Henrik I Christensen

Robotics & Intelligent Machines © GT
Georgia Institute of Technology,
Atlanta, GA 30332-0280
hic@cc.gatech.edu
Outline

1. Introduction
2. Linear Discriminant Functions
3. LSQ for Classification
4. Fisher’s Discriminant Method
5. Perceptrons
6. Summary
Introduction

- Last time: prediction of new functional values
- Today: linear classification of data
  - Basic pattern recognition
  - Separation of data: buy/sell
  - Segmentation of line data, ...
Simple Example - Bolts or Needles

![Graph showing the distribution of length and head diameter for bolts and needles.](image-url)
Classification

- Given
  - An input vector: $X$
  - A set of classes: $c_i \in C$, $i = 1, \ldots, k$
- Mapping $m : X \rightarrow C$
- Separation of space into decision regions
- Boundaries termed decision boundaries/surfaces
Basis Formulation

- It is a 1-of-K coding problem
- Target vector: \( \mathbf{t} = (0, \ldots, 1, \ldots, 0) \)
- Consideration of 3 different approaches
  1. Optimization of discriminant function
  2. Bayesian Formulation: \( p(c_i | x) \)
  3. Learning & Decision fusion
There are data sets and sample code available

- NETLAB: http://www.ncrg.aston.ac.uk/netlab/index.php
- NAVTOOLBOX: http://www.cas.kth.se/toolbox/
- SLAM Dataset: http://kaspar.informatik.uni-freiburg.de/~slamEvaluation/datasets.php
Outline

1. Introduction
2. Linear Discriminant Functions
3. LSQ for Classification
4. Fisher’s Discriminant Method
5. Perceptrons
6. Summary
Discriminant Functions

- Objective: input vector $x$ assigned to a class $c_i$
- Simple formulation:
  $$y(x) = w^T x + w_0$$
- $w$ is termed a weight vector
- $w_0$ is termed a bias
- Two class example: $c_1$ if $y(x) \geq 0$ otherwise $c_2$
Basic Design

- Two points on decision surface $x_a$ and $x_b$
- $y(x_a) = y(x_b) = 0 \Rightarrow w^T(x_a - x_b) = 0$
- $w$ perpendicular to decision surface

\[
\frac{w^T x}{||w||} = - \frac{w_0}{||w||}
\]

- Define: $\tilde{w} = (w_0, w)$ and $\tilde{x} = (1, x)$ so that:

\[
y(x) = \tilde{w}^T \tilde{x}
\]
Linear discriminant function

\[ y > 0 \]
\[ y = 0 \]
\[ y < 0 \]

\[ R_1 \]
\[ R_2 \]
Multi Class Discrimination

- Generation of multiple decision functions

\[ y_k(x) = w_k^T x + w_{k0} \]

- Decision strategy

\[ j = \arg \max_{i \in 1..k} y_i(x) \]
Multi-Class Decision Regions

\[ R_i \]
\[ R_j \]
\[ R_k \]
\[ x_A \]
\[ \hat{x} \]
\[ x_B \]
Example - Bolts or Needles

![Graph showing head diameter vs. length for bolts and needles.]

Henrik I Christensen (RIM@GT)
Minimum distance classification

- Suppose we have computed the mean value for each of the classes
- \( m_{\text{needle}} = [0.86, 2.34]^T \) and \( m_{\text{bolt}} = [5.74, 5, 85]^T \)
- We can then compute the minimum distance
  \[
d_j(x) = ||x - m_j||
  \]
- \( \text{argmin}_i d_i(x) \) is the best fit
- Decision functions can be derived
Bolts / Needle Decision Functions

Needle \[ d_{\text{needle}}(x) = 0.86x_1 + 2.34x_2 - 3.10 \]

Bolt \[ d_{\text{bolt}}(x) = 5.74x_1 + 5.85x_2 - 33.59 \]

Decision boundary

\[ d_i(x) - d_j(x) = 0 \]

\[ d_{\text{needle/bolt}}(x) = -4.88x_1 - 3.51x_2 + 30.49 \]
Example decision surface
Outline

1. Introduction
2. Linear Discriminant Functions
3. LSQ for Classification
4. Fisher’s Discriminant Method
5. Perceptrons
6. Summary
Just like we could do LSQ for regression we can perform an approximation to the classification vector \( C \)

Consider again

\[
y_k(x) = \mathbf{w}_k^T \mathbf{x} + w_{k0}
\]

Rewrite to

\[
y(x) = \mathbf{\tilde{W}}^T \mathbf{\tilde{x}}
\]

Assuming we have a target vector \( \mathbf{T} \)
Least Squares for Classification

- The error is then:

\[ E_D(\tilde{W}) = \frac{1}{2} Tr \left\{ (\tilde{X}\tilde{W} - T)^T (\tilde{X}\tilde{W} - T) \right\} \]

- The solution is then

\[ \tilde{W} = \left( \tilde{X}^T \tilde{X} \right)^{-1} \tilde{X}^T T \]
LSQ and Outliers

- Two sets of data points are plotted on separate graphs.
- The left graph shows a linear relationship with outliers affecting the LSQ fit.
- The right graph demonstrates a linear model with outliers present, emphasizing the impact on the LSQ and how it handles outliers differently.

Henrik I Christensen (RIM@GT)
Outline

1. Introduction
2. Linear Discriminant Functions
3. LSQ for Classification
4. Fisher’s Discriminant Method
5. Perceptrons
6. Summary
Fisher’s linear discriminant

- Selection of a decision function that maximizes distance between classes
- Assume for a start
  \[ y = W^T x \]
- Compute \( m_1 \) and \( m_2 \)
  \[ m_1 = \frac{1}{N_1} \sum_{i \in C_1} x_i \quad m_2 = \frac{1}{N_2} \sum_{j \in C_2} x_j \]
- Distance:
  \[ m_2 - m_1 = w^T (m_2 - m_1) \]
- where \( m_i = wm_i \)
The suboptimal solution
The Fisher criterion

- Consider the expression

\[ J(w) = \frac{w^T S_B w}{w^T S_W w} \]

where \( S_B \) is the between class covariance and \( S_W \) is the within class covariance, i.e.

\[ S_B = (m_1 - m_2)(m_1 - m_2)^T \]

and

\[ S_W = \sum_{i=C_1} (x_i - m_1)(x_i - m_1)^T + \sum_{i=C_2} (x_i - m_2)(x_i - m_2)^T \]

- Optimized when

\[ (w^T S_B w)S_W w = (w^T S_W w)S_B w \]

or

\[ w \propto S_W^{-1}(m_2 - m_1) \]
The Fisher result
Generalization to $N \geq 2$

- Define a stacked weight factor

$$y = W^T x$$

- The within class covariance generalizes to

$$S_w = \sum_{k=1}^{K} S_k$$

- The between class covariance is

$$S_B = \sum_{k=1}^{K} N_k (m_k - m)(m_k - m)^T$$

- It can be shown that $J(w)$ is optimized by the eigenvectors to the equation

$$S = S_w^{-1} S_B$$
Outline

1. Introduction
2. Linear Discriminant Functions
3. LSQ for Classification
4. Fisher’s Discriminant Method
5. Perceptrons
6. Summary
Perceptron Algorithm

- Developed by Rosenblatt (1962)
- Formed an important basis for neural networks
- Use a non-linear transformation $\phi(x)$
- Construct a decision function

$$y(x) = f \left( w^T \phi(x) \right)$$

- where

$$f(a) = \begin{cases} 
+1, & a \geq 0 \\
-1, & a < 0 
\end{cases}$$
The perceptron criterion

- Normally we want
  \[ w^T \phi(x_n) > 0 \]

- Given the target vector definition
  \[ E_p(w) = - \sum_{n \text{ in } M} w^T \phi_n t_n \]

- Where \( M \) represents all the mis-classified samples

- We can make this a gradient descent as seen in last lecture
  \[ w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E_p(w) = w^{(\tau)} + \eta \phi_n t_n \]
Perceptron learning example
Summary

- Basics for discrimination / classification
- Obviously not all problems are linear
- Optimization of the distance/overlap between classes
  - Minimizing the probability of error classification
- Basic formulation as an optimization problem
- How to optimize between cluster distance? Covariance Weighted
- Basic recursive formulation
- Could we make it more robust?