Outline

1. Overview of Classification
2. Discriminant Functions
   - From Two Classes to Multiple Classes
   - Least Square
   - Fisher’s Linear Discriminant
   - Find a Separating Hyperplane
3. Prob. Generative Models
   - Linear Discriminant Analysis
   - Quadratic Discriminant Analysis
4. Prob. Discrimintive Models
   - Logistic Regression
   - LDA or Logistic Regression?
5. Bonus
   - Multi-Class, Multi-Label, Struct-Output
Outline of this section

1. Overview of Classification
2. Discriminant Functions
3. Prob. Generative Models
4. Prob. Discriminative Models
5. Bonus
What is Classification?

**Definition (Classification)**

**Classification** is the problem of identifying to which of a set of categories (sub-populations) a new observation belongs, on the basis of a training set of data containing observations (or instances) whose category membership is known.

- **Input:** Feature vector \( \mathbf{x} \), continuous or discrete;
- **Output:** Qualitative values (i.e. categorical or discrete values) \( y \), where \( y \in \mathbf{Y} \), \( \mathbf{Y} = \{y_1, y_2, \ldots, y_n\} \);
- **Mapping:** \( y = f(\mathbf{x}) : \mathbf{x} \rightarrow y \);
- **Supervised Learning/Semi-supervised Learning.**
Example: Image Classification

Figure: An example of image classification. **Input:** Visual Features (e.g. SIFT, Color Moment); **Output:** Scene Category (e.g. "Bridge", "Castle"); **Mapping:** A Sparse Factor Representation (see "Multi-Label Image Categorization With Sparse Factor Representation", Sun, et.al. *IEEE TIP* 2014)
Outline of this section

1. Overview of Classification

2. Discriminant Functions
   - From Two Classes to Multiple Classes
   - Least Square
   - Fisher’s Linear Discriminant
   - Find a Separating Hyperplane

3. Prob. Generative Models

4. Prob. Discriminative Models

5. Bonus
Let’s start with the simplest case: Two-class Classification, where:

- **Input:** Feature vector \( x \), continuous or discrete;
- **Output:** Binary values \( y \), where \( y \in Y \), \( Y = \{-1, +1\} \) or \( Y = \{0, 1\} \);
- **Model:** \( y(X) = w^T x + w_0 \), \( x \) is assigned with \(-1\), if \( y(x) \geq 0 \); Otherwise, \( x \) is assigned with \(+1\).
- **Examples:** Male or Female; Toy Data.
Example of Two-Class Problem: On "Toy" Dataset

Figure: A demonstration of Classification on "toy" dataset via Semi-supervised Learning. (see "Graph Transduction via Alternating", Wang, et.al, ICML, 2008)
From Two-Class to Multiple-Class

A complex case: **Multiple-class Classification**, where:

- **Input:** Feature vector $\mathbf{x}$, continuous or discrete;
- **Output:** For a $n$-class problem, Qualitive values $C$, where $C \in \mathbf{C}$, $\mathbf{C} = \{C_1, C_2, ..., C_n\}$;
- **Model:**
  - **Binary:** One-vs-One
  - **Binary:** One-vs-Rest
  - **Binary:** ECOC
  - **Directly comprising** $n$ Linear functions
From Two-Class to Multiple-Class (Cont.)

Scheme 1: Decompose a \( n \)-class problem into several two-classes problems:

- **One-vs-Rest:** Use \( n - 1 \) binary classifiers each of which solves a two-class problem of separating points in a particular class \( C_k \) from points not in that class.
  - Pros: Easy and Efficient;
  - Cons: Ambiguous samples; **Unbalanced Samples**.

- **One-vs-One:** Use \( (n - 1)n/2 \) binary classifiers, one for every possible pairs of classes. Each point is classified according to a **majority vote** amongst the discriminant functions.
  - Pros: "Better" performance than "One-vs-Rest"
  - Cons: Ambiguous samples; Too many classifiers;
Problems from 1-vs-1 and 1-vs-R

Figure: Attempting to construct a $n$ class discriminant from a set of two class discriminants leads to ambiguous regions, shown in green. On the left is an example involving the use of two discriminants designed to distinguish points in class $C_k$ from points not in class $C_k$. On the right is an example involving three discriminant functions each of which is used to separate a pair of classes $C_k$ and $C_j$. (PRML pp.183)
**From Two-Class to Multiple-Class (Cont.)**

**ECOC (Error-Correcting Output Codes):** Given a set of $n$ classes, the basis of the ECOC framework consists of designing a codeword for each of the classes. These codewords encode the membership information of each class for a given binary problem.

- **Pros:** Robust;
- **Cons:** The construction of codebook.

```
<table>
<thead>
<tr>
<th>y_1</th>
<th>y_2</th>
<th>y_3</th>
<th>y_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>+1</td>
<td>-1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>+1</td>
<td>-1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>+1</td>
</tr>
</tbody>
</table>
```

**ECOC Extension**

```
| +1  | +1  | -1  |
| +1  | -1  | +1  |
| -1  | +1  | +1  |
| -1  | -1  | -1  |
```

*Figure: ECOC for a 4-Class Problem*
A comprehensive evaluation on different schemes for "TwoClass2MultipleClass" has been conducted (R. Rifkin and A. Klautau, "In Defense of One-Vs-All Classification", JMLR, 2004). Their main thesis is that a simple "one-vs-all" scheme is as accurate as any other approach, assuming that the underlying binary classifiers are well-tuned regularized classifiers such as support vector machines. This thesis is interesting in that it disagrees with a large body of recent published work on multiclass classification. They support our position by means of a critical review of the existing literature, a substantial collection of carefully controlled experimental work, and theoretical arguments.
Directly comprising $n$ Linear functions:

$$y_k(x) = w^T x + w_0, \ k = 1, 2, ..., n$$

$x$ is assigned with Class $C_k$ if $y_k \geq y_j (k \in \{1, 2, ..., n\}, k \neq j)$

- Pros: Likelihood Score, Convex;
- Cons: Easy to solve? Cannot use the binary mode;
Methods for Classification

- **Discriminant Functions:**
  - **Data Fitting:** Least Squares Estimation;
  - **Feature Reduction:** Fisher’s Discriminant Analysis;
  - **Find a Separating Hyperplane:** Perceptron and SVM.

- **Probabilistic Generative Model:** LDA/QDA.

- **Probabilistic Discriminative Model:** Logistic Regression.
Least Squares For Classification

**Least Squares:** Similar to linear regression, we can solve the classification via least squares estimation.

- **Data:** Given the training set \( \{(x_n, t_n)\}, n = 1, 2, \ldots, N, x_n \) is the feature vector; \( t_n \) is a indicator vector. For a \( K \)-Class problem, \( t_{n,k} = 1, t_{n,i} = 0(i \neq k) \), if \( x_n \) belongs to \( C_k \).

- **Decision Function:** For Class \( C_k \), \( y_k((x)) = w_k^T x + w_{k,0} \). \( x \) is assigned with \( C_k \), if \( y_k(x) \geq y_i(x), k \neq i \)

- **Opt Problem:**

\[
\min_W \quad L(W) = \frac{1}{2} Tr\{(W^T X - T)(W^T X - T)^T\}
\]

\[
\text{s.t.} \quad X = \{x_1, x_2, \ldots, x_N\}; \\
W = \{w_1^T, w_2^T, \ldots, w_K^T\}^T; \\
T = \{t_1^T, t_2^T, \ldots, t_K^T\}^T.
\]

- **Cons:** Sensitive to singular data; Poor performance with asymmetry distributed data.
Least Squares For Classification (Cont.)

Figure: Failure cases: Left: Singular data; Right: Asymmetry data
Fisher’s Linear Discriminant (Fisher, 1936)

- **Idea:** Treat the classification problem as feature reduction (projected to 1-D);

**Figure:** The corresponding projection based on the Fisher linear discriminant.
**Fisher’s Linear Discriminant (Cont.)**

- **Criterion:** Maximize the between-class variance and Minimize the within-class variance.
- **Data:** For a 2-Class problem, training set \( \{(x_n, y_n)\}. n = 1, 2, ..., N, y_n \in \{-1, +1\} \)
- **Decision Function:** \( y = w^T x \)
- **Opt Problem:**

\[
\begin{align*}
\max_w \quad & J(w) = \frac{(m_2 - m_1)^2}{S_1^2 + S_2^2} = \frac{w^T S_B w}{w^T S_W w} \\
\text{s.t.} \quad & m_k = w^T m_k; m_k = \frac{1}{N_k} \sum_{x_n \in C_k} x_n; \\
& S_k^2 = \sum_{x_n \in C_k} (y_n - m_k)^2; y_k = w^T x_k; \\
& S_B = (m_2 - m_1)(m_2 - m_1)^T; \\
& S_W = \sum_k \sum_{n \in C_k} (x_n - m_k)(x_n - m_k)^T.
\end{align*}
\]
Fisher’s Linear Discriminant (Cont.)

- **MultiClass Cases:** (Reduced to K-Dimension)
  - **Data:** For a K-Class problem, training set \( \{(x_n, t_n) \} \) \( n = 1, 2, ..., N \), \( t_n \) is a K-dim indicator vector.
  - **Decision Function:** \( y = W^T x \)
  - **Opt Problem:**

\[
\begin{align*}
\max_w \quad & J(W) = \text{Tr}\{s_W^{-1}s_B\} \\
\text{s.t.} \quad & s_W = \sum_{k=1}^{N} \sum_{n \in C_k} (y_n - \mu_k)(y_n - \mu_k)^T \\
& s_B = \sum_{k=1}^{K} N_k (\mu_k - \mu)(\mu_k - \mu)^T \\
& \mu_k = \frac{1}{N_k} \sum_{n \in C_k} y_n, \mu = \frac{1}{N} \sum_{k=1}^{N} \mu_k N_k.
\end{align*}
\]
Find a Separating Hyperplane

This procedure tries to conduct linear decision boundaries that explicitly separate the data into different classes.

- **Perceptron**
- **Support Vector Machine**

**Figure**: The orange line is the least squares solution, which misclassifies one of the training points. Two blue separating hyperplanes are found by the perceptron with different random starts.
Preliminary

The distance between decision boundary and point:

- **Decision boundary**: $y(x) = w^T x + w_0 = 0$;
- **Point**: $x = x_\perp + r \frac{w}{\|w\|}$;
- **Projection**: $r = \frac{y(x)}{\|w\|}$ (Pos/Neg).
The **Perceptron** learning algorithms tries to find a separating hyperplane by minimizing the distance of misclassified points to the decision boundary. If a response $y_i = 1$ is misclassified, then $y = w^T \phi(x)$, and the opposite for a misclassified response with $y_i = -1$.

- **Problem:** Binary Classification;
- **Decision Function:** $y = w^T \phi(x) + w_0$;
- **Opt Model:**

$$\min_{w, \|w\|=1} - \sum_{i \in M} y_i (w^T \phi(x))$$

where $M$ indexes the set of misclassified points. This problem can be solved via Stochastic Gradient Descent (SGD) algorithm.

- **Cons:** The solution is not unique; The number of iteration can be very large.
Optimal Separating Hyperplanes (Vapnik, 1996)

This approach tries to provide a unique solution to separating hyperplane problem via **maximizing** the margin between the two classes. For a binary problem with $N$ training data, we have:

$$\begin{align*}
\max_{w, w_0} & \quad C \\
\text{s.t.} & \quad y_i \frac{(w^T x_k + w_0)}{||w||} \geq C, \ i = 1, 2, \ldots, N
\end{align*}$$

Set $C ||w|| = 1$, the problem can be written as:

$$\begin{align*}
\min_{w, w_0} & \quad \frac{1}{2} ||w|| \\
\text{s.t.} & \quad y_i (w^T x_k + w_0) \geq 1, \ i = 1, 2, \ldots, N
\end{align*}$$

This is the basic form of Linear Support Vector Machine.
Outline of this section

1. Overview of Classification

2. Discriminant Functions

3. Prob. Generative Models
   - Linear Discriminant Analysis
   - Quadratic Discriminant Analysis

4. Prob. Discriminative Models

5. Bonus
Generative Model Review

This part is based on Bayesian Posterior Inference as follows:

\[ P(C_k|x) = \frac{P(x|C_k)P(C_k)}{P(x)} = \frac{P(x|C_k)P(C_k)}{\sum_j P(x|C_j)P(C_j)} \]

If \( P(x|C_k) \) is Gaussian, then the "log-odds" \( c \) is:

\[
\log \frac{P(C_i|x)}{P(C_j|x)} = \frac{1}{2} \log \frac{|\Sigma_j|}{|\Sigma_i|} + \frac{1}{2}[(x - \mu_j)^T \Sigma_j^{-1}(x - \mu_j) - (x - \mu_i)^T \Sigma_i^{-1}(x - \mu_i)] \\
+ \log \frac{P(C_i)}{P(C_j)}
\]

The task is to estimate the covariance and expectation of each class.
Linear Discriminant Analysis (LDA) arises in the special case when we assume that the classes have a common covariance matrix $\Sigma_k = \Sigma, \forall k$. So we have:

$$\log \frac{P(C_i|x)}{P(C_j|x)} = \log \frac{P(C_i)}{P(C_j)} - \frac{1}{2} (\mu_i + \mu_j)^T \Sigma^{-1} (\mu_i - \mu_j) + x^T \Sigma^{-1} (\mu_i - \mu_j)$$

an equation linear in $x$. For each class, we have the linear discriminant functions:

$$\delta_k(x) = x^T \Sigma^{-1} \mu_k - \frac{1}{2} \mu_k^T \Sigma^{-1} \mu_k + \log P(C_k)$$
Linear Discriminant Analysis (Cont.)

How to estimate the parameters? MLE.

- \( P(\hat{C}_k) = \frac{N_k}{N} \), where \( N_k \) is the number of Class-\( k \) observations;
- \( \hat{\mu}_k = \frac{1}{N_k} \sum_{n \in C_k} x_n; \)
- \( \hat{\Sigma} = \sum_{k=1}^{K} \sum_{n \in C_k} \frac{1}{N-K} (x - \mu_k)(x - \mu_k)^T \)
Getting back to the general discriminant problem, if the $\Sigma_k$ are not assumed to be equal, then the pieces quadratic in $x$ remain. We get quadratic discriminant functions:

$$\delta_k(x) = -\frac{1}{2} \log |\Sigma_k| - \frac{1}{2} (x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k) + \log P(C_k)$$

The decision boundary between each pair of classes $K$ and $L$ is described by a quadratic equation $\{x : \delta_k(x) = \delta_l(x)\}$

- **Note:** Compared with LDA, QDA performed better for unlinear problems.
Outline of this section

1. Overview of Classification
2. Discriminant Functions
3. Prob. Generative Models
4. Prob. Discriminative Models
   - Logistic Regression
   - LDA or Logistic Regression?
5. Bonus
The Logistic Regression Model arises from the desire to model the posterior probabilities of the $K$ classes via linear functions in $x$, while at the same time they sum to one and remain in $[0, 1]$. The model has the form:

\[
\begin{align*}
\log \frac{P(C_1|x)}{P(C_k|x)} &= w_0 + w_1^T x \\
\log \frac{P(C_2|x)}{P(C_k|x)} &= w_0 + w_2^T x \\
&\quad \vdots \\
\log \frac{P(C_{k-1}|x)}{P(C_k|x)} &= w_{(k-1)0} + w_{k-1}^T x
\end{align*}
\]
Logistic Regression (Cont.)

The model is specified in terms of $K - 1$ log-odds. We use the logistic function $f(\sigma) = \frac{1}{1 + \exp(-\sigma)}$ and we have:

$$P(C_k | x) = \frac{\exp(w_{k0} + w_k^T x)}{1 + \sum_{l=1}^{K-1} \exp(w_{l0} + w_l^T x)}, \quad k = 1, 2, ..., K - 1$$

$$P(C_K | x) = \frac{1}{1 + \sum_{l=1}^{K-1} \exp(w_{l0} + w_l^T x)}$$

- **Computation:** Maximum Likelihood Estimation;
LDA or Logistic Regression?

For LDA, we find that the log-posterior odds between class $k$ and $K$ are linear functions of $x$:

$$\log \frac{P(C_i|x)}{P(C_j|x)} = \log \frac{P(C_i)}{P(C_j)} - \frac{1}{2} (\mu_i + \mu_j)^T \Sigma^{-1} (\mu_i - \mu_j) + x^T \Sigma^{-1} (\mu_i - \mu_j)$$

$$= \alpha_{k0} + \alpha_k^T x$$

For Logistic Regression, we have:

$$\log \frac{P(C_{k-1}|x)}{P(C_k|x)} = w_{(k-1)0} + w_{k-1}^T x$$

It seems that the models are the same. The Logistic Regression model is more general, in that it makes less assumptions.
Outline of this section

1. Overview of Classification
2. Discriminant Functions
3. Prob. Generative Models
4. Prob. Discrimintive Models
5. Bonus
   - Multi-Class, Multi-Label, Struct-Output
Roadmap of Machine Learning in Computer Vision

Early 1990s

Classification
  - Binary Classification

2000s

Multiple-Labeling
  - Multi-Label
  - Multi-Instance

Structure Output
  - Bounding-Box
  - Attributive

Today

- Event & Behaviour
- Pixel-wise Recognition

Y. Gu

LINEAR MODELS FOR CLASSIFICATION
Multi-Label and Multi-Instance

Figure: Examples of Multi-label Learning (Li, CIKM'13)
Structure-Output: Bounding-Box

Figure: Examples of Bounding Box (Lan, ECCV'12)
Structure-Output: Sentence

Figure: Examples of "Sentence" (Lan, ECCV’12)
Structure-Output: Semantic Description

Figure: Examples of "Semantic Description" (Lan, CVPR'12)
Structure-Output: Pixel-wise Annotation

Figure : Examples of ”Pixel-wise Annotation” (Ladicky, CVPR’13)
Fine-grained Categorization

Figure: Examples of "Fine-grained Categorization" (Yao, CVPR'11)
Challenges & Future Work

Figure: Challenges of Visual Recognition
Challenges & Future Work

- **Learning:**
  - Transfer Learning;
  - Weakly-supervised Learning;
  - Deep Learning.

- **Data:**
  - Visual Saliency, Objectness, Descriptors;
  - Semantic Network.
Thank you.

Q&A.