Metric Embedding for Kernel Classification Rules

Bharath K. Sriperumbudur

University of California, San Diego

(Joint work with Omer Lang & Gert Lanckriet)
Parzen window methods are popular in density estimation, kernel regression etc.

We consider these rules for classification.

Set up:

- Binary classification: \( \{(X_i, Y_i)\}_{i=1}^n \sim D, X_i \in \mathbb{R}^D \) and \( Y_i \in \{0, 1\} \). Classify \( x \in \mathbb{R}^D \).

- Kernel classification rule: [Devroye et al., 1996]

\[
g_n(x) = \begin{cases} 
0 & \text{if } \sum_{i=1}^n \mathbb{1}\{Y_i=0\} K\left(\frac{x-X_i}{h}\right) \geq \sum_{i=1}^n \mathbb{1}\{Y_i=1\} K\left(\frac{x-X_i}{h}\right) \\
1 & \text{otherwise},
\end{cases}
\]

(1)

where \( K : \mathbb{R}^D \to \mathbb{R} \) is a smoothing kernel function which is usually non-negative (not necessarily positive definite).
Introduction

- Parzen window methods are popular in density estimation, kernel regression etc.

- We consider these rules for classification.

Set up:

- **Binary classification:** \( \{(X_i, Y_i)\}_{i=1}^n \sim \mathcal{D}, \ X_i \in \mathbb{R}^D \text{ and } Y_i \in \{0, 1\}. \) Classify \( x \in \mathbb{R}^D. \)

- **Kernel classification rule:** [Devroye et al., 1996]

\[
g_n(x) = \begin{cases} 
0 & \text{if } \sum_{i=1}^n 1\{Y_i=0\} K \left( \frac{x-X_i}{h} \right) \geq \sum_{i=1}^n 1\{Y_i=1\} K \left( \frac{x-X_i}{h} \right) \\
1 & \text{otherwise,}
\end{cases}
\]

where \( K : \mathbb{R}^D \to \mathbb{R} \) is a smoothing kernel function which is usually non-negative (not necessarily positive definite).
Kernel Classification Rule

Examples:

- **Gaussian kernel**: \( K(x) = e^{-\|x\|_2^2} \) (p.d.)
- **Cauchy kernel**: \( K(x) = (1 + \|x\|_2^{D+1})^{-1} \) (p.d.)
- **Naïve kernel**: \( K(x) = 1_{\{\|x\|_2 \leq 1\}} \) (not p.d.)
- **Epanechnikov kernel**: \( K(x) = (1 - \|x\|_2^2)1_{\{\|x\|_2 \leq 1\}} \) (not p.d.)

- **Naïve kernel**: performs \( h \)-ball nearest neighbor (NN) classification.
- **\( K \) is p.d.**: Eq. (1) is similar to the RKHS based kernel rule.
- **How to choose \( h \)**: only asymptotic guarantees for universal consistency are available [Devroye and Krzyżak, 1989].
Kernel Classification Rule

Examples:

- **Gaussian kernel:** \( K(x) = e^{-\|x\|^2} \) (p.d.)
- **Cauchy kernel:** \( K(x) = (1 + \|x\|^{D+1})^{-1} \) (p.d.)
- **Naïve kernel:** \( K(x) = 1_{\{\|x\|_2 \leq 1\}} \) (not p.d.)
- **Epanechnikov kernel:** \( K(x) = (1 - \|x\|^2)1_{\{\|x\|_2 \leq 1\}} \) (not p.d.)

- **Naïve kernel:** performs \( h \)-ball nearest neighbor (NN) classification.
- **\( K \) is p.d.:** Eq. (1) is similar to the RKHS based kernel rule.
- **How to choose \( h \):** only asymptotic guarantees for universal consistency are available [Devroye and Krzyżak, 1989].
Metric Learning for $k$-NN

- **Dependence on the metric:** Finite-sample risk of the $k$-NN rule may be reduced by using a weighted Euclidean metric, even though the infinite sample risk is independent of the metric used [Snapp and Venkatesh, 1998].

  - Experimentally verified by:
    - [Xing et al., 2003]
    - NCA [Goldberger et al., 2005]
    - MLCC [Globerson and Roweis, 2006]
    - LMNN [Weinberger et al., 2006]

- All these methods learn $L \in \mathbb{R}^{D \times D}$ so that $x \mapsto Lx$. 
Metric Learning for $k$-NN

- **Dependence on the metric:** Finite-sample risk of the $k$-NN rule may be reduced by using a **weighted Euclidean metric**, even though the infinite sample risk is independent of the metric used [Snapp and Venkatesh, 1998].

- **Experimentally verified by:**
  - [Xing et al., 2003]
  - NCA [Goldberger et al., 2005]
  - MLCC [Globerson and Roweis, 2006]
  - LMNN [Weinberger et al., 2006]

- All these methods learn $L \in \mathbb{R}^{D \times D}$ so that $x \mapsto Lx$. 
Some applications need natural distance measures that reflect the underlying structure of the data.

- Distance between images: **tangent distance**
- Distance between points on a manifold: **geodesic distance**

Usually, Euclidean or weighted Euclidean distance is used as a surrogate.

In the absence of prior knowledge, the data may be used to select the suitable metric.

**Questions we address:** Find

- \( \varphi : (\mathcal{X}, \rho) \rightarrow (\mathcal{Y}, \ell_2) \).
- \( h \)
Some applications need natural distance measures that reflect the underlying structure of the data.

- Distance between images: tangent distance
- Distance between points on a manifold: geodesic distance

Usually, Euclidean or weighted Euclidean distance is used as a surrogate.

In the absence of prior knowledge, the data may be used to select the suitable metric.

Questions we address: Find

\[ \varphi : (X, \rho) \rightarrow (Y, \ell_2). \]

\[ h \]
Problem Formulation

Multi-class classification:

\[ g_n(x) = \arg \max_{l \in [L]} \sum_{i=1}^{n} \mathbb{1}_{Y_i = l} [\rho(x, X_i) \leq h] \]  

(2)

where \([L] := \{1, \ldots, L\}\), \([a] = \mathbb{1}_{\{a\}}\) and \(\rho(x, X_i) = \|\varphi(x) - \varphi(X_i)\|_2\).

Goal: To learn \(\varphi\) and \(h\) by minimizing the probability of error associated with \(g_n\).

\[(\varphi^*, h^*) = \arg \min_{\varphi, h} \Pr_{(X, Y) \in \mathcal{D}}(g_n(X) \neq Y)\]  

(3)

Regularized problem:

\[
\min_{\varphi, h > 0} \sum_{i=1}^{n} [g_n(X_i) \neq Y_i] + \lambda \Omega[\varphi], \quad \lambda > 0
\]  

(4)
Problem Formulation

Multi-class classification:

\[
g_n(x) = \arg \max_{l \in [L]} \sum_{i=1}^{n} [Y_i = l] [\rho(x, X_i) \leq h]
\]  \hspace{1cm} (2)

where \([L] := \{1, \ldots, L\}\), \([a] = 1_{\{a\}}\) and \(\rho(x, X_i) = \|\varphi(x) - \varphi(X_i)\|_2\).

Goal: To learn \(\varphi\) and \(h\) by minimizing the probability of error associated with \(g_n\).

\[(\varphi^*, h^*) = \arg \min_{\varphi, h} \Pr_{(X, Y) \in D}(g_n(X) \neq Y) \]  \hspace{1cm} (3)

Regularized problem:

\[
\min_{\varphi, h > 0} \sum_{i=1}^{n} [g_n(X_i) \neq Y_i] + \lambda \Omega[\varphi], \quad \lambda > 0
\]  \hspace{1cm} (4)
Problem Formulation

Multi-class classification:

\[ g_n(x) = \arg \max_{l \in [L]} \sum_{i=1}^{n} [Y_i = l] \left[ \rho(x, X_i) \leq h \right] \]  \hspace{1cm} (2)

where \([L] := \{1, \ldots, L\}\), \([a] = 1_{\{a\}}\) and \(\rho(x, X_i) = \frac{1}{2} \| \varphi(x) - \varphi(X_i) \|_2\).

Goal: To learn \(\varphi\) and \(h\) by minimizing the probability of error associated with \(g_n\).

\[ (\varphi^*, h^*) = \arg \min_{\varphi, h} \Pr_{(X, Y) \in D}(g_n(X) \neq Y) \]  \hspace{1cm} (3)

Regularized problem:

\[ \min_{\varphi, h > 0} \sum_{i=1}^{n} [g_n(X_i) \neq Y_i] + \lambda \Omega[\varphi], \quad \lambda > 0 \]  \hspace{1cm} (4)
Problem Formulation

Multi-class classification:

\[
g_n(x) = \arg\max_{l \in [L]} \sum_{i=1}^{n} [Y_i = l] \left\{ \rho(x, X_i) \leq h \right\}
\]  

(2)

where \([L] := \{1, \ldots, L\}, \left[a\right] = 1_{\{a\}}\) and \(\rho(x, X_i) = \frac{1}{2} \| \varphi(x) - \varphi(X_i) \|_2\).

Goal: To learn \(\varphi\) and \(h\) by minimizing the probability of error associated with \(g_n\).

\[
(\varphi^*, h^*) = \arg\min_{\varphi, h} \Pr_{(X, Y) \in \mathcal{D}}(g_n(X) \neq Y)
\]  

(3)

Regularized problem:

\[
\min_{\varphi, h > 0} \sum_{i=1}^{n} [g_n(X_i) \neq Y_i] + \lambda \Omega[\varphi], \quad \lambda > 0
\]  

(4)
Problem Formulation

Minimize an upper bound on $\sum_{i=1}^{n} [g_n(X_i) \neq Y_i]$, followed with hinge-relaxation of $[.]$, we get

$$\min_{\varphi, \tilde{h}} \sum_{i=1}^{n} \left[ 2 - n_i^+ + \sum_{j=1}^{n} \left[ 1 + \tau_{ij} \|\varphi(X_i) - \varphi(X_j)\|_2^2 - \tau_{ij} \tilde{h} \right] \right] + \lambda \Omega[\varphi]$$  (5)

where $\tilde{h} = h^2$, $\tau_{ij} = 2\delta_{Y_i, Y_j} - 1$ and $n_i^+ = \sum_{j=1}^{n}[\tau_{ij} = 1]$.

Choice of $\varphi$:

- Suppose $\varphi$ is a Mercer kernel map: $\langle \varphi(x), \varphi(y) \rangle_{\ell_2} = \mathcal{K}(x, y)$.
- $\|\varphi(X_i) - \varphi(X_j)\|_2^2$ is a function of $\mathcal{K}$ alone.
- $\Omega[\varphi]$ is usually chosen as $\text{tr}(K)$, $\|K\|_F^2$ etc.
- This choice does not provide an out-of-sample extension.
Problem Formulation

Minimize an upper bound on $\sum_{i=1}^{n} [g_n(X_i) \neq Y_i]$, followed with hinge-relaxation of $[.]$, we get

$$
\min_{\varphi, \tilde{h}} \sum_{i=1}^{n} \left[ 2 - n_i^+ + \sum_{j=1}^{n} \left[ 1 + \tau_{ij} \|\varphi(X_i) - \varphi(X_j)\|_2^2 - \tau_{ij} \tilde{h} \right]_+ \right]_+ + \lambda \Omega[\varphi]
$$

(5)

where $\tilde{h} = h^2$, $\tau_{ij} = 2 \delta_{Y_i, Y_j} - 1$ and $n_i^+ = \sum_{j=1}^{n} [\tau_{ij} = 1]$.

Choice of $\varphi$:

- Suppose $\varphi$ is a Mercer kernel map: $\langle \varphi(x), \varphi(y) \rangle_{\ell_2^2} = \mathcal{K}(x, y)$.
- $\|\varphi(X_i) - \varphi(X_j)\|_2^2$ is a function of $\mathcal{K}$ alone.
- $\Omega[\varphi]$ is usually chosen as $\text{tr}(K)$, $\|K\|_F^2$ etc.
- This choice does not provide an out-of-sample extension.
Problem Formulation

Minimize an upper bound on \( \sum_{i=1}^{n} [g_n(X_i) \neq Y_i] \), followed with hinge-relaxation of \([.]\), we get

\[
\min_{\varphi, \tilde{h}} \sum_{i=1}^{n} \left[ 2 - n_i^+ + \sum_{j=1}^{n} \left[ 1 + \tau_{ij} \|\varphi(X_i) - \varphi(X_j)\|_2^2 - \tau_{ij} \tilde{h} \right] \right]_+ + \lambda \Omega[\varphi]
\]

where \( \tilde{h} = h^2 \), \( \tau_{ij} = 2\delta_{Y_i, Y_j} - 1 \) and \( n_i^+ = \sum_{j=1}^{n} [\tau_{ij} = 1] \).

Choice of \( \varphi \):

- Suppose \( \varphi \) is a Mercer kernel map: \( \langle \varphi(x), \varphi(y) \rangle_{\ell_2} = \mathcal{K}(x, y) \).
- \( \|\varphi(X_i) - \varphi(X_j)\|_2^2 \) is a function of \( \mathcal{K} \) alone.
- \( \Omega[\varphi] \) is usually chosen as \( \text{tr}(K), \|K\|_F^2 \) etc.
- This choice does not provide an out-of-sample extension.
Problem Formulation
Problem Formulation
\( \varphi \) in an RKHS

Theorem (Multi-output regularization)

**Suppose**

- \( \varphi = (\varphi_1, \ldots, \varphi_d) \), \( \varphi_i : \mathcal{X} \to \mathbb{R} \).
- \( \varphi_i \in (\mathcal{H}_i, \mathcal{K}_i) \).

**Then**

- **Minimizer of Eq. (5) with** \( \Omega[\varphi] = \sum_{i=1}^{d} \| \varphi_i \|_{\mathcal{H}_i}^2 \) **is of the form**

\[
\varphi_j = \sum_{i=1}^{n} c_{ij} \mathcal{K}_j(., \mathcal{X}_i), \quad \forall j \in [d],
\]

where \( c_{ij} \in \mathbb{R} \) and \( \sum_{i=1}^{n} c_{ij} = 0 \), \( \forall i \in [n] \), \( \forall j \in [d] \).
Corollary

Suppose

\( K_1 = \ldots = K_d = K. \)

Then, \( \| \varphi(x) - \varphi(y) \|_2^2 \) is the Mahalanobis distance between the empirical kernel maps at \( x \) and \( y \).

Corollary (Linear kernel)

Let

\( \mathcal{X} = \mathbb{R}^D. \)

\( K(z, w) = \langle z, w \rangle_2 = z^T w. \)

Then \( \| \varphi(z) - \varphi(w) \|_2^2 \) is the Mahalanobis distance between \( z \) and \( w \).
Corollary

Suppose

\[ K_1 = \ldots = K_d = K. \]

Then, \( \| \varphi(x) - \varphi(y) \|_2^2 \) is the Mahalanobis distance between the empirical kernel maps at \( x \) and \( y \).

Corollary (Linear kernel)

Let

\[ \mathcal{X} = \mathbb{R}^D. \]

\[ K(z, w) = \langle z, w \rangle_2 = z^T w. \]

Then \( \| \varphi(z) - \varphi(w) \|_2^2 \) is the Mahalanobis distance between \( z \) and \( w \).
Semidefinite Relaxation

- Non-convex (d.c. program):

\[
\min_{C, \tilde{h}} \sum_{i=1}^{n} \left[ 2 - n_i^+ + \sum_{j=1}^{n} \left[ 1 + \tau_{ij} \text{tr}(CM_{ij}C^T) - \tau_{ij} \tilde{h} \right]_+ \right] + \lambda \text{tr}(CKC^T) \\
\text{s.t. } C \in \mathbb{R}^{d \times n}, \ C1 = 0, \ \tilde{h} > 0,
\]

where \( M_{ij} := (k^{X_i} - k^{X_j})(k^{X_i} - k^{X_j})^T \) and \( k^{X_i} = [\mathcal{K}(X_1, X_i), \ldots, \mathcal{K}(X_n, X_i)]^T \).

- Semidefinite relaxation:

\[
\min_{\Sigma, \tilde{h}} \sum_{i=1}^{n} \left[ 2 - n_i^+ + \sum_{j=1}^{n} \left[ 1 + \tau_{ij} \text{tr}(M_{ij}\Sigma) - \tau_{ij} \tilde{h} \right]_+ \right] + \lambda \text{tr}(K\Sigma) \\
\text{s.t. } \Sigma \succeq 0, \ 1^T\Sigma1 = 0, \ \tilde{h} > 0.
\]
Semidefinite Relaxation

- Non-convex (d.c. program):

\[
\min_{C, \tilde{h}} \sum_{i=1}^{n} \left[ 2 - n_i^+ + \sum_{j=1}^{n} \left[ 1 + \tau_{ij} \text{tr}(CM_{ij}C^T) - \tau_{ij} \tilde{h} \right]_+ \right]_+ + \lambda \text{tr}(CKC^T)
\]

s.t. \( C \in \mathbb{R}^{d \times n}, C1 = 0, \tilde{h} > 0, \) (7)

where \( M_{ij} := (k^{X_i} - k^{X_j})(k^{X_i} - k^{X_j})^T \) and \( k^{X_i} = [\mathcal{K}(X_1, X_i), \ldots, \mathcal{K}(X_n, X_i)]^T. \)

- Semidefinite relaxation:

\[
\min_{\Sigma, \tilde{h}} \sum_{i=1}^{n} \left[ 2 - n_i^+ + \sum_{j=1}^{n} \left[ 1 + \tau_{ij} \text{tr}(M_{ij}\Sigma) - \tau_{ij} \tilde{h} \right]_+ \right]_+ + \lambda \text{tr}(K\Sigma)
\]

s.t. \( \Sigma \succeq 0, 1^T\Sigma1 = 0, \tilde{h} > 0. \) (8)
Active sets ($\mathcal{A}$): Find $(i,j)$ for which the hinge functions are active.

The program reduces to the form,

$$\min_{\Sigma, \tilde{h}} \operatorname{tr}(A\Sigma) + r\tilde{h} \quad \text{s.t.} \quad \Sigma \succeq 0, \quad 1^T \Sigma 1 = 0, \quad \tilde{h} > 0.$$  

(9)

where $A = \lambda K + \sum_{(i,j)\in\mathcal{A}} \tau_{ij} M_{ij}$ and $r = -\sum_{(i,j)\in\mathcal{A}} \tau_{ij}$.

Alternatively solve for $\Sigma$ and $\tilde{h}$ by gradient descent and projecting onto the convex constraint set.
Algorithm

- **Active sets ($\mathcal{A}$):** Find $(i,j)$ for which the hinge functions are active.

- The program reduces to the form,

\[
\begin{align*}
\min_{\Sigma, \tilde{h}} & \quad \text{tr}(A\Sigma) + r\tilde{h} \\
\text{s.t.} & \quad \Sigma \succeq 0, \ 1^T\Sigma 1 = 0, \ \tilde{h} > 0.
\end{align*}
\]  

(9)

where \( A = \lambda K + \sum_{(i,j) \in \mathcal{A}} \tau_{ij} M_{ij} \) and \( r = -\sum_{(i,j) \in \mathcal{A}} \tau_{ij} \).

- Alternatively solve for $\Sigma$ and $\tilde{h}$ by gradient descent and projecting onto the convex constraint set.
Active sets ($\mathcal{A}$): Find $(i, j)$ for which the hinge functions are active.

The program reduces to the form,

$$
\min_{\Sigma, \tilde{h}} \quad \text{tr}(A\Sigma) + r\tilde{h}
$$

s.t. \[ \Sigma \succeq 0, \quad 1^T \Sigma 1 = 0, \quad \tilde{h} > 0. \]  \hspace{1cm} (9)

where $A = \lambda K + \sum_{(i,j) \in \mathcal{A}} \tau_{ij} M_{ij}$ and $r = -\sum_{(i,j) \in \mathcal{A}} \tau_{ij}$.

Alternatively solve for $\Sigma$ and $\tilde{h}$ by gradient descent and projecting onto the convex constraint set.
Require: \( \{M_{ij}\}_{i,j=1}^n, K, \{\tau_{ij}\}_{i,j=1}^n, \{n_i^+\}_{i=1}^n, \lambda > 0, \epsilon > 0 \) and \( \{\alpha_i, \beta_i\} > 0 \)

1: Set \( t = 0 \). Choose \( \Sigma_0 \in A \) and \( \tilde{h}_0 > 0 \).

2: repeat

3: \( A_t = \{i : \sum_{j=1}^n \left[ 1 + \tau_{ij} \text{tr}(M_{ij} \Sigma_t) - \tau_{ij} \tilde{h}_t \right] + 2 \leq n_i^+ \} \times \{j : j \in [n]\} \)

4: \( B_t = \{(i, j) : 1 + \tau_{ij} \text{tr}(M_{ij} \Sigma_t) > \tau_{ij} \tilde{h}_t\} \)

5: \( N_t = B_t \setminus A_t \)

6: \( \Sigma_{t+1} = P_N(\Sigma_t - \alpha_t \sum_{(i,j) \in N_t} \tau_{ij} M_{ij} - \alpha_t \lambda K) \)

7: \( \tilde{h}_{t+1} = \max(\epsilon, \tilde{h}_t + \beta_t \sum_{(i,j) \in N_t} \tau_{ij}) \)

8: \( t = t + 1 \)

9: until convergence

10: return \( \Sigma_t, \tilde{h}_t \)
Experiments & Results

Set up:

- 5 UCI datasets
- Methods: \(k\)-NN, LMNN, Kernel-NN, KMLCC, KLMCA and KCR (proposed).
- Average error (training/testing) over 20 different splits.
Experiments & Results

- **Iris (150,4,3)**
- **Wine (178,13,3)**
- **Ionosphere (351,34,2)**
- **Balance (625,4,3)**

<table>
<thead>
<tr>
<th>Methods</th>
<th>Train error</th>
<th>Test error</th>
</tr>
</thead>
<tbody>
<tr>
<td>k-NN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>LMNN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Kernel-NN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KMLCC</td>
<td></td>
<td></td>
</tr>
<tr>
<td>KLMCA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>G-KCR</td>
<td></td>
<td></td>
</tr>
<tr>
<td>L-KCR</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bharath K. Sriperumbudur (UCSD)
Proposed a method to embed \((\mathcal{X}, \rho)\) into an \(\ell_2\) space for kernel classification rules.

Learned the bandwidth of the Parzen window.

LMNN requires target neighbors to be defined a priori whereas KCR does not require any such neighbors to be defined.

Compared to LMNN and KLMNN, our method involves fewer tuning parameters.

KCR provides a unified and formal treatment for solving metric learning problems while other methods have been built on heuristics.

Issues: Computationally intensive \(\sim O(n^3)\).
Proposed a method to embed $(\mathcal{X}, \rho)$ into an $\ell_2$ space for kernel classification rules.

Learned the bandwidth of the Parzen window.

LMNN requires target neighbors to be defined a priori whereas KCR does not require any such neighbors to be defined.

Compared to LMNN and KLMNN, our method involves fewer tuning parameters.

KCR provides a unified and formal treatment for solving metric learning problems while other methods have been built on heuristics.

Issues: Computationally intensive $\sim O(n^3)$. 
Discussion & Summary

- Proposed a method to embed \((\mathcal{X}, \rho)\) into an \(\ell_2\) space for kernel classification rules.
- Learned the bandwidth of the Parzen window.
- LMNN requires target neighbors to be defined a priori whereas KCR does not require any such neighbors to be defined.
- Compared to LMNN and KLMNN, our method involves fewer tuning parameters.
- KCR provides a unified and formal treatment for solving metric learning problems while other methods have been built on heuristics.
- Issues: Computationally intensive \(\sim O(n^3)\).
Discussion & Summary

- Proposed a method to embed \((X, \rho)\) into an \(\ell_2\) space for kernel classification rules.
- Learned the bandwidth of the Parzen window.
- LMNN requires target neighbors to be defined a priori whereas KCR does not require any such neighbors to be defined.
- Compared to LMNN and KLMNN, our method involves fewer tuning parameters.
- KCR provides a unified and formal treatment for solving metric learning problems while other methods have been built on heuristics.

Issues: Computationally intensive \(\sim O(n^3)\).
Discussion & Summary

- Proposed a method to embed $(\mathcal{X}, \rho)$ into an $\ell_2$ space for kernel classification rules.

- Learned the bandwidth of the Parzen window.

- LMNN requires target neighbors to be defined a priori whereas KCR does not require any such neighbors to be defined.

- Compared to LMNN and KLMNN, our method involves fewer tuning parameters.

- KCR provides a unified and formal treatment for solving metric learning problems while other methods have been built on heuristics.

- **Issues:** Computationally intensive $\sim O(n^3)$. 
A Probabilistic Theory of Pattern Recognition.
Springer-Verlag, New York.

An equivalence theorem for $L_1$ convergence of the kernel regression estimate.
Journal of Statistical Planning and Inference, 23:71–82.

Metric learning by collapsing classes.

Neighbourhood components analysis.

Asymptotic expansions of the $k$ nearest neighbor risk.

Distance metric learning for large margin nearest neighbor classification.

Distance metric learning with application to clustering with side-information.
Questions
Thank You