



# Review

## 课程Slides

1. 什么是模式：广义地说，存在于时间和空间中可观察的物体，如果我们可以区别它们是否相同或是否相似，都可以称之为模式。
2. 模式的直观特性：可观察性，可区分性，相似性
3. 假说的两种获得方法：归纳假说，演绎假说

## 复习Slides

### 概论

1. 风险的定义：风险一般可以通过期望损失（Expected Loss）来度量，表示在某个决策过程中的平均损失。
2. 回归，分类，聚类

回归是一种**监督学习**任务，其目标是预测一个连续的数值输出。给定一组输入数据（特征），回归模型根据这些输入数据预测一个连续的目标变量。

评价指标：RMSE，R<sup>2</sup>

分类是**监督学习**任务的一种，其目标是根据输入特征将数据点分配到**预定义的类别**（标签）中。给定一组输入数据，分类模型学习如何将其划分到不同的类别。

聚类是**无监督学习**任务，其目标是将数据分组（聚类），使得同一组内的数据点具有较高的相似性，而不同组之间的数据点相似性较低。聚类算法不依赖于标签信息，而是基于数据的内在结构进行分组。

3. 机器学习的三个方面：函数族F，目标函数J(f)，优化方法
- ▼ 模型函数族

## 1. 线性模型: $f(x) = w^T x + b$

目标函数:

线性模型的目标是学习一个函数, 使得模型预测值和实际值之间的误差最小。常见的目标函数包括:

- **最小二乘法** (用于回归问题): 最小化平方误差, 即  $\sum_{i=1}^N (y_i - f(x_i))^2$ , 其中  $y_i$  是真实值,  $f(x_i)$  是模型的预测。
- **逻辑回归** (用于分类问题): 最小化对数损失函数 (交叉熵), 即  $-\sum_{i=1}^N [y_i \log(f(x_i)) + (1 - y_i) \log(1 - f(x_i))]$ 。

## 2. 多项式模型: $f(x) = w_0 + w_1 x + w_2 x^2 + \cdots + w_n x^n$

目标函数:

- **最小二乘法**: 最小化平方误差, 同线性模型一样, 目标是最小化  $\sum_{i=1}^N (y_i - f(x_i))^2$ 。

优化算法:

- **梯度下降**: 同样适用于多项式回归问题。
- **正规方程**: 适用于不涉及正则化的多项式回归问题。
- **正则化方法** (如L1、L2正则化): 帮助控制多项式模型的复杂性, 防止过拟合。

## 3. 核方法 (Kernel Methods)

核方法主要通过核函数将数据映射到高维空间, 使得在原始空间无法线性分割的问题, 变得在高维空间中线性可分。支持向量机 (SVM) 就是一个经典的应用。

目标函数:

- **支持向量机 (SVM)** 中的目标函数是最大化间隔, 同时最小化分类错误:

$$\min_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2 \quad \text{subject to} \quad y_i(\mathbf{w}^T \mathbf{x}_i + b) \geq 1$$

- **核技巧**: 使用核函数  $K(x, x') = \phi(x)^T \phi(x')$ , 将数据映射到更高维的特征空间, 在该空间中应用线性方法。

## 4. 决策树 (Piecewise Constant Model)

决策树通过递归地将数据划分为不同的区域, 每个区域内的输出通常是常数。决策树可以处理非线性关系, 且可以解释每个决策过程。

## 5. 神经网络（由网络结构决定）

神经网络通过多个神经元的连接和层次结构来建模复杂的非线性关系。每个神经元通过激活函数计算输出，并将结果传递到下一层。

目标函数：

- 损失函数：用于衡量神经网络预测结果与真实值之间的误差，常见的损失函数有：
  - 均方误差（MSE）：用于回归问题。
  - 交叉熵（Cross-Entropy）：用于分类问题。

优化算法：

- 反向传播算法：通过链式法则计算梯度，并用梯度下降法调整神经网络权重。
- Adam优化器：一种自适应的梯度优化算法，结合了动量和RMSProp的优点。
- 随机梯度下降（SGD）：逐步更新网络权重，常配合小批量数据（Mini-batch）训练。

## 6. 集成学习：多棵决策树的加权平均

集成学习通过组合多个模型的预测结果来提升性能。常见的集成学习方法包括随机森林和梯度提升决策树（GBDT）。

随机森林：

- 目标函数：每棵树通过最大化数据的基尼不纯度或信息增益进行训练。随机森林通过聚合所有树的预测（通常是投票或平均）来做最终决策。
- 优化算法：通过构造多棵随机选择特征和样本的决策树，避免过拟合。

GBDT（梯度提升决策树）：

- 目标函数：每次训练新的决策树时，通过优化损失函数的梯度来调整模型。
- 优化算法：逐步构建树模型，每次根据当前模型的残差拟合新树，使用梯度下降法来最小化目标损失函数。

## 7. 概率图模型：利用条件独立假设简化概率计算

概率图模型（例如贝叶斯网络、马尔可夫网络）通过图的结构来表示变量之间的依赖关系。通过条件独立性假设简化概率计算，使得推理和学习变得高效。

目标函数：

- 最大似然估计（MLE）：通过最大化观察数据的似然函数来学习模型的参数。
- 贝叶斯推理：通过贝叶斯定理计算后验分布。

$$P(\theta|D) = \frac{P(D|\theta)P(\theta)}{P(D)}$$

优化算法：

- EM算法：期望最大化（EM）算法用于隐变量模型，通过迭代优化目标函数（如对数似然）。
- 变分推理：用于近似推理，尤其是当数据量很大时。

## 4. 参数模型，非参数模型

### ▼ 参数模型

线性模型，多项式，神经网络，概率图

### ▼ 非参数模型

核方法，决策树，集成学习

## 5. 目标函数

### ▼ 损失函数

#### 1. 负对数似然损失 (Negative Log-Likelihood Loss)

负对数似然损失函数通常用于分类问题，特别是在概率模型中，如逻辑回归、神经网络等。它是对似然函数取对数后的负数。

#### 2. L2 和 L1 损失

L2 损失（均方误差，MSE）L2 损失常用于回归问题，计算的是预测值与真实值之间差的平方的平均值。

L1 损失（绝对误差，MAE）L1 损失是计算预测值与真实值之间差的绝对值的平均。

#### 3. 交叉熵损失

交叉熵损失是用来度量分类模型预测的概率分布与真实标签的概率分布之间差异的函数。通常用于二分类和多分类问题中，尤其是用于神经网络中。

#### 4. 合页损失 (Hinge Loss)

合页损失主要用于支持向量机 (SVM) 中，用于分类任务。它是为了保证分类边界最大化而设计的。

#### ε-不敏感损失 ( $\epsilon$ -Insensitive Loss)

$\epsilon$ -不敏感损失函数是支持向量回归 (SVR) 中常用的一种损失函数，它对预测误差小于 $\epsilon$ 的部分不产生惩罚。

### ▼ 正则项

#### 正则项的作用？

**L2 正则化** 和 **L1 正则化** 是机器学习和深度学习中常用的两种正则化方法，旨在通过**对模型的权重进行约束，减少过拟合现象**。它们的核心目标是**避免模型过于复杂，提升模型的泛化能力**。

### ▼ 正则参数

$\lambda$ 越小，模型越复杂，训练误差越小，偏差小、方差大

$\lambda$ 越大，模型越简单，训练误差越大，偏差大、方差小

## 6. 🐕

## 7. 优化算法

### ▼ 梯度下降/上升

**梯度下降**是一种常用的优化算法，主要用于寻找函数的局部最小值。其基本思想是根据目标函数的梯度（即目标函数在当前位置的导数）来更新参数，从而朝着最小值方向移动。

### ▼ 坐标下降/上升

在每一次迭代中，固定除了一个参数以外的所有参数，然后最小化（或最大化）目标函数关于该参数的值。这种方法通过在每个坐标轴上优化目标函数来逐渐逼近最优解。



## 生成式分类器和判别式分类器

- 首先对联合分布进行推断:

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y})p(\mathbf{x}|\mathbf{y})$$

- 然后使用贝叶斯定理来计算条件分布  $p(\mathbf{y}|\mathbf{x})$

$$p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{x}, \mathbf{y})}{p(\mathbf{x})} = \frac{p(\mathbf{y})p(\mathbf{x}|\mathbf{y})}{\int p(\mathbf{y})p(\mathbf{x}|\mathbf{y})d\mathbf{y}}$$

- 利用条件概率密度来预测。

直接估计条件概率分布  $P(\mathbf{y}|\mathbf{x})$  或条件概率密度函数  $p(\mathbf{y}|\mathbf{x})$  或判别函数。

根据估计的函数确定输出的类别。

特性	生成式分类器	判别式分类器
目标	学习联合分布 $p(\mathbf{x}, \mathbf{y})$	学习条件概率 $(p(\mathbf{y} \mathbf{x}))$
数据假设	假设可以描述数据的生成过程	不关心数据生成过程，专注于类别区分
常见算法	朴素贝叶斯、隐马尔可夫模型、高斯混合模型	逻辑回归、SVM、神经网络
优点	可以处理缺失数据，建模数据生成过程有助于一些任务	通常能更准确地区分类别，计算更高效
缺点	计算复杂，假设较强，容易导致过拟合	对数据生成过程缺乏解释，不关心数据分布细节
适用场景	数据生成过程已知、缺失数据处理	只关心类别区分，数据分布复杂
实例	朴素贝叶斯分类器（假设特征条件独立）	支持向量机（SVM）、决策树、神经网络
	不关心划分各类的边界	
	生成模型能够应付存在隐变量的情况	

## 生成式分类器

▼ 决策规则结论

结论：最小化误差概率条件下，决策规则为：

$P(\omega_1|\mathbf{x}) > P(\omega_2|\mathbf{x})$  则判别  $\mathbf{x} \in \omega_1$  ; 否则判别  $\mathbf{x} \in \omega_2$

▼ 贝叶斯判别公式：

$$P(\omega_i|\mathbf{x}) = \frac{p(\mathbf{x}|\omega_i)P(\omega_i)}{p(\mathbf{x})} = \frac{p(\mathbf{x}|\omega_i)P(\omega_i)}{\sum_{i=2}^2 p(\mathbf{x}|\omega_i)P(\omega_i)}$$

### ■类先验 $P(y = c)$

- 两类:  $y \sim \text{Bernoulli}(\theta)$
- 多类:  $y \sim \text{Multinoulli}(\boldsymbol{\theta})$

### ■类条件 $p(x|y = c)$

- 高斯判别分析:  $p(x|y = c) = N(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$

- 朴素贝叶斯:  $p(x|y = c) = \prod_{j=1}^D p(x_j|y = c)$

- 深度生成模型: 学习从简单分布(高斯)到  $p(x|y = c)$  的映射(神经网络)

#### ▼ 概率分布参数估计

矩估计—用样本的矩估计总体的矩  
 最大似然估计 (MLE) —把参数看作非随机变量  
 最大后验估计 (MAP) —  
 最大化参数后验分布, 把参数看作随机变量  
 贝叶斯参数估计—估计参数的后验分布, 把参数看成是随机变量

#### ▼ 正态分布的贝叶斯分类器 (计算过程)

每一类样本均服从正态分布(均值, 协方差)

$$p(x|y = c) = N(\boldsymbol{\mu}_c, \boldsymbol{\Sigma}_c)$$

协方差矩阵决定了模型的复杂程度, 假设是两类, 协方差都相等的时候, 就变成了线性模型。造成过拟合, 可以对协方差矩阵加结构约束一对角阵

$$f_c(x) = -\frac{1}{2} \ln(|\boldsymbol{\Sigma}_c|) - \frac{1}{2} (x - \boldsymbol{\mu}_c)^T \boldsymbol{\Sigma}_c^{-1} (x - \boldsymbol{\mu}_c) + \ln P(Y = c)$$

### 记忆公式

#### ▼ 高斯判别

$$d_1(x) = \ln P(\omega_1) - \frac{1}{2} \ln |\mathbf{C}_1| - \frac{1}{2} (x - \mathbf{m}_1)^T \mathbf{C}_1^{-1} (x - \mathbf{m}_1) \leftarrow$$

$$d_2(x) = \ln P(\omega_2) - \frac{1}{2} \ln |\mathbf{C}_2| - \frac{1}{2} (x - \mathbf{m}_2)^T \mathbf{C}_2^{-1} (x - \mathbf{m}_2) \leftarrow$$

$$d_1(x) - d_2(x) = \begin{cases} > 0 & x \in \omega_1 \\ < 0 & x \in \omega_2 \end{cases} \leftarrow$$

$$d_1(x) - d_2(x) = \ln P(\omega_1) - \ln P(\omega_2) + (m_1 - m_2)^T C^{-1}x -$$

$$\frac{1}{2} m_1^T C^{-1} m_1 + \frac{1}{2} m_2^T C^{-1} m_2 = 0 \quad \leftarrow$$

## Homework

▼ example2

### 1 Homework1

The equation of mean vector of patterns is (1). The equation of covariance matrix is (2).

$$m_i = \frac{1}{N_i} \sum_{j=1}^{N_i} x_{ij}; i = 1, 2 \quad (1)$$

$$C_i = \frac{1}{N_i} \sum_{j=1}^{N_i} (x_{ij} - m_i)(x_{ij} - m_i)^T; i = 1, 2 \quad (2)$$

With the patterns  $\omega_1, \omega_2$ , we have

$$\begin{cases} m_1 = \frac{1}{4}(4, 4)^T & = (1, 1)^T \\ m_2 = \frac{1}{4}(20, 20)^T & = (5, 5)^T \end{cases} \quad (3)$$

$$\begin{cases} C_1 = \begin{bmatrix} 1, 0 \\ 0, 1 \end{bmatrix} \\ C_2 = \begin{bmatrix} 1, 0 \\ 0, 1 \end{bmatrix} \end{cases} \quad (4)$$

With equstion (4), we know that

$$C_1 = C_2 \quad (5)$$

$$C^{-1} = \begin{bmatrix} 1, 0 \\ 0, 1 \end{bmatrix} \quad (6)$$

#### 1.1 Question1

From the question, the probability of patterns is  $P(\omega_1) = P(\omega_2) = \frac{1}{2}$ . With the equation (3),(5),(6), we have

$$d_1(x) - d_2(x) = \ln(P(\omega_1)) - \ln(P(\omega_2)) + (m_1 - m_2)^T C^{-1}x - \frac{1}{2} m_1^T C^{-1} m_1 + \frac{1}{2} m_2^T C^{-1} m_2 = 0$$

$$d_1(x) - d_2(x) = [-4, -4]\mathbf{x} - 1 + 25 = -4x_1 - 4x_2 + 24 = x_1 + x_2 - 6 = 0 \quad (7)$$

Thus, the equation for Bayesian Discriminant Interface is  $x_1 + x_2 - 6 = 0$ .

## 1.2 Question 2

With equation 7, the discrimination interface is shown in the following figure.

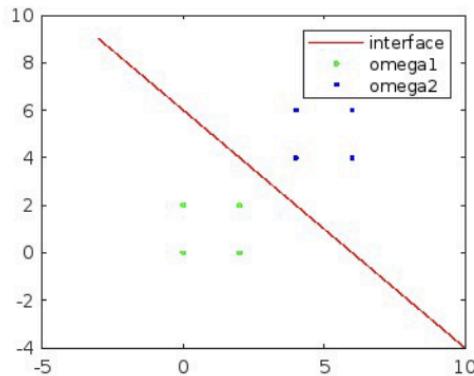


Figure 1: Discrimination Interface

## 判别式分类器



线性判别函数  
广义线性判别函数  
分段线性判别函数  
Fisher线性判别

感知器算法

采用感知器算法的多类模式的分类

可训练的确定性分类器的迭代算法  
势函数法 —— 一种确定性的非线性分类算法  
决策树简介

### ▼ 线性判别函数

两类问题

多类情况1

[多类情况1.doc](#)

多类情况2

[多类情况2.doc](#)

### 多类情况3

[多类情况3.doc](#)

### ▼ 广义线性函数

[线性判别函数&二次多项式函数.doc](#)

■对于 $n$ 维 $x$ 向量，若用 $r$ 次多项式， $d(x)$ 的权系数的总项数为：

$$N_w = C_{n+r}^r = \frac{(n+r)!}{r!n!}$$

### ▼ Fisher线性判别

如何根据实际情况找到一条最好的、最易于分类的投影线，这就是Fisher判别方法所要解决的基本问题。

Fisher线性判别 (FLD, Fisher Linear Discriminant) 是一种经典的用于分类的算法，尤其用于**二分类问题**。它通过寻找一个最佳的线性变换，将数据投影到一个一维空间，使得不同类别之间的可分性最大化。Fisher线性判别是一种基于**类间散度**和**类内散度**的线性判别方法。

Fisher提出了一个准则，来度量类别之间的差异以及类内的紧密程度。



### 类间散度 (Between-class scatter) :

类间散度衡量了不同类别之间的分离程度。若不同类别的均值距离较远，则类间散度较大。

### 类内散度 (Within-class scatter) :

类内散度衡量了同一类别内数据点的紧密程度。类内散度越小，说明同一类别的样本越集中。

Fisher线性判别通过最大化类间散度与类内散度的比值来优化投影方向，从而达到分类效果的最优化。

#### 2.1 类内散度矩阵 $S_W$

类内散度矩阵度量的是同一类别内的样本离其均值的散布情况。对于每个类别，类内散度矩阵  $S_W$  是各类样本离该类均值的平方距离的加权和。

假设  $N_1$  和  $N_2$  分别为类  $C_1$  和类  $C_2$  的样本数， $\mathbf{X}_1$  和  $\mathbf{X}_2$  分别为类  $C_1$  和类  $C_2$  的样本矩阵（每列为一个样本）。

类内散度矩阵  $S_W$  计算公式为：

$$S_W = S_1 + S_2 = \sum_{x_i \in C_1} (x_i - \mu_1)(x_i - \mu_1)^T + \sum_{x_j \in C_2} (x_j - \mu_2)(x_j - \mu_2)^T$$

其中， $\mu_1$  和  $\mu_2$  分别为类  $C_1$  和类  $C_2$  的均值。

写成矩阵形式： $S_b = (\mathbf{m}_1 - \mathbf{m}_2)(\mathbf{m}_1 - \mathbf{m}_2)^T$  为两类模式的类间散布

矩阵。[←](#)

对三个以上的类别，类间散布矩阵常写成：[←](#)

$$S_b = \sum_{i=1}^c P(\omega_i)(\mathbf{m}_i - \mathbf{m}_0)(\mathbf{m}_i - \mathbf{m}_0)^T \quad \leftarrow$$

#### 2.2 类间散度矩阵 $S_B$

类间散度矩阵度量的是不同类别均值之间的差异。其目标是使得类别的均值尽量远离。

类间散度矩阵  $S_B$  计算公式为：

$$S_B = N_1(\mu_1 - \mu)(\mu_1 - \mu)^T + N_2(\mu_2 - \mu)(\mu_2 - \mu)^T$$

其中， $\mu$  是所有样本的总均值， $\mu_1$  和  $\mu_2$  是两个类别的均值。

#### 2.3 目标优化准则

Fisher的目标是通过线性变换找到一个投影向量  $w$ ，使得在该投影下，类间散度与类内散度的比值最大化。换句话说，Fisher线性判别的目标是最大化下列准则函数：

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

其中， $w$  是我们需要优化的投影向量， $S_B$  是类间散度矩阵， $S_W$  是类内散度矩阵。

#### 2.4 求解优化问题

这个优化问题的解决方法是通过求解以下广义特征值问题：

$$S_W^{-1} S_B w = \lambda w$$

解出特征向量  $w$  对应于最大的特征值  $\lambda$ ，然后将数据投影到该方向上，即  $w^T x$ 。

名称	目标函数	优化算法
Fisher判别分析	$J(\mathbf{w}) = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$	解析解 $\mathbf{w} = \mathbf{S}_w^{-1}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)$

■两类Fisher判别的判别面为： $\mathbf{w}^T \mathbf{x} = \frac{1}{2} \mathbf{w}^T (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)$   
 • 投影后的类中心连线的垂直平分线为判别面  $(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \mathbf{S}_w^{-1} \mathbf{x} = \frac{1}{2} (\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2)^T \mathbf{S}_w^{-1} (\boldsymbol{\mu}_1 + \boldsymbol{\mu}_2)$



### Fisher线性判别与主成分分析 (PCA)

- 主成分分析 (PCA) 是一种无监督的降维方法，它的目标是通过最大化数据的方差来进行降维。PCA寻找的是数据中的主成分，而不考虑类别标签。
- Fisher线性判别 (FLD) 是一种有监督的降维方法，它通过最大化类间散度与类内散度的比值来寻找最佳的投影方向，从而在降维时最大化类之间的可分性。

计算过程：<https://blog.csdn.net/DOUBLE121PIG/article/details/96178580>

Fisher方法实现步骤总结：

1)把来自两类  $\omega_1 / \omega_2$  的训练样本集  $\mathbf{x}$  分成  
与  $\omega_1$  对应的子集  $X_1$  和与  $\omega_2$  对应的子集  $X_2$ ;

2)由  $\tilde{\mathbf{m}}_i = \frac{1}{N_i} \sum_j \tilde{\mathbf{x}}_j^{(i)} (i=1,2)$ , 计算  $\mathbf{m}_i$ ;

3)由  $\mathbf{S}_{\bar{\mathbf{w}}_i} = \sum_j (\tilde{\mathbf{x}}_j^{(i)} - \tilde{\mathbf{m}}_i)(\tilde{\mathbf{x}}_j^{(i)} - \tilde{\mathbf{m}}_i)^T$ ,

计算各类的类内离差阵  $\mathbf{S}_{\bar{\mathbf{w}}_1} \mathbf{S}_{\bar{\mathbf{w}}_2}$ ;

4)计算类内总离差阵  $\mathbf{S}_{\bar{\mathbf{w}}} = \mathbf{S}_{\bar{\mathbf{w}}_1} + \mathbf{S}_{\bar{\mathbf{w}}_2}$ ;

5)计算  $\mathbf{S}_{\bar{\mathbf{w}}}$  的逆矩阵  $\mathbf{S}_{\bar{\mathbf{w}}}^{-1}$ ;

6)按  $\vec{u} = \mathbf{S}_{\bar{\mathbf{w}}}^{-1}(\tilde{\mathbf{m}}_1 - \tilde{\mathbf{m}}_2)$  求解  $\vec{u}$ ;

7)计算  $\tilde{\mathbf{m}}_i : \tilde{\mathbf{m}}_i = \frac{1}{N_i} \sum_j y_j^{(i)} = \frac{1}{N_i} \sum_j \vec{u}^T \tilde{\mathbf{x}}_j^{(i)} = \vec{u}^T \tilde{\mathbf{m}}_i$

8)计算  $y_t = \frac{\tilde{\mathbf{m}}_1 + \tilde{\mathbf{m}}_2}{2}$

9)对未知模式  $\mathbf{x}$  判定模式类:

$\vec{u}^T \tilde{\mathbf{x}} = y > y_t \Rightarrow \tilde{\mathbf{x}} \in \omega_1$

$\vec{u}^T \tilde{\mathbf{x}} = y < y_t \Rightarrow \tilde{\mathbf{x}} \in \omega_2$

### ▼ 决策树

**分类：Gini指数**：Gini指数是决策树中常用的分类损失函数，用于衡量一个节点的不纯度。其值越小，表示节点越纯（即样本大部分属于同一类）。

**回归：L2损失（均方误差，MSE）：**L2损失是回归任务中常用的损失函数，用于度量预测值与真实值之间的误差。对于决策树回归，L2损失通常用于计算每个叶子节点的均值。

**正则化方法：叶子节点数目：**决策树的复杂度通常通过叶子节点的数目来控制。增加叶子节点数目通常会增加模型的复杂度，导致可能的过拟合。因此，限制叶子节点的数量是一个有效的正则化方法。

## Homework



- 在一个10类的模式识别问题中，有3类单独满足多类情况1，其余的类别满足多类情况2。问该模式识别问题所需判别函数的最少数目是多少？

$$3+7*6/2 = 24$$

- 一个三类问题，其判别函数如下：

$$d1(x) = -x_1, d2(x) = x_1 + x_2 - 1, d3(x) = x_1 - x_2 - 1$$

1. 设这些函数是在多类情况1条件下确定的，绘出其判别界面和每一个模式类别的区域。

2. 设为多类情况2，并使： $d_{12}(x) = d1(x)$ ,  $d_{13}(x) = d2(x)$ ,  $d_{23}(x) = d3(x)$ 。绘出其判别界面和多类情况2的区域。

3. 设 $d1(x)$ ,  $d2(x)$ 和 $d3(x)$ 是在多类情况3的条件下确定的，绘出其判别界面和每类的区域。

- 两类模式，每类包括5个3维不同的模式，且良好分布。如果它们是线性可分的，问权向量至少需要几个系数分量？假如要建立二次的多项式判别函数，又至少需要几个系数分量？（设模式的良好分布不因模式变化而改变。）

$$d(x) = \mathbf{w}^T \mathbf{x} + w_{n+1}$$

$$3+1 = 4$$

建立二次的多项式判别函数

$$d(x) = w_{11}x_1^2 + w_{22}x_2^2 + w_{33}x_3^2 + w_{12}x_1x_2 + w_{13}x_1x_3 + w_{23}x_2x_3 + w_1x_1 + w_2x_2 + w_3x_3 + w_4$$

10个参数



对于n维

$n$ 个一次； $n$ 个平方项； $\frac{n(n-1)}{2}$ 个二次；1个常量

## Logistic回归

### Homework

### SVM

- 函数间隔

$$y(\mathbf{w}^T \mathbf{x} + b)$$

- 几何间隔

$$y\left(\left(\frac{\mathbf{w}}{\|\mathbf{w}\|_2}\right)^T \mathbf{x} + \frac{b}{\|\mathbf{w}\|_2}\right)$$

## 几何间隔具有不变性

### 1. 硬线性SVM

■ 输入 : 线性可分的训练数据集  $S = \{(\mathbf{x}^i, y^i), i=1, \dots, N\}$

■ 输出 : 分离超平面和判别函数.

• 通过求解对偶问题来得到最优解  $\alpha^* = (\alpha_1^*, \dots, \alpha_N^*)$

$$\max_{\alpha} \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i,j=1}^N y^i y^j \alpha_i \alpha_j (\mathbf{x}^i)^T \mathbf{x}^j$$

$$s.t. \alpha_i \geq 0, i=1, \dots, N,$$

$$\sum_{i=1}^N \alpha_i y^i = 0.$$

• 得到原问题的最优解  $(\mathbf{w}^*, b^*)$

$$\mathbf{w}^* = \sum_{i=1}^N \alpha_i^* y^i \mathbf{x}^i, b^* = y^j - \sum_{i=1}^N \alpha_i^* y^i (\mathbf{x}^i)^T \mathbf{x}^j. \quad \alpha_i^* > 0$$

• 分离超平面:  $(\mathbf{w}^*)^T \mathbf{x} + b^* = 0$  , 判别函数:  $f_{w,b}(\mathbf{x}) = (\mathbf{w}^*)^T \mathbf{x} + b^*$ .

数据线性不可分 : 允许一些样本(离群点或噪声样本)违反原来的不等式约束条件, 但数目要少

### 2. 软间隔SVM

#### ■ 软间隔SVM

$$\text{合页损失: } \xi = L_{Hinge}(y, \hat{y}) = \begin{cases} 0 & y\hat{y} \geq 1 \\ 1 - y\hat{y} & \text{otherwise} \end{cases}$$

$$\cdot \text{ 原问题 } J(\mathbf{w}, b, C) = C \sum_{i=1}^N \xi_i + \frac{1}{2} \|\mathbf{w}\|_2^2$$

$$\text{s.t. } y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1 - \xi_i, \quad i = 1, 2, \dots, N$$

$$\xi_i \geq 0, \quad i = 1, 2, \dots, N$$

$$\cdot \text{ 对偶问题 } \max \left( \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \right)$$

$$\text{s.t. } \sum_{i=1}^N \alpha_i y_i = 0$$

$$0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, N$$

### 3. 核化SVM

证明高斯核函数可以升入无限维

多项式核

高斯核

径向基函数 (Radial Basis Function, RBF)

拉普拉斯核

Sigmoid核

## Homework

### 特征选择和提取

#### 特征选择 (Feature Selection)

特征选择是从原始特征中选取最相关、最重要的子集，从而减少特征空间的维度，去除冗余或无关的特征。通过特征选择，模型能够专注于最有信息的特征，避免过拟合并提高计算效率。

可分性准则函数

#### 特征提取 (Feature Extraction)

特征提取是通过某些方法将原始数据转换成新的、更有用的特征表示，以减少特征空间的维度并增强数据的可解释性。特征提取常用于处理高维数据，如图像、文本和时间序列数据。

类内散布矩阵-类间散布矩阵

### KL变换

K-L变换就是一种适用于任意概率密度函数的正交变换

K-L展开式的根本性质是将随机向量 $x$ 展开为另一组正交向量 $\phi_{ij}$ 的线性和

为使误差最小，不采用的特征向量，其对应的特征值应尽可能小

## Homework

### 统计学习基础

### 集成学习

### 聚类

聚类算法的基本思想是将一组数据集中的对象根据某些相似性度量，分成若干个不同的组（即簇），使得同一组中的对象之间的相似性尽可能大，而不同组之间的相似性尽可能小。

▼ 聚类



# K均值聚类

- 问题：给定 $N$ 个样本点 $X = \{\mathbf{x}_i\}_{i=1}^N$  进行聚类
  - 输入：数据  $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$ , 簇数目为 $K$
- 

1. 初始化选择 $K$ 个种子数据点(seeds)作为 $K$ 个簇的中心
  2. repeat
  3.     for each  $\mathbf{x}_i \in \mathcal{D}$  do
  4.         计算 $\mathbf{x}_i$ 与每一个簇中心的距离
  5.         将 $\mathbf{x}_i$ 指配到距离最近的簇中心
  6.     end for
  7.     用当前的簇内点重新计算 $K$ 个簇中心位置
  8. until 当前簇中心未更新
- 

## ■ 预处理

- 标准化数据
- 消除离群点 (outlier)

## ■ 后处理

- 删减小的簇：可能代表离群点
- 分裂松散的簇：簇内节点间距离之和很高
- 合并距离较近的簇



## 高斯混合模型：与K-means的联系

- 高斯混合模型的E步是一个软划分版本的 K-means:

$$r_{ik} \in [0,1]$$

- 高斯混合模型的M步估计除了估计均值外还估计协方差矩阵
- 当所有  $\pi_k$  相等,

$$\Sigma_k = \delta^2 I, \text{ 当 } \delta^2 \rightarrow 0, r_{ik} \rightarrow \{0, 1\}$$

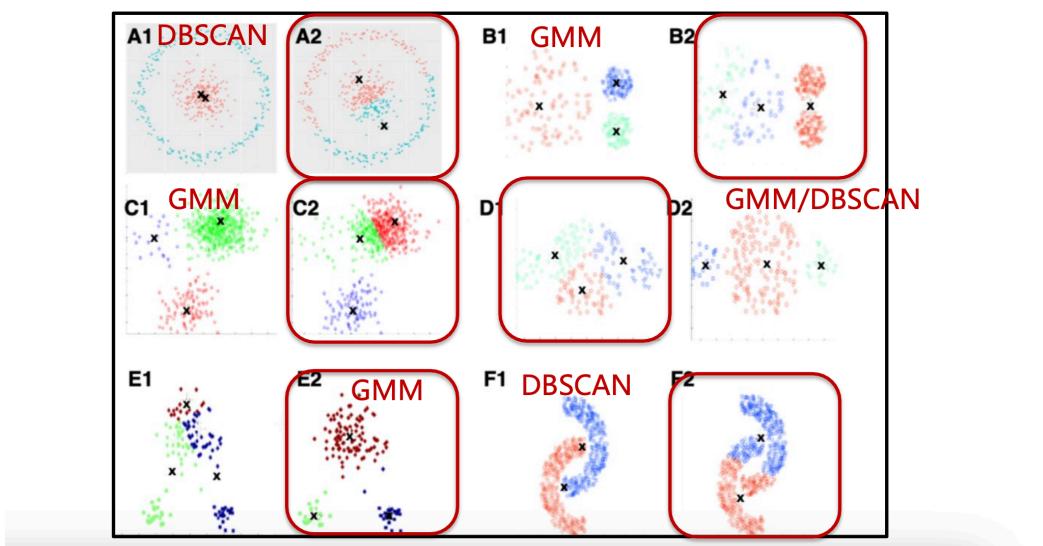
则两个方法是一致的。

### Homework

下图给出6个数据集A-F分别用两种算法得到的聚类结果，其中一种是K均值聚类。请问哪些最可能是K均值聚类的结果？

二维特征空间K均值决策界面：簇中心连线的中垂线

如果K均值聚类结果不够理想，建议采用哪种聚类算法？



### 降维

▼ ISOMAP(背！！！！！！)



# ISOMAP算法

数据集 $\mathbf{X}$ 中任意两个点的距离为 $d_{ij}$

- 构建邻接图:
  - 定义图 $G$ : 两个点 $i$ 和 $j$ 距离小于 $\epsilon(\epsilon - lsomap)$ , 或者 $i$ 是 $j$ 的 $K$ 近邻之一( $K-lsomap$ ), 连接这两个点, 且边的长度为 $d_{ij}$ 。
- 计算最短路径:
  - 如果 $i$ 和 $j$ 相连接, 初始化 $d_G(i,j) = d_{ij}$ ; 否则 $d_G(i,j) = \infty$ .
  - 对于每一个 $k = 1, 2, \dots, N$ , 替换所有的 $d_G(i,j)$ 为 $\min(d_G(i,j), d_G(i,k) + d_G(k,j))$ , 则 $D_G = [d_G(i,j)]$ 包含 $G$ 中所有点对的最短路径。
- 通过MDS构建低维的数据嵌入 (MDS只能得到样本点的坐标)
- 对于新数据点, 可以训练一个回归模型预测该点的低维表示: 高维空间中的坐标做为输入, 低维空间中的坐标做为输出, 有监督回归输入与输出之间的关系。

Homework

## ➤ 课后思考题

■ 简述PCA的主要过程 (训练与测试)。

■ 试简述MDS和ISOMAP的区别与联系。

■ 流形学习中的几何结构通常怎么表示?

半监督学习

Homework



# 作业

- 1. 请列举半监督学习常用的三种假设，并分别举例举出一种依据该假设而设计的半监督学习算法。
- 2. 简述直推式和归纳式半监督学习算法的异同，并分别例举出两种代表性算法。

## 人工神经网络



# 为什么ReLU?

- (1) 对于深层网络，sigmoid函数反向传播时，容易出现“梯度消失”（在sigmoid接近饱和区时，变换太缓慢，导数趋于0，无法完成深层网络的训练）
- (2) Relu会使一部分神经元的输出为0，这样就造成了网络的稀疏性，并且减少了参数的相互依存关系，缓解了过拟合问题的发生
- (3) sigmoid函数是指数运算，BP求误差梯度时，求导涉及除法，计算量相对大；而Relu是分段线性函数，节省计算量，且收敛较快