Course Outline

- Fundamentals (2 weeks)
  - Bayes Decision Theory
  - Probability Density Estimation
  - Mixture Models and EM
- Discriminative Approaches (5 weeks)
  - Linear Discriminant Functions
  - Statistical Learning Theory & SVMs
  - Boosting, Decision Trees
- Generative Models (5 weeks)
  - Bayesian Networks
  - Markov Random Fields
- Regression Problems (2 weeks)
  - Gaussian Processes

Recap: Mixture of Gaussians (MoG)

- "Generative model"

\[ p(j) = \pi_j \]

\[ p(j = 1|x_n) = \frac{\sum_{i=1}^{N} h(j = 1|x_n) x_n}{\sum_{i=1}^{N} h(j = 1|x_n)} \]

\[ p(j = 2|x_n) = \frac{\sum_{i=1}^{N} h(j = 2|x_n) x_n}{\sum_{i=1}^{N} h(j = 2|x_n)} \]

Recap: MoG - Iterative Strategy

- Assuming we knew the values of the hidden variable...

\[ f(x) \]

ML for Gaussian #1

assumed known

1 1 1

2 2 2

j

ML for Gaussian #2

Recap: MoG - Iterative Strategy

- Assuming we knew the mixture components...

\[ f(x) \]

\[ p(j = 1|x) \]

\[ p(j = 2|x) \]

\[ p(j = 1|x_n) > p(j = 2|x_n) \]

Bayes decision rule: Decide \( j = 1 \) if
Recap: K-Means Clustering

- Iterative procedure
  1. Initialization: pick $K$ arbitrary centroids (cluster means)
  2. Assign each sample to the closest centroid.
  3. Adjust the centroids to be the means of the samples assigned to them.
  4. Go to step 2 (until no change)
- Algorithm is guaranteed to converge after finite #iterations.
  - Local optimum
  - Final result depends on initialization.

Topics of This Lecture

- Linear discriminant functions
  - Definition
  - Extension to multiple classes
- Least-squares classification
  - Derivation
  - Shortcomings
- Generalized linear models
  - Connection to neural networks
  - Generalized linear discriminants & gradient descent
- Fisher’s linear discriminant (FLD)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications

Discriminant Functions

- Formulate classification in terms of comparisons
  - Discriminant functions
    - $y_1(x), \ldots, y_K(x)$
    - Classify $x$ as class $C_k$ if $y_k(x) > y_j(x)$, $\forall j \neq k$
  - Examples (Bayes Decision Theory)
    - $y_k(x) = p(C_k|x)$
    - $y_x(x) = p(x|C_k)p(C_k)$
    - $y_k(x) = \log p(x|C_k) + \log p(C_k)$

Recap: EM Algorithm

- Expectation-Maximization (EM) Algorithm
  - E-Step: softly assign samples to mixture components
    - $\gamma_{j}(x_n) \leftarrow \sum_{k=1}^{K} \frac{p_j(x_n|\mu_k, \Sigma_k)}{p_j(x_n|\mu_k, \Sigma_k)}$
    - $\forall j = 1, \ldots, K$, $n = 1, \ldots, N$
  - M-Step: re-estimate the parameters (separately for each mixture component) based on the soft assignments
    - $\mu_{j}\leftarrow \frac{1}{N_j} \sum_{n=1}^{N} \gamma_j(x_n)x_n$
    - $\Sigma_{j}\leftarrow \frac{1}{N_j} \sum_{n=1}^{N} \gamma_j(x_n)(x_n - \mu_{j})^T(x_n - \mu_{j})$

Discriminant Functions

- Bayesian Decision Theory
  - $p(C_j|x) = \frac{p(x|C_j)p(C_j)}{p(x)}$
  - Model conditional probability densities $p(x|C_j)$ and priors $p(C_j)$
  - Compute posteriors $p(C_j|x)$ (using Bayes’ rule)
  - Minimize probability of misclassification by maximizing $p(C|x)$
  - New approach
    - Directly encode decision boundary
    - Without explicit modeling of probability densities
    - Minimize misclassification probability directly.

- Example: 2 classes
  - $y_1(x) > y_2(x)$
  - $y_1(x) - y_2(x) > 0$
  - $y(x) > 0$
  - Decision functions (from Bayes Decision Theory)
    - $y(x) = p(C_1|x) - p(C_2|x)$
    - $y(x) = \log \frac{p(x|C_1)}{p(x|C_2)} + \log \frac{p(C_1)}{p(C_2)}$
**Learning Discriminant Functions**

- **General classification problem**
  - Goal: take a new input $x$ and assign it to one of $K$ classes $C_k$.
  - Given: training set $X = \{x_1, \ldots, x_n\}$ with target values $T = \{t_1, \ldots, t_n\}$.
  - Learn a discriminant function $y(x)$ to perform the classification.

- **2-class problem**
  - Binary target values: $t_n \in \{0, 1\}$

- **K-class problem**
  - 1-of-$K$ coding scheme, e.g. $t_n = (0, 1, 0, 0)^T$

**Linear Discriminant Functions**

- **Decision boundary $y(x) = 0$ defines a hyperplane**
  - Normal vector: $w$
  - Offset: $-\frac{w_0}{\|w\|}$

- **2-class problem**
  - $y(x) > 0$: Decide for class $C_k$, else for class $C_{\neg k}$

- **K-class problem**
  - One-vs-all: Learn $K$ linear discriminant functions $y(x)$
  - One-vs-one: Learn all pair-wise linear discriminant functions $y(x)$

- **Extension to Multiple Classes**
  - Two simple strategies
    - One-vs-all classifiers
    - One-vs-one classifiers

- **Problem**
  - Both strategies result in regions for which the pure classification result ($y_i > 0$) is ambiguous.
  - In the one-vs-all case, it is still possible to classify those inputs based on the continuous classifier outputs $y_i > y_j \iff j \neq k$.

- **Solution**
  - We can avoid those difficulties by taking $K$ linear functions of the form $y_k(x) = w_k^T x + w_{k0}$ and defining the decision boundaries directly by deciding for $C_k$ if $y_k > y_j \forall j \neq k$.
  - This results in the decision hyperplanes:
    $$(w_i^T x + w_{i0}) - (w_j^T x + w_{j0}) = 0$$
Extension to Multiple Classes

- K-class discriminant
  - Combination of K linear functions
    \[ y_k(x) = w_k^T x + w_{k0} \]
  - Resulting decision hyperplanes:
    \[ (w_k - w_j)^T x + (w_{k0} - w_{j0}) = 0 \]
  - It can be shown that the decision regions of such a discriminant are always singularly connected and convex.
  - This makes linear discriminant models particularly suitable for problems for which the conditional densities \( y(x|w_k) \) are unimodal.

Topics of This Lecture

- Linear discriminant functions
  - Definition
  - Extension to multiple classes
- Least-squares classification
  - Derivation
  - Shortcomings
- Generalized linear models
  - Connection to neural networks
  - Generalized linear discriminants & gradient descent
- Fisher’s linear discriminant (FLD)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications

General Classification Problem

- Classification problem
  - Let’s consider K classes described by linear models
    \[ y_k(x) = w_k^T x + w_{k0}, \quad k = 1, \ldots, K \]
  - We can group those together using vector notation
    \[ \hat{y}(x) = \hat{W}^T \hat{x} \]
    \[ \hat{W} = [\hat{w}_1, \ldots, \hat{w}_K] = \begin{bmatrix} w_{10} & \cdots & w_{1K} \\ w_{21} & \cdots & w_{2K} \\ \vdots & \ddots & \vdots \\ w_{D1} & \cdots & w_{DK} \end{bmatrix} \]
  - The output will again be in 1-of-K notation.
  - We can directly compare it to the target value \( t = [t_1, \ldots, t_k]^T \).

Least-Squares Classification

- Simplest approach
  - Directly try to minimize the sum-of-squares error
    \[ E_D(\hat{W}) = \frac{1}{2} \text{Tr} \left\{ (\hat{X}\hat{W} - T)^T (\hat{X}\hat{W} - T) \right\} \]
  - Taking the derivative yields
    \[ \frac{\partial}{\partial \hat{W}} E_D(\hat{W}) = \frac{1}{2} \text{Tr} \left\{ (\hat{X}\hat{W} - T)^T (\hat{X}\hat{W} - T) \right\} \]
    \[ = \frac{1}{2} \frac{\partial}{\partial \hat{W}} \text{Tr} \left\{ (\hat{X}\hat{W} - T)^T (\hat{X}\hat{W} - T) \right\} \]
    \[ = \frac{\partial}{\partial \hat{W}} \text{Tr} \left\{ \hat{X}^T (\hat{X} \hat{W} - T) \right\} \]
    \[ = \hat{X}^T (\hat{X} \hat{W} - T) \]
  - Chain rule: \[ \frac{\partial}{\partial Z} \frac{\partial}{\partial Y} \frac{\partial}{\partial X} = \frac{\partial}{\partial Y} \frac{\partial}{\partial X} \]
  - Using:
    \[ \frac{\partial}{\partial A} \text{Tr} \{ A \} = I \]
  - We then obtain the discriminant function as
    \[ y(x) = \hat{W}^T \hat{x} = \hat{T}^T (\hat{X}^T)^{-1} \hat{X}^T \hat{x} \]
  - Exact, closed-form solution for the discriminant function parameters.
Problems with Least Squares

- Least-squares is very sensitive to outliers!
  - The error function penalizes predictions that are “too correct”.

Another example:
- 3 classes (red, green, blue)
- Linearly separable problem
- Least-squares solution:
  Most green points are misclassified!

Deeper reason for the failure
- Least-squares corresponds to Maximum Likelihood under the assumption of a Gaussian conditional distribution.
- However, our binary target vectors have a distribution that is clearly non-Gaussian!
⇒ Least-squares is the wrong probabilistic tool in this case!

Topics of This Lecture
- Linear discriminant functions
  - Definition
  - Extension to multiple classes
- Least-squares classification
  - Derivation
  - Shortcomings
- Generalized linear models
  - Connection to neural networks
  - Generalized linear discriminants & gradient descent
- Fisher's linear discriminant (F.L.D)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications

Generalized Linear Models
- Linear model
  \[ y(x) = w^T x + w_0 \]
- 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 \( x \).

Generalized Linear Models
- Consider 2 classes:
  \[
  p(C_1|x) = \frac{p(x|C_1)p(C_1)}{p(x|C_1)p(C_1) + p(x|C_2)p(C_2)}
  = \frac{1}{1 + \frac{p(x|C_2)p(C_2)}{p(x|C_1)p(C_1)}}
  = \frac{1}{1 + \exp(-a)} \equiv g(a)
  \]
  with \( a = \ln \frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)} \)

Logistic Sigmoid Activation Function
- Example: Normal distributions with identical covariance
- \[ g(a) = \frac{1}{1 + \exp(-a)} \]
- \[ p(x|a) \] and \[ p(x|b) \]
- \[ p(a|x) \] and \[ p(b|x) \]
### Normalized Exponential
- General case of $K \geq 2$ classes:
  \[
p(C_k | x) = \frac{p(x | C_k)p(C_k)}{\sum_j p(x | C_j)p(C_j)} = \frac{\exp(a_k)}{\sum_j \exp(a_j)}
\]
  with $a_k = \ln p(x | C_k)p(C_k)$
  - This is known as the normalized exponential or softmax function.
  - Can be regarded as a multiclass generalization of the logistic sigmoid.

### Relationship to Neural Networks
- 2-Class case
  \[
y(x) = \sum_{i=0}^{D} g(w_i x_i) \text{ with } x_0 = 1 \text{ constant}
\]
- Multi-class perceptron
  \[
y_k(x) = \sum_{i=0}^{D} g(w_i x_i) \text{ with } x_0 = 1 \text{ constant}
\]
- Multi-class perceptron
  \[
y_k(x) = \sum_{i=0}^{D} g(w_i x_i)
\]
  - can be used to limit the effect of outliers.

### Other Motivation for Nonlinearity
- Recall least-squares classification
  - One of the problems was that data points that are “too correct” have a strong influence on the decision surface under a squared-error criterion.
    \[
    E(w) = \sum_{n=1}^{N} (y(x_n; w) - t_n)^2
    \]
    - Reason: the output of $y(x_n; w)$ can grow arbitrarily large for some $x_n$.
  - By choosing a suitable nonlinearity (e.g. a sigmoid), we can limit those influences
    \[
y(x; w) = g(w^T x + w_0)
    \]

### Discussion: Generalized Linear Models
- Advantages
  - The nonlinearity gives us more flexibility.
  - Can be used to limit the effect of outliers.
  - Choice of a sigmoid leads to a nice probabilistic interpretation.
- Disadvantage
  - Least-squares minimization in general no longer leads to a closed-form analytical solution.
  - Need to apply iterative methods.
  - Gradient descent.
Linear Separability
- Up to now: restrictive assumption
  - Only consider linear decision boundaries
- Classical counterexample: XOR

\[
x_2
\begin{cases}
  \bullet C_2 & \text{if } x_1 < 0 \\
  \bullet C_1 & \text{if } x_1 \geq 0
\end{cases}
\]

Gradient Descent
- Learning the weights \( w \):
  - \( N \) training data points: \( X = \{ x_1, \ldots, x_N \} \)
  - \( K \) outputs of decision functions: \( y_k(x,w) \)
  - Target vector for each data point: \( T = \{ t_1, \ldots, t_N \} \)
  - Error function (least-squares error) of linear model
    \[
    E(w) = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} \left( y_k(x_n,w) - t_{kn} \right)^2
    \]
    \[
    = \frac{1}{2} \sum_{n=1}^{N} \sum_{k=1}^{K} \left( \sum_{j=1}^{M} w_{kj} \phi_j(x_n) - t_{kn} \right)^2
    \]

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

Generalized Linear Discriminants
- Model
  \[
  y_k(x) = \sum_{j=0}^{M} w_{kj} \phi_j(x) = y_k(x;w)
  \]
  - \( K \) 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
- 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 \( w_k^{(0)} \).
    - Move towards a (local) minimum by following the gradient.
      \[
      w_{k}^{(r+1)} = w_{k}^{(r)} - \eta \frac{\partial E(w)}{\partial w_k} \bigg|_{w(r)}
      \]
      \( \eta \): Learning rate
  - This simple scheme corresponds to a 1st-order Taylor expansion (There are more complex procedures available).
Perceptual and Sensory Augmented Computing

### Gradient Descent - Basic Strategies

#### “Batch learning”

\[
W_{kj}^{(r+1)} = W_{kj}^{(r)} - \eta \frac{\partial E(w)}{\partial w_{kj}} \bigg|_{w^{(r)}}
\]

\(\eta\): Learning rate

- Compute the gradient based on all training data:

\[
\frac{\partial E(w)}{\partial w_{kj}}
\]

#### “Sequential updating”

\[
E(w) = \sum_{n=1}^{N} E_n(w)
\]

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

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

\[
W_{kj}^{(r+1)} = W_{kj}^{(r)} - \eta \frac{\partial E_n(w)}{\partial w_{kj}} \bigg|_{w^{(r)}}
\]

\(\eta\): Learning rate

- Compute the gradient based on a single data point at a time:

\[
\frac{\partial E_n(w)}{\partial w_{kj}}
\]

### Summary: Generalized Linear Discriminants

#### Properties

- General class of decision functions.
- Nonlinearity \(g(\cdot)\) and basis functions \(\phi_j\) allow 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 low-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

- Linear discriminant functions
  - Definition
  - Extension to multiple classes
- Least-squares classification
  - Derivation
  - Shortcomings
- Generalized linear models
  - Connection to neural networks
  - Generalized linear discriminants & gradient descent
- Fisher’s linear discriminant (FLD)
  - Classification as dimensionality reduction
  - Linear discriminant analysis
  - Multiple discriminant analysis
  - Applications

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 \( D \)-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.

  ⇒ New interpretation of the learning problem
  - Try to find the projection vector \( w \) that maximizes the class separation.

Classification as Dimensionality Reduction

Two questions
- How to measure class separation?
- How to find the best projection (with maximal class separation)?

Fisher’s Linear Discriminant Analysis (FLD)

- Better idea:
  - Find a projection that maximizes the ratio of the between-class variance to the within-class variance:
    \[ J(w) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \]
    with \( s_k^2 = \sum_{a \in C_k} (y_a - m_k)^2 \)
  - Usually, this is written as
    \[ J(w) = \frac{w^T S_B w}{w^T S_W w} \]
  - where
    \[ S_B = (m_2 - m_1)(m_2 - m_1)^T \]
    between-class scatter matrix
    \[ S_W = \frac{1}{2} \sum_{a \in C_1} (x_a - m_1)(x_a - m_1)^T + \frac{1}{2} \sum_{a \in C_2} (x_a - m_2)(x_a - m_2)^T \]
    within-class scatter matrix

- Problems with this approach
  1. This expression can be made arbitrarily large by increasing \( ||w|| \).
  2. Need to enforce additional constraint \( ||w|| = 1 \).

  Classification function:
  \[ y(x) = w^T x + w_0 \]
  \( y(x) \geq 0 \) for Class 1, \( y(x) < 0 \) for Class 2.
**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 \( J(W) \)**

- Generalized eigenvalue problem
  \[
  J(W) = \frac{W^T S_B W}{W^T S_W W}
  \]
  - The columns of the optimal \( W \) are the eigenvectors corresponding to the largest eigenvalues of
    \[
    S_B W_i = \lambda_i S_W W_i
    \]
  - Defining \( v = S_B^T W \), we get
    \[
    S_B^T S_B^{-1} S_B^T v = \lambda v
    \]
    which is a regular eigenvalue problem.
  - Solve for eigenvectors of \( v \), then from that of \( w \).
- For the \( K \)-class case we obtain (at most) \( K \) 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 \( \text{sign}(x) = \text{sign}(w) \).
  - 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
  - Training data:
    - \( C_1 \): Subjects with glasses
    - \( C_2 \): Subjects without glasses
  - 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.
  - Implicit assumption that class distributions are unimodal and well-approximated by a Gaussian/hyperellipsoid.
  - Assumption that \( N > D \) (more training examples than dims.)
  - This may not be given in some domains (e.g. vision)
  - Solution: apply PCA first to get problem with \( D = N \)-1 dimensions.
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).

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