Recap: Linear Discriminant Functions

- **Basic idea**
  - Directly encode decision boundary
  - Minimize misclassification probability directly.

- **Linear discriminant functions**
  
  \[ y(x) = w^T x + w_0 \]

  - \( w, w_0 \) define a hyperplane in \( \mathbb{R}^d \).
  - If a dataset can be perfectly classified by a linear discriminant, then we call it linearly separable.

Recap: Least-Squares Classification

- **Simplest approach**
  - Directly try to minimize the sum-of-squares error
    
    \[ E(w) = \sum_{i=1}^{N} (y(x_i;w) - t_i)^2 \]

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

  - Setting the derivative to zero yields
    \[ W = (X^T X)^{-1} X^T T = \frac{1}{N} X^T T \]

  - We then obtain the discriminant function as
    \[ y(x) = W^T x = T^T \left( \frac{1}{N} X \right)^T x \]

  \( \Rightarrow \) Exact, closed-form solution for the discriminant function parameters.

Recap: Generalized Linear Models

- **Generalized linear model**
  \[ y(x) = g(w^T x + w_0) \]

  - \( g(\cdot) \) is called an activation function and may be nonlinear.
  - The decision surfaces correspond to
    \[ y(x) = \text{const.} \iff w^T x + w_0 = \text{const.} \]

  - If \( g \) is monotonous (which is typically the case), the resulting decision boundaries are still linear functions of \( x \).

- **Advantages of the non-linearity**
  - Can be used to bound the influence of outliers and “too correct” data points.
  - When using a sigmoid for \( g(\cdot) \), we can interpret \( y(x) \) as posterior probabilities.
Recap: Linear Separability

- Up to now: restrictive assumption
  - Only consider linear decision boundaries

- Classical counterexample: XOR

\[ x_2 \]
\[ C_2 \]
\[ C_1 \]
\[ C_1 \]
\[ C_2 \]

 Linear Separability

- Even if the data is not linearly separable, a linear decision boundary may still be “optimal”.
  - Generalization
  - E.g. in the case of Normal distributed data (with equal covariance matrices)

- Choice of the right discriminant function is important and should be based on
  - Prior knowledge (of the general functional form)
  - Empirical comparison of alternative models
  - Linear discriminants are often used as benchmark.

Generalized Linear Discriminants

- Generalization
  - Transform vector \( x \) with \( M \) nonlinear basis functions \( \phi_j(x) \):
    \[ y_k(x) = \sum_{j=1}^{M} w_{kj} \phi_j(x) + w_k0 \]
  - Purpose of \( \phi_j(x) \): basis functions
  - Allow non-linear decision boundaries.
  - By choosing the right \( \phi_j \), every continuous function can (in principle) be approximated with arbitrary accuracy.

- Notation
  \( y_k(x) = \sum_{j=0}^{M} w_{kj} \phi_j(x) \quad \text{with} \quad \phi_0(x) = 1 \)

Generalized Linear Discriminants

- Model
  \[ y_k(x) = \sum_{j=1}^{M} w_{kj} \phi_j(x) = y_k(x; w) \]
  - \( K \) functions (outputs) \( y_k(x; w) \)
  - Learning in Neural Networks
    - Single-layer networks: \( \phi_j \) are fixed, only weights \( w \) are learned.
    - Multi-layer networks: both the \( w \) and the \( \phi_j \) are learned.

  - In the following, we will not go into details about neural networks in particular, but consider generalized linear discriminants in general.

Gradient Descent

- Learning the weights \( w \):
  - \( N \) training data points: \( X = \{x_1, ..., x_N\} \)
  - \( K \) outputs of decision functions: \( y_k(x; w) \)
  - Target vector for each data point: \( T = \{t_1, ..., t_N\} \)

- Error function (least-squares error) of linear model
  \[ E(w) = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} (y_k(x_n; w) - t_{kn})^2 \]
  \[ = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} \left( \sum_{j=1}^{M} w_{kj} \phi_j(x_n) - t_{kn} \right)^2 \]

Gradient Descent

- Problem
  - The error function can in general no longer be minimized in closed form.

- Idea (Gradient Descent)
  - Iterative minimization
  - Start with an initial guess for the parameter values \( w_{kj}^{(0)} \)
  - Move towards a (local) minimum by following the gradient.
    \[ w_{kj}^{(t+1)} = w_{kj}^{(t)} - \eta \frac{\partial E(w)}{\partial w_{kj}} \bigg|_{w^{(t)}} \] 
    \( \eta \): Learning rate

  - This simple scheme corresponds to a 1st-order Taylor expansion (There are more complex procedures available).
**Gradient Descent - Basic Strategies**

- **“Batch learning”**
  \[
  w_{kj}^{(r+1)} = w_{kj}^{(r)} - \eta \frac{\partial E(w)}{\partial w_{kj}} 
  \]
  \(\eta\): Learning rate

  - Compute the gradient based on all training data:
  \[
  \frac{\partial E(w)}{\partial w_{kj}}
  \]

- **“Sequential updating”**
  \[
  w_{kj}^{(r+1)} = w_{kj}^{(r)} - \eta \frac{\partial E_n(w)}{\partial w_{kj}} 
  \]
  \(\eta\): Learning rate

  - Compute the gradient based on a single data point at a time:
  \[
  \frac{\partial E_n(w)}{\partial w_{kj}}
  \]

**Gradient Descent**

- **Error function**
  \[
  E(w) = \sum_{n=1}^{N} E_n(w) = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} \left( \sum_{j=1}^{M} w_{kj} \phi_j(x_n) - t_{kn} \right)^2
  \]

  \[
  E_n(w) = \frac{1}{2} \sum_{k=1}^{K} \left( \sum_{j=1}^{M} w_{kj} \phi_j(x_n) - t_{kn} \right)^2
  \]

  \[
  \frac{\partial E_n(w)}{\partial w_{kj}} = \left( \sum_{j=1}^{M} w_{kj} \phi_j(x_n) - t_{kn} \right) \phi_j(x_n)
  \]

  \[
  = (y_k(x_n; w) - t_{kn}) \phi_j(x_n)
  \]

**Gradient Descent**

- **Cases with differentiable, non-linear activation function**
  \[
  y_k(x) = g(a_k) = g \left( \sum_{j=0}^{M} w_{kj} \phi_j(x_n) \right)
  \]

- **Gradient descent**
  \[
  \frac{\partial E_n(w)}{\partial w_{kj}} = \frac{\partial g(a_k)}{\partial w_{kj}} (y_k(x_n; w) - t_{kn}) \phi_j(x_n)
  \]

  \[
  \frac{w_{kj}^{(r+1)}}{w_{kj}^{(r)}} = w_{kj}^{(r)} - \eta \frac{\partial E_n(w)}{\partial w_{kj}} 
  \]  \(\eta\): Learning rate

  \[
  \delta_{kn} = \frac{\partial g(a_k)}{\partial w_{kj}} (y_k(x_n; w) - t_{kn})
  \]

**Summary: Generalized Linear Discriminants**

- **Properties**
  - General class of decision functions.
  - Nonlinearity \(g(\cdot)\) and basis functions \(\phi_j\) allow us to address linearly non-separable problems.
  - Shown simple sequential learning approach for parameter estimation using gradient descent.
  - Better 2nd order gradient descent approaches available (e.g., Newton-Raphson).

- **Limitations / Caveats**
  - Flexibility of model is limited by curse of dimensionality
    - \(g(\cdot)\) and \(\phi_j\) often introduce additional parameters.
    - Models are either limited to lower-dimensional input space or need to share parameters.
  - Linearly separable case often leads to overfitting.
    - Several possible parameter choices minimize training error.
Topics of This Lecture

- Fisher’s linear discriminant (FLD)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications
- Logistic Regression
  - Probabilistic discriminative models
  - Logistic sigmoid (logit function)
  - Cross-entropy error
  - Gradient descent
  - Iteratively Reweighted Least Squares
- Note on Error Functions

Classification as Dimensionality Reduction

- Classification as dimensionality reduction
  - We can interpret the linear classification model as a projection onto a lower-dimensional space.
  - E.g., take the $J$-dimensional input vector $x$ and project it down to one dimension by applying the function $y = w^T x$
  - If we now place a threshold at $y \geq -w_0$, we obtain our standard two-class linear classifier.
  - The classifier will have a lower error the better this projection separates the two classes.

$\Rightarrow$ New interpretation of the learning problem

- Try to find the projection vector $w$ that maximizes the class separation.

Classification as Dimensionality Reduction

- Measuring class separation
  - We could simply measure the cross class variance to the within class variance:

$$J(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2}$$

- Usually, this is written as

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

- where

$S_B = (m_2 - m_1)(m_2 - m_1)^T$

$S_W = \sum_{k=1}^{2} \sum_{n \in C_k} (x_n - m_k)(x_n - m_k)^T$

- This expression can be made arbitrarily large by increasing $\|w\|$. We need to enforce additional constraint $\|w\| = 1$.

- This constrained minimization results in

$w \propto (m_2 - m_1)$

- Problems with this approach
  1. This expression can be made arbitrarily large by increasing $\|w\|$.
     $\Rightarrow$ Need to enforce additional constraint $\|w\| = 1$.
     $\Rightarrow$ This constrained minimization results in $w \propto (m_2 - m_1)$
  2. The criterion may result in bad separation if the clusters have elongated shapes.

Fisher’s Linear Discriminant Analysis (FLD)

- Better idea:
  - Find a projection that maximizes the ratio of the between-class scatter matrix to the within-class scatter matrix:

$$J(w) = \frac{(m_2 - m_1)^2}{s_b^2} = \sum_{k=1}^{2} (y_k - m_k)^2$$

- Usually, this is written as

$$J(w) = w^T S_B w / w^T S_W w$$

- where

$S_B = (m_2 - m_1)(m_2 - m_1)^T$

$S_W = \sum_{k=1}^{2} \sum_{n \in C_k} (x_n - m_k)(x_n - m_k)^T$

- This constrained minimization results in

$$w \propto (m_2 - m_1)$$

- The optimal solution for $w$ can be obtained as:

$w \propto S_W^{-1}(m_2 - m_1)$

- Classification function:

$$y(x) = w^T x + w_0$$

$\Rightarrow$ Classification as dimensionality reduction

- We can interpret the linear classification model as a projection onto a lower-dimensional space.

- E.g., take the $J$-dimensional input vector $x$ and project it down to one dimension by applying the function $y = w^T x$

- If we now place a threshold at $y \geq -w_0$, we obtain our standard two-class linear classifier.

- The classifier will have a lower error the better this projection separates the two classes.

$\Rightarrow$ New interpretation of the learning problem

- Try to find the projection vector $w$ that maximizes the class separation.
Multiple Discriminant Analysis

- Generalization to \( K \) classes
  \[
  J(W) = \frac{W^T S_B W}{W^T S_W W}
  \]
  where
  \[
  W = [w_1, \ldots, w_K] \quad m = \frac{1}{N} \sum_{n=1}^{N} x_n = \frac{1}{N} \sum_{k=1}^{K} N_k m_k
  \]
  \[
  S_B = \sum_{k=1}^{K} N_k (m_k - m)(m_k - m)^T
  \]
  \[
  S_W = \sum_{k=1}^{K} \sum_{n \in C_k} (x_n - m_k)(x_n - m_k)^T
  \]

Maximizing \( J(W) \)

- "Rayleigh quotient" \( \Rightarrow \) Generalized eigenvalue problem
  \[
  J(W) = \frac{W^T S_B W}{W^T S_W W}
  \]
  \[
  \Rightarrow \text{The columns of the optimal } W \text{ are the eigenvectors corresponding to the largest eigenvalues of }\]
  \[
  S_B W_i = \lambda_i S_W W_i
  \]
  \[
  \Rightarrow \text{Defining } V = S_B^{1/2} W \text{, we get }\]
  \[
  S_B^{1/2} S_W^{1/2} V = \lambda V
  \]
  which is a regular eigenvalue problem.
  \[
  \Rightarrow \text{Solve to get eigenvectors of } V \text{, then from that of } W.
  \]
- For the \( K \)-class case we obtain (at most) \( K-1 \) projections,
  (i.e. eigenvectors corresponding to non-zero eigenvalues.)

What Does It Mean?

- What does it mean to apply a linear classifier?
  \[
  y(x) = W^T x
  \]
  Weight vector  Input vector

- Classifier interpretation
  - The weight vector has the same dimensionality as \( x \).
  - Positive contributions where \( \text{sign}(x_i) = \text{sign}(w_i) \).
  - The weight vector identifies which input dimensions are important for positive or negative classification (large \( |w_i| \)) and which ones are irrelevant (near-zero \( w_i \)).
  - If the inputs \( x \) are normalized, we can interpret \( w \) as a "template" vector that the classifier tries to match.
  \[
  W^T x = |w||x| \cos \theta
  \]

Example Application: Fisherfaces

- Visual discrimination task
  \[
  C_1: \text{Subjects with glasses}
  \]
  \[
  C_2: \text{Subjects without glasses}
  \]
  - Training data:
  - Test:
    \[
    - \text{glasses?}
    \]
    Take each image as a vector of pixel values and apply FLD...

Fisherfaces: Interpretability

- Resulting weight vector for "Glasses/NoGlasses"

Summary: Fisher's Linear Discriminant

- Properties
  - Simple method for dimensionality reduction, preserves class discriminability.
  - Can use parametric methods in reduced-dim. space that might not be feasible in original higher-dim. space.
  - Widely used in practical applications.

- Restrictions / Caveats
  - Not possible to get more than \( K-1 \) projections.
  - FLD reduces the computation to class means and covariances.
  \[
  \Rightarrow \text{Implicit assumption that class distributions are unimodal and well-approximated by a Gaussian/hyperellipsoid.}
  \]
Fisher’s linear discriminant (FLD)
- Classification as dimensionality reduction
- Linear discriminant analysis
- Multiple discriminant analysis
- Applications

Logistic Regression
- Probabilistic discriminative models
- Logistic sigmoid (logit function)
- Cross-entropy error
- Gradient descent
- Iteratively Reweighted Least Squares

Note on Error Functions

Topics of This Lecture

Probabilistic Discriminative Models

- We have seen that we can write
  \[ p(C_1|x) = \sigma(a) \]
  \[ = \frac{1}{1 + \exp(-a)} \]
- We can obtain the familiar probabilistic model by setting
  \[ a = \ln \frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)} \]
- Or we can use generalized linear discriminant models
  \[ a = w^T \phi(x) \]

Comparison

- Let’s look at the number of parameters...
  - Assume we have an \( M \)-dimensional feature space \( \phi \).
  - And assume we represent \( p(\phi(C_1)) \) and \( p(\phi(C_2)) \) by Gaussians.
  - How many parameters do we need?
    - For the means: \( 2M \)
    - For the covariances: \( M(M+1)/2 \)
    - Together with the class priors, this gives \( M(M+5)/2 + 1 \) parameters!
  - How many parameters do we need for logistic regression?
    - \( p(C_1|\phi) = y(\phi) = \sigma(w^T \phi) \)
    - Just the values of \( w \rightarrow M \) parameters!

\( \Rightarrow \) For large \( M \), logistic regression has clear advantages!

Logistic Sigmoid

- Definition:
  \[ \sigma(a) = \frac{1}{1 + \exp(-a)} \]
- Inverse:
  \[ a = \ln \left( \frac{\sigma}{1 - \sigma} \right) \]
- Symmetry property:
  \[ \sigma(-a) = 1 - \sigma(a) \]
- Derivative:
  \[ \frac{d\sigma}{da} = \sigma(1 - \sigma) \]

Logistic Regression

- Let’s consider a data set \( \{ \phi_n, t_n \} \) with \( n = 1, \ldots, N \), where \( \phi_n = \phi(x_n) \) and \( t_n \in \{0,1\} \), \( t = (t_1, \ldots, t_N)^T \).
- With \( y_n = p(C_1|\phi_n) \), we can write the likelihood as
  \[ p(t|w) = \prod_{n=1}^{N} y_n^{t_n} (1 - y_n)^{1-t_n} \]
- Define the error function as the negative log-likelihood
  \[ E(w) = - \ln p(t|w) \]
  \[ = - \sum_{n=1}^{N} \{ t_n \ln y_n + (1-t_n) \ln(1-y_n) \} \]
- This is the so-called cross-entropy error function.
Gradient of the Error Function

• Error function
\[ E(w) = -\sum_{n=1}^{N} \{ t_n \ln y_n + (1 - t_n) \ln(1 - y_n) \} \]
\[ \frac{\partial E}{\partial w} = y_n(1 - y_n) \phi_n \]

• Gradient
\[ \nabla E(w) = -\sum_{n=1}^{N} \left\{ t_n \frac{\partial}{\partial y_n} \ln y_n + (1 - t_n) \frac{\partial}{\partial y_n} \ln(1 - y_n) \right\} \]
\[ = -\sum_{n=1}^{N} \left\{ t_n \frac{y_n(1 - y_n)}{y_n} \phi_n - (1 - t_n) \frac{y_n(1 - y_n)}{1 - y_n} \phi_n \right\} \]
\[ = -\sum_{n=1}^{N} \left\{ \left(t_n - 4y_n^2 - y_n + 3y_n \phi_n \right) \phi_n \right\} \]
\[ = \sum_{n=1}^{N} (y_n - t_n) \phi_n \]

A More Efficient Iterative Method...

• Second-order Newton-Raphson gradient descent scheme
\[ w^{(r+1)} = w^{(r)} - H^{-1} \nabla E(w) \]
where \( H = \nabla^2 E(w) \) is the Hessian matrix, i.e. the matrix of second derivatives.

• Properties
  - Local quadratic approximation to the log-likelihood.
  - Faster convergence.

Newton-Raphson for Logistic Regression

• Now, let’s try Newton-Raphson on the cross-entropy error function:
\[ E(w) = -\sum_{n=1}^{N} \{ t_n \ln y_n + (1 - t_n) \ln(1 - y_n) \} \]
\[ \frac{\partial E}{\partial w} = y_n(1 - y_n) \phi_n \]
\[ \nabla E(w) = \sum_{n=1}^{N} (y_n - t_n) \phi_n = \Phi^T(y - t) \]
\[ H = \nabla^2 E(w) = \sum_{n=1}^{N} \phi_n \phi_n^T = \Phi^T \Phi \]
where \( \Phi \) is an \( N \times 1 \) vector of features.
\[ \Rightarrow \text{The Hessian is no longer constant, but depends on } w \text{ through the weighting matrix } R. \]

Newton-Raphson for Least-Squares Estimation

• Let’s first apply Newton-Raphson to the least-squares error function:
\[ E(w) = \frac{1}{2} \sum_{n=1}^{N} \left( w^T \phi_n - t_n \right)^2 \]
\[ \nabla E(w) = \sum_{n=1}^{N} (w^T \phi_n - t_n) \phi_n = \Phi^T \phi w - \Phi^T t \]
\[ H = \nabla^2 E(w) = \sum_{n=1}^{N} \phi_n \phi_n^T = \Phi^T \Phi \quad \text{where } \Phi = \begin{bmatrix} \phi_1^T \\ \vdots \\ \phi_N^T \end{bmatrix} \]
• Resulting update scheme:
\[ w^{(r+1)} = w^{(r)} - (\Phi^T \Phi)^{-1} \Phi^T (\Phi w^{(r)} - \Phi^T t) \]
\[ = (\Phi^T \Phi)^{-1} \Phi^T t \quad \text{Closed-form solution!} \]

Iteratively Reweighted Least Squares

• Update equations
\[ w^{(r+1)} = w^{(r)} - (\Phi^T R \Phi)^{-1} \Phi^T (y - t) \]
\[ = (\Phi^T R \Phi)^{-1} \left\{ \Phi^T R \Phi w^{(r)} - \Phi^T t \right\} \]
\[ = (\Phi^T R \Phi)^{-1} \Phi^T R z \]
with \( z = \Phi w^{(r)} - R^{-1} (y - t) \)
• Again very similar form (normal equations)
  - But now with non-constant weighting matrix \( R \) (depends on \( w \)).
  - Need to apply normal equations iteratively.
\[ \Rightarrow \text{Iteratively Reweighted Least-Squares (IRLS)} \]
Summary: Logistic Regression

- Properties
  - Directly represent posterior distribution \( p(\phi | C_k) \)
  - Requires fewer parameters than modeling the likelihood + prior.
  - Very often used in statistics.
  - It can be shown that the cross-entropy error function is concave
  - Optimization leads to unique minimum
  - But no closed-form solution exists
  - Iterative optimization (IRLS)
  - Both online and batch optimizations exist
  - There is a multi-class version described in (Bishop Ch.4.3.4).
- Caveat
  - Logistic regression tends to systematically overestimate odds ratios when the sample size is less than ~500.

Topics of This Lecture

- Fisher’s Linear Discriminant (FLD)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications
- Logistic Regression
  - Probabilistic discriminative models
  - Logistic sigmoid (logit function)
  - Cross-entropy error
  - Gradient descent
  - Iteratively Reweighted Least Squares
- Note on Error Functions

Note on Error Functions

- Ideal misclassification error function (black)
  - This is what we would like to approximate.
  - Unfortunately, it is not differentiable.
  - The gradient is zero for misclassified points.
  - \( z_n = t_n y(x_n) \)
  - We cannot minimize it by gradient descent.

Note on Error Functions

- Squared error used in Least-Squares Classification
  - Very popular, leads to closed-form solutions.
  - However, sensitive to outliers due to squared penalty.
  - Penalizes “too correct” data points
  - \( z_n = t_n y(x_n) \)
  - Generally does not lead to good classifiers.

Comparing Error Functions (Loss Functions)

- Cross-Entropy Error
  - Minimizer of this error is given by posterior class probabilities.
  - Concave error function, unique minimum exists.
  - Robust to outliers, error increases only roughly linearly
  - But no closed-form solution, requires iterative estimation.

Overview: Error Functions

- Ideal Misclassification Error
  - This is what we would like to optimize.
  - But cannot compute gradients here.
- Quadratic Error
  - Easy to optimize, closed-form solutions exist.
  - But not robust to outliers.
- Cross-Entropy Error
  - Minimizer of this error is given by posterior class probabilities.
  - Concave error function, unique minimum exists.
  - But no closed-form solution, requires iterative estimation.
  - \( \Rightarrow \text{Analysis tool to compare classification approaches} \)
References and Further Reading

• More information on Linear Discriminant Functions can be found in Chapter 4 of Bishop’s book (in particular Chapter 4.1 - 4.3).

Christopher M. Bishop
Pattern Recognition and Machine Learning
Springer, 2006