机器学习笔记

黎雷蕾

2017 年 10 月 8 日

摘要

学医三年, 自谓天下无不治之症。
行医三年, 始信世间无可用之方。——孙思邈

纸上得来终觉浅，绝知此事要躬行。——陆游

目录

Chapter 1

线性模型 (linear model)

1.1 基本形式

给定一个由 d 个属性描述的样本 $x = (x_1; x_2; \cdots; x_i)$, 其中 x_i 表示 x 在第 i 个属性上的取值，那么线性模型可以表示为:

$$f(x) = w_1x_1 + w_2x_2 + \cdots + w_dx_d + b \tag{1.1}$$

用向量形式来表示:

$$f(x) = w^Tx + b \tag{1.2}$$

其中 $w = (w_1; w_2; \cdots; w_d)$，一旦确定 w, b 模型就可以得到确定。

1.2 线性回归 (linear regression)

线性回归试图学习:

$$f(x_i) = wx_i + b, \text{使得 } f(x_i) \simeq y_i \tag{1.3}$$

为了得到最好的 w^*, b^*，我们可以采用均方误差（欧氏距离）最小化的

方法：

$$
\begin{aligned}
E_{(w,b)} = (w^*, b^*) &= \arg\min_{(w,b)} \sum_{i=1}^{m} (f(x_i) - y_i)^2 \\
&= \arg\min_{(w,b)} \sum_{i=1}^{m} (y_i - wx_i - b)^2
\end{aligned}
\tag{1.4}
$$

求解 1.4的过程，被称为线性回归模型的最小二乘“参数估计 (parameter estimation)”：将其分别对 w, b 求偏导：

$$
\begin{aligned}
\frac{\partial E_{(w,b)}}{\partial w} &= 2\left(w\sum_{i=1}^{m} x_i^2 - \sum_{i=1}^{m}(y_i - b)x_i\right) \\
\frac{\partial E_{(w,b)}}{\partial b} &= 2\left(wb - \sum_{i=1}^{m}(y_i - wx_i)\right)
\end{aligned}
\tag{1.5}
$$

令 1.5中两式为 0，即可求出 w, b 的最优解的闭式解：

$$
\begin{aligned}
w &= \frac{\sum_{i=1}^{m} y_i(x_i - \overline{x})}{\sum_{i=1}^{m} x_i^2 - \frac{1}{m}(\sum_{i=1}^{m} x_i)^2} \\
b &= \frac{1}{m}\sum_{i=1}^{m}(y_i - wx_i) \\
\overline{x} &= \frac{1}{m}\sum_{i=1}^{m} x_i, \quad \overline{x} \text{ 为 } x \text{ 的均值}
\end{aligned}
\tag{1.6}
$$

1.3 多元线性回归 (multivariate linear regression)

上一节中的 x 仅由单个属性描述，若其由 d 个属性进行了描述, 就可以拓展为多元线性回归。

将 w, b 写成向量形式 $\hat{w} = (w; b)$，同时把数据集表示成一个 $m \times (d+1)$ 大小的矩阵 X(m 代表样本个数，d 代表样本对应的属性个数),X 中的元素 x_{ij} 代表第 i 个样本的第 j 个属性，最后一列恒为 1：

$$
X = \begin{pmatrix}
x_{11} & x_{12} & \cdots & x_{1d} & 1 \\
x_{21} & x_{22} & \cdots & x_{2d} & 1 \\
\vdots & \vdots & \ddots & \vdots & \\
x_{m1} & x_{m2} & \cdots & x_{md} & 1
\end{pmatrix} = \begin{pmatrix}
x_1^T & 1 \\
x_2^T & 1 \\
\vdots & \vdots \\
x_m^T & 1
\end{pmatrix}
\tag{1.7}
$$

与 1.4类似

$$\begin{aligned} E_{\hat{w}} &= \hat{w}^* = \arg\min_{\hat{w}} (y - X\hat{w})^T (y - X\hat{w}) \\ \frac{\partial E_{\hat{w}}}{\partial \hat{w}} &= 2X^T (X\hat{w} - y) \end{aligned} \tag{1.8}$$

若 $X^T X$ 为满秩矩阵或者正定矩阵，则:

$$\hat{w}^* = (X^T X)^{-1} X^T y \tag{1.9}$$

令 $\hat{x_i} = (x_i, 1)$，则最终学得的多元回归模型:

$$f(\hat{x_i}) = \hat{x_i}^T (X^T X)^{-1} X^T y \tag{1.10}$$

1.4 对数线性回归 (log-linear regression)

一般来说我们得到的线性回归模型可以简写为:

$$y = w^T x + b \tag{1.11}$$

我们把 y 取对数，那它的本质是试图让 $e^{w^T x + b}$ 逼近 y，即:

$$\ln y = w^T x + b \tag{1.12}$$

一般地，考虑单调可微的函数 $g(\cdot)$，令:

$$y = g^{-1}(w^T x + b) \tag{1.13}$$

这个模型就被称为广义上的线性模型。

1.5 对数几率回归/逻辑回归 (logistic regression)

首先介绍 *Sigmoid* 函数:

$$y = \frac{1}{1 + e^{-z}} \tag{1.14}$$

它将 z 值转化为一个接近 0 或者 1 的 y 值，并且在 $z = 0$ 附近变化很陡，带入上节的对数几率函数：

$$y = \frac{1}{1 + e^{-(w^T x + b)}} \tag{1.15}$$

若将 y 视为样本 x 作为正例的可能性，那么 $1 - y$ 为其反例的可能性，两者比值取对数：

$$\ln \frac{y}{1 - y} = w^T x + b \tag{1.16}$$

这个比值称为对数几率 (log odds, 也叫 logit)。

若将 y 视为后验概率 $p(y = 1|x)$，则：

$$\begin{aligned} p(y = 1|x) &= \frac{e^{w^T x + b}}{1 + e^{w^T x + b}} \\ p(y = 0|x) &= \frac{1}{1 + e^{w^T x + b}} \end{aligned} \tag{1.17}$$

我们可以采用极大似然估计法 (maximum likehood method) 来估计 w 和 b, 那么上述的对数似然可以写为:

$$\ell(w, b) = \sum_{i=1}^{m} \ln p(y_i|x_i; w, b) \tag{1.18}$$

令 $\beta = (w; b), \hat{x} = (x; 1)$，那么 $w^T x + b = \beta^T \hat{x}$。
令 $p_1(\hat{x}; \beta) = p(y = 1|\hat{x}; \beta)$, 那么 $p_0(\hat{x}; \beta) = p(y = 0|\hat{x}; \beta) = 1 - p_1(\hat{x}; \beta)$
那么 1.18可重写为：

$$p(y_i|x_i; w, b) = y_i p_1(\hat{x_i}; \beta) + (1 - y_i) p_0(\hat{x_i}; \beta) \tag{1.19}$$

由上述几个公式，重写 1.18:

$$\ell(\beta) = \sum_{i=1}^{m} (-y_i \beta^T \hat{x}_i + \ln(1 + e^{\beta^T \hat{x_i}})) \tag{1.20}$$

1.20是高阶可导的连续凸函数，根据凸优化理论，梯度下降法或者牛顿迭代法均可以求出最优解。

$$\beta^* = \arg\min_{\beta} \ell(\beta) \tag{1.21}$$

1.6 线性判别分析 (linear discriminant analysis, LDA)

LDA 思想特别朴素：给定训练样例集，设法将样例投影到一条直线上，使得同类样例的投影点尽可能的近，异类样例的投影点尽可能远。即不同分类的样例在直线上是聚集在一起的，像一个个部落一样。

设 μ_0, μ_1 分别是两个分类的样本中心点，那么他们在直线上的投影分别是 $w^T\mu_0, w^T\mu_1$。若将所有样本点都投影到直线上，这两类样本的协方差分别为 $w^T\sum_0 w, w^T\sum_1 w$。

- 欲使同类样例投影点尽可能接近，可以让协方差尽可能小：$w^T\sum_0 w + w^T\sum_1 w$
- 欲使异类样例的投影点尽可能远离，可以让类中心之间的距离尽可能大：$||w^T\mu_0 - w^T\mu_1||_2^2$

综合上面两点，最大化目标 J 可写为：

$$\begin{aligned} J &= \frac{||w^T\mu_0 - w^T\mu_1||_2^2}{w^T\sum_0 w + w^T\sum_1 w} \\ &= \frac{w^T(\mu_0 - \mu_1)(\mu_0 - \mu_1)^T w}{w^T(\sum_0 + \sum_1)w} \end{aligned} \tag{1.22}$$

定义“类内散度矩阵”(within-class scatter matrix)：

$$\begin{aligned} S_w &= \sum_0 + \sum_1 \\ &= \sum_{x\in X_0}(x-\mu_0)(x-\mu_0)^T + \sum_{x\in X_1}(x-\mu_1)(x-\mu_1)^T \end{aligned} \tag{1.23}$$

定义“类间散度矩阵”(between-class scatter matrix):

$$S_b = (\mu_0 - \mu_1)(\mu_0 - \mu_1)^T \tag{1.24}$$

重写 1.22:

$$J = \frac{w^T S_b w}{w^T S_w w} \tag{1.25}$$

由于 1.25的解与 w 的长度无关, 只与其方向有关, 不是一般性, 令 $w^T S_w w = 1$, 最大化分子 $w^T S_b w$，一般采用拉格朗日乘子法。可得:

$$w = S_w^{-1}(\mu_0 - \mu_1) \tag{1.26}$$

为了求 S_w^{-1}, 通常的做法是对 S_w 进行奇异值分解，$S_w = U \sum V^T$, 得到 $S_w^{-1} = V \sum^{-1} U^T$，从而进行求解。

结合贝叶斯理论，当两类数据满足先验概率相同、服从高斯分布且协方差相等，LDA 可以达到最优分类

1.7 多元 LDA

假定存在 N 个类，且第 i 类示例数为 m_i，我们可以定义 S_w, S_b 和全局散度矩阵 S_t

$$\begin{aligned} S_w &= \sum_{i=1}^{N} S_{w_i} = \sum_{x \in X_i} (x - \mu_i)(x - \mu_i)^T \\ S_b &= \sum_{i=1}^{N} m_i (\mu_i - \mu)(\mu_i - \mu)^T \\ S_t &= S_w + S_b = \sum_{i=1}^{m} (x_i - \mu)(x_i - \mu)^T \end{aligned} \tag{1.27}$$

其中 μ 是所有示例的均值向量。通常知道上式三者中的两者即可，采用优化目标:

$$\max_W \frac{tr(W^T S_b W)}{tr(W^T S_w W)} \tag{1.28}$$

$$S_b W = \lambda S_w W \tag{1.29}$$

故 W 的闭式解是 $S_w^{-1} S_b$ 的 $N-1$ 个最大广义特征值所对应的特征向量组成的矩阵。**由于投影有降维的作用，故 LDA 也被视为一种经典的监督降维技术。**

1.8 多分类学习

经典的拆分策略有三个，涉及到编码不详细说明，详情见周志华《机器学习》$p.63 \sim p.66$:

- 一对一：One vs One, OvO
- 一对其余：One vs Rest, OvR
- 多对多：Many vs Many,MvM

1.9 类别不平衡问题 (class imbalance)

若训练集中正 (m^+) 反 (m^-) 例子数目不均等，那么：

$$\frac{y}{1-y} > \frac{m^+}{m^-}, \text{预测为正例} \tag{1.30}$$

故对样本进行再缩放 (rescaling)：

$$\frac{y'}{1+y'} = \frac{y}{1+y} \times \frac{m^-}{m^+} \tag{1.31}$$

总之，处理类别不平衡主要有三种方法：

- 欠采样 (undersampling): 去除一定样本，使得正负样本数目趋于平衡。
- 过采样 (oversampling): 增加一些样本。
- 再缩放策略。

1.10 小结

本章是机器学习理论的基础基础章节，介绍的是最基本的线性模型。

- 线性模型：
  - 形式简单、易于建模，许多功能更加强大的非线性模型大都都是在线性模型的基础上通过引入层级结构或者高维映射而得。

  - 线性模型中的 w 具有很好的解释性，便于理解。

- 逻辑回归，又名对数几率回归:
  - 可以直接对分类可能性进行建模，无需事先假设数据的分布，就可以避免假设分布不准确所带来的问题。
  - 它不仅可以进行分类，还可以得到近似的概率预测。
  - 对率函数是任意阶可导的凸函数，具有很好的数学性质许多数值化方法都可以用于求取最优解。
- 线性判别分析 (LDA): 是一种采用投影的策略，被视为一种经典的监督降维技术。

Chapter 2

决策树 (decision tree)

2.1 基本流程

决策树其实是一个递归建树的流程，具体步骤如下：

1. 对于属性集 A，采用信息增益或者基尼指数等方式确定当做根节点的 a_i
2. 根据 a_i 的取值将样本分成几份 $D = \{D_1^{a_i}, D_2^{a_i}, \cdots, D_n^{a_i}\}$
3. 对于每一个样本子集 $D_j^{a_i}$，重复上述的 (1), (2) 两步。直到样本子集里的每个样本属于同一类别 C。

2.2 划分选择

2.2.1 信息增益 (information gain)

首先需要了解信息熵 (information entropy)，信息熵代表了信息不确定的程度：信息熵越大，说明信息越不确定，那么纯度越低；反之，若信息熵很小，说明信息确定程度高，那么信息纯度越高。假设当前样本集合 D 中第 k 类样本所占的比例为 $p_k, (k = 1, 2, \cdots, |\mathcal{Y}|)$，那么信息熵可以被定义为：

$$Ent(D) = -\sum_{k=1}^{|\mathcal{Y}|} p_k \log_2 p_k \tag{2.1}$$

由于信息熵取值和纯度大小呈反比，为了便于理解，我们引入“信息增益”，假设一个属性 a 的取值为 $\{a^1, a^2, \cdots, a^V\}$，那么样本集合 D 可以按照 a 分成 V 份，假设按照 a^v 分的样本子集是 D^v，那么信息增益 $Gain(D, a)$ 可以写为：

$$Gain(D, a) = Ent(D) - \sum_{v=1}^{V} \frac{|D_v|}{|D|} Ent(D^v) \tag{2.2}$$

为了得到最大的信息增益，我们可以求出所有属性 a^i 的信息增益 $Gain(D, a^i)$，从中选择信息增益最大的 a_* 作为当前的节点，这个思想就是**ID3 决策树学习算法。**

$$a_* = \arg\max_{a \in A} \ Gain(D, a) \tag{2.3}$$

2.2.2 信息增益率 (information gain ratio)

信息增益准则会对取值数目较多的属性有所偏好，为了减少这种不利影响，我们采用增益率 (gain ratio) 来划分最优属性，这个思想就是C4.5 决策树算法。

$$\begin{aligned} Gain\_ratio(D, a) &= \frac{Gain(D, a)}{IV(a)} \\ IV(a) &= -\sum_{v=1}^{V} \frac{|D^v|}{|D|} \log_2 \frac{|D^v|}{|D|} \end{aligned} \tag{2.4}$$

2.3 基尼指数 (Gini index)

CART 决策树采用基尼指数来选择划分属性，数据集 D 的纯度可以用基尼值 $Gini(D)$ 来测量：

$$Gini(D) = \sum_{k=1}^{|\mathcal{Y}|} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{|\mathcal{Y}|} p_k^2 \tag{2.5}$$

和信息熵类似，基尼值越小，数据集的纯度越高。基尼指数可以定义为：

$$Gini\_index(D, a) = \sum_{v=1}^{V} \frac{|D^v|}{|D|} Gini(D^v) \tag{2.6}$$

与信息增益相反，我们选择基尼系数最小的属性当做划分属性：

$$a_* = \arg\min_{a \in A} \ Gini\_index(D, a) \tag{2.7}$$

2.4 剪枝处理 (pruning)

剪枝是为了避免决策树算法是否进入‘过拟合’的手段。

2.4.1 预剪枝 (prepruning)

预剪枝是指在决策树生成过程中对当前节点进行估计，若当前节点的划分不能带来决策树泛化性能的提升，则停止划分并将当前节点标记为叶节点。

- 计算不分叶节点之前验证集的精度 p_{pre}。
- 计算分开的叶节点之后的验证集精度 p_{post}
- 若 $p_{post} > p_{pre}$ 则扩展该节点，否则直接将其作为叶节点。
- 预剪枝可以减少很多不必要的分支，时间开销较小，但是会带来更大的欠拟合风险。

2.4.2 后剪枝 (postpruning)

- 和预剪枝类似，也是采用划分前后的验证集精度来决定是否进行剪枝。
- 不同的是，后剪枝是先分叶节点后再剪枝。
- 相比于预剪枝，后剪枝通常可以保留更加多的叶节点，所以后剪枝的欠拟合风险较小，泛化性能优于预剪枝，但是后剪枝需要在决策树生成后才能进行，训练的开销要大于预剪枝。

2.5 连续之和缺失值

2.5.1 连续值

对于连续值，一般采用连续属性离散化技术，最简单的策略是采用二分法 (bi-partition) 对连续属性进行处理，C4.5 决策树算法就是采用这种方法。

给定样本集 D 和连续属性 a, 假定 a 在 D 上出现了 n 个不同的取值，将其按从大到小的顺序进行排列, 记为 $\{a_1.a_2,\cdots,a_n\}$，基于划分点 t 可将 D 分为子集 D_t^- 和 D_t^+, 其中 $a_i \leq a_t, a_i \in D_t^-$ 且 $a_j > a_t, a_j \in D_t^+$，对于相邻的属性 a_t, a_{t+1},t 在区间 $[a_t, a_{t+1})$ 上取任意值的划分相同。那么对于一个连续属性 a_t 我们就可以考察包含 $n-1$ 个袁术的划分点集合，即：

$$T_n = \left\{ \frac{a_i + a_{i+1}}{2} | 1 \leq i \leq n-1 \right\} \tag{2.8}$$

上述公式的意思就是把区间 $[a_t, a_{t+1})$ 上的中位点 $\frac{a_i+a_{i+1}}{2}$ 作为候选的划分点，从而连续值就变成了离散值。

2.5.2 缺失值

对于缺失值，我们通常是采用给特定的信息增益赋予权值 ρ 的方式。

1. 假设某一个属性 a_i 的集合为 A, 缺失的属性集合为 A^*，那么在 D 上有，$|D_A| = |A| + |A^*|$, 其中 $|A|$ 代表集合 A 中的属性个数。

2. 我们把 A 当做一个无缺失值的属性，计算出相应的信息增益 $Gain(A, a_i)$.

3. 实际上属性的信息增益：

$$Gain(D_A, a_i) = \rho \times Gain(A, a_i) = \frac{|A|}{|A| + |A^*|} \times Gain(A, a_i) \tag{2.9}$$

4. 若用该属性作为父节点，那么缺失值将同时进入所有的子节点，在每个节点计算信息增益时，它的权重：

$$\rho_{child} = \frac{\text{该子节点不缺失的属性个数}}{\text{父节点不缺失属性的个数}} \tag{2.10}$$

2.6 多变量决策树 (multivariate decision tree)

上述所有的决策树算法的节点都是以单个属性为准，而我们实际的情况下，经常会用到多变量作为决策树的分界点，每个非叶结点都是形如 $\sum_{i=1}^{d} w_i a_i = t$ 的线性分类器，这个分类器可能会采取 $softmax$ 之类的方式进行决策，不太便于解释。

2.7 随机森林

2.8 小结

Chapter 3

贝叶斯分类器 (bayes classifier)

由于在深极做过一次 ppt 的演讲了，所以这一章不用写得太详细。

3.1 贝叶斯公式

贝叶斯公式的本质就是通过条件概率 (也叫似然) 之间的转化，建立先验概率 $P(x|c)$ 与后验概率 $P(c|x)$ 之间的联系。

$$P(c|x) = \frac{P(c,x)}{P(x)} = \frac{P(x|c)P(c)}{P(x)} \tag{3.1}$$

3.2 贝叶斯决策论 (bayesian decision theory)

假设有 N 种可能的类别标记。即 $\mathcal{Y} = \{c_1, c_2, \cdots, c_N\}$,$\lambda_{ij}$ 是将一个真实标记为 c_j 的样本误分类为 c_i 的损失，那么基于后验概率 $P(c_i|x)$ 就可以获得将样本 x 分类为 c_i 所产生的期望损失，即 x 上的条件风险 (conditional risk)。

$$R(c_i|x) = \sum_{j=1}^{N} \lambda_{ij} P(c_j|x) \tag{3.2}$$

我们的目标即找出 x 的分类，使得期望风险 $R(c|x)$ 最小:

$$h^*(x) = \arg\min_{c \in \mathcal{Y}} R(c|x) \tag{3.3}$$

3.3 极大似然估计 (Maximum Likelihood Estimation,MLE)

- 对 θ_c 进行极大似然估计。就是寻找能最大化似然 $P(D_c|\theta_c)$ 的参数值 $\hat{\theta}_c$。换句话说，就是遍历 θ_c 所有可能的取值，找出一个使数据出现的“可能性”最大的一个。

- 为了加快计算速度和减少溢出的可能性，可以用取对数相加代替连乘的操作：

$$LL(\theta_c) = \log P(D_c|\theta_c) = \sum_{x \in D_c} \log P(x|\theta_c) \tag{3.4}$$

- 此时参数 θ_c 的极大似然估计 $\hat{\theta}_c$ 可以写为：

$$\hat{\theta}_c = \arg\max_{\theta_c} LL(\theta_c) \tag{3.5}$$

- 总之，MLE 的思想可以总结为：已知某个参数能使这个样本出现的概率最大，我们当然不会再去选择其他小概率的样本，所以干脆就把这个参数作为估计的真实值。

3.4 朴素贝叶斯分类器 (Naïve Bayes Classifier)

- 朴素 (Naïve)，指的是“属性条件独立性假设”，即对于已知类别，假设所有的属性相互独立。聊天监控系统里采用的贝叶斯算法就是基于 NBC 的。

- 在这个前提下，贝叶斯公式可以改写：

$$P(c|x) = \frac{P(c)P(x|c)}{P(x)} = \frac{P(c)}{P(x)} \prod_{i=1}^{d} P(x_i|c) \tag{3.6}$$

  其中，d 为属性的数目，x_i 为 x 在第 i 个属性上的取值。

- 由上，我们可得朴素贝叶斯分类器的表达式。

$$h_{nb}(x) = \arg\max_{c \in \mathcal{Y}} P(c) \prod_{i=1}^{d} P(x_i|c) \tag{3.7}$$

- 对于贝叶斯分类器，若前提有大量的训练集，我们可以事先训练贝叶斯分类所有的概率估值，进行测试时，我们只需进行查表操作即可，借助哈希表、二叉树等数据结构进行存储，贝叶斯分类器的时间复杂度为 $O(n)$。

3.5 半朴素贝叶斯分类器 (semi-Naïve Bayes Classifier)

- NBC 是基于属性之间是相互独立的假设，但是在现实条件下这个假设是很难实现的，所以我们提出了半朴素贝叶斯分类器，它的基本想法是适当考虑一部分属性间的相互依赖信息，从而既不需要进行完全联合计算，又不至于彻底忽略了比较强的属性依赖关系。

- 采用得比较多得是“独依赖估计”(One-Dependent Estimator,ODE), 指的是每个属性最多仅依赖一个其他属性，即:

$$P(c|x) \propto P(c) \prod_{i=1}^{d} P(x_i|c, pa_i) \tag{3.8}$$

其中 pa_i 为属性 x_i 所依赖的属性，称为 x_i 的父属性。

3.6 贝叶斯网 (Bayesian network)

- 贝叶斯网也称为信念网 (belief network)。它借助有向无环图来刻画属性间的依赖关系，并使用条件概率表来描述属性的联合概率分布。

- 贝叶斯网 B 可以表示成如下公式:

$$B = \langle G, \Theta \rangle \tag{3.9}$$

其中 G 表示一个有向无环图，Θ 定量描述两个属性之间的直接依赖关系。假设属性 x_i 在 G 中的父结点集为 π_i, 则 Θ 包含了每个属性的条件概率表 $\theta_{x_i|\pi_i} = P_B(x_i|\pi_i)$。

- 贝叶斯网学习的首要任务是通过训练集构建一个最合理的贝叶斯网，一般采用评分搜索的办法。
- 首先定义一个评分函数 (score function)，基于信息论准则，其目标是找到一个能以最小编码长度描述训练模型，即“最小描述长度”(Minimal Description Length,MDL)

3.6.1 贝叶斯网 (Bayesian network)-学习

求 MDL 的过程可以描述如下：

1. 给定训练集 $D = \{x_1, x_2, \cdots, x_m\}$，那么贝叶斯网 $B = \langle G, \Theta \rangle$ 的评分函数可以写为：

$$s(B|D) = f(\theta)|B| - LL(B|D) \tag{3.10}$$

其中 $|B|$ 是贝叶斯网的参数个数；$f(\theta)$ 表示描述每个参数所需的字节数，$LL(B|D)$ 表示贝叶斯网 B 的对数似然。

$$LL(B|D) = \sum_{i=1}^{m} \log P_B(x_i) \tag{3.11}$$

2. 学习任务转化为一个优化任务，即寻找一个贝叶斯网 B 使评分函数 $s(B|D)$ 最小。

3. 若 $f(\theta) = 0$，即不计算对网络编码的长度，评分函数退化成负对数似然，那么学习任务退化成极大似然估计。

$$s(B|D) = -LL(B|D) = -\sum_{i=1}^{m} \log P_B(x_i) \tag{3.12}$$

4. 若 $B = \langle G, \Theta \rangle$ 的网络结构 G 固定，则 $s(B|D)$ 等价于对参数 Θ 的极大似然估计，那么 $\theta_{x_i|\pi_i}$ 可以直接在训练数据 D 上通过经验估计得到：

$$\theta_{x_i|\pi_i} = \hat{P}_D(x_i|\pi_i) \tag{3.13}$$

其中 $\hat{P}_D(\cdot)$ 是 D 上的经验分布。

5. 为了最小化评分函数 $s(B|D)$，只需要对网络结构进行搜索，而候选结构的最优参数可以直接在训练集上计算得到。

6. 搜索出贝叶斯网最优结构是一个 NP 难的问题，难以快速求解，一般常用两种方法保证在有限时间内求得近似解。

    - 采用贪心策略，从某个网络结构出发，每次调整一条边，直到评分函数值不再降低为止。
    - 通过网络结构施加约束来削减搜索空间，例如将网络结构限定为树形结构。

3.6.2 贝叶斯网 (Bayesian network)-推断

- 通过前面的训练和学习，贝叶斯网就可以通过一些属性变量的观测值来推测其他属性变量的取值，这个过程我们称为“推断”(inference)，已知变量观测值称为“证据”(evidence)。
- 理想情况下是直接根据贝叶斯网定义的联合概率分布计算后验概率，但前面已经说明搜索最优结构是 NP 难的，在现实应用中，贝叶斯网的近似推断常使用吉布斯采样 (Gibbs sampling) 来完成。

3.7 EM(Expectation-Maximization) 算法

- 在实际应用的时候，我们很难获得所有属性变量的值，即训练样本是不完整的，像这种无法获得属性变量的“未观测”变量，我们称为“隐变量”(latent variable)。
- EM(Expectation-Maximization) 算法，就是常用的估计参数隐变量的算法。它的基本思想很简单：若参数 Θ 已知，则可根据训练数据推断出最优隐变量 Z 的值（E 步）；反之，若 Z 的值已知，则可方便地对 Θ 做极大似然估计（M 步）。

若我们不是取 Z 的期望,而是基于 Θ^t 计算隐变量 Z 的概率分布 $P(Z|X,\Theta^t)$，那么 em 算法两步可以直接定义:

- E 步 (Expectation)：以当前参数 Θ^t 推断隐变量分布 $P(Z|X,\Theta^t)$，并计算对数似然 $LL(\Theta|X,Z)$ 关于 Z 的期望。

$$Q(\Theta|\Theta^t) = \mathbb{E}_{Z|X,\Theta^t} LL(\Theta|X,Z) \tag{3.14}$$

- M 步 (Maximization)：寻找参数最大化期望似然：

$$\Theta^{t+1} = \arg\max_{\Theta} Q(\Theta|\Theta^t) \tag{3.15}$$

3.8 小结

贝叶斯模型是一个 precision 很高的模型，而且属于线性模型，十分便于工程实现。

最简单的朴素贝叶斯分类器，在很多情况下都能够获得相当好的性能，十分适合信息检索领域，是常用的文本分类策略之一。