PRML 笔记

Notes on Pattern Recognition and Machine Learning (Bishop)

Version 1.0

Jian Xiao[①]

目录

① iamxiaojian@gmail.com

Checklist

Frequentist-Bayesian对峙构成的主要内容

Frequentist 版本	Bayesian 版本	解模型所用的方法
Linear basis function regression	Bayesian linear basis function regression	前者和后者皆有 closed-form solution
Logistic regression	Bayesian logitstic regression	前者牛顿迭代(IRLS)，后者 Laplace approximation
Neural network (for regression, classification)	Bayesian Neural network (for regression, classification)	前者 gradient decent，后者 Laplace approximation
SVM (for regression, classification)	RVM (for regression, classification)	前者解二次规划，后者迭代、Laplace approximation
Gaussian mixture model	Bayesian Gaussian mixture model	前者用 EM，后者 Variation inferencce
Probabilistic PCA	Bayesian probabilistic PCA	前者 closed-form solution 或 EM，后者 Laplace approximation
Hidden markov model	Bayesian Hidden markov model	前者 EM，后者未详提
Linear dynamic system	Bayesian Linear dynamic system	前者 EM，后者未详提

三种形式的Bayesian

Fully Bayesian：需要 marginalize with respect to hyper-parameters as well as parameters。而这往往是 analytical intractable 的。

对于 curve fitting 的例子($p(\boldsymbol{w}\mid\alpha)=N(\boldsymbol{w}\mid\boldsymbol{0},\alpha^{-1}\boldsymbol{I})$，$p(t\mid\boldsymbol{x},\boldsymbol{w},\beta)=N(t\mid y(\boldsymbol{x},\boldsymbol{w}),\beta^{-1})$) 来说，就是：

$$p(t\mid\boldsymbol{t})=\iiint p(t\mid\boldsymbol{w},\beta)\,p(\boldsymbol{w}\mid\boldsymbol{t},\alpha,\beta)\,p(\alpha,\beta\mid\boldsymbol{t})d\boldsymbol{w}d\alpha d\beta$$

Empirical Bayes/type 2 maximum likelihood/evidence approximation：对 hyper-parameter 采取这样的策略，即先求出使得 marginal likelihood 最大化的参数 α^* 和 β^*，然后使 hyper-parameter 取固定的值 α^* 和 β^*，再对 **w** 进行 marginalize：

$$p(t\mid\boldsymbol{t})\approx p(t\mid\boldsymbol{t},\alpha^*,\beta^*)=\int p(t\mid\boldsymbol{w},\beta^*)\,p(\boldsymbol{w}\mid\boldsymbol{t},\alpha^*,\beta^*)d\boldsymbol{w}$$

MAP(poor man's Bayesian)：不涉及 marginalization，仅仅是一种按后验概率最大化的 point estimate。

PRML 说的 Bayesian 主要还是指 Empirical Bayesian。

Optimization/approximation

Linear/Quadratic/Convex optimization：求线性/二次/凸函数的最值

Lagrange multiplier：带（等式或不等式）约束的最值

Gradient decent：求最值

Newton iteration：解方程

Laplace approximation：近似

Expectation Maximation (EM)：求最值/近似，latent variable model 中几乎无处不在

Variational inference：求泛函最值

Expectation Propagation (EP)：求泛函最值

MCMC/Gibbs sampling：采样

Latent variable model

	Latent variable 离散	Latent variable 连续
Latent variable 独立	GMM	Probabilistic PCA/ICA/Factor Analysis
Latent variable 是 Markov chain	HMM	LDS

Objective function/ Error function/Estimator

Likelihood：MLE 参数估计的 estimator，最常用的目标函数

Marginal likelihood：emprical Bayes/evidence approximation 中用于估计 hyper-parameter

Sum-of-square error：regression 常用

Posterior：MAP 参数估计

Negative log likelihood/cross-entropy：Logistic regression

Exponential error：Adaboost 的目标函数

Hinge error：SVM 的目标函数

Chapter 1 Introduction

1. Bayesian interpretation of probability

与其说是贝叶斯学派对“概率”这个概念的解释，不如说是概率碰巧可以作为量化贝叶斯学派“degree of belief”这个概念的手段。

贝叶斯学派的出发点是“uncertainty”这个概念，对此给予“degree of belief”以表示不确定性。Cox showed that if numerical values are used to represent degrees of belief, then a simple set of axioms encoding common sense properties of such beliefs leads uniquely to a set of rules for manipulating degrees of belief that are equivalent to the sum and product rules of probability.

因此之故，我们才可以 use the machinery of probability theory to describe the uncertainty in model parameters.

2. 对parameter的观点，以及Bayesian对先验、后验概率的解释

对于 Frequentist 来说，model parameter **w** 是一个 fixed 的量，用“estimator”来估计；最常见的 estimator 是 likelihood。

对 Bayesian 来说，**w** 本身是一个不确定量，其不确定性用 prior probability p(**w**)表示。

为了获知 fixed 的 **w**，Frequentist 进行重复多次的试验，获得不同的 data sets D；

对于 Bayesian 而言，there is only a single data set D, namely the one that is actually observed. 在得到一个 observation D 后，贝叶斯学派要调整原来对于 **w** 的 belief（prior probability），用后验概率 P(**w**|D)表示调整后的 belief。调整的方法是贝叶斯定理。

Bayesian 的中心定理是贝叶斯定理，该定理 convert a prior probability into a posterior probability by incorporating the evidence provided by the observed data。其中的条件概率 P(D|**w**)表示的是，how probable the observed data set is for different settings of parameter vector **w**。

$$p(w \mid D) = \frac{p(D \mid w)\,p(w)}{p(D)} = \frac{p(D \mid w)\,p(w)}{\int p(D \mid w)\,p(w)\,dw}$$

其中分母里的 p(D)只是用于归一化的量，使得 p(**w**|D)确实是一个概率。而 p(D)的计算已经给出在上面的分母中。

（理解后验概率：即修正后的先验概率。例如，有 C1,…,Ck 个类别，先验为 P(C1)，…，P(Ck)，这个时候如果给一个未知类别的数据让我们猜它是哪个类别，显然应该猜先验概率最大的那个类别。在观察到数据 ***x*** *后，计算后验概率 P(C1|**x**)，…，P(Ck|**x**)；于是此时的“先验”修正为 P’(C1)=P(C1|**x**)，…，P’(Ck)=P(Ck|**x**)。如果现在再来一个未知类别的数据让我们猜，我们猜的方法仍旧是找先验概率最大的那个类别，只不过此时的先验概率是 P’(C1)，…，P’(Ck)。）*

3. Bayesian和Frequentist的缺点

Bayesian 常受的批评之一：prior distribution is often selected on the basis of mathematical convenience rather than as a reflection of any prior beliefs。 例如常选择 conjugate prior。

Frequentist 方法的缺点：Over-fitting problem can be understood as a general property of maximum likelihood。

4. 应对over-fitting问题

Frequentist 控制 over-fitting 的方法：

1) regularization，即在目标函数中加入一个 penalty term。

L2 regularizer 被称为 ridge regression

L1 regularizer 被称为 **Lasso regression**

加入 penalty 的方法也叫 shrinkage method，因为它可以 reduce the value of the coefficients.

2) cross-validation，即留出一部分数据做 validation

Cross-validation 也是一种进行 model selection 的方法。利用留出来的 validation data，可以选择多个所训练 model 中的最好的一个。

Bayesian 控制 over-fitting 的方法：Prior probability

5. Bayesian方法面临的主要问题：marginalization计算困难

Marginalization lies at the heart of Bayesian methods.

Bayesian methods 的应用长期受制于 marginalization。对于一个 full Bayesian procedure 来说 要 make prediction 或 compare different models，必要的一步是 marginalize (sum or integrate) over the whole of parameter space.

两方面发展起来的方法克服了做 marginalization 的困难：

第一种是 sampling，例如 Markov chain Monte Carlo。Monte Carlo method 的优点是 flexible 而广泛用于各种 model 中；缺点是 computationally intensive，主要用于 small-scale problems.

第二种是 deterministic approximation，例如 variational Bayes 和 expectation propagation，优点是可用于 large-scale applications.

6. Curve fitting为例子演示三种方法

1) **MLE**，直接对 likelihood function 求最大值，得到参数 **w**。该方法属于 point estimation。

2) **MAP** (***poor man's bayes***)，引入 prior probability，对 posterior probability 求最大值，得到 **w**。MAP 此时相当于在 MLE 的目标函数（likelihood function）中加入一个 L2 penalty。该方

法仍属于 point estimation。

3) **fully Bayesian approach**，需要使用 sum rule 和 product rule（因为“degree of belief”的 machinery 和概率相同，因此这两个 rule 对 “degree of belief”成立），而要获得 predictive distribution 又需要 marginalize (sum or integrate) over the whole of parameter space w。

$$p(t \mid \boldsymbol{x}, \boldsymbol{X}, \boldsymbol{t}) = \int p(t \mid \boldsymbol{x}, \boldsymbol{w}) p(\boldsymbol{w} \mid \boldsymbol{X}, \boldsymbol{t}) d\boldsymbol{w}$$

其中，**x** 是待预测的点，**X** 是观察到的数据集，**t** 是数据集中每个数据点相应的 label。

其实是用参数 w 的后验概率为权，对 probability 进行一次加权平均；因此这个过程需要对 w 进行积分，即 marginalization。

7. PRML所需要的三论：Probability theory, decision theory and information theory

8. 两个阶段：inference与decision

整个问题的 solution 分为两阶段：先做 inference，然后做 decision。在 inference stage，要得到联合概率分布或者后验概率分布；在 decision stage，则用 posterior probability to make optimal class assignments。

9. 三种解决decision problem的不同方式

1) discriminant function: map inputs x directly into decisions. 因此 discriminant function 把 inference 和 decision 合作一部解决了。

2) discriminant model: 第一步，解决 inference problem，determining the posterior class probabilities, $P(C_k \mid \boldsymbol{x})$；第二步，解决 decision problem，对于新给定的 **x**，把它分配给某一个 class。

3) generative model：explicitly or implicitly model the distribution of inputs as well as outputs. 第一步，得到 class-conditional density $P(\boldsymbol{x} \mid C_k)$ 或者 joint distribution $P(\boldsymbol{x}, C_k)$；第二步，在前一步的基础上得出 posterior；第三步，decision problem。

10. 评述三种解决决策问题方式的缺点

Generative model 的缺点：如果只是 make classification decision，计算 joint distribution 是 wasteful of computational resources and excessively demanding of data。一般有后验概率就足够了。

Discriminant function 的缺点：该方法不求后验概率，然而 there are powerful reasons for wanting to compute the posterior probabilities：

(1) 当 **loss matrix 可能随时间改变**时（例如 financial application），如果计算出来后验概率，那么解决 decision problem 只需要修改 expected loss 目标函数；没有后验概率的 discriminant

function 只能全部重新学习。

(2) 当**正负样本不平衡**时（例如 X-ray 图诊断癌症，99.9%可能都是无癌症样本；又如垃圾邮件的情况，绝大多数邮件都不是垃圾邮件），为了获得好的分类器，需要人工做出一个 balanced data set 用于训练；训练得到一个后验概率 $P(C_k \mid \boldsymbol{x})$ 后，需要 compensate for the effects of the modification to the training data，即把 $P(C_k \mid \boldsymbol{x})$ 除以 balanced data set 的先验 $P(C_k)$，再乘以真实数据的先验 $P(C_k)$，从而得到了真实数据的后验概率 $P(C_k \mid \boldsymbol{x})$。没有后验概率的 discriminant function 是无法用以上方法应对正负样本不平衡问题的。

(3) Combining models（模型融合）：对于复杂应用，一个问题可能分解成多个子问题，例如疾病诊断，可能除了 X-ray 图片数据 x_I，还有血液检查数据 x_B。与其把这些 heterogeneous information 合在一起作为 input，更有效的方法是 build one system to interpret the X-ray images and a different one to interpret the blood data。即有:

$$P(C_k \mid x_I, x_B) \propto P(x_I, x_B \mid C_k)P(C_k)$$

$$\propto P(x_I \mid C_k)P(x_B \mid C_k)P(C_k) \propto \frac{P(C_k \mid x_I)P(C_k \mid x_B)}{P(C_k)}$$

其中假设了 x_I 和 x_B 关于类别 C_k 条件独立。

11. Criteria for making decisions

1) Minimizing the misclassification rate

2) minimizing the expected loss：两类错误的后果可能是不同的，例如“把癌症诊断为无癌症”的后果比“把无癌症诊断为癌症”的后果更严重，又如“把正常邮件诊断为垃圾邮件”的后果比“把垃圾邮件诊断为正常邮件”的后果更严重；这时候，少犯前一错误比少犯后一错误更有意义。为此需要 loss function 对不同的错误的代价做量化。设集合 A={$a_1, \ldots, a_p$}是所有可能的决策，决策函α：$\chi \to$ A把每个观察数据x ϵ χ映射到一个决策α(x)，那么有

$$\alpha(x) = \arg\min_{a_i \in A} R(a_i \mid x)$$

其中 $R(a_i \mid x)$ 表示的条件风险：

$$R(a_i \mid x) = \sum_{j=1}^{c} \lambda(a_i \mid \omega_j)P(\omega_j \mid x)$$

$\lambda(a_i \mid \omega_j)$ 是对于特定的真实类别是 ω_j 的数据，决策为 a_i 时的 loss 或 risk，即表示 $\lambda(a_i \mid x \in \omega_j)$； $P(\omega_j \mid x)$ 是当观察到 x 后，类别 ω_j 的后验概率。

决策函数α(x)总是把观察数据映射到条件风险最小的决策上。

于是，决策函数α(x)的总的 risk 是：

$$R[\alpha] = \int R(\alpha(x) \mid x) p(x) dx$$

即条件风险 $R(\alpha(x) \mid x)$ 在所有可能观察数据（特征空间）分布下的期望值。

12. 熵

Entropy is a lower bound on the number of bits needed to transmit the state of a random variable.

某随机变量 X，x 为其一个取值，概率为 p(x)，则

1) 该值的编码长度是 $-\log_2 p(x)$

2)该随机变量的熵：$-\int p(x)\log_2 p(x)dx$，为平均编码长度

以上以 2 为低，单位是“比特”；如果以自然对数 e 为地，则是“奈特”。

13. 条件熵，相对熵，互信息

Conditional entropy：设有 joint distribution p(X,Y)，则条件熵 H[Y|X]为一个期望/平均值：

$$H[Y \mid X] = \int H(Y \mid X = x) p(X = x) d$$

$$= -\int\int p(x, y) \log p(y \mid x) dx dy$$

根据上面的定义，可见要定义条件熵 H[Y|X]，先需定义当给定 X=x 时，Y 的熵。即：

$$H(Y \mid X = x) = -\int p(y \mid x) \log p(y \mid x) dy$$

Relative entropy：设有一个未知的分布 p(x)，而 q(x)为我们所获得的一个对 p(x)的近似；按照 q(x) (而非真实分布 p(x))对该随机变量的各个值进行编码，平均编码长度比用真实分布 p(x)进行编码要额外长多少？答案是相对熵(KL 距离)KL(p||q)。即

$$(-\int p(x)\ln q(x)dx) - (-\int p(x)\ln p(x)dx) = -\int p(x)\ln\{\frac{q(x)}{p(x)}\}dx$$

注意第一项，由于按照 q(x)来对随机变量的值编码，因此 q(x)在对数里面；而实际的分布是 p(x)，因此实际的平均长度是按照 p(x)计算的，p(x)在外面。

Mutual information：如果两个随即变量 X,Y 是独立的，那么有 p(x, y)=P(x)P(y)；当二者并不独立时，我们希望可以度量它们离独立还有多远，这个度量就是互信息：

I[x, y] = KL (p(x, y) || p(x) p(y)) = H[x] – H[x|y] = H[y] – H[y|x]

对互信息的另一个解释是：当观察到 x（或 y）后，y（或 x）的 uncertainty 的减少量。

注意到-lnx 是凸函数，利用凸函数的 Jensen’s inequality，易证明 KL 距离非负，且 KL(p(x)|q(x)) = 0 当且仅当 p(x) = q(x)对每个 x 成立（因为-lnx 是严格凸的）。

14. 习题 1.14

任意矩阵可以分解成一个对称矩阵加上一个反对称矩阵。

这个命题说明，考虑二次型的时候，只需要考虑对称的。

15 习题 1.32

一个连续随即向量 x 的熵为 H[x]；x 经过一个可逆线性变换后得到 y=Ax，则 H[y]=H[x]+ln|A|。

补充：

Feature extraction: a special form of dimensionality reduction, the input data be transformed into a reduced representation set of features (features vector)

例如：real-time face detection in a high-resolution video stream, 直接处理的话将是 huge numbers of pixels per second。这时候就要特征提取以 speed up computation。

相关的算法：PCA, kernel PCA, manifold leaning 等

Feature selection: variable selection, feature reduction, attributes selection

相关的算法：Feature selection algorithms typically fall into two categories, feature ranking and subset selection.

Feature ranking ranks the features by a metric and eliminates all features that do not achieve an adequate score.

Subset selection searches the set of possible features for the optimal subset. 这是一个组合优化问题，可以采取一些组合优化策略近似按优化某个metric来选择一个feature subset。

In statistics, the most popular form of feature selection is **stepwise regression**. It is a greedy algorithm that **adds the best feature (or deletes the worst feature) at each round**. The main control issue is deciding when to stop the iteration.

In machine learning, this is typically done by cross-validation.

Unsupervised leaning: the training data consists of a set of input vectors without any corresponding target values.

包括：Clustering, density estimation (determine the distribution of data within the input space), visualization (project the data from a high-dimensional space down to two or three dimensions)

p(D|w)的含义：

对于 Frequentist，w 是一个固定的值，所以 p(D|w)不同于条件概率；

对于 Bayesian，w 是一个随机变量，所以 p(D|w)具有和条件概率一样的意义。

Chapter 2 Probability Distribution

1. 参数方法与非参数方法

Parametric method: assume a specific functional form for the distribution.

Nonparametric method: form of distribution typically depends on the size of the data set. Such models **still contain parameters**, but control the model complexity rather than the form of the distribution.

2. **Conjugate prior**: lead to posterior distribution having the same functional form as the prior

Distribution	**Conjugate Prior**
Bernoulli	Beta distribution
Multinomial	Dirichlet distribution
Gaussian , Given variance, mean unknown	Gaussian distribution
Gaussian, Given mean, variance unknown	Gamma distribution
Gaussian, both mean and variance unknown	Gaussian-Gamma distribution

3. Conjugate prior的意义：方便进行Bayesian inference，甚至是sequential Bayesian inference

sequential Bayesian inference：得到一个 observation 后，可以算出 posterior；由于选取的是共轭先验，所以 posterior 和原来的 prior 形式一样，可以把该 posterior 当作新的 prior，用于下一个 observation，如此迭代下去。对于 stream of data 的情况，这种方式可以实现 real-time learning。

4. Conjugate prior的计算：以multinomial分布为例，计算一个分布的共轭先验分布

D 是一个 N 点的数据集，multinomial 有 K 中结果，那么

$$p(D \mid \mu) = \prod_{k=1}^{K} \mu_k^{m_k}$$

其中 m_k 是在 N 个点中第 k 个结果（状态）出现的次数。

这个分布的 prior 是参数 μ 的分布，由 hype parameter 控制，为了使其为 conjugate prior，假设这个分布：

$$p(\mu \mid \alpha) \propto \prod_{k=1}^{K} \mu_k^{\alpha_k - 1}$$

注意上面还不是等号，因为右端可能不是一个分布。但是假如可以为其添加一个系数（用 α 表达），使得右端积分后为 1，那么它就是一个分布了，事实上：

$$Dir(\mu \mid \alpha) = \frac{\Gamma(\alpha_1 + \ldots + \alpha_K)}{\Gamma(\alpha_1) + \ldots + \Gamma(\alpha_K)} \prod_{k=1}^{K} \mu_k^{\alpha_k - 1}$$

这个分布就是 Dirichlet 分布，它是 Multinomial 的共轭先验分布。

其他分布的共轭先验可类似推导。基本思路是：**先确定参数在 prior 中具有的 functional form，然后归一化求系数。**

5. 多元高斯分布

基础知识：Linear Algebra + Matrix Theory, Multivariate Calculus

准备：

一元高斯函数在 R 上的积分计算；

任意 random vector 的协方差矩阵是非负定的；

任意正定矩阵 A，存在非奇异矩阵 G，使得 A = G'G；

重积分换元定理；

特征函数（probability density function 的一个傅立叶变换）；

线性变换的特征向量和特征值；

待解决的问题：

证明多元高斯的 probability density function 是归一化的；

推导 conditional gauss distribution；（涉及：completing the square，即**配方法**；**待定系数法**；inverse of partitioned matrix 公式）

推导 marginal gauss distribution；

对多元高斯的评述：

(1) 参数太多，计算复杂（协方差矩阵是维度的平方级的参数个数）

(2) 单峰函数，建模能力有限

以上两者形成某种矛盾：一方面是参数多，模型应该是更 flexible 的；而另一方面，它却不能建模 multimodal function。

多元高斯的扩展：支持 multimodal function

(1) introducing discrete latent variables：例如 Gaussian Mixtures model。

(2) introducing continuous latent variables：例如 Linear Dynamic System。

以及还有其他的扩展。

6. Linear Gaussian Model

Given Gaussian distribution p(x) and p(y|x)，而且 p(y|x)的 mean 是 x 的线性函数，covariance 与 x 无关。求：p(y)，p(x|y)。

7. 函数的微分

(1) $f : R \to R$，一般的一元函数求导

(2) $f : R^n \to R$，一般的多元函数求导，求导后为一个 n 维向量，即梯度（gradient），

$(\frac{\partial f(x_1)}{x_1},...,\frac{\partial f(x_n)}{x_n})^T$

(3) $f:R\to R^n$，求导后为一个 n 维向量，即 $(\frac{\partial f_1(x)}{x},...,\frac{\partial f_n(x)}{x})^T$

(4) $f:R^n\to R^n$，求导后为一个矩阵，即雅可比矩阵(Jacobian matrix)，即 $(\frac{\partial f_i}{x_j})_{ij}$

(5) $f:R\to\{A_{m\times n}:R^n\to R^m\}$，即从 scalar 到 matrix 的映射，求导后仍为矩阵，即 $(\frac{\partial A_{ij}}{x})_{ij}$

(6) $f:\{A_{m\times n}:R^n\to R^m\}\to R$，即从 matrix 到 scalar 的映射，求导后仍为矩阵，即 $(\frac{x}{\partial A_{ij}})_{ij}$

根据以上定义，可以推导常见的求导公式，例如 Ax 对 x 求导， x^TAx 对 x 求导等等。

8. The Exponential Family

指数族分布的形式：

$$p(\vec{x}\,|\,\vec{\eta})=h(\vec{x})g(\vec{\eta})\exp\{\vec{\eta}^T\vec{u}(\vec{x})\}$$

其中 $\vec{\eta}$ 是 natural parameter，它跟一个分布通常说的参数可能不同，而是由通常的参数经过变换而来（以符合指数族分别的形式）。

假设用 MLE 方法进行参数估计，对 $g(\vec{\eta})\int h(\vec{x})\exp\{\vec{\eta}^T\vec{u}(\vec{x})\}d\vec{x}=1$ 关于 $\vec{\eta}$ 求导，并令导数为 0 后，得到：

$$-\nabla\ln g(\vec{\eta})=E[\vec{u}(\vec{x})]=\frac{1}{N}\sum_{n=1}^{N}\vec{u}(\vec{x}_n)$$

最右端得到的 vector 的各个分量都是指数族分别的充分统计量。

9. 无信息先验

The prior intended to have as litter influence on the posterior distribution as possible.

当参数的取值有界时，均匀分布是无信息先验；

当参数的取值无界时，均匀分布不能 normalize，是 improper 的。

Translation invariant 和 Scale invariant 的两类分布的无信息先验。

10. Nonparametric methods

问题：给定 D 维空间中观察到的 N 个数据样本，估计密度函数 p(x)（这是一个 unsupervised learning）

方法：在足够小的区域 R 中考虑问题。任取一个点 x，设落入 R 的概率是 P。设观察到

N 个样本，则 R 中落入 K 个点的概率是分别 Bin(K|N,P)。

由于 R 足够小，所以 p(x)在 R 中近似常数，所以：P = p(x) * V，V 是 R 的测度（体积）；

由于 N 足够大，二项分别 Bin(K|N,P)的取值集中在均值 N*P 附近，即：K = N * P。

以上两式联立，可以得到区域 R 上的密度函数近似值：p(x) = K / (N * V)。

Kernel density estimator(即 Parzen window 方法)：固定 V（一个超立方体），在数据集上计算 V 范围内的 K。常采用高斯函数做 smoothing kernel function。

为了方便计算某个区域内的样本点数，可以定义一个 kernel function：

$$k(\vec{u}) = \begin{cases} 1, |u_i| \le 1/2, i = 1, ..., D \\ 0, otherwise \end{cases}$$

$k(\frac{x - x_n}{h}) = 1$ 表示的是：数据点 x_n 落在了以 x 为中心，边长为 h 的 hypercube 中；否则不在该 hypercube 中。那么在 N 个点中，落入该 hypercube 中的点数是：

$$K = \sum_{n=1}^{N} k(\frac{x - x_n}{h})$$

带入 p(x) = K / (N * V)就求出了 hypercube 中的概率密度。

这种 kernel function 将导致 artificial discontinuity，因此需要平滑的函数，通常是 Gaussian：

$$p(x) = \frac{1}{N} \sum_{n=1}^{N} \frac{1}{(2\pi h^2)^{1/2}} \exp\{-\frac{|| x - x_n ||^2}{2h^2}\}$$

缺点讨论：h 的大小难以确定。在 regions of high data density，h 应该小一些，否则可能 lead to over-smoothing and washing out of structure that might otherwise be extracted from the data；相反，数据稀疏的地方，h 太小可能 lead to noisy estimate。所以 h 的取值和 location 有关，而不应该一刀切。

kNN：固定 K，在数据集上计算为了含有 K 个点所需要的 V（一个超球）。注意，kNN 也可以用于 classification，kNN 分类算法是一个最大 posterior 的分类（MAP）。

Chapter 3 Linear Models for Regression

1. 关于curve fitting的几种方法

问题：给定 N 个样本数据点及其对应的函数值，找出该函数

minimize the sum of squares of error：假设函数的形式是 y(x,w)，其中 w 是该函数的待估计参数，x 则是 input variable；该方法认为，该函数应该使得平方误差之和最小化：

$$E(w)=\frac{1}{2}\sum_{n=1}^{N}\{y(x_n,w)-t_n\}^2$$

MLE：假设每个观察到的样本数据点的函数值的 t 是一个以真实的函数 y(x, w)均值的高斯分布产生的。然后求使似然最大的参数值 w，即完成参数估计。

MAP：与 MLE 类似，不过是求解使后验概率最大的参数值 w。

minimize the sum of squares of error with regularizer：往无 regularizer 的 E(w)加入一个 regularizer，例如：

$$E(w)=\frac{1}{2}\sum_{n=1}^{N}\{y(x_n,w)-t_n\}^2+\frac{\lambda}{2}\|w\|^2$$

在以上问题中：

MLE 等价于 minimize the sum of squares of error

MAP 等价于 minimize the sum of squares of error with regularizer (MAP with Gaussian prior)

2. 什么是linear model

函数关于参数，而不是 input variable 是线性，则是线性模型。

例如：linear basis function model

$$y(x,w)=w_0+\sum_{j=1}^{M-1}w_j\phi_j(x)=w^T\phi(x)$$

这里面，basis function 可以任意的选择，而函数 y(x, w)关于 w 始终是线性的。Basis function 的选择有：

Polynomial, Gaussian, Logistic sigmoid function, Fourier basis, wavelets

3. Loss function for regression，Expected Loss最小化，最佳预测即regression function

假设对于 input x，估计函数值是 y(x)，而实际的函数值是 t，这种可能的不准确导致的 loss function 是 L(t, y(x))。那么 Expected loss 是：

$$E[L]=\iint L(t,y(x))\,p(x,t)\,dxdt$$

对于 regression，通常采用的是 squared loss：

$$L(t, y(x)) = \{y(x) - t\}^2$$

把这个 L 带入 E[L]中后，为了最小化 expected loss E[L]，对函数 y(x)求变分，可以得到的使 E[L]最小化的 y(x)是：

$$y(x) = \int tp(t \mid x)d \quad \Leftarrow E_t[t \mid x]$$

也就是 t 关于 x 的 conditional expectation。

对 squared loss 的 L 函数变形：

$$L(t, y(x)) = \{y(x) - t\}^2 = \{y(x) - E[t \mid x] + E[t \mid x] - t\}^2$$

$$= \{y(x) - E[t \mid x]\}^2 + 2\{y(x) - E[t \mid x]\}\{E[t \mid x] - t\} + \{E[t \mid x] - t\}^2$$

把这个式子带入 E[L]公式中，得到：

$$E[L] = \int \{y(x) - E[t \mid x]\}^2 p(x)dx + \iint \{E[t \mid x] - t\}^2 p(x,t)dxdt$$

记 h(x)=E[t|x]，则上式为：

$$E[L] = \int \{y(x) - h(x)\}^2 p(x)dx + \iint \{h(x) - t\}^2 p(x,t)dxdt$$

这里：h(x)是最佳的预测（可以使期望损失 E[L]最小），y(x)是实际的预测，而 t 是实际的函数值。

其中，E[L]中的第二项 $\iint \{h(x) - t\}^2 p(x,t)dxdt$ 是一个与我们自己的预测 y(x)无关的量，arise from the intrinsic noise on the data the represents the minimum achievable value of expected loss。对于做回归的任务，就是找到 y(x)，使其中的第一项尽可能的小。

4. Frequentist的model complexity理论：Bias-Variance trade-off

以 linear basis function model 为例说明。

(1) Frequentist 基于给定的 data set D 对参数 w 进行 point estimate；

(2) 不同的 data set D，估计出来的参数 w 可能不同，因而导致 y(x)不同，故可以把 D 对 y(x)的这种影响记作 y(x; D)——换句话说，每个 D 对应了一个其所训练出来的模型 y(x; D)；

(3) 进行一个 thought experiment：假设有很多不同的 data sets，每个都是从 p(t, x)中采用出来，并且每个 data set 含 N 个样本；

(4) 考虑某个 data set D 所训练的模型 y(x; D)，那么：

$$\{y(x;D) - h(x)\}^2 = \{y(x;D) - E_D[y(x;D)] + E_D[y(x;D)] - h(x)\}^2$$

$$= \{y(x;D) - E_D[y(x;D)]\}^2 + \{E_D[y(x;D)] - h(x)\}^2$$

$$+2\{y(x;D) - E_D[y(x;D)]\}\{E_D[y(x;D)] - h(x)\}$$

现在，可以 take the expectation of this expression **with respect to D**，得到：

$$E_D[\{y(x;D)-h(x)\}^2]=\{E_D[y(x;D)]-h(x)\}^2+E_D[\{y(x;D)-E_D[y(x;D)]\}^2]$$

上面：第一项为 bias^2，表示的是 average prediction over all data sets 与 desired regression function（也就是最佳预测 h(x)）之间的差距，也就是说 **average model 与 best model 的接近程度**；第二项为 variance，表示的是**在单个 data set 上的预测模型 y(x; D)在整体平均值 E[y(x; D)]附近波动的程度（这本身其实就是一个方差），也就是说模型 y(x; D)对数据集 D 的选择的敏感度**。

(5) 分解 expected loss：把 $E_D[\{y(x;D)-h(x)\}^2]$ 替换掉

$$E[L]=\int\{y(x)-h(x)\}^2\,p(x)dx+\iint\{h(x)-t\}^2\,p(x,t)dxdt$$

中的 $\{y(x)-h(x)\}^2$，得到：

Expected loss (E[L]) = (bias)^2 + variance + noise

其中：

$$(bias)^2=\int\{E_D[y(x;D)]-h(x)\}^2\,p(x)dx$$

$$\text{var}\,iance=\int E_D[\{y(x;D)-E_D[y(x;D)]\}^2]p(x)dx$$

$$noise=\iint\{h(x)-t\}^2\,p(x,t)dxdt$$

分析：对于 regression，目标是找到函数 y(x)，使得期望损失 E[L]最小化；而 E[L]本身可以分解开来，由 bias, variance 和 noise 构成。noise 是一个和 y(x)无关的量，也就是说与 regression 做的好不好无关，是一个不可避免的量。而 bias 与 variance 则是一对矛盾量：flexible model 可以具有 low bias（average model 能够很接近 best model），high variance（但是 single model 对 data set D 的选取很敏感），rigid model 则有 high bias，low variance。好的模型应该在 bias 与 variance 之间达到一个折中。

5. 实际的bias-variance计算

仍旧以 curve fitting 为例说明。假设有 L 个 data sets，每个都含有 N 个 data points。模型 $y^{(l)}(x)$ 是有数据集 $D^{(l)}$ 训练得到。那么 average model 可以用平均值来估计：

$$\overline{y}(x)=\frac{1}{L}\sum_{l=1}^{L}y^{(l)}(x)$$

类似得到 bias 和 variance：

$$(bias)^2=\frac{1}{N}\sum_{n=1}^{N}\{\overline{y}(x_n)-h(x_n)\}^2$$

$$\text{var}\,iance = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{L}\sum_{l=1}^{L}\{y^{(l)}(x_n) - \bar{y}(x_n)\}^2$$

以上的提供了一种分析 model 的 bias 和 variance 的方法，就是备好多份数据，各自训练模型，然后按公式计算即可。

6. Bayesian的model complexity理论：Model evidence/marginal likelihood

(1) Bayesian 方法能够避免 over-fitting 的原因是：Marginalizing over the model parameters instead of making point estimates of their values.

(2) 假设有多个 model，$\{M_i : i = 1,...,L\}$；观察到的 data set 是 D。Bayesian 的 model comparison 方法是，比较各个模型的后验概率，即：

$$p(M_i \mid D) \propto p(M_i)p(D \mid M_i)$$

先验概率 $p(M_i)$ allows us to express a preference for different model。可以假设每个模型的先验概率相等，那么剩下要比较的关键是：$p(D \mid M_i)$ ——model evidence 或 marginal likelihood。

(3) Model averaging V.S. model selection

Model averaging：把多个模型，用各自模型的后验概率加权平均，得到 predictive distribution 为 $p(t \mid x, D) = \sum_{i=1}^{L} p(t \mid x, M_i, D)\,p(M_i \mid D)$

Model selection：只选择一个模型，即其中后验概率最大的模型。这是一种 approximation to model averaging。

以上分析可以看出，各个 model 的后验概率是关键，而计算后验概率的关键又是 model evidence。

(4) Model evidence

又叫做 marginal likelihood。之所以这么叫，是因为它的计算/定义：

$$p(D \mid M_i) = \int p(D \mid w, M_i)\,p(w \mid M_i)\,d\ w$$

其中涉及到把 w 进行 marginalize。

从 sampling 的角度看，M_i 相当于 hyper-parameter，而 w 则是 parameter。一个 model 不同于另一个 model，是因为 hyper-parameter。例如：在多项式做 basis function 的 curve fitting 中，多项式的阶 M 就是一个 hyper-parameter，确定一个 M 值（多项式的阶）就是确定了一个 model。在取定一个阶 M 后，参数 w 还有无数种取值可能；把这些取值可能进行 marginalize，得到的就是 model evidence，即由阶为 M 的多项式 model 生成当前所观察到的数据集 D 的概率。

7. Bayesian方法

Fully Bayesian：需要 marginalize with respect to hyper-parameters as well as parameters。而这往往是 analytical intractable 的。

对于 curve fitting 的例子($p(\boldsymbol{w}\mid\alpha)=N(\boldsymbol{w}\mid\boldsymbol{0},\alpha^{-1}\boldsymbol{I})$，$p(t\mid\boldsymbol{x},\boldsymbol{w},\beta)=N(t\mid y(\boldsymbol{x},\boldsymbol{w}),\beta^{-1})$) 来说，就是：

$$p(t\mid\boldsymbol{t})=\iiint p(t\mid\boldsymbol{w},\beta)\,p(\boldsymbol{w}\mid\boldsymbol{t},\alpha,\beta)\,p(\alpha,\beta\mid\boldsymbol{t})d\boldsymbol{w}d\alpha d\beta$$

Empirical Bayes/type 2 maximum likelihood/evidence approximation：对 hyper-parameter 采取这样的策略，即先求出使得 marginal likelihood 最大化的参数 α^* 和 β^*，然后使 hyper-parameter 取固定的值 α^* 和 β^*，再对 **w** 进行 marginalize：

$$p(t\mid\boldsymbol{t})\approx p(t\mid\boldsymbol{t},\alpha^*,\beta^*)=\int p(t\mid\boldsymbol{w},\beta^*)\,p(\boldsymbol{w}\mid\boldsymbol{t},\alpha^*,\beta^*)d\boldsymbol{w}$$

MAP(poor man's Bayesian)：不涉及 marginalization，仅仅是一种按后验概率最大化的 point estimate。

补充：

结论：$p(y\mid x)=\int p(y\mid x,z)\,p(z\mid x)d\ z$

证明：

$$p(y\mid x)=\frac{p(x,y)}{p(x)}=\frac{\int p(x,y,z)d\ z}{p(x)}=\frac{\int p(y\mid x,z)\,p(z\mid x)\,p(x)d\ z}{p(x)}=\int p(y\mid x,z)\,p(z\mid x)d\ z$$

Chapter 4 Linear Models for Classification

1. hyperplane, 超平面

超平面：在一个 D 维 Euclidean space 中的（超）平面是一它的一个 D-1 维流形，而且该空间是一个线性空间。

Linearly separable：分布于 D 维空间中的全部数据点可以用超平面无错地分隔成类

Coding scheme：1-of-K binary coding scheme，即如果有 K 个类，某数据点属于第 i 个类，则表示为一个 K 维向量，该向量除了第 i 个分量是 1，其余都是 0。

Feature vector：对于一个 D 维 input x，使用一个 fixed nonlinear transformation 将其映射成同样是 D 维的向量 $\phi(x)$，即 feature vector。

2. Generalized Linear Model: an **activation function** acting on a linear function of the feature variables.

$$y(x) = f(w^T\phi(x) + w_0)$$

称为 Generalized Linear Model (GLM)，其中 x 为输入数据向量，$\phi(x)$ 为 x 的 feature vector，而 f 是一个函数，称为 activation function，f 的反函数称为 link function。

(1) 当 f 是 nonlinear function 的时候，GLM 是一个 classification model；

(2) 当 f 是 identity function 的时候，GLM 是一个 regression model。

需要注意的是：用于 classification 的 GLM 跟 linear regression model 是不同的，在 regression 里，函数关系是 $y(x) = w^T\phi(x) + w_0$，这个函数是关于参数 w 的线性函数（尽管关于输入 x 可以不是线性的，如 polynomial function basis 等回归）；而**在 GLM 中，由于 activation function 的作用，y 不再是 w 的线性函数**。

用于分类的 GLM 有：

(1) Logistic regression model：就是令 f 是一个 logistic sigmoid 函数；

(2) Probit regression：就是令 activation function f 是一个 probit 函数。

3. 三种分类方法

相应于第 1 章的三种解决 decision problem 的不同方式，对于分类方法也有三种

(1) Discriminant function（判别函数）：直接把输入 x 判归某一个类别；

(2) Generative model（生成模型）：通过计算 joint distribution $p(x, C_k)$ 或 class-conditional distribution $p(x|C_k)$ 而计算 posterior $p(C_k|x)$。

(3) Discriminant model（判别模型）：直接计算 posterior $p(C_k|x)$。**对于 GLM 而言，后**

验概率 $p(C_k \mid x)$ 将表达为 $P(C_k \mid x) = f(w^T \phi(x) + w_0)$，然后利用 training data 直接 infer 出 GLM 中的参数 w；有了这个参数，也就得到了后验概率 $p(C_k \mid x)$，接着就可以进入 decision stage 了。Logistic Regression 和 Probit regression 就是两个这样的分类器。而依 infer 阶段采用的方法不同，可以有 Frequentist 的 Logistic Regression 以及 Bayesian Logistic Regression（对 Probit regression 和其他分类器也一样）。

4. Linear discriminant function（线性判别函数）

Linear discriminant，也就是 decision surface 是 hyperplane 的判别函数，即用超平面作为决策面对输入数据点 x 判定其类别。这属于 non-probabilistic method。

4.1 从 binary classification 到 multiclass classification 的问题

One-versus-the-rest：用 K 个 two-class discriminant 来完成 K 类的分类，每个 discriminant 把一个类的数据与非该类的数据分开。

One-versus-one：用 K(K-1)/2 个 binary discriminant 来完成 K 类的分类。

以上两种方法都存在不足，即有些 decision region 是 ambiguous 的（用一个 region 被分配给了不同的类）。

A single K-class discriminant：由 K 个 linear function 组成的 K-class discriminant，每个 linear function 的形式是 $y_k(x) = w_k{}^T x + w_{k0}$；类别 k 和类别 j 之间的 decision boundary 是 $y_k(x) = y_j(x)$，也就是超平面：$(w_k - w_j)^T x + (w_{k0} - w_{j0}) = 0$；一个数据点 x 被分配给类别 k，如果对于任意 $j \neq k$，都有 $y_k(x) > y_j(x)$。可以证明这种 K-class discriminant 形成的 decision region 是连通凸空间，所以不存在 ambiguous region 问题。

用于学习 Linear discriminant function 的参数的方法有三种：least squares, Fisher's linear discriminant 和 perceptron algorithm。

4.2 **least squares**

如同 regression 中的 sum-of-squares error 方法，有解析解

4.3 **Fisher's linear discriminant**

注意到输入数据点 x 是一个 D 维向量，但是 $y = w^T x$ 确是一个只有一维的 scalar，这个过程可以看作是 D 维的一个向量投影到 1 维空间上。Fisher 判别的思想是，把分类看作选择一个 1 维空间，并把原 D 维数据投影到该空间的过程；选择 1 维空间的准则是 Fisher criterion，包含两方面的要求：一方面要求投影到 1 维空间后，不同类的数据是分开的；而另一方面要

求同一类的数据能够尽可能聚集在一块。Fisher criterion 是这两方面要求的量化。求解 Fisher criterion 最大化后，得到参数 w，即确定了 decision hyperplane 的（法）方向；剩下只需要再在 1 维上确定一个阈值 y0，表明该超平面的位置即可。

4.4 **perceptron algorithm**：就是 Generalized Linear Model $y(x)=f(w^T\phi(x)+w_0)$ 中令 activation function f 为 step function（具体地说，采用的是符号函数 sign(x)），优化的目标函数是 perceptron criterion。

首先修改一下类的标号方式：把第 1 类记 t_n 为+1，第 2 类记 t_n 为-1。对于一个数据 x_n 如果属于第 1 类，则有 $w^T x_n>0$；如果属于第 2 类，则有 $w^T x_n<0$；因此正确分类的数据 x_n 总是有 $w^T x_n t_n>0$。对于每个 misclassified pattern 数据 x_n，$w^T x_n t_n<0$，我们的目标是最小化 $-w^T x_n t_n$。perceptron criterion，就是把所有 misclassified pattern 的这个目标相加，得到 $E_p(w)=\sum_{n\in M}-w^T x_n t_n$，其中 M 是全部 misclassified pattern 的集合。最小化这个目标函数没有 closed 解，可以用 stochastic gradient descent 算法解。

通过最小化 $E_p(w)$，求得参数 w；从而可以利用上面的 Generalized Linear Model（符号函数）来对新的数据 x 做分类了。

缺点：不能对 K>2 类的情况分类；当数据不是线性可分时，迭代求解算法不收敛。

5. Generative model（生成模型）

需要假设 input 的分布，即得到 class-conditional distribution，用贝叶斯定理转化成后验概率后，就是和 Discriminant model 一样进行 make decision 了。

以 2 类分类问题为例说明。

目标：求类别的后验概率 $p(C_k|x)$

过程：

(1) 对后验概率变形：$p(C_1|x)=\dfrac{p(x|C_1)p(C_1)}{p(x|C_1)p(C_1)+p(x|C_2)p(C_2)}$

$$=\frac{1}{1+\exp(-a)}=\sigma(a)$$

其中，$a=\ln\dfrac{p(x|C_1)p(C_1)}{p(x|C_2)p(C_2)}$

(2) 假设 class-conditional distribution 的分布形式：设为 Gaussian，而且进一步假设每个类别的 class-conditional distribution 具有相同的 covariance matrix Σ，不同的仅仅是各自的均值向量 μ_k，即有 $p(x|C_k)=N(\mu_k,\Sigma)$。

(3) 如果已经求得了 class-conditional distribution 和先验概率，那么可以得出：

$$p(C_1 \mid x) = \sigma(\ln \frac{p(x \mid C_1)p(C_1)}{p(x \mid C_2)p(C_2)}) = \sigma(w^T x + w_0)$$

其中：

$$w = \Sigma^{-1}(\mu_1 - \mu_2)$$

$$w_0 = -\frac{1}{2}\mu_1{}^T\Sigma^{-1}\mu_1 + \frac{1}{2}\mu_2{}^T\Sigma^{-1}\mu_2 + \ln \frac{p(C_1)}{p(C_2)}$$

这里面，需要进行参数估计的是每个类别的均值向量 μ_k，共同的 covariance Σ，以及 2 个类别的先验概率。

注意：正是由于假定了各个类别同 covariance，所以才使得 class-conditional distribution 带入 $\ln \frac{p(x \mid C_1)p(C_1)}{p(x \mid C_2)p(C_2)}$ 后可以使得 x 的二次型消除，只剩下一个关于 x 的线性函数 $w^T x + w_0$。

(4) MLE 参数估计

假设：$p(C_1) = \pi$，则 $p(C_2) = 1 - \pi$；训练数据集是 $\{x_n, t_n\}$，若 $x_n \in C_1$，则 $t_n = 1$，否则 $t_n = 0$。

于是有：

$$p(x_n, C_1) = \pi N(x_n \mid \mu_1, \Sigma)$$

$$p(x_n, C_2) = (1-\pi) N(x_n \mid \mu_2, \Sigma)$$

从而得到似然函数：

$$p(t \mid \pi, \mu_1, \mu_2, \Sigma) = \prod_{n=1}^{N} [\pi N(x_n \mid \mu_1, \Sigma)]^{t_n} [(1-\pi) N(x_n \mid \mu_2, \Sigma)]^{1-t_n}$$

通过对各个参数求导就得出了 MLE 的参数估计值。这些值带入(3)中的后验概率，就完成了求解。

步骤小结：假定 class-conditional distribution 的分布形式

→MLE 估计该分布中的参数（从而得到了 class-conditional distribution）

→计算每个类别的后验概率。

在上面的例子中，得到的后验概率刚好是一个 GLM 模型（Logistic）。

6. Discriminant model（判别模型）

优点：参数少，是维数 D 的线性函数；相比之下 generative model 的参数是 D 的平方级。

6.1 **logistic function 和 Softmax function**

Logistic： $\sigma(a)=\frac{1}{1+\exp(-a)}$，是平滑化的 step function，无穷可微（光滑函数）；

Softmax：$s(a_k;a_1,...,a_n)=\frac{\exp(a_k)}{\sum_j \exp(a_j)}$，是平滑化的 max function，无穷可微（光滑函数）。

Softmax 函数是 logistic 函数的扩展。

6.2 **Logistic regression**

Generalized Linear Model 的一种，处理的是：posterior can be achieved by a Softmax transformation of linear function of the feature variables，数学地表示出来就是：

$$p(C_k\mid\phi)=s(a_k;a_1,...,a_n)=\frac{\exp(a_k)}{\sum_j \exp(a_j)}\text{，其中 } a_k=w_k{}^T\phi$$

当只有 2 个类时，上面的公式退化成 logistic function。

以 2 个类的情况为例，现在的目标就变成为，给定 training data 集合$\{\phi_n,t_n\}$，其中ϕ_n是一个 D 维的 feature vector，而$t_n\in\{0,1\}$是类属标记；此时的似然函数是：

$$p(t\mid w)=\prod_{n=1}^{N} y_n{}^{t_n}\{1-y_n\}^{1-t_n}$$

其中 $y_n=p(C_1\mid\phi_n)=\sigma(a_n)$， $a_n=w^T\phi_n$。

求出最小化 negative logarithm of the likelihood（即 cross-entropy error function）：

$$E(w)=-\ln p(t\mid w)=-\sum_{n=1}^{N}\{t_n\ln y_n+(1-t_n)\ln(1-y_n)\}$$

的 w，从而完成对后验概率 $p(C_k\mid\phi)$ 的 inference，剩下的就是依此 make decision 了。

Multiclass 的情况可做相应推导。

另外，如果 activation function 取为 probit function，那么就得到了 probit regression。probit function 是标准高斯分布的 CDF(Cumulative Distribution Function)，与形状 logistic 函数相似。

6.3 **iterative reweighted least squares (IRLS)**

最小化 E(w)应求出其梯度，并令梯度为 0，$\nabla E(w)=\sum_{n=1}^{N}(y_n-t_n)\phi_n=0$。

注意到方程$\nabla E(w)=0$不存在 closed-form solution，因为这个方程含有多个非线性的 logistic function（其中的y_n）的求和。

由于 E(w)是一个凹函数（concave），所以存在唯一的最小值，这个最小值点就是 E(w)的驻点，即满足$\nabla E(w)=0$的 w。

IRLS 其实就是牛顿迭代法，用于解如下方程：

$$\nabla E(w) = 0$$

因此会涉及到 Hessian 矩阵，求解该方程的迭代公式为：

$$w^{(new)} = w^{(old)} - H^{-1}\nabla E(w)$$

7. Laplace approximation

即找一个高斯分布 q(z)，近似一个复杂的分布率 $p(z) = \frac{1}{Z} f(z)$，其中 Z 是一个对 f(z)的归一化因子。

找 q(z)的方法是：首先找到 f(z)的一个驻点 x_0，然后在这点处将 ln[f(z)]泰勒展开：

$$\ln f(z) \simeq \ln f(z_0) - \frac{1}{2} A(z - z_0)^2$$

其中，$A = -\frac{d^2}{dz^2} \ln f(z)|_{z=z_0}$，因此有 f(z)的近似值：

$$f(z) \simeq f(z_0) \exp\{-\frac{A}{2}(z - z_0)^2\}$$

指数部分是 z 的平方函数，所以可以用高斯分布近似，得到：

$$q(z) \simeq (\frac{A}{2\pi})^{1/2} \exp\{-\frac{A}{2}(z - z_0)^2\}$$

这就是用来近似原复杂分布 p(z)的高斯分布。

对于高维情况，用相应的高维泰勒展开和高维高斯分布即可。

8. Bayesian Logistic Regression

以 2 类的情况考虑。对于一个新的 feature vector ϕ，现在要计算它的 predictive distribution，也就是：$p(C_1 | \phi, t) = \int p(C_1 | \phi, w) p(w | t) dw = \int \sigma(w^T x) p(w | t) dw$，涉及到 Bayesian 方法的典型问题，即 marginalize over parameter space；以及在 logistic 变换下变得很复杂的后验概率 p(w|t)。

现在首先要为复杂的 p(w|t)找一个 Gaussian approximation q(w)。按照 Laplace approximation 的方法，得先找到 p(w|t)的：stationary point，以及 lnp(w|t)的 Hessian matrix。

如果假设了先验概率 $p(w) = N(w | m_0, S_0)$，那么对最大化 lnp(w|t)可以求得参数 m_{MAP}，两次求导得到 Hessian matrix S_N，从而近似的高斯分布是 $q(w) = N(w | m_{MAP}, S_N)$。

用 q(w)替换 p(w|t)，完成剩下的 marginalization 的工作，这里就是计算 logistic 函数与 Gaussian 分布的卷积。计算出最终的 predictive distribution。

补充：

关于likelihood function

Wikipedia 上的定义是：The *likelihood* of **a set of parameter values** given some observed outcomes is equal to the ***probability* of those observed outcomes** given those parameter values.

这里面模糊的地方是，什么是一个“observed outcome”。如果对什么是“observed outcome”进行不同的定义，得到的 likelihood 也是不一样的。例如在 PRML 上，出现过这两个例子。

例子之一：*用 generative model 进行 classification 中，假设：* $p(C_1)=\pi$ *，则* $p(C_2)=1-\pi$ *；训练数据集是* $\{x_n,t_n\}$ *，若* $x_n \in C_1$ *，则* $t_n=1$ *，否则* $t_n=0$ 。

于是有：

$$p(x_n,C_1)=\pi N(x_n \mid \mu_1,\Sigma)$$

$$p(x_n,C_2)=(1-\pi)N(x_n \mid \mu_2,\Sigma)$$

从而得到似然函数：

$$p(t \mid \pi,\mu_1,\mu_2,\Sigma)=\prod_{n=1}^{N}[\pi N(x_n \mid \mu_1,\Sigma)]^{t_n}[(1-\pi)N(x_n \mid \mu_2,\Sigma)]^{1-t_n}$$

注意，这里面计算 likelihood 时，使用的是 joint distribution $p(x_n,C_1)$ *或* $p(x_n,C_2)$；也就是说，所谓一个 observed outcome，指的是 $\{x_n,t_n\}$，即“出现 input x_n，并且 x_n 属于类别 t_n”这一事件。因此，这时候要对 input variable 进行建模，考虑出现该 input 的概率。

例子之二：*在 logistic regression 中，给定 training data 集合* $\{\phi_n,t_n\}$ *，其中* ϕ_n *是一个 D 维的 feature vector，而* $t_n \in \{0,1\}$ *是类属标记；此时的似然函数是：*

$$p(t \mid w)=\prod_{n=1}^{N} y_n^{t_n}\{1-y_n\}^{1-t_n}$$

其中 $y_n=p(C_1 \mid \phi_n)=\sigma(a_n)$ *，* $a_n=w^T\phi_n$ 。

注意，这里计算 likelihood 时，使用的是 posterior distribution $y_n=p(C_1 \mid \phi_n)$；也就是说，所谓一个 observed outcome，指的是给定一个 input x_n 之后，发生“ x_n 属于类别 t_n ”这一事件。因此这是 p(x_n 属于类别 $t_n \mid x_n$)，这个概率的确等于后验概率 $y_n=p(C_1 \mid \phi_n)$ 。

Chapter 5 Neural Networks

1. Generalized Linear Model: an activation function acting on a linear function of the feature variables

$$y(x) = f(w^T \phi(x) + w_0)$$

其中 x 为输入数据向量，$\phi(x)$ 为 x 的 feature vector，而 f 是一个函数，称为 activation function，f 的反函数称为 link function。

(1) 当 f 是 nonlinear function 的时候，GLM 是一个 classification model，此时 y(x)被解释为一个 posterior probability（因此 Generalized Linear Model 属于 Discriminative model）。常见的 f 采用 logistic 函数，得到 logistic regression。采用 cross-entropy error function 作为优化目标。

(2) 当 f 是 identity function 的时候，GLM 是一个 regression model，此时 y(x)是原本的函数数值，没有像上面的 probability interpretation。采用 sum-of-squares error function 作为优化目标。

2. Neural Network的定义

就是在上述 Generalized Linear Model 的基础上在加入一个新的 nonlinear function，即：

$$y_k(\mathrm{x}, \mathrm{w}) = f(\sum_{j=0}^{M} w_{kj}^{(2)} h(\sum_{i=0}^{D} w_{ji}^{(1)} x_i))$$

此处 $y_k(\mathrm{x}, \mathrm{w})$ 为神经网络的第 k 个 output（Classification 中的第 k 个类别，Regression 中输出值的第 k 个维度）；M 为 hidden layer 的 unit 个数；D 为输入数据点 x 的维度；h 为 nonlinear function，f 为 linear 或 nonlinear function（视问题为 regression 或 classification 而定）。这个式子已经把 bias 包含进去。

通常 h 取 logistic 函数，那么可见神经网络其实是 logistic regression 的推广而已。但这一推广意义重大，理论上它可以以任意精度一致逼近任意连续函数；而且，这个三层的神经网络能力与更多层的神经网络相同。

3. Neural Network的优化目标函数

对于 regression，采用 sum-of-squares error function，即：

$$E(\mathrm{w}) = \frac{1}{2} \sum_{n=1}^{N} \| \mathrm{y}(\mathrm{x}_n, \mathrm{w}) - \mathrm{t}_n \|^2$$

此处 $\mathrm{y}(\mathrm{x}_n, \mathrm{w})$ 是一个 K 维的 output vector；x_n 是一个 D 维的 input vector；t_n 是一个 target vector（维度与 output vector 相同为 K）；w 是含全部各个层的 weight 的 vector。

对于 classification，采用 negative logarithm of likelihood function，即：

$$E(\mathrm{w}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_{nk} \ln[\mathrm{y_k}(\mathrm{x}_n, \mathrm{w})] \quad (= -\sum_{n=1}^{N} \ln \prod_{k=1}^{K} [\mathrm{y_k}(\mathrm{x}_n, \mathrm{w})]^{t_{nk}} \ )$$

其中，输入数据 x_n 的类属输出值 t_n 采取 1-of-K 的 coding scheme（如果一个 input 属于第 k 类，则除 $t_{nk} = 1$ 外其他的分量都是 0）；第 k 个输出的解释是 $y_k(x_n, w) = p(t_k = 1 | x_n)$。因此上述的 E(w)的确是 likelihood function。

以上两种目标函数（回归的和分类的）

(1) 都可以按逐个 input vector 来写成：

$$E(\mathrm{w}) = \sum_{n=1}^{N} E_n(\mathrm{w})$$

(2) 优化的方向都是最小化（sum-of-squares error function 显然应该最小化，而 likelihood function 则加了负号所以变成为最小化）。

4. 目标函数的优化方法

也就是给定 observation data，求出使得 E(w)最小的参数 w。采用的方法是 Gradient decent，具体分为两种：

(1) Off-line gradient 或 batch gradient。这个类型的算法仍旧有许多，如 steepest decent，conjugate gradients 和 quasi-Newton methods。其中最简单的 steepest decent 方法是：

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E(w^{(\tau)})$$

此处 $\eta > 0$ 是 learning rate，$\nabla E(w^{(\tau)})$ 是梯度，即 batch error function gradient，考虑了全部 observation data。

(2) On-line gradient decent 或 sequential gradient decent 或 stochastic gradient decent。迭代公式是：

$$w^{(\tau+1)} = w^{(\tau)} - \eta \nabla E_n(w^{(\tau)})$$

这种方法中，observation data 可以是逐个考虑，也可以 random selection with replacement。

On-line 方法的优点是可以较好应对 data redundancy，也更有可能 escaping from local minima。

5. 计算梯度 ∇E_n：Error Back-propagation

上述 Gradient decent 方法在迭代过程中的关键在于计算梯度。

令 a_j 表示（hidden layer 的）节点 j 的输入，它来自于前一层节点的输出的加权和，即

$$a_j = \sum_i w_{ji} z_i$$

其中 z_i 表示（input layer 的）节点 i 的输出。于是节点 j 的输出是：

$$z_j = h(a_j)$$

记 $\delta_j = \frac{\partial E_n}{\partial a_j}$，该值表示的是对神经元的输入的敏感度，那么有：

$$\frac{\partial E_n}{\partial w_{ji}} = \frac{\partial E_n}{\partial a_j}\frac{\partial a_j}{\partial w_{ji}} = \delta_j z_i \quad \text{『为了计算} \frac{\partial E_n}{\partial w_{ji}}\text{，线需计算} \delta_j \text{』}$$

设节点 k 是 output layer 的一个 unit，那么有：

$$\delta_j = \frac{\partial E_n}{\partial a_j} = \sum_k \frac{\partial E_n}{\partial a_k}\frac{\partial a_k}{\partial a_j} = \sum_k \delta_k \frac{\partial a_k}{\partial a_j}$$

而 $a_j = \sum_i w_{ji} z_i = \sum_i w_{ji} h(a_i)$ 因此：

$$\frac{\partial a_k}{\partial a_j} = \frac{\partial(\sum_i w_{ki} h(a_i))}{\partial a_j} = w_{kj} h'(a_j)$$

从而得到：

$$\delta_j = \sum_k \delta_k w_{kj} h'(a_j) = h'(a_j)\sum_k \delta_k w_{kj} \quad \text{『为了计算} \delta_j \text{，先需计算} \delta_k \text{』}$$

对于（output layer 的）节点 k，容易计算得出 δ_k。然后由 δ_k 根据上面的式子得出 δ_j。那么全部的 $\frac{\partial E_n}{\partial w_{ji}}$ 便可计算。

对于 output layer 与 hidden layer 之间的权重，有 $\frac{\partial E_n}{\partial w_{kj}} = \frac{\partial E_n}{\partial a_k}\frac{\partial a_k}{\partial w_{kj}} = \delta_k z_j$

由上，从而得出梯度 ∇E_n。

Error Back-propagation 是指先计算（output layer 的）节点 k 的 δ_k，然后是（hidden layer 的）节点 j 的 δ_j；如果有更多层，这个过程会继续下去。如果总共有 W 个参数，该算法的时间复杂度是 O(W)。对于完全连接的三层神经网络，这些参数可以表示为两个矩阵以及两个 bias，从而时间复杂度是 O(N*M + M*K)。

小结 BP 的过程：(1) 在当前的 w 值下，计算每个节点的输入和输出值（上面的 a 与 z 值），这其实就是神经网络的 evaluation（forward propagation）；(2) 根据 E(w)，计算 δ_k 的值；(3) 反

向传播，计算 δ_j 的值；(4) 最后是得出结果，也就是 E(w)的梯度。

小结：训练神经网络的其实就是迭代地寻找到使得 E(w)最小化的参数 w。

为此采用的是 Gradient decent 的方法，该方法有一个核心就是计算 E(w)的梯度。

而计算梯度的方法是 BP。

所以训练 w 的流程就是：

(1)初始化一个参数 w；

(2)按当前的 w 值，用 BP 计算 E(w)的梯度；

(3)按 Gradient decent 公式更新，得到一个新的参数 w；

(4)反复(2)和(3)，直到 w 收敛。

6. Neural Network的evaluation：forward propagation

Evaluation 指的是：给定已知参数 w 的神经网络和一个 D 维 input vector x，计算相应的 output vector。

这个比较容易，方法称为 forward propagation，具体而言，就是用上面曾用到的公式：

$$a_j = \sum_i w_{ji} z_i$$

$$z_j = h(a_j)$$

逐层进行计算即可。时间复杂度也是 O(W)。

7. Optimization theory and method

可以看出，machine learning 的组合拳是：**modeling + optimization**。先建立一个（通常是 statistical 的）模型，然后针对一个目标函数优化，解出模型的参数。

随便举例一些 optimization techniques：

Linear and Quadratic Programming, Convex Optimization；

Combinatorial Optimization；

Probabilistic Optimization: Genetic Algorithm, Simulated Annealing, Particle Swarm Optimization, Ant Colony Optimization;

Calculus of variation;

Numerical Optimization: Gradient decent, Conjugate gradient, Newton method

8. error propagation方法用于计算Hessian矩阵 $\nabla\nabla E_n$ 和Jacobian矩阵J

Jacobian 矩阵 J 定义：

$$J_{ki} = \frac{\partial y_k}{\partial x_i}$$

也就是 output vector 对 input vector 的导数，属于 $f: R^n \to R^n$ 的（一阶）微分问题。

Hessian 矩阵则属于 $f: R^n \to R$ 的（二阶）微分问题。

类似于求梯度的方法可以推导出求上面两种矩阵的公式，也是 error propagation 式的从 output layer 往后递推。

另外一个精彩的例子是计算 Hessian 矩阵与一个向量的乘积的巧妙算法，即计算 $v^T H = v^T \nabla(\nabla E(w))$。这个计算可以在 O(W)时间内完成。

9. Regularization of neural network

对于 Frequentist method 来说，有两种方法：

(1) regularizer：跟 regression 中一样，往目标优化函数中加入一个 w 的 penalty 函数。通常是 quadratic。但这种 regularizer 不满足神经网络的 linear transformation invariance 要求。

(2) early stopping：即用一个 validation set，来确定恰当的迭代停止时间。

10. 对neural network的invariance要求

包括 linear transformation invariance，translation invariance 和 scale invariance。

linear transformation invariance：如果在 error function 上加入 regularizer $w^T w$，将导致 inconsistency。比如：给 input data 一个线性变换 $x_i \to \tilde{x}_i = a x_i + b$，那么我们只要给网络的权重一个相应的变换，$w_{ji} \to \tilde{w}_{ji} = \frac{1}{a} w_{ji}$，$w_{j0} \to \tilde{w}_{j0} = w_{j0} - \frac{b}{a}\sum_i w_{ji}$，易见变换前的网络对变换前数据的作用，等价于变换后的网络对变换后的数据的作用；如果用两份数据分别训练两个神经网络，一份是原来的数据，一份经过上述线性变换的，那么后者训练得到的网络权重 w 应该等于前者得到的权重按如上权重变换的结果。然而，如果用 regularizer $w^T w$，这种 consistency 关系就会被破坏掉，因为这会使得两个理应是 equivalent 的模型变得有优有劣。一个能够满足这种要求的 regularizer 是 $\frac{\lambda_1}{2}\sum_{w\in W_1} w^2 + \frac{\lambda_2}{2}\sum_{w\in W_2} w^2$。

Translation invariance 和 scale invariance：这种要求来自实际的需求，例如图片的识别处理，一个图片不管它的位置（translation）和大小(scale)如何，它都还应该是它自己，neural network 不应该将经过这样变换的同一个图片判断为两个不同的图片。有 4 个方面的思路来处理这个

问题：

(1) data replication，形成 augmented data。也就是说通过把一个数据经过变换后成为多个数据（它们当然被 label 成一个数据），让算法去 learn 其中的 invariance。这种方法导致计算的开销太大。（modify data）

(2) regularization，penalize changes in the model when the input is transformed。代表方法：tangent propagation。（modify error function）

(3) 通过一个 pre-processing，提取出那些 transformation-invariant 的 feature，再用这些 feature 去训练模型。（extraction of transformation-invariant features）

(4) 构造出具有 invariance 性质的神经网络，从而 build the invariance properties into the structure of a neural network。代表方法：convolutional neural network。 （the model with transformation-invariant structure）

11. Bayesian Neural Networks

考虑只有一个 output 的神经网络。

首先对 network 的权重 w 引入先验概率：

$$p(w|\alpha) = N(w|0, \alpha^{-1}I)$$

对于一个给定的 x，conditional distribution $p(t|x,w,\beta) = N(t|y(x,w),\beta^{-1})$，其中 $y(x,w)$ 是神经网络的输出。假设有 N 个 observation $\{x_n\}$，相应的 target value 是 $D=\{t_n\}$。那么，后验概率是：

$$p(w|D,\alpha,\beta) \propto p(w|\alpha)p(D|w,\beta) = p(w|\alpha)\prod_{n=1}^{N} N(t_n|y(x_n,w),\beta^{-1})$$

那么，对于一个新的 x，predictive distribution 是：

$$p(t|x,D) = \int p(t|x,w)p(w|D,\alpha,\beta)dw$$

现在剩余的问题是：

(1) 后验概率很难算，要用 Gaussian 来近似，近似方法见 Laplace approximation；假设 $q(w|D)$ 是其近似分布，则 predictive distribution 变为 $p(t|x,D) = \int p(t|x,w)q(w|D)dw$，由于 p(t|x,w)含有 network function y(x, w)，所以该积分仍旧是 analytically intractable 的。

(2) 对 network function y(x, w)用一阶 Taylor 展开来近似，这样可以使得 $p(t|x,w,\beta)$ 成为一个 linear-Gaussian model，这样 p(t|x, D)就可以积分出来了。

(3) 最后剩下的问题是 hyper-parameter α,β。在 predictive distribution 中并没有对它们 marginalize，表明这里进行的是一个 empirical Bayesian，需要对 marginal likelihood 对

hyper-parameter 进行 point estimate。

以上完成了 Bayesian Neural Network 进行 regression 的整个过程。

<u>补充</u>：感想

Machine learning = Modeling + optimization/approximation。

当说到机器学习的时候，大多数人想的是“算法”一词。这可能并不准确。ML 里大量是在讲怎么 modeling。比如 SVM，很多人可能会自动在其后面加上算法一词（“SVM 算法”），可是实际上它仅仅是一个 modeling；后边怎么解模型还得看采取什么优化算法呢。

ML 的另一半是 optimization 或 approximation。对于 Frequentist 而言，解模型就是参数估计（point estimate），这时候会采取一个 estimator 来估计，比如似然函数，然后选择使得 estimator 最大化/最小化的参数作为模型的参数——这便是 optimization 的过程，涉及到许多最优化算法。对于 Bayesian 而言，参数不做 point estimate，而要 marginalize；这是一个困难的工作，marginalize 往往没有 analytical solution，所以只能 approximation，主要可以分为 analytical approximation（Taylor expansion, Laplace approximation, Variational Bayes）和 sampling approximation。

Chapter 6 Kernel methods

1. 两种model类型

在训练完成后，training data 有可能舍弃也有可能保留用于做 prediction。

第一种类型是把 training data 仅仅用于参数估计，估计出来后训练数据即可舍弃。例如：linear basis function model，generalized linear model，neural network 等。

第二种类型是在做 predication 时仍旧需要用到 training data。这又可细分为两种。一种是全部 training data 需要保存，例如：kNN，Gaussian process 等；另一种只需要保存一部分 training data，例如：SVM（只需要保存 Support Vector）。

2. Kernel的定义和构造

假设ϕ是一个 non-linear feature space mapping，将 input x 映射到特征空间中。那么 kernel function 就是：$k(x,x')=\phi(x)^T\phi(x')$。

两种典型 kernel：stationary kernel 和 homogeneous kernel。前者的性质是$k(x,x')=k(x-x')$，具有平移不变性；后者的性质是$k(x,x')=k(\|x-x'\|)$，仅依赖于参数之间的距离（radial basis kernel）。

Kernel 函数的构造有两条途径。第一种，按照 kernel 的定义，即先确定出一个特征映射，然后求得 kernel。第二种，直接确定一个函数 k，使其存在一个特征映射，满足$k(x,x')=\phi(x)^T\phi(x')$。例如定义$k(x,z)=(x^Tz)^2$，假设 x 和 z 是 2 维变量，那么可以验证 k 是一个 kernel 函数，因为有：$k(x,z)=(x_1^2,\sqrt{2}x_1x_2,x_2^2)(z_1^2,\sqrt{2}z_1z_2,z_2^2)^T$，形式符合 kernel 的定义。

为了更方便的利用第二种方法构造 kernel，需要研究按前面方式定义的 kernel 函数，具有怎样的性质。例如，可以证明两个 kernel 函数相加、相乘后都还是 kernel；一个 kernel 乘上一个正数后也还是 kernel；一个 kernel 的指数函数也还是 kernel。基于这些性质，可以通过简单的 kernel 构造出复杂的 kernel。例如 Gaussian kernel 可用上面的性质证明符合 kernel 的定义。

Kernel 不仅可以定义在 vector of real numbers（欧几里德空间）上，还可以定义在其他结构上。例如：给定集合 A，考虑其幂集 P(A)构成的空间。这并不是一个 vector space，但可以在其上定义 kernel。设$A_1,A_2\in P(A)$，则$k(A_1,A_2)=2^{|A_1\cap A_2|}$。事实上，空间 P(A)经由$\phi$映射后的空间是一个 inner product vector space。

3. 基于generative model构造kernel

Generative model V.S. discriminant model：generative model can deal naturally with missing data，并且在 HMM 模型中，可以 handle sequences of varying length；但是 discriminative model generally gives better performance on discriminative tasks.

融合 generative model 与 discriminant model 的 方法是：先用 generative model 定义出一个 kernel 函数，然后将 kernel 使用于 discriminant model。

利用 generative model，有两种方式方法定义 kernel：

第一种，简单定义 $k(x,x')=p(x)p(x')$。这显然是一个 kernel（p(x)对 input 进行了 modeling，因而是 generative model），其中概率 p(x)>=0，p(x)相当于是一个从 D 维空间（x 的维度）到 1 维空间的特征映射；因此 k 相当于是 1 维空间里的内积。根据“kernel 相加是 kernel”和“kernel 乘上正数是 kernel”，可以得到更复杂的函数形式：$k(x,x')=\sum_i p(x|i)p(x'|i)p(i)$，其中的变量 i 相当于是一个 latent variable。

第二种，构建 Fish kernel。考虑一个 parametric generative model，$p(x|\theta)$，其中 θ 是一个 parameter vector。首先定义 Fisher score：$g(\theta,x)=\nabla_\theta \ln p(x|\theta)$。这是一个 scalar 函数对一个 vector 求导，所以得到的是一个 vector（梯度）。

由此得到 Fisher Kernel：$k(x,x')=g(\theta,x)^T F^{-1} g(\theta,x')$。其中 F 是 Fisher information matrix，公式是：$F=E_x[g(\theta,x)g(\theta,x)^T]$。注意方括号中是 vector outer product，得到的是一个矩阵；该值是在概率分布 $p(x|\theta)$ 关于 x 的期望。Fisher 信息量在实际中可以用 sample average 来近似。

事实上，按此定义，**Fisher score 是对数概率密度函数关于参数的梯度，Fisher information matrix 是 Fisher score（一个 random vector）的 covariance matrix；而 Fisher kernel 则是两个 Fisher score 之间的 Mahalanobis 距离**。如果直接用 Euclidean distance，也得到的是 kernel：$k(x,x')=g(\theta,x)^T g(\theta,x')$。

Fisher kernel 在 document retrieval 中有应用。Kernel 用于 measure the similarity of two input vectors。

4. Gaussian process

定义：Gaussian process is defined as a probability distribution over functions y(x)，such that the set of values of y(x) evaluated at an arbitrary set of points $x_1,...,x_N$ jointly have a Gaussian distribution。

也就是说，任意的 $x_1,...,x_N$，得到一组随机变量 $y(x_1),...,y(x_N)$，这组随机变量是一个 Gaussian distribution。在大多数应用中，我们并没有随机变量 y(x)的 mean value 的 prior，所以可以假设其为 0。因此 Gaussian process 将由 $(y(x_1),...,y(x_N))$ 的 covariance 完全确定。事实上可以再简化一下只考虑两个随机变量，因为如果知道任意两个 x_n, x_m 对应的随机变量 $(y(x_n), y(x_m))$ 的 covariance，就可以知道任意 $(y(x_1),...,y(x_N))$ 的 covariance。我们用 kernel 函数来确定 $(y(x_n), y(x_m))$ 的 covariance：

$$E[(y(x_n)y(x_m)] = k(x_n, x_m)$$

5. Gaussian process for regression

考虑到每个 observation 都还有一定的 noise，也就是对于随机变量 $y(x_n)$，真正观察到的值 t_n 是分布： $p(t_n \mid y_n) = N(t_n \mid y_n, \beta^{-1})$。假设当前有 N 个 observations，则相应的分布是多维的 Gaussian：

$$p(\mathbf{t} \mid \mathbf{y}) = N(\mathbf{t} \mid \mathbf{y}, \beta^{-1}\mathbf{I}_N)$$

其中 $\mathbf{t} = (t_1,...,t_N)$， $\mathbf{y} = (y_1,...,y_N)$， $\mathbf{I}_N$ 是 N 阶单位矩阵。

根据 Gaussian process 的定义， $\mathbf{y} = (y_1,...,y_N)$ 是一个 Gaussian 分布，也就是说：

$$p(\mathbf{y}) = N(\mathbf{y} \mid 0, \mathbf{K})$$

其中 $\mathbf{K}$ 是 kernel。

根据上面两个，可以得到关于 $\mathbf{t} = (t_1,...,t_N)$ 联合概率分布 $p(\mathbf{t})$（两个 Gaussian 分布的卷积）。

剩下要做的事情是，已经有的 observation $\mathbf{t} = (t_1,...,t_N)$，预测某个新的 input x_{N+1} 的 observation t_{N+1} 是什么。也就是计算概率 $p(t_{N+1} \mid \mathbf{t})$。由于 $p(\mathbf{t})$ 已经计算出来了，也就是相当于知道了 $p(t_1,...,t_N,t_{N+1})$；而这是一个 Gaussian 分布，显然可以求出 conditional probability $p(t_{N+1} \mid \mathbf{t})$ 的 analytical solution。

复杂度分析：假设有 N 个训练数据，那么训练的时间复杂度是 $O(N^3)$；进行一次 prediction 的复杂度是 $O(N^2)$。这是非常低的效率。对比 linear basis function 的 model，如果采用了 M 个 basis，则训练和预测的复杂度分别是 $O(M^3)$ 和 $O(M^2)$；这要比 Gaussian process 高效得多，因为 basis function M 的数目通常要远远小于训练数据的数目 N。

Gaussian 的优点何在呢？它实际上使用了无穷多个 basis function。或许其优点是：以 $O(N^3)$-$O(N^2)$ 为代价，实现了无穷个 basis function 的 linear basis function model。

6. 为Gaussian process选kernel

在实际中，rather than fixing the covariance function，we may prefer to use a parametric family of functions and then infer the parameter values from the data。

这样做增强了模型的灵活性。例如可以这样选择 kernel：

$$k(x_n, x_m) = \theta_0 \exp\{-\frac{\theta_1}{2}\| x_n - x_m \|^2\} + \theta_2 + \theta_3 x_n^T x_m$$

这里即含有 Gaussian kernel，也含有 linear kernel 与 constant。

在得到 observation $\mathbf{t} = (t_1, ..., t_N)$ 后，需要从中将参数 $\boldsymbol{\theta} = (\theta_0, ..., \theta_3)$ infer 出来。对于多个 kernel 线性组合出来的 kernel 而言，这个过程似乎是在进行“**kernel selection**”。$\boldsymbol{\theta}$ 可视为一个 hyper-parameter，因此对于它的估计可视为是 empirical Bayes。此时 marginal likelihood 是基于联合概率分布 $p(\mathbf{t})$ 得到，因为这里 $p(\mathbf{t}) \to p(\mathbf{t}|\boldsymbol{\theta})$。而这是一个 Gaussian 分布，容易得到 $\ln p(\mathbf{t}|\boldsymbol{\theta})$。之后就是 MLE 了。

7. Gaussian process for classification

Gaussian process 中，对于每个 input point x，y(x)是一个取值可以为任意实数的随机变量（Gaussian 分布）；为了 GP 能够用于分类，需要将 y(x)变换到(0, 1)区间中。采用 logistic 函数做此变换。

于是问题是这样（binary classification）：target variable $t \in \{0,1\}$；Gaussian process 定义在函数空间 $a(\mathbf{x})$ 上；logistic 函数对 a 函数做变换 $y = \sigma(a)$，y 则是一个非 Gaussian process，具有概率解释，即 $p(t|a) = \sigma(a)^t (1-\sigma(a))^{1-t}$。

由 Gaussian process 的定义，有：

$$p(\mathbf{a}_{N+1}) = N(|\mathbf{0}, \mathbf{C}_{N+1})$$

其中，$\mathbf{C}_{N+1}$ 由 kernel $\mathbf{K}$ 加上一个正因子而得到：$C(x_n, x_m) = k(x_n, x_m) + \nu\delta_n$，其中 $\nu > 0$ 事先固定，δ_{nm} 是克朗涅克函数。这么做是因为 kernel 只是非负定函数，为了使得 C 可逆，加入 ν 使得 C 是正定函数。

现在的目标是计算：

$$p(t_{N+1}|\mathbf{t}_N) = \int p(t_{N+1}=1|a_{N+1}) p(a_{N+1}|\mathbf{t}_N) d\, a_{N+1}$$

上式中，$p(t_{N+1}=1|a_{N+1}) = \sigma(a_{N+1})$。关键问题在于计算：

$$p(a_{N+1}|\mathbf{t}_N) = \int p(a_{N+1}, \mathbf{a}_N|\mathbf{t}_N) d\mathbf{a}_N = \int p(a_{N+1}|\mathbf{a}_N) p(\mathbf{a}_N|\mathbf{t}_N) d\mathbf{a}_N$$

上式中，有 Gaussian process 定义，$p(a_{N+1}|\mathbf{a}_N)$ 是一个 conditional Gaussian；因此关键问题

在于计算 $p(\mathbf{a}_N \mid \mathbf{t}_N)$。

$p(\mathbf{a}_N \mid \mathbf{t}_N)$ 并不好算，因此采取 Laplace approximation 的方法，用一个 Gaussian 分布近似它。这样 $p(a_{N+1} \mid \mathbf{t}_N)$ 就是 2 个 Gaussian 分布的卷积，仍旧是 Gaussian 分布； $p(t_{N+1} \mid \mathbf{t}_N)$ 则是一个 Logistic 与一个 Gaussian 的卷积，计算比较复杂，需要用近似。这样最终把已知 N 个观察，第 N+1 观察所属分类的概率计算得出来了。

<u>补充</u>:

几个用于求解 model parameters 时常用的知识

(1) Gaussian distribution

marginal Gaussian distribution， conditional Gaussian distribution； Convolution of two Gaussians， convolution of logistic and Gaussian

由于 logistic 可以用 probit 来近似，而后者是标准 Gaussian 的 CDF，所以 convolution of logistic and Gaussian 可以用 Convolution of two Gaussians 来近似。

关于 Convolution of two Gaussians：

(i) marginalize 的计算是两个函数的卷积

$$\int N(\mathbf{y};\mathbf{x},\Sigma_1)N(\mathbf{x};\boldsymbol{\mu}_2,\Sigma_2)d\mathbf{x} = \int N(\mathbf{y}-\mathbf{x};\mathbf{0},\Sigma_1)N(\mathbf{x};\boldsymbol{\mu}_2,\Sigma_2)d\mathbf{x}$$

观察上式的左端，其实是对参数 $\mathbf{x}$ 的 marginalize；而右端，发现其确实是两个函数的卷积：一个函数是 $f_{\Sigma_1}(\mathbf{y}-\mathbf{x}) = N(\mathbf{y}-\mathbf{x};\mathbf{0},\Sigma_1)$ ，其中 $\mathbf{0},\Sigma_1$ 是常量；另一个函数是 $g_{\boldsymbol{\mu}_2,\Sigma_2}(\mathbf{x}) = N(\mathbf{x};\boldsymbol{\mu}_2,\Sigma_2)$ ，其中 $\boldsymbol{\mu}_2,\Sigma_2$ 是常量。从而上式其实就是计算 $\int f_{\Sigma_1}(\mathbf{y}-\mathbf{x})g_{\boldsymbol{\mu}_2,\Sigma_2}(\mathbf{x})d\mathbf{x} = (f_{\Sigma_1} \otimes g_{\boldsymbol{\mu}_2,\Sigma_2})(\mathbf{y})$，显然是一个卷积。

注意：在这里，两个 Gaussian 的维度是相同的。

(ii) 是哪两个函数的卷积

假设有两个 Gaussian， $N(\boldsymbol{\mu}_1,\Sigma_1)$ 和 $N(\boldsymbol{\mu}_2,\Sigma_2)$，现在我们要求它们的卷积。那么就有：

$$(N(\boldsymbol{\mu}_1,\Sigma_1) \otimes N(\boldsymbol{\mu}_2,\Sigma_2))(\mathbf{t}) = \int N(\mathbf{t}-\mathbf{z};\boldsymbol{\mu}_1,\Sigma_1)N(\mathbf{z};\boldsymbol{\mu}_2,\Sigma_2)d\mathbf{z}$$

将此式对照(i)里的式子，可以发现：

$$\begin{aligned}\int N(\mathbf{y};\mathbf{x},\Sigma_1)N(\mathbf{x};\boldsymbol{\mu}_2,\Sigma_2)d\mathbf{x} &= \int N(\mathbf{y}-\mathbf{x};\mathbf{0},\Sigma_1)N(\mathbf{x};\boldsymbol{\mu}_2,\Sigma_2)d\mathbf{x} \\ &= (N(\mathbf{0},\Sigma_1) \otimes N(\boldsymbol{\mu}_2,\Sigma_2))(y)\end{aligned}$$

也就是说，这个 marginalization 计算的是 $N(\mathbf{0},\Sigma_1)$ 与 $N(\boldsymbol{\mu}_2,\Sigma_2)$ 的卷积。

(iii) 计算结果

由于 Gaussian $N(\boldsymbol{\mu},\Sigma)$的特征函数是：$e^{i\mathbf{t}^T\boldsymbol{\mu}-\frac{1}{2}\mathbf{t}^T\Sigma\mathbf{t}}$。根据卷积定理，$N(\boldsymbol{\mu}_1,\Sigma_1)$与 $N(\boldsymbol{\mu}_2,\Sigma_2)$的卷积的特征函数等于二者的特征函数的乘积，也就是$e^{i\mathbf{t}^T(\boldsymbol{\mu}_1+\boldsymbol{\mu}_2)-\frac{1}{2}\mathbf{t}^T(\Sigma_1+\Sigma_2)\mathbf{t}}$；该特征函数对应的分布正是：$N(\boldsymbol{\mu}_1+\boldsymbol{\mu}_2,\Sigma_1+\Sigma_2)$。

所以前面进行 marginalize 计算的结果是：

$$\int N(\mathbf{y};\mathbf{x},\Sigma_1)N(\mathbf{x};\boldsymbol{\mu}_2,\Sigma_2)d\mathbf{x}=(N(\mathbf{0},\Sigma_1)\otimes N(\boldsymbol{\mu}_2,\Sigma_2))(y)=N(\boldsymbol{\mu}_2,\Sigma_1+\Sigma_2)$$

(2) Taylor expansion

主要是多元的展开

(3) Laplace approximation

用 Gaussian 来近似一个复杂分布的方法

Chapter 7 Sparse Kernel Machine

1. Lagrange multiplier与KTT condition

PRML 讲的并不好。可以参考 Andrew Ng 的 Lecture note。

2. 关于SVM

SVM 是一个 discriminant function，即直接把一个 input 判别到一个分类中。相比之下，RVM 是一个 discriminant model，即训练后得到一个后验概率。

SVM 是一个 sparse model，即训练后只需要保存一部分 training data（support vectors）。相比之下，Gaussian process 训练后仍旧要保存全部 training data 以用于做 prediction。

SVM 的应用：classification，regression 和 novelty detection。

3. SVM的modeling过程：linearly separable的情况

考虑二类分类问题。有 N 个数据$\{\mathbf{x}_n, t_n\}$，其中$t_n \in \{-1,1\}$。

(1) **modeling 的基本思路**

假设有$y(\mathbf{x}) = \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}) + b$，那么$y(\mathbf{x}) = 0$就定义了一个高维空间的超平面。SVM 希望找到一个使得 margin 最大化的超平面作为 decision boundary。所谓 margin，指的是在全部 N 个数据中，离 decision boundary 最近的数据点到该超平面的距离。因此只要找到作为 decision boundary 的超平面$y(\mathbf{x}) = 0$的参数 w 和 b 的值，SVM 就训练完成了。在做 prediction 时，对于一个新的数据 x，计算 y(x)，判断其正负情况即可知道 x 当属于哪一类别。

(2) **modeling 的 mathematical formulation**

首先，任意一个点$\mathbf{x}_n$到超平面$y(\mathbf{x}) = 0$的距离是：$\frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|}$。而当 linearly separable 时，约束条件$t_n y(\boldsymbol{x}_n) > 0$对每个点都可以满足。因此可以写出 margin 最大化的目标函数：

$$\arg\max_{\mathbf{w},b}\{\frac{1}{\|\mathbf{w}\|}\min_n[t_n y(\mathbf{x}_n)]\}$$

$$s.t\ t_n y(\boldsymbol{x}_n) > 0,\ n = 1,...,N$$

其中，约束条件表明找出来的超平面必须是将两个类正确分开的；而目标函数表明在所有正确分开两个类别的超平面中，找 margin 最大的一个。直接求解这个问题非常困难。注意到一个性质：用同一个因子对 w 和 b 进行伸缩变换$\mathbf{w} \to \kappa\mathbf{w}, b \to \kappa b$后，任意点$\mathbf{x}_n$到超平面$y(\mathbf{x}) = 0$的距离是保持不变（即$\frac{t_n y(\mathbf{x}_n)}{\|\mathbf{w}\|}$不变）。我们可以据此人为规定：距超平面最近的点，到该平面的距离是 1，即$t_n y(\mathbf{x}_n) = 1$。那么，所有点到超平面的距离显然就有：

$t_n y(\mathbf{x}_n) \geq 1, n = 1, ..., N$ 。于是原目标函数就变成为在该约束条件下，求 $\arg\max_{\mathbf{w},b}\{\frac{1}{\|\mathbf{w}\|}\}$。

这等价于：

$$\arg\min_{\mathbf{w},b}\{\frac{1}{2}\|\mathbf{w}\|^2\}$$

$$s.t.\ t_n y(\mathbf{x}_n) \geq 1, n = 1, ..., N$$

这是一个含 inequality constraint 的 quadratic programming 问题。（**原问题**）

对此，先构造 Lagrange function：

$$L(\mathbf{w}, b, \mathbf{a}) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{n=1}^{N} a_n\{t_n(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}_n) + b) - 1\}$$

其中 Lagrange multiplier 向量 $\mathbf{a} \geq 0$。

求解原问题，等价于：$L(\mathbf{w}, b, \mathbf{a})$ 先对 $\mathbf{w}, b$ 求最小值，再对 $\mathbf{a}$ 求最大值。为此，先对 $\mathbf{w}, b$ 求偏导并置偏导为 0，从而得到二者与 $\mathbf{a}$ 的关系：$\mathbf{w} = \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}(\mathbf{x}_n)$ 和 $\sum_{n=1}^{N} a_n t_n = 0$ 。然后，再用此关系消除 $L(\mathbf{w}, b, \mathbf{a})$ 表达式中的 $\mathbf{w}, b$，从而得到了目标函数：

$$\tilde{L}(\mathbf{a}) = \sum_{n=1}^{N} a_n - \frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N} a_n t_n a_m t_m k(\mathbf{x}_n, \mathbf{x}_m)$$

$$s.t.\ a_n \geq 0, n = 1, ..., N$$

$$\sum_{n=1}^{N} a_n t_n = 0$$

最大化 $\tilde{L}(\mathbf{a})$ 并满足上述约束条件的 $\mathbf{a}$ 就是模型的解。

把 $\mathbf{w} = \sum_{n=1}^{N} a_n t_n \boldsymbol{\phi}(\mathbf{x}_n)$ 代入 $y(\mathbf{x}) = \mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}) + b$，得到

$$y(\mathbf{x}) = \sum_{n=1}^{N} a_n t_n k(\mathbf{x}, \mathbf{x}_n)$$

其中的 kernel 函数 k 和上面定义的一样，都是 $k(\mathbf{x}, \mathbf{x}') = \boldsymbol{\phi}(\mathbf{x})^T\boldsymbol{\phi}(\mathbf{x}')$。

(3) **模型的解的特点**

根据 KKT 条件，上述问题的解必满足：

$$a_n \geq 0$$

$$t_n y(\mathbf{x}_n) \geq 1$$

$$a_n\{t_n(y(\mathbf{x}_n) - 1\} = 0$$

这 3 条对任意 n 成立。注意第 3 条，表明 $a_n = 0$ 或 $t_n y(\mathbf{x}_n) = 1$ 至少有一个成立。如果 $a_n = 0$，

那么当做 prediction 时，用 $y(\mathbf{x})=\sum_{n=1}^{N}a_n t_n k(\mathbf{x},\mathbf{x}_n)$ 时，第 n 项就将为 0；这也就是说 $\mathbf{x}_n$ 对于做 prediction 不起作用，可以在训练完后就扔掉。而剩下 $a_n \neq 0$ 的点，就是 support vector。这些点满足 $t_n y(\mathbf{x}_n)=1$，也就是说它们是到超平面 $y(\mathbf{x})=0$ 最近的点。

SVM 模型的解的这些特点使其为一个 sparse model。

(4) 解 SVM 模型——SMO 算法

4. SVM：The overlapping case

实际中的数据点可能无法线性分离；而且即使做到了线性分离，也可能导致 overfitting。

为此对每个 training data $\mathbf{x}_n$，引入一个 slack variable $\xi_n \geq 0$，使得原来要求每个点到 decision boundary 的距离不小于 1，变成为不小于 $1-\xi_n$。也就是说约束条件 $t_n y(\mathbf{x}_n)\geq 1$ 变成为了 $t_n y(\mathbf{x}_n)\geq 1-\xi_n$。

相应的，目标函数改为最小化：$\arg\min_{\mathbf{w},b}\{\frac{1}{2}\|\mathbf{w}\|^2+C\sum_{n=1}^{N}\xi_n\}$，满足两个线性约束条件 $t_n y(\mathbf{x}_n)\geq 1-\xi_n$，$\xi_n \geq 0$。其中 C 是一个控制 penalty 与 margin 的量；在前面不考虑 overlapping 或 overfitting 的模型中，C 相当于无穷大。

该问题同样可用前面的方法引入 Lagrange function，然后对参数 $\mathbf{w},b$ 和 ξ_n 求最小化；再对 Lagrange multiplier 求最大化。在这个问题中，由于 ξ_n 作为参数引入，因此也要为其配备一个 Lagrange multiplier。不过在最后得到的 dual Lagrangian 中并不含 ξ_n 的 multiplier，dual Lagrangian 仍旧是 $\tilde{L}(\mathbf{a})=\sum_{n=1}^{N}a_n-\frac{1}{2}\sum_{n=1}^{N}\sum_{m=1}^{N}a_n t_n a_m t_m k(\mathbf{x}_n,\mathbf{x}_m)$，不过约束条件变为：$0\leq a_n \leq C, n=1,...,N$ 和 $\sum_{n=1}^{N}a_n t_n=0$（和之前的唯一不同之处是每个 a_n 有上限 C，这叫做 box constraints）。

和前面一样，$a_n=0$ 的 $\mathbf{x}_n$ 对 prediction 不起作用；但是 $a_n \neq 0$ 也不全是 support vector。如果 $0<a_n<C$，那么必有 $\xi_n=0$，此时 $\mathbf{x}_n$ 在 margin 上或在 margin 以里；但 $a_n \neq 0$，那么 $\mathbf{x}_n$ 只能是在 margin 上。对于 $a_n=C$，ξ_n 的取值并没有规律，所以此时的 $\mathbf{x}_n$ 可能正确分在了其所属类别，也可能分错了，要视 ξ_n 而定。

5. multiclass情况的SVM

这仍旧是一个 open problem。

一般采用 one-versus-the rest 的方法，这样就要训练 K 个 SVM。这种有两方面的不足：第一，存在 ambiguous region，这个区域中的点会被判定为属于多个类别，出现了 inconsistency；第二，面临正负样本不均衡的可能，因为“the rest”的一方很可能远比“one”的一方数据要多。

6. single-class情况的SVM

属于 unsupervised learning，类似于 probability density estimation，但其目的并不是估计 density of data，而是 find a smooth boundary enclosing a region of high density。该 boundary 的选取方法是：给定一个 0 到 1 之间的概率值，使得从数据的 distribution 中采样出的数据点落入 region 的概率为给定的概率值。

有两种常见的思路解决该问题。

第一种，在 feature space 中，找一个最小的超球，使其中包含的数据点不低于总数据的某一个比例值。

第二种，在 feature space 中，找一个超平面，要求该超平面与原点的距离最大化，同时满足在超平面的原点一侧的数据点数不超过总数据的某一个比例值。

与 Single-Class Classification 相类似的术语有：Outlier Detection，Novelty Detection 以及 Concept Learning。

7. SVM for regression

$\varepsilon - SVM$ 用于 regression 是这样：寻找一个尽可能 flat 的函数 y，使得对于所有的 training data，target value 与函数 y 计算的 value 之间的差值不超过 ε（对于每个数据 $\mathbf{x}_n$，都有 $|\mathrm{y}(\mathbf{x}_n)-t_n|<\varepsilon$）。所谓尽可能 flat 的函数，就是要求这个函数尽可能的“简单”（model complexity），以免 overfitting。这一点正可用 regularizer 来表达。因此可以把这个思想形式化（**optimization problem I**）：

$$\min\{\frac{1}{2}\|\mathbf{w}\|^2\}$$

$$st.\ |\mathrm{y}(\mathbf{x}_n)-t_n|<\varepsilon,\ n=1,...,N$$

以上 modeling 存在的问题跟 hard margin 分类的 SVM 一样，包括：(1) 可能无解（对应线性不可分）；(2) 即使有解，导致的模型也可能 complexity 太高，处于 overfitting 状态，对泛化不利。为此，我们允许每个点可以超出在 $\varepsilon - tube$ 之外——当然前提是需要为此付出一些“代价”。$\varepsilon - SVM$ 用来衡量超出 $\varepsilon - tube$ 的代价的函数是：

$$\mathrm{E}_{\varepsilon}(\mathrm{y}(\mathbf{x})-t)=\begin{cases}0, & |\mathrm{y}(\mathbf{x})-t|<\varepsilon \\ |\mathrm{y}(\mathbf{x})-t|-\varepsilon, & otherwise\end{cases}$$

此时的目标函数就可以改成（**optimization problem II**）：

$\min\{C\sum_{n=1}^{N}\mathrm{E}_{\varepsilon}(\mathrm{y}(\mathbf{x}_n)-t)+\frac{1}{2}\|\mathbf{w}\|^2\}$，其中 C > 0。

跟分类的 SVM 类似，这个问题可以引入 slack variables 来重新表述。对于每个数据 $\mathbf{x}_n$，引入两个 slack variables，$\xi_n \geq 0$ 和 $\hat{\xi}_n \geq 0$。当 $t_n > \mathrm{y}(\mathbf{x}_n)+\varepsilon$ 时，$\xi_n > 0$；否则 $\xi_n = 0$。当 $t_n < \mathrm{y}(\mathbf{x}_n)-\varepsilon$ 时，$\hat{\xi}_n > 0$；否则 $\hat{\xi}_n = 0$。用这两个变量为数据 $\mathbf{x}_n$ 提供比 $\varepsilon-tube$ 更大的“允许空间”，即要求：

$t_n \leq \mathrm{y}(\mathbf{x}_n)+\varepsilon+\xi_n$

$t_n \geq \mathrm{y}(\mathbf{x}_n)-\varepsilon-\hat{\xi}_n$

为这个更大的“允许空间”付出的“代价”就是 $\xi_n+\hat{\xi}_n$。因此得到目标函数（**optimization problem III**）：

$$\min\{C\sum_{n=1}^{N}(\xi_n+\hat{\xi}_n)+\frac{1}{2}\|\mathbf{w}\|^2\}$$

$$st.\ t_n \leq \mathrm{y}(\mathbf{x}_n)+\varepsilon+\xi_n$$

$$t_n \geq \mathrm{y}(\mathbf{x}_n)-\varepsilon-\hat{\xi}_n$$

$$\xi_n \geq 0$$

$$\hat{\xi}_n \geq 0$$

其中 $n=1,...,N$。

求解该目标函数关于 ξ_n，$\hat{\xi}_n$ 和 $\mathbf{w}$ 的最值，即解出了模型。方法还是用 Lagrange multiplier，而分析解的性质则用 KKT condition。

那么，以上 **optimization problem III** 和 **optimization problem II**、**optimization problem I** 的关系是什么呢？

如果 $\xi_n = 0$（$\hat{\xi}_n = 0$），那么数据 $\mathbf{x}_n$ 在 $\varepsilon-tube$ 之中；这时 $\mathbf{x}_n$ 满足 **optimization problem I** 的约束，并且使得 **optimization problem II** 和 **optimization problem III** 中引发的 penalty 均为 0，也就是说三者一样。

如果 $\xi_n > 0$（$\hat{\xi}_n > 0$），根据 KKT 条件分析发现，此时必有 $t_n = \mathrm{y}(\mathbf{x}_n)+\varepsilon+\xi_n$

（$t_n = \mathrm{y}(\mathbf{x}_n) - \varepsilon - \hat{\xi}_n$），这就导致的 $\mathrm{E}_\varepsilon(\mathrm{y}(\mathbf{x}_n) - t)$ 恰好等于 ξ_n（$\hat{\xi}_n$），同时有 $\hat{\xi}_n = 0$（$\xi_n = 0$）；这表明，施加在 **optimization problem II** 和 **optimization problem III** 的目标函数上的 penalty 是一样的。也就是说，二者一样。

综上分析，**optimization problem II** 和 **optimization problem III** 等价。

8. Relevance Vector Machine：RVM

这是一个 sparse Bayesian model。

相比 discriminant function 的 SVM，RVM 是一个 discriminant model，训练得到后验概率。而且 RVM 得到的结果一般比 SVM 更稀疏，因此更加有利 prediction 的效率。

而 SVM 与之相比，缺点是：complexity 参数 C 需要事先指定或者用 cross-validation 来寻找；by definition，SVM 只能用来二类分类等。

对于 RVM，方便而自然的思路是先从 regression 开始，然后应用于 classification。正和 SVM 的过程正好相反。

RVM for regression

RVM 其实就是一个 linear basis function model，只不过参数 $\mathbf{w}$ 的先验概率稍有不同。因此仍旧有以下：

$$y(\mathbf{x}, \mathbf{w}) = \mathbf{w}^T \boldsymbol{\phi}(\mathbf{x})$$

$$p(t \mid \mathbf{x}, \mathbf{w}, \beta) = N(t \mid y(\mathbf{x}, \mathbf{w}), \beta^{-1})$$

但是对于先验概率，则采用：

$$p(\mathbf{w} \mid \boldsymbol{\alpha}) = \prod_{i=1}^{M} N(w_i \mid 0, \alpha_i^{-1})$$

注意，这跟原来（$p(\mathbf{w} \mid \alpha) = N(\mathbf{w} \mid \mathbf{0}, \alpha \mathbf{I})$）的不同。在 RVM 中，参数 $\mathbf{w}$ 的每个分量都有一个独立的 hyper-parameter 来控制。

作为一个 Bayesian model，RVM 以通过 marginalize 来求 predictive distribution 为目标；但在此之前，需要用 maximize marginal likelihood 来估计出 hyper-parameter 的值，$\boldsymbol{\alpha}$ 和 β。假设有 N 个 observation $\mathbf{x}_n$（每个可以高维度），组合起来得到一个矩阵 $\mathbf{X}$；相应的 target values 也构成一个向量 $\mathbf{t} = (\mathrm{t}_1, ..., t_N)$。于是有 marginal likelihood：

$$p(\mathbf{t} \mid \mathbf{X}, \boldsymbol{\alpha}, \beta) = \int p(\mathbf{t} \mid \mathbf{X}, \mathbf{w}, \beta)\, p(\mathbf{w} \mid \boldsymbol{\alpha}) d\mathbf{w}$$

其中 $p(\mathbf{t} \mid \mathbf{X}, \mathbf{w}, \beta)$ 是一般的 likelihood。最大化此式（with respect to $\boldsymbol{\alpha}$ 和 β）得到 $\boldsymbol{\alpha}^*$ 和 β^*。那么，给定一个新的数据 $\mathbf{x}$，就有 predictive distribution：

$$p(t \mid \mathbf{x}, \mathbf{X}, \mathbf{t}, \boldsymbol{\alpha}^*, \beta^*) = \int p(t \mid \mathbf{x}, \mathbf{w}, \beta^*)\, p(\mathbf{w} \mid \mathbf{X}, \mathbf{t}, \boldsymbol{\alpha}^*, \beta^*) d\mathbf{w}$$

其中， $p(\mathbf{w}\mid\mathbf{X},\mathbf{t},\boldsymbol{\alpha}^*,\beta^*)$ 是 $\mathbf{w}$ 的后验概率。

RVM 的稀疏性：在对 marginal likelihood 最大化（with respect to $\boldsymbol{\alpha}$ 和 β ）的过程中，$\boldsymbol{\alpha}$ 向量的大量分量都将趋于无穷大，从而使分布 $N(w_i\mid 0,\alpha_i^{-1})$ 退化为一个固定的值 0；也就是说，这导致权重向量 $\mathbf{w}$ 中大量的 w_i 都将是 0。

RVM for classification

考虑 binary classification。将 $y(\mathbf{x},\mathbf{w})$ 做一个 logistic 变换，使其具有概率解释（认为是后验概率），即 $y(\mathbf{x},\mathbf{w})=\sigma(\mathbf{w}^T\boldsymbol{\phi}(\mathbf{x}))=p(C_1\mid\mathbf{x},\mathbf{w})$。然后假定参数的先验概率分布：

$$p(\mathbf{w}\mid\boldsymbol{\alpha})=\prod_{i=1}^{M}N(w_i\mid 0,\alpha_i^{-1})$$

先用 marginal likelihood 最大化的方法求出 $\boldsymbol{\alpha}^*$ （对于 classification，已经没有 β ）。剩下的工作就和 Bayesian Logistic Regression 类似了。即：先用 Laplace approximation 估计出后验概率，然后再用 marginalization 的方法求出 predictive distribution。

RVM 和 Bayesian Logistic Regression 的不同只有一个，即 RVM 的 prior 和后者不同（Bayesian Logistic Regression 的 prior 是 $p(\mathbf{w})=N(\mathbf{w}\mid\mathbf{m}_0,S_0)$，没有用超参数，而是取了固定的 parameter）。所以 RVM 有一个先行的工作是估计 $\boldsymbol{\alpha}^*$ 。之后就跟 Bayesian Logistic Regression 一样了。

Model	Time for Training	Time for Prediction	Remark
SVM (**Frequentist** Sparse Model)	N^2 approx.	V	
RVM(**Bayesian** Sparse Model)			
Gaussian Process	N^3	N^2	
Frequentist linear basis function model for regression	$M^2(N+M)$	M	
Bayesian linear basis function model for regression	$M^2(N+M)$ (get the necessities of the predictive distribution)	M^2 (specialize the predictive distribution for given data point)	
Frequentist Logistic regression			
Bayesian Logistic regression			
Frequentist Neural network			
Bayesian Neural network			

N: number of training data

M: number of basis function

V: number of support vectors

	Parametric/Non-parametric	Frequentist/Bayesian	Discriminative/Generative

Chapter 8 Graphical Models

1. 什么是Probabilistic Graphical Model（PGM）

一个 joint distribution 的 PGM 是指一个图，图中的每个顶点对应为该分布中的一个随机变量，而图中的边的含义是：

(1) 如果是有向图（Bayesian network），则对任意一个点 x_i，设其边的直接前驱节点的集合是 pa_i，那么：对于该 joint distribution，当给定 pa_i 后，x_i 与任意非其 descendant 的点条件独立。

(2) 如果是无向图（Markov network），则对任意一条边 (x_i, x_j)，那么：对于该 joint distribution，当给定除这两个点之外的节点（$\mathbf{x}\backslash\{x_i, x_j\}$）后，$x_i$ 与 x_j 不条件独立。

以上定义中，对一个 graph，用 joint distribution 的性质对其赋予了 semantics。这使得该 graph 中的图论性质可以对应解释为 joint distribution 中随机变量之间的性质（条件独立性）。因此，可把 graph 理解为对 joint distribution 中独立性的 modeling 或 encoding。

2. joint distribution中的独立性与graph中的独立性

上述对 PGM 的定义，要依赖于一个 joint distribution P；graph G 中每一条边能不能存在都要看该 joint distribution 是否满足相应的条件独立性质。

现在问题是：

(1) **除了在定义中所体现的外，图 G 是否还蕴含其他的条件独立性**？

完全有可能。

Bayesian network 在定义中体现了给定后与其非 descendant 节点的独立性，记全部这些独立性为集合 $I_\ell(G)$；而 Markov network 在定义中体现了对于不存在边直连的点 x_i, x_j，当除此之外其他点都给定后，二者的条件独立性，记全部这些独立性为集合 $I_p(G)$。

假设 I(G)表示图 G（有向或无向）所蕴含的全部条件独立性， I(P)表示 joint distribution P 所蕴含的全部条件独立性。这个问题就是，I(G)是否 $I_\ell(G)$ 或 $I_p(G)$ 可能比或更大？

对于 Bayesian network，假设由 **D-separation** 判定的全部条件独立性记为 $I_{d-sep}(G)$。所谓 D-separation 是指：假设 A，B，C 是三个互不相交的随机变量集合（都是 $\mathbf{x}=\{x_1,...,x_K\}$ 的子集），现在要判定 A 与 B 是否关于 C 条件独立。

考虑 A、B 中任意两点之间的所有可能的 path。一条 path 被 blocked 当且仅当其包含了这么一个节点：

(i)该节点上的边箭头出现的是 head-to-tail 或 tail-to-tail 的情况，并且该节点属于 C；**或者**

(ii)该节点上的边箭头出现的是 head-to-head 的情况，并且该节点以及该节点的全部 decedents 都不属于 C。

如果 A 与 B 之间全部的 path 都被 blocked 了，那么称 A 与 B 被 C 给 d-separated 了；此时就有 A 与 B 关于 C 条件独立。

（上述 **path 的含义：就是忽略掉有向图中的箭头，当作一个无向边的路径**。只是在检查该 path 是否 blocked 的时候，才又把箭头考虑进去。）

对于 Markov network，假设由 **U-separation** 判定的全部条件独立记为 $I_{u-sep}(G)$。U-separation 是指：对于两个 G 中节点的子集 X 和 Y，如果对于给定的另一个节点子集 Z，使得 X 和 Y 中任意两个节点之间的任意路径都必经过 Z，那么 X 和 Y 关于 Z 条件独立。

现在的要解决的问题有两个：第一，D-separation 和 U-separation 所判定为条件独立的 statement，对于 joint distribution P 而言成立（它们找出来的确实是条件独立性）。第二，$I_{d-sep}(G)=I(G)$（有向图）或 $I_{u-sep}(G)=I(G)$（无向图）。

幸运的是，以上两条都可证明是正确的。

(2) **图 G 的是否蕴含了分布 P 中全部的条件独立性？**

不一定。能够保证的是，对于由分布 P 定义出来的 G，有 $I(G)\subseteq I(P)$。也就是说，G 是 P 的一个 I-map。

(3) **图 G 是否会蕴含 P 中不满足的条件独立性？**

不会。其实(2)和(3)可以通过证明 $I(G)\subseteq I(P)$ 而一起回答。

I-map：如果一个 graph 满足 $I(G)\subseteq I(P)$，则成这个 graph 是这个 distribution 的 I-map。

D-map：如果一个 graph 满足 $I(G)\supseteq I(P)$，则成这个 graph 是这个 distribution 的 D-map。

Perfect map: 如果一个 graph 满足 $I(G)=I(P)$，则成这个 graph 是这个 distribution 的 D-map。

并不是所有的分布都有 perfect map；

记：全部有 Bayesian network 作为 perfect map 的分布为 DD，全部有 Markov network 作为 perfect map 的分布为 UD，那么 $DD\neq UD,\ DD\cap UD\neq\varnothing$。

3. PGM的基本意义是什么

对于一个分布 P，以及由之定义出来的 PGM G，G 对于 P 有什么意义？根据上面的讨论，

我们唯一知道的事情是：G 是 P 的一个 I-map。也就是说，这么定义出来的一个图 G，它所蕴含的全部条件独立性 I(G)是 P 所蕴含全部条件独立性 I(P)的一个子集。

这件事情的基本意义是：

定理（factorization 与 I-map）

对于 Bayesian network：分布 P 可以 factorizes over G，当且仅当 G 是 P 的 I-map。

对于 Markov network：如果 P 是一个 Gibbs distribution factorizes over G，那么 G 是 P 的 I-map；反之，假设 P 是一个 positive distribution，如果 G 是 P 的一个 I-map，那么 P 是一个 Gibbs distribution factorizes over G。

上述需要解释的地方是“P factorizes over G”的意思。

P factorizes over G：

对于 Bayesian network，是指：给定图 G，分布 P 可以按照 G 进行这样的分解 $p(\mathbf{x})=\prod_{k=1}^{K} p(x_k \mid pa_k)$，其中 pa_k 表示的是节点 x_k 的前驱节点集合，K 是全部 random variables 数。

对于 Markov network，是指：给定图 G，分布 P 可以按照 G 进行这样的分解 $p(\mathbf{x})=\frac{1}{Z}\prod_{C}\psi_C(X_C)$，其中 X_C 是图中的一个极大团，$\psi_C(X_C)\geq 0$ 定义在该极大团上的函数（potential function），$p(\mathbf{x})$ 把全部的极大团的 potential function 相乘，然后用 Z 来归一化。

因此，对于图的理解是：**Conditional Independence 的角度，以图表达条件独立性；Factorization 的角度，以图表达分解性。**

4. 从joint distribution到graph

上面仍有以下问题没解决：

(1) 一个分布 P 可以定义多少个 graph？

由于 P 可定义 G，当且仅当 G 是 P 的 I-map；所以，这个问题等价于：分布 P 的 I-map 有多少个。G 是 P 的一个 I-map，也就是说 $I(G)\subseteq I(P)$。因此，任意一个图 G，只要它满足这一点，它就是 P 的 I-map，也就能够由分布 P 定义出来。最 trivial 的例子是完全图，因为这种图的 $I(G)\neq\varnothing$，没表达任何条件独立性。所以完全图是任意分布 P 的 I-map。

(2) 哪个 I-map 蕴含的条件独立数最接近分布 P 所蕴含的条件独立数？

这个问题等价于：寻找分布 P 一个 I-map G，使 I(G)最大。或者说，怎么构造出这样的一个 I-map 图。事实上，这样的 I-map 叫做 **minimal I-map**。

构造算法是：

首先，每个 random variable 表达为图中的一个顶点。

然后，在节点之间加入边。这时候有两套方法，即 Bayesian network 和 Markov network。

对于无向图 Markov network，边是这样建立：任意两个节点 x_i, x_j，如果它们关于 $\mathbf{x}\setminus\{x_i, x_j\}$ 条件独立，那么二者之间无边连接；否则连接一条无向边。对于给定的 joint distribution 来说，我们可以反复的询问它上述问题 C_K^2 次（K 是 random variable 个数），从而构建起该分布的无向图 I-map。

对于有向图的 Bayesian network，边则这样建立：任意节点 x_i，考虑条件分布 $p(x_i \mid x_1,...,x_{i-1})$。假如存在变量 $x_j \in \{x_1,...,x_{i-1}\}$，在此 joint distribution 中满足条件独立 $p(x_i \mid x_1,...,x_{i-1}) = p(x_i \mid \{x_1,...,x_{i-1}\}\setminus x_j)$，那么把 x_j 从 $\{x_1,...,x_{i-1}\}$ 中删除；此过程持续，直到不存在这样的变量，得到一个子集 $pa_i \subset \{x_1,...,x_{i-1}\}$。那么就以每个 pa_i 中的顶点为起点，以 x_i 为终点连有向边。对于给定的 joint distribution 来说，我们可以逐个节点的考虑，从而构建起该分布的有向图 I-map。（注意：考虑节点 x_i 时，我们给予它的 condition 是 $\{x_1,...,x_{i-1}\}$，这样做确保了最后的 graph 一定是一个 DAG 图，从而符合 Bayesian network 的定义。）

以上算法需要证明：这样构造出来的的确是 I-map；以及不存在比这更小（蕴含条件独立更多）的 I-map。

(3) 是否存在两个不同的 graph，它们都是 P 的 I-map，而且蕴含的条件独立集合相同？

由于 graph 可以看作是一个 conditional independences 的集合，那么，如果两个 graph 其所蕴含的全部 conditional independences 一样，那么就认为这两个图是等价的，称为 **I-equivalent**。这是一个等价关系，全部 graph 可在此关系下划分成等价类。两个结构很不一样的 graph，在 I-equivalent 关系下都可能是等价的。

以有向图 DAG 为例：

Skeleton：一个有向图的 skeleton 是指把图中的有向箭头去掉后剩下的无向图基图。

Immorality：有向图中的一个 head-to-head 结构（即 X→Z←Y 结构）是一个 immorality，如果 X 和 Y 之间没有直接相连的边。

定理：假如 G1 和 G2 是同一组随机变量上的 graph，二者 I-equivalent 当且仅当 G1 与 G2 的 skeleton 相同，而且具有相同的 immorality。

5. 再回到 graph 中的独立性

Local independence 与 global independences：

对于一个 Bayesian network G：

Local independence 指的是：任意一个 x_k，当给定 pa_k 后，x_k 与除它的 descendants 之外的任何节点独立。记全部 Local independence 为 $I_\ell(G)$。

Global independence 指的是：根据 D-separation 导出的条件独立的集合。记全部 Global independence 为 $I(G)$。

二者满足关系：$I_\ell(G) \subseteq I(G) \subseteq I(P)$，P 是 joint distribution。事实上，$I_\ell(G)$ 与 $I(G)$ 是等价的，$I_\ell(G)$ 可以推出 $I(G)$，反之则显然。

另外，$I(G)$ 是完备的，也就是说，任意 G 可反映的 conditional independence，都一定已经包含在了 $I(G)$ 中。

对于一个 Markov network G：

Local independence 有两种。第一种是 **pairwise independence**，即对任意两个节点 x_i, x_j，当给定除它们外的其他点后，二者独立；这种独立性记为 $I_p(G)$。第二种是 **Markov blanket independence**，即对任意一个节点 x_k，当给定它的 Markov blanket 后，该节点与其他非 Markov blanket 上的节点独立；这种独立性记为 $I_\ell(G)$。

Global independence 指的是：对于两个 G 中节点的子集 X 和 Y，如果对于给定的另一个节点子集 Z，使得 X 和 Y 中任意两个节点之间的任意路径都必经过 Z，那么 X 和 Y 关于 Z 条件独立。记这种独立性为 $I(G)$。

以上三者的关系是 $I_p(G) \subseteq I_\ell(G) \subseteq I(G)$。事实上如果分布 P 是一个 positive distribution，那么三者是等价的。

另外，$I(G)$ 是完备的。

6. <u>PGM的思想过程与实际过程</u>

PGM 思想过程

Joint distribution 是困难的（random variable 太多，之间的关系太复杂，导致分布的维度太高，难以获得 direct representation）。但是 random variables 之间存在的独立性为简化 joint distribution 提供了可能。把 joint distribution 所蕴含的独立性找出来，表达到一个 graph 中。然后 work on the graph，基于 graph 研究 inference 的算法（计算 marginal 或 conditional）。

PGM 的实际过程

实际中连 joint distribution 本身可能都难以获得或表达，更别说为它寻找 I-map 了。所以实际过程中，是**直接把 graph 接受下来，把其当作 joint distribution 的 I-map，然后 work on this graph**。

Chapter 9 Mixture Models and EM

1. EM算法（Frequentist：**Maximum Likelihood EM**）

Probabilistic model 中有 observed variables **X** 和 latent variables **Z**。二者的 joint distribution 为 $p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\theta})$，由参数 $\boldsymbol{\theta}$ 控制。（对于 Frequentist 而言，显然 $\boldsymbol{\theta}$ 是一个固定不变的值，而不是随机变量。）

问题目标：找到 $\boldsymbol{\theta}$ 参数，使得 observed data 的似然最大化：$p(\boldsymbol{X}\mid\boldsymbol{\theta})=\sum_{\boldsymbol{Z}} p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\theta})$。

问题特点：直接对 $p(\boldsymbol{X}\mid\boldsymbol{\theta})$ 求最值困难，但对 complete-data likelihood $p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\theta})$ 相对容易。但是，由于 $\boldsymbol{Z}$ 观察不到，这个容易只是数学上的。

EM 算法：

引入一个分布 $q(\boldsymbol{Z})$，从而把对数似然 $\ln p(\boldsymbol{X}\mid\boldsymbol{\theta})$ 分解如下（这个分解对任意 $q(\boldsymbol{Z})$ 成立）：

$$\ln p(\boldsymbol{X}\mid\boldsymbol{\theta})=L(q,\boldsymbol{\theta})+KL(q\,\|\,p)$$

其中：

$$L(q,\boldsymbol{\theta})=\sum_{\boldsymbol{Z}} q(\boldsymbol{Z})\ln\{\frac{p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\theta})}{q(\boldsymbol{Z})}\}$$

$$KL(q\,\|\,p)=-\sum_{\boldsymbol{Z}} q(\boldsymbol{Z})\ln\{\frac{p(\boldsymbol{Z}\mid\boldsymbol{X},\boldsymbol{\theta})}{q(\boldsymbol{Z})}\}$$

EM 算法分两步迭代进行：假设 $\boldsymbol{\theta}$ 当前的值是 $\boldsymbol{\theta}^{old}$。

(1) 固定在 $\boldsymbol{\theta}^{old}$ 下，maximize $L(q,\boldsymbol{\theta})$ with respect to $q(\boldsymbol{Z})$。这只需要使 $q(\boldsymbol{Z})=p(\boldsymbol{Z}\mid\boldsymbol{X},\boldsymbol{\theta}^{old})$ 即可。（E step）

(2) 在固定 $q(\boldsymbol{Z})=p(\boldsymbol{Z}\mid\boldsymbol{X},\boldsymbol{\theta}^{old})$ 下，maximize $L(q,\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$。这样就会得到一个新的参数 $\boldsymbol{\theta}^{new}$。（M step）

注意，上面一轮迭代后为什么可以 $\ln p(\boldsymbol{X}\mid\boldsymbol{\theta})$ 使增大。在 E step 中，其实 $\ln p(\boldsymbol{X}\mid\boldsymbol{\theta})$ 没有改变，因为这步是针对 $q(\boldsymbol{Z})$ 优化，而改变 $q(\boldsymbol{Z})$ 并不能改变 $\ln p(\boldsymbol{X}\mid\boldsymbol{\theta})$；在 M step 中，$\ln p(\boldsymbol{X}\mid\boldsymbol{\theta})$ 由于新的参数而增大，因为 $L(q,\boldsymbol{\theta}^{new})\geq L(q,\boldsymbol{\theta}^{old})$，而在新的参数 $\boldsymbol{\theta}^{new}$ 下，$q(\boldsymbol{Z})$（已经在 E step 中置为 $p(\boldsymbol{Z}\mid\boldsymbol{X},\boldsymbol{\theta}^{old})$），不等于 $p(\boldsymbol{Z}\mid\boldsymbol{X},\boldsymbol{\theta}^{new})$，所以 KL(q||p)大于 0。这些变化表明：$\ln p(\boldsymbol{X}\mid\boldsymbol{\theta}^{new})>\ln p(\boldsymbol{X}\mid\boldsymbol{\theta}^{old})$。这样 EM 算法每迭代一次，参数都在向着目标前进。

进一步解释 EM 算法：

(1) $\ln p(\boldsymbol{X}\mid\boldsymbol{\theta})$ 与 $q(\boldsymbol{Z})$ 无关，因此任意改变 $q(\boldsymbol{Z})$ 不影响的 $\ln p(\boldsymbol{X}\mid\boldsymbol{\theta})$ 值。所以，关于

$q(\boldsymbol{Z})$ 对 $L(q,\boldsymbol{\theta})$ 最大化，只需要使 $q(\boldsymbol{Z}) = p(\boldsymbol{Z} \mid \boldsymbol{X},\boldsymbol{\theta})$，即 KL(q||p)为 0。

(2) 在 M step 中，如何 maximize $L(q,\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$？事实上，固定 $q(\boldsymbol{Z})$ 为（观察到 **X** 后）**Z** 的后验概率后，

$$L(q,\boldsymbol{\theta}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{Z} \mid \boldsymbol{X},\boldsymbol{\theta}^{olds}) \ln p(\boldsymbol{X},\boldsymbol{Z} \mid \boldsymbol{\theta}) - \sum_{\boldsymbol{Z}} p(\boldsymbol{Z} \mid \boldsymbol{X},\boldsymbol{\theta}^{olds}) \ln p(\boldsymbol{Z} \mid \boldsymbol{X},\boldsymbol{\theta}^{olds})$$

$$= \sum_{\boldsymbol{Z}} p(\boldsymbol{Z} \mid \boldsymbol{X},\boldsymbol{\theta}^{olds}) \ln p(\boldsymbol{X},\boldsymbol{Z} \mid \boldsymbol{\theta}) + \mathrm{const}$$

注意上式中减号后面的部分已经与 $\boldsymbol{\theta}$ 无关，所以 maximize $L(q,\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ 等价于最大化：

$$Q(\boldsymbol{\theta},\boldsymbol{\theta}^{old}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{Z} \mid \boldsymbol{X},\boldsymbol{\theta}^{old}) \ln p(\boldsymbol{X},\boldsymbol{Z} \mid \boldsymbol{\theta})$$

with respect to $\boldsymbol{\theta}$。而 $Q(\boldsymbol{\theta},\boldsymbol{\theta}^{old})$，正是 complete-data likelihood $\ln p(\boldsymbol{X},\boldsymbol{Z} \mid \boldsymbol{\theta})$（观察到 **X** 后的）**Z** 后验概率 $p(\boldsymbol{Z} \mid \boldsymbol{X},\boldsymbol{\theta}^{old})$ 下的期望。可见，M step 的 Maximize 的正就是 complete-data likelihood 的期望。

EM 算法的名称得到了解释（E step 是期望步，计算出期望 $Q(\boldsymbol{\theta},\boldsymbol{\theta}^{old})$；M step 是最大化步，就是关于 $\boldsymbol{\theta}$ 最大化 $Q(\boldsymbol{\theta},\boldsymbol{\theta}^{old})$）。

以上 EM 求解的是（使 observed variables 的）似然最大化的参数，这属于 Frequentist 的框架；这暗示着还有 Bayesian 的 EM。

2. Gaussian Mixture Model（GMM）

首先，Gaussian mixture distribution 是指分布：$p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$。可以验证这的确是一个分布。而且它不属于 Exponential Family。

假设 random vector **z** 是一个 1-of-K 的编码方式，其分布是：$p(\boldsymbol{z}) = \prod_{k=1}^{K} \pi_k^{z_k}$。其中，$0 \le \pi_k \le 1$，且 $\sum_{k=1}^{K} \pi_k = 1$。

假设有条件分布：$p(\boldsymbol{x} \mid z_k = 1) = N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$。也就是说，有 **x** 关于 **z** 的条件分布：$p(\boldsymbol{x} \mid \boldsymbol{z}) = \prod_{k=1}^{K} N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)^{z_k}$。

那么就可以得到 **x** 的边缘分布：$p(\boldsymbol{x}) = \sum_{\boldsymbol{z}} p(\boldsymbol{z}) p(\boldsymbol{x} \mid \boldsymbol{z}) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$。

GMM 的问题是：当得到 N 个 **x** 的观察后，确定的 Gaussian mixture distribution 中的参数。

以上问题对应到 EM 的思路中：**z** 是 latent variable，**x** 是 observed variable。假设得到 N 个观察，记为 **X**，其中每行是一个观察值 **x**；每个观察值都会有一个相应的 latent variable 值，记为 **Z**，同样也是每行一个 **z**。

根据这些 observation，需要对 Gaussian mixture distribution 进行 inference，找到其中的三大组参数：$\boldsymbol{\pi}=\{\pi_k : k=1,...,K\}$，$\boldsymbol{\mu}=\{\boldsymbol{\mu}_k : k=1,...,K\}$，$\boldsymbol{\Sigma}=\{\boldsymbol{\Sigma}_k : k=1,...,K\}$。

从 EM 的思路来说，这些参数将通过最大化观察的似然 $p(\boldsymbol{X}\mid\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})$ 来得到。根据 GMM，这个似然是：$\ln p(\boldsymbol{X}\mid\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})=\sum_{n=1}^{N}\ln\{\sum_{k=1}^{K}\pi_k N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\}$。其实的对数 ln 不再是直接作用于 Gaussian，而是有一个求和。因此对参数求导再置为 0，无法得到 closed form solution。

另一方面，complete-data log likelihood 却相对容易求解。根据 GMM，这个似然是：$p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})=\prod_{n=1}^{K}\prod_{k=1}^{K}\pi_k^{z_{nk}}N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)^{z_{nk}}$。取对数后是：

$$p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})=\sum_{n=1}^{K}\sum_{k=1}^{K}z_{nk}\{\ln\pi_k+\ln N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\}$$

可以发现，在数学上 complete-data log likelihood 甚至可以直接求出解析解。但其问题在于，其中的 **Z** 是一个 latent variable，根本没有实际观察到的值，因此 complete-data log likelihood 没法算。按照 EM，应该计算的是这个 likelihood 在 **Z** 的后验概率下的期望。

Z 的后验概率是：$p(\boldsymbol{Z}\mid\boldsymbol{X},\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})\propto\prod_{n=1}^{K}\prod_{k=1}^{K}[\pi_k N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)]^{z_{nk}}$。（根据 p(**z**)和 p(**x**|**z**)）

在 **Z** 的后验概率下，complete-data log likelihood 的期望是：

$E_{\boldsymbol{Z}}[p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})]=\sum_{n=1}^{K}\sum_{k=1}^{K}\gamma(z_{nk})\{\ln\pi_k+\ln N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)\}$，其中：$\gamma(z_{nk})=E[z_{nk}]$ 是 z_{nk} 在 **Z** 的后验概率下的期望。计算公式是：

$$\gamma(z_{nk})=E[z_{nk}]=\frac{\pi_k N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_k,\boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K}\pi_j N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_j,\boldsymbol{\Sigma}_j)}$$

通过对 $\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma}$ 求导最大化 $E_{\boldsymbol{Z}}[p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})]$ 的方式，可以得到公式：

$$\boldsymbol{\mu}_k=\frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_{nk})\boldsymbol{x}_n$$

$$\pi_k=\frac{N_k}{N}$$

$$\boldsymbol{\Sigma}_k=\frac{1}{N_k}\sum_{n=1}^{N}\gamma(z_n)(\boldsymbol{x}_{kn}-\boldsymbol{\mu}_k)(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^T$$

其中，$N_k = \sum_{n=1}^{N} \gamma(z_{nk})$

总结 GMM 的过程：假设已经有参数 $\boldsymbol{\pi}^{old}, \boldsymbol{\mu}^{old}, \boldsymbol{\Sigma}^{old}$ 。那么算法就是两步的循环迭代：第一步，计算在这组参数下，在 **Z** 的后验概率下 z_{nk} 的期望 $\gamma(z_{nk})$（E step，**Z** 的后验概率已经隐含在 $\gamma(z_{nk}) = E[z_{nk}]$ 的计算中）；第二步，计算 $E_Z[p(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma})]$ 关于三个参数 $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ 的最大值，从而得到新的参数 $\boldsymbol{\pi}^{new}, \boldsymbol{\mu}^{new}, \boldsymbol{\Sigma}^{new}$ 。

GMM 的应用于 clustering：假设将 GMM 应用于 clustering 问题。总共有 K 个类别。对于每个 **x**，采用 1-of-K 方式的向量 **z** 其实编码的信息是 **x** 所属的类别；其中为 1 的那个维度代表其所属的类别。注意到上述 $\gamma(z_{nk})$，其实是：

$p(z_{nk}=1 \mid \boldsymbol{x}_n) = \dfrac{p(z_{nk}=1)p(\boldsymbol{x}_n \mid z_{nk}=1)}{\sum_{j=1}^{K} p(z_{nj}=1)p(\boldsymbol{x}_n \mid z_{nj}=1)} = \dfrac{\pi_k N(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)}{\sum_{j=1}^{K} \pi_j N(\boldsymbol{x}_n \mid \boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j)} = \gamma(z_{nk})$。也就是说，

$\gamma(z_{nk})$ 是当 $\boldsymbol{x}_n$ 给定后，$\boldsymbol{x}_n$ 属于第 k 个类别的后验概率。那么，当 GMM 的训练过程结束时，用三大参数 $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ 就可以计算出全部 $\gamma(z_{nk})$，从而给出了一个聚类的方案。

3. GMM观点的k-means

这个观点简单概括就是：k-means 是 GMM 的一个特例。

首先应该认识到，对于 clustering，k-means 是一个 hard assignment of data points to clusters；但是 GMM 是一个 soft 的 assignment，因为它计算的是一个后验概率 $\gamma(z_{nk})$。

假设有一个 GMM，每个分支的均值向量是 $\boldsymbol{\mu}_k$，协方差矩阵是 $\varepsilon \mathbf{I}$。也就是，每个分支有各自的均值，但是它们的协方差相同。代入上面 GMM 的公式，可以得到：

$$\gamma(z_{nk}) = \frac{\pi_k \exp\{-\| \boldsymbol{x}_n - \boldsymbol{\mu}_k \|^2 / 2\varepsilon\}}{\sum_{j=1}^{K} \pi_j \exp\{-\| \boldsymbol{x}_n - \boldsymbol{\mu}_j \|^2 / 2\varepsilon\}}$$

假设在所有 K 个 $\| \boldsymbol{x}_n - \boldsymbol{\mu}_j \|^2$ 中，最小的一个是 $\| \boldsymbol{x}_n - \boldsymbol{\mu}_{j^*} \|^2$。上式的分子分母同除以 $\exp\{-\| \boldsymbol{x}_n - \boldsymbol{\mu}_{j^*} \|^2 / 2\varepsilon\}$，可以发现，当 $\varepsilon \to 0$ 时：若 $k \neq j^*$，分母为 1，分子为 0；若 $k = j^*$，分母和分子都为 1。这就表示：

$$\lim_{\varepsilon \to 0} \gamma(z_{nk}) = \begin{cases} 0, k \neq j^* \\ 1, k = j^* \end{cases}$$

这表明，$(\lim_{\varepsilon \to 0} \gamma(z_{n1}), \ldots, \lim_{\varepsilon \to 0} \gamma(z_{nK}))$ 成为了一个了一个 1-of-K 编码方式。这时候 GMM 也成

为了一个 hard assignment 的聚类了。

对于其他参数，可以看到在 $\varepsilon \to 0$ 的过程中，都变得跟 k-means 一样。N_k 成为第 k 类中的样本点数；π_k 成为第 k 类的样本数占总数的比例，但在 k-means 中这个量已经没有用处；$\boldsymbol{\mu}_k$ 成为第 k 类所有样本的中心点。K-means 只估计 cluster means 而不考虑 cluster covariance。最后，expected complete-data log likelihood 在 **Z** 的后验概率下的期望：

$$E_{\boldsymbol{Z}}[p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})]\to-\frac{1}{2}\sum_{n=1}^{K}\sum_{k=1}^{K}\gamma(z_{nk})\|\boldsymbol{x}_n-\boldsymbol{\mu}_k\|^2+const$$

也就是，GMM 最大化 $E_{\boldsymbol{Z}}[p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Sigma})]$ 等价于 k-means 的最小化 distortion。

补充：

识别一个分布 p(x)是何种分布的方法

——取对数 lnp(x)，然观察其中的 x 的形式。此处 x 可以是一个 random variable，包括 Bayesian 中作为参数的 random variable。

例如：

如果 lnp(x)是 x 是二次函数，那么 p(x)就是 Gaussian。

如果 lnp(x)是 x 和 lnx 的线性组合函数，那么 p(x)就是 Gamma。

Chapter 10 Approximate Inference

1. Approximation

Probabilistic model 中的一个 central task：给定一组 observation **X** 后，计算 latent variables **Z** 的后验概率 P(**Z**|**X**)。以及一些 expectation with respect to P(**Z**|**X**)。很多情况下 P(**Z**|**X**)是 analytically intractable 的。这就需要有 approximation 方法。

Latent variable：只要没有观察到的都归为 latent variable，比如在 Bayesian 中的 parameter（它们是 random variable）。在 Probablistic Graphical Model 的观点看，parameter 和狭义的 latent variable 的不同就是，parameter 的个数和观察到的数据的个数无关，但是狭义的 latent variable 则与其相关。

Approximation 方法：分为 deterministic 方法和 stochatic 方法。前者包括 Laplace approximation，variational inference 等；后者包括 MCMC sampling 等。

2. Variational inference

问题：一个 probablistic model P(**X**, **Z**)，含有 observed variables $\boldsymbol{X}=\{\boldsymbol{x}_1,...,\boldsymbol{x}_N\}$ 和 latent variable $\boldsymbol{Z}=\{\boldsymbol{z}_1,...,\boldsymbol{z}_N\}$

目的：为后验概率 P(**Z**|**X**)和 model evidence P(**X**)找 approximation。

思路：

引入一个分布 $q(\boldsymbol{Z})$，从而把 P(**X**)分解开来：$\ln p(\boldsymbol{X})=L(q)+KL(q\|p)$。其中

$$L(q)=\int q(\boldsymbol{Z})\ln\{\frac{p(\boldsymbol{X},\boldsymbol{Z})}{q(\boldsymbol{Z})}\}d\boldsymbol{Z}$$

$$KL(q\|p)=-\int q(\boldsymbol{Z})\ln\{\frac{p(\boldsymbol{Z}\mid\boldsymbol{X})}{q(\boldsymbol{Z})}\}d\boldsymbol{Z}$$

注意，现在要用 $q(\boldsymbol{Z})$ 来近似 P(**Z**|**X**)。如何衡量二者的相近程度呢？上式中的 KL(q||p)正是一个合适的指标。因此，现在就要找到一个 q(**Z**)，使 KL(q||p) 最小化。

然后，P(**Z**|**X**)本身就是 intractable 的，所以直接难以找到使 KL(q||p) 最小化的 q(**Z**)。但是如果 joint distribution P(**X**, **Z**)更容易处理，那么就有了一个思路：由于 $\ln p(\boldsymbol{X})$ 的值跟 q(**Z**)的选取无关，所以最小化 KL(q||p)，等价于最大化 L(q)。

假设：q(**Z**)的范围是极其大的，为了便于求出最大化 L(q)的解，需要给 q(**Z**)一些限制。给予限制的原则是兼顾 tractable 与 flexible。常用的限制/假设是：

$$q(\boldsymbol{Z})=\prod_{i=1}^{M}q_i(\boldsymbol{Z}_i)$$

即分解性质。其中的 $\boldsymbol{Z}_i$ 构成 **Z** 的一个不交子集族。

q(**Z**)被称为 variational distribution。

推导：把 q(**Z**)的分解性质代入 L(q)之中，得到

$$L(q)=\int\prod_i q_i\{\ln p(\boldsymbol{X},\boldsymbol{Z})-\sum_i \ln q_i\}d\boldsymbol{Z}$$

$$=\int q_j \ln \tilde{p}(\boldsymbol{X},\boldsymbol{Z}_j)d\boldsymbol{Z}_j-\int q_j \ln q_j d\boldsymbol{Z}_j+const$$

$$=-KL(q_j \| \ln \tilde{p}(\boldsymbol{X},\boldsymbol{Z}_j))+const$$

其中，$\ln \tilde{p}(\boldsymbol{X},\boldsymbol{Z}_j)=E_{i\neq j}[p(\boldsymbol{X},\boldsymbol{Z})]+const$，而 $E_{i\neq j}$ 表示的是在分布 $\prod_{i\neq j} q_i(\boldsymbol{Z}_i)$ 下求期望（可以验证这确实是一个分布）。最大化 L(q) with respect to q_j，等价于最小化 $KL(q_j \| \ln \tilde{p}(\boldsymbol{X},\boldsymbol{Z}_j))$ with respect to q_j。根据 KL 的性质，有：

$$\ln q^*(\boldsymbol{Z}_j)=E_{i\neq j}[p(\boldsymbol{X},\boldsymbol{Z})]+const$$

结果：

以上给出了 $\ln q^*(\boldsymbol{Z}_j)$ 计算公式，但是它依赖于其他的 M-1 个因子组成的分布 $\prod_{i\neq j} q_i(\boldsymbol{Z}_i)$。

所以 variational inference 求解使 L(q)最大化的 q(**Z**)的方法是：

首先，初始化每一个 $q_i(\boldsymbol{Z}_i)$；

然后，逐个考虑 $q_i(\boldsymbol{Z}_i)$，在其他不变的条件下，求出 $q_i^*(\boldsymbol{Z}_i)$。

这个迭代过程必收敛，因为 L(q)对各个因子 $q_i(\boldsymbol{Z}_i)$ 是 convex 的。

3. Variatioinal inference应用示例：Bayesian Gaussian Mixture Model（Bayesian GMM）

Bayesian GMM：

首先，GMM 是 $p(\boldsymbol{X}\mid\boldsymbol{Z},\boldsymbol{\mu},\Lambda)=\prod_{n=1}^{K}\prod_{k=1}^{K}N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_k,\Lambda_k^{-1})^{z_{nk}}$。在 Bayesian 中，各个参数都是 random variable，需要为它们引入 prior（都是 conjugate prior）：

$$p(\boldsymbol{Z}\mid\boldsymbol{\pi})=\prod_{n=1}^{N}\prod_{k=1}^{K}\pi_k^{z_{nk}}$$

$p(\boldsymbol{\pi})=Dir(\boldsymbol{\pi}\mid\boldsymbol{\alpha}_0)$，其中 $\boldsymbol{\alpha}_0=(\alpha_0,...,\alpha_0)$

$p(\boldsymbol{\mu},\Lambda)=p(\boldsymbol{\mu}\mid\Lambda)p(\Lambda)=\prod_{k=1}^{K}N(\boldsymbol{\mu}_k\mid\boldsymbol{m}_0,(\beta_0\Lambda_k)^{-1})W(\Lambda_k\mid\boldsymbol{W}_0,\nu_0)$，即每个 component 的参数的 conjugate prior 都是 Gaussian-Wishart 分布。

最后，根据这些 random variable 之间的关系，可以得到 PGM 和相应的 joint distribution 分解：$p(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)=p(\boldsymbol{X}\mid\boldsymbol{Z},\boldsymbol{\mu},\Lambda)p(\boldsymbol{Z}\mid\boldsymbol{\pi})p(\boldsymbol{\pi})p(\boldsymbol{\mu}\mid\Lambda)p(\Lambda)$

近似后验分布 $p(\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\mu},\Lambda\mid\boldsymbol{X})$：

考虑 variational distribution $q(\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$，并假设其分解性质：

$$q(\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)=q(\boldsymbol{Z})q(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$$

也就是在 parameter random variable 和狭义 latent variable 之间的分解性。

接下来的就是标准的 variational inference 过程了。

首先，针对 **q(Z)**优化 L(q)，得到：

$$\begin{aligned}\ln q^*(\boldsymbol{Z}) &= E_{\boldsymbol{\pi},\boldsymbol{\mu},\Lambda}[p(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)]+const\\ &= E_{\boldsymbol{\pi}}[p(\boldsymbol{Z}\mid\boldsymbol{\pi})]+E_{\boldsymbol{\mu},\Lambda}[p(\boldsymbol{X}\mid\boldsymbol{Z},\boldsymbol{\mu},\Lambda)]+const\end{aligned}$$

其中利用了 joint distribution $p(\boldsymbol{X},\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$ 的分解性质，以及把与 **Z** 无关的部分都合入 const。进一步解出来就是：

$$\ln q^*(\boldsymbol{Z})=\sum_{n=1}^{N}\sum_{k=1}^{K}z_{nk}\ln\rho_{nk}+const$$

其中，$\ln\rho_n = E_k[\ln\pi_k]+\frac{1}{2}E[\ln|\boldsymbol{\Lambda}_k|]-\frac{D}{2}\ln(2\pi)-\frac{1}{2}E_{\boldsymbol{\mu}_k,\Lambda_k}[(\boldsymbol{x}_n-\boldsymbol{\mu}_k)^T\Lambda_k(\boldsymbol{x}_n-\boldsymbol{\mu}_k)]$。可见，计算 $\ln q^*(\boldsymbol{Z})$ 将依赖于关于参数的期望。

然后，针对 $q(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$ 优化 L(q)，得到：

$$\ln q^*(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)=\ln p(\boldsymbol{\pi})+\sum_{k=1}^{K}\ln p(\boldsymbol{\mu}_k,\Lambda_k)+E_{\boldsymbol{Z}}[p(\boldsymbol{Z}\mid\boldsymbol{\pi})]+\sum_{n=1}^{N}\sum_{k=1}^{K}E[z_n]N(\boldsymbol{x}_n\mid\boldsymbol{\mu}_k,\Lambda_k^{-1})+const$$

可见，计算 $q^*(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$ 依赖于关于 Z 的期望。另外，上式还表明，$\ln q^*(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$ 是可以分解成关于 $\boldsymbol{\pi}$ 的一部分以及关于 $\boldsymbol{\mu},\Lambda$ 的一部分的；而且 $\boldsymbol{\mu},\Lambda$ 还可以关于 k 分解。这使我们得到 $q(\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)=q(\boldsymbol{Z})q(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)=q(\boldsymbol{Z})q(\boldsymbol{\pi})q(\boldsymbol{\mu},\Lambda)=q(\boldsymbol{Z})q(\boldsymbol{\pi})\prod_{k=1}^{K}q(\boldsymbol{\mu}_k,\Lambda_k)$。所以 variational inference 的针对 $q(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$ 最优化 L(q)可细分为针对 $q(\boldsymbol{\pi})$ 和 K 个 $q(\boldsymbol{\mu}_k,\Lambda_k)$。

以上两个过程不断迭代，直到收敛，就得到了最优化 L(q)的泛函解：$q^*(\boldsymbol{Z})$ 和 $q^*(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$。因此也就得到了后验概率的 $p(\boldsymbol{Z},\boldsymbol{\pi},\boldsymbol{\mu},\Lambda\mid\boldsymbol{X})$ 近似。

Bayesian GMM 自动选择最优的分支数 K：

在上面的求解过程中，值得关注各个分支的混合因子 mixing coefficient $\boldsymbol{\pi}$ 在后验分布下的期望值。近似的后验分布 $q^*(\boldsymbol{\pi})$ 是一个 Dirchelet 分布。经计算可以发现，$\boldsymbol{\pi}$ 的 K 个分量中，最后会有部分的期望 $E[\pi_k]$ 趋于 0，从而在 GMM 中不起作用。这样的话，我们可以实现自动的训练出最优的分支数：设置一个较大的 K，然后用上述 variational inference 得到近似后验分布，再把其中 $E[\pi_k]$ 接近 0 的分支去掉，剩下的即最终的 K^* 值。

Predictive distribution:

对 Bayesian 来说，得到后验概率（的近似）不是最终目的。剩下还要计算对给定一个新的 **x** 的 predictive distribution。

$$p(\boldsymbol{x}\mid\boldsymbol{X})=\sum_{\boldsymbol{z}}\iiint p(\boldsymbol{x}\mid\boldsymbol{z},\boldsymbol{\mu},\boldsymbol{\Lambda})\,p(\boldsymbol{z}\mid\boldsymbol{\pi})\,p(\boldsymbol{\pi},\boldsymbol{\mu},\boldsymbol{\Lambda}\mid\boldsymbol{X})d\boldsymbol{\pi}d\boldsymbol{\mu}d\boldsymbol{\Lambda}$$

把前面得到的近似后验分布 $q^*(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda)$ 代替上式的 $p(\boldsymbol{\pi},\boldsymbol{\mu},\Lambda\mid\boldsymbol{X})$，并把其他的已知（假设）分布带入，最后得到 $p(\boldsymbol{x}\mid\boldsymbol{X})$ 是 K 个 student t 分布的线性组合。

4. Expectation Propagation

方法的背景：

在 variational inference 中，近似后验概率的方法是最小化 $KL(q(\boldsymbol{Z})\|p(\boldsymbol{Z}\mid\boldsymbol{X}))$ with respect to $q(\boldsymbol{Z})$，其中 $p(\boldsymbol{Z}\mid\boldsymbol{X})$ 作为近似的目标是固定的，而第一个参数则是可变化的（该 KL 是 $q(\boldsymbol{Z})$ 的泛函）。在 expectation propagation 中，考虑的是 reversed form 的 KL，即：最小化 KL(p||q) with respect to q，而 p 固定。也就是关于 KL 的第二个参数进行优化。如果限定 q(**z**)在 exponential family（$q(\mathbf{z})=h(\mathbf{z})g(\boldsymbol{\eta})\exp\{\boldsymbol{\eta}^T\boldsymbol{u}(\mathbf{z})\}$）的范围内变化，那么就有 KL 的形式：$KL(p\|q)=-\ln g(\boldsymbol{\eta})-\boldsymbol{\eta}^T\mathrm{E}_{p(\mathbf{z})}[\boldsymbol{u}(\mathbf{z})]+const$，其中的 const 是与 natuaral parameter $\boldsymbol{\eta}$ 无关的量。为了最小化 KL，将其对 $\boldsymbol{\eta}$ 求导并置为 0，得到：$-\nabla\ln g(\boldsymbol{\eta})=\mathrm{E}_{p(\mathbf{z})}[\boldsymbol{u}(\mathbf{z})]$。然而根据 q(**z**)是 exponential family 的分布，所以其本身满足性质：$-\nabla\ln g(\boldsymbol{\eta})=\mathrm{E}_{q(\mathbf{z})}[\boldsymbol{u}(\mathbf{z})]$。因此得到：$\mathrm{E}_{p(\mathbf{z})}[\boldsymbol{u}(\mathbf{z})]=\mathrm{E}_{q(\mathbf{z})}[\boldsymbol{u}(\mathbf{z})]$。最终，确定 q(**z**)的方法就是，让 q(**z**)的各个成分统计量等于 p(**z**)的相应统计量。此过程即 **moment matching**。

问题：已知某个 probabilistic model 中，joint distribution 可以这样分解：$p(D,\boldsymbol{\theta})=\prod_i f_i(\boldsymbol{\theta})$。其中，D 是 observed data，$\boldsymbol{\theta}$ 是 latent variables（包括参数）。现在要计算后验分布 $p(\boldsymbol{\theta}\mid D)$ 和 model evidence p(D)。易见，$p(\boldsymbol{\theta}\mid D)=\frac{1}{p(D)}\prod_i f_i(\boldsymbol{\theta})$，$p(D)=\int\prod_i f_i(\boldsymbol{\theta})d\boldsymbol{\theta}$。

这个 model 的一个实例是独立同分布（i.i.d.）的情况。这时候 $f_i(\boldsymbol{\theta})=p(\boldsymbol{x}_i\mid\boldsymbol{\theta})$，以及一个额外的 $f_0(\boldsymbol{\theta})=p(\boldsymbol{\theta})$。

基本思路：

用分布 $q(\boldsymbol{\theta})=\frac{1}{Z}\prod_i\tilde{f}_i(\boldsymbol{\theta})$ 来近似 $p(\boldsymbol{\theta}\mid D)$。其中，每个 factor $\tilde{f}_i(\boldsymbol{\theta})$ 相应于 model 中的一个 $f_i(\boldsymbol{\theta})$。我们限定：$\tilde{f}_i(\boldsymbol{\theta})$ 为 exponential family 的分布。这样的话，显然 $q(\boldsymbol{\theta})$ 也是 exponential family 的分布（相乘后的还是此 family 的分布）。为此实现此近似，方法是最小

化 $KL(p(\boldsymbol{\theta}|D)\|q(\boldsymbol{\theta}))$ with respect $q(\boldsymbol{\theta})$。

详细思路：

对 $q(\boldsymbol{\theta})$，逐个考虑因子 $\tilde{f}_j(\boldsymbol{\theta})$，我们用 $f_j(\boldsymbol{\theta})$ 替换掉 $\tilde{f}_j(\boldsymbol{\theta})$，从而得到另一个分布：$\frac{1}{Z_j} f_j(\boldsymbol{\theta})q^{\setminus j}(\boldsymbol{\theta})$。其中，$q^{\setminus j}(\boldsymbol{\theta}) = \frac{q(\boldsymbol{\theta})}{\tilde{f}_j(\boldsymbol{\theta})}$，$Z_j = \int f_j(\boldsymbol{\theta})q^{\setminus j}(\boldsymbol{\theta})d\boldsymbol{\theta}$ 是归一化因子。现在要计算一个 revised $\tilde{f}_j(\boldsymbol{\theta})$，方法是最小化 $KL(\frac{1}{Z_j} f_j(\boldsymbol{\theta})q^{\setminus j}(\boldsymbol{\theta})\|q^{new}(\boldsymbol{\theta}))$ with respect to $q^{new}(\boldsymbol{\theta})$，并且限定 $q^{new}(\boldsymbol{\theta})$ 是 exponential family 的分布。这可以通过 moment matching 来得到。得到 $q^{new}(\boldsymbol{\theta})$ 后，就可以得到：$\tilde{f}_j^{new}(\boldsymbol{\theta}) = K\frac{q^{new}(\boldsymbol{\theta})}{q^{\setminus j}(\boldsymbol{\theta})}$。K 是为了确保 $q^{new}(\boldsymbol{\theta})$ 可以归一化的一个因子。可以证明 $K = Z_j$。

以上过程进行多轮，每一轮对每个 $\tilde{f}_j(\boldsymbol{\theta})$ 逐个考虑，从而不断迭代直至收敛，就得到了最后的 $q(\boldsymbol{\theta})$。

Chapter 11 Sampling Method

假设：已经一个生成器可以产生(0, 1)区间上的均匀分布随机数。

问题：从分布 p(**z**)中采样

1. 万能sampling method

假设有某分布的分布函数是 F(x)，若 y 是(0, 1)上的均匀分布随机变量，那么随机变量 $F^{-1}(y)$ 的分布函数是 F(x)。

这个方法的困难是：$F^{-1}(y)$ 不容易计算。

2. 基于proposal distribution的sampling

假设：从分布 p(**z**)采样很困难，但可以找到一个相对更容易采用的分布 q(**z**)，即 proposal distribution。

Rejection sampling

以 single-varaible 的情况考虑。假设分布 $p(z)=\frac{1}{Z_p}\tilde{p}(z)$，其中 Z_p 是 p(z)中与 z 无关的一个因子。

之所以要分离开来写，是因为有时候 p(z)中与 z 无关的部分可能比较复杂，难以计算。例如，$Gam(z\mid a,b)=\frac{b^a z^{a-1}\exp(-bz)}{\Gamma(a)}$，其中，$\frac{b^a}{\Gamma(a)}$ 与 z 无关，可以作为 Z_p 提出来。更重要的可能是，有时候我们对分布 p(z)的认识集中在 p(z)的与 z 相关的部分；其 normalization 的 Z_p 还处于一个难以计算的未知状态。因此，在 sampling 中，如果可以发明只利用与 z 有关部分即可进行采样的算法，就避开了对复杂的 Z_p 的计算。

Rejection sampling 要为 p(z)找一个 proposal distribution q(z)，而且 q(z)必须是很容易进行 sampling 的。然后找到一个尽可能小的常数 k，使得 $kq(z)\geq\tilde{p}(z)$ 对任意 z 成立。

Rejection sampling 的过程是：首先从 q(z)中 sample 出一个数 z_0；然后从均匀分布 $[0,kq(z_0)]$ 中 sample 出另一个数 u_0；这时候，平面上的点（z_0，u_0）是 kq(z)下方区域中的均匀分布。如果 $u_0>\tilde{p}(z_0)$，则拒绝点 z_0 并重复前面步骤，否则接收 z_0 为符合分布 p(z)的点。

上述过程看出了 k 为什么要尽可能小。K 越小，才能使 z_0 被拒绝的概率尽可能小，从而提高 rejection sampling 的效率。

Rejection sampling 的缺点是：维数越高，拒绝率越高，采样效率越低。例如高维的球，可计算其测度主要集中在球的表面；而 rejection sampling 中，$u_0 > \tilde{p}(z_0)$ 的部分正是高维几何体的表层。这就是导致很高的拒绝率。

Importance sampling

假设：对 p(**z**)采样是困难的，不过对于一个给定的 **z**，却可容易的计算其概率值 p(**z**)。

假设现在不是要对分布 p(**z**)采样，而是要计算一个函数 f(**z**)在该分布下的期望。采样现在是作为近似计算该期望的方法，比如有 L 个 p(**z**)的采样，那么：$E[f] \simeq \frac{1}{L}\sum_{l=1}^{L} f(\boldsymbol{z}^{(l)})$。但是 p(**z**)本身采样困难，所以我们还是得像 Rejection sampling 那样，找到一个更容易采样的分布 q(**z**)，并且假设从 q(**z**)采样了 L 个样本。那么：

$$E[f] = \int p(\boldsymbol{z}) f(\boldsymbol{z}) d\boldsymbol{z} = \int \frac{p(\boldsymbol{z}) f(\boldsymbol{z})}{q(\boldsymbol{z})} q(\boldsymbol{z}) d\boldsymbol{z} \approx \frac{1}{L}\sum_{l=1}^{L} \frac{p(\boldsymbol{z}^{(l)})}{q(\boldsymbol{z}^{(l)})} f(\boldsymbol{z}^{(l)})$$

其中 $r_l = \frac{p(\boldsymbol{z}^{(l)})}{q(\boldsymbol{z}^{(l)})}$ 被成为 importance weights。

再进一步假设：对分布 p(**z**)的认识集中在 p(**z**)的与 **z** 相关的部分 $\tilde{p}(\boldsymbol{z})$，其 normalization constant Z_p 还未知。同时也从 q(**z**)中分离出一个常数 Z_q，那么：

$$E[f] = \int p(\boldsymbol{z}) f(\boldsymbol{z}) d\boldsymbol{z} = \frac{Z_q}{Z_p} \int \frac{\tilde{p}(\boldsymbol{z}) f(\boldsymbol{z})}{\tilde{q}(\boldsymbol{z})} q(\boldsymbol{z}) d\boldsymbol{z} \approx \frac{Z_q}{Z_p} \frac{1}{L} \sum_{l=1}^{L} \tilde{r}_l f(\boldsymbol{z}^{(l)})$$

其中，$\tilde{r} = \frac{\tilde{p}(\boldsymbol{z}^{(l)})}{\tilde{q}(\boldsymbol{z}^{(l)})}$。

同样地，可以计算：

$$\frac{Z_q}{Z_p} = \frac{1}{Z_p}\int \tilde{p}(\boldsymbol{x}) d\boldsymbol{x} = \frac{q(\boldsymbol{z})}{\tilde{q}(\boldsymbol{z})} \int \tilde{p}(\boldsymbol{x}) d\boldsymbol{x} = \int \frac{q(\boldsymbol{z})}{\tilde{q}(\boldsymbol{z})} \tilde{p}(\boldsymbol{x}) d\boldsymbol{x} = \int \frac{q(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})} \tilde{p}(\boldsymbol{x}) d\boldsymbol{x}$$（**z** 变量换名为 **x**，因为 $\frac{q(\boldsymbol{z})}{\tilde{q}(\boldsymbol{z})} = \frac{q(\boldsymbol{x})}{\tilde{q}(\boldsymbol{x})}$ 是常数）

$$= \frac{1}{L}\sum_{l=1}^{L} \tilde{r}_l$$

于是最终得到：

$$E[f] = \frac{1}{L}\sum_{l=1}^{L} w_l f(\boldsymbol{z}^{(l)})$$

其中，$\boldsymbol{z}^{(l)}$ 是分布从 q(**z**)采样的 L 个样本，而

$$w_l = \frac{\tilde{r}_l}{\sum_{m=1}^{L} \tilde{r}_m} = \frac{\tilde{p}(\boldsymbol{z}^{(l)}) / q(\boldsymbol{z}^{(l)})}{\sum_m \tilde{p}(\boldsymbol{z}^{(m)}) / q(\boldsymbol{z}^{(m)})}$$

注意，在 w_l 的计算中，已经只需要分布 p(**z**)的与 **z** 有关部分 $\tilde{p}(\boldsymbol{z})$。从而达到了目的。

Sampling-importance-resampling (SIR)

Rejection sampling 要找一个常数 k，Importance sampling 不能得到 p(**z**)的样本而只有期望，Sampling-importance-resampling 克服了这两个问题。SIR 同样基于一个易于采用的 proposal distribution q(**z**)，其做法分两步：

第一步，从 q(**z**)中采样 L 个样本 $\boldsymbol{z}^{(l)}$，并且计算 Importance sampling 中所定义的 w_l；

第二步，从集合 $\{\boldsymbol{z}^{(l)} : l = 1, ..., L\}$ 中有放回地采样 L 个样本，其中 $\boldsymbol{z}^{(l)}$ 被抽中的概率是 w_l。

可以证明，这样得出来的 L 个样本近似目标分布 p(**z**)；当 L 趋于无穷时，将完全服从目标分布。

MCMC

见下面。

3. Monte Carlo EM algorithm

在 EM 算法中，E 步是计算 complete-log likelihood 在 latent variables 的后验概率下的期望，也就是：$Q(\boldsymbol{\theta}, \boldsymbol{\theta}^{old}) = \sum_{\boldsymbol{Z}} p(\boldsymbol{Z} \mid \boldsymbol{X}, \boldsymbol{\theta}^{old}) \ln p(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta})$。现在用 sampling 的方法来近似这个求和（或积分）：从当前对 latent variables 的后验概率的估计分布 $p(Z \mid X, \theta^{old})$ 中采样出 L 个样本 $\{\boldsymbol{Z}^{(l)} : l = 1, ..., L\}$，于是得到：$Q(\theta, \theta^{old}) \approx \frac{1}{L} \sum_l \ln p(Z^{(l)}, X \mid \theta)$。

在 M 步中，还是跟 EM 算法一样，最大化 $Q(\theta, \theta^{old})$ with respect to θ。

4. Markov chain

（一阶）Markov chain 是指 a series of random variables $\{\boldsymbol{z}^{(l)} : l = 1, ..., M\}$，它们满足如下条件独立性：$p(\boldsymbol{z}^{(m+1)} \mid \boldsymbol{z}^{(1)}, ..., \boldsymbol{z}^{(m)}) = p(\boldsymbol{z}^{(m+1)} \mid \boldsymbol{z}^{(m)})$ 对任意 m 成立。

Transiton probablisty：$T_m(\boldsymbol{z}^{(m)}, \boldsymbol{z}^{(m+1)}) = p(\boldsymbol{z}^{(m+1)} \mid \boldsymbol{z}^{(m)})$。

Homogeneous：对于任意 m，Transiton probablisty T_m 相同。现在只考虑 homogeneous Markov chain。

Invariant distribution： $p^*(\boldsymbol{z}) = \sum_{\boldsymbol{z}'} T(\boldsymbol{z}', \boldsymbol{z}) p^*(\boldsymbol{z}')$。Markov chain 的 invariant distribution 可能不止一个。（T 的下标去除，以示该 Markov chain 是 homogeneous 的）

Detailed balance：分布 p(**z**)是 Markov chain 的一个 invariant distribution，如果其满足：$p(\boldsymbol{z})T(\boldsymbol{z}, \boldsymbol{z}') = p(\boldsymbol{z}')T(\boldsymbol{z}', \boldsymbol{z})$。这是一个充分不必要条件。

5. Metropolis-Hastings算法

与 Rejection sampling 和 Importance sampling 一样，需要为目标分布 p(**z**)找一个易于采用的 proposal sampling q(**z**)。该算法的采样过程是：假设当前已经 sampling 出的样本是 $\boldsymbol{z}^{(\tau)}$，那么下一个样本从分布 $q(\boldsymbol{z} \mid \boldsymbol{z}^{(\tau)})$ 中取得；记新采出的样本是 $\boldsymbol{z}^*$，我们以概率 $A(\boldsymbol{z}^*, \boldsymbol{z}^{(\tau)})$ 接受该样本，即：

$$\boldsymbol{z}^{(\tau+1)} = \begin{cases} \boldsymbol{z}^{(*)}, if\ accpet \\ \boldsymbol{z}^{(\tau)}, if\ reject \end{cases}$$

其中，$A(\boldsymbol{z}^*, \boldsymbol{z}^{(\tau)}) = \min\{1, \frac{\tilde{p}(\boldsymbol{z}^*) q(\boldsymbol{z}^{(\tau)} \mid \boldsymbol{z}^*)}{\tilde{p}(\boldsymbol{z}^{(\tau)}) q(\boldsymbol{z}^* \mid \boldsymbol{z}^{(\tau)})}\}$，而 $\tilde{p}(\boldsymbol{z})$ 是 p(**z**)只与 z 相关的部分。

以上算法其实定义了一个一阶 Markov chain $\{\boldsymbol{z}^{(\tau)} : \tau = 1, ...\}$。需要证明的是，当 $\tau \to \infty$ 时，$\boldsymbol{z}^{(\tau)}$ 的分布趋于目标分布 p(**z**)。可以验证 p(**z**)满足 detailed balance 条件，因此是该 Markov chain 的 invariant distribution。

对于 continuous state space，一般用 Gaussian centred on the current state 作为 proposal distribution。而该分布的 variance 就是一个选择的难点了：variance 太小，则遍历 state space 慢，效率低；variance 太大，则导致拒绝率高，同样也效率低。

6. Gibbs sampling

Metropolis-Hastings 算法的一个特例。假设要从分布 $p(\boldsymbol{z}) = p(z_1, ..., z_M)$ 中采样。Gibbs sampling 的做法是：

首先，给每个 $\{z_i : i = 1...M\}$ 一个初始值，作为第 $\tau = 1$ 次 sampling 的结果；

然后，考虑从第 τ 到第 $\tau + 1$ 次的 sampling：

- sampling $z_1^{(\tau+1)} \sim p(z_1 \mid z_2^{(\tau)}, z_3^{(\tau)}, ..., z_M^{(\tau)})$
- sampling $z_2^{(\tau+1)} \sim p(z_2 \mid z_1^{(\tau+1)}, z_3^{(\tau)}, ..., z_M^{(\tau)})$
- …
- sampling $z_j^{(\tau+1)} \sim p(z_j \mid z_1^{(\tau+1)}, ..., z_{j-1}^{(\tau+1)}, z_{j+1}^{(\tau)} ..., z_M^{(\tau)})$
- sampling $z_M^{(\tau+1)} \sim p(z_M \mid z_1^{(\tau+1)}, z_2^{(\tau+1)}, ..., z_{M-1}^{(\tau+1)})$

现在从 Metropolis-Hastings 算法的角度认识 Gibbs sampling。考虑对 z_k 的更新采样，在得到 $z_k^{(\tau+1)}$ 后的 $\mathbf{z}$ 记为 $\boldsymbol{z}^*$，那么有 $q(\boldsymbol{z}^* \mid \boldsymbol{z}) = p(z_k \mid \boldsymbol{z}_{\setminus k})$，其中 $\boldsymbol{z}_{\setminus k}$ 表示集合 $\{z_i : i = 1...M\}$ 去掉 z_k。而且在 $\mathbf{z}$ 和 $\boldsymbol{z}^*$ 之间，只有 z_k 是不同的，因此 $\boldsymbol{z}_{\setminus k} = \boldsymbol{z}^*_{\setminus k}$。于是在 Metropolis-Hastings 算法中的接受概率为：$A(\boldsymbol{z}^*, \boldsymbol{z}^{(\tau)}) = \dfrac{p(\boldsymbol{z}^*)q(\boldsymbol{z} \mid \boldsymbol{z}^*)}{p(\boldsymbol{z})q(\boldsymbol{z}^* \mid \boldsymbol{z})} = \dfrac{p(z_k^* \mid \boldsymbol{z}^*_{\setminus k})p(\boldsymbol{z}^*_{\setminus k})p(z_k \mid \boldsymbol{z}^*_{\setminus k})}{p(z_k \mid \boldsymbol{z}_{\setminus k})p(\boldsymbol{z}_{\setminus k})p(z_k^* \mid \boldsymbol{z}_{\setminus k})} = 1$。也就是说，所有 Metropolis-Hastings 的状态都会接受。这正是 Gibbs sampling 所呈现的。

Chapter 12 Continuous Latent Variables

1. Principal Component Analysis (PCA)

一种 unsupervised learning，应用于 dimentionality reduction，feature extraction，data visualization 以及 lossy data compression 等。

问题之一：Maximum Variance Subspace

假设有 N 个 D 维的 observation data $\{\boldsymbol{x}_n : n = 1...N\}$，现在要找到一个 M<D 维的空间，使得 N 个 data 往该空间投影后，variance of the projected data is maximize。

在 M>1 维的空间中，variance 无定义。所以可以逐个维度考虑：先找到一个维度 $\boldsymbol{u}_1$，使得在维度 $\boldsymbol{u}_1$ 上的 projected data 具有最大的方差；第二次再找一个与 $\boldsymbol{u}_1$ 正交的维度 $\boldsymbol{u}_2$，使 variance of the projected data 最大。这个过程不断进行，直到找到 M 个正交维度，从而构成一个 M 维空间。

Formulation：首先考虑寻找 $\boldsymbol{u}_1$， N 个 observation data 在该向量方向上的 variance of projected data 是：

$$\frac{1}{N}\sum_{n=1}^{N}\{\boldsymbol{u}_1^T\boldsymbol{x}_n - \boldsymbol{u}_1^T\bar{\boldsymbol{x}}\}^2 = \boldsymbol{u}_1^T S\boldsymbol{u}_1$$

其中，$\bar{\boldsymbol{x}} = \frac{1}{N}\sum_n \boldsymbol{x}_n$，S 是 sample covariance matrix：$\boldsymbol{S} = \frac{1}{N}\sum_n (\boldsymbol{x}_n - \bar{\boldsymbol{x}})(\boldsymbol{x}_n - \bar{\boldsymbol{x}})^T$。现在问题就是：最大化 $\boldsymbol{u}_1^T S\boldsymbol{u}_1$ with respect to $\boldsymbol{u}_1$。为了避免 $||\boldsymbol{u}_1|| \to \infty$，需要对其做限定。合理的限定是让其为单位向量，即 $\boldsymbol{u}_1^T\boldsymbol{u}_1 = 1$。

用 Lagrange multiplier 计算，可以得到：$S\boldsymbol{u}_1 = \lambda_1\boldsymbol{u}_1$。可见 λ_1 是 S 的特征向量。进一步可以得出：$\boldsymbol{u}_1^T S\boldsymbol{u}_1 = \lambda_1$。由于 $\boldsymbol{u}_1^T S\boldsymbol{u}_1$ 是目标函数，因此 λ_1 取 S 的最大特征值可使目标最大化。此时的 $\boldsymbol{u}_1$ 是对应最大特征值 λ_1 的单位特征向量。

用于的推导可发现，$\boldsymbol{u}_2$ 取为 S 的第二大特征值对应的单位特征向量。从而得到了 M 最大特征值对应的特征向量。

问题之二：Minimum Projection Error

与上个问题一样，不过此处最小化的是 $\boldsymbol{x}_n$ 与其 projection $\tilde{\boldsymbol{x}}_n$ 之间的差异。

Formulation：跟 Maximum Variance Subspace 一样，逐个考虑。比如 $\boldsymbol{u}_1$，则是最小化 $\frac{1}{N}\sum_n ||\boldsymbol{x}_n - (\boldsymbol{u}_1^T\boldsymbol{x}_n)\boldsymbol{u}_1||^2$，满足约束 $\boldsymbol{u}_1^T\boldsymbol{u}_1 = 1$。推导以后发现结果和 Maximum Variance

Subspace 是相同的。

2. Probabilistic PCA (PPCA)

用 latent variable **z** 对应 principle-component subspace；observed data **x** 对应于原来的 data point。并且假设：

$$p(\boldsymbol{z}) = N(\boldsymbol{z} \mid \boldsymbol{0}, \boldsymbol{I})$$

$$p(\boldsymbol{x} \mid \boldsymbol{z}) = N(\boldsymbol{x} \mid W\boldsymbol{z} + \boldsymbol{\mu}, \sigma^2 \boldsymbol{I})$$

W 是一个从 principle-component subspace 到原空间的线性变换。

这是一个 linear-Gaussian 模型，所以可以相应得到：

$$p(\boldsymbol{x}) = N(\boldsymbol{x} \mid \boldsymbol{\mu}, \boldsymbol{C})$$

$$p(\boldsymbol{z} \mid \boldsymbol{x}) = N(\boldsymbol{z} \mid \boldsymbol{M}^{-1}\boldsymbol{W}^T(\boldsymbol{x} - \boldsymbol{\mu}), \sigma^2 \boldsymbol{M})$$

其中，$\boldsymbol{C} = \boldsymbol{W}\boldsymbol{W}^T + \sigma^2 \boldsymbol{I}_D$，$\boldsymbol{M} = \boldsymbol{W}^T\boldsymbol{W} + \sigma^2 \boldsymbol{I}_M$。

关于 $p(\boldsymbol{x})$，有一个特点值得注意：任意 latent space 中的 **z**，对其进行 rotation 后，得到 **Rz**，其中 **R** 是一个 orthogonal matrix；那么得到的 $p(\boldsymbol{x})$ 是不变的。因为，设 $\tilde{\boldsymbol{W}} = \boldsymbol{W}\boldsymbol{R}$，那么利用 **R** 的正交性，$\tilde{\boldsymbol{W}}\tilde{\boldsymbol{W}}^T = \boldsymbol{W}\boldsymbol{R}\boldsymbol{R}^T\boldsymbol{W}^T = \boldsymbol{W}\boldsymbol{W}^T$。

在得到一个 observed data set 后，可以对 PPCA 中的参数 $\boldsymbol{W}, \boldsymbol{\mu}, \sigma^2$ 进行估计。估计的方法可以是直接 MLE（有 closed-form solution），但为了避免矩阵运算，提高计算效率也可以 EM。

另外，还可以为 $\boldsymbol{W}, \boldsymbol{\mu}, \sigma^2$ 引入 Prior，从而得到 Bayesian PPCA。这时候没有 closed-form solution，只能 EM 了。

3. Factor analysis

与 PPCA 相似：

$$p(\boldsymbol{z}) = N(\boldsymbol{z} \mid \boldsymbol{0}, \boldsymbol{I})$$

$$p(\boldsymbol{x} \mid \boldsymbol{z}) = N(\boldsymbol{x} \mid W\boldsymbol{z} + \boldsymbol{\mu}, \Psi)$$

其中，Ψ 是一个 diagonal matrix。这是 Factor analysis 与 PPCA 的唯一不同之处（PPCA 是一个 isotropic matrix）。

4. Kernel PCA

在 PCA 中，从原空间到 principle-component subspace 的 projection 是线性的。而 Kernel PCA 要做的是一个非线性的 projection。

考虑一个非线性的 feature map $\boldsymbol{\phi}(\boldsymbol{x})$，它将原空间的每一个 data point 映射到高维（设

为M维)feature space中。然后在feature space进行PCA，这对应于原空间的非线性projection。

假设feature space中的data point的均值为0，即 $\sum_n \boldsymbol{\phi}(\boldsymbol{x}_n) = 0$。那么在这个M维空间中的sample covariance是：$C = \frac{1}{N}\sum_n \boldsymbol{\phi}(\boldsymbol{x}_n)\boldsymbol{\phi}(\boldsymbol{x}_n)^T$。按照PCA的套路，应该求得C的特征值及特征向量，$Cv_i = \lambda_i v_i$，i=1,…,M。跟其他kernel方法一样，我们不能直接在这个M维空间中解此eigenvector equation，而应该寻找kernel trick。

经过一些替换，可以得到：$\boldsymbol{K}\boldsymbol{a}_i = \lambda_i N\boldsymbol{a}_i$。其中，**K** 由 $\kappa(\boldsymbol{x}_n, \boldsymbol{x}_m) = \boldsymbol{\phi}(\boldsymbol{x}_n)^T\boldsymbol{\phi}(\boldsymbol{x}_m)$ 构成的N-by-N矩阵。这样我们就可以通过求解一个N为方阵的eigenvector equation，而避免了直接在feature space上求解。

用以上方法得到 **K** 的特征向量后，就可以在feature space中对任意data point **x** 进行projection了；其向第i个特征向量 v_i 的投影是：

$$y_i(\boldsymbol{x}) = \boldsymbol{\phi}(\boldsymbol{x})^T\boldsymbol{v}_i = \sum_{n=1}^{N} a_{in}k(\boldsymbol{x}, \boldsymbol{x}_n)$$

上面的过程假设了 $\sum_n \boldsymbol{\phi}(\boldsymbol{x}_n) = 0$。对于一般的情况，可以对data进行centralize，即 $\tilde{\boldsymbol{\phi}}(\boldsymbol{x}_n) = \boldsymbol{\phi}(\boldsymbol{x}_n) - \frac{1}{N}\sum_n \boldsymbol{\phi}(\boldsymbol{x}_n)$。计算出 $\tilde{K}_{nm} = \tilde{\boldsymbol{\phi}}(\boldsymbol{x}_n)^T\tilde{\boldsymbol{\phi}}(\boldsymbol{x}_m)$ 与 $K_{nm} = \boldsymbol{\phi}(\boldsymbol{x}_n)^T\boldsymbol{\phi}(\boldsymbol{x}_m)$ 之间的关系。用后者把前者表示计算出来，剩下的就是针对 $\tilde{K}_{nm}$ 解eigenvector equation了。

4. Nonlinear latent variable models

PPCA的latent variable尽管是continuous（相较GMM和HMM的latent variable是discrete）的，但observed variable对其的依赖仍旧是线性的，即linear-Gaussian；而且latent variable本身的分布是Gaussian。

Independent component analysis

假设：Observed variable对latent variable是线性依赖关系（线性组合），但是latent variable不再是Gaussian，而是满足性质 $p(\boldsymbol{z}) = \prod_{j=1}^{M} p(z_j)$ 的任意分布。

Autoassociative neural network

也叫auto-encoder，是neural network用于unsupervised learning的一种方法。这种NN的网络结构是：D个inputs，D个outputs，M (< D)个hidden units；构成3层网络。优化的目

标函数是：$E(\boldsymbol{w})=\frac{1}{2}\sum_n \| \boldsymbol{y}(\boldsymbol{x}_n,\boldsymbol{w})-\boldsymbol{x}_n \|^2$ 。也就是使得输入和输出尽可能的接近。

如果 hidden unit 采用 linear activation function，那么 $E(\boldsymbol{w})$ 有全局唯一的最优解。这个解相应于将这 N 个 D 维的数据投影到一个 M 维的子空间。在 neural network 上，每个 hidden unit 的 D 条入边的权重构成一个 D 维向量，为 principle component。不过这些向量不一定是正交和归一化的。

如果 hidden unit 采用 nonlinear activation function，最小化目标函数的解仍旧是对应于将数据映射到 M 维 principle-component space。

要进行 nonlinear principle component analysis，只能增加 hidden layer。

Chapter 13 Sequential Data

1. Hidden Markov Model (HMM)

HMM 的定义

对于每个 observation $\boldsymbol{x}_n$，都有一个相应的 discrete latant variable $\boldsymbol{z}_n$，二者的维度尽可不同；而 latent variable 形成一个 Markov chain。因此 HMM 可以表示成一个 Bayesian Network。

考虑 latent Markov chain 是 homogeneous，也就是说，transition probability $p(\boldsymbol{z}_n \mid \boldsymbol{z}_{n-1})$ 对任意的 n 是一样的。由于 $\boldsymbol{z}_n$ 是离散的，假设其可取 K 个不同的值（K 个状态），那么可用 1-of-K 的编码方法表示 $\boldsymbol{z}_n$。并且 **transition probability** $p(\boldsymbol{z}_n \mid \boldsymbol{z}_{n-1})$ 可以表示成一个 K-by-K 的矩阵 A（A 的每个 entry 是一个概率值，并且每一行都是归一的）。

现在给出 HMM 的描述：

$$p(\boldsymbol{z}_n \mid \boldsymbol{z}_{n-1}, A) = \prod_{k=1}^{K}\prod_{j=1}^{K} A_{jk}^{z_{n-1,j}z_{zk}}$$

$p(\boldsymbol{z}_1) = \prod_{k=1}^{K} \pi_k^{z_{1k}}$，其中 $\sum_k \pi_k = 1,\ 0 \le \pi_k \le 1$

$p(\boldsymbol{x}_n \mid \boldsymbol{z}_n, \phi) = \prod_{k=1}^{K} p(\boldsymbol{x}_n \mid \phi_k)^{z_{nk}}$ （**emission probability**，可离散可连续）

HMM 的 likelihood function

与之前很多模型不同，HMM 中获得的 observation $\{\boldsymbol{x}_n : n = 1..N\}$ 并不是独立同分布的。因此 likelihood function 不能够写成 the product over all data points of the probability evaluated at each data point。HMM 的 likelihood 可按照 Bayesian Network 的分解写出：

$$p(\boldsymbol{X}, \boldsymbol{Z} \mid \boldsymbol{\theta}) = p(\boldsymbol{z}_1 \mid \boldsymbol{\pi})[\prod_{n=2}^{N} p(\boldsymbol{z}_n \mid \boldsymbol{z}_{n-1}, A)][\prod_{m=1}^{N} p(\boldsymbol{x}_m \mid \boldsymbol{z}_m, \phi)]$$

其中，$\boldsymbol{X} = \{\boldsymbol{x}_n : n = 1..N\}, \boldsymbol{Z} = \{\boldsymbol{z}_n : n = 1..N\}, \boldsymbol{\theta} = \{\boldsymbol{\pi}, A, \phi\}$。可见 HMM 的全部参数是 $\boldsymbol{\theta} = \{\boldsymbol{\pi}, A, \phi\}$。

HMM 的 learning：Maximum Likelihood EM

所谓 learning，就是 根据 observation $\boldsymbol{X}=\{\boldsymbol{x}_n : n=1..N\}$ 来学习 HMM 的参数 $\boldsymbol{\theta}=\{\boldsymbol{\pi}, A, \boldsymbol{\phi}\}$。采用 MLE 进行参数估计。由于含有 latent variable，所以只能求得 observation 的 likelihood，即： $p(\boldsymbol{X}\mid\boldsymbol{\theta})=\sum_{\boldsymbol{Z}} p(\boldsymbol{X},\boldsymbol{Z}\mid\boldsymbol{\theta})$。用 EM 算法求解使该 likelihood 最大化的参数 $\boldsymbol{\theta}$。

在 EM 算法迭代中，将会涉及到一个 HMM 的 inference 问题，就是计算 local posterior marginals for latent variables。包括两个：

$\gamma(\boldsymbol{z}_n)=p(\boldsymbol{z}_n\mid\boldsymbol{X},\boldsymbol{\theta}^{old})$

$\xi(\boldsymbol{z}_{n-1},\boldsymbol{z}_n)=p(\boldsymbol{z}_{n-1},\boldsymbol{z}_n\mid\boldsymbol{X},\boldsymbol{\theta}^{old})$

Forward-backward 算法可计算这两个 marginal。由于 HMM 的 factor graph 是一颗树，所以也可以用 PGM 通用的 sum-product 算法。

HMM 的 inference：Viterbi 算法

所谓 inference，就是计算 marginal 或者 conditional (表现为后验概率)。HMM 中最重要的 inference 就是计算： $\boldsymbol{Z}^{*}=\arg\max_{\boldsymbol{Z}} p(\boldsymbol{Z}\mid\boldsymbol{X},\boldsymbol{\theta})$。其含义是，对于给定的 HMM（也就是给定 $\boldsymbol{\theta}$），在观察到 $\boldsymbol{X}=\{\boldsymbol{x}_n : n=1..N\}$ 的情况下，最可能的 latent states（sequence of states）是什么。求解的方法是在 PGM 通用的 max-sum 算法，在 HMM 中则对应为 Viterbi 算法。

2. Linear Dynamical System (LDS)

LDS 具有和 HMM 完全一个的 PGM，其所不同的是：LDS 的 latent variables 是连续的；并且其 transition probabilisty 与 emission probabilisty 都是 linear-Gaussion。所以 LDS 的描述是：

$$p(\boldsymbol{z}_n\mid\boldsymbol{z}_{n-1})=N(\boldsymbol{z}_n\mid\boldsymbol{A}\boldsymbol{z}_{n-1},\boldsymbol{\Gamma})$$

$$p(\boldsymbol{x}_n\mid\boldsymbol{z}_n)=N(\boldsymbol{x}_n\mid\boldsymbol{C}\boldsymbol{z}_n,\boldsymbol{\Sigma})$$

$$p(\boldsymbol{z}_1)=N(\boldsymbol{z}_1\mid\boldsymbol{\mu}_0,\boldsymbol{V}_0)$$

Kernelize

Probabilistize

Chapter 14 Combining Models

1. Model combination的两种方式

model averaging 和 model selection。

Model averaging 的例子是：Boost，mixture of linear regressioin，mixture of logtistic regression 以及之前的 GMM。

Model selection 的例子是：Decision tree。

2. Bayesian model averaging V.S. Combined model

二者的区别是：对于观察到的 data set D，Bayesian model averaging 认为它来自 a single model，尽管当前还无法确定到底是哪一个；而 Combined model 中，D 的数据可能来自不同的 model，并由它们混合而成。后者的例子是 GMM，对于一个 D，它里面的数据可能是来自 K 个不同的 Gaussian model。

3. Decision tree

Decision tree 包括：CART (Classification And Regression Tree)及其变化体 ID3 和 C4.5。

作为 combined model 的 decision tree

PRML 把 decision tree 视为一种 combined model：每片树叶就是一个 model，负责 input space 中某一个 region 的 data point；对于给定一个 data point 的 regression/classification，从 decision tree 的 root 到 leaf 的决策过程，就是一个 model selection 的过程，即决定这个 data point 将由哪一个 model（leaf）来处理。

Decision tree 的目标函数

假设有 D 维的数据集 $\boldsymbol{X}=\{\boldsymbol{x}_n : n=1..N\}$，以及它们相应的 labels $\boldsymbol{t}=\{t_n : n=1..N\}$（对 regression 是连续值，对 classification 是离散值）。

假设最后训练所得的树中 leaf 的集合是 T。第 τ 个 leaf 代表了 input space 中的 region R_τ，并设其中含有个 N_τ data point。

对于 regression，区域 R_τ（model R_τ）的 optimal prediction 是：

$$y_\tau = \frac{1}{N_\tau}\sum_{x_n \in R_\tau} t_n$$

因此该区域中的 sum-of-squares error 是：

$$Q_\tau(T) = \sum_{x_n \in R_\tau} \{t_n - y_\tau\}^2$$

对于 classification，设 $p_{\tau k}$ 是 region R_τ 中属于类别 k 的样本数。有两个常用的函数来度量训练的目标：

$$Q_\tau(T) = \sum_k p_{\tau k} \ln p_{\tau k}$$ ——cross entropy

$$Q_\tau(T) = \sum_k p_{\tau k}(1 - p_{\tau k})$$ ——Gini index

为了避免 over-fitting（比如每个 leaf 只有一个 data point，那么 training error 就是 0 了，但这样显然不合理），需要对 decision tree 进行控制 model complexity。一种方法是控制 Leaf 的数目。这时候得到 regression/classification 的目标函数：

$$C(T) = \sum_{\tau=1}^{|T|} Q_\tau(T) + \lambda |T|$$

Decision tree 的训练过程就是最小化 C(T)的过程。

Decision tree 的训练过程

初始时整个 tree 就一个 root。然后反复进行下述步骤，直到满足一定的停止准则：

选择当前 tree 的一个 leaf；

选择 D 个 feature（D-dimension data point 就是每个 data point 有 D 个 feature）中的一个 feature；

为选出来的 feature 选择一个 threshold；

根据该 threshold，从当前 leaf 分裂出两个子节点，leaf 中所有其 feature 值大于 threshold 的 data point 分入一个子节点，剩下的 data point 分入另一个子节点；

从而得到一颗新的树，新树的 leaf 比原树多一个；

Deision tree 的优缺点

优点：human interpretability 好

缺点：tree struture 对 data set 很敏感，对 training data 稍许的改变，可能就导致训练出一个 tree struture 很不一样的树。

4. 不同类型的Mixture Model

Unconditional Mixture Model：例如，经典的混合模型 Gaussian Mixture Model (GMM)，有 $p(\boldsymbol{x}) = \sum_{k=1}^{K} \pi_k N(\boldsymbol{x} \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k)$。这个分布仅仅由参数 $\boldsymbol{\pi}, \boldsymbol{\mu}, \boldsymbol{\Sigma}$ 控制。

Conditional Mixture Model：例如，Mixtures of linear regression models，即将多个 linear regression model 混合成一个模型：$p(t|\boldsymbol{\theta})=\sum_{k=1}^{K}\pi_k N(t|\boldsymbol{w}_k^T\boldsymbol{\phi},\beta^{-1})$，其中$\boldsymbol{\theta}=\{\boldsymbol{W},\boldsymbol{\pi},\beta\}$。这个分布不仅依赖于参数$\boldsymbol{\theta}=\{\boldsymbol{W},\boldsymbol{\pi},\beta\}$，还依赖于 input $\boldsymbol{\phi}$，所以是 conditional mixture model。

Mixture of Expert：在 Mixtures of linear regression models 中，mixing coefficients $\boldsymbol{\pi}$ 是不依赖于 input 的；如果把对此条件放松，使$\boldsymbol{\pi}$ 也依赖于 input，那么就得到了 mixture of expert 模型 $p(t|\boldsymbol{x})=\sum_{k=1}^{K}\pi_k(\boldsymbol{x})p_k(t|\boldsymbol{x})$。其中一个分支 $p_k(t|\boldsymbol{x})$ 就是一个 expert。

Hierarchical Mixture of Expert：让 mixture of expert 的每一个分支 $p_k(t|\boldsymbol{x})$ 本身为一个 mixture of expert，这样就有了层次结构。

以上连最简单的 Unconditional Mixture Model 例如 GMM 都没有 closed-form solution，因此都需要 EM 算法来求解。

5. AdaBoost

属于 model combination 中的 model averaging。假设训练出 M 个 model $\{y_m(\boldsymbol{x}):m=1..M\}$，那么最后的 model 由这 M 个 Model 加权组合得到$Y(\boldsymbol{x})=\text{sgn}(\sum_{m=1}^{M}\alpha_m y_m(\boldsymbol{x}))$，其中 α_m 是模型 $y_m(\boldsymbol{x})$ 的权值，将在训练中给出。这里考虑的是 binary classification，所以用符号 sgn 判断类别。

M 个 model $y_m(\boldsymbol{x})$ 将按顺序依次训练得到。而且每个模型 $y_m(\boldsymbol{x})$ 的训练将根据前一模型的情况 $y_{m-1}(\boldsymbol{x})$ 进行。具体做法是：

为每个 data point 赋予一个权重 $w_n^{(m)}$，初始时（m=1）为 1/N（等权）；

每次训练一个模型 $y_m(\boldsymbol{x})$ 时，将以最小化函数 $J_m=\sum_{n=1}^{N}w_n^{(m)}I(y_m(\boldsymbol{x}_n)\neq t_n)$ 为目标，其中 $t_n\in\{-1,+1\}$ 是类别标记，I 是 indicator function；

计算模型 $y_m(\boldsymbol{x})$ 的加权错误率 $\varepsilon_m=\dfrac{\sum_{n=1}^{N}w_n^{(m)}I(y_m(\boldsymbol{x}_n)\neq t_n)}{\sum_{n=1}^{N}w_n^{(m)}}$，因此得到模型 $y_m(\boldsymbol{x})$ 的权值 $\alpha_m=\ln\{\dfrac{1-\varepsilon_m}{\varepsilon_m}\}$；

根据 $y_m(\boldsymbol{x})$ 的训练情况重新调整每个 data point 的权值，以为下一个模型的训练用：

$w_n^{(m+1)} = w_n^{(m)} \exp\{\alpha_m I(y_m(\boldsymbol{x}_n) \neq t_n)\}$。

问题：AdaBoost 整体（即 $Y(\boldsymbol{x}) = \text{sgn}(\sum_{m=1}^{M} \alpha_m y_m(\boldsymbol{x}))$）的目标优化函数是什么？

回答：是一个 **exponential error function**，$E = \sum_{n=1}^{N} \exp\{-t_n f(\boldsymbol{x}_n)\}$，其中 $t_n \in \{-1, +1\}$ 是类别标记，$\{\boldsymbol{x}_n, t_n\}$ 构成训练集，$f(\boldsymbol{x}_n) = \frac{1}{2} \sum_{m=1}^{M} \alpha_m y_m(\boldsymbol{x})$ 是 combined model。AdaBoost 是一个对此目标函数的 sequential optimization。