Decision Estimation

- Decision estimation and classification are ones of active research areas.
  - Classic measurements of the environment, ... , AI (vision, speech recognition.)
    - Systems perform "pattern recognition" or "decision making".
  - Often the information is less than precise, and frequently the decision procedures are statistical in nature.

- Objects of interests are classified into one of classes.
  - These objects are "patterns": printed letters, biological cells, ...
  - Systems learn the training data to classify the testing data.
    - Supervised vs. un-supervised.

Training samples \rightarrow Derive a classification algorithm

Testing samples \rightarrow Classification algorithm

Modification algorithm \rightarrow “Teacher”

Results of classification
Classification Approach

• The observation vector $x$ is first transformed into another vector $y$ whose components called **features**.
  - **Feature extraction**: the features are intended to be fewer in numbers than the observations.
  - However, they should collectively contain most discernible information for pattern classification.
• Reduction of the observations to a smaller number of features is anticipated to help design a reliable decision rule.

- Extraction procedures or transformations attempt to compute the components based on intuition or physical considerations of the problem. $\Rightarrow$ dimensionality reduction.
Three-class Recognition Example

if \( y_1 + 3y_2 < 9 \) \( w_1 \): class of 1's,
else if \( -y_1 + 3y_2 > 3 \) \( w_2 \): class of x's,
else \( w_3 \): class of 0's.

Discriminant functions:
\[
\begin{align*}
g_1(y) &= -y_1 - 3y_2 + 9, \\
g_2(y) &= -y_1 + 3y_2 - 3, \\
g_3(y) &= g_1(y) \cdot g_2(y).
\end{align*}
\]

Decision Rule:
Choose \( w_i \) where \( g_i(y) = \max_j [g_j(y)] \).

• Decision region \( R_i \) is the set \( R_i = \{y : g_i(y) = \max_j [g_j(y)]\} \).
  - Discriminant functions can be evaluated computationally.

• Decision boundaries are defined by \( g_i(y) = g_j(y), \quad i \neq j \).
Probability Theory for Random Vectors

- Event A has the associated probability $P(A)$.
  - $P(\text{not } A) = 1 - P(A)$.
  - The joint probability of two events $A$ & $B$, denoted as $P(AB)$ or $P(A \text{ and } B)$ is the probability that $A$ and $B$ both occur simultaneously.
    - $P(A \text{ or } B) = P(A) + P(B) - P(A \text{ and } B)$.

- Suppose $\mathbf{x}$ is a random vector.
  - Its distribution function $F(\mathbf{x})$ is defined as
    $$F_x(\mathbf{x}) = F_{x_1, x_2, \ldots, x_n}(\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n) = P(\mathbf{x} \leq \mathbf{\tilde{x}}) = P(x_1 \leq \tilde{x}_1, x_2 \leq \tilde{x}_2, \ldots, x_n \leq \tilde{x}_n).$$
  - $F(-\infty)=0, F(+\infty)=1$.
  - Its density function $f(\mathbf{x})$ is defined as
    $$f_x(\mathbf{x}) = \frac{dF_x(\mathbf{\tilde{x}})}{d\mathbf{x}} = \left[ \frac{\partial^n F_x(\mathbf{\tilde{x}})}{\partial x_1 \partial x_2 \cdots \partial x_n} \right]_{\mathbf{x}=\mathbf{\tilde{x}}} \iff F_x(\mathbf{\tilde{x}}) = \int_{-\infty}^{\tilde{x}_1} f_x(\mathbf{x}) dx_1 \int_{-\infty}^{\tilde{x}_2} f_x(\mathbf{x}) dx_2 \cdots \int_{-\infty}^{\tilde{x}_n} f_x(\mathbf{x}) dx_n.$$
Joint Distribution and Density Functions

- Suppose \(y\) is another random vector.

- The joint distribution of \(x\) and \(y\) is defined by

\[
F_{xy}(\tilde{x}, \tilde{y}) = P(x \leq \tilde{x}, y \leq \tilde{y}).
\]

- The joint density is

\[
f_{xy}(\tilde{x}, \tilde{y}) = \frac{d^2 F_x(\tilde{x})}{dxdy} = \left[ \frac{\partial^m \partial^n F_x(\tilde{x})}{\partial x_1 \partial x_2 \cdots \partial x_n \partial y_1 \partial y_2 \cdots \partial y_m} \right]_{x=\tilde{x}, y=\tilde{y}}.
\]

\[\Rightarrow\]

\[
F_{xy}(\infty, \infty) = 1, \quad F_{xy}(\tilde{x}, \infty) = F_x(\tilde{x}), \quad F_{xy}(\infty, \tilde{y}) = F_y(\tilde{y}).
\]

Marginal p.d.f.

- Example: the joint p.d.f. is

\[
f_{xy}(\tilde{x}, \tilde{y}) = \begin{cases} (\tilde{x}_1 + 3\tilde{x}_2)\tilde{y}_1 & 0 \leq \tilde{x}_1, \tilde{x}_2, \tilde{y}_1 \leq 1, \\ 0 & \text{otherwise}. \end{cases}
\]

\[
f_x(\tilde{x}) = \begin{cases} \int_0^1 (\tilde{x}_1 + 3\tilde{x}_2) y_1 dy_1 = \frac{1}{2} (\tilde{x}_1 + 3\tilde{x}_2) & 0 \leq \tilde{x}_1, \tilde{x}_2 \leq 1, \\ 0 & \text{otherwise}. \end{cases}
\]
Probability Functions jointly with Events

- The joint distribution of a random vector $\mathbf{x}$ and an event $A$ is defined by

$$F_{xA}(\tilde{\mathbf{x}}, A) = P(\mathbf{x} \leq \tilde{\mathbf{x}}, A) = \sum_{i=1}^{m} P(\mathbf{x} \leq \tilde{\mathbf{x}}, A_i), \quad A = \bigcup_{i=1}^{m} A_i.$$ 

- The conditional probability:

$$P(A | B) = \frac{P(A \cap B)}{P(B)} = \frac{P(B | A)P(A)}{P(B)}.$$ 

$$\Rightarrow F_{x|A}(\tilde{\mathbf{x}} | A) = P(\mathbf{x} \leq \tilde{\mathbf{x}} | A) = \frac{P(\mathbf{x} \leq \tilde{\mathbf{x}}, A)}{P(A)} = \frac{F_{xA}(\tilde{\mathbf{x}}, A)}{P(A)}.$$ 

Bayes's Rule

$$\Rightarrow f_{x|y}(\tilde{x} | \tilde{y}) = \frac{f_{xy}(\tilde{x}, \tilde{y})}{f_y(\tilde{y})}.$$ 

$$\Rightarrow f_{xy}(\tilde{x}, \tilde{y}) = f_x(\tilde{x}) \cdot f_y(\tilde{y}), \text{ if } \mathbf{x}, \mathbf{y} \text{ are independent.}$$

$$P(A_i | B) = \frac{P(A_i \cap B)}{P(B)} = \frac{P(B | A_i)P(A_i)}{\sum_{i=1}^{m} P(B | A_i)}$$

$$\Rightarrow f_{x|y}(\tilde{x} | \tilde{y}) = \frac{f_{xy}(\tilde{x}, \tilde{y})}{f_y(\tilde{y})} = \frac{f_{y|x}(\tilde{y} | \tilde{x})f_x(\tilde{x})}{\int_{-\infty}^{\infty} f_{y|x}(\tilde{y} | \tilde{x})f_x(\mathbf{x})d\mathbf{x}}.$$ 

Prior density

Posterior density
Likelihood Ratio Test

• The hypothesis that a given pattern \( \mathbf{x} \) belongs one of \( N_c \) classes is tested to minimize the probability of error.

\[
\text{If } f_{w_1|x}(w_1 | \tilde{x}) = \frac{f_{x|w_1} (\tilde{x} | w_1) P(w_1)}{f_x (\tilde{x})} > f_{w_2|x} (w_2 | \tilde{x}) = \frac{f_{x|w_2} (\tilde{x} | w_2) P(w_2)}{f_x (\tilde{x})} \quad \Rightarrow \quad w_1 \text{ class,}
\]

\[
\text{else } w_2 \text{ class.}
\]

• The decision rule can be

\[
L(\tilde{x}) \overset{\Delta}{=} \frac{f_{x|w_1} (\tilde{x} | w_1)}{f_{x|w_2} (\tilde{x} | w_2)} > \frac{P(w_2)}{P(w_1)}\]

Likelihood Ratio

• Example: the conditional p.d.f. are

\[
P(w_1) = P(w_2).
\]

\[
\Rightarrow \quad L(\tilde{x}) \overset{\Delta}{=} \exp[- \frac{1}{2} (x - 4)^2 + \frac{1}{2} (x - 10)^2] > 1 \quad \Rightarrow \quad (x - 4)^2 - (x - 10)^2 < 0.
\]

\[
\Rightarrow \quad \tilde{x} = x < 7.
\]

\[
\Rightarrow \quad \text{Usually, the log likelihood Ratio is used.}
\]
Probability of Misclassification

- The probability of error, i.e. Bayes risk, determines the quality of a decision rule.
  - A lower value implies a better rule.

\[ P(\text{error}) = \int_{-\infty}^{\infty} P(\text{error} \mid x) f_x(x) dx = P(\text{error} \mid w_1)P(w_1) + P(\text{error} \mid w_2)P(w_2). \]

\[ \varepsilon_1 = P(\text{error} \mid w_1) = P(\text{choose } w_2 \mid w_1) = \int_{R_2} f_{x \mid w_1}(x \mid w_1) dx. \]

\[ \varepsilon_2 = P(\text{error} \mid w_2) = P(\text{choose } w_1 \mid w_2) = \int_{R_1} f_{x \mid w_2}(x \mid w_2) dx. \]

- Bayes risk under Multiple hypotheses can be defined:

\[ P(\text{error} \mid x) = \sum_{j=1, j \neq i}^{N_c} P(w_j \mid x) = 1 - P(w_i \mid x), \quad \text{if } x \in R_i. \]

- \( R_i \) should be defined to be the region where \( P(w_i \mid x) \) is largest.

Choose \( w_i \) where \( P(w_i \mid x) = \max_j [P(w_j \mid x)] \).
Distance Functions

• There are several ways to measure the distance \( d(x, y) \) between two vectors \( x \) & \( y \).
  
  - Generally, a distance function is any scalar-valued function satisfying the following conditions:
    
    • \( d(x, y) > 0 \) for \( x \neq y \); \( d(x, y) = 0 \) if \( x = y \).
    
    • \( d(x, y) = d(y, x) \).
    
    • [Triangular inequality] \( d(x, y) + d(y, z) \geq d(x, z) \).
  
  - Euclidean distance: \( d_E(x, y) = |x - y| = \left( \sum_{i=1}^{n} (x_i - y_i)^2 \right)^{\frac{1}{2}} \).
  
  - Maximum value distance: \( d_M(x, y) = \max_i |x_i - y_i| \).
  
  - Absolute value distance (city block): \( d_A(x, y) = \sum_{i=1}^{n} |x_i - y_i| \).
Linear Transformation

• If \( \mathbf{x} \) is a vector in \( X \) and \( \mathbf{y} \) is the corresponding (mapped) vector in \( Y \), then \( \mathbf{y} = A \mathbf{x} \).

• Matrix \( A \) is said to be **positive definite**
  - if the quadratic product, \( \mathbf{x}^T A \mathbf{x} \), is strictly greater than zero for all non-zero vector \( \mathbf{x} \).

\[
\mathbf{x}^T A \mathbf{x} = \sum_{i=1}^{n} \sum_{j=1}^{n} a_{ij} x_i x_j.
\]

• Matrix \( A \) is said to be **positive semidefinite**
  - if the quadratic product, \( \mathbf{x}^T A \mathbf{x} \), \( \geq 0 \) for all non-zero vector \( \mathbf{x} \).

• Example: the positive definite matrix

\[
A = \begin{pmatrix} 1 & -1 \\ -1 & 2 \end{pmatrix}.
\]

\[
\mathbf{x}^T A \mathbf{x} = \sum_{i=1}^{2} \sum_{j=1}^{2} a_{ij} x_i x_j = x_1^2 - 2x_1x_2 + 2x_2^2 = (x_1 - x_2)^2 + x_2^2 > 0, \ \forall \mathbf{x} \neq \mathbf{0}.
\]

\[
B = \begin{pmatrix} 2 & 1 & 1 \\ 1 & 2 & -1 \\ 3 & 3 & 0 \end{pmatrix}
\]

is not a positive definite matrix.
Differentiation w.r.t. Vectors

- If \( s \) is a scalar function of a vector \( \mathbf{x} \in \mathbb{R}^n \),
  - the derivative of \( s \) w.r.t. \( \mathbf{x} \) is defined as the vector (gradient)
    \[
    \frac{ds}{d\mathbf{x}} = \left[ \frac{\partial s}{\partial x_1} \quad \frac{\partial s}{\partial x_2} \quad \cdots \quad \frac{\partial s}{\partial x_n} \right]^T.
    \]

- If \( \mathbf{s} \) is a vector \( \in \mathbb{R}^m \),
  - the derivative of \( \mathbf{s} \) w.r.t. \( \mathbf{x} \) is the matrix
    \[
    \frac{d\mathbf{s}}{d\mathbf{x}} = \begin{bmatrix}
        \frac{\partial s_1}{\partial x_1} & \cdots & \frac{\partial s_m}{\partial x_1} \\
        \vdots & \ddots & \vdots \\
        \frac{\partial s_1}{\partial x_n} & \cdots & \frac{\partial s_m}{\partial x_n}
    \end{bmatrix}.
    \]

- If \( \mathbf{r} \) is a linear transformation of \( \mathbf{s} \),
  - Namely, \( \mathbf{r} = \mathbf{B}\mathbf{s} \), then
    \[
    \frac{d\mathbf{r}}{d\mathbf{x}} = \mathbf{B} \frac{d\mathbf{s}}{d\mathbf{x}}.
    \]

- If \( \mathbf{r} \) is a nonlinear transformation of \( \mathbf{s} \),

- For the quadratic product,
  \[
  \frac{d(\mathbf{x}^T \mathbf{B}\mathbf{x})}{d\mathbf{x}} = \cdots = (\mathbf{B} + \mathbf{B}^T)\mathbf{x} = 2\mathbf{B}\mathbf{x}.
  \]

If \( \mathbf{B} \) is symmetric
Correlation and Covariance Matrices

• The correlation matrix $\mathbf{R}$ of a random vector $\mathbf{x}$ is

$$\mathbf{R} = E(\mathbf{x}\mathbf{x}^\top) = (r_{ij}), \quad r_{ij} = E(x_i x_j).$$

• The covariance matrix $\mathbf{K}$ of a random vector $\mathbf{x}$ is

$$\mathbf{K} = E[(\mathbf{x} - \mathbf{m})(\mathbf{x} - \mathbf{m})^\top] = (k_{ij}), \quad k_{ij} = E[(x_i - m_i)(x_j - m_j)].$$

  - The diagonal elements $k_{ii}$ are the variances of the vector components.

$$k_{ii} = \sigma_i^2 = \text{Var}(x_i) = E[(x_i - m_i)^2].$$

$$\mathbf{K} = E[(\mathbf{x} - \mathbf{m})(\mathbf{x} - \mathbf{m})^\top] = E[\mathbf{x}\mathbf{x}^\top - \mathbf{x}\mathbf{m}^\top - \mathbf{m}\mathbf{x}^\top + \mathbf{m}\mathbf{m}^\top]$$

$$= E[\mathbf{x}\mathbf{x}^\top] - E[\mathbf{x}]\mathbf{m}^\top - \mathbf{m}E[\mathbf{x}^\top] + \mathbf{m}\mathbf{m}^\top = E[\mathbf{x}\mathbf{x}^\top] - \mathbf{m}\mathbf{m}^\top = \mathbf{R} - \mathbf{m}\mathbf{m}^\top. \quad \Rightarrow \quad \mathbf{R} = \mathbf{K} + \mathbf{m}\mathbf{m}^\top.$$

• Given a vector $\mathbf{y} = A \mathbf{x}$, $\mathbf{m}_y = E[A\mathbf{x}] = AE[\mathbf{x}] = A\mathbf{m}_x$.

$$\mathbf{R}_y = E[(A\mathbf{x})(A\mathbf{x})^\top] = AE[\mathbf{x}\mathbf{x}^\top]A^\top = A\mathbf{R}_x A^\top. \quad \mathbf{K}_y = AK_x A^\top.$$

  - If $A$ is orthogonal, i.e. has orthonormal column vectors,

$$|\mathbf{R}_y| = |\mathbf{R}_x|, \quad \text{tr}(\mathbf{R}_y) = \text{tr}(\mathbf{R}_x), \quad |\mathbf{K}_y| = |\mathbf{K}_x|, \quad \text{tr}(\mathbf{K}_y) = \text{tr}(\mathbf{K}_x).$$
Independent Random Vectors

- If random vectors $\mathbf{x}$ and $\mathbf{y}$ are independent, they are uncorrelated.
  - The converse is generally not true.
  - Both are uncorrelated if
    \[
    R_{xy} = E(\mathbf{x}\mathbf{y}^T) = E(\mathbf{x})E(\mathbf{y}^T) = \mathbf{m}_x\mathbf{m}_y^T, \\
    K_{xy} = E[(\mathbf{x} - \mathbf{m}_x)(\mathbf{y} - \mathbf{m}_y)^T] = 0.
    \]

- Gaussian Random Vectors:
  \[
  f(\mathbf{x}) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\mathbf{x}^T\mathbf{x}}{2\sigma^2}} \Rightarrow f(\mathbf{x}) = \frac{1}{(2\pi)^{n/2}\sqrt{|K_{xx}|}} \exp\left[-\frac{(\mathbf{x} - \mathbf{m})K_{xx}^{-1}(\mathbf{x} - \mathbf{m})^T}{2}\right].
  \]

- Recall: $a/(a+c)$, better for a smaller value of $c$.
  - The ratio of the number of shots detected correctly over the actual number of shots.

- Precision: $a/(a+b)$, better for a smaller value of $b$.
  - The ratio of the number of shots detected correctly over the total number of shots detected.
Support Vector Machines (SVM)

- SVM is a novel kind of Neural Networks.
  - Multi-Layer-Perceptron (MLP): Classifier, regressor, etc.
    - Single-layer & Multi-layer with feed-forward connections.
    - Back propagation algorithm, maximum likelihood principle.
  - Training, self-structured: supervised, unsupervised.
  - The performance is justified by a loss function (say, MSE) over unseen samples of the test set.
    - The expected risk of the classifier on the test set \([2]\) \(\leq\) The empirical risk on the training set \([0]\) + the estimation error \([1]\).

\[
\text{Estimation error} \approx \sqrt{\frac{h}{c}} \log(1 + 2 \frac{c}{h}), \quad c = |\text{Training Set}|, \\
h = \text{VC dimension of the classifier.}
\]

\([0]\) & \([1]\) should be both min.

✓ Minimizing \([0]\) alone do no good!!

(Vapnik-Chervonenkis)= the maximal number of samples correctly classified in the training set.
• In Modeling, the approximation error stems from the model mismatch.
  - The true $f(x)$ may lie outside the hypothesis space.

• In Learning, the estimation error occurs due to the imperfect learning procedure.
  - The non-optimal model (empirically obtained) may be chosen.

• During the testing (evaluation), the generalization error is met.

• SVM minimizes the Expected risk by controlling VC dimension.
  - Learning becomes solving the problem of Quadratic Programming.
SVM = Optimal Hyperplane Algorithm

- Learning how to classify is estimating a function \( f : \mathbb{R}^n \rightarrow \pm 1 \) over the training data set \( \{(x_i, y_i) \in \mathbb{R}^n \times \pm 1 : i=1\ldots c\} \)
  - \( f \) will correctly classify other unseen example \((x, y)\) under the same unknown probability distribution \( P(x, y) \). \( \Rightarrow \) namely, \( f(x) = y \).
  - It is often assumed the data are i.i.d. (identically independent distributed).

Hyperplanes: \( w \cdot x + b = 0, \quad w \in \mathbb{R}^n, \quad b \in \mathbb{R} \).

\( \Rightarrow \) \exists \ a unique hyperplane \((w, b) \) s.t. \( \max_{w,b} \min_i \left( \|x - x_i\| : x \in \mathbb{R}^n, w \cdot x + b = 0, i = 1\ldots c \right) \).

Maximize the separation margin.

\( \Rightarrow \) Optimization problem:

\[
\begin{align*}
\min L(w) &= \frac{1}{2} \|w\|^2 & \text{Good separation} \\
\sum_{i=1}^c \alpha_i[y_i(w \cdot x_i + b) - 1] &= 1 & \text{Correct}
\end{align*}
\]

\( \Rightarrow \) Solution = the saddle point of the Lagrangian:

\[
L(w, b, \alpha) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^c \alpha_i[y_i(w \cdot x_i + b) - 1].
\]

Minimized w.r.t. \( w \) & \( b \), maximized w.r.t. \( \alpha \).
Solution

\[
\frac{\partial L(w,b,\alpha)}{\partial b} = 0 \implies \sum_{i=1}^{c} \alpha_i y_i = 0, \quad \alpha_i \neq 0 \implies \text{Support vectors}
\]

\[
\frac{\partial L(w,b,\alpha)}{\partial w} = 0 \implies w = \sum_{i=1}^{c} \alpha_i y_i x_i.
\]

The solution vector is a linear combination of a subset of the training patterns.

\[\Rightarrow \text{Support vectors summarize the information.}\]

\[
\Rightarrow b = -\frac{1}{2} w \cdot (x_p + x_q), \alpha_p > 0, \alpha_q > 0, y_p = 1, y_q = -1, \text{for any SV } x_p, x_q.
\]

- However, most classification problems are not linear separable.
  - Transform \( x_i \) to a high-dimension space to regain linear separation.

\[
x \implies \Phi(x). \quad \Phi(x) \text{ is hard to compute.}
\]

Decision Functions:

\[
f(x) = \text{sgn}(w \cdot x + b) = \text{sgn}\left(\sum_{i=1}^{c} \alpha_i y_i (x_i \cdot x) + b\right)
\]

\[
\Rightarrow f(x) = \text{sgn}\left(\sum_{i=1}^{c} \alpha_i y_i (\Phi(x_i) \cdot \Phi(x)) + b\right).
\]

The scalar (inner) product, \( \Phi(x_i) \cdot \Phi(x) \), is easy to compute by a simple kernel.

As an example, the polynomial kernel \( k(x,y) = \Phi(x) \cdot \Phi(y) = (x \cdot y)^d \).

✓ Matrices \( (K_{ij}) \) are positive definite, where \( K_{ij} = k(x_i,x_j), i, j = 1 \cdots c \).
Dilemma

• Typically, the data will only be linearly separable in some, possibly very high dimensional space.
  - Separating the data exactly, particularly for a finite amount of data with noise, is favorable. However it will generalize badly.
  - In practice, it may be necessary to employ the non-separable approach (allow some classification error).

• To allow some overlapping between classes, the slack variables \( \tau_i \geq 0 \) is introduced.

\[
\Rightarrow \text{Optimization problem:} \quad \begin{cases} 
\min L(w, \tau) = \frac{1}{2} \|w\|^2 + C \sum_{i=1}^c \tau_i, & C \text{ is some constant } \geq 0. \\
y_i(w \cdot x_i + b) \geq 1 - \tau_i, & i = 1 \cdots c. \\
\tau_i \geq 0
\end{cases}
\]

\[
\Rightarrow L(w, b, \alpha, \tau, \beta) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^c \alpha_i [y_i(w \cdot x_i + b) - 1 + \tau_i] - \sum_{i=1}^c \beta_i \tau_i. \\
C \geq \alpha_i \geq 0 \quad \sum_{i=1}^c \alpha_i y_i = 0, \quad w = \sum_{i=1}^c \alpha_i y_i x_i, \quad \alpha_i + \beta_i = C. \quad \text{(Any SV } x_i \text{ has } \tau_i=0.\) \\
\Rightarrow \max_\alpha W(\alpha) = \sum_{i=1}^c \alpha_i - \frac{1}{2} \sum_{i=1}^c \sum_{j=1}^c \alpha_i \alpha_j y_i y_j (x_i \cdot x_j).
Non-linear Separation

• Using the kernel,
  \[ \max_{\alpha} W(\tilde{\alpha}) = \sum_{i=1}^{c} \alpha_i - \frac{1}{2} \sum_{i=1}^{c} \sum_{j=1}^{c} \alpha_i \alpha_j y_i y_j k(x_i, x_j). \]

• Define the matrix \( Q \), \( Q_{ij} = y_i y_j k(\tilde{x}_i, \tilde{x}_j). \) \[ \Rightarrow W(\tilde{\alpha}) = \tilde{\alpha}^T \tilde{1} - \frac{1}{2} \tilde{\alpha}^T Q \tilde{\alpha} = \tilde{\alpha}^T (\tilde{1} - \frac{1}{2} Q \tilde{\alpha}). \]

• Decomposition: break the entire training set into smaller ones.
  - Select the working (active) subset.
    • Other \( \alpha_i \) are fixed in the current iteration.
  - Shrink the problem.
    • There are much less SVs than \( c \).
    • Many SVs have \( \alpha_i = C \).
    • Caching and incremental updates of the gradient & the termination criteria.
Generalized Discriminant Analysis (GDA)

- GDA is the eigenvalue problem resolution for nonlinear discriminant analysis.
  - It is similar in functionality to SVM.
- The input set X has m vectors, $x_1 \ldots x_m$, belong to n classes, $X_1 \ldots X_n$.
  - The cardinality of the subset $X_i$ is $m_i$. \( X = \bigcup_{i=1}^{n} X_i, \sum_{i=1}^{n} m_i = m. \)
  - The covariance matrix C of all $x_i$: \[ C = \frac{1}{m} \sum_{i=1}^{m} x_i x_i^T. \]

Suppose \( x \Rightarrow \Phi(x). \)

\[ C \Rightarrow V = \frac{1}{m} \sum_{i=1}^{m} \Phi(x_i)\Phi^T(x_i). \]

Center $\Phi(x_i)$ in the transform space: \[ \tilde{\Phi}(x_i) = \Phi(x_i) - \frac{1}{m} \sum_{k=1}^{m} \Phi(x_k). \]

- The inter-class inertia $B$ is the covariance matrix of the class centers.
  \[ B = \frac{1}{m} \sum_{i=1}^{n} m_i \Phi_i \Phi_i^T, \quad \Phi_i = \frac{1}{m_i} \sum_{k=1}^{m_i} \Phi(x_{i,k}). \]

Likewise, \[ V = \frac{1}{m} \sum_{i=1}^{m} \Phi(x_i)\Phi^T(x_i) = \frac{1}{m} \sum_{i=1}^{n} \sum_{k=1}^{m_i} \Phi(x_{i,k})\Phi^T(x_{i,k}). \text{total inertia} \]
Formulation

• Using Kernel function: \( k(x_i, x_j) = k_{i,j} = \Phi^T(x_i)\Phi(x_j). \)

For classes p & q,

\[
\left( k_{i,j} \right)_{p,q} = \Phi^T(x_{p,i})\Phi(x_{q,j}).
\]

• Define a mxm matrix \( K: \)

\[
K = \left( K_{p,q} \right)_{p,q=1 \ldots n}, \quad K_{p,q} = \left( k_{i,j} \right)_{i=1 \ldots m_p, j=1 \ldots m_q} = K_{q,p}^T.
\]

A mxm block diagonal matrix \( W: \)

\[
W = \left( W_t \right)_{t=1 \ldots n}, \quad W_t = \left( \frac{1}{m_t} \right)_{m_t \times m_t}.
\]

• The classical criteria for class separability is defined by the quotient between the inter-class inertia and the intra-classes inertia.
  - Its maximization is equivalent to the eigenvalue resolution.

• Assume the classes follow a multivariate Gaussian distribution, and each observation can be assigned to the class having the maximum posterior probability using the **Mahalanobis distance**.
Eigenvalue Resolution

- Given two symmetric matrices $A$ & $B$ with the same size, and $B^{-1}$ exists,
  - The quotient $\frac{v^T Av}{v^T Bv}$ is maximal for eigenvector $v$ of $B^{-1}A$ associated to the large eigenvalue $\lambda$.

Since

$$\frac{(v^T Bv)(2Av) - (v^T Av)(2Bv)}{(v^T Bv)^2} = 0 \Rightarrow B^{-1}Av = \left(\frac{v^T Av}{v^T Bv}\right)v.$$

$$\Rightarrow \left(\frac{v^T Av}{v^T Bv}\right): \text{eigenvalue}, v: \text{eigenvector of } B^{-1}A.$$

Therefore, the quotient $\frac{B}{V}$ of both inertia's in the problem is maximized:

$$\lambda Vv = Bv \quad \Rightarrow \quad \left(\frac{v^T Bv}{v^T Vv}\right) = \lambda : \text{the largest eigenvalue}, v: \text{eigenvector of } V^{-1}B.$$

$$B = \frac{1}{m} \sum_{i=1}^{n} m_i \Phi_i \Phi_i^T, \quad \Phi_i = \frac{1}{m_i} \sum_{k=1}^{m_i} \Phi(x_{i,k}).$$

$$V = \frac{1}{m} \sum_{i=1}^{n} \Phi(x_i)\Phi^T(x_i) = \frac{1}{m} \sum_{i=1}^{n} \sum_{k=1}^{m_i} \Phi(x_{i,k})\Phi^T(x_{i,k}).$$

$$v = \sum_{i=1}^{n} \sum_{k=1}^{m_i} \alpha_{i,k} \Phi(x_{i,k}).$$

Linear combination
Formulation

\[ \alpha = (\alpha_i)_{i=1\ldots n}, \quad \alpha_i = (\alpha_{i,k})_{k=1\ldots m_i} \quad \Rightarrow \quad \begin{pmatrix} v^T Bv \\
\nu^T V_v \end{pmatrix} = \lambda = \begin{pmatrix} \alpha^T KWK\alpha \\
\alpha^T KK\alpha \end{pmatrix}. \]

- **Proof:**

\[ \lambda V_v = Bv \quad \Rightarrow \quad \lambda \Phi^T (x_{r,s}) V_v = \Phi^T (x_{r,s}) Bv. \]

\[ V_v = \frac{1}{m} \sum_{p=1}^n \sum_{i=1}^{m_p} \Phi(x_{p,i}) \Phi^T (x_{p,i}) \times \sum_{q=1}^n \sum_{k=1}^{m_q} \alpha_{q,k} \Phi(x_{q,k}) \]

\[ = \frac{1}{m} \sum_{q=1}^n \sum_{k=1}^{m_q} \alpha_{q,k} \sum_{p=1}^n \sum_{i=1}^{m_p} \Phi(x_{p,i}) \Phi^T (x_{p,i}) \Phi(x_{q,k}). \]

\[ \lambda \Phi^T (x_{r,s}) V_v = \frac{\lambda}{m} \sum_{q=1}^n \sum_{k=1}^{m_q} \alpha_{q,k} \sum_{p=1}^n \sum_{i=1}^{m_p} \left[ \Phi^T (x_{r,s}) \Phi(x_{p,i}) \right] \left[ \Phi^T (x_{p,i}) \Phi(x_{q,k}) \right]. \]

\[ \Rightarrow \quad \lambda \left[ \Phi^T (x_{1,m_1}), \ldots, \Phi^T (x_{m_1,m_1}), \ldots, \Phi^T (x_{1,m_n}), \ldots, \Phi^T (x_{m_n,m_n}) \right] V_v = \frac{\lambda}{m} KK\alpha. \]

\[ Bv = \frac{1}{m} \sum_{p=1}^n m_p \left[ \frac{1}{m_p} \sum_{i=1}^{m_p} \Phi(x_{p,i}) \right] \left[ \frac{1}{m_p} \sum_{i=1}^{m_p} \Phi(x_{p,i}) \right]^T \times \sum_{q=1}^n \sum_{k=1}^{m_q} \alpha_{q,k} \Phi(x_{q,k}) \]

\[ = \frac{1}{m} \sum_{q=1}^n \sum_{k=1}^{m_q} \alpha_{q,k} \sum_{p=1}^n \left[ \sum_{i=1}^{m_p} \Phi(x_{p,i}) \right] \left[ \frac{1}{m_p} \right] \left[ \sum_{i=1}^{m_p} \Phi^T (x_{p,i}) \Phi(x_{q,k}) \right]. \]

\[ \Rightarrow \quad \left[ \Phi^T (x_{1,m_1}), \ldots, \Phi^T (x_{m_1,m_1}), \ldots, \Phi^T (x_{1,m_n}), \ldots, \Phi^T (x_{m_n,m_n}) \right] Bv = \frac{1}{m} KWK\alpha. \]
Eigenvalue Resolution

• By the eigenvectors decomposition of $K$, $K = P \Gamma P^T$
  
  - $P$ contains the normalized eigenvectors, say $v$.
  
  - $P$ is orthonormal since $K$ is symmetric.
  
  - $\Gamma$ is the diagonal matrix with non-zero eigenvalues.

\[
\begin{align*}
\left( \frac{v^T B v}{v^T V v} \right) &= \lambda = \left( \frac{\alpha^T K WK \alpha}{\alpha^T KK \alpha} \right) = \frac{\alpha^T \left( P \Gamma P^T \right) W \left( P \Gamma P^T \right) \alpha}{\alpha^T \left( P \Gamma P^T \right) \left( P \Gamma P^T \right) \alpha} = \frac{\left( \Gamma P^T \alpha \right)^T P^T WP \left( \Gamma P^T \alpha \right)}{\left( \Gamma P^T \alpha \right)^T P^T \left( \Gamma P^T \alpha \right)}.
\end{align*}
\]

$\beta = \Gamma P^T \alpha \quad \Rightarrow \quad \lambda P^T P \beta = \lambda \beta = P^T WP \beta \quad \Rightarrow \quad \alpha = P \Gamma^{-1} \beta.$

Also,

\[
1 = v^T v = \sum_{p=1}^{n} \sum_{k=1}^{m_p} \alpha_{p,k} \Phi^T(x_{p,k}) \sum_{q=1}^{n} \sum_{k=1}^{m_q} \alpha_{q,k} \Phi(x_{q,k}) = \sum_{p=1}^{n} \sum_{q=1}^{n} \alpha^T_p K_{p,q} \alpha_q = \alpha^T K \alpha \quad \Rightarrow \quad \alpha \text{ should be normalized by } \sqrt{\alpha^T K \alpha}.
\]

Given a test vector $z$, the projections can be computed as

\[
v^T z = \sum_{p=1}^{n} \sum_{k=1}^{m_p} \alpha_{p,k} \Phi^T(x_{p,k}) z = \sum_{p=1}^{n} \sum_{k=1}^{m_p} \alpha_{p,k} k(x_{p,k}, z).
\]
Summary

- GDA procedure is summarized in the following steps:

1. Compute K and W. 
   
   \[
   (k_{i,j})_{p,q} = \Phi^T(x_{p,i})\Phi(x_{q,j}).
   \]

   \[
   K = \left(K_{p,q}\right)_{p,q=1\ldots n}, \quad K_{p,q} = \left(k_{i,j}\right)_{i=1\ldots m_p, j=1\ldots m_q}, \quad W = (W_t)_{t=1\ldots n}, \quad W_t = \left(\frac{1}{m_t}\right)_{m_t \times m_t}.
   \]

2. Decompose K using eigenvectors decompositions.

   \[
   K = P\Gamma P^T
   \]

3. Compute eigenvectors \( \beta \) and eigenvalues of the system.

   \[
   \beta = \Gamma P^T\alpha \quad \Rightarrow \quad \lambda P^T P \beta = \lambda \beta = P^T W P \beta \quad \Rightarrow \quad \alpha = P \Gamma^{-1} \beta.
   \]

4. Compute eigenvectors \( v \) using \( \alpha \) and normalize them.

   \[
   v = \sum_{p=1}^n \sum_{k=1}^{m_p} \alpha_{p,k} \Phi(x_{p,k}^T). \quad \alpha \text{ should be normalized by } \sqrt{\alpha^T K \alpha}.
   \]

5. Compute projections of test points onto the eigenvectors \( v \).

   \[
   v^T z = \sum_{p=1}^n \sum_{k=1}^{m_p} \alpha_{p,k} \Phi^T(x_{p,k}) z = \sum_{p=1}^n \sum_{k=1}^{m_p} \alpha_{p,k} k(x_{p,k}, z).
   \]
Kernel Functions

- Various kernel functions can be used:
  - Gaussian kernel, RBF-kernel: 
    \[ k(x, y) = \exp\left(\frac{-\|x-y\|^2}{2\sigma^2}\right). \]
  - Polynomial kernel: 
    \[ k(x, y) = (x \cdot y)^d. \]
    \[ d = 2 \implies (x_1^2, \ldots, x_i^2, x_i x_2, \ldots, x_i x_j, \ldots): \frac{t(t-1)}{2} \text{ terms for } x \in R^t. \]

- Threshold values are learned and chosen.
  - The number of classes minus one is the number of thresholds chosen for classification.
Biased Discriminant Transform (BDT)

• MM information retrieval relies on the descriptors (or feature vectors), a set of real numbers.
  – Effectiveness of the representation in descriptors.
  – Selection of similarity metric.

• Difference between Traditional and MM DB:
  – Binary “Hit-or-Miss” decision using keywords in traditional DB.
    • The occurrences of the keywords or their synonyms, or
    • Rule-based ranking, etc.
  – In MMDB, the feature space is $R^n$ (continuous).
    • Inherently, it is a nearest neighbor or a top-k ranking problem.
Why On-Line Learning?

• "Consensus" interpretation on MM contents:
  - Among all the users
  - Among all the times
    - The correct answer should match the context of conversation.
  - “The bat slipped from his hand.” shows different meaning in the context of a baseball game or a cave exploring.
  - Medical image DB may define specific functionalities to perform off-line pre-clustering.

• On-learning is indispensable.
  - The system need to communicate with the user to perceive the specific goal of the queries.
    - In CBIR, a user is required to offer the feature-weighting scheme.
    - In "Relevance Feedback", a user is kept in the loop to tell the relevance of an image or video. (NO R/W of textual description)
Supervised Classification Problem

• One descriptor is assumed to represent the MM object.
  - By it, the media type becomes transparent to the system.
  - The object can be an whole image, image block, segmented region, shorts, frames, or a key frame.
  ✓ A point is associated with the descriptor in the feature space.

• Relevance feedback: supervised classification problem.
  - Learning Speed: the number of iterations.
  - Training Size: the number of samples, i.e. their population.
    • Class density, positive/negative samples, etc.
  - Top-k returns: not a binary decision.
    • Binary classification (two-class) may not be optimal.

  ➢ Initial results are returned; returns/evaluation are iterated.
  ➢ The goal is to learn the discriminating \textit{subspace}. 
Variants of Relevance Feedback

• Objectives:
  – A user may look for a particular object or a similar one.

• Feedbacks:
  – A user may give back the positive feedback, negative, or both.
  – The degree of relevance for each result may be returned.
  – Partial likeness: it is like A in color, like B in shape, etc.

• Multiple Descriptors per Sample:
  – A mixed model can be used for refinement (intersection, union) to emphasize the local features.

• Class distribution:
  – Two or more target classes may be assumed.
    • Gaussian: two; Kernel-based: more for non-linearity.

• Data Organization:
  – A hierarchical tree structure may slow learning in real-time.

• Focus:
  – To learn a linear transformation, consider the correlations of feature components, estimate the class density, etc.
Fisher & Multiple Discriminant Analyses

• The consensus is to find the features to best cluster & separate the positive examples from the negative.

• Traditional approaches:
  - Two-class assumption (FDA): to find a lower dimensional space in which the ratio of between-class scatter over within-class scatter is maximized.
    \[ W = \arg \max \left| \frac{W^T S_b W}{W^T S_w W} \right|. \]
    \[ S_b = (m_x - m)(m_x - m)^T + (m_y - m)(m_y - m)^T. \]
    \[ S_w = \sum_{i=1}^{N_x} (x_i - m_x)(x_i - m_x)^T + \sum_{i=1}^{N_y} (y_i - m_y)(y_i - m_y)^T. \]
    \( x_i \): positive; \( y_i \): negative.
    
  - Two-class assumption (MDA):
    \[ S_b = (m_x - m)(m_x - m)^T + \sum_{i=1}^{N_y} (y_i - m)(y_i - m)^T. \]
    \[ S_w = \sum_{i=1}^{N_x} (x_i - m_x)(x_i - m_x)^T. \]
Biased Discriminant Analysis (BDA)

- *(1+x)*-class assumption:
  - The user is only interested in one class, while there are an unknown number of other classes.
  - “All happy families are alike, each unhappy family is unhappy in its own fashion” - Leo Tolstoy's Anna Karenina.

- All positive examples are alike in a way; each negative example is negative in its own way.

\[
W = \arg_w \max \frac{W^T S_y W}{W^T S_x W}.
\]

\[
S_y = \sum_{i=1}^{N_y} (y_i - m_x)(y_i - m_x)^T. \\
S_x = \sum_{i=1}^{N_x} (x_i - m_x)(x_i - m_x)^T.
\]

- Regularization and Discounting Factors:
  - Sample-based estimates may be severely biased for small number of training examples.

\[
S'_x = (1 - \mu)S_x + \frac{\mu}{n} tr[S_x]I. \\
S_d = (1 - \gamma)S_y + \frac{\gamma}{n} tr[S_y]I.
\]

n=dim(original space).
Kernel-based BDA (KBDA)

- For non-linearity in the data, a non-linear mapping \( \Phi : \mathbf{x} \rightarrow \Phi(\mathbf{x}) \) is used to restore linearity in the transform space.

  - The evaluation of kernel \( K = (k_{ij}) \), where \( k_{ij} = \Phi^T(\mathbf{x}_i)\Phi(\mathbf{x}_j) \).

\[
W = \arg_{\mathbf{w}} \max \frac{\mathbf{w}^T S_y^\Phi \mathbf{w}}{\mathbf{w}^T S_x^\Phi \mathbf{w}}.
\]

\[
S_y^\Phi = \sum_{i=1}^{N_y} (\Phi(\mathbf{y}_i) - \mathbf{m}_x^\Phi)(\Phi(\mathbf{y}_i) - \mathbf{m}_x^\Phi)^T.
\]

\[
S_x^\Phi = \sum_{i=1}^{N_x} (\Phi(\mathbf{x}_i) - \mathbf{m}_x^\Phi)(\Phi(\mathbf{x}_i) - \mathbf{m}_x^\Phi)^T.
\]

- Let \( \mathbf{w} \) is the eigenvector associated with the largest eigenvalue for \( W \).

\[
\mathbf{w} = \sum_{i=1}^{N_x} \alpha_i \Phi(\mathbf{x}_i) + \sum_{j=1}^{N_y} \alpha_{i+N_x} \Phi(\mathbf{y}_i) = \Phi \mathbf{a}.
\]

\[
\mathbf{K}_{\mathbf{y}_i} = \Phi^T \Phi(\mathbf{y}_i) = \left( \mathbf{K}_y \right)_{:,i},
\]

\[
\mathbf{K}_{\mathbf{m}_x} = \Phi^T \mathbf{m}_x^\Phi,
\]

\[
\mathbf{I}_{N_x}^y = \frac{1}{N_x} (\mathbf{1})_{N_x \times N_y}.
\]
KBDA (cont’d)

\[
K_x = N \times N_x, \\
I_N = \frac{1}{N} (I)_{N \times N_x}. 
\]

\[
w^T S_x \Phi w = \alpha^T \left[ (K_x - K_x I_{N_x}^x) (K_x - K_x I_{N_x}^x)^T \right] \alpha \\
= \alpha^T K_x \left[ (I - I_{N_x}^x) (I - I_{N_x}^x)^T \right] K_x^T \alpha \\
= \alpha^T K_x (I - I_{N_x}^x)^2 K_x^T \alpha.
\]

- Solve to get \( \alpha \): the eigenvector with the largest eigenvalue.
- Given a new pattern \( z \), find its projection onto \( w \) by

\[
w^T \Phi(z) = \sum_{i=1}^{N_x} \alpha_i k(x_i, z) + \sum_{j=1}^{N_y} \alpha_{i+N_x} k(y_i, z).
\]

• In this new space, the nearest neighbors of the positive centroid are returned in each iteration.
  - Combined with the subsequent feedbacks, the new nearest neighbors are output.
RBF-Kernel:

\[ k(x, y) = \exp \left( -\frac{||x - y||^2}{2\sigma^2} \right) \]

Primal optimization problem:

\[
\begin{align*}
\text{minimize} & \quad \tau(w) = \frac{1}{2} ||w||^2 \\
\text{subject to} & \quad y_i \cdot ((w \cdot x_i) + b) \geq 1, \quad i = 1, \ldots, \ell.
\end{align*}
\]

Decision function:

\[ f(x) = \text{sgn} \left( \sum_{i=1}^{\ell} y_i \alpha_i \cdot (x \cdot x_i) + b \right) \]

Figure 5 Test results on synthetic training data: six different configurations of non-linearity. The circles are positive examples and the crosses negative. A simulated query process is used for training sample selection, i.e., the 20 nearest neighbors of a randomly selected positive point are used as training samples. The bar diagram shows the averaged hit rate in top 20 returns.