

## **Mathematical Basis**

对概率的阐述

频率派的观点

贝叶斯派的观点

高斯分布

一维情况 MLE

多维情况

Distance Metrics

极大似然估计(MLE)

特征值分解

奇异值分解

## **Clustering/聚类**

K-mean clustering

Hierarchical clustering/层次聚类

## **Adaptive learning/适应性学习**

Competitive learning(CL)/竞争性学习

Frequency sensitive competitive learning (FSCL) /频率敏感

Rival penalized competitive learning (RPCL)/竞争对手惩罚

## **EM算法/期望最大**

收敛性证明

公式导出

广义 EM

EM 的推广

## **Gaussian Mixture Models (GMM)/高斯混合模型**

概率知识基础

Bernoulli distribution/伯努利分布:

Conditional probability/条件概率:

Bayes Theorem/贝叶斯公式:

The Rules of Probability/常用公式:

先验/后验概率:

定义

极大似然估计

EM 求解 GMM

推导过程:

总结

## **Maximum Likelihood Estimation(MLE)/最大似然估计**

## **Maximum A Posteriori (MAP)/最大后验估计**

## **Model Selection**

AIC

BIC

AIC和BIC该如何选择?

Variational Bayes expectation Maximization (VBEM)

## **Principal Component Analysis (PCA)**

损失函数

SVD

Probabilistic PCA (p-PCA)

步骤

性质

## **Factor Analysis(FA)**

Model selection for PCA/FA

## **Preliminary on probability and statistics/概率与统计初探**

独立性: 联合分布可分解

不相关性:

Conditional independence/条件独立

高斯分布的性质

## Independent Component Analysis (ICA)

估计独立分量分布的关键是非高斯性

## PCA vs ICA

### Linear regression/线性回归

定义

最小二乘法

### Logistic regression/逻辑回归

决策边界 (Decision Boundary)

正则化

L1 Lasso

L2 Ridge

### Perceptron model/感知机模型

### Backpropagation (BP) algorithm

### CNN-卷积神经网络CNN

### Reference-批标准化 (Batch Normalization)

### 自编码器AutoEncoder

### 变分自编码器VAE

### GAN学习指南

### 支撑向量机

约束优化问题

Hard-margin SVM

Soft-margin SVM

Kernel Method

小结

### Causal inference and causal discovery

有向图-贝叶斯网络

贝叶斯网络与其结构学习算法

# Mathematical Basis

## 对概率的阐述

对概率的诠释有两大学派，一种是频率派另一种是贝叶斯派。后面我们对观测集采用下面记号：

$$X_{N \times p} = (x_1, x_2, \dots, x_N)^T, x_i = (x_{i1}, x_{i2}, \dots, x_{ip})^T$$

这个记号表示有  $N$  个样本，每个样本都是  $p$  维向量。其中每个观测都是由  $p(x|\theta)$  生成的。

## 频率派的观点

$p(x|\theta)$  中的  $\theta$  是一个常量。对于  $N$  个观测来说观测集的概率为  $p(X|\theta) = \prod_{i=1}^N p(x_i|\theta)$ 。为了求  $\theta$  的大

小，我们采用最大对数似然MLE的方法：

$$\theta_{MLE} = \underset{\theta}{\operatorname{argmax}} \log p(X|\theta) = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^N \log p(x_i|\theta)$$

## 贝叶斯派的观点

贝叶斯派认为  $p(x|\theta)$  中的  $\theta$  不是一个常量。这个  $\theta$  满足一个预设的先验的分布  $\theta \sim p(\theta)$ 。于是根据贝叶斯定理依赖观测集参数的后验可以写成：

$$p(\theta|X) = \frac{p(X|\theta) \cdot p(\theta)}{p(X)} = \frac{p(X|\theta) \cdot p(\theta)}{\int_{\theta} p(X|\theta) \cdot p(\theta) d\theta}$$

为了求  $\theta$  的值，我们要最大化这个参数后验MAP：

$$\theta_{MAP} = \underset{\theta}{argmax} p(\theta|X) = \underset{\theta}{argmax} p(X|\theta) \cdot p(\theta)$$

其中第二个等号是由于分母和  $\theta$  没有关系。求解这个  $\theta$  值后计算  $\frac{p(X|\theta) \cdot p(\theta)}{\int_{\theta} p(X|\theta) \cdot p(\theta) d\theta}$ ，就得到了参数的后验概率。

其中  $p(X|\theta)$  叫似然，是我们的模型分布。得到了参数的后验分布后，我们可以将这个分布用于预测贝叶斯预测：

$$p(x_{new}|X) = \int_{\theta} p(x_{new}|\theta) \cdot p(\theta|X) d\theta$$

其中积分中的被乘数是模型，乘数是后验分布。

## 高斯分布

### 一维情况 MLE

高斯分布在机器学习中占有举足轻重的作用。在 MLE 方法中：

$$\theta = (\mu, \Sigma) = (\mu, \sigma^2), \theta_{MLE} = \underset{\theta}{argmax} \log p(X|\theta) \stackrel{iid}{=} \underset{\theta}{argmax} \sum_{i=1}^N \log p(x_i|\theta)$$

一般地，高斯分布的概率密度函数PDF写为：

$$p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

带入 MLE 中我们考虑一维的情况

$$\log p(X|\theta) = \sum_{i=1}^N \log p(x_i|\theta) = \sum_{i=1}^N \log \frac{1}{\sqrt{2\pi}\sigma} \exp(-(x_i - \mu)^2 / 2\sigma^2)$$

首先对  $\mu$  的极值可以得到：

$$\mu_{MLE} = \underset{\mu}{argmax} \log p(X|\theta) = \underset{\mu}{argmax} \sum_{i=1}^N (x_i - \mu)^2$$

于是：

$$\frac{\partial}{\partial \mu} \sum_{i=1}^N (x_i - \mu)^2 = 0 \longrightarrow \mu_{MLE} = \frac{1}{N} \sum_{i=1}^N x_i$$

其次对  $\theta$  中的另一个参数  $\sigma$ ，有：

$$\begin{aligned}\sigma_{MLE} &= \underset{\sigma}{argmax} \log p(X|\theta) = \underset{\sigma}{argmax} \sum_{i=1}^N [-\log \sigma - \frac{1}{2\sigma^2}(x_i - \mu)^2] \\ &= \underset{\sigma}{argmin} \sum_{i=1}^N [\log \sigma + \frac{1}{2\sigma^2}(x_i - \mu)^2]\end{aligned}$$

于是：

$$\frac{\partial}{\partial \sigma} \sum_{i=1}^N [\log \sigma + \frac{1}{2\sigma^2}(x_i - \mu)^2] = 0 \longrightarrow \sigma_{MLE}^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \mu)^2$$

值得注意的是，上面的推导中，首先对  $\mu$  求 MLE，然后利用这个结果求  $\sigma_{MLE}$ ，因此可以预期的是对数据集求期望时  $\mathbb{E}_{\mathcal{D}}[\mu_{MLE}]$  是无偏差的：

$$\mathbb{E}_{\mathcal{D}}[\mu_{MLE}] = \mathbb{E}_{\mathcal{D}}[\frac{1}{N} \sum_{i=1}^N x_i] = \frac{1}{N} \sum_{i=1}^N \mathbb{E}_{\mathcal{D}}[x_i] = \mu$$

但是当对  $\sigma_{MLE}$  求期望的时候由于使用了单个数据集的  $\mu_{MLE}$ ，因此对所有数据集求期望的时候我们会发现  $\sigma_{MLE}$  是有偏的：

$$\begin{aligned}\mathbb{E}_{\mathcal{D}}[\sigma_{MLE}^2] &= \mathbb{E}_{\mathcal{D}}[\frac{1}{N} \sum_{i=1}^N (x_i - \mu_{MLE})^2] = \mathbb{E}_{\mathcal{D}}[\frac{1}{N} \sum_{i=1}^N (x_i^2 - 2x_i\mu_{MLE} + \mu_{MLE}^2)] \\ &= \mathbb{E}_{\mathcal{D}}[\frac{1}{N} \sum_{i=1}^N x_i^2 - \mu_{MLE}^2] = \mathbb{E}_{\mathcal{D}}[\frac{1}{N} \sum_{i=1}^N x_i^2] - \mu^2 + \mu^2 - \mu_{MLE}^2 \\ &= \mathbb{E}_{\mathcal{D}}[\frac{1}{N} \sum_{i=1}^N x_i^2] - \mathbb{E}_{\mathcal{D}}[\mu_{MLE}^2] = \sigma^2 - (\mathbb{E}_{\mathcal{D}}[\mu_{MLE}^2] - \mu^2) \\ &= \sigma^2 - (\mathbb{E}_{\mathcal{D}}[\mu_{MLE}^2] - \mathbb{E}_{\mathcal{D}}^2[\mu_{MLE}]) = \sigma^2 - Var[\mu_{MLE}] \\ &= \sigma^2 - Var[\frac{1}{N} \sum_{i=1}^N x_i] = \sigma^2 - \frac{1}{N^2} \sum_{i=1}^N Var[x_i] = \frac{N-1}{N} \sigma^2\end{aligned}$$

$$D(X) = E[X - E(X)]^2 = E(X^2) - E^2(X)$$

所以：

$$\hat{\sigma}^2 = \frac{1}{N-1} \sum_{i=1}^N (x_i - \mu)^2$$

## 多维情况

多维高斯分布表达式为：

$$p(x|\mu, \Sigma) = \frac{1}{(2\pi)^{p/2} |\Sigma|^{1/2}} e^{-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)}$$

其中  $x, \mu \in \mathbb{R}^p, \Sigma \in \mathbb{R}^{p \times p}$ ， $\Sigma$  为协方差矩阵，一般而言也是半正定矩阵。这里我们只考虑正定矩阵。  
首先我们处理指数上的数字，指数上的数字可以记为  $x$  和  $\mu$  之间的马氏距离。对于对称的协方差矩阵可进行特征值分解， $\Sigma = U \Lambda U^T = (u_1, u_2, \dots, u_p) diag(\lambda_i) (u_1, u_2, \dots, u_p)^T = \sum_{i=1}^p u_i \lambda_i u_i^T$ ，于是：

$$\Sigma^{-1} = \sum_{i=1}^p u_i \frac{1}{\lambda_i} u_i^T$$

$$\Delta = (x - \mu)^T \Sigma^{-1} (x - \mu) = \sum_{i=1}^p (x - \mu)^T u_i \frac{1}{\lambda_i} u_i^T (x - \mu) = \sum_{i=1}^p \frac{y_i^2}{\lambda_i}$$

我们注意到  $y_i$  是  $x - \mu$  在特征向量  $u_i$  上的投影长度，因此上式子就是  $\Delta$  取不同值时的同心椭圆。

下面我们看多维高斯模型在实际应用时的两个问题

1. 参数  $\Sigma, \mu$  的自由度为  $O(p^2)$  对于维度很高的数据其自由度太高。解决方案：高自由度的来源是  $\Sigma$  有  $\frac{p(p+1)}{2}$  个自由参数，可以假设其是对角矩阵，甚至在各向同性假设中假设其对角线上的元素都相同。前一种的算法有 Factor Analysis，后一种有概率 PCA(p-PCA)。
2. 第二个问题是单个高斯分布是单峰的，对有多个峰的数据分布不能得到好的结果。解决方案：高斯混合GMM 模型。

下面对多维高斯分布的常用定理进行介绍。

我们记  $x = (x_1, x_2, \dots, x_p)^T = (x_{a,m \times 1}, x_{b,n \times 1})^T, \mu = (\mu_{a,m \times 1}, \mu_{b,n \times 1}), \Sigma = \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix}$ , 已知  $x \sim \mathcal{N}(\mu, \Sigma)$ 。

首先是一个高斯分布的定理：

**定理：**已知  $x \sim \mathcal{N}(\mu, \Sigma), y \sim Ax + b$ , 那么  $y \sim \mathcal{N}(A\mu + b, A\Sigma A^T)$ 。

**证明：** $\mathbb{E}[y] = \mathbb{E}[Ax + b] = A\mathbb{E}[x] + b = A\mu + b$ ,  
 $Var[y] = Var[Ax + b] = Var[Ax] = A \cdot Var[x] \cdot A^T$ 。

**协方差公式：** $Var(a, b) = E[ab] - E[a]E[b]$

下面利用这个定理得到  $p(x_a), p(x_b), p(x_a|x_b), p(x_b|x_a)$  这四个量。

1.  $x_a = (\mathbb{I}_{m \times m} \quad \mathbb{O}_{m \times n}) \begin{pmatrix} x_a \\ x_b \end{pmatrix}$ , 代入定理中得到：

$$\mathbb{E}[x_a] = (\mathbb{I} \quad \mathbb{O}) \begin{pmatrix} \mu_a \\ \mu_b \end{pmatrix} = \mu_a$$

$$Var[x_a] = (\mathbb{I} \quad \mathbb{O}) \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix} \begin{pmatrix} \mathbb{I} \\ \mathbb{O} \end{pmatrix} = \Sigma_{aa}$$

所以  $x_a \sim \mathcal{N}(\mu_a, \Sigma_{aa})$ 。

2. 同样的,  $x_b \sim \mathcal{N}(\mu_b, \Sigma_{bb})$ 。

3. 对于两个条件概率, 我们引入三个量：

$$x_{b \cdot a} = x_b - \Sigma_{ba} \Sigma_{aa}^{-1} x_a$$

$$\mu_{b \cdot a} = \mu_b - \Sigma_{ba} \Sigma_{aa}^{-1} \mu_a$$

$$\Sigma_{bb \cdot a} = \Sigma_{bb} - \Sigma_{ba} \Sigma_{aa}^{-1} \Sigma_{ab}$$

特别的, 最后一个式子叫做  $\Sigma_{bb}$  的 Schur Complementary。可以看到：

$$x_{b \cdot a} = (-\Sigma_{ba} \Sigma_{aa}^{-1} \quad \mathbb{I}_{n \times n}) \begin{pmatrix} x_a \\ x_b \end{pmatrix}$$

所以：

$$\mathbb{E}[x_{b \cdot a}] = (-\Sigma_{ba} \Sigma_{aa}^{-1} \quad \mathbb{I}_{n \times n}) \begin{pmatrix} \mu_a \\ \mu_b \end{pmatrix} = \mu_{b \cdot a}$$

$$Var[x_{b \cdot a}] = (-\Sigma_{ba} \Sigma_{aa}^{-1} \quad \mathbb{I}_{n \times n}) \begin{pmatrix} \Sigma_{aa} & \Sigma_{ab} \\ \Sigma_{ba} & \Sigma_{bb} \end{pmatrix} \begin{pmatrix} -\Sigma_{aa}^{-1} \Sigma_{ba}^T \\ \mathbb{I}_{n \times n} \end{pmatrix} = \Sigma_{bb \cdot a}$$

利用这三个量可以得到  $x_b = x_{b \cdot a} + \Sigma_{ba} \Sigma_{aa}^{-1} x_a$ 。因此：

$$\mathbb{E}[x_b | x_a] = \mu_{b \cdot a} + \Sigma_{ba} \Sigma_{aa}^{-1} x_a$$

$$Var[x_b | x_a] = \Sigma_{bb \cdot a}$$

这里同样用到了定理。

4. 同样：

$$\begin{aligned}x_{a \cdot b} &= x_a - \Sigma_{ab} \Sigma_{bb}^{-1} x_b \\ \mu_{a \cdot b} &= \mu_a - \Sigma_{ab} \Sigma_{bb}^{-1} \mu_b \\ \Sigma_{aa \cdot b} &= \Sigma_{aa} - \Sigma_{ab} \Sigma_{bb}^{-1} \Sigma_{ba}\end{aligned}$$

所以：

$$\mathbb{E}[x_a | x_b] = \mu_{a \cdot b} + \Sigma_{ab} \Sigma_{bb}^{-1} x_b$$

$$Var[x_a | x_b] = \Sigma_{aa \cdot b}$$

下面利用上边四个量，求解线性模型：

已知： $p(x) = \mathcal{N}(\mu, \Lambda^{-1})$ ,  $p(y|x) = \mathcal{N}(Ax + b, L^{-1})$ , 求解： $p(y), p(x|y)$ 。

解：令  $y = Ax + b + \epsilon, \epsilon \sim \mathcal{N}(0, L^{-1})$ , 所以  $\mathbb{E}[y] = \mathbb{E}[Ax + b + \epsilon] = A\mu + b$ ,  $Var[y] = A\Lambda^{-1}A^T + L^{-1}$ , 因此：

$$p(y) = \mathcal{N}(A\mu + b, L^{-1} + A\Lambda^{-1}A^T)$$

引入  $z = \begin{pmatrix} x \\ y \end{pmatrix}$ , 我们可以得到  $Cov[x, y] = \mathbb{E}[(x - \mathbb{E}[x])(y - \mathbb{E}[y])^T]$ 。对于这个协方差可以直接计算：

$$Cov(x, y) = \mathbb{E}[(x - \mu)(Ax - A\mu + \epsilon)^T] = \mathbb{E}[(x - \mu)(x - \mu)^T A^T] = Var[x]A^T = \Lambda^{-1}A^T$$

注意到协方差矩阵的对称性，所以  $p(z) = \mathcal{N}\left(\begin{pmatrix} \mu \\ A\mu + b \end{pmatrix}, \begin{pmatrix} \Lambda^{-1} & \Lambda^{-1}A^T \\ A\Lambda^{-1} & L^{-1} + A\Lambda^{-1}A^T \end{pmatrix}\right)$ 。根据之前的公式，我们可以得到：

$$\mathbb{E}[x|y] = \mu + \Lambda^{-1}A^T(L^{-1} + A\Lambda^{-1}A^T)^{-1}(y - A\mu - b)$$

$$Var[x|y] = \Lambda^{-1} - \Lambda^{-1}A^T(L^{-1} + A\Lambda^{-1}A^T)^{-1}A\Lambda^{-1}$$

## Distance Metrics

Manhattan distance (city-block distance, L1 norm)	$d_{fg} = \sum_c  \mathbf{e}_{fc} - \mathbf{e}_{gc} $
Euclidean distance (L2 norm)	$d_{fg} = \sqrt{\sum_c (\mathbf{e}_{fc} - \mathbf{e}_{gc})^2}$
Mahalanobis distance	$d_{fg} = (\mathbf{e}_f - \mathbf{e}_g)' \Sigma^{-1} (\mathbf{e}_f - \mathbf{e}_g)$ , where $\Sigma$ is the (full or within-cluster) covariance matrix of the data
Pearson correlation (centered correlation)	$d_{fg} = 1 - r_{fg}$ , with $r_{fg} = \frac{\sum_c (\mathbf{e}_{fc} - \bar{e}_f)(\mathbf{e}_{gc} - \bar{e}_g)}{\sqrt{\sum_c (\mathbf{e}_{fc} - \bar{e}_f)^2 \sum_c (\mathbf{e}_{gc} - \bar{e}_g)^2}}$
Uncentered correlation (angular separation, cosine angle)	$d_{fg} = 1 - r_{fg}$ , with $r_{fg} = \frac{\sum_c \mathbf{e}_{fc} \mathbf{e}_{gc}}{\sqrt{\sum_c \mathbf{e}_{fc}^2 \sum_c \mathbf{e}_{gc}^2}}$
Spellman rank correlation	As Pearson correlation, but replace $\mathbf{e}_{gc}$ with the rank of $\mathbf{e}_{gc}$ within the expression values of gene $g$ across all conditions $c = 1 \dots C$
Absolute or squared correlation	$d_{fg} = 1 -  r_{fg} $ or $d_{fg} = 1 - r_{fg}^2$

## 极大似然估计(MLE)

## 特征值分解

适用范围：

特征值分解要求矩阵是方阵。SVD分解无此要求。

### 矩阵特征值分解的计算：

求方阵  $A_{n*n}$  的特征值：

$$\text{构造 } \phi(\lambda) = \det(A - \lambda I) = \begin{vmatrix} a_{11} - \lambda & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} - \lambda & \dots & a_{2n} \\ \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} - \lambda \end{vmatrix},$$

称  $\phi(\lambda) = 0$  为矩阵  $A_{n*n}$  的特征方程，易知  $\phi(\lambda) = 0$  为  $n$  次代数方程，求解该方程得到  $n$  个解  $\lambda_1, \lambda_2, \dots, \lambda_n$  即为方阵  $A_{n*n}$  的  $Q_{n*n}$   $n$  个特征值。 $\lambda_i$  可能为多重代数根，即存在  $i, j$ ，满足  $\lambda_i = \lambda_j$ 。 $\lambda_i$  对应的特征向量个数不大于  $\lambda_i$  的代数重数。

求方阵  $A_{n*n}$  的特征值  $\lambda_i$  对应的特征向量：

将  $\lambda_i$  依次代入相应的齐次方程  $(A - \lambda_i I)\vec{x} = 0$ ，得到的  $\vec{x}$  即为  $A_{n*n}$  的特征值  $\lambda_i$  对应的特征向量。

不同的特征  $A_{n*n}$  值对应的特征向量之间线性无关。多重代数根若存在多个特征向量，则这些特征向量也是线性无关的。

### 特征值分解：

方阵  $A_{n*n}$  可以做特征值分解的充要条件是其有  $n$  个线性无关的特征向量。即  $A_{n*n}$  有  $n$  个线性无关的特征向量时， $A_{n*n}$  才可以做特征值分解。

特征值分解的表达形式为： $A_{n*n} = Q_{n*n} \Lambda_{n*n} Q_{n*n}^{-1}$ ，其中  $Q_{n*n} = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$  是由  $A_{n*n}$  的列特征向量  $\vec{x}_i$  组成的非奇异矩阵（即可逆矩阵）， $\Lambda_{n*n}$  是由  $A_{n*n}$  的特征值  $\lambda_i$  组成的对角阵。 $\lambda_i$  和  $\vec{x}_i$  按顺序对应。

### 实对称阵和正交阵的性质：

实对称阵的性质：若  $A_{n*n}$  为实对称阵，则一定存在  $n$  个线性无关特征向量，即一定可以进行特征值分解。且必存在正交阵  $P_{n*n}$ ， $A_{n*n} = P_{n*n}^{-1} \Lambda_{n*n} P_{n*n} = A_{n*n} = P_{n*n}^T \Lambda_{n*n} P_{n*n}$ 。

$P_{n*n} = \{\vec{x}_1, \vec{x}_2, \dots, \vec{x}_n\}$  是由  $A_{n*n}$  的列特征向量  $\vec{x}_i$  进行单位化和正交化（可以使用施密特正交化方法）得到的可逆矩阵， $\Lambda_{n*n}$  是由  $A_{n*n}$  的特征值  $\lambda_i$  组成的对角阵。 $\lambda_i$  和  $\vec{x}_i$  按顺序对应。

正交阵的性质：若  $P_{n*n}$  为正交阵， $\lambda_i, \vec{p}_i$  为其特征值及其对应的特征向量，则  $1/\lambda_i, \vec{p}_i$  为  $P_{n*n}^{-1}$  的特征值和特征向量，且  $P_{n*n}^{-1} = P_{n*n}^T$ 。

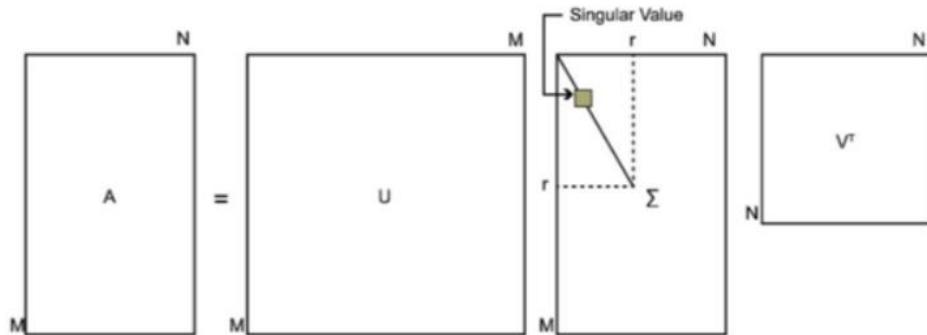
## 奇异值分解

---

$$A = U\Sigma V^T$$

其中  $U$  是一个  $m \times m$  的矩阵， $\Sigma$  是一个  $m \times n$  的矩阵，除了主对角线上的元素以外全为0，主对角线上的每个元素都称为奇异值， $V$  是一个  $n \times n$  的矩阵。 $U$  和  $V$  都是酉矩阵，即满足

$U^T U = I, V^T V = I$ 。下图可以很形象的看出上面SVD的定义：



那么我们如何求出SVD分解后的 $U, \Sigma, V$ 这三个矩阵呢？

如果我们将 $A$ 的转置和 $A$ 做矩阵乘法，那么会得到 $n \times n$ 的一个方阵  $A^T A$ 。既然  $A^T A$  是方阵，那么我们就可以进行特征分解，得到的特征值和特征向量满足下式：

$$(A^T A)v_i = \lambda_i v_i$$

这样我们就可以得到矩阵  $A^T A$  的n个特征值和对应的n个特征向量v了。将  $A^T A$  的所有特征向量张成一个 $n \times n$ 的矩阵V，就是我们SVD公式里面的V矩阵了。一般我们将V中的每个特征向量叫做A的右奇异向量。

如果我们将 $A$ 和 $A$ 的转置做矩阵乘法，那么会得到 $m \times m$ 的一个方阵  $AA^T$ 。既然  $AA^T$  是方阵，那么我们就可以进行特征分解，得到的特征值和特征向量满足下式：

$$(AA^T)u_i = \lambda_i u_i$$

这样我们就可以得到矩阵  $AA^T$  的m个特征值和对应的m个特征向量u了。将  $AA^T$  的所有特征向量张成一个 $m \times m$ 的矩阵U，就是我们SVD公式里面的U矩阵了。一般我们将U中的每个特征向量叫做A的左奇异向量。

U和V我们都求出来了，现在就剩下奇异值矩阵 $\Sigma$ 没有求出了。

由于 $\Sigma$ 除了对角线上是奇异值其他位置都是0，那我们只需要求出每个奇异值 $\sigma$ 就可以了。

我们注意到：

$$A = U\Sigma V^T \Rightarrow AV = U\Sigma V^T V \Rightarrow AV = U\Sigma \Rightarrow Av_i = \sigma_i u_i \Rightarrow \sigma_i = Av_i / u_i$$

这样我们可以求出我们的每个奇异值，进而求出奇异值矩阵 $\Sigma$ 。

上面还有一个问题没有讲，就是我们说  $A^T A$  的特征向量组成的就是我们SVD中的V矩阵，而

$AA^T$  的特征向量组成的就是我们SVD中的U矩阵，这有什么根据吗？这个其实很容易证明，我们以V矩阵的证明为例。

$$A = U\Sigma V^T \Rightarrow A^T = V\Sigma U^T \Rightarrow A^T A = V\Sigma U^T U\Sigma V^T = V\Sigma^2 V^T$$

上式证明使用了  $U^T = I, \Sigma^T = \Sigma$ 。可以看出  $A^T A$  的特征向量组成的确就是我们SVD中的V矩阵。类似的方法可以得到  $AA^T$  的特征向量组成的就是我们SVD中的U矩阵。

进一步我们还可以看出我们的特征值矩阵等于奇异值矩阵的平方，也就是说特征值和奇异值满足如下关系：

$$\sigma_i = \sqrt{\lambda_i}$$

这样也就是说，我们可以不用  $\sigma_i = \frac{Av_i}{u_i}$  来计算奇异值，也可以通过求出  $A^T A$  的特征值取平方根来求奇异值。

### 3. SVD计算举例

这里我们用一个简单的例子来说明矩阵是如何进行奇异值分解的。我们的矩阵A定义为：

$$\mathbf{A} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix}$$

首先求出  $A^T A$  和  $AA^T$

$$\mathbf{A}^T \mathbf{A} = \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 2 & 1 \\ 1 & 2 \end{pmatrix}$$

$$\mathbf{A} \mathbf{A}^T = \begin{pmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 & 1 \\ 1 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 2 & 1 \\ 0 & 1 & 1 \end{pmatrix}$$

进而求出  $A^T A$  的特征值和特征向量：

$$\lambda_1 = 3; v_1 = \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}; \lambda_2 = 1; v_2 = \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix}$$

接着求出  $AA^T$  的特征值和特征向量：

$$\lambda_1 = 3; u_1 = \begin{pmatrix} 1/\sqrt{6} \\ 2/\sqrt{6} \\ 1/\sqrt{6} \end{pmatrix}; \lambda_2 = 1; u_2 = \begin{pmatrix} 1/\sqrt{2} \\ 0 \\ -1/\sqrt{2} \end{pmatrix}; \lambda_3 = 0; u_3 = \begin{pmatrix} 1/\sqrt{3} \\ -1/\sqrt{3} \\ 1/\sqrt{3} \end{pmatrix}$$

利用  $Av_i = \sigma_i u_i, i = 1, 2$  求奇异值：

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} = \sigma_1 \begin{pmatrix} 1/\sqrt{6} \\ 2/\sqrt{6} \\ 1/\sqrt{6} \end{pmatrix} \Rightarrow \sigma_1 = \sqrt{3}$$

$$\begin{pmatrix} 0 & 1 \\ 1 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} -1/\sqrt{2} \\ 1/\sqrt{2} \end{pmatrix} = \sigma_2 \begin{pmatrix} 1/\sqrt{2} \\ 0 \\ -1/\sqrt{2} \end{pmatrix} \Rightarrow \sigma_2 = 1$$

也可以用  $\sigma_i = \sqrt{\lambda_i}$  直接求出奇异值为  $\sqrt{3}$  和 1.

最终得到 A 的奇异值分解为：

$$A = U \Sigma V^T = \begin{pmatrix} 1/\sqrt{6} & 1/\sqrt{2} & 1/\sqrt{3} \\ 2/\sqrt{6} & 0 & -1/\sqrt{3} \\ 1/\sqrt{6} & -1/\sqrt{2} & 1/\sqrt{3} \end{pmatrix} \begin{pmatrix} \sqrt{3} & 0 \\ 0 & 1 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 1/\sqrt{2} & 1/\sqrt{2} \\ -1/\sqrt{2} & 1/\sqrt{2} \end{pmatrix}$$

#### 4. SVD的一些性质

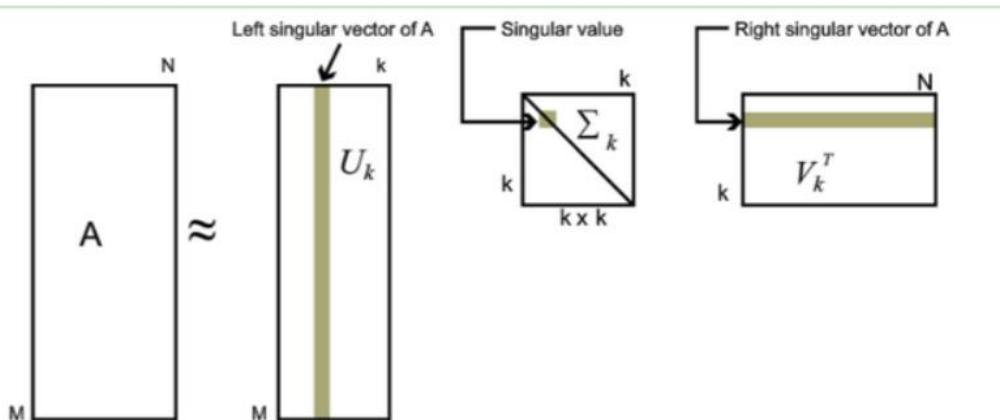
对于奇异值，它跟我们特征分解中的特征值类似，在奇异值矩阵中也是按照从大到小排列，而且奇异值的减少特别的快，在很多情况下，前 10% 甚至 1% 的奇异值的和就占了全部的奇异值之和的 99% 以上的比例。

也就是说，我们也可以用最大的 k 个的奇异值和对应的左右奇异向量来近似描述矩阵。

也就是说：

$$A_{m \times n} = U_{m \times m} \Sigma_{m \times n} V_{n \times n}^T \approx U_{m \times k} \Sigma_{k \times k} V_{k \times n}^T$$

其中 k 要比 n 小很多，也就是一个大的矩阵 A 可以用三个小的矩阵  $U_{m \times k}, \Sigma_{k \times k}, V_{k \times n}^T$  来表示。如下图所示，现在我们的矩阵 A 只需要灰色的部分的三个小矩阵就可以近似描述了。

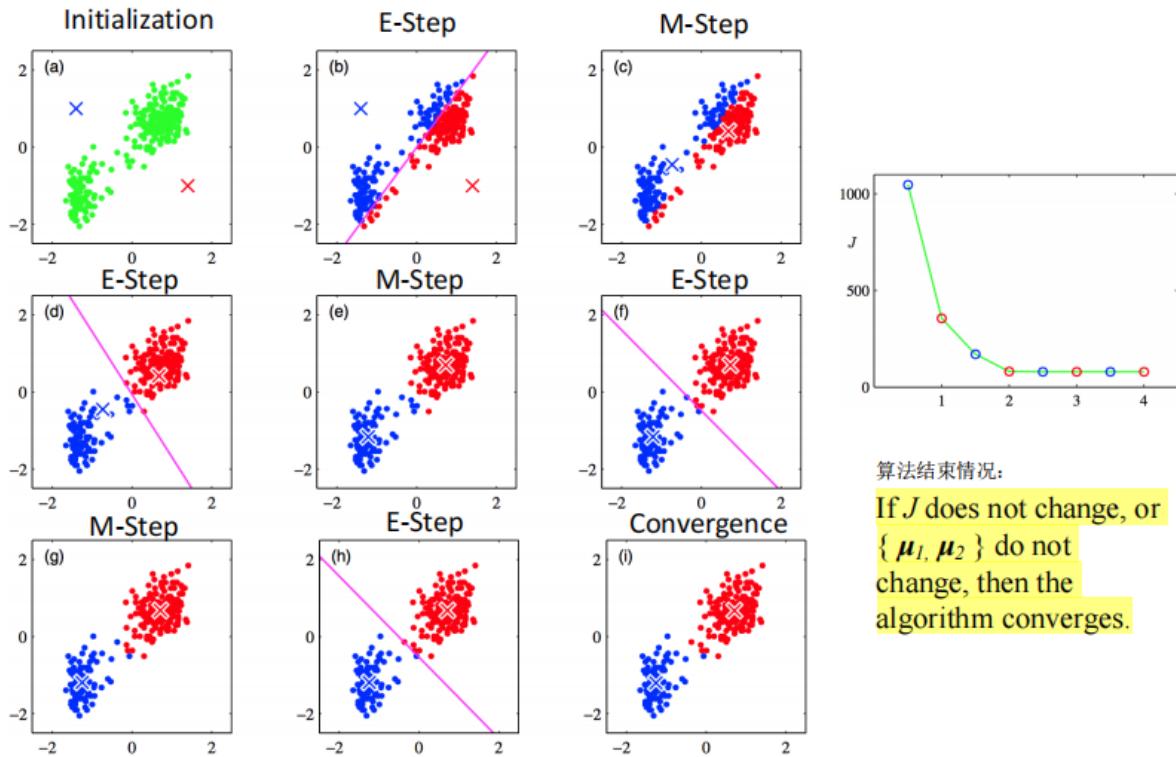


由于这个重要的性质，SVD 可以用于 PCA 降维，来做数据压缩和去噪。也可以用于推荐算法，将用户和喜好对应的矩阵做特征分解，进而得到隐含的用户需求来做推荐。同时也可以用于 NLP 中的算法，比如潜在语义索引 (LSI)。

下面我们就对 SVD 用于 PCA 降维做一个介绍。

# Clustering/聚类

## K-mean clustering



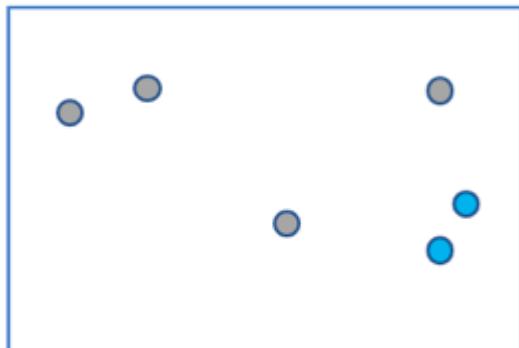
### Questions

- How many possible assignments for K-mean clustering?
  - $K^n$
- Can K-mean algorithm always converge? Why?
  - 能, 但不一定是全局最优
- Possible limitations of K-mean clustering?
  - K-Means的主要优点有: 1) 原理比较简单, 实现也是很容易, 收敛速度快。2) 聚类效果较优。3) 算法的可解释度比较强。4) 主要需要调参的参数仅仅是簇数k。
  - K-Means的主要缺点有: 1) **K值的选取不好把握**2) 对于**不是凸的数据集**比较难收敛。3) 如果各隐含类别的数据不平衡, 比如各**隐含类别的数据量严重失衡**, 或者各隐含类别的方差不同, 则聚类效果不佳。4) 采用迭代方法, 得到的结果只是**局部最优**。5) **对噪音和异常点比较的敏感**。

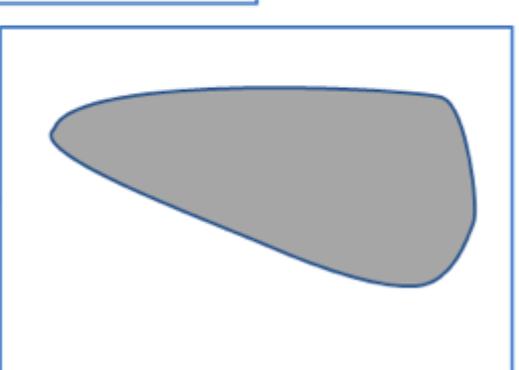
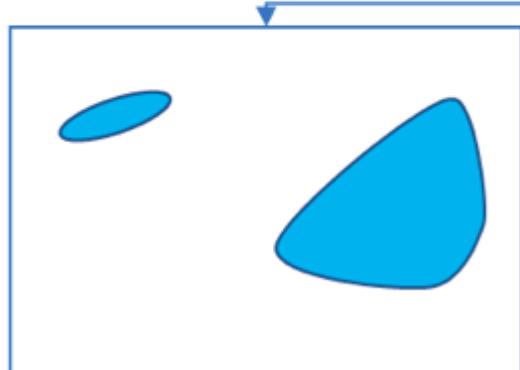
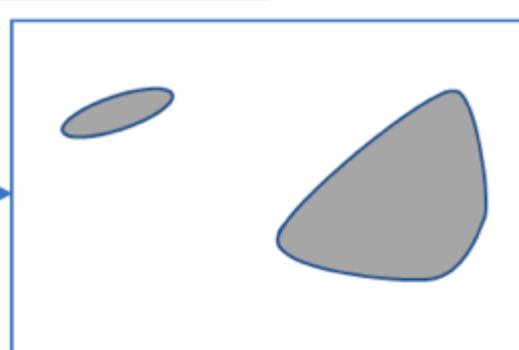
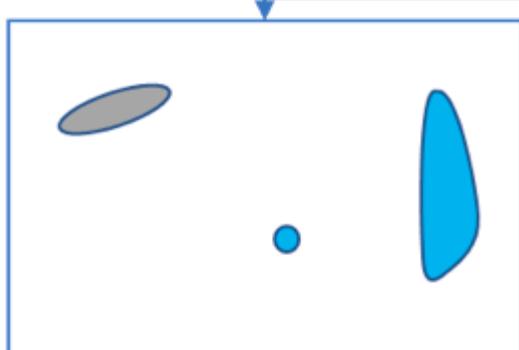
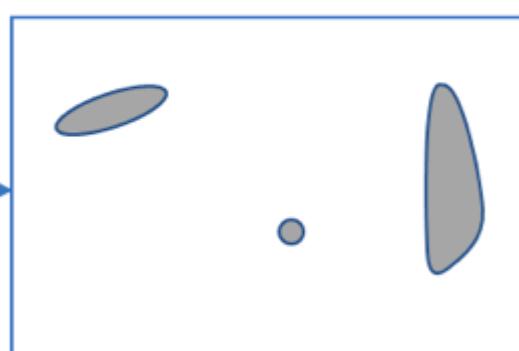
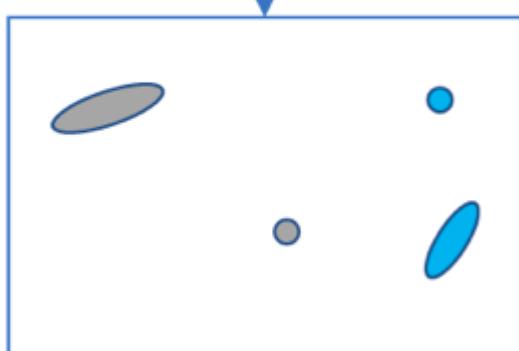
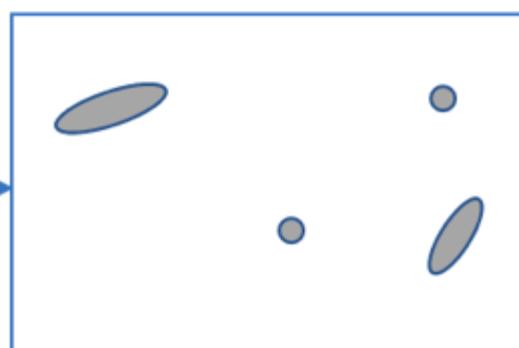
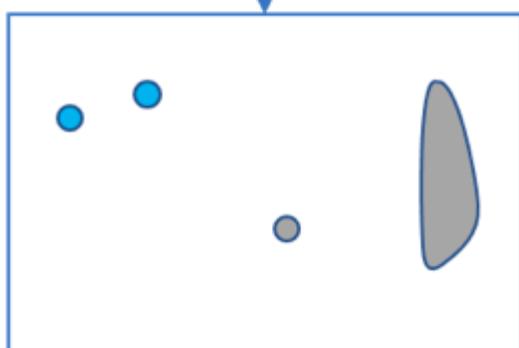
## Hierarchical clustering/层次聚类

分层聚类首先将每个观察值视为一个单独的聚类。然后, 它重复执行以下两个步骤: (1) 识别最接近的两个集群, 以及 (2) 合并两个最相似的集群。这个迭代过程一直持续到所有集群合并在一起。下图说明了这一点。

Identify the two clusters that  
are **closest** together



Merge the two most similar  
clusters



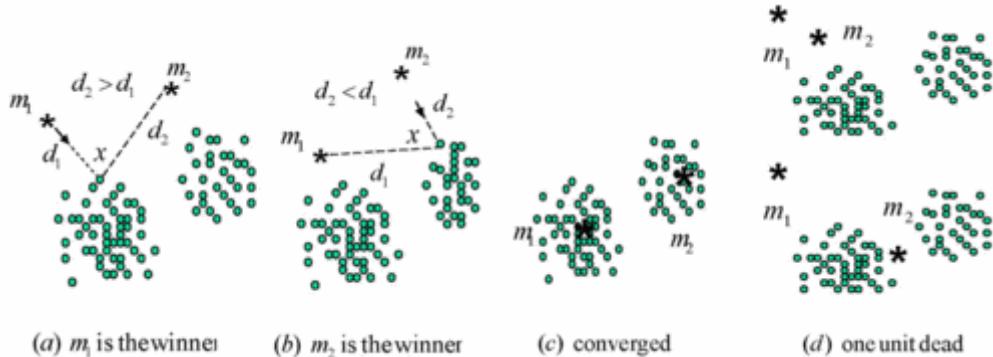
# Adaptive learning/适应性学习

## Competitive learning(CL)/竞争性学习

$$\varepsilon_t(\theta_j) = \|x_t - m_j\|^2$$

$$p_{j,t} = \begin{cases} 1, & \text{if } j = c, \\ 0, & \text{otherwise;} \end{cases} \quad c = \arg \min_j \varepsilon_t(\theta_j) \quad (1)$$

$$m_j^{new} = m_j^{old} + \eta p_{j,t}(x_t - m_j^{old}). \quad (2)$$



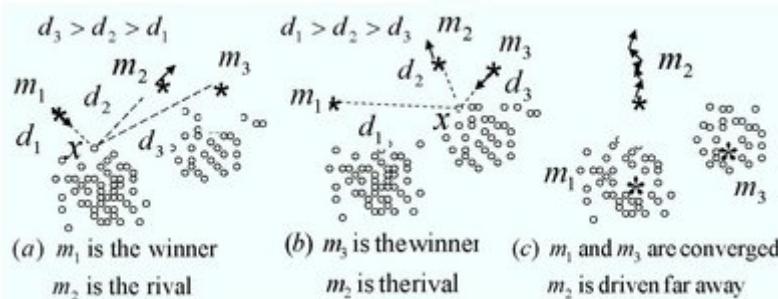
## Frequency sensitive competitive learning (FSCL) /频率敏感

$$\varepsilon_t(\theta_j) = \alpha_j \|x_t - m_j\|^2 \quad (3)$$

where  $\alpha_j$  is the frequency that the  $j$ -th agent won in past.

## Rival penalized competitive learning (RPCL)/竞争对手惩罚

$$p_{j,t} = \begin{cases} 1, & \text{if } j = c, \\ -\gamma, & \text{if } j = r, \\ 0, & \text{otherwise,} \end{cases} \quad \begin{cases} c = \arg \min_j \varepsilon_t(\theta_j), \\ r = \arg \min_{j \neq c} \varepsilon_t(\theta_j), \end{cases} \quad (4)$$



## Questions

- Are competitive learning (CL) and K-mean equivalent?
  -
- Could you come up with new algorithms to tackle the “bad initialization” problem of competitive learning (or K-mean)?
- Can you design a K-mean version of RPCL?

# EM算法/期望最大

## 收敛性证明

期望最大算法的目的是解决具有隐变量 $z$ 的混合模型的参数估计（极大似然估计）。MLE 对 $p(x|\theta)$ 参数的估计记为： $\theta_{\text{MLE}} = \arg\max_{\theta} \log p(x|\theta)$ 。EM 算法对这个问题的解决方法是采用迭代的方法：

$$\theta^{t+1} = \arg\max_{\theta} \int_z \log[p(x, z|\theta)] p(z|x, \theta^t) dz = \mathbb{E}_{z|x, \theta^t} [\log p(x, z|\theta)]$$

这个公式包含了迭代的两步：

1. E step：计算 $\log p(x, z|\theta)$ 在概率分布 $p(z|x, \theta^t)$ 下的期望

$$p(z|x, \theta^t) \rightarrow \mathbb{E}_{z|x, \theta^t} [\log p(x, z|\theta)]$$

2. M step：计算使这个期望最大化的参数得到下一个 EM 步骤的输入

$$\theta^{t+1} = \mathbb{E}_{z|x, \theta^t} [\log p(x, z|\theta)]$$

求证： $\log p(x|\theta^t) \leq \log p(x|\theta^{t+1})$

证明： $\log p(x|\theta) = \log \frac{p(z, x|\theta)}{p(z|x|\theta)} = \log p(z|x|\theta) - \log p(z|x|\theta)$ ，对左右两边求积分：

$$\text{Left : } \int_z p(z|x, \theta^t) \log p(x|\theta) dz = \int_z p(z|x, \theta^t) dz \cdot \log p(x|\theta) = \log p(x|\theta)$$

$$\text{Right : } \int_z p(z|x, \theta^t) \log p(x, z|\theta) dz - \int_z p(z|x, \theta^t) \log p(z|x, \theta) dz = Q(\theta, \theta^t) - H(\theta, \theta^t)$$

所以：

$$\log p(x|\theta) = Q(\theta, \theta^t) - H(\theta, \theta^t)$$

由于 $Q(\theta, \theta^t) = \int_z p(z|x, \theta^t) \log p(x, z|\theta) dz$ ，因而

$$\begin{aligned} Q(\theta^t, \theta^t) &= \int_z p(z|x, \theta^t) \log p(x, z|\theta^t) dz \\ Q(\theta^{t+1}, \theta^t) &= \int_z p(z|x, \theta^t) \log p(x, z|\theta^{t+1}) dz \end{aligned}$$

而根据定义， $\theta^{t+1} = \arg\max_{\theta} \int_z \log p(x, z|\theta) p(z|x, \theta^t) dz$ ，所以 $Q(\theta^{t+1}, \theta^t) \geq Q(\theta^t, \theta^t)$ 。此时要证 $\log p(x|\theta^t) \leq \log p(x|\theta^{t+1})$ ，只需证： $H(\theta^t, \theta^t) \geq H(\theta^{t+1}, \theta^t)$ ：

$$\begin{aligned} H(\theta^{t+1}, \theta^t) - H(\theta^t, \theta^t) &= \int_z p(z|x, \theta^t) \log p(z|x, \theta^{t+1}) dz - \int_z p(z|x, \theta^t) \log p(z|x, \theta^t) dz \\ &= \int_z p(z|x, \theta^t) \log \frac{p(z|x, \theta^{t+1})}{p(z|x, \theta^t)} = -KL(p(z|x, \theta^t), p(z|x, \theta^{t+1})) \leq 0 \end{aligned}$$

综合上面的结果：

$$\log p(x|\theta^t) \leq \log p(x|\theta^{t+1})$$

**KL散度：** $D_{\text{KL}}(P||Q) = \sum_x P(x) \log \frac{Q(x)}{P(x)}$ ，当且仅当 $P(x) = Q(x)$ 时等号成立

## 公式导出

根据上面的证明，我们看到，似然函数在每一步都会增大进一步的，我们看 EM 迭代过程中的式子是怎么来的：

$$\log p(x|\theta) = \log p(z, x|\theta) - \log p(z|x, \theta) = \log \frac{p(z, x|\theta)}{q(z)} - \log \frac{p(z|x, \theta)}{q(z)}$$

分别对两边求期望  $\mathbb{E}_q(z)$ ：

$$\begin{aligned}Left &: \int_z q(z) \log p(x|\theta) dz = \log p(x|\theta) \\Right &: \int_z q(z) \log \frac{p(z, x|\theta)}{q(z)} dz - \int_z q(z) \log \frac{p(z|x, \theta)}{q(z)} dz = ELBO + KL(q(z), p(z|x, \theta))\end{aligned}$$

上式中，Evidence Lower Bound(ELBO)，是一个下界，所以  $\log p(x|\theta) \geq ELBO$ ，等于号取在 KL 散度为0时，即： $q(z)=p(z|x,\theta)$ ，EM 算法的目的是将 ELBO 最大化，根据上面的证明过程，在每一步 EM 后，求得了最大的ELBO，并根据这个使 ELBO 最大的参数代入下一步中：

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} ELBO = \underset{\theta}{\operatorname{argmax}} \int_z q(z) \log \frac{p(x, z|\theta)}{q(z)} dz$$

由于  $q(z)=p(z|x,\theta^t)$  的时候，这一步的最大值才能取等号，所以：

$$\begin{aligned}\hat{\theta} &= \underset{\theta}{\operatorname{argmax}} ELBO = \underset{\theta}{\operatorname{argmax}} \int_z q(z) \log \frac{p(x, z|\theta)}{q(z)} dz = \underset{\theta}{\operatorname{argmax}} \int_z p(z|x, \theta^t) \log \frac{p(x, z|\theta)}{p(z|x, \theta^t)} dz \\&= \underset{\theta}{\operatorname{argmax}} \int_z p(z|x, \theta^t) \log p(x, z|\theta)\end{aligned}$$

这个式子就是上面 EM 迭代过程中的式子。

从 Jensen 不等式出发，也可以导出这个式子：

**Jensen 不等式**：若  $f(x)$  是 convex function(凸函数)，则  $E[f(x)] \geq f(E[x])$ .

$$\begin{aligned}\log p(x|\theta) &= \log \int_z p(x, z|\theta) dz = \log \int_z \frac{p(x, z|\theta)q(z)}{q(z)} dz \\&= \log \mathbb{E}_{q(z)} \left[ \frac{p(x, z|\theta)}{q(z)} \right] \geq \mathbb{E}_{q(z)} \left[ \log \frac{p(x, z|\theta)}{q(z)} \right]\end{aligned}$$

其中，右边的式子就是 ELBO，等号在  $p(x,z|\theta)=Cq(z)$  时成立。于是：

$$\begin{aligned}\int_z q(z) dz &= \frac{1}{C} \int_z p(x, z|\theta) dz = \frac{1}{C} p(x|\theta) = 1 \\&\Rightarrow q(z) = \frac{1}{p(x|\theta)} p(x, z|\theta) = p(z|x, \theta)\end{aligned}$$

我们发现，这个过程就是上面的最大值取等号的条件。

## 广义 EM

EM 模型解决了概率生成模型的参数估计的问题，通过引入隐变量  $z$ ，来学习  $\theta$ ，具体的模型对  $z$  有不同的假设。对学习任务  $p(x|\theta)$ ，就是学习任务  $\frac{p(x,z|\theta)}{p(z|x,\theta)}$ 。在这个式子中，我们假定了在 E 步骤中， $q(z)=p(z|x,\theta)$ ，但是这个  $p(z|x,\theta)$  如果无法求解，那么必须使用采样 (MCMC) 或者变分推断等方法来近似推断这个后验。我们观察 KL 散度的表达式，为了最大化 ELBO，在固定的  $\theta$  时，我们需要最小化 KL 散度，于是：

$$\hat{q}(z) = \underset{q}{\operatorname{argmin}} KL(p, q) = \underset{q}{\operatorname{argmax}} ELBO$$

这就是广义 EM 的基本思路：

1. E step:

$$\hat{q}^{t+1}(z) = \underset{q}{\operatorname{argmax}} \int_z q^t(z) \log \frac{p(x, z|\theta)}{q^t(z)} dz, \text{fixed } \theta$$

2. M step:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}} \int_z q^{t+1}(z) \log \frac{p(x, z|\theta)}{q^{t+1}(z)} dz, \text{fixed } \hat{q}$$

对于上面的积分：

$$ELBO = \int_z q(z) \log \frac{p(x, z|\theta)}{q(z)} dz = \mathbb{E}_{q(z)}[p(x, z|\theta)] + Entropy(q(z))$$

因此，我们看到，广义 EM 相当于在原来的式子中加入熵这一项。

## EM 的推广

EM 算法类似于坐标上升法，固定部分坐标，优化其他坐标，再一遍一遍的迭代。如果在 EM 框架中，无法求解 \$z\$ 后验概率，那么需要采用一些变种的 EM 来估算这个后验。

1. 基于平均场的变分推断，VBEM/VEM
2. 基于蒙特卡洛的EM，MCEM

## Gaussian Mixture Models (GMM)/高斯混合模型

### 概率知识基础

**Bernoulli distribution/伯努利分布：**

$$\begin{aligned} p(x=1|\mu) &= \mu \\ p(x=0|\mu) &= 1 - \mu \\ Bern(x|\mu) &= \mu^x(1-\mu)^{1-x} \\ E[x] &= \mu \\ var[x] &= \mu(1-\mu) \end{aligned}$$

**Conditional probability/条件概率：**

$$P(A|B) = \frac{P(A \cap B)}{P(B)}$$

**Bayes Theorem/贝叶斯公式：**

$$\begin{aligned} p(X|Y) &= \frac{p(X|Y)p(Y)}{P(X)} \\ p(X) &= \sum_Y p(X|Y)p(Y) \end{aligned}$$

## The Rules of Probability/常用公式:

$$p(X) = \sum_Y p(X, Y)$$
$$p(X, Y) = p(X|Y)p(Y)$$

### 先验/后验概率:

- $P(x)$  为  $x$  的先验概率 (prior probability)
- $P(x|Y)$  为  $x$  的后验概率

Remember: **The closer the distance, the more likely the probability.**

距离越近，概率越大

## 定义

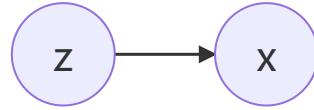
为了解决高斯模型的单峰性的问题，我们引入多个高斯模型的加权平均来拟合多峰数据：

$$p(x) = \sum_{k=1}^K \alpha_k \mathcal{N}(\mu_k, \Sigma_k)$$

引入隐变量  $z$ ，这个变量表示对应的样本  $x$  属于哪一个高斯分布，这个变量是一个离散的随机变量：

$$p(z = i) = p_i, \sum_{i=1}^k p(z = i) = 1$$

作为一个生成式模型，高斯混合模型通过隐变量  $z$  的分布来生成样本。用概率图来表示：



其中，节点  $z$  就是上面的概率， $x$  就是生成的高斯分布。于是对  $p(x)$ ：

$$p(x) = \sum_z p(x, z) = \sum_{k=1}^K p(x, z = k) = \sum_{k=1}^K p(z = k)p(x|z = k)$$

因此：

$$p(x) = \sum_{k=1}^K p_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

### One-dimensional Model

$$p(x) = \sum_{i=1}^K \phi_i \mathcal{N}(x | \mu_i, \sigma_i)$$
$$\mathcal{N}(x | \mu_i, \sigma_i) = \frac{1}{\sigma_i \sqrt{2\pi}} \exp\left(-\frac{(x - \mu_i)^2}{2\sigma_i^2}\right)$$
$$\sum_{i=1}^K \phi_i = 1$$

### Multi-dimensional Model

$$p(\vec{x}) = \sum_{i=1}^K \phi_i \mathcal{N}(\vec{x} | \vec{\mu}_i, \Sigma_i)$$

$$\mathcal{N}(\vec{x} | \vec{\mu}_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^K |\Sigma_i|}} \exp\left(-\frac{1}{2} (\vec{x} - \vec{\mu}_i)^T \Sigma_i^{-1} (\vec{x} - \vec{\mu}_i)\right)$$

$$\sum_{i=1}^K \phi_i = 1$$

## 极大似然估计

样本为  $X=(x_1, x_2, \dots, x_N)$ ,  $(X, Z)$  为完全参数, 参数为  $\theta = \{p_1, p_2, \dots, p_K, \mu_1, \mu_2, \dots, \mu_K, \Sigma_1, \Sigma_2, \dots, \Sigma_K\}$ 。我们通过极大似然估计得到  $\theta$  的值:

$$\begin{aligned}\theta_{MLE} &= \underset{\theta}{\operatorname{argmax}} \log p(X) = \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^N \log p(x_i) \\ &= \underset{\theta}{\operatorname{argmax}} \sum_{i=1}^N \log \sum_{k=1}^K p_k \mathcal{N}(x_i | \mu_k, \Sigma_k)\end{aligned}$$

这个表达式直接通过求导, 由于连加号的存在, 无法得到解析解。因此需要使用 EM 算法。

## EM 求解 GMM

Initialization Step:

- 从数据集  $X=\{x_1, \dots, x_N\}$  中随机分配样本到高斯分量的均值估计  $\hat{\mu}_1, \dots, \hat{\mu}_K$ , 并且无需替换。例如, 对于  $K=3$  和  $N=100$ , 设置  $\hat{\mu}_1 = x_{45}, \hat{\mu}_2 = x_{32}, \hat{\mu}_3 = x_{10}$ 。
- 将所有高斯分量方差估计设置为样本方差  $\hat{\sigma}_1^2, \dots, \hat{\sigma}_K^2 = \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})^2$ , 其中  $\bar{x}$  是样本均值  $\bar{x} = \frac{1}{N} \sum_{i=1}^N x_i$ 。
- 将所有高斯分量分布先验估计设置为均匀分布  $\hat{\phi}_1, \dots, \hat{\phi}_K = \frac{1}{K}$ 。

Expectation (E) Step:

Calculate  $\hat{\gamma}_{ik}$  for all  $i, k$

$$\hat{\gamma}_{ik} = \frac{\hat{\phi}_k \mathcal{N}(x_i | \hat{\mu}_k, \hat{\sigma}_k)}{\sum_{j=1}^K \hat{\phi}_j \mathcal{N}(x_i | \hat{\mu}_j, \hat{\sigma}_j)}$$

其中  $\hat{\gamma}_{ik}$  是  $x_i$  由组件  $C_k$  生成的概率。也就是说,  $\hat{\gamma}_{ik} = p(C_k | x_i, \hat{\phi}_k, \hat{\mu}_k, \hat{\sigma}_k)$ 。

Maximization (M) Step:

使用在期望步骤中计算的  $\hat{\gamma}_{ik}$ , 按顺序计算  $\hat{\phi}_k$  的以下内容:

- $\hat{\phi}_k = \frac{1}{N} \sum_{i=1}^N \hat{\gamma}_{ik}$
- $\hat{\mu}_k = \frac{1}{\sum_{i=1}^N \hat{\gamma}_{ik}} \sum_{i=1}^N \hat{\gamma}_{ik} x_i$
- $\hat{\sigma}_k^2 = \frac{1}{\sum_{i=1}^N \hat{\gamma}_{ik}} \sum_{i=1}^N \hat{\gamma}_{ik} (x_i - \hat{\mu}_k)^2$

当组件的数量  $K$  事先未知时, 通常会猜测组件的数量并使用 EM 算法将该模型拟合到数据中。这是针对许多不同的  $K$  值完成的。通常, 在拟合和组件数量之间具有最佳权衡的模型 (更简单的模型具有更少的组件) 会被保留。

## 推导过程：

EM 算法的基本表达式为：

$\theta^{t+1} = \operatorname{argmax}_{\theta} \sum_z p(x_i, z_i | \theta) \log p(z_i | x_i, \theta)$ 。套用 GMM 的表达式，对数据集来说：

$$\begin{aligned} Q(\theta, \theta^t) &= \sum_z \left[ \log \prod_{i=1}^N p(x_i, z_i | \theta) \right] \prod_{i=1}^N p(z_i | x_i, \theta^t) \\ &= \sum_z \left[ \sum_{i=1}^N \log p(x_i, z_i | \theta) \right] \prod_{i=1}^N p(z_i | x_i, \theta^t) \end{aligned}$$

对于中间的那个求和号，展开，第一项为：

$$\begin{aligned} \sum_z \log p(x_1, z_1 | \theta) \prod_{i=1}^N p(z_i | x_i, \theta^t) &= \sum_z \log p(x_1, z_1 | \theta) p(z_1 | x_1, \theta^t) \prod_{i=2}^N p(z_i | x_i, \theta^t) \\ &= \sum_{z_1} \log p(x_1, z_1 | \theta) p(z_1 | x_1, \theta^t) \sum_{z_2, \dots, z_N} \prod_{i=2}^N p(z_i | x_i, \theta^t) \\ &= \sum_{z_1} \log p(x_1, z_1 | \theta) p(z_1 | x_1, \theta^t) \end{aligned}$$

类似地， $Q$  可以写为：

$$Q(\theta, \theta^t) = \sum_{i=1}^N \sum_{z_i} \log p(x_i, z_i | \theta) p(z_i | x_i, \theta^t)$$

对于  $p(x, z | \theta)$ ：

$$p(x, z | \theta) = p(z | \theta) p(x | z, \theta) = p_z \mathcal{N}(x | \mu_z, \Sigma_z)$$

对  $p(z | x, \theta^t)$ ：

$$p(z | x, \theta^t) = \frac{p(x, z | \theta^t)}{p(x | \theta^t)} = \frac{p_z^t \mathcal{N}(x | \mu_z^t, \Sigma_z^t)}{\sum_k p_k^t \mathcal{N}(x | \mu_k^t, \Sigma_k^t)}$$

代入  $Q$ ：

$$Q = \sum_{i=1}^N \sum_{z_i} \log p_{z_i} \mathcal{N}(x_i | \mu_{z_i}, \Sigma_{z_i}) \frac{p_{z_i}^t \mathcal{N}(x_i | \mu_{z_i}^t, \Sigma_{z_i}^t)}{\sum_k p_k^t \mathcal{N}(x_i | \mu_k^t, \Sigma_k^t)}$$

下面需要对  $Q$  值求最大值：

$$Q = \sum_{k=1}^K \sum_{i=1}^N [\log p_k + \log \mathcal{N}(x_i | \mu_k, \Sigma_k)] p(z_i = k | x_i, \theta^t)$$

1.  $p_k^{t+1}$ ：

$$p_k^{t+1} = \operatorname{argmax}_{p_k} \sum_{k=1}^K \sum_{i=1}^N [\log p_k + \log \mathcal{N}(x_i | \mu_k, \Sigma_k)] p(z_i = k | x_i, \theta^t) \text{ s.t. } \sum_{k=1}^K p_k = 1$$

即：

$$p_k^{t+1} = \operatorname{argmax}_{p_k} \sum_{k=1}^K \sum_{i=1}^N \log p_k p(z_i = k | x_i, \theta^t) \text{ s.t. } \sum_{k=1}^K p_k = 1$$

引入 Lagrange 乘子:  $L(p_k, \lambda) = \sum \limits_{k=1}^K \sum \limits_{i=1}^N \log p_k(z_i=k|x_i, \theta^t) - \lambda(1 - \sum \limits_{k=1}^K p_k)$ 。所以:

$$\begin{aligned}\frac{\partial}{\partial p_k} L &= \sum_{i=1}^N \frac{1}{p_k} p(z_i=k|x_i, \theta^t) + \lambda = 0 \\ \Rightarrow \sum_k \sum_{i=1}^N \frac{1}{p_k} p(z_i=k|x_i, \theta^t) + \lambda \sum_k p_k &= 0 \\ \Rightarrow \lambda &= -N\end{aligned}$$

于是有:

$$p_k^{t+1} = \frac{1}{N} \sum_{i=1}^N p(z_i=k|x_i, \theta^t)$$

2.  $\mu_k, \Sigma_k$ , 这两个参数是无约束的, 直接求导即可。

## 总结

1. 也不一定能找到全局最优解, 和k-means类似, 只是局部最优并且优化依赖初始化。
2. GMM 比 K-means 更通用考虑混合权重, 协方差矩阵和软分配。
3. 像 K-means 一样, 它不会告诉你最好的 K。

## Maximum Likelihood Estimation(MLE)/最大似然估计

$$\begin{aligned}\theta_{MLE} &= \arg \max_{\theta} \log P(X|\theta) \\ &= \arg \max_{\theta} \log \prod_i P(x_i|\theta) \\ &= \arg \max_{\theta} \sum_i \log P(x_i|\theta)\end{aligned}$$

## Maximum A Posteriori (MAP)/最大后验估计

$$\begin{aligned}P(\theta|X) &= \frac{P(X|\theta)P(\theta)}{P(X)} \\ &\propto P(X|\theta)P(\theta) \\ \theta_{MAP} &= \arg \max_{\theta} P(X|\theta)P(\theta) \\ &= \arg \max_{\theta} \log P(X|\theta) + \log P(\theta) \\ &= \arg \max_{\theta} \log \prod_i P(x_i|\theta) + \log P(\theta) \\ &= \arg \max_{\theta} \sum_i \log P(x_i|\theta) + \log P(\theta)\end{aligned}$$

**MLE is a special case of MAP:** MLE是MAP的特殊情况

### When is MAP the same as MLE?

Answer:

当先验概率 $P(\theta)$ 为常数时。

# Model Selection

## AIC

赤池信息量准则 (Akaike information criterion, 简称AIC) 是评估统计模型的复杂度和衡量统计模型“拟合”资料之优良性(Goodness of fit)的一种标准，是由日本统计学家赤池弘次创立和发展的。赤池信息量准则建立在信息熵的概念基础上。

在一般的情况下，AIC可以表示为：

$$AIC = 2k - 2 \ln(L)$$

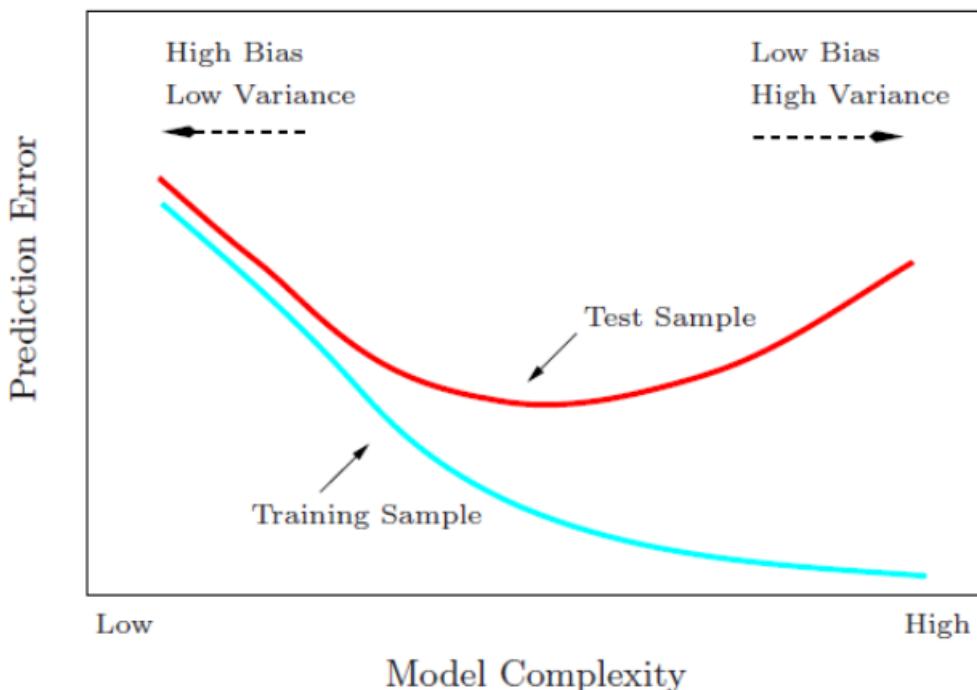
其中：

- $k$ 是参数的数量
- $L$ 是似然函数

**假设条件是模型的误差服从独立正态分布。**让 $n$ 为观察数，RSS为残差平方和，那么AIC变为：

$$AIC = 2k + n \ln(RSS/n)$$

增加自由参数的数目提高了拟合的优良性，AIC鼓励数据拟合的优良性但尽量避免出现过度拟合(Overfitting)的情况。所以优先考虑的模型应是AIC值最小的那个。赤池信息量准则的方法是寻找可以最好地解释数据但包含最少自由参数的模型。



当两个模型之间存在较大差异时，差异主要体现在似然函数项，当似然函数差异不显著时，上式第一项，即模型复杂度则起作用，从而参数个数少的模型是较好的选择。一般而言，当模型复杂度提高（ $k$ 增大）时，似然函数 $L$ 也会增大，从而使 $AIC$ 变小，但是 $k$ 过大时，似然函数增速减缓，导致 $AIC$ 增大，模型过于复杂容易造成过拟合现象。 $AIC$ 不仅要提高模型拟合度（极大似然），而且引入

了惩罚项，使模型参数尽可能少，有助于降低过拟合的可能性。

## BIC

贝叶斯信息准则，也称为Bayesian Information Criterion (BIC)。贝叶斯决策理论是主观贝叶斯派归纳理论的重要组成部分。是在不完全情报下，对部分未知的状态用主观概率估计，然后用贝叶斯公式对发生概率进行修正，最后再利用期望值和修正概率做出最优决策。公式为：

$$BIC = \ln(n)k - 2\ln(L)$$

其中，

- $k$ 为模型参数个数
- $n$ 为样本数量
- $L$ 为似然函数

$\ln(n)k$ 惩罚项在维数过大且训练样本数据相对较少的情况下，可以有效避免出现维度灾难现象。

与AIC相似，训练模型时，增加参数数量，也就是增加模型复杂度，会增大似然函数，但是也会导致过拟合现象，针对该问题，AIC和BIC均引入了与模型参数个数相关的惩罚项，BIC的惩罚项比AIC的大，考虑了样本数量，样本数量过多时，可有效防止模型精度过高造成的模型复杂度过高。

## AIC和BIC该如何选择？

AIC和BIC的原理是不同的，AIC是从预测角度，选择一个好的模型用来预测，BIC是从拟合角度，选择一个对现有数据拟合最好的模型，从贝叶斯因子的解释来讲，就是边际似然最大的那个模型。

共性

- 构造这些统计量所遵循的统计思想是一致的，就是在考虑拟合残差的同时，依自变量个数施加“惩罚”。

不同点

- BIC的惩罚项比AIC大，考虑了样本个数，样本数量多，可以防止模型精度过高造成的模型复杂度过高。
- AIC和BIC前半部分是一样的，BIC考虑了样本数量，样本数量过多时，可有效防止模型精度过高造成的模型复杂度过高。

AIC和BIC前半部分是惩罚项，当 $n \geq 8$ 时， $\ln(n) \geq 2k\ln(n) \geq 2k$ ，所以，**BIC相比AIC在大数据量时对模型参数惩罚得更多，导致BIC更倾向于选择参数少的简单模型。**

**奥卡姆剃刀原则**，粗略地说，人们应该更喜欢更简单的解释而不是更复杂的解释。

## Variational Bayes expectation Maximization (VBEM)

### 变分推断 (Variational Inference)

Gibbs采样：使用邻居结点的主题采样值

变分：采用相邻结点的期望。

这使得变分往往比采样算法更高效：用一次期望计算代替了大量的采样。直观上，均值的信息是高密(dense)的，而采样值的信息是稀疏(sparse)的。

变分既能够推断隐变量，也能推断未知参数，是非常有力的参数学习工具。其难点在于公式演算略复杂，和采样相对：一个容易计算但速度慢，一个不容易计算但运行效率高。

# Principal Component Analysis (PCA)

[Reference-降维-PCA](#)

## 损失函数

主成分分析中，我们的基本想法是将所有数据投影到一个子空间中，从而达到降维的目标，为了寻找这个子空间，我们基本想法是：

1. 所有数据在子空间中更为分散
2. 损失的信息最小，即：在补空间的分量少

原来的数据很有可能各个维度之间是相关的，于是我们希望找到一组  $p$  个新的线性无关的单位基  $u_j$ ，降维就是取其中的  $q$  个基。于是对于一个样本  $x_i$ ，经过这个坐标变换后：

$$\hat{x}_i = \sum_{j=1}^p (u_j^T x_i) u_j = \sum_{i=1}^q (u_j^T x_i) u_j + \sum_{j=q+1}^p (u_j^T x_i) u_j$$

对于数据集来说，我们首先将其中心化然后再去上面的式子的第一项，并使用其系数的平方平均作为损失函数并最大化：

$$\begin{aligned} J &= \frac{1}{N} \sum_{i=1}^N \sum_{j=1}^q ((x_i - \bar{x})^T u_j)^2 \\ &= \sum_{j=1}^q u_j^T S u_j, \text{ s.t. } u_j^T u_j = 1 \end{aligned}$$

为了方便，我们将协方差矩阵（数据集）写成中心化的形式：

$$\begin{aligned} S &= \frac{1}{N} \sum_{i=1}^N (x_i - \bar{x})(x_i - \bar{x})^T \\ &= \frac{1}{N} (x_1 - \bar{x}, x_2 - \bar{x}, \dots, x_N - \bar{x})(x_1 - \bar{x}, x_2 - \bar{x}, \dots, x_N - \bar{x})^T \\ &= \frac{1}{N} (X^T - \frac{1}{N} X^T \mathbb{I}_{N1} \mathbb{I}_{N1}^T) (X^T - \frac{1}{N} X^T \mathbb{I}_{N1} \mathbb{I}_{N1}^T)^T \\ &= \frac{1}{N} X^T (E_N - \frac{1}{N} \mathbb{I}_{N1} \mathbb{I}_{1N}) (E_N - \frac{1}{N} \mathbb{I}_{N1} \mathbb{I}_{1N})^T X \\ &= \frac{1}{N} X^T H_N H_N^T X \\ &= \frac{1}{N} X^T H_N H_N X = \frac{1}{N} X^T H X \end{aligned}$$

$H$  矩阵有如下性质：

$$\begin{aligned} H^T &= (E_N - \frac{1}{N} \mathbb{I}_{N1} \mathbb{I}_{1N})^T = (E_N - \frac{1}{N} \mathbb{I}_{N1} \mathbb{I}_{1N}) = H \\ H^2 &= (E_N - \frac{1}{N} \mathbb{I}_{N1} \mathbb{I}_{1N})^T (E_N - \frac{1}{N} \mathbb{I}_{N1} \mathbb{I}_{1N}) = H \end{aligned}$$

这个式子利用了中心矩阵  $H$  的对称性，这也是一个投影矩阵。

由于每个基都是线性无关的，于是每一个  $u_j$  的求解可以分别进行，使用拉格朗日乘子法：

$$\begin{aligned} \underset{u_j}{\operatorname{argmax}} L(u_j, \lambda) &= \underset{u_j}{\operatorname{argmax}} u_j^T S u_j + \lambda(1 - u_j^T u_j) \\ L(u_j, \lambda) &= u_j^T S u_j + \lambda(1 - u_j^T u_j) \\ \frac{\partial L}{\partial u_1} &= 2S u_1 - 2\lambda u_1 = 0 \end{aligned}$$

于是：

$$S u_j = \lambda u_j$$

## SVD

---

下面使用实际训练时常常使用的 SVD 直接求得这个  $q$  个特征向量。

对中心化后的数据集进行奇异值分解：

$$H X = U \Sigma V^T, U^T U = E_N, V^T V = E_p, \Sigma : N \times p$$

于是：

$$S = \frac{1}{N} X^T H X = \frac{1}{N} X^T H^T H X = \frac{1}{N} V \Sigma^T \Sigma V^T$$

因此，我们直接对中心化后的数据集进行 SVD，就可以得到特征值( $\Sigma$ 是特征值矩阵)和特征向量  $V$ ，在新坐标系中的坐标就是：

$$H X \cdot V$$

## Probabilistic PCA (p-PCA)

---

下面从概率的角度对 PCA 进行分析，概率方法也叫 p-PCA。我们使用线性模型，类似之前 LDA，我们选定一个方向，对原数据  $x \in \mathbb{R}^p$ ，降维后的数据为  $z \in \mathbb{R}^q, q < p$ 。降维通过一个矩阵变换（投影）进行：

$$\begin{aligned} z &\sim \mathcal{N}(\mathbb{O}_{q1}, \mathbb{I}_{qq}) \\ x &= Wz + \mu + \varepsilon \\ \varepsilon &\sim \mathcal{N}(0, \sigma^2 \mathbb{I}_{pp}) \end{aligned}$$

$$x \sim N(\mu, \omega \omega^T + \sigma^2)$$

对于这个模型，我么可以使用期望-最大 (EM) 的算法进行学习，在进行推断的时候需要求得  $p(z|x)$ ，推断的求解过程和线性高斯模型类似。

$$\begin{aligned} p(z|x) &= \frac{p(x|z)p(z)}{p(x)} \\ \mathbb{E}[x] &= \mathbb{E}[Wz + \mu + \varepsilon] = \mu \\ Var[x] &= WW^T + \sigma^2 \mathbb{I}_{pp} \\ \implies p(z|x) &= \mathcal{N}(W^T(WW^T + \sigma^2 \mathbb{I})^{-1}(x - \mu), \mathbb{I} - W^T(WW^T + \sigma^2 \mathbb{I})^{-1}W) \end{aligned}$$

##

## 步骤

---

总结一下 PCA 的算法步骤：

设有  $m$  条  $n$  维数据。

1. 将原始数据按列组成  $n$  行  $m$  列矩阵  $X$ ;
2. 将  $X$  的每一行进行零均值化，即减去这一行的均值;
3. 求出协方差矩阵  $C = \frac{1}{m}XX^T$ ;
4. 求出协方差矩阵的特征值及对应的特征向量;
5. 将特征向量按对应特征值大小从上到下按行排列成矩阵，取前  $k$  行组成矩阵  $P$ ;
6.  $Y = PX$  即为降维到  $k$  维后的数据。

## 性质

1. **缓解维度灾难**: PCA 算法通过舍去一部分信息之后能使得样本的采样密度增大（因为维数降低了），这是缓解维度灾难的重要手段;
2. **降噪**: 当数据受到噪声影响时，最小特征值对应的特征向量往往与噪声有关，将它们舍弃能在一定程度上起到降噪的效果;
3. **过拟合**: PCA 保留了主要信息，但这个主要信息只是针对训练集的，而且这个主要信息未必是重要信息。有可能舍弃了一些看似无用的信息，但是这些看似无用的信息恰好是重要信息，只是在训练集上没有很大的表现，所以 PCA 也可能加剧了过拟合;
4. **特征独立**: PCA 不仅将数据压缩到低维，它也使得降维之后的数据各特征相互独立;

## Algorithms for PCA

$$\Sigma_x = \frac{1}{N} \sum_{t=1}^N \mathbf{x}_t \mathbf{x}_t^T$$

- Eigen-decomposition  $\Sigma_x \mathbf{w} = (-\lambda) \cdot \mathbf{w}$
- SVD  $X = UDV^T$   $XX^T = UDV^T \cdot VDU^T = UD^2U^T$
- Hebbian learning rule  $\tau^W \frac{dW}{dt} = \bar{z}\bar{x}^t$
- Oja learning rule  $\tau^W \frac{dW}{dt} = \bar{z}\bar{x}^t - \bar{y}\bar{u}^t$
- Lmser rule  $\tau^W \frac{dW}{dt} = \bar{z}\bar{x}^t - \bar{y}\bar{u}^t + \bar{z}\bar{x}^t - \bar{y}^t\bar{x}^t$

$$\vec{z} = \vec{y} \quad \vec{y} = W\vec{x}, \vec{u} = W^t\vec{y}, \vec{y}^r = W\vec{u}$$

5

Hebbian:  $w$  值会趋向无穷大

Oja: 解决了 Hebbian 值趋向无穷大 (泰勒展开)

LSMER: 在此基础上进一步细化

## Factor Analysis(FA)

## Model selection for PCA/FA

# Preliminary on probability and statistics/概率与统计初探

## 独立性：联合分布可分解

$$p(y_1, y_2) = p_1(y_1)p_2(y_2)$$

$$E[h_1(y_1)h_2(y_2)] = E[h_1(y_1)]E[h_2(y_2)]$$

## 不相关性：

不相关的变量只是部分独立，不相关性不等于独立性

$$E[y_1y_2] - E[y_1]E[y_2] = 0$$

## Conditional independence/条件独立

$$p(X, Y|Z) = p(X|Z)p(Y|Z)$$

## 高斯分布的性质

1. 不相关性说明独立性
2. 由于中心极限定理，在许多情况下近似成立

中心极限定理 (Central Limit Theorem)：在一定条件下，独立随机变量之和的分布趋向于高斯分布。

3. 克莱默分解定理：让随机变量  $\xi$  服从正态分布，并允许分解为两个独立随机变量的和  $\xi=\xi_1+\xi_2$ 。那么被加数  $\xi_1$  和  $\xi_2$  也服从正态分布。

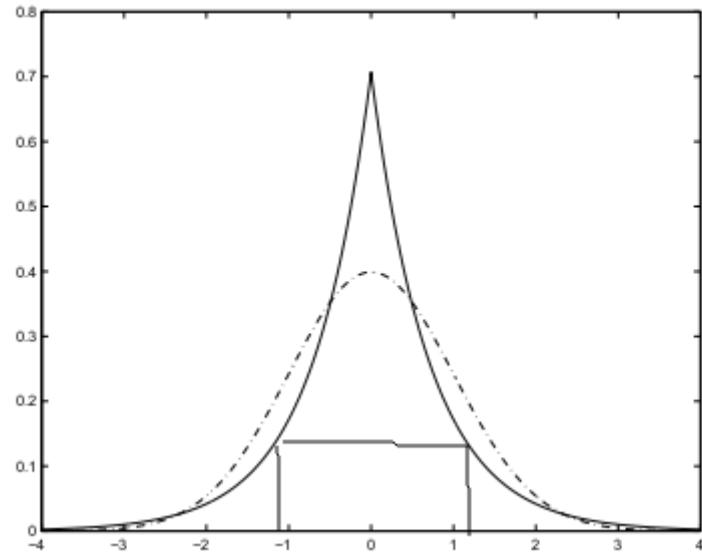
互信息(mutual information):统计相关性的自然度量

$$I(X;Y) = \sum_y \sum_x P(x,y) \log\left(\frac{P(x,y)}{P(x)P(y)}\right)$$
$$I(X;Y) = \int_y \int_x P(x,y) \log\left(\frac{P(x,y)}{P(x)P(y)}\right) dx dy$$

非负性，当且仅当X,Y互相独立时为0

Sub-Gaussian and Super-Gaussian

如果随机变量的分布是次高斯的 (subgaussian) 或者说是平峰的 (platykurtic)，峰度为负；超高斯的 (supergaussian) 或者说是尖峰的 (leptokurtic)，峰度为正。

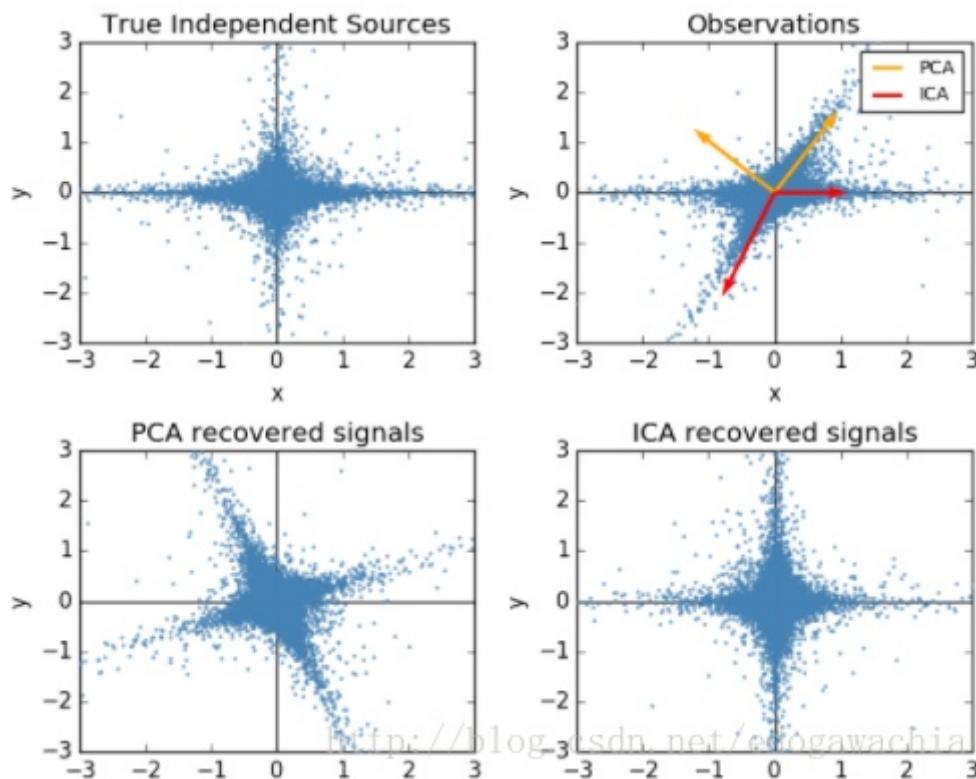


## Independent Component Analysis (ICA)

### 估计独立分量分布的关键是非高斯性

越是混合的变量，越是Gaussian，越是独立的，则Gaussian性越弱。因此，我们利用**非高斯性 (Nongaussianity)** 来度量信号之间的独立性。具体可以用**峰度 (kurtosis)** 或者**KL散度**。我们知道 KL-divergence 可以用来衡量两个概率分布的相似程度，因此可以计算出当前分布与高斯的相异程度。

[Reference-ICA \(Independent Component Analysis\) 独立成分分析](#)



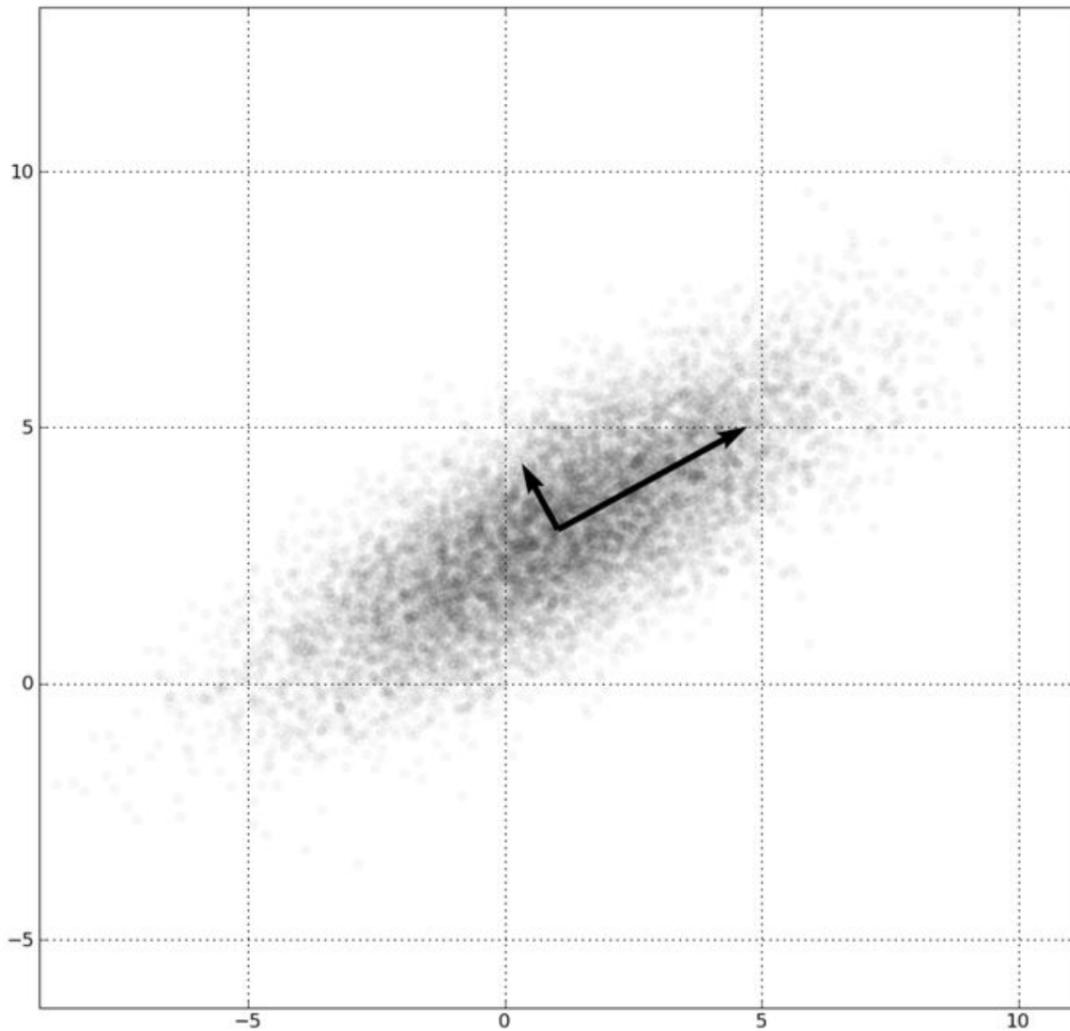
## PCA vs ICA

**不管是PCA还是ICA，都不需要你对源信号的分布做具体的假设；如果观察到的信号为高斯，那么源信号也为高斯，此时PCA和ICA等价。**

假设你观察到的信号是n维随机变量  $\mathbf{x} = (x_1, \dots, x_n)^T$ . 主成分分析 (PCA) 和独立成分分析 (ICA) 的目的都是找到一个方向, 即一个n维向量  $\mathbf{w} = (w_1, \dots, w_n)^T$  使得线性组合  $\sum_{i=1}^n w_i x_i = \mathbf{w}^T \mathbf{x}$  的某种特征最大化。

### 主成分分析 (PCA)

PCA认为一个随机信号最有用的信息体包含在方差里。为此我们需要找到一个方向  $\mathbf{w}_1$ , 使得随机信号  $\mathbf{x}$  在该方向上的投影  $\mathbf{w}_1^T \mathbf{x}$  的方差最大。接下来, 我们在与  $\mathbf{w}_1$  正交的空间里找到方向  $\mathbf{w}_2$ , 使得  $\mathbf{w}_2^T \mathbf{x}$  的方差最大, 以此类推直到找到所有的n个方向  $\mathbf{w}_1, \dots, \mathbf{w}_n$ . 用这种方法我们最终可以得到一列不相关的随机变量:  $\mathbf{w}_1^T \mathbf{x}, \dots, \mathbf{w}_n^T \mathbf{x}$ .



【图片来自wikipedia】

如果用矩阵的形式, 记  $\mathbf{W} = (\mathbf{w}_1, \dots, \mathbf{w}_n)$ , 那么本质上PCA是把原随机信号  $\mathbf{x}$  变换成了  $\mathbf{y} = \mathbf{W}\mathbf{x}$ , 其中  $\mathbf{y}$  满足下面的性质:

- $\mathbf{y}$  的各分量不相关;
- $y_1, \dots, y_n$  的方差递减。

特别地, 当原随机信号  $\mathbf{x}$  为高斯随机向量的时候, 得到的  $\mathbf{y}$  仍为高斯随机向量, 此时它的各个分量不

仅仅是线性无关的，它们还是独立的。

通过PCA，我们可以得到一列不相关的随机变量  $\mathbf{w}_1^T \mathbf{x}, \dots, \mathbf{w}_n^T \mathbf{x}$ ，至于这些随机变量是不是真的有意义，那必须根据具体情况具体分析。最常见的例子是，如果x的各分量的单位（量纲）不同，那么一般不能直接套用PCA。比如，若x的几个分量分别代表某国GDP, 人口, 失业率, 政府清廉指数，这些分量的单位全都不同，而且可以自行随意选取：GDP的单位可以是美元或者日元；人口单位可以是人或者千人或者百万人；失业率可以是百分比或者千分比，等等。对同一个对象（如GDP）选用不同的单位将会改变其数值，从而改变PCA的结果；而依赖“单位选择”的结果显然是没有意义的。

### 独立成分分析 (ICA)

ICA又称盲源分离(Blind source separation, BSS)，它假设观察到的随机信号x服从模型  $\mathbf{x} = \mathbf{As}$ ，其中s为未知源信号，其分量相互独立，A为一未知混合矩阵。ICA的目的是通过且仅通过观察x来估计混合矩阵A以及源信号s。

大多数ICA的算法需要进行“数据预处理”(data preprocessing)：先用PCA得到y，再把y的各个分量标准化（即让各分量除以自身的标准差）得到z。预处理后得到的z满足下面性质：

- z的各个分量不相关；
- z的各个分量的方差都为1。

有许多不同的ICA算法可以通过z把A和s估计出来。以著名的FastICA算法为例，该算法寻找方向  $\mathbf{w}$  使得随机变量  $\mathbf{w}^T \mathbf{z}$  的某种“非高斯性”(non-Gaussianity)的度量最大化。一种常用的非高斯性的度量是四阶矩  $\mathbb{E}[(\mathbf{w}^T \mathbf{x})^4]$ 。类似PCA的流程，我们首先找  $\mathbf{w}_1$  使得  $\mathbb{E}[(\mathbf{w}_1^T \mathbf{x})^4]$  最大；然后在与  $\mathbf{w}_1$  正交的空间里找  $\mathbf{w}_2$ ，使得  $\mathbb{E}[(\mathbf{w}_2^T \mathbf{x})^4]$  最大，以此类推直到找到所有的  $\mathbf{w}_1, \dots, \mathbf{w}_n$ 。可以证明，用这种方法得到的  $\mathbf{w}_1^T \mathbf{z}, \dots, \mathbf{w}_n^T \mathbf{z}$  是相互独立的。

ICA认为一个信号可以被分解成若干个统计独立的分量的线性组合，而后者携带更多的信息。我们可以证明，只要源信号非高斯，那么这种分解是唯一的。若源信号为高斯的话，那么显然可能有无穷多这样的分解。

=====一些技术细节=====

实际上PCA等价于求随机信号x的协方差矩阵的特征值分解(eigenvalue decomposition, EVD)或者奇异值分解(singular value decomposition, SVD)。比如，求  $\mathbf{w}_1$  的过程可以写成

$$\max_{\|\mathbf{w}\|=1} \text{Var}(\mathbf{w}^T \mathbf{x})$$

注意其中上式中包含欧氏范数为1的约束条件，这是因为如果没有这个约束条件那么右边方差可以无限大，这个问题因而也就变得没有意义了。现假设x的协方差矩阵C为已知，那么上式可以化为

$$\max_{\|\mathbf{w}\|=1} \mathbf{w}^T \mathbf{C} \mathbf{w}$$

不难看出这个问题的解  $\mathbf{w}_1$  为对应于矩阵C的最大的特征值  $\lambda_1$  的那个特征向量。

类似的，求第n个方向  $\mathbf{w}_k$  需要解

$$\max_{\|\mathbf{w}\|=1; \mathbf{w}^T \mathbf{w}_i=0, \forall i=1, \dots, k-1} \mathbf{w}^T \mathbf{C} \mathbf{w}$$

这个问题的解  $\mathbf{w}_k$  为对应于矩阵C的第k大的特征值  $\lambda_k$  的那个特征向量。

另外关于ICA，我们有下面的“ICA基本定理”：

**定理 (Pierre Comon, 1994)** 假设随机信号z服从模型  $\mathbf{z} = \mathbf{Bs}$ ，其中s的分量相互独立，且其

中至多可以有一个为高斯； $B$ 为满秩方阵。那么若 $z$ 的分量相互独立且仅当 $B=PD$ ，其中 $P$ 为排列矩阵(permuation matrix)， $D$ 为对角矩阵。

这个定理告诉我们，对于原信号 $x$ 做线性变换得到的新随机向量  $\mathbf{z} = \mathbf{Q}\mathbf{x}$ ，若 $z$ 的分量相互独立，那么 $z$ 的各个分量  $z_i$  一定对应于某个源信号分量  $s_j$  乘以一个系数。到这里，我们可以看到ICA的解具有内在的不确定性(inherent indeterminacy)。实际上，因为  $\mathbf{x} = \mathbf{As} = (\alpha\mathbf{A})(\alpha^{-1}\mathbf{s})$ ，即具备相同统计特征的 $x$ 可能来自两个不同的系统，这意味着单从观察 $x$ 我们不可能知道它来自于哪一个，从而我们就不可能推断出源信号 $s$ 的强度（方差）。为了在技术上消除这种不确定性，人们干脆约定源信号 $s$ 的方差为1。有了这个约定，再通过数据预处理的方法，我们可以把原混合矩阵 $A$ 化为一个自由度更低的正交矩阵：

数据预处理的过程又称为“数据白化”(data whitening)。这里预处理以后得到的 $z$ 和源信号 $s$ 的关系为

$\mathbf{z} = \mathbf{C}^{-1/2}\mathbf{As} = (\mathbf{AA}^T)^{-1/2}\mathbf{As}$ 。取  $\widetilde{\mathbf{A}} = (\mathbf{AA}^T)^{-1/2}\mathbf{A}$ ，则它可以看做一个新的混合矩阵。容易看出这是一个正交矩阵，它仅有  $n(n - 1)/2$  个自由度；而原混合矩阵一般有  $n^2$  个自由度。

## Linear regression/线性回归

### 定义

假设数据集为：

$$\mathcal{D} = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$$

后面我们记：

$$X = (x_1, x_2, \dots, x_N)^T, Y = (y_1, y_2, \dots, y_N)^T$$

线性回归假设：

$$f(w) = w^T x$$

### 最小二乘法

对这个问题，采用二范数定义的平方误差来定义损失函数：

$$L(w) = \sum_{i=1}^N \|w^T x_i - y_i\|_2^2$$

展开得到：

$$\begin{aligned} L(w) &= (w^T x_1 - y_1, \dots, w^T x_N - y_N) \cdot (w^T x_1 - y_1, \dots, w^T x_N - y_N)^T \\ &= (w^T X^T - Y^T) \cdot (Xw - Y) = w^T X^T X w - Y^T X w - w^T X^T Y + Y^T Y \\ &= w^T X^T X w - 2w^T X^T Y + Y^T Y \end{aligned}$$

最小化这个值的  $\hat{w}$ ：

$$\begin{aligned} \hat{w} &= \underset{w}{\operatorname{argmin}} L(w) \longrightarrow \frac{\partial}{\partial w} L(w) = 0 \\ &\longrightarrow 2X^T X \hat{w} - 2X^T Y = 0 \\ &\longrightarrow \hat{w} = (X^T X)^{-1} X^T Y = X^+ Y \end{aligned}$$

这个式子中  $(X^T X)^{-1} X^T$  又被称为伪逆。对于行满秩或者列满秩的  $X$ ，可以直接求解，但是对于非满秩的样本集合，需要使用奇异值分解（SVD）的方法，对  $X$  求奇异值分解，得到

$$X = U \Sigma V^T$$

于是：

$$X^+ = V \Sigma^{-1} U^T$$

在几何上，最小二乘法相当于模型（这里就是直线）和试验值的距离的平方求和，假设我们的试验样本张成一个  $p$  维空间（满秩的情况）： $X = \text{Span}(x_1, \dots, x_N)$ ，而模型可以写成  $f(w) = X\beta$ ，也就是  $x_1, \dots, x_N$  的某种组合，而最小二乘法就是说希望  $Y$  和这个模型距离越小越好，于是它们的差应该与这个张成的空间垂直：

$$X^T \cdot (Y - X\beta) = 0 \longrightarrow \beta = (X^T X)^{-1} X^T Y$$

## Logistic regression/逻辑回归

[Reference-逻辑回归\(Logistic Regression\)](#)

有时候我们只要得到一个类别的概率，那么我们需要一种能输出  $[0, 1]$  区间的值的函数。考虑两分类模型，我们利用判别模型，希望对  $p(C|x)$  建模，利用贝叶斯定理：

$$p(C_1|x) = \frac{p(x|C_1)p(C_1)}{p(x|C_1)p(C_1) + p(x|C_2)p(C_2)}$$

取  $a = \ln \frac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)}$ ，于是：

$$p(C_1|x) = \frac{1}{1 + \exp(-a)}$$

上面的式子叫 Logistic Sigmoid 函数，其参数表示了两类联合概率比值的对数。在判别式中，不关心这个参数的具体值，模型假设直接对  $a$  进行。

Logistic 回归的模型假设是：

$$a = w^T x$$

于是，通过寻找  $w$  的最佳值可以得到在这个模型假设下的最佳模型。概率判别模型常用最大似然估计的方式来确定参数。

对于一次观测，获得分类  $y$  的概率为（假定  $C_1=1, C_2=0$ ）：

$$p(y|x) = p_1^y p_0^{1-y}$$

那么对于  $N$  次独立全同的观测 MLE 为：

$$\begin{aligned} \hat{w} &= \underset{w}{\operatorname{argmax}} J(w) = \underset{w}{\operatorname{argmax}} \log(P(Y|X)) \\ &= \underset{w}{\operatorname{argmax}} \log \left( \prod_{i=1}^N P(y_i|x_i) \right) = \underset{w}{\operatorname{argmax}} \sum_{i=1}^N \log(P(y_i|x_i)) \\ &= \underset{w}{\operatorname{argmax}} \sum_{i=1}^N \log(p_1^y p_0^{1-y}) = \underset{w}{\operatorname{argmax}} \sum_{i=1}^N (y_i \log p_1 + (1 - y_i) \log p_0) \end{aligned}$$

注意到，这个表达式是交叉熵表达式的相反数乘  $N$ ，MLE 中的对数也保证了可以和指数函数相匹配，从而在大的区间汇总获取稳定的梯度。

对这个函数求导数，注意到：

$$p'_1 = \left( \frac{1}{1 + \exp(-a)} \right)' = p_1(1 - p_1)$$

则：

$$J'(w) = \sum_{i=1}^N y_i(1 - p_1)x_i - p_1x_i + y_ip_1x_i = \sum_{i=1}^N (y_i - p_1)x_i$$

由于概率值的非线性，放在求和符号中时，这个式子无法直接求解。于是在实际训练的时候，和感知机类似，也可以使用不同大小的批量随机梯度上升（对于最小化就是梯度下降）来获得这个函数的极大值。

## 决策边界 (Decision Boundary)

决策边界，也称为决策面，是用于在N维空间，将不同类别样本分开的平面或曲面。

注意：决策边界是假设函数的属性，由参数决定，而不是由数据集的特征决定。

- 线性决策边界
- 非线性决策边界

## 正则化

在实际应用时，如果样本容量不远远大于样本的特征维度，很可能造成过拟合，对这种情况，我们有下面三个解决方式：

1. 加数据
2. 特征选择（降低特征维度）如 PCA 算法。
3. 正则化

正则化一般是在损失函数（如上面介绍的最小二乘损失）上加入正则化项（表示模型的复杂度对模型的惩罚），下面我们介绍一般情况下的两种正则化框架。

$$L1 : \underset{w}{\operatorname{argmin}} L(w) + \lambda \|w\|_1, \lambda > 0$$

$$L2 : \underset{w}{\operatorname{argmin}} L(w) + \lambda \|w\|_2^2, \lambda > 0$$

### L1 Lasso

L1正则化可以引起稀疏解。

从最小化损失的角度看，由于 L1 项求导在0附近的左右导数都不是0，因此更容易取到0解。

从另一个方面看，L1 正则化相当于：

$$\begin{aligned} &\underset{w}{\operatorname{argmin}} L(w) \\ &\text{s.t. } \|w\|_1 < C \end{aligned}$$

平方误差损失函数在 \$w\$ 空间是一个椭球，因此上式求解就是椭球和 \$||w||\_1=C\$ 的切点，因此更容易相切在坐标轴上。

## L2 Ridge

$$\begin{aligned}\hat{w} = \underset{w}{\operatorname{argmin}} L(w) + \lambda w^T w &\longrightarrow \frac{\partial}{\partial w} L(w) + 2\lambda w = 0 \\ &\longrightarrow 2X^T X \hat{w} - 2X^T Y + 2\lambda \hat{w} = 0 \\ &\longrightarrow \hat{w} = (X^T X + \lambda \mathbb{I})^{-1} X^T Y\end{aligned}$$

利用2范数进行正则化不仅可以是模型选择  $w$  较小的参数，同时也避免  $X^T X$  不可逆的问题。

## Perceptron model/感知机模型

神经网络是多层感知机

**思想：错误驱动**

**模型：**  $f(x) = \operatorname{sign}(w^T x), x \in \mathbb{R}^p, w \in \mathbb{R}^p$

我们选取激活函数为：

$$\operatorname{sign}(a) = \begin{cases} +1, & a \geq 0 \\ -1, & a < 0 \end{cases}$$

这样就可以将线性回归的结果映射到两分类的结果上了。

定义损失函数为**错误分类的数目**，比较直观的方式是使用指示函数，但是指示函数不可导，因此可以定义：

$$L(w) = \sum_{x_i \in \mathcal{D}_{\text{wrong}}} -y_i w^T x_i$$

$$\begin{cases} w^T x_i > 0 \rightarrow y_i = 1 \\ w^T x_i < 0 \rightarrow y_i = -1 \end{cases} \Rightarrow y_i w^T x > 0 \text{ 代表正确分类时}$$

因此  $y_i w^T x < 0$  为错误分类

其中， $\mathcal{D}_{\text{wrong}}$  是错误分类集合，实际在每一次训练的时候，我们采用梯度下降的算法。损失函数对  $w$  的偏导为：

$$\frac{\partial}{\partial w} L(w) = \sum_{x_i \in \mathcal{D}_{\text{wrong}}} -y_i x_i$$

$$w^{t+1} \leftarrow w^t + \lambda y_i x_i$$

是可以收敛的，同时使用单个观测更新也可以在一定程度上增加不确定度，从而减轻陷入局部最小的可能。在更大规模的数据上，常用的是小批量随机梯度下降法。

**感知器收敛定理：**如果存在精确解(这意味着训练数据是线性可分的)，那么感知器学习算法保证在有限步数内找到精确解。

**定理：**具有线性输出的两层网络可以以任意精度一致地逼近紧致输入域上的任何连续函数，只要该网络具有足够多的隐藏单元

## Backpropagation (BP) algorithm

# CNN-卷积神经网络CNN

## Reference-批标准化 (Batch Normalization)

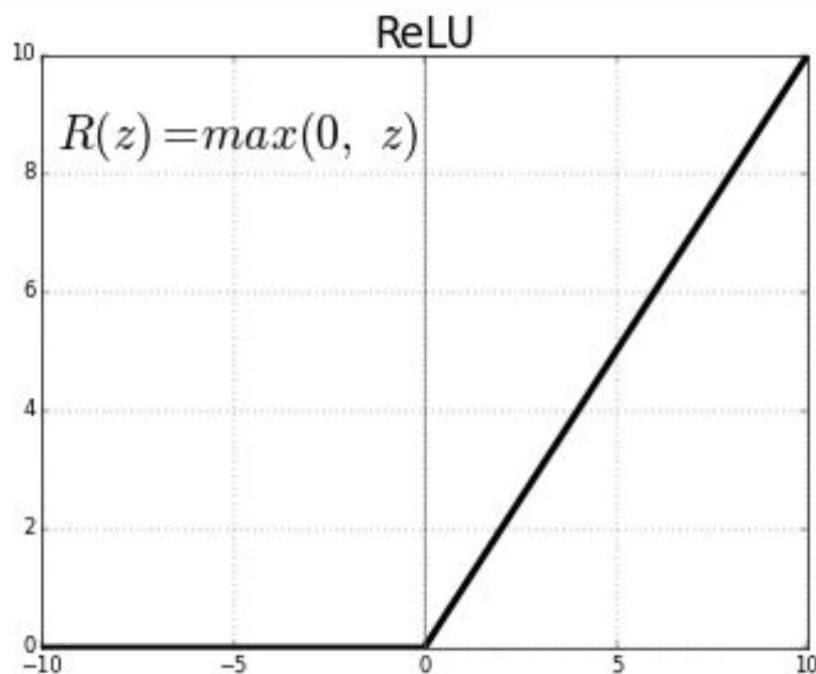
**Input:** Values of  $x$  over a mini-batch:  $\mathcal{B} = \{x_1 \dots m\}$ ;

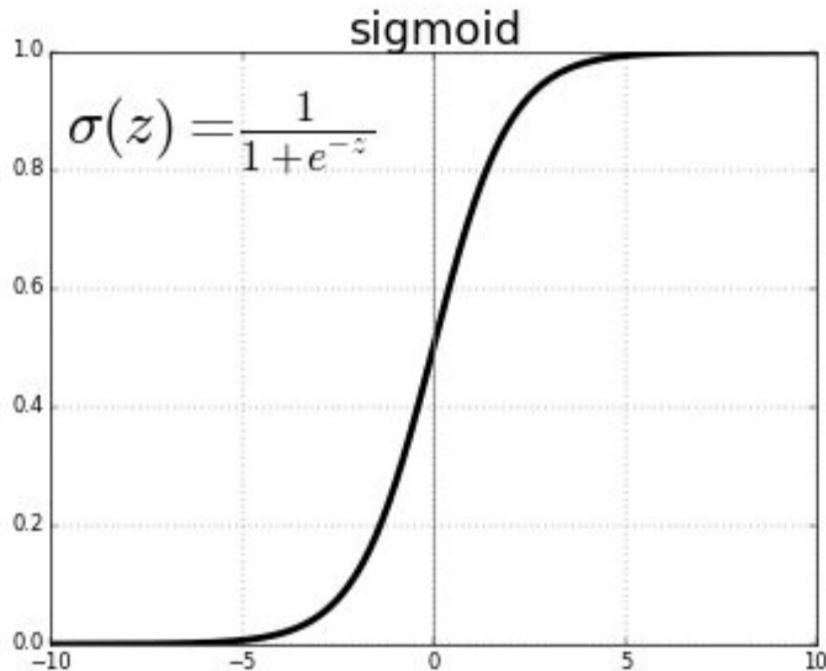
Parameters to be learned:  $\gamma, \beta$

**Output:**  $\{y_i = \text{BN}_{\gamma, \beta}(x_i)\}$

$$\begin{aligned}\mu_{\mathcal{B}} &\leftarrow \frac{1}{m} \sum_{i=1}^m x_i && // \text{mini-batch mean} \\ \sigma_{\mathcal{B}}^2 &\leftarrow \frac{1}{m} \sum_{i=1}^m (x_i - \mu_{\mathcal{B}})^2 && // \text{mini-batch variance} \\ \hat{x}_i &\leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} && // \text{normalize} \\ y_i &\leftarrow \gamma \hat{x}_i + \beta \equiv \text{BN}_{\gamma, \beta}(x_i) && // \text{scale and shift}\end{aligned}$$

**Algorithm 1:** Batch Normalizing Transform, applied to activation  $x$  over a mini-batch.





## Loss layer

- **L1, L2 loss**
- **Cross-Entropy loss**  
(works well for classification, e.g., image classification)
- **Hinge Loss**
- **Huber Loss**, more resilient to outliers with smooth gradient
- **Minimum Squared Error** (works well for regression task, e.g., Behavioral Cloning)

$$J = - \sum_{i=1}^N y_i \log(h_\theta(x_i)) + (1 - y_i) \log(1 - h_\theta(x_i))$$

$$p(y_i|x_i) = [h_\theta(x_i)]^{(y_i)}[1 - h_\theta(x_i)]^{(1-y_i)}$$

$$p(y_i = 1|x_i) = h_\theta(x_i) \quad p(y_i = 0|x_i) = 1 - h_\theta(x_i)$$

$$\sum_i \max(0, 1 - y_i * h_\theta(x_i))$$

$$L_\delta(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & \text{for } |y - f(x)| \leq \delta, \\ \delta |y - f(x)| - \frac{1}{2}\delta^2 & \text{otherwise.} \end{cases}$$

$$J = \frac{1}{N} \sum_{i=1}^N (y_i - h_\theta(x_i))^2$$

## 自编码器AutoEncoder

[Reference-Introduction to autoencoders.](#)

## 变分自编码器VAE

[Reference-Variational autoencoders](#)

## GAN学习指南

# 支撑向量机

[Reference-支持向量机SVM](#)

[Reference-SVM](#)

支撑向量机（SVM）算法在分类问题中有着重要地位，其主要思想是最大化两类之间的间隔。按照数据集的特点：

1. 线性可分问题，如之前的感知机算法处理的问题
2. 线性可分，只有一点点错误点，如感知机算法发展出来的 Pocket 算法处理的问题
3. 非线性问题，完全不可分，如在感知机问题发展出来的多层感知机和深度学习

这三种情况对于 SVM 分别有下面三种处理手段：

1. hard-margin SVM
2. soft-margin SVM
3. kernel Method

SVM 的求解中，大量用到了 Lagrange 乘子法，首先对这种方法进行介绍。

## 约束优化问题

一般地，约束优化问题（原问题）可以写成：

$$\begin{aligned} & \min_{x \in \mathbb{R}^p} f(x) \\ \text{s.t. } & m_i(x) \leq 0, i = 1, 2, \dots, M \\ & n_j(x) = 0, j = 1, 2, \dots, N \end{aligned}$$

定义 Lagrange 函数：

$$L(x, \lambda, \eta) = f(x) + \sum_{i=1}^M \lambda_i m_i(x) + \sum_{j=1}^N \eta_j n_j(x)$$

那么原问题可以等价于无约束形式：

$$\min_{x \in \mathbb{R}^p} \max_{\lambda, \eta} L(x, \lambda, \eta) \text{ s.t. } \lambda_i \geq 0$$

这是由于，当满足原问题的不等式约束的时候， $\lambda_i = 0$  才能取得最大值，直接等价于原问题，如果不满足原问题的不等式约束，那么最大值就为  $+\infty$ ，由于需要取最小值，于是不会取到这个情况。

这个问题的对偶形式：

$$\max_{\lambda, \eta} \min_{x \in \mathbb{R}^p} L(x, \lambda, \eta) \text{ s.t. } \lambda_i \geq 0$$

对偶问题是关于  $\lambda, \eta$  的最大化问题。

由于：

$$\max_{\lambda_i, \eta_j} \min_x L(x, \lambda_i, \eta_j) \leq \min_x \max_{\lambda_i, \eta_j} L(x, \lambda_i, \eta_j)$$

证明：显然有  $\min_x L \leq \max_x L$ ，于是显然有  $\max_{\lambda_i, \eta_j} \min_x L \leq \min_x \max_{\lambda_i, \eta_j} L$ ，且  $\min_x \max_{\lambda_i, \eta_j} L \geq L$ 。

对偶问题的解小于原问题，有两种情况：

1. 强对偶：可以取等于号
2. 弱对偶：不可以取等于号

上面介绍了原问题和对偶问题的对偶关系，但是实际还需要对参数进行求解，求解方法使用 KKT 条件进行：

KKT 条件和强对偶关系是等价关系。KKT 条件对最优解的条件为：

1. 可行域：

$$\begin{aligned} m_i(x^*) &\leq 0 \\ n_j(x^*) &= 0 \\ \lambda^* &\geq 0 \end{aligned}$$

2. 互补松弛  $\lambda^* m_i(x^*) = 0, \forall i$ ，对偶问题的最佳值为  $d^*$ ，原问题为  $p^*$

$$\begin{aligned} d^* &= \max_{\lambda, \eta} g(\lambda, \eta) = g(\lambda^*, \eta^*) \\ &= \min_x L(x, \lambda^*, \eta^*) \\ &\leq L(x^*, \lambda^*, \eta^*) \\ &= f(x^*) + \sum_{i=1}^M \lambda^* m_i(x^*) \\ &\leq f(x^*) = p^* \end{aligned}$$

为了满足相等，两个不等式必须成立，于是，对于第一个不等于号，需要有梯度为0条件，对于第二个不等于号需要满足互补松弛条件。

3. 梯度为0： $\frac{\partial L(x, \lambda^*, \eta^*)}{\partial x}|_{x=x^*} = 0$

## Hard-margin SVM

支撑向量机也是一种硬分类模型，在之前的感知机模型中，我们在线性模型的基础上叠加了符号函数，在几何直观上，可以看到，如果两类分的很开的话，那么其实会存在无穷多条线可以将两类分开。在 SVM 中，我们引入最大化间隔这个概念，间隔指的是数据和直线的距离的最小值，因此最大化这个值反映了我们的模型倾向。

分割的超平面可以写为：

$$0 = w^T x + b$$

那么最大化间隔（约束为分类任务的要求）：

$$\begin{aligned} &\underset{w, b}{\operatorname{argmax}} \left[ \min_i \frac{|w^T x_i + b|}{\|w\|} \right] \text{ s.t. } y_i(w^T x_i + b) > 0 \\ \implies &\underset{w, b}{\operatorname{argmax}} \left[ \min_i \frac{y_i(w^T x_i + b)}{\|w\|} \right] \text{ s.t. } y_i(w^T x_i + b) > 0 \end{aligned}$$

对于这个约束  $y_i(w^T x_i + b) > 0$ ，不妨固定  $\min_i y_i(w^T x_i + b) = 1 > 0$ ，这是由于分开两类的超平面的系数经过比例放缩不会改变这个平面，这也相当于给超平面的系数作出了约束。化简后的式子可以表示为：

$$\begin{aligned} &\underset{w, b}{\operatorname{argmin}} \frac{1}{2} w^T w \text{ s.t. } \min_i y_i(w^T x_i + b) = 1 \\ \Rightarrow &\underset{w, b}{\operatorname{argmin}} \frac{1}{2} w^T w \text{ s.t. } y_i(w^T x_i + b) \geq 1, i = 1, 2, \dots, N \end{aligned}$$

这就是一个包含  $N$  个约束的凸优化问题，有很多求解这种问题的软件。

但是，如果样本数量或维度非常高，直接求解困难甚至不可解，于是需要对这个问题进一步处理。引入 Lagrange 函数：

$$L(w, b, \lambda) = \frac{1}{2} w^T w + \sum_{i=1}^N \lambda_i (1 - y_i (w^T x_i + b))$$

我们有原问题就等价于：

$$\operatorname{argmin}_{w,b} \max_{\lambda} L(w, b, \lambda_i) \text{ s.t. } \lambda_i \geq 0$$

我们交换最小和最大值的符号得到对偶问题：

$$\max_{\lambda_i} \min_{w,b} L(w, b, \lambda_i) \text{ s.t. } \lambda_i \geq 0$$

由于不等式约束是仿射函数，对偶问题和原问题等价：

- \$b\$：  $\frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^N \lambda_i y_i = 0$
- \$w\$：首先将 \$b\$ 代入：

$$L(w, b, \lambda_i) = \frac{1}{2} w^T w + \sum_{i=1}^N \lambda_i (1 - y_i w^T x_i - y_i b) = \frac{1}{2} w^T w + \sum_{i=1}^N \lambda_i - \sum_{i=1}^N \lambda_i y_i w^T x_i$$

所以：

$$\frac{\partial}{\partial w} L = 0 \Rightarrow w = \sum_{i=1}^N \lambda_i y_i x_i$$

- 将上面两个参数代入：

$$L(w, b, \lambda_i) = -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j y_i y_j x_i^T x_j + \sum_{i=1}^N \lambda_i$$

因此，对偶问题就是：

$$\max_{\lambda} -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j y_i y_j x_i^T x_j + \sum_{i=1}^N \lambda_i, \text{ s.t. } \lambda_i \geq 0$$

从 KKT 条件得到超平面的参数：

原问题和对偶问题满足强对偶关系的充要条件为其满足 KKT 条件：

$$\begin{aligned} \frac{\partial L}{\partial w} &= 0, \frac{\partial L}{\partial b} = 0 \\ \lambda_k (1 - y_k (w^T x_k + b)) &= 0 \quad (\text{slackness complementary}) \\ \lambda_i &\geq 0 \\ 1 - y_i (w^T x_i + b) &\leq 0 \end{aligned}$$

根据这个条件就得到了对应的最佳参数：

$$\begin{aligned} \hat{w} &= \sum_{i=1}^N \lambda_i y_i x_i \\ \hat{b} &= y_k - w^T x_k = y_k - \sum_{i=1}^N \lambda_i y_i x_i^T x_k, \exists k, 1 - y_k (w^T x_k + b) = 0 \end{aligned}$$

于是这个超平面的参数 \$w\$ 就是数据点的线性组合，最终的参数值就是部分满足  $y_i (w^T x_i + b) = 1$  向量的线性组合（互补松弛条件给出），这些向量也叫支撑向量。

## Soft-margin SVM

Hard-margin 的 SVM 只对可分数据可解，如果不可分的情况，我们的基本想法是在损失函数中加入错误分类的可能性。错误分类的个数可以写成：

$$\text{error} = \sum_{i=1}^N \mathbb{I}\{y_i(w^T x_i + b) < 1\}$$

这个函数不连续，可以将其改写为：

$$\text{error} = \sum_{i=1}^N \max\{0, 1 - y_i(w^T x_i + b)\}$$

求和符号中的式子又叫做 Hinge Function。

将这个错误加入 Hard-margin SVM 中，于是：

$$\underset{w,b}{\operatorname{argmin}} \frac{1}{2} w^T w + C \sum_{i=1}^N \max\{0, 1 - y_i(w^T x_i + b)\} \text{ s.t. } y_i(w^T x_i + b) \geq 1 - \xi_i, i = 1, 2, \dots, N$$

这个式子中，常数  $C$  可以看作允许的错误水平，同时上式为了进一步消除  $\max$  符号，对数据集中的每一个观测，我们可以认为其大部分满足约束，但是其中部分违反约束，因此这部分约束变成  $y_i(w^T x_i + b) \geq 1 - \xi_i$ ，其中  $\xi_i \geq 0$ ，进一步的化简：

$$\underset{w,b}{\operatorname{argmin}} \frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i \text{ s.t. } y_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, 2, \dots, N$$

## Kernel Method

核方法可以应用在很多问题上，在分类问题中，对于严格不可分问题，我们引入一个特征转换函数将原来的数据集变为可分的数据集，然后再来应用已有的模型。往往将低维空间的数据集变为高维空间的数据集后，数据会变得可分（数据变得更为稀疏）：

Cover TH：高维空间比低维空间更易线性可分。

应用在 SVM 中时，观察上面的 SVM 对偶问题：

$$\max_{\lambda} -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j y_i y_j x_i^T x_j + \sum_{i=1}^N \lambda_i, \text{ s.t. } \lambda_i \geq 0$$

在求解的时候需要求得内积，于是不可分数据在通过特征变换后，需要求得变换后的内积。我们常常很难求得变换函数的内积。于是直接引入内积的变换函数：

$$\forall x, x' \in \mathcal{X}, \exists \phi \in \mathcal{H} : x \rightarrow z \text{ s.t. } k(x, x') = \phi(x)^T \phi(x')$$

称  $k(x, x')$  为一个正定核函数，其中  $\mathcal{H}$  是 Hilbert 空间（完备的线性内积空间），如果去掉内积这个条件我们简单地称为核函数。

$k(x, x') = \exp(-\frac{(x-x')^2}{2\sigma^2})$  是一个核函数。

证明：

$$\begin{aligned} \exp\left(-\frac{(x-x')^2}{2\sigma^2}\right) &= \exp\left(-\frac{x^2}{2\sigma^2}\right) \exp\left(\frac{xx'}{\sigma^2}\right) \exp\left(-\frac{x'^2}{2\sigma^2}\right) \\ &= \exp\left(-\frac{x^2}{2\sigma^2}\right) \sum_{n=0}^{+\infty} \frac{x^n x'^n}{\sigma^{2n} n!} \exp\left(-\frac{x'^2}{2\sigma^2}\right) \\ &= \exp\left(-\frac{x^2}{2\sigma^2}\right) \varphi(x) \varphi(x') \exp\left(-\frac{x'^2}{2\sigma^2}\right) \\ &= \phi(x) \phi(x') \end{aligned}$$

正定核函数有下面的等价定义：

如果核函数满足：

1. 对称性
2. 正定性

那么这个核函数时正定核函数。

证明：

1. 对称性  $\Rightarrow k(x, z) = k(z, x)$ , 显然满足内积的定义
2. 正定性  $\Rightarrow \forall N, x_1, x_2, \dots, x_N \in \mathcal{X}$ , 对应的 Gram Matrix  $K = [k(x_i, x_j)]$  是半正定的。

要证：  $k(x, z) = \phi(x)^T \phi(z) \Rightarrow K$  半正定+对称性。

1.  $\Rightarrow$ : 首先, 对称性是显然的, 对于正定性:

$$K = \begin{pmatrix} k(x_1, x_2) & \cdots & k(x_1, x_N) \\ \vdots & \ddots & \vdots \\ k(x_N, x_1) & \cdots & k(x_N, x_N) \end{pmatrix}$$

任意取  $\alpha \in \mathbb{R}^N$ , 即需要证明  $\alpha^T K \alpha \geq 0$ :

$$\alpha^T K \alpha = \sum_{i,j} \alpha_i \alpha_j K_{ij} = \sum_{i,j} \alpha_i \phi^T(x_i) \phi(x_j) \alpha_j = \sum_i \alpha_i \phi^T(x_i) \sum_j \alpha_j \phi(x_j)$$

这个式子就是内积的形式, Hilbert 空间满足线性性, 于是正定性的证。

2.  $\Leftarrow$ : 对于  $K$  进行分解, 对于对称矩阵  $K = V \Lambda V^T$ , 那么令  $\phi(x_i) = \sqrt{\lambda_i} v_i$ , 其中  $v_i$  是特征向量, 于是就构造了  $k(x, z) = \sqrt{\lambda_i} \sqrt{\lambda_j} v_i^T v_j$

## 小结

分类问题在很长一段时间都依赖 SVM, 对于严格可分的数据集, Hard-margin SVM 选定一个超平面, 保证所有数据到这个超平面的距离最大, 对这个平面施加约束, 固定  $y_i(w^T x_i + b) = 1$ , 得到了一个凸优化问题并且所有的约束条件都是仿射函数, 于是满足 Slater 条件, 将这个问题变换成为对偶的问题, 可以得到等价的解, 并求出约束参数:

$$\max_{\lambda} -\frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \lambda_i \lambda_j y_i y_j x_i^T x_j + \sum_{i=1}^N \lambda_i, \text{ s.t. } \lambda_i \geq 0$$

对需要的超平面参数的求解采用强对偶问题的 KKT 条件进行。

$$\begin{aligned} \frac{\partial L}{\partial w} &= 0, \frac{\partial L}{\partial b} = 0 \\ \lambda_k (1 - y_k (w^T x_k + b)) &= 0 \quad (\text{slackness complementary}) \\ \lambda_i &\geq 0 \\ 1 - y_i (w^T x_i + b) &\leq 0 \end{aligned}$$

解就是:

$$\begin{aligned} \hat{w} &= \sum_{i=1}^N \lambda_i y_i x_i \\ \hat{b} &= y_k - w^T x_k = y_k - \sum_{i=1}^N \lambda_i y_i x_i^T x_k, \exists k, 1 - y_k (w^T x_k + b) = 0 \end{aligned}$$

当允许一点错误的时候，可以在 Hard-margin SVM 中加入错误项。用 Hinge Function 表示错误项的大小，得到：

$$\underset{w,b}{\operatorname{argmin}} \frac{1}{2} w^T w + C \sum_{i=1}^N \xi_i \text{ s.t. } y_i(w^T x_i + b) \geq 1 - \xi_i, \xi_i \geq 0, i = 1, 2, \dots, N$$

对于完全不可分的问题，我们采用特征转换的方式，在 SVM 中，我们引入正定核函数来直接对内积进行变换，只要这个变换满足对称性和正定性，那么就可以用做核函数。

Table 1: Yule-Simpson's Paradox

Population		Survive	Die	Survive Rate
Treatment	20	20	50%	
Control	16	24	40%	
<b>Male</b>				
		Survive	Die	Survive Rate
Treatment	18	12	60%	
Control	7	3	70%	
<b>Female</b>				
		Survive	Die	Survive Rate
Treatment	2	8	20%	
Control	9	21	30%	

## Causal inference and causal discovery

1. 相关性不代表因果性。
2. 相关性是对称的，而因果性是不对称的。

$$p(\mathbf{x}|\mathbf{y}) = \frac{p(\mathbf{xy})}{p(\mathbf{y})} \quad p(\mathbf{y}|\mathbf{x}) = \frac{p(\mathbf{xy})}{p(\mathbf{x})}$$

We say  $\mathbf{x}$  is independent with  $\mathbf{y}$  given  $\mathbf{z}$ , i.e.,  $\mathbf{x} \perp\!\!\!\perp \mathbf{y} | \mathbf{z}$  , if

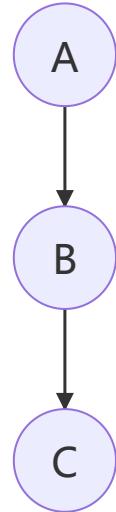
$$p(\mathbf{x}|\mathbf{y}, \mathbf{z}) = p(\mathbf{x}|\mathbf{z})$$

## 有向图-贝叶斯网络

已知联合分布中，各个随机变量之间的依赖关系，那么可以通过拓扑排序（根据依赖关系）可以获得一个有向图。而如果已知一个图，也可以直接得到联合概率分布的因子分解：

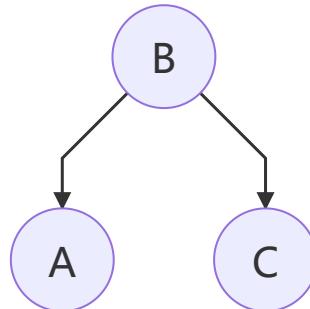
$$p(x_1, x_2, \dots, x_p) = \prod_{i=1}^p p(x_i | x_{\text{parent}(i)})$$

那么实际的图中条件独立性是如何体现的呢？在局部任何三个节点，可以有三种结构：



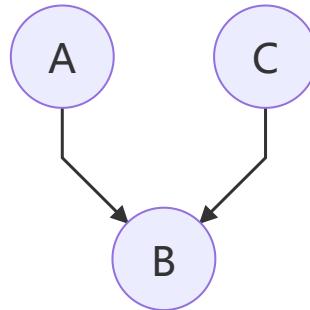
1.

$$\begin{aligned}
 p(A, B, C) &= p(A)p(B|A)p(C|B) = p(A)p(B|A)p(C|B, A) \\
 &\implies p(C|B) = p(C|B, A) \\
 \Leftrightarrow p(C|B)p(A|B) &= p(C|A, B)p(A|B) = p(C, A|B) \\
 &\implies C \perp A|B
 \end{aligned}$$



2.

$$\begin{aligned}
 p(A, B, C) &= p(A|B)p(B)p(C|B) = p(B)p(A|B)p(C|A, B) \\
 &\implies p(C|B) = p(C|B, A) \\
 \Leftrightarrow p(C|B)p(A|B) &= p(C|A, B)p(A|B) = p(C, A|B) \\
 &\implies C \perp A|B
 \end{aligned}$$



3.

$$\begin{aligned}
 p(A, B, C) &= p(A)p(C)p(B|C, A) = p(A)p(C|A)p(B|C, A) \\
 &\implies p(C) = p(C|A) \\
 \Leftrightarrow C &\perp A
 \end{aligned}$$

对这种结构， $A, C$  不与  $B$  条件独立。

从整体的图来看，可以引入 D 划分的概念。对于类似上面图 1 和图 2 的关系，引入集合  $A, B$ ，那么满足  $A \perp\!\!\!\perp B | C$  的  $C$  集合中的点与  $A, B$  中的点的关系都满足图 1, 2，满足图 3 关系的点都不在  $C$  中。D 划分应用在贝叶斯定理中：

$$p(x_i | x_{-i}) = \frac{p(x)}{\int p(x) dx_i} = \frac{\prod_{j=1}^p p(x_j | x_{parents(j)})}{\int \prod_{j=1}^p p(x_j | x_{parents(j)}) dx_i}$$

可以发现，上下部分可以分为两部分，一部分是和  $x_i$  相关的，另一部分是和  $x_i$  无关的，而这个无关的部分可以相互约掉。于是计算只涉及和  $x_i$  相关的部分。

与  $x_i$  相关的部分可以写成：

$$p(x_i | x_{parents(i)}) p(x_{child(i)} | x_i)$$

这些相关的部分又叫做 Markov 链。

## 贝叶斯网络与其结构学习算法