semidefinite matrix, $\mathbf{x}_a$ and $\mathbf{x}_b$ are variables (not necessarily disjoint) with $\mathbf{x} = (\mathbf{x}_a, \mathbf{x}_b)$, and k_a and k_b are valid kernel functions over their respective spaces.

Kernelized KNN

- KNN:

$$\left\|\mathrm{x}_i - \mathrm{x}_j\right\|^2 = \langle \mathrm{x}_i, \mathrm{x}_i \rangle + \langle \mathrm{x}_j, \mathrm{x}_j \rangle - 2\langle \mathrm{x}_i, \mathrm{x}_j \rangle$$

- Kernel KNN :

$$\begin{aligned}\left\|\varphi(\mathrm{x}_i) - \varphi(\mathrm{x}_j)\right\|^2 &= \langle \varphi(\mathrm{x}_i), \varphi(\mathrm{x}_i) \rangle + \langle \varphi(\mathrm{x}_j), \varphi(\mathrm{x}_j) \rangle - 2\langle \varphi(\mathrm{x}_i), \varphi(\mathrm{x}_j) \rangle \\ &= K(i,i) + K(j,j) - 2K(i,j)\end{aligned}$$

Principal Component Analysis (PCA)

- k -dimensional representation: Let $\mathbf{x} = \mathbf{m} + \sum_{i=1}^{k} a_i \mathbf{e}_i$

$$\mathbf{v}_1, \ldots, \mathbf{v}_k = arg \max_{\mathbf{e}_1, \ldots, \mathbf{e}_k} J_k = \sum_{i=1}^{n} \left\| \left(\mathbf{m} + \sum_{j=1}^{k} a_j \mathbf{e}_j \right) - \mathbf{x}_i \right\|^2, \qquad \text{for } k \ll d$$

where, $\mathbf{S} = \sum_{i=1}^{n} (\mathbf{x}_i - \mathbf{m})(\mathbf{x}_i - \mathbf{m})^{\boldsymbol{T}} = \sum_{i=1}^{n} \widetilde{\mathbf{x}}_i \, \widetilde{\mathbf{x}}_i^{\boldsymbol{T}}$

$$\mathbf{S}\mathbf{v}_i = \lambda_i \mathbf{v}_i , \qquad \mathbf{v}_i \perp \mathbf{v}_j , \|\mathbf{v}_i\| = 1 \; \forall \; i, j \; \in \{1, .., k\}$$

$$\sum_{i=1}^{n} \widetilde{\mathbf{x}}_i \, \widetilde{\mathbf{x}}_i^{\boldsymbol{T}} \mathbf{v}_j = \lambda_j \mathbf{v}_j \Rightarrow \boxed{\mathbf{v}_j = \frac{1}{\lambda_j} \sum_{i=1}^{n} \widetilde{\mathbf{x}}_i \, \widetilde{\mathbf{x}}_i^{\boldsymbol{T}} \mathbf{v}_j = \sum_{i=1}^{n} \alpha_i \widetilde{\mathbf{x}}_i}$$

Kernel PCA

- Let $\boldsymbol{y}_i = \varphi(\mathbf{x}_i)$ be the centered non-linear projection (mapping) of the data such that $\sum_{i=1}^{n} \varphi(\mathbf{x}_i) = 0$.
- Then $C = \sum_{i=1}^{n} \varphi(\mathbf{x}_i)\varphi(\mathbf{x}_i)^T$ will be the scatter matrix of the *centered mapping*.
- Let $\mathbf{w}_i$ be the eigenvector of the C matrix:

  $C\mathbf{w} = \lambda\mathbf{w}$ and $\mathbf{w} = \sum_{k=1}^{n} \alpha_k \, \varphi(\mathbf{x}_k)$
- Combining these equations:

$$\sum_{i=1}^{n} \varphi(\mathbf{x}_i)\varphi(\mathbf{x}_i)^T \sum_{k=1}^{n} \alpha_k \, \varphi(\mathbf{x}_k) = \lambda \sum_{k=1}^{n} \alpha_k \, \varphi(\mathbf{x}_k)$$

Kernel PCA

$$\sum_{k=1}^{n}\sum_{i=1}^{n}\varphi(\mathbf{x}_i)\varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_k)\alpha_k = \lambda\sum_{k=1}^{n}\alpha_k\,\varphi(\mathbf{x}_k)$$

$$\sum_{k=1}^{n}\sum_{i=1}^{n}\varphi(\mathbf{x}_l)^T\varphi(\mathbf{x}_i)\varphi(\mathbf{x}_i)^T\varphi(\mathbf{x}_k)\alpha_k = \lambda\sum_{k=1}^{n}\alpha_k\,\varphi(\mathbf{x}_l)^T\varphi(\mathbf{x}_k)\quad \forall l = 1{:}\,n$$

$$K^2\boldsymbol{\alpha} = \lambda K\boldsymbol{\alpha} \;\Rightarrow K\boldsymbol{\alpha} = \lambda\boldsymbol{\alpha}$$

$$\|\mathbf{w}\| = \mathbf{w}^{\boldsymbol{T}}\mathbf{w} \; = \textstyle\sum_{k=1}^{n}\alpha_k\,\varphi(\mathbf{x}_k)^{\boldsymbol{T}}\sum_{k=1}^{n}\alpha_k\,\varphi(\mathbf{x}_k) = \boldsymbol{\alpha}^T K\boldsymbol{\alpha} = \lambda$$

$$Let\ Y = \varphi(X)\ and\ \mathbf{w} = \frac{Y\boldsymbol{\alpha}}{\sqrt{\lambda}}\ so\ that\ \mathbf{w}^{\boldsymbol{T}}\mathbf{w} = \mathbf{1}$$

- Though we don't have $Y = \varphi(X)$, we can still compute lower dimensional representation o a test point $\mathbf{x}_*$ as $k_*\mathbf{w}$ where

$$k_* = [k(\mathbf{x}_*, \mathbf{x}_1), \dots, k(\mathbf{x}_*, \mathbf{x}_n)]$$

- For centered mapping:

$$\widetilde{K} = (I - \mathbf{1}\mathbf{1}^T/n)K(I - \mathbf{1}\mathbf{1}^T/n), \qquad \sum_{k=1}^{n}\varphi(\mathbf{x}_k) = 0$$

Kernel PCA

- Compute $n \times n$ **Gram Matrix** K using any kernel function.
- Compute eigen-(values/vectors) or K as $\lambda_j, \boldsymbol{\alpha}^j \quad \forall j = 1\!:\!m$
- Normalize the eigenvectors: $\boldsymbol{\alpha}^j = \boldsymbol{\alpha}^j / \lambda_j$ such that eigenvector of C matrix is: $\mathbf{w}^l = \sum_{k=1}^{n} {\alpha^l}_k \, \varphi(\mathbf{x}_k)$
- Project any data point $\varphi(\mathbf{x})$ onto $\mathbf{w}^l$ as:

$$\varphi(\mathbf{x})^T \mathbf{w}^l = \varphi(\mathbf{x})^T \sum_{k=1}^{n} {\alpha^l}_k \, \varphi(\mathbf{x}_k) = \sum_{k=1}^{n} {\alpha^l}_k \, K(\mathbf{x}_k, \mathbf{x})$$

Scholkopf, Smola, Muller, “Nonlinear component analysis as a kernel eigenvalue problem,” Technical report #44, Max Plank Institute, 1996.

Fisher's LDA

inter-class: $|\tilde{m}_1 - \tilde{m}_2| = |w^{\mathrm{T}}(m_1 - m_2)|$

intra-class: $\tilde{s}_i^2 = \sum_{y \in Y_i} (y - \tilde{m}_i)^2$

want to maximize: $J(w) = \dfrac{|\tilde{m}_1 - \tilde{m}_2|^2}{\tilde{s}_1^2 + \tilde{s}_2^2}$

$$y, \tilde{m}_1, \tilde{m}_2 : \underset{1}{[\;\;]}^{1} \quad (w^{\mathrm{T}}x - w^{\mathrm{T}}m_i) : \underset{1}{[\;\;]}^{1}$$

$$x, w, m_1, m_2 : \underset{1}{[\;\;]}^{\mathrm{D}} \qquad S_{\mathrm{B}}, S_{\mathrm{w}} : \underset{\mathrm{D}}{[\;\;]}^{\mathrm{D}}$$

$$\tilde{s}_i^2 = \sum_{x \in D_i} (w^{\mathrm{T}}x - w^{\mathrm{T}}m_i)(w^{\mathrm{T}}x - w^{\mathrm{T}}m_i)^{\mathrm{T}} = \sum_{x \in D_i} w^{\mathrm{T}}(x - m_i)(x - m_i)^{\mathrm{T}} w = w^{\mathrm{T}} S_i w$$

$$\tilde{s}_1^2 + \tilde{s}_2^2 = w^{\mathrm{T}} S_1 w + w^{\mathrm{T}} S_2 w = w^{\mathrm{T}} S_{\mathrm{w}} w$$

$$|\tilde{m}_1 - \tilde{m}_2|^2 = (w^{\mathrm{T}}m_1 - w^{\mathrm{T}}m_2)^2 = w^{\mathrm{T}}(m_1 - m_2)(m_1 - m_2)^{\mathrm{T}} w = w^{\mathrm{T}} S_{\mathrm{B}} w$$

want to maximize: $J(w) = \dfrac{w^{\mathrm{T}} S_{\mathrm{B}} w}{w^{\mathrm{T}} S_{\mathrm{w}} w}$

$$S_{\mathrm{B}} w = \lambda S_{\mathrm{w}} w$$

Kernel LDA

- Let, $\mathbf{m}_i^{\phi} = \frac{1}{l_i} \sum_{j=1}^{l_i} \phi(\mathbf{x}_j^i)$. $\mathbf{S}_B^{\phi} = (\mathbf{m}_2^{\phi} - \mathbf{m}_1^{\phi})(\mathbf{m}_2^{\phi} - \mathbf{m}_1^{\phi})^{\mathrm{T}}$

$$\mathbf{w} = \sum_{i=1}^{l} \alpha_i \phi(\mathbf{x}_i).$$

$$\mathbf{S}_W^{\phi} = \sum_{i=1,2} \sum_{n=1}^{l_i} (\phi(\mathbf{x}_n^i) - \mathbf{m}_i^{\phi})(\phi(\mathbf{x}_n^i) - \mathbf{m}_i^{\phi})^{\mathrm{T}},$$

- We can write the criterion function as: $J(\mathbf{w}) = \frac{\mathbf{w}^{\mathrm{T}} \mathbf{S}_B^{\phi} \mathbf{w}}{\mathbf{w}^{\mathrm{T}} \mathbf{S}_W^{\phi} \mathbf{w}},$

- This can further be rewritten as: $J(\alpha) = \frac{\alpha^{\mathrm{T}} \mathbf{M} \alpha}{\alpha^{\mathrm{T}} \mathbf{N} \alpha}$,

  where,

$$\mathbf{M} = (\mathbf{M}_2 - \mathbf{M}_1)(\mathbf{M}_2 - \mathbf{M}_1)^{\mathrm{T}} \qquad (\mathbf{M}_i)_j = \frac{1}{l_i} \sum_{k=1}^{l_i} k(\mathbf{x}_j, \mathbf{x}_k^i).$$

$$\mathbf{N} = \sum_{j=1,2} \mathbf{K}_j (\mathbf{I} - \mathbf{1}_{l_j}) \mathbf{K}_j^{\mathrm{T}},$$

Kernel LDA

- After setting analytical derivative of criterion function $J(\alpha)$ to 0:

$$(\alpha^{\mathrm{T}}\mathbf{M}\alpha)\mathbf{N}\alpha = (\alpha^{\mathrm{T}}\mathbf{N}\alpha)\mathbf{M}\alpha.$$

$$\alpha = \mathbf{N}^{-1}(\mathbf{M}_2 - \mathbf{M}_1).$$

$$\mathbf{N}_\epsilon = \mathbf{N} + \epsilon\mathbf{I}.$$

- Given solution vector α, we can project a data point to lower dimensional discriminating space as:

$$y(\mathbf{x}) = (\mathbf{w} \cdot \phi(\mathbf{x})) = \sum_{i=1}^{l} \alpha_i k(\mathbf{x}_i, \mathbf{x}).$$

https://en.wikipedia.org/wiki/Kernel_Fisher_discriminant_analysis

Self Study

- Kernel LDA
- Multiple Kernel Learning
  - Seeking optimal parameters for combining multiple kernels
    - https://en.wikipedia.org/wiki/Multiple_kernel_learning
- Non-linear Dimensionality Reduction
  - Higher dimensional data sampled from lower dimensional manifold
    - https://en.wikipedia.org/wiki/Nonlinear_dimensionality_reduction

Mid Term 2 Syllabus

- What all is covered in the class & tutorial.
- Chapter 2 (Normal Density, DF, Mahalanobis Distance)
  - ❖ 2.1—2.3, 2.5, 2.6, 2.8.3
- Chapter 3 (Parameter Estimation, BPE, MLE, PCA, LDA)
  - ❖ 3.1, 3.2, 3.3, 3.4, 3.5, 3.5.1, 3.7, 3.8
- Chapter 5 (SVM, Kernel SVM, Kernel definition/trick/properties)
  - ❖ 5.11, 5.12,
- Do refer to related public material from books/online resources.