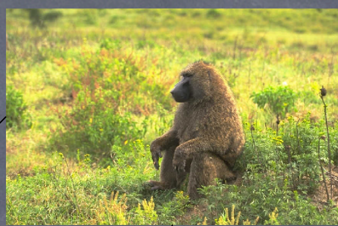
Distance Metric Learning

Lecture 3

Why learn distance functions?

- Nearest Neighbor
- Image retrieval: given a query image, return the K-nearest neighbors on the image from the database
- Euclidean distance on color coherence vectors returns both images as similar to the query image



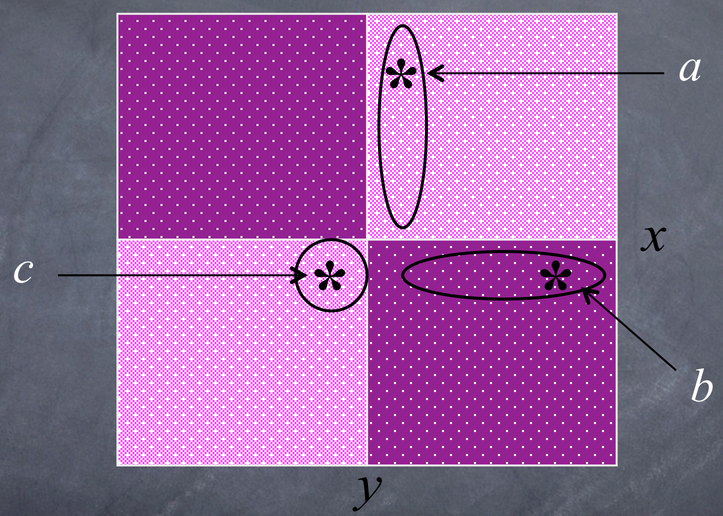
Different Metric Learning Methods

- Supervised:
  - Labels
  - Constraints
- Semi-supervised:
  - Constraints and unlabeled data
- Unsupervised

Global vs. Local

- Global distance metric learning: learns a metric that applies equally over the entire input space; e.g., a metric that satisfies all pairwise constraints simultaneously.
- Local distance metric learning: learns a metric that depends on the location in input space; e.g., a metric that satisfied only local constraints.

Why a local metric?



- Feature relevance changes from location to location: compare a, b, and c.

Examples of Supervised and Global Distance Metric Learning Algorithms

Relevance Component Analysis

N. Shental, T. Hertz, D. Weinshall, and M. Pavel
ECCV 2002

RCA

- Supervised (uses equivalence relations)
- Global
- Learns a Mahalanobis distance measure to improve subsequent unsupervised learning techniques

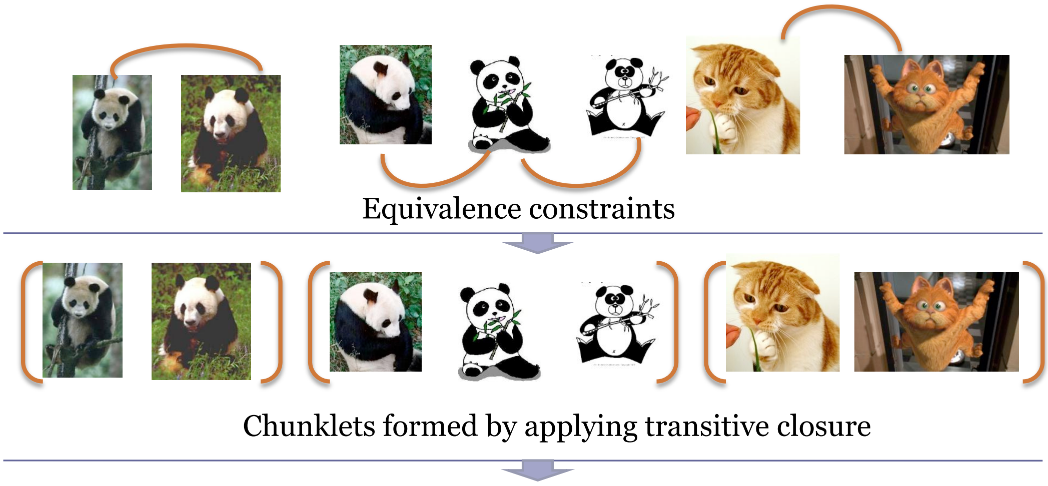
RCA: Basic Idea

- Changes the feature space by assigning large weights to "relevant dimensions" and low weights to "irrelevant dimensions"
- The "relevant dimensions" are estimated using equivalence constraints

Equivalence Constraints and Chunklets

- A chunklet is defined as a subset of points that are known to belong to the same although unknown class
- Chunklets are obtained from equivalence relations by applying a transitive closure

Chunklets and RCA

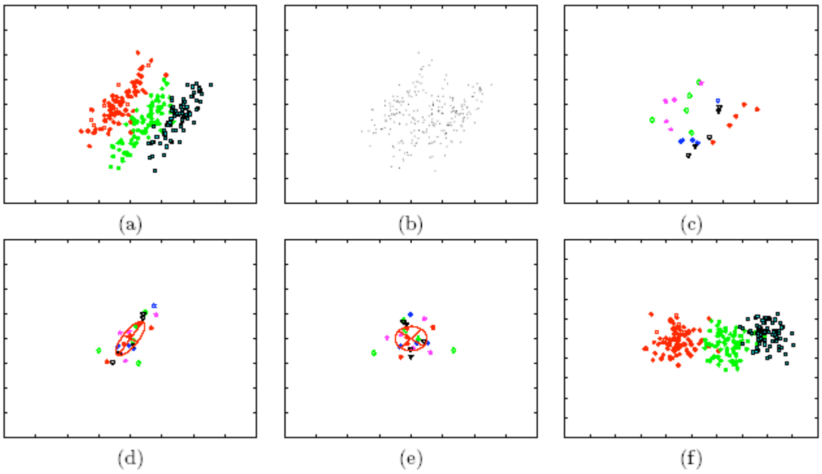


Estimate the within class covariance
dimensions correspond to large with-in covariance are not relevant
dimensions correspond to small with-in covariance are relevant

RCA: Objectives

- RCA identifies and down-scales global unwanted variability within the data
- The RCA transformation is intended to reduce clutter, so that in the new transformed space, the inherent structure of the data can be more easily unrevealed
- The method can be used as a preprocessing step for the unsupervised clustering of data, or KNN classification

Synthetic Gaussian Data [Bar-Hillel et.al. 05]



(a) The fully labeled data set with 3 classes.
(b) Same data unlabeled; classes' structure is less evident.
(c) The set of chunklets that are provided to the RCA algorithm
(d) The centered chunklets, and their empirical covariance.
(e) The RCA transformation applied to the chunklets. (centered)
(f) The original data after applying the RCA transformation.

RCA: The Algorithm

- For each chunklet, subtract the chunklet's mean from all the points it contains
- Compute sum of in-chunklet covariance matrices. Assume a total of p points in k chunklets, where chunklet j consists of $\{x_{ji}\}_{i=1}^{n_j}$ and its mean is $\hat{m}_j$ RCA computes the following matrix:

$$\hat{C} = \frac{1}{p}\sum_{j=1}^{k}\sum_{i=1}^{n_j}(x_{ji}-\hat{m}_j)(x_{ji}-\hat{m}_j)^t$$

RCA: The Algorithm (continued)

$$\hat{C} = \frac{1}{p}\sum_{j=1}^{k}\sum_{i=1}^{n_j}(x_{ji} - \hat{m}_j)(x_{ji} - \hat{m}_j)^t$$

- Compute the transformation (whitening) $W = \hat{C}^{-1/2}$ and apply it to the original data:

$$x_{new} = Wx$$

- Or use the inverse of $\hat{C}$ as a Mahalanobis distance

RCA

- The whitening transformation W assigns lower weight to the directions in which the data variability is mainly due to within class variability, and are therefore "irrelevant" for the task of classification

RCA applied to face images

Top: facial images of two subjects under different lighting conditions.
Bottom: the same images from the top row after applying PCA and RCA and then reconstructing the images

RCA dramatically reduces the effect of different lighting conditions, and the reconstructed images of each person look very similar to each other. [Bar-Hillel, et al. , 2005]

Neighbourhood Components Analysis

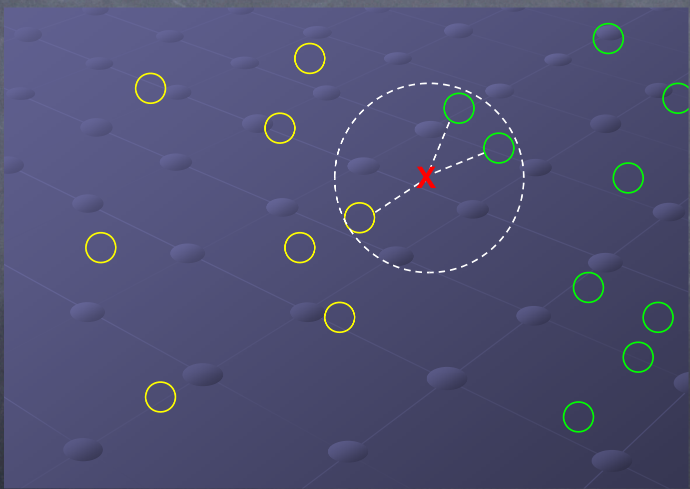
Jacob Goldberger, Sam Roweis, Geoff Hinton, Ruslan Salakhutdinov
NIPS 2005

NCA

- Supervised (uses labels)
- Global
- Learns a Mahalanobis distance measure for KNN classification
- It can also be used for dimensionality reduction

K-nearest neighbor algorithm

- KNN is an extremely simple yet surprisingly effective method for classification



KNN: Advantages and Disadvantages

- Advantages:
  - Simple
  - Nonlinear decision surfaces
  - Quality of prediction automatically improves as the number of training data increases
- Disadvantages:
  - Expensive: must store and search through the entire training set to classify a single test point
  - Must define what we mean by "nearest"

NCA

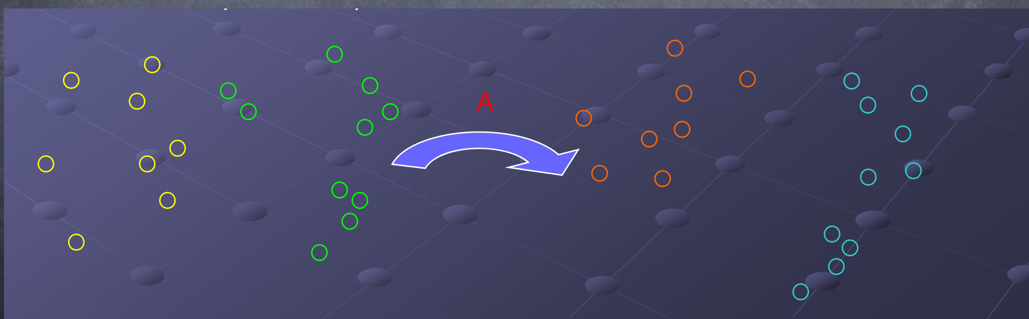
- Restrict to learn Mahalanobis distance metrics:

$$d(x,y) = (x-y)^T Q (x-y)$$

- Q is a symmetric positive semi-definite matrix: $Q = A^T A$

$$d(x,y) = (Ax - Ay)^T (Ax - Ay)$$

- The method learns a linear transformation of the input space



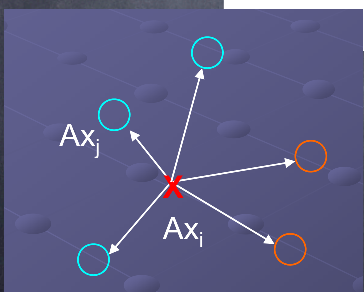
NCA

- Neighbor selection: select as neighbors those points with the same class label as the test point
- Learn a distance metric (matrix A) that achieves this goal over the training data

NCA

- Stochastic neighbor assignments: each point i selects another point j as its neighbor with some probability p_{ij}, and inherits the class label from the point it selects
- The probability is defined as a softmax over the Euclidean distances in the trasformed space:

$$p_{ij} = \frac{\exp(-\|Ax_i - Ax_j\|^2)}{\sum_{k \neq i} \exp(-\|Ax_i - Ax_k\|^2)} \quad , \quad p_{ii} = 0$$



NCA

- Under the stochastic neighbor assignment rule, we can compute the probability that a point i will be correctly classified:

$$p_i = \sum_{j \in C_i} p_{ij} \qquad C_i = \{j | c_i = c_j\}$$

- We want to maximize the expected number of points correctly classified:

$$f(A) = \sum_i \sum_{j \in C_i} p_{ij} = \sum_i p_i$$

$$\max_A f(A)$$

NCA

- Differentiating f with respect to the transformation matrix A ($x_{ij} = x_i - x_j$):

$$\frac{\partial f}{\partial A} = -2A \sum_i \sum_{j \in C_i} p_{ij} (x_{ij} x_{ij}^\top - \sum_k p_{ik} x_{ik} x_{ik}^\top)$$

$$\frac{\partial f}{\partial A} = 2A \sum_i \left(p_i \sum_k p_{ik} x_{ik} x_{ik}^\top - \sum_{j \in C_i} p_{ij} x_{ij} x_{ij}^\top \right)$$

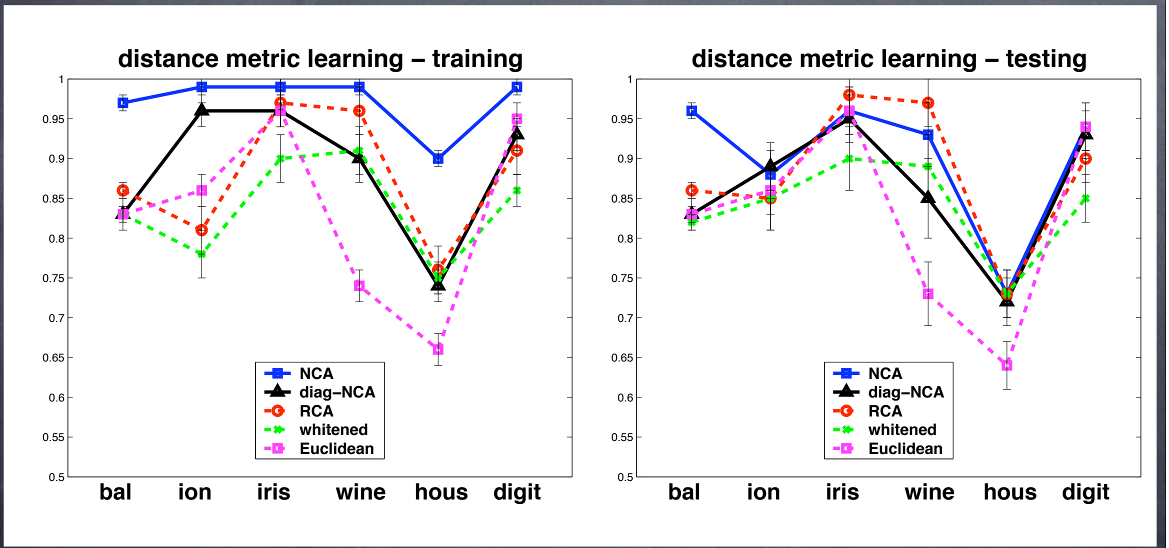
NCA for Dimensionality Reduction

- By restricting A to be a nonsquare matrix of size $d \times D$, NCA can also perform linear dimensionality reduction
- The transformed input will lie in $\mathbb{R}^d$
- The learning algorithm remains the same: maximize the cost function f(A) using gradient ascent over a nonquare A

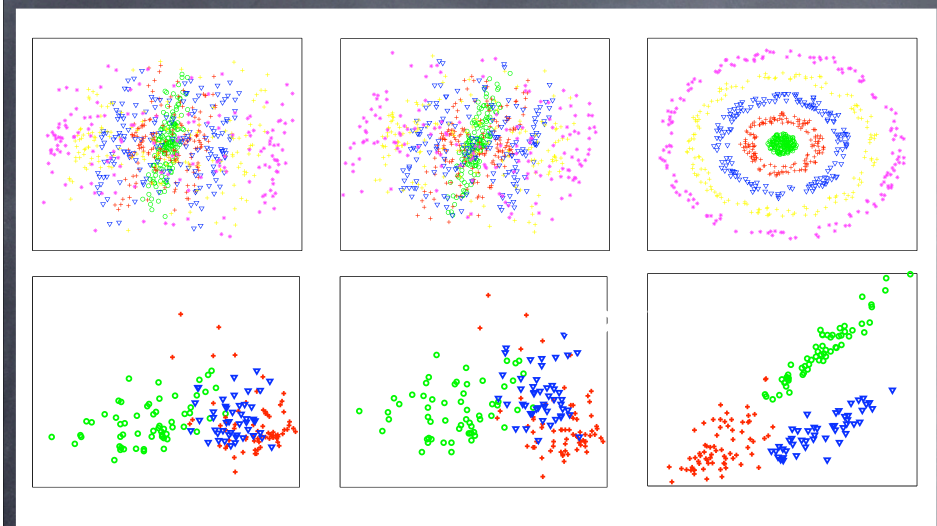
NCA for Dimensionality Reduction

- By using $d << D$ we can significantly reduce the computational load of KNN:
  - Run NCA to find optimal A
  - Store only the projection of training data: $y_n = Ax_n$
  - Given a test point x_{test}
    - Compute its projection $y_{test} = Ax_{test}$
    - Apply KNN on y_{test} using the y_n (and their labels) and a simple Euclidean distance

Experimental Results



Results: Dimensionality Reduction

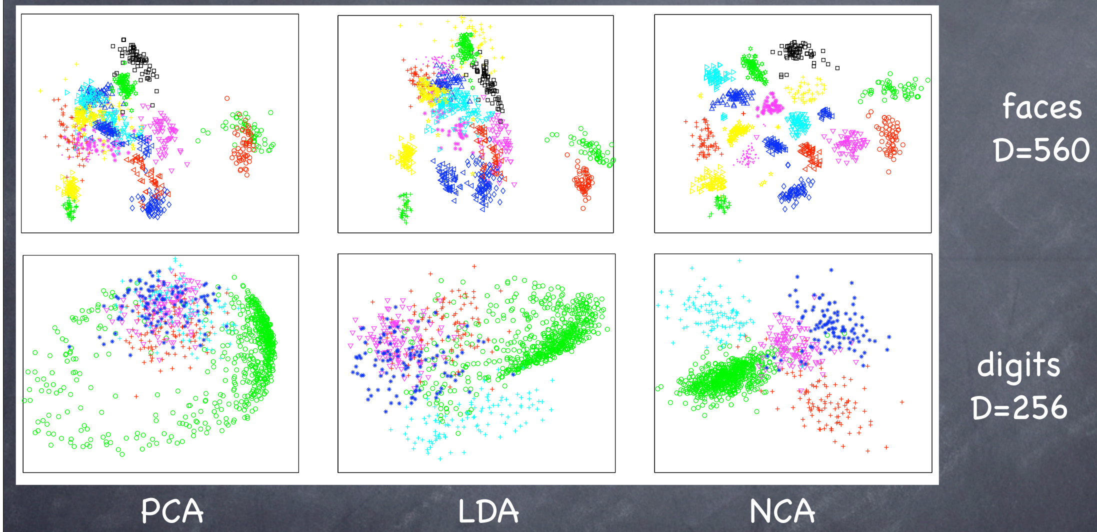


concentric rings D=3

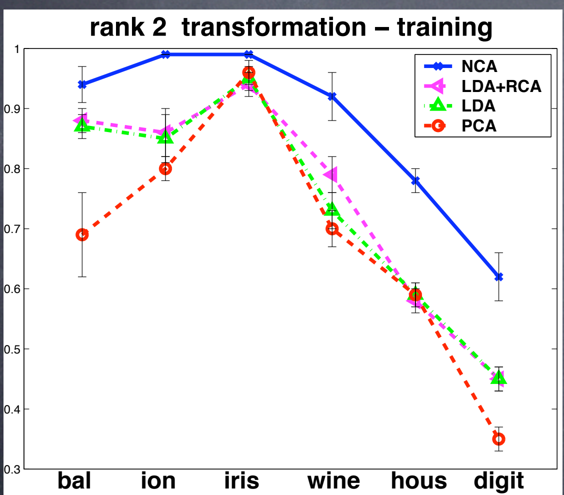
wine D=13

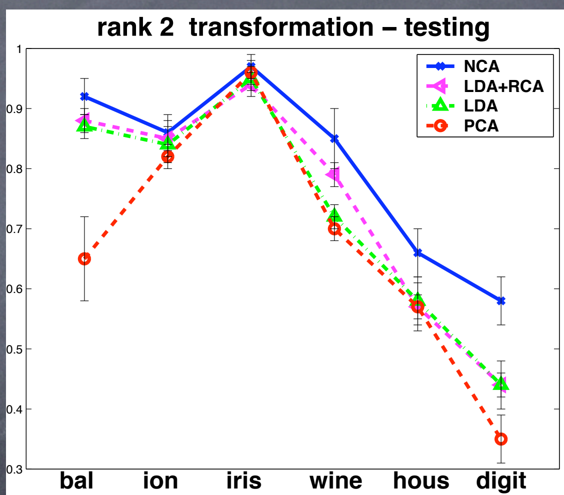
PCA LDA NCA

Results: Dimensionality Reduction



Results: Dimensionality Reduction

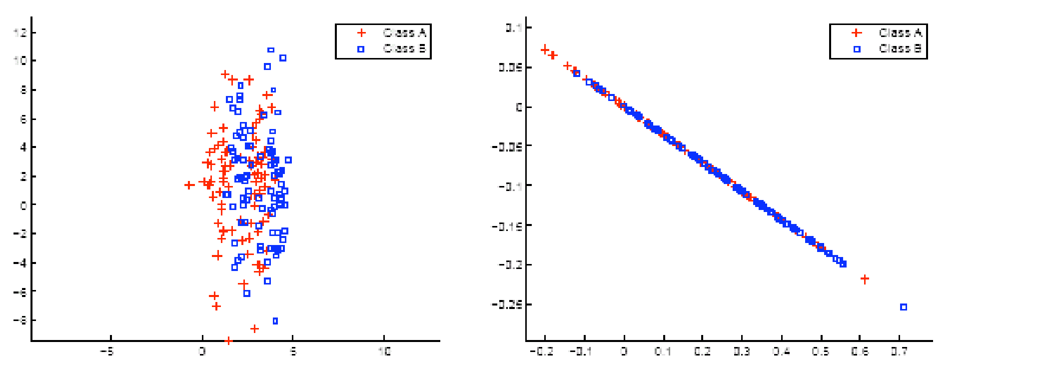




Problems with Global Methods

- The satisfaction of some constraints may conflict with the satisfaction of other constraints

Multimodal data distributions



(a) Data Dist. of the original dataset

(b) Data scaled by the global metric

Multimodal data distributions prevent global distance metrics from simultaneously satisfying constraints on within-class compactness and between-class separability.

Solution

- Instead of attempting to satisfy all constraints, satisfy only local constraints

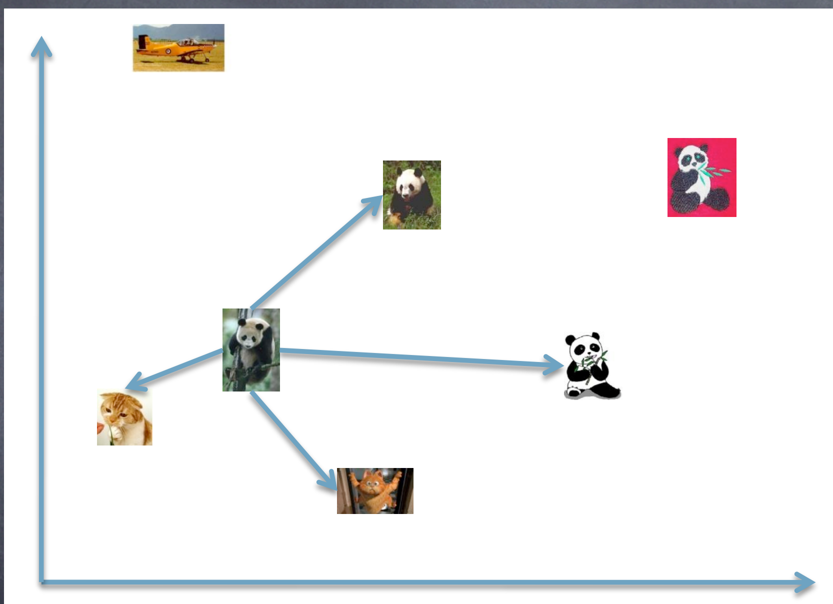
Distance Metric Learning for Large Margin Nearest Neighbor Classification

Kilian Weinberger, John Blitzer, and Lawrence Saul
NIPS 2006

LMNN

- Supervised (uses labels)
- Enforces local constraints
- Final metric is still global
- Learns a Mahalanobis distance measure for KNN classification

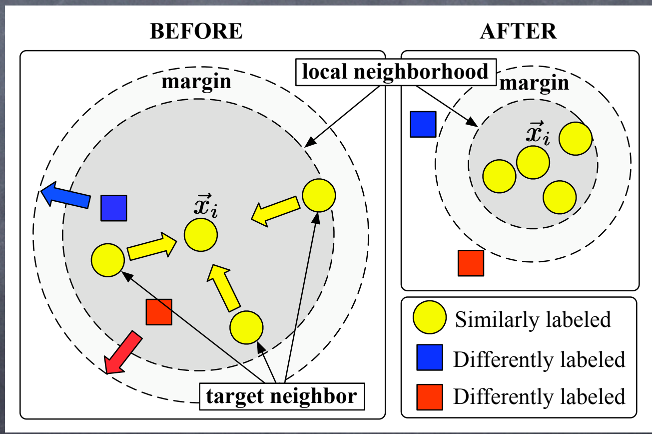
KNN Classification



- We only care about the k nearest neighbors

LMNN: The Overall Idea

- Learns a Mahalanobis distance metric that:
  - Enforces the k-nearest neighbors to belong to the same class
  - Enforces examples from different classes to be separated by a large margin



LMNN: The Approach

- Formulated as an optimization problem
- Solved using a semi-definite programming method

LMNN: The Cost Function

$$\varepsilon(\mathbf{L}) = \sum_{ij} \eta_{ij} \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 + c \sum_{ijl} \eta_{ij}(1 - y_{il}) \left[1 + \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 - \|\mathbf{L}(\vec{x}_i - \vec{x}_l)\|^2\right]_+$$

Distance Function: $\mathcal{D}(\vec{x}_i, \vec{x}_j) = \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2$

Another form of Mahalanobis Distance: $\mathcal{D}(\vec{x}_i, \vec{x}_j) = (\vec{x}_i - \vec{x}_j)^\top \mathbf{M} (\vec{x}_i - \vec{x}_j)$

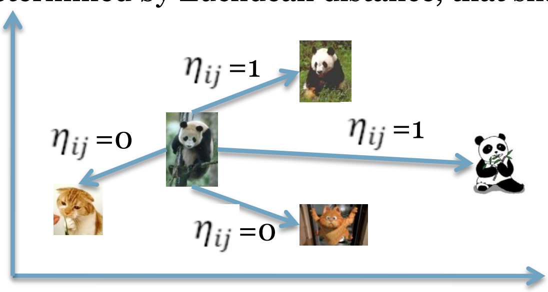
$\mathbf{M} = \mathbf{L}^\top \mathbf{L}$.

LMNN: The Cost Function

$$\varepsilon(\mathbf{L}) = \sum_{ij} \eta_{ij} \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 + c \sum_{ijl} \eta_{ij}(1 - y_{il}) \left[1 + \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 - \|\mathbf{L}(\vec{x}_i - \vec{x}_l)\|^2\right]_+$$

$\eta_{ij} \in \{0, 1\}$ indicate whether input $\vec{x}_j$ is a target neighbor of input $\vec{x}_i$

Target Neighbors: identified as the k-nearest neighbors, determined by Euclidean distance, that share the same label

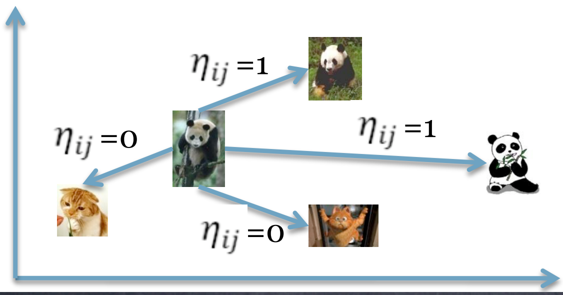


When K=2

LMNN: The Cost Function

$$\varepsilon(\mathbf{L}) = \underbrace{\sum_{ij} \eta_{ij} \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2}_{} + c \sum_{ijl} \eta_{ij}(1 - y_{il}) \left[1 + \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 - \|\mathbf{L}(\vec{x}_i - \vec{x}_l)\|^2\right]_+$$

Penalizes large distances between inputs and target neighbors. In other words, making similar neighbors close



LMNN: The Cost Function

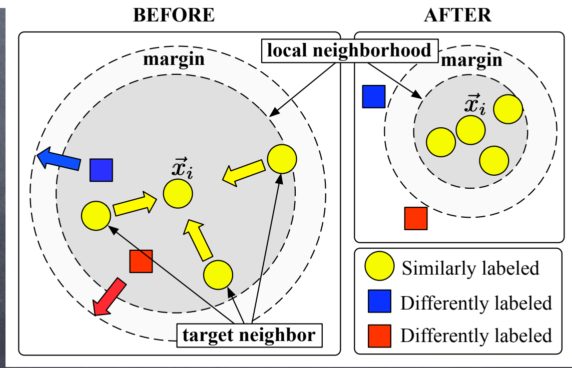
$$\varepsilon(\mathbf{L}) = \sum_{ij} \eta_{ij} \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 + c \sum_{ijl} \eta_{ij}(1 - y_{il}) \underbrace{\left[1 + \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 - \|\mathbf{L}(\vec{x}_i - \vec{x}_l)\|^2\right]_+}_{}$$

$$[z]_+ = \max(z, 0)$$

LMNN: The Cost Function

$$\varepsilon(\mathbf{L}) = \sum_{ij} \eta_{ij} \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 + c\sum_{ijl} \eta_{ij}(1-y_{il})\left[1 + \|\mathbf{L}(\vec{x}_i-\vec{x}_j)\|^2 - \|\mathbf{L}(\vec{x}_i-\vec{x}_l)\|^2\right]_+$$

For inputs and target neighbors
It is equal to 1

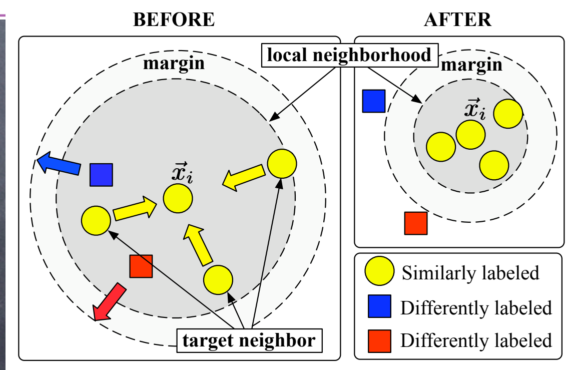


LMNN: The Cost Function

$$\varepsilon(\mathbf{L}) = \sum_{ij} \eta_{ij} \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 + c\sum_{ijl} \eta_{ij}(1-y_{il})\left[1 + \|\mathbf{L}(\vec{x}_i-\vec{x}_j)\|^2 - \|\mathbf{L}(\vec{x}_i-\vec{x}_l)\|^2\right]_+$$

For inputs and target neighbors
It is equal to 1

$y_{il} \in \{0,1\}$ indicates if $\vec{x}_i$ and $\vec{x}_l$ has same label. So For input and neighbors having different labels, it is equal to 1



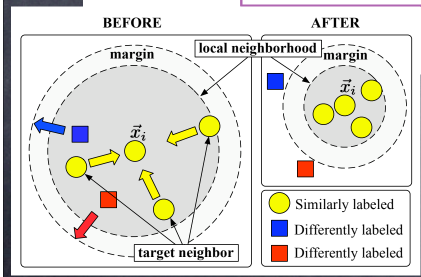
LMNN: The Cost Function

$$\varepsilon(\mathbf{L}) = \sum_{ij}\eta_{ij}\|\mathbf{L}(\vec{x}_i-\vec{x}_j)\|^2 + c\sum_{ijl}\eta_{ij}(1-y_{il})\left[1+\|\mathbf{L}(\vec{x}_i-\vec{x}_j)\|^2-\|\mathbf{L}(\vec{x}_i-\vec{x}_l)\|^2\right]_+$$

For inputs and target neighbors
It is equal to 1

$y_{il} \in \{0,1\}$ indicates if $\vec{x}_i$ and $\vec{x}_l$ has same label. So For input and neighbors having different labels, it is equal to 1

Distance between inputs and target neighbors



LMNN: The Cost Function

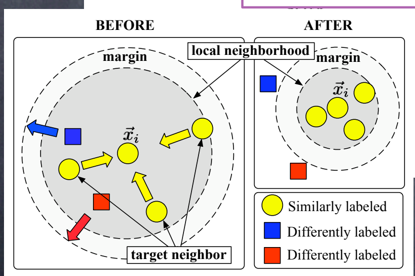
$$\varepsilon(\mathbf{L}) = \sum_{ij}\eta_{ij}\|\mathbf{L}(\vec{x}_i-\vec{x}_j)\|^2 + c\sum_{ijl}\eta_{ij}(1-y_{il})\left[1+\|\mathbf{L}(\vec{x}_i-\vec{x}_j)\|^2-\|\mathbf{L}(\vec{x}_i-\vec{x}_l)\|^2\right]_+$$

For inputs and target neighbors
It is equal to 1

$y_{il} \in \{0,1\}$ indicates if $\vec{x}_i$ and $\vec{x}_l$ has same label. So For input and neighbors having different labels, it is equal to 1

Distance between inputs and target neighbors

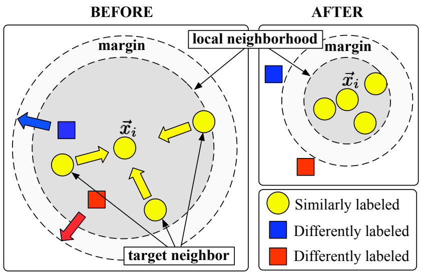
Distance between input and neighbors with different labels



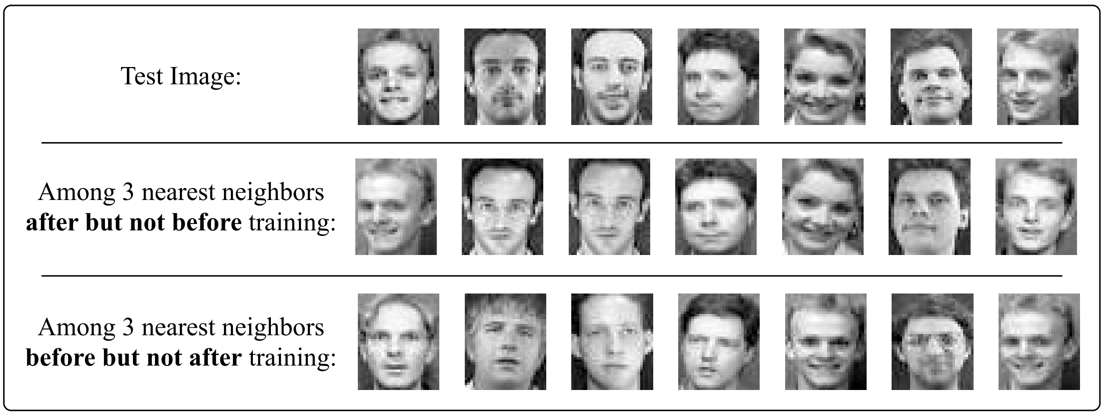
LMNN: The Cost Function

$$\varepsilon(\mathbf{L}) = \sum_{ij} \eta_{ij} \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 + c \sum_{ijl} \eta_{ij}(1 - y_{il}) \left[1 + \|\mathbf{L}(\vec{x}_i - \vec{x}_j)\|^2 - \|\mathbf{L}(\vec{x}_i - \vec{x}_l)\|^2\right]_+$$

Differently labeled neighbors lie outside the smaller radius with a margin of at least one unit distance

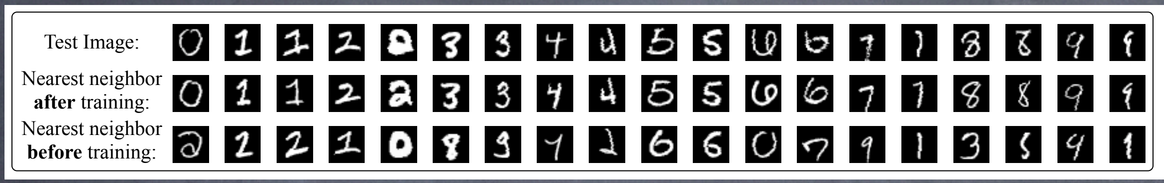


Test on the AT&T face recognition database



- Top row: an image correctly classified by KNN classification (k=3) with Mahalanobis distance, but not with Euclidean distance
- Middle row: correct match among the k=3 nearest neighbors according to the Mahalanobis distance, but not Euclidean distance
- Bottom row: incorrect match among the k=3 nearest neighbors according to the Euclidean distance, but not Mahalanobis distance

Test on MNIST handwritten digit database



- Top row: examples of MNIST images whose nearest neighbor changes during training
- Middle row: nearest neighbor after training, using the Mahalanobis distance metric
- Bottom row: nearest neighbor before training, using the Euclidean distance metric