机器学习笔记

tszhang

2019 年 7 月 20 日

目录

1 文本表示模型

词的表示方法分为：**one-hot表示**和**分布式表示**。基于分布式表示的词方法根据建模不同可以分为：基于矩阵的分布表示、基于聚类的分布表示和基于神经网络的分布表示。

语言模型包括文法语言模型和统计语言模型。

文本表示模型包括：词袋模型，TF-IDF，主题模型，词嵌入模型。下面分别描述各个模型的原理。

1.1 词袋模型和N-gram模型

bag-of-words方法(BOW)，不考虑文档内的词的顺序关系和语法，只考虑该文档是否出现过这个单词。构建一个出现的词的字典，然后观察某个文档中是否出现过词典中的词，出现的话词典对应位置为1 （或出现的次数），否则为0. 然而如果连续的将N个词分开，而不是一个词一个词的分开，就形成了N-gram。

1.2 TF-IDF

如果某个单词在一篇文章中出现的频率TF高，并且在其他文章中很少出现，则认为此词或者短语具有很好的类别区分能力，适合用来分类。公式为

$$TF-IDF(t,d)=TF(t,d)\times IDF(t) \tag{1.1}$$

其中TF(t,d)为单词t在文档中出现的频率，IDF(t)是逆文档频率，用来衡量单词t的重要性，反应了一个词在所有文本中出现的频率，如果一个词在很多的文本中出现，那么它的IDF 值应该低，而反过来如果一个词在比较少的文本中出现，那么它的IDF值应该高。表示为

$$IDF(t)=log\frac{\text{语料库的文档总数}}{\text{包含词跳t的文档总数+1}} \tag{1.2}$$

应用：搜索引擎，关键字提取，文本相似性。

1.3 主题模型

1.4 词嵌入模型

嵌入(embedding)表示寻找一个映射函数，将源数据映射到另外一个空间，该函数具有单射和单调性。分为两种模型：Word2vec与glove。

Word2vec模型分为：1)Skip-gram模型:输入词汇预测上下文。2) CBOW模型:输入上下文预测词汇。

2 模型评估

2.1 评估方法

留出法，交叉验证法，Hold-Out方法(先抽取一部分做验证，剩下的数据作为训练集)，自助法。

2.2 P-R曲线

P值是指精确率是指分类正确的正样本个数占分类器判定为正样本的样本个数的比例（查准率）。R是指召回率是指分类正确的正样本个数占真正的正样本个数的比例（查全率）。P-R曲线的横轴是召回率，纵轴是精确率。设定一个阈值，大于此阈值的判定为正样本，小于此阈值的判定为负样本。依次改变此阈值就可以画出P-R 曲线。

$$F_1 = \frac{2 \times P \times R}{P + R} \tag{2.1}$$

2.3 ROC曲线

TP：预测positive正确，即将正类预测为正类。FN：将正类预测为负类。FP：将负类预测为正类。TN：将负类预测为负类。纵坐标为真正例率：$TPR = \frac{TP}{P}$，横坐标为假正例率：$FPR = \frac{FP}{N}$. ROC曲线是通过不断移动分类器的“截断点”来生成曲线上的一组关键点的。截断点为每个样本的预测概率。ROC曲线下面的面积为AUC。

2.4 P-R曲线与ROC曲线对比

当正负样本的分布发生变化时，ROC曲线的形状能够基本保持不变，而P-R曲线的形状一般会发生较剧烈的变化。

具体证明可以根据公式，例如假设正样本数量不变，负样本增加10倍，看横纵坐标的比值变化。

2.5 过拟合与欠拟合

降低过拟合方法：获得更多的训练数据，降低模型复杂度，正则化方法，集成学习方法。

降低欠拟合方法：添加新特征，增加模型复杂度，减小正则化系数。

3 逻辑回归与最大熵模型

3.1 逻辑回归

一种分类器。模型公式如下：

$$P(y=1|x;\theta)=\frac{1}{1+e^{-\theta^T x}},\quad P(y=0|x;\theta)=1-P(y=1|x;\theta) \tag{3.1}$$

使用极大似然估计求参数θ，似然函数为

$$\prod_{i=1}^{N}P(y=1|x_i)^{y_i}[1-P(y=1|x_i)]^{1-y_i} \tag{3.2}$$

对数似然函数为

$$L(w)=\sum_{i=1}^{N}[y_i logP(y=1|x_i)+(1-y_i)log(1-P(y=1|x_i))] \tag{3.3}$$

对$L(w)$求极大值即可得到参数的估计值。

损失函数是找最小值，故逻辑回归的损失函数为$-L(w)$。

3.2 最大熵原理

学习概率模型时，在所有满足约束条件的概率模型中，熵最大的模型是最好的模型。当变量为均匀分布时，熵最大。

3.3 最大熵模型

那么约束条件从哪里来呢？最大熵模型做法是：从训练集T中抽取特征，然后要求这些特征在训练集中的经验分布$\hat{P}(X,Y)$的期望等于它们关于模型P(Y|X)与经验分布$\hat{P}(X)$的期望值。用特征函数f描述输入与输出的一个事实。

$$f(x,y)=\begin{cases}1 & \text{x与y满足某一事实}\\ 0, & \text{否则}\end{cases} \tag{3.4}$$

特征函数在训练数据上关于经验分布$\hat{P}(X,Y)$ 的期望值

$$E_{\hat{p}}(f)=\sum_{x,y}\hat{P}(x,y)f(x,y) \tag{3.5}$$

特征函数关于模型P(y|x)的数学期望

$$E_p(f)=\sum_{x,y}\hat{P}(x)P(y|x)f(x,y) \tag{3.6}$$

我们希望特征f的期望应该和从训练数据中得到的特征的期望是一样的，即$E_{\hat{p}}(f)=E_p(f)$，就是约束条件。

最大熵模型就是约束条件下条件熵最大的模型:

$$H(P) = -\sum_{x,y} \hat{P}(x)P(y|x)logP(y|x) \tag{3.7}$$

为什么使用条件熵呢？因为使用条件熵就有我们要求的P(y|x).

故最大熵模型的学习就等价于求解约束问题: 并且将求最大值转为最小值。

$$\begin{aligned} &\min_{P\in C} \sum_{x,y} \hat{P}(x)P(y|x)logP(y|x) \\ &s.t. E_{\hat{p}}(f_i) = E_p(f_i), i = 1, 2, \ldots, n \\ &\sum_y P(y|x) = 1 \end{aligned} \tag{3.8}$$

引入拉格朗日乘子后公式为:

$$\begin{aligned} L(P, w) &= -H(P) + w_0[1 - \sum_y P(y|x)] + \sum_{i=1}^{n} w_i(E_{\hat{p}}(f_i) - E_p(f_i)) \\ &= \sum_{x,y} \hat{P}(x)P(y|x)logP(y|x) + w_0[1 - \sum_y P(y|x)] + \\ &\sum_{i=1}^{n} w_i(\sum_{x,y} \hat{P}(x, y)f(x, y) - \sum_{x,y} \hat{P}(x)P(y|x)f(x, y)) \end{aligned} \tag{3.9}$$

最优化原始问题为$\min_{P\in C} \max_{w} L(P, w)$, 对偶问题为$\max_{w} \min_{P\in C} L(P, w)$, 二者是等价的。

使用偏导为零的方法求P,

$$\begin{aligned} \frac{\partial L(P, w)}{\partial P(y|x)} &= \sum_{x,y} \hat{P}(x)(logP(y|x) + 1) - \sum_y w_0 - \sum_{x,y} \hat{P}(x) \sum_{i=1}^{n} w_i f_i(x, y) \\ &= \sum_{x,y} \hat{P}(x)[logP(y|x) + 1 - w_0 - \sum_{i=1}^{n} w_i f_i(x, y)] \end{aligned} \tag{3.10}$$

令其为0，在$\hat{P}(x) > 0$的情况下，得到

$$P(y|x) = exp[\sum_{i=1}^{n} w_i f_i(x, y) + w_0 - 1] = exp(w_0 - 1)exp[\sum_{i=1}^{n} w_i f_i(x, y)] \tag{3.11}$$

将上式代入$\sum_y P(y|x) = 1$得,

$$\sum_y exp(w_0 - 1)exp[\sum_{i=1}^{n} w_i f_i(x, y)] = exp(w_0 - 1) \sum_y exp[\sum_{i=1}^{n} w_i f_i(x, y)] = 1 \tag{3.12}$$

故

$$exp(w_0 - 1) = \frac{1}{\sum_y exp[\sum_{i=1}^{n} w_i f_i(x, y)]} \tag{3.13}$$

故

$$P(y|x)=\frac{exp[\sum_{i=1}^{n}w_if_i(x,y)]}{\sum_y exp[\sum_{i=1}^{n}w_if_i(x,y)]} \tag{3.14}$$

这个$P(y|x)$就是最大熵模型的解

将$P(y|x)$代入原始式子化简最终得到:

$$\psi(w)=\sum_{x,y}\hat{P}(x,y)\sum_{i=1}^{n}w_if_i(x,y)-\sum_x\hat{P}(x)logZ_w(x) \tag{3.15}$$

可以使用梯度下降法，拟牛顿法，IIS方法来求解

3.4 与极大似然估计的关系

最大熵模型学习中的对偶函数最大化等价于最大熵模型的极大似然估计。

证明：已知数据集的$\hat{P}(X,Y)$,条件概率分布$P(Y|X)$对对数似然函数表示为:

$$\begin{aligned}L_{\hat{P}}(P_w)&=\sum_{x,y}\hat{P}(x,y)logP(y|x)\\&=\sum_{x,y}\hat{P}(x,y)\sum_{i=1}^{n}w_if_i(x,y)-\sum_{x,y}\hat{P}(x,y)logZ_w(x)\\&=\sum_{x,y}\hat{P}(x,y)\sum_{i=1}^{n}w_if_i(x,y)-\sum_{x}\hat{P}(x)logZ_w(x)\end{aligned} \tag{3.16}$$

这样，最大熵模型的学习问题就转换为具体求解对数似然函数极大化或对偶函数极大化的问题。

4 支持向量机

4.1 SVM原理推导

函数间隔:

$$\hat{\gamma} = \min\ \hat{\gamma}_i = \min\ y_i(w \cdot x_i + b)$$

几何间隔:

$$\gamma = \min\ \gamma_i = \min\ y_i \frac{w \cdot x_i + b}{\|w\|}$$

最大间隔分离超平面:

$$\begin{aligned} &\max_{w,b}\ \gamma \\ &s.t.\ y_i[\frac{w}{\|w\|} \cdot x_i + \frac{b}{\|w\|}] \geq \gamma,\ i = 1, 2, \cdots, m \end{aligned} \tag{4.1}$$

将γ换成$\hat{\gamma}$,得到

$$\begin{aligned} &\max_{w,b}\ \frac{\hat{\gamma}}{\|w\|} \\ &s.t.\ y_i(w \cdot x_i + b) \geq \hat{\gamma},\ i = 1, 2, \cdots, m \end{aligned} \tag{4.2}$$

函数间隔的取值并不影响优化问题的解。取值$\hat{\gamma} = 1$,且将问题变为最小化。可以得到:

$$\begin{aligned} &\min_{w,b}\ \frac{1}{2}\|w\|^2 \\ &s.t.\ y_i(w \cdot x_i + b) - 1 \geq 0,\ i = 1, 2, \cdots, m \end{aligned} \tag{4.3}$$

构建拉格朗日函数:

$$L(w, b, \alpha) = \frac{1}{2}\|w\|^2 + \sum_{i=1}^{m} \lambda_i [1 - y_i(w x_i + b)] \tag{4.4}$$

分别对w,b求偏导且令偏导为零。得到:

$$\begin{aligned} w &= \sum_{i=1}^{m} \alpha_i y_i x_i \\ 0 &= \sum_{i=1}^{m} \alpha_i y_i \end{aligned}$$

将此结果代入到拉格朗日函数即可得到对偶问题:

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j x_i^T x_j \\ &s.t.\ \sum_{i=1}^{m} \alpha_i y_i = 0, \qquad \alpha_i \geq 0,\ i = 1, 2, \cdots, m \end{aligned} \tag{4.5}$$

可以使用二次规划来求解，但是该问题的规模正比于训练样本的个数，会造成很大的开销。因此使用SMO 算法。

4.2 核函数

若在低维空间中线性不可分，则要映射到高维空间中，以便线性可分。

令$\phi(x)$表示将x映射后的特征向量。根据前文，其问题可以转化为:

$$\begin{aligned} &\max_{\alpha} \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \phi(x_i)^T \phi(x_j) \\ &s.t. \sum_{i=1}^{m} \alpha_i y_i = 0, \qquad \alpha_i \geq 0,\ i = 1, 2, \cdots, m \end{aligned} \tag{4.6}$$

由于映射后的特征空间维数可能很高，甚至可能无穷维，因此直接计算$\phi(x_i)^T \phi(x_j)$通常是困难的，为了避开这个障碍，引入核函数。

列举常用的核函数

名称	表达式	参数
线性核	$\kappa(x_i, x_j) = x_i^T x_j$	
多项式核	$\kappa(x_i, x_j) = (x_i^T x_j)^d$	$d \geq 1$为多项式的次数
高斯核	$\kappa(x_i, x_j) = exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2})$	$\sigma > 0$为高斯核的带宽(width)
拉普拉斯核	$\kappa(x_i, x_j) = exp(-\frac{\|x_i - x_j\|}{\sigma})$	$\sigma > 0$
Sigmoid核	$\kappa(x_i, x_j) = tanh(\beta x_i^T x_j + \theta)$	tanh为双曲正切函数,$\beta > 0, \theta < 0$

4.3 软间隔

前面讨论中，一直假定是线性可分的，然而在现实生活中往往很难有这样的样本。

缓解该问题的一个办法就是允许支持向量机在一些样本上出错，为此，要引入软间隔概念。在最大化间隔的同时，不满足约束条件的样本尽可能少。于是，优化函数如下公式:

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{m} l_{0/1}(y_i(w^T x_i + b) - 1) \tag{4.6}$$

其中$C > 0$是一个常数，$l_{0/1}$是‘0/1损失函数’。

然而$l_{0/1}$非凸、非连续，数学性质不好。所以给出了三种常用的替代损失函数:

$$\begin{aligned} &hinge\text{损失}: l_{hinge}(z) = max(0, 1 - z) \\ &\text{指数损失}: l_{exp}(z) = exp(-z) \\ &\text{对率损失}: l_{log}(z) = log(1 + exp(-z)) \end{aligned}$$

采用hinge损失，则式(4.6)变为

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2 + C \sum_{i=1}^{m} \max(0, 1 - y_i(w^T x_i + b)) \tag{4.7}$$

引入松弛变量$\xi_i \geq 0$,可重写为

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i \\ s.t. \quad & y_i(w^T x_i + b) \geq 1 - \xi_i \\ & \xi_i \geq 0, \quad i = 1, 2, \cdots, m. \end{aligned} \tag{4.8}$$

这就是常用的‘软间隔支持向量机’。

则拉格朗日函数为:

$$L(w, b, \alpha, \xi, \mu) = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i(1 - \xi_i - y_i(w^T x_i + b)) - \sum_{i=1}^{m}\mu_i\xi_i \tag{4.9}$$

其中$\alpha \geq 0, \mu \geq 0$是拉格朗日乘子。

令$L(w, b, \alpha, \xi, \mu)$对$w, b, \xi_i$的偏导数为零可得:

$$\begin{aligned} w &= \sum_{i=1}^{m}\alpha_i y_i x_i \\ 0 &= \sum_{i=1}^{m}\alpha_i y_i \\ C &= \alpha_i + \mu_i \end{aligned}$$

代入到式(4.9)可以得到式(4.8)的对偶问题

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j x_i^T x_j \\ s.t. \quad & \sum_{i=1}^{m}\alpha_i y_i = 0, \qquad 0 \leq \alpha_i \leq C, \ i = 1, 2, \cdots, m \end{aligned} \tag{4.10}$$

4.4 SMO算法

有点类似坐标上升法：每次只优化一个变量。然而因为满足关系$\sum_{i=1}^{m}\alpha_i y_i = 0$，所以每次优化两个变量，把剩下的变量当作常数。即$\alpha_1 y_1 + \alpha_2 y_2 = c, \ \alpha_1 = (c - \alpha_2 y_2) y_1$

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K_{i,j} - \sum_{i=1}^{m}\alpha_i \\ s.t. \quad & \sum_{i=1}^{m}\alpha_i y_i = 0, \qquad \alpha_i \geq 0, \ i = 1, 2, \cdots, m \end{aligned} \tag{4.11}$$

将α_1, α_2单独分离开来得到:

$$\frac{1}{2}\alpha_1^2 K_{11} + \alpha_1\alpha_2 y_1 y_2 K_{12} + \frac{1}{2}\alpha_2^2 K_{22} - \alpha_1 - \alpha_2 + \alpha_1 y_1 \sum_{i=3}^{m}\alpha_i y_i K_{1i} + \alpha_2 y_2 \sum_{i=3}^{m}\alpha_i y_i K_{2i} + \text{常数} \tag{4.12}$$

将$\alpha_1=(c-\alpha_2y_2)y_1$代入到式(4.12) 中，化简得到

$$\begin{aligned}\min_{\alpha}\ &\frac{1}{2}(c-\alpha_2y_2)^2K_{11}+(c-\alpha_2y_2)\alpha_2y_2K_{12}+\frac{1}{2}\alpha_2^2K_{22}-\\&(c-\alpha_2y_2)y_1-\alpha_2+(c-\alpha_2y_2)\sum_{i=3}^{m}\alpha_iy_iK_{1i}+\alpha_2y_2\sum_{i=3}^{m}\alpha_iy_iK_{2i}\end{aligned}\tag{4.12}$$

要求最小值，求上式对α_2偏导数，

$$\begin{aligned}&-y_2(c-\alpha_2y_2)K_{11}+y_2K_{12}(c-\alpha_2y_2)-\alpha_2K_{12}+\alpha_2K_{22}+y_1y_2-1-y_2\sum_{i=3}^{m}\alpha_iy_iK_{1i}+y_2\sum_{i=3}^{m}\alpha_iy_iK_{2i}\\=&(K_{11}-2K_{12}+K_{22})\alpha_2-cK_{11}y_2+cK_{12}y_2+y_1y_2-1-y_2\sum_{i=3}^{m}\alpha_iy_iK_{1i}+y_2\sum_{i=3}^{m}\alpha_iy_iK_{2i}\end{aligned}$$

已知

$$\begin{aligned}&\alpha_1^{old}y_1+\alpha_2^{old}y_2=c\\&\sum_{i=3}^{m}\alpha_iy_iK_{1i}=f(x_1)-b-\alpha_1y_1K_{11}-\alpha_2y_2K_{12}\\&\sum_{i=3}^{m}\alpha_iy_iK_{2i}=f(x_2)-b-\alpha_1y_1K_{21}-\alpha_2y_2K_{22}\end{aligned}$$

代入化简得：

$$(K_{11}-2K_{12}+K_{22})\alpha_2^{new}=(K_{11}-2K_{12}+K_{22})\alpha_2^{old}-y_2[(f(x_1)-y_1)-(f(x_2-y_2))]$$

故

$$\alpha_2^{new}=\alpha_2^{old}+\frac{y_2(E_1-E_2)}{\eta}\tag{4.12}$$

其中$\eta=K_{11}-2K_{12}+K_{22}$

然后对α_2进行裁剪。然后根据$\alpha_1^{old}y_1+\alpha_2^{old}y_2=\alpha_1^{new}y_1+\alpha_2^{new}y_2$ 求得

$$\alpha_1^{new}=\alpha_1^{old}+y_1y_2\alpha_2^{old}-\alpha_2^{new}\tag{4.13}$$

接下来求b，需要使用支持向量，即支持向量满足$y_1(w^Tx_1+b)=1$，同乘以y_1得到$\sum_{i=1}^{m}\alpha_1y_1K_{i1}+b=y_1$，化简得

$$b_1^{new}=y_1-\sum_{i=3}^{m}\alpha_iy_iK_{i1}-\alpha_1^{old}y_1K_{11}-\alpha_2^{old}y_2K_{12}$$

同理可求

$$b_2^{new}=y_2-\sum_{i=3}^{m}\alpha_iy_iK_{i2}-\alpha_1^{old}y_1K_{12}-\alpha_2^{old}y_2K_{22}$$

$$b^{new}=\frac{b_1^{new}+b_2^{new}}{2}\tag{4.14}$$

如何选取α_1, α_2呢？首先将α 置零，然后选取α_1：找到违反KKT条件最严重的点。α_2选取标准为使$|E_1 - E_2|$ 的值最大。KKT条件:

$$\alpha_1 = 0 \Leftrightarrow y_i g(x_i) \geq 0$$
$$0 < \alpha_i < C \Leftrightarrow y_i g(x_i) = 0$$
$$\alpha_i = C \Leftrightarrow y_i g(x_i) \leq 0$$

结束条件：所有点都满足KKT条件。

5 决策树

熵表示表示随机变量的不确定度：

$$H(x) = -\sum_{i=1}^{m} p_i \log\ p_i = -\sum_{k=1}^{K} \frac{|C_k|}{|D|} \log \frac{C_k}{|D|}$$

条件熵：

$$H(y|x) = -\sum_{i=1}^{m} p(x_i)p(y|x_i) \log\ p(y|x_i) = \sum_{i=1}^{m} p_i H(y|x = x_i) = \sum_{i=1}^{n} \frac{|D_i|}{|D|}(-\sum_{k=1}^{k} \frac{|D_{ik|}}{|D_i|} \log \frac{|D_{ik|}}{|D_i|})$$

信息增益：得知特征X的信息而使类Y的信息不确定性减少的程度$g(D, A) = H(D) - H(D|A)$

数据集D关于A的取值熵：

$$H_A(D) = -\sum_{ii=1}^{n} \frac{|D_i|}{|D|} \log \frac{|D_i|}{|D|}$$

信息增益比：信息增益与 数据集D关于A的取值熵 的比值:

$$g_R(D, A) = \frac{g(D, A)}{H_A(D)}$$

基尼指数：Gini描述的是数据的纯度，与信息熵含义类似

$$Gini(D) = 1 - \sum_{i=1}^{K} (\frac{|C_i|}{|D|})^2$$

在特征A的条件下，集合D的基尼指数定义

$$Gini(D|A) = \frac{|D_1|}{|D|} Gini(D_1) + \frac{|D_2|}{|D|} Gini(D_2)$$

其中$D_1 = \{(x, y) \in D|A(x) = a\},\ D_2 = D - D_1$.

5.1 ID3算法

使用信息增益作为选特征的标准，选择信息增益最大的特征作为分裂特征。

5.2 C4.5算法

使用信息增益比作为选特征的标准，选择信息增益比最大的特征作为分裂特征。

5.3 CART算法

分类树：使用基尼指数作为选特征的标准，选择基尼指数最小的特征作为分裂特征。

回归树：一个回归树对应着输入空间的划分以及在划分单元上的输出值。假设有m个空间划分，对应的输出值c_m，则树的模型可以表示为

$$f(x) = \sum_{i=1}^{m} c_i I(x \in R_m) \tag{5.1}$$

当输入空间划分确定时，使用平方误差来$\sum_{x_i \in R_m}(y_i - f(x_i))^2$ 表示回归树对于训练数据的预测误差，使用平方误差最小准则来求解每个单元的最优输出值。易知最优值为对应单元的所有输入数据x_i对应的y_i的均值。

问题是怎样对输入空间划分？选择第j个特征和它的取值s，作为切分特征和切分点。并定义两个区域

$$R_1(j,s) = \{x|x^j \leq s\}和R_2(j,s) = \{x|x^j > s\}$$

然后寻找最优切分特征j和最优切分点s。即求解：

$$\min_{j,s} \Big[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \Big] \tag{5.2}$$

遍历所有输入变量，找到最优的切分变量j，构成一个对(j,s)。依此将输入空间划分为两个区域。接着，对每个区域重复上述划分过程，直到满足停止条件为止。这样就生成一棵回归树。

5.4 剪枝

决策树的剪枝分为预剪枝和后剪枝。

预剪枝：先计算当前的划分是否能带来模型泛化能力的提升，如果不能，则不再继续生长子树。此时可能存在不同类别的样本同时存于结点中，按照多数投票的原则判断该结点所属类别。

停止生长条件：1)树到达一定深度的时候。2)到达当前结点的样本数量小于某个阈值的时候。3)计算每次分裂对测试集的准确度提升，小于某个阈值的时候。

优点：思想直接、算法简单、效率高等特点，适合解决大规模问题。

缺点：树的停止条件不同问题可能很大不同，需要一定的经验；有欠拟合的风险，虽然当前划分会导致测试集准确率降低，但在之后的划分中，准确率可能会有显著上升。

后剪枝：核心思想是让算法生成一棵完全生长的决策树，然后从最底层向上计算是否剪枝。剪枝过程将子树删除，用一个叶子结点替代，该结点的类别同样按照多数投票的原则进行判断。同样地，后剪枝也可以通过在测试集上的准确率进行判断，如果剪枝过后准确率有所提升，则进行剪枝

优缺点：相比于预剪枝，后剪枝方法通常可以得到泛化能力更强的决策树，但时间开销会更大。

6 集成模型

集成学习分为Boosting和Bagging.

6.1 Boosting

训练基分类器时采用串行的方式，各个基分类器之间有依赖。它的基本思路是将基分类器层层叠加，每一层在训练的时候，对前一层基分类器分错的样本，给予更高的权重。

6.2 Bagging

各基分类器之间无强依赖，可以进行并行训练。它更像是一个集体决策的过程，每个个体都进行单独学习，学习的内容可以相同，也可以不同，也可以部分重叠。

6.3 比较

Boosting方法是通过逐步聚焦于基分类器分错的样本，减小集成分类器的偏差。Bagging方法则是采取分而治之的策略，通过对训练样本多次采样，并分别训练出多个不同模型，然后做综合，来减小集成分类器的方差。

1）样本选择上：

Bagging：训练集是在原始集中有放回选取的，从原始集中选出的各轮训练集之间是独立的。

Boosting：每一轮的训练集不变，只是训练集中每个样例在分类器中的权重发生变化。而权值是根据上一轮的分类结果进行调整。

2）样例权重：

Bagging：使用均匀取样，每个样例的权重相等

Boosting：根据错误率不断调整样例的权值，错误率越大则权重越大。

3）预测函数：

Bagging：所有预测函数的权重相等。

Boosting：每个弱分类器都有相应的权重，对于分类误差小的分类器会有更大的权重。

4）并行计算：

Bagging：各个预测函数可以并行生成

Boosting：各个预测函数只能顺序生成，因为后一个模型参数需要前一轮模型的结果。

6.4 XGBoost

XGB原理：基于经验损失函数的负梯度来构造新的决策树并且加入了正则项。

$$L_t = \sum_i l\left(y_i, F_{t-1}\left(x_i\right) + f_t\left(x_i\right)\right) + \Omega\left(f_t\right)$$

其中$F_{t-1}\left(x_i\right)$表示现有的$t-1$棵树最优解。关于树结构的正则项定义为

$$\Omega\left(f_t\right) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

其中T为叶子结点的个数，w_j表示第j个叶子的值，因为XGB基模型为cart回归树，故每个叶子会有一个划分空间的预测值。对该损失函数在F_{t-1}处进行二阶泰勒展开，将l看作关于$F_{t-1}+f_t$的函数,因为y_i 为定值,且将F_{t-1}看作x, f_t看作Δx,可以推导出

$$\begin{aligned} L_t &= \sum_i l\left(y_i, F_{t-1}\left(x_i\right)+f_t\left(x_i\right)\right)+\Omega\left(f_t\right) \\ &\simeq \sum_{i=1}^{n}\left[l\left(y_i, F_{t-1}\right)+\partial_{F_{t-1}} l\left(y_i, F_{t-1}\right) f_t\left(x_i\right)+\frac{1}{2} \partial_{F_{t-1}}^2 l\left(y_i, F_{t-1}\right) f_t^2\left(x_i\right)\right]+\Omega\left(f_t\right) \\ &= \sum_{i=1}^{n}\left[l\left(y_i, F_{t-1}\right)+g_i f_t\left(x_i\right)+\frac{1}{2} h_i f_t^2\left(x_i\right)\right]+\Omega\left(f_t\right) \\ &= \sum_{i=1}^{n}\left[g_i f_t\left(x_i\right)+\frac{1}{2} h_i f_t^2\left(x_i\right)\right]+\Omega\left(f_t\right)+constant \end{aligned} \tag{6.1}$$

设I_j为叶子节点j的样本集合，且$f_t(x_i)=\{q|q_i=w_j\}$则式(6.1)可以重写为

$$L_t=\sum_{j=1}^{T}[(\sum_{i\in I_j} g_i)w_j+\frac{1}{2}(\sum_{i\in I_j} h_i+\lambda)w_j^2]+\gamma T$$

即最小化L_t,将w_j看作自变量，二次函数求解。

$$w_j^*=-\frac{b}{2a}=-\frac{\sum_{i\in I_j} g_i}{\sum_{i\in I_j} h_i+\lambda}$$

$$L_t^*=-\frac{1}{2}\sum_{j=1}^{T}\frac{(\sum_{i\in I_j} g_i)^2}{\sum_{i\in I_j} h_i+\lambda}+\gamma T \tag{6.2}$$

式(6.2)可以评价所建的树q的质量好坏。若能把所有可嫩的树都建出来然后评价它们的好坏，选取损失函数最小的树作为本次的树。但是把所有可能的q枚举出来也是不可能的。所以使用贪婪算法，递归的将分裂节点加入到树中。是否分裂使用式(6.2) 来衡量。即用分裂前的L_t减去分裂后的L_t，如果大于0，说明分裂后损失函数会减小，则分裂，否则不分裂。因为n个叶子节点的树分裂前的叶子个数为n，分裂后的个数为n+1.且对于未分裂的叶子节点，分裂前和分裂后相减会相互抵消。故分裂后叶子节点数为2，分裂前叶子节点数为1.

$$L_{split}=\frac{1}{2}\left[\frac{\left(\sum_{i\in I_L} g_i\right)^2}{\sum_{i\in I_L} h_i+\lambda}+\frac{\left(\sum_{i\in I_R} g_i\right)^2}{\sum_{i\in I_R} h_i+\lambda}-\frac{\left(\sum_{i\in I} g_i\right)^2}{\sum_{i\in I} h_i+\lambda}\right]-\gamma \tag{6.3}$$

处理缺失值：在树的每一个节点加入默认方向，如果有缺失值，则会把样本分到默认方向的子树中。默认方向是从数据中学习而来的。稀疏感知的划分查找算法计算切分后的评判标准时只关注没有缺失的条目.然后再通过将实例以枚举的方式，枚举项为默认分到左分支和默认分到右分支，计算切分前后相差最大的Score，然后将缺失值的实例分到该分支上。

6.5 XGBoost与GBDT的联系和区别

传统GBDT以CART作为基分类器，xgboost还支持线性分类器，这个时候xgboost相当于带L1和L2正则化项的逻辑斯蒂回归（分类问题）或者线性回归（回归问题）。

传统GBDT在优化时只用到一阶导数信息，xgboost则对代价函数进行了二阶泰勒展开，同时用到了一阶和二阶导数。顺便提一下，xgboost工具支持自定义代价函数，只要函数可一阶和二阶求导。

xgboost在代价函数里加入了正则项，用于控制模型的复杂度。正则项里包含了树的叶子节点个数、每个叶子节点上输出的score的L2模的平方和。从Bias-variance tradeoff角度来讲，正则项降低了模型的variance，使学习出来的模型更加简单，防止过拟合，这也是xgboost优于传统GBDT的一个特性。

Shrinkage（缩减），相当于学习速率（xgboost中的eta）。xgboost在进行完一次迭代后，会将叶子节点的权重乘上该系数，主要是为了削弱每棵树的影响，让后面有更大的学习空间。实际应用中，一般把eta设置得小一点，然后迭代次数设置得大一点。（补充：传统GBDT的实现也有学习速率）

列抽样（column subsampling）。xgboost借鉴了随机森林的做法，支持列抽样，不仅能降低过拟合，还能减少计算，这也是xgboost异于传统gbdt的一个特性。对缺失值的处理。对于特征的值有缺失的样本，xgboost可以自动学习出它的分裂方向。

xgboost工具支持并行。boosting不是一种串行的结构吗?怎么并行的？注意xgboost的并行不是tree粒度的并行，xgboost也是一次迭代完才能进行下一次迭代的（第t次迭代的代价函数里包含了前面t-1次迭代的预测值）。xgboost的并行是在特征粒度上的。我们知道，决策树的学习最耗时的一个步骤就是对特征的值进行排序（因为要确定最佳分割点），xgboost在训练之前，预先对数据进行了排序，然后保存为block结构，后面的迭代中重复地使用这个结构，大大减小计算量。这个block结构也使得并行成为了可能，在进行节点的分裂时，需要计算每个特征的增益，最终选增益最大的那个特征去做分裂，那么各个特征的增益计算就可以开多线程进行。

可并行的近似直方图算法。树节点在进行分裂时，我们需要计算每个特征的每个分割点对应的增益，即用贪心法枚举所有可能的分割点。当数据无法一次载入内存或者在分布式情况下，贪心算法效率就会变得很低，所以xgboost还提出了一种可并行的近似直方图算法，用于高效地生成候选的分割点。

7 降维

7.1 主成分分析PCA

信息论中数据的方差越大，则包含的信息越多。因此PCA的目标是最大化投影方差。两种解释，第一种解释是样本点到这个直线的距离足够近，第二种解释是样本点在这个直线上的投影能尽可能的分开。

7.1.1 最大方差形式

$\mathbf{x_n}$在向量$\mathbf{u}$的投影是$\mathbf{u^T x_n}$

数据的均值为:

$$\overline{\mathbf{x}} = \frac{1}{N}\sum_{n=1}^{N}\mathbf{x}_n$$

而均值$\overline{\mathbf{x}}$在向量$\mathbf{u}$的投影是$\mathbf{u}^T\overline{\mathbf{x}}$

故根据定义可得数据点在$\mathbf{u}$方向上的方差:

$$\frac{1}{N}\sum_{n=1}^{N}\left\{\mathbf{u}^{\mathrm{T}}\mathbf{x}_n - \mathbf{u}^{\mathrm{T}}\overline{\mathbf{x}}\right\}^2 = \mathbf{u}^{\mathrm{T}}\frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{x}_n - \overline{\mathbf{x}}\right)\left(\mathbf{x}_n - \overline{\mathbf{x}}\right)^{\mathrm{T}}\mathbf{u} = \mathbf{u^T S u}$$

因此我们要最大化$\mathbf{u}^{\mathrm{T}}\mathbf{S}\mathbf{u}$,为了抑制$\|\mathbf{u}\| \to \infty$，故令$\mathbf{u^T u} = 1$.即

$$\begin{aligned} &\max_{\mathbf{u}}\ \mathbf{u^T S u} \\ &s.t.\ \mathbf{u^T u} = 1 \\ &\mathbf{S} = \frac{1}{N}\sum_{n=1}^{N}\left(\mathbf{x}_n - \overline{\mathbf{x}}\right)\left(\mathbf{x}_n - \overline{\mathbf{x}}\right)^{\mathrm{T}} \end{aligned} \tag{7.1}$$

引入拉格朗日乘子λ_1,构造拉格朗日函数:

$$\mathbf{u}^{\mathrm{T}}\mathbf{S}\mathbf{u} + \lambda_1\left(1 - \mathbf{u}^{\mathrm{T}}\mathbf{u}\right)$$

对$\mathbf{u^T}$求偏导等于0,得到:

$$\mathbf{S}\mathbf{u} = \lambda_1\mathbf{u}$$

可以看出$\mathbf{u}$是矩阵$\mathbf{S}$的特征向量,λ_1是特征向量$\mathbf{u}$对应的特征值。方程两边左乘$\mathbf{u^T}$,则

$$\mathbf{u^T S u} = \lambda_1$$

而$\mathbf{u^T S u}$正是我们要优化的目标函数，所以目标函数取得最大值的条件是特征向量$\mathbf{u}$所对应的特征值λ_1取最大值，即矩阵的SVD分解的前几项。

7.1.2 最小距离形式

对于D维的正交向量$\{\mathbf{u}_i\}$有

$$\mathbf{u}_i^{\mathrm{T}}\mathbf{u}_j = \delta_{ij}$$

对于D维上的每一个去中心化的点$\mathbf{X}_n$，都有

$$\mathbf{x}_n = \sum_{i=1}^{D} \alpha_{ni}\mathbf{u}_i$$

对上式乘以$\mathbf{u}_j$，因为正交向量的性质，可以得到$\alpha_{nj} = \mathbf{x}_n^{\mathrm{T}}\mathbf{u}_j$，所以不失一般性我们可以写

$$\mathbf{x}_n = \sum_{i=1}^{D} \left(\mathbf{x}_n^{\mathrm{T}}\mathbf{u}_i\right)\mathbf{u}_i$$

我们的目标是使用限定数量$d < D$个变量的一种表示方法来近似数据点。假设降维的超平面由d个标准正交基$\boldsymbol{W} = \{\boldsymbol{\omega}_1, \boldsymbol{\omega}_2, \ldots, \boldsymbol{\omega}_d\}$构成，同理可得

$$\widetilde{\boldsymbol{x}_k} = \sum_{i=1}^{d} \left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\boldsymbol{\omega}_i \tag{7.2}$$

根据定义，优化的目标为

$$\begin{aligned} &\arg\min_{\omega_1,\ldots,\omega_d} \sum_{k=1}^{n} \|\boldsymbol{x}_k - \widetilde{\boldsymbol{x}_k}\|_2^2 \\ &\text{s.t.} \quad \boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{\omega}_j = \delta_{ij} = \begin{cases} 1, i = j \\ 0, i \neq j \end{cases} \end{aligned}$$

根据内积的性质，知道$\boldsymbol{x}_k^{\mathrm{T}}\widetilde{\boldsymbol{x}_k} = \widetilde{\boldsymbol{x}_k}^{\mathrm{T}}\boldsymbol{x}_k$ 故

$$\begin{aligned} \|\boldsymbol{x}_k - \widetilde{\boldsymbol{x}}_k\|_2^2 &= \left(\boldsymbol{x}_k - \widetilde{\boldsymbol{x}}_k\right)^{\mathrm{T}}\left(\boldsymbol{x}_k - \widetilde{\boldsymbol{x}}_k\right) \\ &= \boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{x}_k - \boldsymbol{x}_k^{\mathrm{T}}\widetilde{\boldsymbol{x}}_k - \widetilde{\boldsymbol{x}}_k^{\mathrm{T}}\boldsymbol{x}_k + \widetilde{\boldsymbol{x}}_k^{\mathrm{T}}\widetilde{\boldsymbol{x}}_k \\ &= \boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{x}_k - 2\boldsymbol{x}_k^{\mathrm{T}}\widetilde{\boldsymbol{x}}_k + \widetilde{\boldsymbol{x}}_k^{\mathrm{T}}\widetilde{\boldsymbol{x}}_k \end{aligned}$$

上式中的第一项与W的选取无关，是个常数。将式(7.2)代入上式的第二项和第三项可以得到

$$\begin{aligned} \boldsymbol{x}_k^{\mathrm{T}}\widetilde{\boldsymbol{x}_k} &= \boldsymbol{x}_k^{\mathrm{T}}\sum_{i=1}^{d}\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\boldsymbol{\omega}_i \\ &= \sum_{i=1}^{d}\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{\omega}_i \\ &= \sum_{i=1}^{d}\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{\omega}_i \end{aligned}$$

$$\begin{aligned} \widetilde{\boldsymbol{x}}_k^{\mathrm{T}}\widetilde{\boldsymbol{x}_k} &= \left(\sum_{i=1}^{d}\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\boldsymbol{\omega}_i\right)^{\mathrm{T}}\left(\sum_{j=1}^{d}\left(\boldsymbol{\omega}_j^{\mathrm{T}}\boldsymbol{x}_k\right)\boldsymbol{\omega}_j\right) \\ &= \sum_{i=1}^{d}\sum_{j=1}^{d}\left(\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\omega_i\right)^{\mathrm{T}}\left(\left(\boldsymbol{\omega}_j^{\mathrm{T}}\boldsymbol{x}_k\right)\omega_j\right) \end{aligned}$$

注意到，其中$\omega_i^{\mathrm{T}}\boldsymbol{x}_k$和$\omega_j^{\mathrm{T}}\boldsymbol{x}_k$表示投影的长度，都是数字，且当$i \neq j$时，$\omega_i^{\mathrm{T}}\omega_j = 0$，因此

$$\begin{aligned}\widetilde{\boldsymbol{x}}_k^{\mathrm{T}}\widetilde{\boldsymbol{x}_k} &= \sum_{i=1}^{d}\left(\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\boldsymbol{\omega}_i\right)^{\mathrm{T}}\left(\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\boldsymbol{\omega}_i\right) = \sum_{i=1}^{d}\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\\ &= \sum_{i=1}^{d}\left(\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\right)\left(\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{\omega}_i\right) = \sum_{i=1}^{d}\omega_i^{\mathrm{T}}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{\omega}_i\end{aligned}$$

注意到，$\sum\limits_{i=1}^{d}\omega_i^{\mathrm{T}}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{\omega}_i$实际就是矩阵$\boldsymbol{W}^{\mathrm{T}}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{W}$的迹(对角线元素之和),所以

$$\begin{aligned}\|\boldsymbol{x}_k - \widetilde{\boldsymbol{x}_k}\|_2^2 &= -\sum_{i=1}^{d}\boldsymbol{\omega}_i^{\mathrm{T}}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{\omega}_i + \boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{x}_k\\ &= -\operatorname{tr}\left(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{W}\right) + \boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{x}_k\end{aligned}$$

因此优化公式可以写成

$$\begin{aligned}\arg\min_{W}\sum_{k=1}^{n}\|\boldsymbol{x}_k - \widetilde{\boldsymbol{x}_k}\|_2^2 &= \sum_{k=1}^{n}\left(-\operatorname{tr}\left(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{W}\right) + \boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{x}_k\right)\\ &= -\sum_{k=1}^{n}\operatorname{tr}\left(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}}\boldsymbol{W}\right) + C\end{aligned}$$

根据矩阵乘法性质：$\sum\limits_{k}\boldsymbol{x}_k\boldsymbol{x}_k^{\mathrm{T}} = \boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}$,故求解优化问题可以转化为

$$\begin{cases}\arg\max\limits_{W} & \operatorname{tr}\left(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{W}\right)\\ \text{s.t.} & \boldsymbol{W}^{\mathrm{T}}\boldsymbol{W} = I\end{cases}$$

如果我们对$\mathbf{W}$中的d个基$\omega_1, \omega_2, \ldots, \omega_d$依次求解，就会发现和最大方差理论的方法完全等价。

7.2 线性判别分析LDA

中心思想:最大化类间距离和最小化类内距离。

假设样本有两类，两类的均值分别为

$$\boldsymbol{\mu}_1 = \frac{1}{N_1}\sum_{x\in C_1}\boldsymbol{x}, \quad \boldsymbol{\mu}_2 = \frac{1}{N_2}\sum_{x\in C_2}\boldsymbol{x}$$

我们希望投影之后两类间的距离尽可能大，类内距离尽可能小。类间距离表示为

$$\widetilde{\boldsymbol{\mu}_1} = \boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_1, \widetilde{\boldsymbol{\mu}_2} = \boldsymbol{\omega}^{\mathrm{T}}\boldsymbol{\mu}_2$$

类内距离表示投影的方差

$$D_1 = \sum_{x\in C_1}\left(\omega^{\mathrm{T}}x - \omega^{\mathrm{T}}\boldsymbol{\mu}_1\right)^2 = \sum_{x\in C_1}\boldsymbol{\omega}^{\mathrm{T}}\left(x - \boldsymbol{\mu}_1\right)\left(x - \boldsymbol{\mu}_1\right)^{\mathrm{T}}\boldsymbol{\omega}$$

$$D_2 = \sum_{x\in C_2}\omega^{\mathrm{T}}\left(x - \mu_2\right)\left(x - \mu_2\right)^{\mathrm{T}}\omega$$

于是引出我们的目标函数

$$\max_{\omega} J(\omega) = \frac{\left\|\omega^{\mathrm{T}}\left(\mu_1-\mu_2\right)\right\|_2^2}{D_1+D_2} \quad = \frac{\omega^{\mathrm{T}}\left(\mu_1-\mu_2\right)\left(\mu_1-\mu_2\right)^{\mathrm{T}}\omega}{\sum\limits_{x\in C_i}\omega^{\mathrm{T}}\left(x-\mu_i\right)\left(x-\mu_i\right)^{\mathrm{T}}\omega}$$

定义类间散度矩阵$\boldsymbol{S}_B=\left(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2\right)\left(\boldsymbol{\mu}_1-\boldsymbol{\mu}_2\right)^{\mathrm{T}}$, 类内散度矩阵$S_w=\sum_{x\in C_i}\left(x-\mu_i\right)\left(x-\mu_i\right)^{\mathrm{T}}$，则上式可以写成

$$J(\omega)=\frac{\omega^{\mathrm{T}}S_B\omega}{\omega^{\mathrm{T}}S_w\omega}$$

注意到上式的分子与分母都是ω的二次项，因此上式的解与ω的长度无关，只与方向有关。令$\boldsymbol{\omega}^T\boldsymbol{S}_w\boldsymbol{\omega}=1$，则优化目标可以转换成

$$\begin{aligned}&\min_{\omega} \ -\boldsymbol{\omega}^T\boldsymbol{S_B}\boldsymbol{\omega}\\ &s.t.\ \boldsymbol{\omega}^T\boldsymbol{S}_w\boldsymbol{\omega}=1\end{aligned}$$

使用拉格朗日乘子法，可以得到

$$\boldsymbol{S}_B\boldsymbol{\omega}=\lambda\boldsymbol{S}_w\boldsymbol{\omega}$$

整理得

$$\boldsymbol{S}_w^{-1}\boldsymbol{S}_B\boldsymbol{\omega}=\lambda\omega$$

若是多分类，类间散度已经不能按照原来两类的定义了。所以引出了全局散度的定义

$$S_t=\sum_{i=1}^{n}\left(x_i-\mu\right)\left(x_i-\mu\right)^{\mathrm{T}}$$

而类内散度矩阵还是和原来的一样。类间散度矩阵

$$\begin{aligned}\boldsymbol{S}_b&=\boldsymbol{S}_t-\boldsymbol{S}_w\\&=\sum_{i=1}^{n}\left(x_i-\boldsymbol{\mu}\right)\left(x_i-\boldsymbol{\mu}\right)^{\mathrm{T}}-\sum_{x\in C_i}\left(x-\boldsymbol{\mu}_i\right)\left(x-\boldsymbol{\mu}_i\right)^{\mathrm{T}}\\&=\sum_{j=1}^{N}\left(\sum_{x\in C_j}(x-\boldsymbol{\mu})(x-\boldsymbol{\mu})^{\mathrm{T}}-\sum_{x\in C_j}\left(x-\boldsymbol{\mu}_j\right)\left(x-\boldsymbol{\mu}_j\right)^{\mathrm{T}}\right)\\&=\sum_{j=1}^{N}\boldsymbol{m}_j\left(\boldsymbol{\mu}_j-\boldsymbol{\mu}\right)\left(\boldsymbol{\mu}_j-\boldsymbol{\mu}\right)^{\mathrm{T}}\end{aligned}$$

其中$\boldsymbol{m_j}$是第j个类别中的样本个数，N是总的类别个数.

8 EM算法及应用

EM算法是参数估计算法，和极大似然估计(MLE)、最大后验概率(MAP)方法一样。但是EM算法是对模型中含有隐含变量的参数进行估计的，而MLE和MAP方法则是对模型中没有隐含变量进行估计的。

8.1 原理

假设有m个观察样本，模型的参数为θ，模型中的隐含变量为Z，最大化对数似然函数可以写成如下形式

$$\begin{aligned}\ell(\theta) &= \sum_{i=1}^{m} \log p(x;\theta) \\ &= \sum_{i=1}^{m} \log \sum_{z} p(x,z;\theta)\end{aligned}$$

直接对θ, z求导可以看出求导后的形式非常复杂，因为包含和的对数。所以很难求解未知参数θ, z

把上式中分子分母同乘以隐含变量Z的概率分布$Q_i(z^{(i)})$，其概率之和等于1。

$$\begin{aligned}\sum_i \log p\left(x^{(i)};\theta\right) &= \sum_i \log \sum_{z^{(i)}} p\left(x^{(i)},z^{(i)};\theta\right) \\ &= \sum_i \log \sum_{z^{(i)}} Q_i\left(z^{(i)}\right) \frac{p\left(x^{(i)},z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)} \\ &\geq \sum_i \sum_{z^{(i)}} Q_i\left(z^{(i)}\right) \log \frac{p\left(x^{(i)},z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}\end{aligned} \tag{8.1}$$

Jensen不等式表述如下：

如果f是凸函数，X是随机变量，那么：$E[f(X)] \geq f(E[X])$，通俗的说法是函数的期望大于等于期望的函数。

特别地，如果f是严格凸函数，当且仅当$P\left(X = EX\right) = 1$，即X 是常量时，上式取等号，即$E\left[f\left(X\right)\right] = f\left(EX\right)$。

所以结合式(8.1)可以看出，$\sum_{z^{(i)}} Q_i\left(z^{(i)}\right)\left[\frac{p\left(x^{(i)},z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)}\right]$就是$\left[p\left(x^{(i)},z^{(i)};\theta\right)/Q_i\left(z^{(i)}\right)\right]$的期望。

现在式(8.1)的(3)可以很容易的求导了，也可以求出极大值。但是因为有不等号的存在，所以(3)的最大值不是(2)的最大值，怎么办？

上面的式(2)和式(3)不等式可以写成：似然函数$L(\theta) \geq J(z,Q)$，可以通过不断的最大化这个下界J，来使得$L(\theta)$不断提高，最终达到$L(\theta)$的最大值。如图1所示

这里还有两个问题：**1)**什么时候下界$J(z,Q)$与$L(\theta)$在此点θ处相等？**2)**为什么一定会收敛？

先解决第一个问题：

在Jensen不等式中说到，当自变量X是常数的时候，等式成立。换言之，为了让(8.1)式取等号，我们需要

$$\frac{p\left(x^{(i)},z^{(i)};\theta\right)}{Q_i\left(z^{(i)}\right)} = c$$

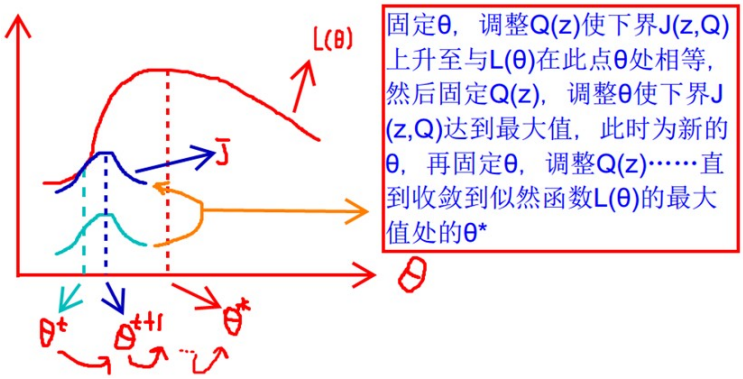

图 1: EM算法解释

其中，c为常数，不依赖于$z^{(i)}$。对该式做个变换，并对所有的z求和，得到

$$\sum_z p\left(x^i, z^{(i)}; \theta\right) = \sum_z Q_i\left(z^{(i)}\right) c$$

因为前面提到$\sum_z Q_i\left(z^{(i)}\right) = 1$，所以可以推导出:

$$\sum_z p\left(x^i, z^{(i)}; \theta\right) = c$$

所以

$$\begin{aligned} Q_i\left(z^{(i)}\right) &= \frac{p\left(x^{(i)}, z^{(i)}; \theta\right)}{\sum_z p\left(x^{(i)}, z; \theta\right)} \\ &= \frac{p\left(x^{(i)}, z^{(i)}; \theta\right)}{p\left(x^{(i)}; \theta\right)} \\ &= p\left(z^{(i)} | x^{(i)}; \theta\right) \end{aligned}$$

至此，我们推出了在固定参数θ后，使下界拉升的Q(z)的计算公式就是条件概率，解决了Q(z)如何选择的问题。这一步就是E步，建立$L(\theta)$的下界。接下来的M步，就是在给定Q(z)后，调整θ，去极大化$L(\theta)$的下界J(z,Q)，毕竟在固定Q(z)后，下界还可以调整的更大。这就是EM算法的步骤。

下面解决第二个问题：假定$\theta^{(t)}$和$\theta^{(t+1)}$是EM第t 次和t+1 次迭代后的结果。如果我们证明了$l(\theta^{(t)}) \leq l(\theta^{(t+1)})$，也就是说极大似然估计单调增加，那么最终会到达最大似然估计的最大值。

下面证明

选定$\theta^{(t)}$后，我们得到E步

$$Q_i^{(t)}(z^{(i)}) := p\left(z^{(i)} | x^{(i)}; \theta^{(t)}\right)$$

这一步保证了在给定$\theta^{(t)}$时，Jensen不等式中的等式成立，也就是

$$\ell\left(\theta^{(t)}\right) = \sum_i \sum_{z^{(i)}} Q_i^{(t)}(z^{(i)}) \log \frac{p\left(x^{(i)}, z^{(i)}; \theta^{(t)}\right)}{Q_i^{(t)}\left(z^{(i)}\right)}$$

然后进行M步，固定$Q_i^{(t)}(z^{(i)})$，并将$\theta^{(t)}$视作变量，对上面的$\ell\left(\theta^{(t)}\right)$求导后，得到$\theta^{(t+1)}$，这样经过一些推

导会有以下式子成立：

$$\ell\left(\theta^{(t+1)}\right) \quad \geq \sum_i \sum_{z^{(i)}} Q_i^{(t)}\left(z^{(i)}\right) \log \frac{p\left(x^{(i)}, z^{(i)}; \theta^{(t+1)}\right)}{Q_i^{(t)}\left(z^{(i)}\right)} \tag{1}$$

$$\geq \sum_i \sum_{z^{(i)}} Q_i^{(t)}\left(z^{(i)}\right) \log \frac{p\left(x^{(i)}, z^{(i)}; \theta^{(t)}\right)}{Q_i^{(t)}\left(z^{(i)}\right)} \tag{2}$$

$$= \ell\left(\theta^{(t)}\right) \tag{3}$$

解释下第(1)步，得到$\theta^{(t+1)}$时，只是最大化$l(\theta^{(t)})$，也就是$l(\theta^{(t+1)})$的下界，而没有使等式成立，等式成立只有是在固定θ，并按E步得到Q_i时才能成立。第(2)步利用了M步的定义，M步就是将$\theta^{(t)}$调整到$\theta^{(t+1)}$，使得下界最大化。因此(2)成立,(3)是之前的等式结果。

实际操作时先求$p\left(z|x;\theta\right)$，再将求得的值代入$\ell\left(\theta\right)$中求得极值。

8.2 应用

主要应用为主题模型pLSA、GMM混合高斯模型、高斯混合据类、HMM隐马尔可夫模型。下面以混合高斯模型为例进行讲解。

假设混合高斯模型由K个高斯模型组成（即数据包含K个类），则GMM的概率密度函数如下

$$p(x) = \sum_{k=1}^{K} p(k)p(x|k) = \sum_{k=1}^{K} \pi_k N\left(x|u_k, \Sigma_k\right)$$

但是问题来了：我们只有样本。不知道样本到底来源于哪一类的高斯模型。那么如何求解样本的生成概率$p(y|t)$

先引入一个隐变量γ。它是一个K维二值随机变量，在它的K维取值中只有某个特定的元素γ_k的取值为1，其它元素的取值为0。实际上，隐变量描述的就是：每一次采样，选择第k个高斯模型的概率，故有：

$$p\left(\gamma_k = 1\right) = \pi_k$$

当给定了γ的一个特定的值之后（也就是知道了这个样本从哪一个高斯模型进行采样），可以得到样本y的条件分布是一个高斯分布，满足：

$$p\left(y|\gamma_k = 1\right) = N\left(y|\mu_k, \Sigma_k\right)$$

而实际上，每个样本到底是从这K个高斯模型中哪个模型进行采样的，是都有可能的。故样本y的概率为：

$$p(y) = \sum_{\gamma} p(\gamma)p(y|\gamma) = \sum_{\mathrm{k}=1}^{K} \pi_k N\left(y|\mu_k, \Sigma_k\right)$$

样本集Y(n个样本点)的联合概率为：

$$L(\mu, \Sigma, \pi) = L\left(y_1, y_2 \ldots y_N; \mu, \Sigma, \pi\right) = \prod_{n=1}^{N} p\left(y_n; \mu, \Sigma, \pi\right) = \prod_{n=1}^{N} \sum_{\mathrm{k}=1}^{K} \pi_k N\left(y_n|\mu_k, \Sigma_k\right)$$

我们现有样本集$Y = (y_1, y_2 \ldots y_T)$,通过隐变量$\gamma_{t,k}$（表示$y_t$这个样本来源于第k个模型）的引入，可以

将数据展开成完全数据：$(y_t, \gamma_{t,1}, \gamma_{t,2} \dots \gamma_{t,K}), t = 1, 2 \dots T$ 有了完全数据概率为

$$\begin{aligned} p(y, \gamma|\mu, \Sigma, \pi) &= \prod_{t=1}^{T} p\left(y_t, \gamma_{t,1}, \gamma_{t,2} \dots \gamma_{t,K}|\mu, \Sigma, \pi\right) \\ &= \prod_{t=1}^{T} \prod_{k=1}^{K} \left(\pi_k N\left(y_t; \mu_k, \Sigma_k\right)\right)^{\gamma_{t,k}} \\ &= \prod_{k=1}^{K} \pi_k^{\sum_{t=1}^{T} \gamma_{t,k}} \prod_{t=1}^{T} \left(N\left(y_t; \mu_k, \Sigma_k\right)\right)^{\gamma t,k} \end{aligned}$$

完全数据的对数似然函数为：

$$\ln p(y, \gamma|\mu, \Sigma, \pi) = \sum_{k=1}^{K} \left(\sum_{t=1}^{T} \gamma_{t,k} \right) \ln \pi_k + \sum_{t=1}^{T} \gamma_{t,k} \left(-\ln(2\pi) - \frac{1}{2} \ln |\Sigma_k| - \frac{1}{2} \left(y_t - \mu_t\right)^T \left(\Sigma_k\right)^{-1} \left(y_t - \mu_t\right) \right)$$

E步：确定Q函数，对带有隐含变量的似然函数求期望，将隐含变量看作变量，其他值看作常数。

M步：将E步计算出的隐含变量的期望值代入到Q函数，令Q函数分别对参数(非隐含变量)求偏导得极值。

如此重复。

9 采样

10 概率图模型