In previous parametric supervised approaches, it is assumed that the form of probability is known.

Now, assume the form of the discriminant function is known.

Assume this form is linear either in components or functions of $x$.

In such cases, LDF are relatively easy to compute and analytically attractive.
LDF and Decision Surfaces (1/2)

- Assume a two-class problem, then LDF:
  \[ g(x) = \langle w, x \rangle + w_0 \]
  
  - Decide \( w_1 \) if \( g(\bar{x}) > 0 \)
  - Decide \( w_2 \) if \( g(\bar{x}) < 0 \)
  - Either class if \( g(\bar{x}) = 0 \)

- \( g(x) = 0 \) is a decision surface and since \( g(x) \) is linear, this surface is a hyperplane
- This hyperplane, \( H \), divides the feature space into two subspaces, \( R_1 \) & \( R_2 \)
- Vector \( w \) is normal to any vector on \( H \)

LDF and Decision Surfaces (2/2)

- Note that \( g(x) \) gives a measure of distance from \( x \) to \( H \):
  \[ \bar{x} = \bar{x}_p + r \frac{\bar{w}}{\|\bar{w}\|} \]
  \[ g(\bar{x}_p) = 0 \]

  \[ g(\bar{x}) = \langle \bar{w}, \bar{x} \rangle + w_0 = r \|\bar{w}\| \]

  \[ \Rightarrow r = \frac{g(\bar{x})}{\|\bar{w}\|} \]
Classification using LDF

- Two main approaches for classification using LDF:
  - Fisher's Linear Discriminant:
    Project data onto a line with "good" discrimination; then classify on this 1-D space easily
  - Linear Discrimination in d-dimensions
    Classify data using "suitable" hyperplanes

Fisher's Linear Discriminant (1/3)

- The Fisher's approach projects d-dimensional data onto a line, \( w \)
- It is expected that these projections are well separated by class

![Diagram showing Fisher's Linear Discriminant]
Fisher’s Linear Discriminant (2/3)

- Feature vector projections: \( y_i = \mathbf{w}^T \mathbf{x}_i \quad i = 1, \ldots, n \)
- Measures for separation based on w:
  - Difference between projection means
  - Variance of within-class projection data
- Choose \( \mathbf{w} \) in order to maximize \( J \)

\[
J(\mathbf{w}') = \frac{(m_1 - m_2)^2}{\frac{1}{\mathbf{S}_1} + \frac{1}{\mathbf{S}_2}}
\]

where \( m_i \) : projection means for class i

\[
\mathbf{S}_i = \sum_{y \in \mathbf{Y}_i} (y - m_i)^2 : \text{scatter}
\]

Fisher’s Linear Discriminant (3/3)

- Relation between sample & projection means:

\[
\bar{m}_i = \frac{1}{n_i} \sum_{x \in \mathbf{X}_i} x \Rightarrow m_i = \frac{1}{n_i} \sum_{y \in \mathbf{Y}_i} y = \frac{1}{n_i} \sum_{x \in \mathbf{X}_i} \mathbf{w}^T x = \mathbf{w}^T \bar{m}_i
\]

- Define scatter matrices \( \mathbf{S}_i \)

\[
\mathbf{S}_i = \sum_{x \in \mathbf{X}_i} (x - \bar{m}_i)(x - \bar{m}_i)^T \quad \text{and} \quad \mathbf{S}_w = \mathbf{S}_1 + \mathbf{S}_2
\]

- Then, \( \mathbf{w} \) which maximizes \( J \) can be found as:

(Read derivation at Duda&Hart)

\[
\tilde{\mathbf{w}} = \mathbf{S}_w^{-1} (\tilde{m}_1 - \tilde{m}_2)
\]
LDF for Multi-class Problems

There are 3 ways to classify multi-category problems:

1. **Wi/not-Wi dichotomies** (c-1 2-class problems)
   - Find a LDF that will separate i-th class from rest of the classes

2. **Wi/Wj dichotomies** (c(c-1)/2 2-class problems)
   - Find a LDF that will separate i-th class from j-th class

3. **Linear Machine**
   - Divides the feature space into c while \( g_i(x) \) being the largest DF in i-th class/region

---

**Generalized LDF (1/2)**

- **Linear Discriminant Function**:
  \[
  g(x) = w_0 + \sum_{i=1}^{d} w_i x_i
  \]

- **Quadratic Discriminant Function**:
  \[
  g(x) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=i+1}^{d} w_{ij} x_i x_j
  \]

- **Polynomial Discriminant Function**:
  \[
  g(x) = w_0 + \sum_{i=1}^{d} w_i x_i + \sum_{i=1}^{d} \sum_{j=i+1}^{d} w_{ij} x_i x_j + \sum_{i=1}^{d} \sum_{j=1}^{d} \sum_{k=1}^{d} w_{ijk} x_i x_j x_k + \cdots
  \]

- **Generalized Linear Discriminant Function**:
  \[
  g(x) = w_0 + \sum_{i=1}^{d} a_i y_i(x) \quad \text{e.g. } y_i(x) = \begin{bmatrix} 1 \\ x_i \\ x_i^2 \end{bmatrix}
  \]
Generalized LDF (2/2)

- Assume a simple quadratic DF: \( g(x) = a_1 + a_2 x + a_3 x^2 \)
- In order to make DF linear, let \( y = [1 \ x \ x^2] \)
- Note that \( a = (-1, 1, 2) \Rightarrow g(x) > 0 \) for \( x < -1 \) & \( x > 0.5 \)

Two-category Linearly Separable Case (1/3)

- If a vector that classifies correctly all the samples of two classes exits, than the samples are called *linearly separable*.
- In order to simplify analysis, perform the conversion

\[
g(\vec{x}) = w_0 + \sum_{i=1}^{n} w_i x_i, \text{ let } \vec{y} = \begin{bmatrix} 1 \\ x_1 \\ \vdots \\ x_n \end{bmatrix}, \vec{a} = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix} \Rightarrow g(\vec{x}) = \vec{a}' \vec{y}
\]

- Assume \( n \)-samples of \( y \) vector, some labeled \( w_1 \) and some \( w_2 \), then the unknown \( a \) vector has the following constraints according to the correct classifications:
  \( \vec{y}_i \) is labelled \( w_1 \) \( \Rightarrow a' \vec{y}_i > 0 \)
  \( \vec{y}_i \) is labelled \( w_2 \) \( \Rightarrow a' \vec{y}_i < 0 \) or \( a' (-\vec{y}_i) > 0 \)
Two-category Linearly Separable Case (2/3)
- In "solution space", each sample $y$ is a constraint to find a solution for vector $a$, such that $y$ vector is normal to the hyperplane $a'y = 0$.

Two-category Linearly Separable Case (3/3)
- Since solution vector is not unique, one option is to choose this vector such that $a'y_i > b > 0$ for all $i$.

- Motivation for going to the 'middle' portion is due to natural belief for better classification of new samples.
Gradient Descent Procedures (1/3)

- In order to find a "solution vector" $a$, a criterion $J(a)$ should be minimized using an optimization method.
  - **Steepest Descent** is a well-known method:
    
    $$a_{k+1} = a_k - \rho_k \nabla J(a_k)$$
    
    - Step size choice is critical:
      - if it is too small $\Rightarrow$ slow convergence
      - if it is too large $\Rightarrow$ convergence overshoot, diverge

Gradient Descent Procedures (2/3)

- If we approximate $J(a)$ by 2nd-order expansion

$$J(a) \approx J'(a) = J(a_k) + \nabla J'(a-a_k) + \frac{1}{2} (a-a_k)' \frac{\partial^2}{\partial a^2} J(a-a_k)$$

1) Step size can be selected optimally:

Let $a_t = a_t = a_t - \rho_t \nabla J(a_t)$

$$\min_{\rho} J'(a_t) = \min_{\rho} \left[ J(a_t) - \rho \nabla J(a_t) \right] = \min_{\rho} \frac{1}{2} \rho^2 \nabla J^2(a_t) \Rightarrow \rho_t = \frac{\|\nabla J\|^2}{\nabla J^2(a_t)}$$

2) $a_{k+1}$ can be chosen to minimize the approximation of $J(a)$ (Newton's descent)

$$\min_{a} J'(a) = \min_{a} \left[ J(a_k) + \nabla J'(a-a_k) + \frac{1}{2} (a-a_k)' \frac{\partial^2}{\partial a^2} J(a-a_k) \right] \Rightarrow a_{k+1} = a_k - \frac{\nabla J}{\nabla J^2(a_k)}$$
Gradient Descent Procedures (3/3)

- Comparing simple steepest-descent to Newton's descent,
  - Newton's algorithm will give better improvement per step even for optimal step size of steepest-descent
  - $D$ for 2\textsuperscript{nd} derivatives (Hessian matrix) can be singular
  - Even $D$ is non-singular, computation for inverse is high

Minimizing Perceptron Criterion (1/2)

- Lets define a criterion function for solving $\alpha'\gamma > 0$
  $$J_p(\alpha) = \sum_{y \in Y(\alpha)} (-\alpha' \bar{y})$$
  \(Y(\alpha)\): set of misclassified samples

- Note that
  - $-\alpha' \bar{y}$ is always positive for misclassified data and equal to zero if all samples are correctly classified
  - Perceptron criterion is proportional to the sum distances of misclassified samples to decision boundary

- Using one of the descent procedures, minimize $J_p(\alpha)$
Minimizing Perceptron Criterion (2/2)

- The gradient of $J_p(a)$ is obtained as: $\nabla J_p(\tilde{a}) = \sum_{j:y_j=1} (-\tilde{y})$
- Using gradient descent, for $k$th iteration: $\tilde{a}_{k+1} = \tilde{a}_k + \rho \sum_{j:y_j=1} \tilde{y}$
- If stepsize is constant \(\rightarrow\) fixed increment case

**Geometrical Interpretation:**
Angle between $a_k$ and $y$ should be $< \pi/2$

**Example:** $a'$

- If samples are linearly separable, convergence to a solution is guaranteed by perceptron method (read the proof at Duda&Hart)

Non-separable Behavior

- Approaches based on separability assumption, relentlessly search for an error-free solution
- In practice, if there is no a priori info about separability,
  - such procedures should be modified with an appropriate termination rule so that divergence is avoided
  - one should seek for other approaches
    - MSE procedures
    - Ho-Kashyap approach
Minimum Squared Error Procedures (1/2)

- Rather than trying to make $a_i y_i > 0$ for all $i$, let's make $a_i y_i = b_i$ for an arbitrary constant $b_i > 0$.

$$
Y = \begin{bmatrix}
  y_1^r \\
  \vdots \\
  y_n^r
\end{bmatrix} ; \quad b = \begin{bmatrix}
  b_1 \\
  \vdots \\
  b_n
\end{bmatrix} \Rightarrow \text{Find } \hat{a} \text{ such that } Y \hat{a} = \hat{b}
$$

- Since $a$ is usually overdetermined, a solution can be found such that the square of the 'error', $e = Ya - b$, is minimized.

Minimum Squared Error Procedures (2/2)

- $e = Ya - b$ is minimized by finding the pseudo inverse of $Y$:

$$\hat{a} = Y^+ \hat{b} = (Y' Y)^{-1} Y' \hat{b}$$

- Note that these approaches do not try to find a separating plane but rather minimize the error.

- Selection of $b$ is critical $\Rightarrow$ MSE method will be equivalent to Fisher's LD with appropriate $b$. 
Ho-Kashyap Procedures (1/2):

- Perceptron procedure finds separating plane if samples are linearly separable, but do not converge for non-separable problems
- MSE procedure yields a weight vector in both separable and non-separable cases, but there is no guarantee to have a separating plane even if the samples are linearly separable
- If margin vector, $b$, is chosen arbitrarily, all we can say $|Ya-b|$ is minimized, but for a linearly separable problem, all the elements of $b$ must be greater than zero; i.e. there exists $a'$ and $b'$ such that $Ya'=b'>0$

Ho-Kashyap Procedures (2/2):

- Minimize $|Ya-b|$ varying both $a$ and $b$ within the criterion function, $J_s(a,b) = \|Ya-b\|^2$
- In order to use a modified version of gradient descent procedure, find the gradients as $\nabla_a J_s(a,b) = 2Y'(Ya-b)$, $\nabla_b J_s(a,b) = -2(Ya-b)$
- For any value of $b$, $a=Y'b$, but for any value of $a$, the same is not true, since we have constraint $b>0$
- Algorithm: $b_1 > 0$ but arbitrary
  
  $a_k = Y^*b_k$
  $b_{k+1} = b_k - \rho_a \nabla_a J_s(a_k, b_k) = b_k + \rho_a (Ya_k-b_k)$ should always (+)

- For non-separable case, $Ya_k-b_k<0$ for all elements of $b$
Multi-category Generalizations (1/4)

- All the methods we have examined so far are proposed for two-class problems.
- *Linear Machine* approach can be utilized to generalize these algorithms to multi-category.
- If a linear machine exits that classifies all the samples correctly, these samples are called linearly separable.
- Assume the samples are linearly separable, then for $c$ classes there exists a set of weight vectors, $\hat{\alpha}_1, \ldots, \hat{\alpha}_c$ such that for $\hat{y} \in Y$, $\hat{\alpha}_i \cdot \hat{y} > \hat{\alpha}_j \cdot \hat{y}$ for all $i \neq j$.

Multi-category Generalizations (2/4)

- Multi-category problems can be reduced to two-class problems by.
- Assume sample $y$ belongs to class 1:
  
  \[ (\hat{\alpha}_i - \hat{\alpha}_j) \hat{y} > 0 \quad \text{for } j = 2, \ldots, c \]

  
  \[ \begin{bmatrix} \hat{\alpha}_1 \\ \hat{\alpha}_2 \\ \vdots \\ \hat{\alpha}_c \end{bmatrix} \text{ should classify } \eta_{i_2} = \begin{bmatrix} \hat{y} \\ \hat{y} \\ \vdots \\ 0 \end{bmatrix}, \ldots, \eta_{i_c} = \begin{bmatrix} \hat{y} \\ -\hat{y} \\ \vdots \\ -\hat{y} \end{bmatrix} \text{ correctly} \]

  \[ \Rightarrow \hat{\alpha} \cdot \eta_{i_j} > 0 \text{ for all } j \neq 1 \]
Multi-category Generalizations (3/4)

Fixed Increment Rule for multi-category problems:

Let \( \tilde{y}_k \in Y_i \) and \( \tilde{a}_i(k) \tilde{y}_k \leq \tilde{a}_j(k) \tilde{y}_k \) \( i \neq j \)

\[
\begin{align*}
\tilde{a}_i(k+1) &= \tilde{a}_i(k) + \tilde{y}_k \\
\tilde{a}_j(k+1) &= \tilde{a}_j(k) - \tilde{y}_k \\
\tilde{a}_l(k+1) &= \tilde{a}_l(k), \quad l \neq i \text{ and } l \neq j
\end{align*}
\]

\( \tilde{a}_i(k) > 0 \) \( \tilde{a}_i(k) < 0 \)

Angle between \( \tilde{a}_i \) and \( \tilde{y} \) should be \( < \pi/2 \)

- For linearly separable problems, it can be shown that fixed increment rule is guaranteed to converge

Multi-category Generalizations (4/4)

MSE for multi-category problems: \( \min ||Ya-b||^2 \)

Find \( \tilde{a} \) such that \( \tilde{a}_j \tilde{y} = 1 \) for all \( \tilde{y} \in Y_i \) and \( \tilde{a}_j \tilde{y} = 0 \) for all \( \tilde{y} \notin Y_i \)

Let \( A_{\text{vec}} = [\tilde{a}_1, \cdots, \tilde{a}_c] \), \( Y_{\text{vec}} = \begin{bmatrix} Y_{11} & \cdots & Y_{1n} \\ \vdots & \ddots & \vdots \\ Y_{c1} & \cdots & Y_{cn} \end{bmatrix} \), \( B_{\text{vec}} = \begin{bmatrix} B_1 \\ \vdots \\ B_c \end{bmatrix} \)

\( Y \) : samples labelled \( \omega \), \( B \) : all zeros except \( i \)th column

\[
A_{\text{vec}} = \begin{bmatrix} \tilde{a}_1 \\ \tilde{a}_2 \\ \vdots \\ \tilde{a}_c \end{bmatrix}, \quad Y_{\text{vec}} = \begin{bmatrix} \tilde{y}_{11} \\ \vdots \\ \tilde{y}_{1n} \\ \vdots \\ \tilde{y}_{cn} \\ \vdots \\ \tilde{y}_{c1} \\ \vdots \\ \tilde{y}_{cn} \end{bmatrix}, \quad B_{\text{vec}} = \begin{bmatrix} 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & 1 \end{bmatrix}
\]

\[
\min \langle YA-B \rangle (YA-B) \Rightarrow A = Y^B
\]
Support Vector Machine (SVM)

- SVM performs classification between two classes by finding a decision surface that is based on the most “informative” points of the training set.
- SVM differs from classical classifiers in the way that it handles the risk concept.
  - Empirical risk: minimize error on training data.
  - Structural risk: minimize probability of misclassifying future test data.
- SVM tries to maximize the margin between samples for different classes.

Support Vector Machine (SVM) is a state-of-the-art PR method with a remarkable classification performance.

Its performance success has been proved in many different areas:
- handwritten digit recognition,
- text classification,
- face recognition,
- object recognition,
- etc…
Assume the following is given:
- a training data set \( \{x_1, \ldots, x_n\} \), consisting of vectors
- their corresponding labels \( \{y_1, \ldots, y_n\} \), taking values +1 or −1.

LDF is defined
\[
g(\bar{x}_i) = \bar{w}^T \bar{x}_i + w_0 \quad i = 1, \ldots, n
\]

Decide
\[
y_i = +1 \text{ if } g(\bar{x}_i) \geq +1
\]
\[
y_i = -1 \text{ if } g(\bar{x}_i) \leq -1 \quad \Rightarrow \quad y_i \left( \bar{w}^T \bar{x}_i + w_0 \right) > +1 \quad i = 1, \ldots, n
\]
either class otherwise.
SVM: Formulation (2/6)

- **Def:** Optimal Separating Hyperplane (OSH) separates feature space, while maximizing the distance from the nearest point:
- **Def:** Support Vectors (SV) are the training patterns nearest to OSH, defining OSH
- SVs are the most difficult samples to classify
- SVs are the most informative for classification

SVM: Formulation (3/6)

- It is known that the distance of point $x_i$ from decision boundary is given as

$$ r = \frac{\tilde{w}' \tilde{x}_i + \tilde{w}_0}{\|\tilde{w}\|} $$

- Lets normalize $(\tilde{w}, \tilde{w}_0)$ so that distance for nearest point becomes $1/|\tilde{w}|$

$$ \frac{\tilde{w}'' \tilde{x}_i + \tilde{w}_0'}{\|\tilde{w}''\|} = \frac{1}{\|\tilde{w}'\|} $$
For a linearly separable problem, OSH can be obtained as a result of an optimization process by maximizing distance of samples closest to OSH:

\[
\max \frac{1}{\|\mathbf{w}\|} \quad \text{or} \quad \min \|\mathbf{w}\|^2
\]

subject to \( y_i (\mathbf{w}^T \mathbf{x}_i + w_0) \geq 1 \quad i = 1, \ldots, n \)

This problem can be solved by using the method of Lagrange multipliers:

\[
\min_{\mathbf{w}, w_0, \alpha_i} \left\{ \mathbf{w}^T \mathbf{w} - \sum_{i=1}^n \alpha_i (y_i (\mathbf{w}^T \mathbf{x}_i + w_0) - 1) \right\}
\]

\[\Rightarrow \mathbf{w}^* = \sum_{i=1}^n \alpha_i^* y_i \mathbf{x}_i, \text{ where } \alpha_i^* \text{ nonzero for only } \text{SV's}\]

If the problem is non-separable:

\[
\min \left\{ \|\mathbf{w}\|^2 + C \sum_i \xi_i \right\} \quad \xi_i \geq 0 \quad C: \text{trade-off parameter}
\]

subject to \( y_i (\mathbf{w}^*^T \mathbf{x}_i + w_0') \geq 1 - \xi_i \quad i = 1, \ldots, n \)

\( \xi_i \) is used to compensate for misclassified samples.

\( C \) gives a compromise between distance of nearest point and data.

The non-separable problem can be similarly solved by the method of Lagrange multipliers:

\[
\min_{\mathbf{w}, w_0, \alpha_i, \xi_i} \left\{ \mathbf{w}^*^T \mathbf{w} + C \sum_i \xi_i - \sum_{i=1}^n \alpha_i (y_i (\mathbf{w}^*^T \mathbf{x}_i + w_0') - 1 + \xi_i) \right\}
\]
SVM: Formulation (6/6)

\[
\begin{align*}
\min \left\{ \|w\|^2 + C \sum \xi_i \right\} & \quad \xi_i \geq 0 \quad C: \text{trade-off parameter} \\
\end{align*}
\]

Linear separability assumption can especially be useful after projecting feature vectors into higher dimensional feature spaces by mapping functions, \( \Phi \):

\[
\vec{x} \rightarrow \Phi(\vec{x}) \quad \Phi: \mathbb{R}^n \rightarrow \mathbb{R}^m
\]

- Define a LDF, \( f(\vec{x}) = \vec{w} \cdot \Phi(\vec{x}) \)
  where \( \vec{w} = \sum_{i=1}^n \alpha_i \Phi(\vec{x}_i) \) and \( \alpha_i \)'s non-zero for only SV's obtained as a solution
- Define a kernel, \( K \), in terms of the mappings, \( \Phi \):
  \[
  K(\vec{x}_i, \vec{x}) = \Phi(\vec{x}_i) \cdot \Phi(\vec{x}) \Rightarrow f(\vec{x}) = \sum_{i=1}^n \alpha_i K(\vec{x}_i, \vec{x})
  \]
**SVM : Nonlinear Kernels (2/2)**

- Without having full information for $\Phi$, $K$ can still be utilized, as long as $K$ is positive, symmetric and continuous (Mercer’s theorem).

- Kernel, $K$, is usually chosen as one of the following
  
  $K(\tilde{x}_i, \tilde{x}) = (\tilde{x}_i \cdot \tilde{x} + 1)^d$ (polynomial type)
  
  $K(\tilde{x}_i, \tilde{x}) = e^{-\frac{(\tilde{x}_i - \tilde{x})^2}{2\sigma^2}}$ (radial-basis style)
  
  $K(\tilde{x}_i, \tilde{x}) = \tanh(\kappa \tilde{x}_i \cdot \tilde{x} - \delta)$ (neural net type)

**Conclusions**

- Choosing a separating plane maximizing the margin between samples
- Using kernels for reaching higher dimensions
- Generally avoiding over-fitting problem
- Expensive to apply for multiclass problems
Final Words on LDF ...

- There are various procedures for determining LDF
- However, none of them requires knowledge of the forms of the underlying probability distributions, but they only require the sample data → nonparametric method
- Main idea is classification of data in multi-D space either using projection to a less-dimensioned space or dividing the space with hyperplanes