Classification

Discriminant Analysis

slides thanks to Greg Shakhnarovich (CS195-5, Brown Univ., 2006)
Distribution in 1D projection

- Consider a scalar projection

\[ f : \mathbf{x} \rightarrow w_0 + \mathbf{w}^T \mathbf{x} \]

- We can study how well the projected values corresponding to different classes are separated
  - This is a function of \( \mathbf{w} \); some projections may be better than others.
Distribution in 1D projection

- Consider a scalar projection

\[ f : x \rightarrow w_0 + w^T x \]

- We can study how well the projected values corresponding to different classes are separated
  - This is a function of \( w \); some projections may be better than others.
Linear discriminant and dimensionality reduction

The *discriminant function* \( f(x; w) = w_0 + w^T x \) reduces the dimension of examples from \( d \) to 1; the components orthogonal to \( w \) become irrelevant.

\[ f(x, w) = 0 \]
\[ f(x, w) = -1 \]
\[ f(x, w) = +1 \]

\[ \hat{y} = +1 \Leftrightarrow f(x; w) > 0 \]
Projections and classification

What objective are we optimizing the 1D projection for?
We want to minimize “overlap” between projections of the two classes.

One approach: make the class projections a) compact, b) far apart.

An obvious idea: maximize separation between the projected means.
Separation of the means

- $N_{+1}$ examples of class $+1$, $N_{-1}$ examples of class $-1$.

- The *empirical mean* of each class:
  
  $$m_{+1} = \frac{1}{N_{+1}} \sum_{y_i=+1} x_i, \quad m_{-1} = \frac{1}{N_{-1}} \sum_{y_i=-1} x_i$$

- We can look for projection $\hat{w}$ such that

  $$\hat{w} = \arg\max_w w^T (m_{+1} - m_{-1})$$
Separation of the means: example

$$\hat{w} = \arg\max_{\|w\|=1} w^T (m_+ - m_-)$$

- Also want to make projection of each class “compact”...
Fisher’s linear discriminant analysis

- Criterion to be maximized:

\[ J_{\text{Fisher}}(w) = \frac{\text{separation between projected means}^2}{\text{sum of projected within-class variances}} \]

- Numerator: *between-class scatter* \( (w^T(m_{+1} - m_{-1}))^2 \)

- Denominator: *within-class scatter* \( w^T(N_{-1}S_{-1} + N_{+1}S_{+1})w \), where

\[ S_c = \frac{1}{N_c} \sum_{y_i = c} (x_i - m_c)(x_i - m_c)^T. \]

- The denominator is the sum of estimated 1D class covariances, after data are projected to \( w \), weighted by number of samples in each class.
Fisher’s LDA

\[ J_{Fisher}(w) = \frac{(w^T(m_{+1} - m_{-1}))^2}{w^T(N_{-1}S_{-1} + N_{+1}S_{+1})w} \]

- Best 1D projection: \( \hat{w} = \arg\max_w J_{Fisher}(w) \)

- Setting the derivative of \( J \) w.r.t. \( w \) to zero, get solution:

\[ \hat{w} \propto (N_{-1}S_{-1} + N_{+1}S_{+1})^{-1}(m_{+1} - m_{-1}) \]

**Notation:** \( \propto \) means “proportional to”, up to a constant factor.
Example of applying Fisher’s LDA

maximize separation of means

maximize Fisher’s LDA criterion
→ better class separation
Using LDA for classification in one dimension

- Fisher’s LDA gives an optimal choice of $w$, the vector for projection down to one dimension.
- For classification, we still need to select a threshold to compare projected values to. Two possibilities:
  - No explicit probabilistic assumptions. Find threshold which minimizes empirical classification error.
  - Make assumptions about data distributions of the classes, and derive theoretically optimal decision boundary.
    - Usual choice for class distributions is multivariate Gaussian.
    - We also will need a bit of decision theory.
To minimize classification error:

\[ \hat{y} = \arg \max_{C} p(C \mid \mathbf{x}) \approx \]

At a given point \( \mathbf{x} \) in feature space, choose as the predicted class the class that has the greatest probability at \( \mathbf{x} \).
Decision theory

\[ \hat{y} = \arg \max_C p(C \mid x) \approx \]

At a given point \( x \) in feature space, choose as the predicted class the class that has the greatest probability at \( x \).
MATLAB interlude

Classification via discriminant analysis, using the classify() function.
Data for each class modeled as multivariate Gaussian.

matlab_demo_06.m

class = classify( sample, training, group, 'type' )

predicted test labels  test data  training data  training labels  model for class covariances
MATLAB classify() function

Models for class covariances

- **'linear'**: all classes have same covariance matrix → linear decision boundary
- **'diaglinear'**: all classes have same diagonal covariance matrix → linear decision boundary
- **'quadratic'**: classes have different covariance matrices → quadratic decision boundary
- **'diagquadratic'**: classes have different diagonal covariance matrices → quadratic decision boundary
MATLAB classify() function

Example with ‘quadratic’ model of class covariances
Relative class probabilities for LDA

'linear':
all classes have same covariance matrix
→ linear decision boundary

relative class probabilities have exactly same sigmoidal form as in logistic regression