机器学习基本方法

郝哲正

2021 年 10 月 15 日

All models are wrong, but some are useful.

目录

1 回归模型

回归是一种研究自变量和因变量之间关系的数学手段 (有确定因果关系).

当回归函数为参数未知的线性函数时, 称为线性回归模型；当函数为参数未知的非线性函数时, 称为非线性回归模型. 当自变量个数大于 1 时称为多元回归, 当因变量个数大于 1 时称为多重回归

1.1 线性回归 (Linear Regression,LR)

最小二乘法通过最小化误差的平方和寻找数据的最佳函数匹配.

1.1.1 线性回归模型及正规方程

首先考虑线性回归, 对于给定的 N 个样本点 $\left\{x^{(i)}, y^{(i)}\right\}_{i=1}^{N}$, 其中 $x^{(i)} \in \mathbb{R}^n$, 其中 N 为样本数,n 为特征数, 一般情况下我们都考虑 $N > n$ 的情况, 线性回归方程写为:

$$f(x) = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b \tag{1}$$

下文中都以上标表示样本编号, 下标表示特征编号. 令 $w = [w_1\ w_2\ \cdots\ w_n\ b]^T$,$x^{(i)} = [x_1^{(i)}\ x_2^{(i)}\ \cdots\ x_n^{(i)}\ 1]^T$, 所以线性回归方程简化为:

$$f(x) = w^T x \tag{2}$$

根据最小二乘思想, 我们希望找到使误差平方和最小的 w, 即

$$\arg\min_{w} \sum_{i=1}^{N} \left(f\left(x^{(i)}\right) - y^{(i)}\right)^2 \tag{3}$$

为方便求导, 设线性回归的损失函数为

$$L(w) = \frac{1}{2} \sum_{i=1}^{N} \left(f\left(x^{(i)}\right) - y^{(i)}\right)^2 = \frac{1}{2} \sum_{i=1}^{N} \left(w^T x^{(i)} - y^{(i)}\right)^2 \tag{4}$$

令

$$X = \begin{bmatrix} \left(x^{(1)}\right)^T \\ \left(x^{(2)}\right)^T \\ \vdots \\ \left(x^{(N)}\right)^T \end{bmatrix}, Y = \begin{bmatrix} \left(y^{(1)}\right) \\ \left(y^{(2)}\right) \\ \vdots \\ \left(y^{(N)}\right) \end{bmatrix}$$

其中 $X \in \mathbb{R}^{N\times(n+1)}$,$Y \in \mathbb{R}^N$ 由此可将 $L(w)$ 写成矩阵形式:

$$L(w) = \frac{1}{2}(Xw - Y)^T (Xw - Y) \tag{5}$$

问题变成了多元函数求最值问题. 先求 L 的驻点,

$$\begin{aligned} \nabla_w L(w) &= \nabla_w \frac{1}{2}(Xw - Y)^T (Xw - Y) \\ &= \frac{1}{2} \nabla_w \left(w^T X^T X w - w^T X^T Y - Y^T X w + Y^T Y\right) \\ &= \frac{1}{2} \left(2X^T X w - 2X^T Y\right) \\ &= X^T X w - X^T Y \end{aligned} \tag{6}$$

令 $\nabla_w L(w)=0$, 称

$$X^TXw=X^TY \tag{7}$$

为正规方程. 这里我们假设 X^TX 可逆, 而 X^TX 和 X 等秩, 故这也等价于 X 列满秩, 从而求得 L 的驻点:

$$w^*=\left(X^TX\right)^{-1}X^TY \tag{8}$$

为判断 w^* 是不是 L 的极值点, 进一步求 L 关于 w 的 Hessian 矩阵:

$$\begin{aligned}\mathcal{H}&=\nabla_w(\nabla_w L(w))\\&=\nabla_w(X^TXw-X^TY)\\&=X^TX\end{aligned} \tag{9}$$

根据线性代数的内容, 实矩阵 X 列满秩, 则 X^TX 对称正定. 即 $\mathcal{H}$ 正定.

由多元实值函数凹凸性判定定理: 设 $D\subset R^n$ 是非空开凸集, $L:R^n\to R$ 且 $L(w)$ 在 D 上二阶连续可微, 如果 $L(w)$ 的 Hessian 矩阵 $\mathcal{H}$ 在 D 上是正定的, 则 $L(w)$ 是 D 上的严格凸函数.

进而由凸充分性定理: 若 $L:R^n\to R$ 是严格凸函数, 且 $L(w)$ 一阶连续可微, 则 w^* 是全局解的充分必要条件是 $\nabla L\left(w^*\right)=0$.

综上我们得到结论, 在 X 列满秩的前提下,$w^*=\left(X^TX\right)^{-1}X^TY$ 为 (3) 式优化问题的全局最优解. 对给定的 N 个样本点 $\left\{x^{(i)},y^{(i)}\right\}_{i=1}^N$, 可求得线性回归方程 $f(x)=w^{*T}x$, 使其满足在最小二乘意义下最优.

1.1.2 线性回归问题的统计解释

回归, 是研究一组变量与另一组变量之间的关系的方法. 从统计理论上看, 我们做线性回归时, 先验地认为随机变量 x 和 y 在理论上确实满足某种线性关系, 即 $y=\hat{w}^Tx$,x 为多维随机变量. 而实际采样得到的数据点 $\left\{x^{(i)},y^{(i)}\right\}_{i=1}^N$ 不严格满足这种理论关系是因为对随机变量进行采样时存在噪声, 我们根据这种噪声满足的分布选择回归的方式. 对于上述回归问题, 我们假设随机变量 x 和 y 满足线性关系:$y=w^Tx$. 对所有采样得到的数据点 (样本)$\left\{x^{(i)},y^{(i)}\right\}_{i=1}^N$, 满足

$$y^{(i)}=w^Tx^{(i)}+\varepsilon^{(i)},\quad i=1,2,\cdots,N \tag{10}$$

其中 $\varepsilon^{(1)},\varepsilon^{(2)},\cdots,\varepsilon^{(N)}$ 独立同分布.

首先假设噪声满足高斯分布,$\varepsilon^{(i)}\sim\mathcal{N}\left(0,\sigma^2\right)$. 随机变量 $\varepsilon^{(i)}$ 的概率密度函数为:

$$p\left(\varepsilon^{(i)}\right)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{\left(\varepsilon^{(i)}\right)^2}{2\sigma^2}}$$

代入 w,x,y 得

$$p\left(\varepsilon^{(i)}\mid w;x;y\right)=\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{\left(w^Tx-y\right)^2}{2\sigma^2}} \tag{11}$$

对样本 $\left\{x^{(i)},y^{(i)}\right\}_{i=1}^N$ 写出似然函数:

$$\begin{aligned}L(w)&=\prod_{i=1}^N p\left(\varepsilon^{(i)}\mid w;x^{(i)};y^{(i)}\right)\\&=\prod_{i=1}^N\frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{\left(w^Tx^{(i)}-y^{(i)}\right)^2}{2\sigma^2}}\end{aligned} \tag{12}$$

对 $L(w)$ 取对数得到对数似然函数:

$$\begin{aligned}\mathcal{L}(w) &= \log(L(w)) \\ &= \log \prod_{i=1}^{N} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\left(w^T x^{(i)} - y^{(i)}\right)^2}{2\sigma^2}} \\ &= \sum_{i=1}^{N} \log \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{\left(w^T x^{(i)} - y^{(i)}\right)^2}{2\sigma^2}} \\ &= N \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(w^T x^{(i)} - y^{(i)}\right)^2\end{aligned} \tag{13}$$

最大化似然函数:

$$\begin{aligned}\arg\max_w L(w) &\Leftrightarrow \arg\max_w \mathcal{L}(w) \\ &\Leftrightarrow \arg\max_w \left\{ N \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(w^T x^{(i)} - y^{(i)}\right)^2 \right\} \\ &\Leftrightarrow \arg\max_w - \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(w^T x^{(i)} - y^{(i)}\right)^2 \\ &\Leftrightarrow \arg\min_w \frac{1}{2\sigma^2} \sum_{i=1}^{N} \left(w^T x^{(i)} - y^{(i)}\right)^2 \\ &\Leftrightarrow \arg\min_w \sum_{i=1}^{N} \left(w^T x^{(i)} - y^{(i)}\right)^2\end{aligned} \tag{14}$$

这与最小二乘回归的优化函数 (3) 一致. 同理, 若已知噪声服从拉普拉斯分布, 则最大似然法推出的优化函数与最小一乘回归一致.

所以结论就是, 采用何种方法拟合取决于样本的噪声被假定为何种分布, 而由中心极限定理, 无穷多独立同分布的随机变量的和近似服从高斯分布, 故我们一般认为自然情况下噪声服从高斯分布, 所以这时做线性回归应该用最小二乘法. 相比于回归问题的其他损失函数, 最小二乘法中的均方误差损失函数被广泛应用还因为其具有:①连续可微；② 对大误差项惩罚较大 (相比于绝对值误差损失函数)；

1.1.3 多重共线性与带罚项的线性回归

重新考虑正规方程 (7), 还有两个遗留的问题. 首先是 X 不是列满秩的情况, 即 X 列向量组线性相关, 实际情况下有两个主要原因: 一、训练集中存在冗余特征, 此时应该剔除掉多余特征；二、特征过多, 此时应该去掉影响较小的特征 (即当样本总数 N 小于特征数量 $n+1$ 时, 则 X 一定不列满秩,X^TX 一定不可逆). 若 X 不是列满秩, 则正规方程要么解不唯一 (求得的解也不具有实际意义), 要么无解. 此时 X^TX 存在零特征值, 为了消除这些零特征值, 我们可以考虑带扰动的正规方程:

$$\left(X^TX + \lambda I\right) w = X^TY \tag{15}$$

其中 $\lambda > 0$. 对任意实矩阵 X,X^TX 半正定, 故显然 $X^TX + \lambda I$ 正定, 从而有 $w^* = \left(X^TX + \lambda I\right)^{-1} X^TY$. 这是岭回归的解析解, 这种添加扰动的方法和线性回归的正则化是一致的, 我们放到后面一起谈.

事实上, 即使 X 列满秩, 如果 X 的列向量线性相关度较高的话,w 的求解也会十分不稳定. 设 $X = [x_1\ x_2\ \cdots\ x_n\ \varepsilon_1]$, 其中 X 与上文相同,ε_1 为全 1 的 N 维列向量. 举例来说, 若 x_1 可被 x_2 和 x_3 近似地线性表出, 即 $\exists a, b \in R$ 使得

$$x_1 = ax_2 + bx_3 + \xi_{N\times 1} \tag{16}$$

且 $\|\xi\|_2$ 接近于 0, 考虑 X^TX 的行列式,

$$\begin{aligned}
|X^TX| &= \left|\begin{bmatrix} x_1^\top \\ x_2^\top \\ \vdots \\ x_{n+1}^\top \end{bmatrix} \begin{bmatrix} x_1 & x_2 & \cdots & x_{n+1} \end{bmatrix}\right| \\
&= \left|\begin{bmatrix} x_1^Tx_1 & x_1^Tx_2 & \cdots & x_1^Tx_{n+1} \\ x_2^Tx_1 & x_2^Tx_2 & \cdots & x_2^Tx_{n+1} \\ x_3^Tx_1 & x_3^Tx_2 & \cdots & x_3^Tx_{n+1} \\ \cdots & \cdots & & \cdots \\ x_{n+1}^Tx_1 & x_{n+1}^Tx_1 & \cdots & x_{n+1}^Tx_{n+1} \end{bmatrix}\right| \\
&= \left|\begin{bmatrix} (ax_2+bx_3+\xi)^Tx_1 & (ax_2+bx_3+\xi)^Tx_2 & \cdots & (ax_2+bx_3+\xi)^Tx_{n+1} \\ x_2^Tx_1 & x_2^Tx_2 & \cdots & x_2^Tx_{n+1} \\ x_3^Tx_1 & x_3^Tx_2 & \cdots & x_3^Tx_{n+1} \\ \cdots & \cdots & & \cdots \\ x_{n+1}^Tx_1 & x_{n+1}^Tx_1 & \cdots & x_{n+1}^Tx_{n+1} \end{bmatrix}\right| \\
&= \left|\begin{bmatrix} \xi^Tx_1 & \xi^Tx_2 & \cdots & \xi^Tx_{n+1} \\ x_2^Tx_1 & x_2^Tx_2 & \cdots & x_2^Tx_{n+1} \\ x_3^Tx_1 & x_3^Tx_2 & \cdots & x_3^Tx_{n+1} \\ \cdots & \cdots & & \cdots \\ x_{n+1}^Tx_1 & x_{n+1}^Tx_1 & \cdots & x_{n+1}^Tx_{n+1} \end{bmatrix}\right| \approx \left|\begin{bmatrix} 0 & 0 & \cdots & 0 \\ x_2^Tx_1 & x_2^Tx_2 & \cdots & x_2^Tx_{n+1} \\ x_3^Tx_1 & x_3^Tx_2 & \cdots & x_3^Tx_{n+1} \\ \cdots & \cdots & & \cdots \\ x_{n+1}^Tx_1 & x_{n+1}^Tx_1 & \cdots & x_{n+1}^Tx_{n+1} \end{bmatrix}\right| = 0
\end{aligned} \tag{17}$$

我们从两方面来思考这个问题, 从计算数学的角度看,$(X^TX)^{-1}$ 在数值上非常大, 进而解析解 $w^* = \left(X^TX\right)^{-1}X^TY$ 中如果样本矩阵 X 发生微小扰动,w 可能会产生巨大变化. 或者说这时正规方程 (7) 的系数矩阵是病态的 (条件数很大), 方程的解数值不稳定, 非常依赖于样本 X 的选取. 从统计角度看, 待估计的参数 w 的方差和 $\left(X^TX\right)^{-1}$ 也有很大关联. 首先我们证明多元线性回归的无偏性, 即 $E(w^*) = \hat{w}$, 其中 $\hat{w}$ 描述随机变量 x 和 y 之间的关系 $y = \hat{w}^Tx$, 而 w^* 为根据样本 $\left\{x^{(i)}, y^{(i)}\right\}_{i=1}^N$ 估计出来的参数, 在这里看做随机变量. 由矩阵期望的性质

$$\begin{aligned}
E(w^*) &= E\left(\left(X^TX\right)^{-1}X^Ty\right) \\
&= \left(X^TX\right)^{-1}X^TE\left(y\right) \\
&= \left(X^TX\right)^{-1}X^TE\left(X\hat{w}+\varepsilon\right) \\
&= \left(X^TX\right)^{-1}X^TX\hat{w} \\
&= \hat{w}
\end{aligned} \tag{18}$$

其中 ε 为噪声的列向量, 与 (10) 一致. 其次考虑 w^* 的方差, 由矩阵方差的性质可证明 $D(w^*) = \sigma^2(X^TX)^{-1}$,

$$\begin{aligned} D(w^*) &= D\left(\left(X^TX\right)^{-1}X^Ty\right) \\ &= \left(X^TX\right)^{-1}X^TD(y)X\left(X^TX\right)^{-1} \\ &= \left(X^TX\right)^{-1}X^TD\left(X\hat{w}+\varepsilon\right)X\left(X^TX\right)^{-1} \\ &= \left(X^TX\right)^{-1}X^T\sigma^2X\left(X^TX\right)^{-1} \\ &= \sigma^2\left(X^TX\right)^{-1} \end{aligned} \tag{19}$$

所以, 当 $\left(X^TX\right)^{-1}$ 元素数值很大时, 参数估计得到的 w^* 方差很大, 效果不好. 由此我们看到, 即使 X 的列向量在数值上线性无关, 也可能有很高的相关性, 这会使求得的解析解波动较大, 这种情况称为多重共线性, 指线性回归模型中的解释变量之间由于存在精确相关关系或高度相关关系而使模型估计失真或难以估计准确.

为解决多重共线性的问题, 我们使用带罚项的线性回归, 带罚项的方式能同时解决过拟合的问题. 主要方法是岭回归 (Ridge Regression) 和 Lasso 回归. 先直接给出岭回归的优化目标

$$\arg\min_w \sum_{i=1}^{N}\left(f_w\left(x^{(i)}\right)-y^{(i)}\right)^2+\lambda\|w\|_2^2 \tag{20}$$

这实际上是由一个约束问题推出:

$$\begin{aligned} &\min f(w)=\sum_{i=1}^{N}\left(y^{(i)}-w^Tx^{(i)}\right)^2 \\ &\text{s.t. } \|w\|_2^2\le t \end{aligned} \tag{21}$$

在线性回归问题中, 这两个优化问题互为对偶问题, 且满足强对偶性, 是等价问题. 由此给出岭回归的目标函数,

$$L(w,\lambda)=\frac{1}{2}\sum_{i=1}^{N}\left(f_w\left(x^{(i)}\right)-y^{(i)}\right)^2+\frac{\lambda}{2}\|w\|_2^2 \tag{22}$$

求全局最优解的过程和未带罚项的线性回归的求解过程同理, 得到全局最优解:

$$w_R^*(\lambda)=\left(X^TX+\lambda I\right)^{-1}X^TY \tag{23}$$

显然 w^* 和 $w_R^*(\lambda)$ 满足关系:

$$w_R^*(\lambda)=A_\lambda w^*,\ \text{where } A_\lambda=\left(X^TX+\lambda I\right)^{-1}X^TX \tag{24}$$

由此可推出重要结论;

$$\|w_R^*(\lambda)\|<\|w^*\|,\quad \forall\lambda>0 \tag{25}$$

下面给出证明: 首先对 X 做奇异值分解 (SVD),$X_{N\times(n+1)}=U_{N\times N}\Sigma_{N\times(n+1)}V_{(n+1)\times(n+1)}$, 其中 U 和 V 是正交阵, 且 Σ 为对角阵, 对角线上是 X 的奇异值从大到小排列, 分别为 $\sigma_1\ge\sigma_2\ge\cdots\ge\sigma_{n+1}\ge0$($X^TX$ 半正定). 从而有 $X^TX+\lambda I=V\left(\Sigma^T\Sigma+\lambda I\right)V^T$, 进而有:

$$
\begin{aligned}
w_R^*(\lambda) &= A_\lambda w^* \\
&= \left(X^TX+\lambda I\right)^{-1}X^TXw^* \\
&= V\left(\Sigma^T\Sigma+\lambda I\right)^{-1}\Sigma^T\Sigma V^Tw^* \\
&= V\begin{bmatrix} \frac{1}{\sigma_1^2+\lambda} & & & \\ & \frac{1}{\sigma_2^2+\lambda} & & \\ & & \ddots & \\ & & & \frac{1}{\sigma_{n+1}^2+\lambda} \end{bmatrix}\begin{bmatrix} \sigma_1^2 & & & \\ & \sigma_2^2 & & \\ & & \ddots & \\ & & & \sigma_{n+1}^2 \end{bmatrix}V^Tw^* \\
&= V\begin{bmatrix} \frac{\sigma_1^2}{\sigma_1^2+\lambda} & & & \\ & \frac{\sigma_2^2}{\sigma_2^2+\lambda} & & \\ & & \ddots & \\ & & & \frac{\sigma_{n+1}^2}{\sigma_{n+1}^2+\lambda} \end{bmatrix}V^Tw^* \\
&= VS(\sigma_1,\sigma_2,\cdots,\sigma_{n+1},\lambda)V^Tw^*
\end{aligned}
\tag{26}
$$

取范数得:

$$
\begin{aligned}
\|w_R^*(\lambda)\| &= \|VS(\sigma_1,\sigma_2,\cdots,\sigma_{n+1},\lambda)V^Tw^*\| \\
&\le \|V\|\|S(\sigma_1,\sigma_2,\cdots,\sigma_{n+1},\lambda)\|\|V^T\|\|w^*\| \\
&= \|S(\sigma_1,\sigma_2,\cdots,\sigma_{n+1},\lambda)\|\|w^*\| \\
&< \|w^*\|
\end{aligned}
\tag{27}
$$

得证. 称 $S(\sigma_1,\sigma_2,\cdots,\sigma_{n+1},\lambda)$ 为收缩因子. 由 (27) 显然可以看出, 对于给定的样本矩阵 X, 随着 λ 增大,$\|S(\sigma_1,\sigma_2,\cdots,\sigma_{n+1},\lambda)\|$ 不断减小, 也就是 $\|w_R^*(\lambda)\|$ 不断收缩, 满足了正则化的需求. 从另一个角度看, 由 (26), 因为 $g(\sigma_i)=\frac{\sigma_i^2}{\sigma_1^2+\lambda},\sigma_i\ge 0$ 关于 σ_i 为增函数, 也就是说对于 σ_i 较大的特征,λ 增大对其权重 w_i 的收缩影响较小. 对 $w_R^*(\lambda)$ 关于 λ 的曲线的研究称为岭迹分析, 我们可以观察岭迹图观察较佳的 λ 取值, 并通过一些相应的规律来判别数据的共线性 (判断共线性还有一些其他方法), 这里不再赘述.

接下来是 LASSO 回归 (Least Absolute Selection and Shrinkage Operator Regression), 直接给出损失函数, 其罚项为参数向量的 1-范数,

$$
L(w,\lambda)=\frac{1}{2}\sum_{i=1}^{N}\left(f_w\left(x^{(i)}\right)-y^{(i)}\right)^2+\lambda\|w\|_1 \tag{28}
$$

其对应的优化问题为:

$$
\begin{aligned}
&\min f(w)=\sum_{i=1}^{N}\left(y^{(i)}-w^Tx^{(i)}\right)^2 \\
&\text{s.t. } \|w\|_1\le t
\end{aligned}
\tag{29}
$$

统计学中经典的对比图片:

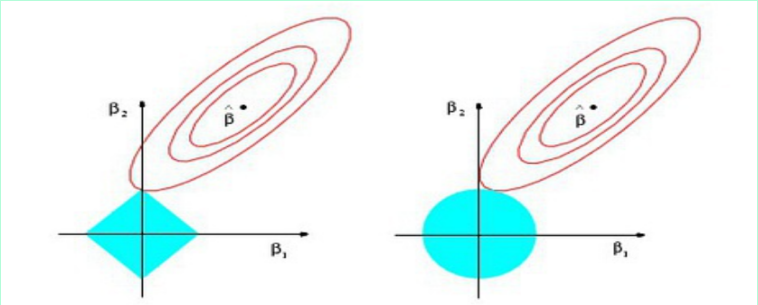


图 1: LASSO 回归与岭回归优化函数投影到二维的对比

由约束问题 (21), 对于给定的 λ, 优化过程不断调整椭圆的大小, 使椭圆面积尽可能小, 且能满足罚项要求 (两个区域有交集). 随着 λ 的增大,LASSO 回归中回归系数往往分别归 0(空间中两个图形在坐标轴处第一次相交), 从而产生稀疏解, 而岭回归中回归系数往往分别缩小后最终同时归 0, 下图展示两者在增大 λ 的过程中 w 的对比:

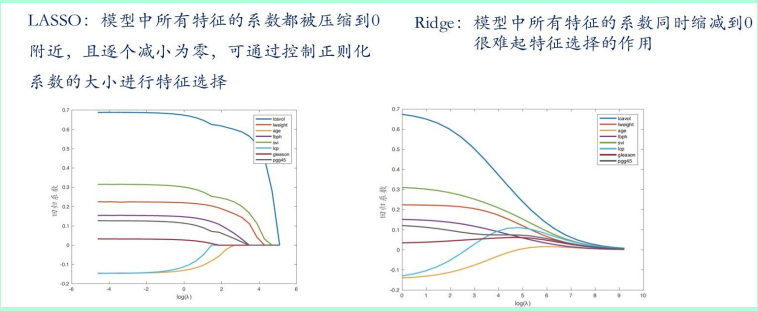


图 2: LASSO 回归与岭回归在二维情况下的对比

这两种方法都可以起到模型正则化的作用, 而岭回归在特征选择上效果一般,LASSO 回归做特征选择可以起到很好的效果. 至于为什么 LASSO 回归可以得到稀疏解? 做如下的解释. 和之前的思路一样, 对目标函数 (28) 求最小值, 但这个目标函数不可导, 我们引入次梯度的概念.

对于凸优化问题:$\min_{x\in C} f(x)$ 其中 $f(x)$ 是一个凸函数, $C = domf$ 是一个凸集. 如果 $f(x)$ 是可微的, 我们知道凸优化问题的解满足:$\nabla f(x^\star) = 0$. 但是如果 $f(x)$ 是不可微, 我们对凸函数引入次梯度 (Subgradient) 的概念, 它是梯度的推广.

如果 $f(x)$ 是可微凸函数, 那么对于任何 $x, y \in C$:

$$f(y) \geq f(x) + \nabla f(x)^T (y - x) \tag{30}$$

次梯度的定义也是来源于此. g 是凸函数 $f(x)$ 在 x_0 点的次梯度当且仅当: $\forall x \in C$ 满足:

$$f(x) \geq f(x_0) + g^T (x - x_0) \tag{31}$$

次梯度满足如下性质:

1. f 在 x_0 处的次梯度总是存在的;

2. 若 f 在 x_0 处可导, 那么在该点的次梯度唯一且等于 $\nabla f(x_0)$.

若 $f(x) = \|x\|_1, x \in \mathbb{R}^n$, 那么 x_0 处的次梯度 g 满足: 如果 $x_i = 0$, 那么 $g_i \in [-1, 1]$; 如果 $x_i \neq 0$, 那么 $g_i = sign(x_i)$. 再定义次微分: f 在 x_0 的次微分是 f 在 x_0 的次梯度取值的集合: $\partial f(x_0) = \{g : g$ 是 f 在 x_0

的次梯度 }. 注意到 $\partial f(x)$ 是一个凸集. 根据定义, 显然有性质: x_0 是凸函数 f 的一个全局最小值点, 当且仅当 $0 \in \partial f\left(x_0\right)$. 从而对于 LASSO 回归, 最优化条件为:

$$\begin{aligned} 0 \in \partial\left(\frac{1}{2}\|Y-Xw\|_2^2+\lambda\|w\|_1\right) &\Leftrightarrow 0 \in -X^T(Y-Xw)+\lambda\partial\|w\|_1 \\ &\Leftrightarrow X^T(Y-Xw) \in \lambda\partial\|w\|_1 \end{aligned} \tag{32}$$

所以次微分优化条件为:

$$\begin{cases} x_i^T(Y-Xw)=\lambda sign\left(w_i\right) & \text{if } w_i>0 \\ \left|x_i^T(Y-Xw)\right| \leq \lambda & \text{if } w_i=0 \end{cases} \tag{33}$$

所以对于 LASSO 回归有如下性质: 如果 $\left|x_i^T(Y-Xw)\right| \leq \lambda$, 那么直接有 $w_i=0$. 这解释了为什么 LASSO 回归可以产生稀疏解.

1.1.4 梯度下降求解线性回归

正规方程 (7) 的另外一个问题是求逆运算可能会消耗很大, 对于特征很多 (n 很大) 的线性回归, 我们也可以采用梯度下降来优化损失函数.

任取回归参数 $w_0 \in \mathbb{R}^{n+1}$, 通过下式迭代, 由于损失函数的凸性, 通过有限步迭代 w 会收敛到全局最优解.

$$w_{i+1}=w_i-\alpha\frac{\partial}{\partial w}L(w), \quad i=0,1,2,\cdots \tag{34}$$

其中 α 为步长 (学习率), 可根据实际情况调整其大小.

	梯度下降法	正规方程法
学习速率 α	需要设置	不需要
计算次数	需要多次迭代	不需要迭代
特征归一化	需要	不需要
时间复杂度	O(kn2)	O(n3)，需要计算 XᵀX 的逆
特征数量	即使 n 很大也可工作	如果 n 很大计算速度慢

图 3: 梯度下降法与正规方程法的对比

在此证明对多元函数 $f\left(x\right)$ 在任意可微点的函数值增长最快的方向是沿梯度方向.

考虑 n 维空间中某个函数上的可微点 $x=(x_1,x_2,\cdots,x_n)$ 的方向导数, 任取单位方向量 α, 用方向余弦表示为 $\alpha=(cos\alpha_1,cos\alpha_2,\cdots,cos\alpha_n), cos^2\alpha_1+cos^2\alpha_2+\cdots+cos^2\alpha_n=1$, 由方向导数的定理,$f$ 在 x 处沿 α 的方向导数为

$$f_\alpha(x)=f_{x_1}(x)cos\alpha_1+f_{x_2}(x)cos\alpha_2+\cdots+f_{x_n}(x)cos\alpha_n=\nabla f\cdot\alpha \tag{35}$$

故增长最快的方向是

$$\max_\alpha = f_\alpha(x) \tag{36}$$

而 $f_\alpha(x)=\nabla f\cdot\alpha=\|\nabla f\|\|\alpha\|cos\theta=\|\nabla f\|cos\theta$,$\theta$ 为 ∇f 和 α 的夹角, 故当 α 与 ∇f 同向时增长最快. 从而多元函数沿负梯度方向下降最快.

此外, 在使用线性回归及其他机器学习模型前, 都不可缺少数据预处理的步骤, 数据标准化、数据离散化、离群值检测等.

至此, 线性回归的内容告一段落.

支持向量回归 SVR 的内容与 SVM 有很多相似之处, 故放到第二章.

2 分类模型

2.1 支持向量机 (Support Vector Machine,SVM)

SVM 一般用于解决二分类问题. 对于全部样本线性可分的情况, 通常采用硬间隔 SVM；对于大部分样本线性可分的情况, 通常采用软间隔 SVM；对于非线性样本, 通常采用带核函数的 SVM.

2.1.1 硬间隔 SVM 的基本形式

首先给出数据集线性可分的定义 (也叫二类线性可分),D 是 d 维欧氏空间中的点集, 如果存在 d 维实向量 w 和实数 b, 且存在 D 的一个子集 D', 使得所有属于 D' 的点 $x^{(i)}$ 都有 $wx^{(i)}+b>0$, 而对于所有属于 $D-D'$ 的点 $x^{(j)}$ 则有 $wx^{(j)}+b<0$, 则称 D (二类) 线性可分.

对于给定样本 $D=\left\{x^{(i)},y^{(i)}\right\}_{i=1}^{N}$, 其中 $x^{(i)}\in\mathbb{R}^d$,$y^{(i)}\in\{-1,1\}$,N 为样本数,d 为特征数, 一般情况下我们都考虑 $N>d$ 的情况. 若 D 线性可分, 我们希望找到一个分隔超平面, 即一组 (w,b), 使得这两组数据可以被完全分开, 由定义这是可行的, 并且显然有无穷多解. 我们一般化的设这个超平面为

$$w^Tx+b=0 \tag{37}$$

其中 $w\in\mathbb{R}^d$,$b\in\mathbb{R}$.

SVM 的思想是希望找到以最大间隔把两类样本分开的超平面 (一组 (w,b)). 把样本中距离超平面最近的点称为支持向量 (显然, 支持向量不唯一, 至少有两个).SVM 的优化目标是使支持向量到超平面的距离最大, 即使到所有点到超平面的最近距离最远. 这个超平面的唯一性可反证得出. 设其中一个支持向量为 x_0(并不是确定的, 而是要在优化过程中寻找), 则 x_0 距超平面的距离为

$$d_0=\frac{|w^Tx_0+b|}{\|w\|_2} \tag{38}$$

所以 SVM 的优化问题即为

$$\arg\max_{w,b} d_0 \tag{39}$$

若令 $(w,b)=(\alpha w,\alpha b)$, 并不影响实际的超平面, 故我们可以取适当的 (w,b) 使得 $\left|w^Tx_0+b\right|=1$ 从而有

$$d_0=\frac{1}{\|w\|_2} \tag{40}$$

同时要保证超平面把数据集 D 分开, 即对 $\forall i\in[N]$, 有

$$y^{(i)}(w^Tx^{(i)}+b)>0,\quad i=1,\cdots,N \tag{41}$$

并且其余数据点到超平面的距离大于 d_0, 即

$$\frac{\left|w^Tx^{(i)}+b\right|}{\|w\|_2}\geqslant d_0,\quad i=1,\cdots N \tag{42}$$

综合上述两个条件, 等价于,

$$\frac{y^{(i)}\left(w^Tx^{(i)}+b\right)}{\|w\|_2}\geqslant d_0,\quad i=1,\cdots N \tag{43}$$

进而等价于

$$y^{(i)}(w^Tx^{(i)}+b)\geqslant 1,\quad i=1,\cdots,N \tag{44}$$

从而优化问题可描述为 (即下面优化的主问题):

$$\begin{aligned} &\min \frac{1}{2}\|w\|_2^2 \\ &\text{s.t. } y^{(i)}\left(w^T x^{(i)}+b\right) \geqslant 1 \quad i=1,\cdots,N \end{aligned} \tag{45}$$

这是一个二次规划问题 (quadratic programming)(目标函数是二次的, 限制条件是一次的). 由于 $\frac{1}{2}\|\alpha w_1+(1-\alpha)w_2\|_2^2 \leqslant \frac{1}{2}\alpha\|w_1\|_2^2+\frac{1}{2}(1-\alpha)\|w_2\|_2^2$, $\alpha\in(0,1)$, 即目标函数和不等式约束条件都是凸的, 故这个优化问题为一个凸二次规划问题, 可应用优化方法通过计算机求解.

2.1.2 硬间隔 SVM 的对偶形

本节将 SVM 的主问题 (45) 转化为对偶问题求解.

对于一般的优化问题 (主问题):

$$\begin{aligned} \min_u \quad & f(u) \\ \text{s.t.} \quad & g_i(u)\leq 0, \quad i=1,2,\ldots,m \\ & h_j(u)=0, \quad j=1,2,\ldots,n \end{aligned} \tag{46}$$

构造拉格朗日函数是求解带约束优化问题的重要方法. 定义其拉格朗日函数为

$$L(u,\alpha,\beta)=f(u)+\sum_{i=1}^{m}\alpha_i g_i(u)+\sum_{j=1}^{n}\beta_j h_j(u) \tag{47}$$

其中 $\alpha_i\geq 0$. 则 (46) 描述的优化问题等价于 (原问题):

$$\begin{aligned} &\min_u\left(\max_{\alpha,\beta} L(u,\alpha,\beta)\right) \\ &\text{s.t. } \alpha_i\geq 0, \quad i=1,2,\ldots,m \end{aligned} \tag{48}$$

证明.

$$\begin{aligned} &\min_u \max_{\alpha,\beta} L(u,\alpha,\beta) \\ =&\min_u\left(f(u)+\max_{\alpha,\beta}\left(\sum_{i=1}^{m}\alpha_i g_i(u)+\sum_{j=1}^{n}\beta_j h_j(u)\right)\right) \\ =&\min_u\left(f(u)+\begin{cases} 0 & \text{若}u\text{ 满足约束;} \\ \infty & \text{否则} \end{cases}\right) \\ =&\min_u f(u),\ \text{且}u\text{ 满足约束,} \end{aligned} \tag{49}$$

其中, 当 g_i 不满足约束时, 即 $g_i(u)>0$, 我们可以取 $\alpha_i=\infty$, 使得 $\alpha_i g_i(u)=\infty$; 当 h_j 不满足约束时, 即 $h_j(u)\neq 0$, 我们可以取 $\beta_j=sign\left(h_j(u)\right)\infty$, 使得 $\beta_j h_j(u)=\infty$. 当 u 满足约束时, 由于 $\alpha_i\geq 0$, $g_i(u)\leq 0$, 则 $\alpha_i g_i(u)\leq 0$. 因此 $\alpha_i g_i(u)$ 最大值为 0 .

(48) 描述的优化问题的对偶问题为 (对偶问题):

$$\begin{aligned} &\max_{\alpha,\beta}\left(\min_u L(u,\alpha,\beta)\right) \\ &\text{s.t. } \alpha_i\geq 0, \quad i=1,2,\ldots,m \end{aligned} \tag{50}$$

对偶问题的最优解是原问题的下界, 即

$$\max_{\alpha,\beta}\min_{u} L(u,\alpha,\beta) \le \min_{u}\max_{\alpha,\beta} L(u,\alpha,\beta)$$

证明. 对任意 (α',β'), $\min_u L(u,\alpha',\beta') \le \min_u \max_{\alpha,\beta} L(u,\alpha,\beta)$. 当 $(\alpha',\beta') = \max_{\alpha',\beta'} \min_u L(u,\alpha',\beta')$ 时, 该式仍然成立, 即 $\max_{\alpha',\beta'} \min_u L(u,\alpha',\beta') \le \min_u \max_{\alpha,\beta} L(u,\alpha,\beta)$.

Slater 条件: 当原问题为凸优化问题, 即 f 和 g_i 为凸函数, h_j 为仿射函数, 且可行域中至少有一点使不等式约束严格成立时, 对偶问题等价于原问题.

KKT 条件:(46) 描述的优化问题在最优值处必须满足如下条件:

1. 主问题可行: $g_i(u) \le 0, h_i(u) = 0$;
2. 对偶问题可行: $\alpha_i \ge 0$;
3. 互补松弛: $\alpha_i g_i(u) = 0$.
4. 梯度为 0:$\nabla_u L(u,\alpha,\beta) = 0$

以上为求解一般凸优化问题的基本理论, 下面我们考虑硬间隔 SVM 的优化.

由 SVM 的主问题 (45), 硬间隔 SVM 的拉格朗日函数写为

$$L(w,b,\alpha) = \frac{1}{2}w^T w + \sum_{i=1}^{N}\alpha_i\left(1 - y^{(i)}\left(w^T x_i + b\right)\right) \tag{51}$$

故优化问题 (45) 等价于 (原问题):

$$\begin{aligned}\min_{w,b}\max_{\alpha} L(w,b,\alpha) = \min_{w,b}\max_{\alpha}\frac{1}{2}w^T w + \sum_{i=1}^{N}\alpha_i\left(1 - y^{(i)}\left(w^T x_i + b\right)\right)\\ \text{s.t. } \alpha_i \ge 0, \quad i = 1,2,\ldots,N\end{aligned} \tag{52}$$

其对偶问题为 (对偶问题):

$$\begin{aligned}\min_{w,b}\max_{\alpha} L(w,b,\alpha) = \max_{\alpha}\min_{w,b}\frac{1}{2}w^T w + \sum_{i=1}^{N}\alpha_i\left(1 - y^{(i)}\left(w^T x_i + b\right)\right)\\ \text{s.t. } \alpha_i \ge 0, \quad i = 1,2,\ldots,N\end{aligned} \tag{53}$$

前文说到 SVM 的主问题是凸问题, 且存在使不等式约束都不取等号地的点, 就是那些非支持向量的点. 由此 SVM 的原问题满足 Slater 条件, 故求解其对偶问题等价于求解原问题.

进而我们开始求解 SVM 的对偶问题 (53), 使用 KKT 条件

1. 主问题可行:$y^{(i)}\left(w^T x^{(i)} + b\right) \ge 1, i = 1,2,\ldots,N$
2. 对偶问题可行:$\alpha_i \ge 0, i = 1,2,\ldots,N$
3. 互补松弛:$\alpha_i\left(1 - y^{(i)}\left(w^T x^{(i)} + b\right)\right) = 0, i = 1,2,\ldots,N$
4. $L(w,b,\alpha)$ 对 w,b 的梯度为 0 ;

$$\begin{aligned}\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{N}\alpha_i y^{(i)} x_i\\ \frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N}\alpha_i y^{(i)} = 0\end{aligned} \tag{54}$$

考虑对偶问题中的 $\min_{w,b} \frac{1}{2}w^Tw + \sum_{i=1}^N \alpha_i \left(1 - y^{(i)} \left(w^T x_i + b\right)\right)$ 是一个无约束优化问题, 将以上两式代入 $L(w,b,\alpha)$:

$$\begin{aligned} L(w,b,\alpha) &= \frac{1}{2}w^Tw + \sum_{i=1}^N \alpha_i \left(1 - y^{(i)} \left(w^T x^{(i)} + b\right)\right) \\ &= \frac{1}{2}w^T \sum_{i=1}^N \alpha_i y^{(i)} x^{(i)} + \sum_{i=1}^N \alpha_i - w^T \sum_{i=1}^N \alpha_i y^{(i)} x^{(i)} - b\sum_{i=1}^N \alpha_i y^{(i)} \\ &= \sum_{i=1}^N \alpha_i - \frac{1}{2}w^T \sum_{i=1}^N \alpha_i y^{(i)} x^{(i)} - b * 0 \\ &= \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \alpha_i y^{(i)} x^{(i)} \sum_{j=1}^N \alpha_j y^{(j)} x^{(j)} \\ &= \sum_{i=1}^N \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} {x^{(i)}}^T x^{(j)} \end{aligned} \tag{55}$$

故对偶问题转化后的求解目标:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^n \alpha_i - \frac{1}{2}\sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y^{(i)} y^{(j)} {x^{(i)}}^T x^{(j)} \\ \text{s.t.} \quad & \alpha_i \geq 0; \\ & \sum_{i=1}^N \alpha_i y^{(i)} = 0 \end{aligned} \tag{56}$$

转化后的优化问题中唯一的变量是 α, 而 w, b 已经被约去了, 有利于求解. 通过求解这个凸优化问题得到 α, 再由 $w = \sum_{i=1}^N \alpha_i y^{(i)} x_i$ 求得 w, 进而由 KKT 条件互不松弛性的结果 $\alpha_i \left(1 - y^{(i)} \left(w^T x^{(i)} + b\right)\right) = 0, i = 1, 2, \ldots, N$ 求得 b, 由此得到决策平面.

由 SVM 的 KKT 条件可知, $\alpha_i(1- \ y^{(i)} \left(w^T x_i + b\right)) = 0$. 当 $\alpha_i > 0$ 时, $1 - y^{(i)} \left(w^T x_i + b\right) = 0$. 即 $y^{(i)} \left(w^T x_i + b\right) = 1$. 所以支持向量是对偶变量 $\alpha_i > 0$ 对应的样本

另外, 由 $w = \sum_{i=1}^N \alpha_i y^{(i)} x_i$ 和 $\alpha_i(1- \ y^{(i)} \left(w^T x_i + b\right)) = 0$, 我们得到 (w,b) 仅由支持向量决定, 与其他样本无关. 实践中, 为了得到对 b 更稳健的估计, 通常使用对所有支持向量求解得到 b 的平均值.

至于为什么要把 SVM 的优化问题转化为对偶问题求解?

一个是因为主问题依赖于 w 的维数, 而对偶问题需要求解的 α 只有支持向量非零, 且有 SMO 等快速算法, 一般来说会比求解主问题或者求解原问题更简便.

另外, 更重要的一点是方便引入核函数.

2.1.3 软间隔 SVM 与合页损失函数

现实中数据集完全线性可分的情况较少, 对于大部分数据线性可分的情况, 我们允许个别样本出现在一个软间隔内, 对硬间隔 SVM 的优化问题加入松弛变量, 得到软间隔 SVM 的优化问题

$$\begin{aligned} &\min_{w,b,\xi} \frac{1}{2}\|w\|_2^2 + C\sum_{i=1}^{N}\xi_i^2 \quad (C>0) \\ &\text{s.t.} \quad \begin{cases} y^{(i)}\left(w^T x^{(i)} + b\right) \geqslant 1-\xi_i \\ \xi_i \geqslant 0 \end{cases} \quad i = 1, \cdots, N \end{aligned} \tag{57}$$

C 是个可调节参数用于权衡优化间隔和少量样本违背大间隔约束这两个目标. 当 C 比较大时, 我们希望更多的样本满足大间隔约束; 当 C 比较小时, 我们允许有一些样本不满足大间隔约束.

如图,

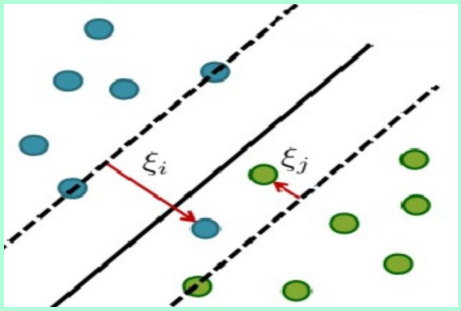


图 4: 软间隔 SVM

软间隔支持向量机的拉格朗日函数为

$$L(w,b,\xi,\alpha,\beta) := \frac{1}{2}w^T w + C\sum_{i=1}^{m}\xi_i + \sum_{i=1}^{m}\alpha_i\left(1-\xi_i - y^{(i)}\left(w^T\phi\left(x^{(i)}\right)+b\right)\right) + \sum_{i=1}^{m}\beta_i\left(-\xi_i\right) \tag{58}$$

其对偶问题为

$$\begin{aligned} \max_{\alpha,\beta}\min_{w,b,\xi} \quad & L(w,b,\xi,\alpha,\beta) \\ \text{s.t.} \quad & \alpha_i \geq 0, \quad i = 1,2,\ldots,N \\ & \beta_i \geq 0, \quad i = 1,2,\ldots,N \end{aligned} \tag{59}$$

因为内层对 (w,b,ξ) 的优化属于无约束优化问题, 我们可以通过令偏导等于零的方法得到 (w,b,ξ) 的最优值.

$$\begin{aligned} \frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^{m}\alpha_i y^{(i)} x^{(i)} \\ \frac{\partial L}{\partial b} = 0 \Rightarrow \sum_{i=1}^{m}\alpha_i y^{(i)} = 0 \\ \frac{\partial L}{\partial \xi} = 0 \Rightarrow \alpha_i + \beta_i = C \end{aligned}$$

因为存在约束 $\beta_i = C - \alpha_i \geq 0$, 不失一般性, 我们可以约束 $0 \leq \alpha_i \leq C$, 从而去掉变量 β_i. 将其代入 (58) , 消

去 (w,b,ξ,β), 和硬间隔 SVM 类似, 即得只含 α 的对偶问题

$$\begin{aligned}\min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y^{(i)}y^{(j)}{x^{(i)}}^T x^{(i)} - \sum_{i=1}^{m}\alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^{m}\alpha_i y^{(i)} = 0 \\ & 0 \le \alpha_i \le C, \quad i = 1,2,\ldots,N\end{aligned} \tag{60}$$

同样可由 SMO 等算法求得最优解 α. 若求得的 α 中存在一个分量 $\alpha_j < 0$ 则满足 Slater 条件, 则对偶问题的最优解也是原问题的最优解, 故也等价于主问题的最优解. 进而可由求得的 α 通过 KKT 条件求 (w,b).

优化问题的最优解满足软间隔 SVM 的 KKT 条件:

1. 主问题可行: $1-\xi_i - y^{(i)}\left(w^T\phi\left(x^{(i)}\right)+b\right) \le 0, -\xi_i \le 0$;

2. 对偶问题可行: $\alpha_i \ge 0, \beta_i \ge 0$;

3. 互补松弛: $\alpha_i\left(1-\xi_i - y^{(i)}\left(w^T\phi\left(x^{(i)}\right)+b\right)\right) = 0, \beta_i\xi_i = 0$.

4.L 关于 w,b,ξ 的梯度为 0

$$\begin{aligned}\nabla_w L &= w^* - \sum_{i=1}^{N}\alpha_i^* y^{(i)}x^{(i)} = 0 \\ \nabla_b L &= -\sum_{i=1}^{N}\alpha_i^* y^{(i)} = 0 \\ \nabla_\xi L &= C - \alpha^* - \mu^* = 0\end{aligned} \tag{61}$$

由上述 KKT 条件得到 (w,b) 为

$$\begin{aligned}w^* &= \sum_{i=1}^{N}\alpha_i^* y^{(i)}x^{(i)} \\ b^* &= y^{(j)} - \sum_{i=1}^{N} y^{(i)}\alpha_i^*\ {x^{(i)}}^T x^{(j)}\end{aligned} \tag{62}$$

进而得到决策平面. 这里同样可以使用对所有支持向量求解得到 b 的平均值.

软间隔 SVM 还有另一种解释, 其基本型为一个合页损失函数

$$\min_{w,b}\frac{1}{m}\sum_{i=1}^{m}\max\left(0, 1-y^{(i)}\left(w^T\left(x^{(i)}\right)+b\right)\right) + \frac{\lambda}{2}\|w\|^2 \tag{63}$$

其中, 第一项是经验风险, 度量模型对训练数据的拟合程度. 当样本被正确分类时,$y^{(i)}\left(w^T\left(x^{(i)}\right)+b\right) > 1$ 对应的损失就为 0, 当样本被错误分类时, 损失就为 $1-y^{(i)}\left(w^T\left(x^{(i)}\right)+b\right)$. 第二项是结构风险, 也称为正则化项, 度量模型自身的复杂度. 正则化项削减了假设空间, 从而降低过拟合风险. λ 是个可调节的超参数, 用于权衡经验风险和结构风险.

证明: 设 $\xi_i = \max\left(0, 1-y^{(i)}\left(w^T\left(x^{(i)}\right)+b\right)\right)$, 则软间隔 SVM 的主问题 (58) 的约束条件自然满足, 再令 $\lambda = \frac{1}{mC}$, 从而合页损失函数等价于 (58).

2.1.4 核方法

核方法用来解决数据集线性不可分的情况. 所谓核方法, 就是我们采用一种映射, 将原来的输入空间通过变换映射到另一个特征空间, 经过变换, 就使原来线性不可分的点转换成线性可分的点.

Vapnik 证明了: 数据集在 $\mathbb{R}^d$ 中线性不可分, 当 d 有限时, 一定存在 $\tilde{d}$, 使得样本在空间 $\mathbb{R}^{\tilde{d}}$ 中线性可分.

核技巧在 SVM 中的应用就是通过一个非线性变换将输入空间 ($\mathbb{R}^d$ 或一个离散空间) 对应于一个特征空间 (希尔伯特空间 H), 使输入空间中用于判别的超曲面在特征空间中是超平面, 这样就可以在特征空间中使用 SVM 进行判别.

设 X 是输入空间 (欧式空间 $\mathbb{R}^n$ 的子集或离散集合), H 为特征空间 (希尔伯特空间), 如果存在一个从 X 到 H 的映射

$$\phi(x) : X \to H$$

使得对所有 $x, y \in X$, 函数 $K(x, y)$ 满足条件

$$K(x, y) = \langle \phi(x), \phi(y) \rangle$$

则称 $K(x, y)$ 为核函数, $\phi(x)$ 为映射函数. 由核函数的定义, 确定一个核函数需要给出一个映射 ϕ 和一个希尔伯特空间 (完备的内积空间) H : 已知映射函数 ϕ , 求其在希尔伯特空间 H 的内积可得核函数. 设 $X \subseteq \mathbb{R}^n, K(x, y)$ 是定义在 $X \times X$ 上的对称函数, 如果对任意 $x^{(i)} \in X, i = 1, 2, \ldots, m, K(x, y)$ 对应的 Gram 矩阵 $K = \left[K\left(x^{(i)}, x^{(j)}\right)\right]_{m \times m}$ 是半正定矩阵, 则称 $K(x, y)$ 是正定核, 即一般说的核函数.

假设 $K(x, y)$ 是定义在 $X \times X$ 上的对称函数, 并且对任意 $x^{(i)} \in X, i = 1, 2, \ldots, m, K(x, y)$ 对应的 Gram 矩阵是半正定的, 定义映射 $\phi : x \to K(\cdot, x)$, 接下来将 ϕ 的像空间完备化为一个希尔伯特空间: 定义集合

$$S = \left\{ f(\cdot) = \sum_{i=1}^{m} \alpha_i K\left(\cdot, x^{(i)}\right) \mid \forall x^{(i)} \in X, \alpha_i \in \mathbb{R}, i = 1, 2, \ldots, m \right\} \tag{64}$$

由于集合 S 对加法和数乘运算是封闭的, 所以 S 构成一个向量空间. 在 S 上定义内积运算 $\langle \cdot, \cdot \rangle$ (非负, 对称, 线性) : $\forall f, g \in S$ $f(\cdot) = \sum_{i=1}^{m} \alpha_i K\left(\cdot, x^{(i)}\right)$, $g(\cdot) = \sum_{j=1}^{l} \beta_j K\left(\cdot, y^{(j)}\right)$

$$\langle f, g \rangle = \sum_{i=1}^{m} \sum_{j=1}^{l} \alpha_i \beta_j K\left(x^{(i)}, y^{(j)}\right) \tag{65}$$

定义了内积运算的向量空间 S 为内积空间. 由内积诱导的范数: $\|f\| = \sqrt{\langle f, f \rangle}$ 则 S 是一个赋范向量空间. 根据泛函分析的内容, 对于不完备的赋范向量空间 S, 一定可以使之完备化, 得到完备的赋范向量空间 H 一个内积空间, 即希尔伯特空间. 这样, 对上述给定的满足对称性、正定性的函数 $K(x, y)$, 可以构造由 X 到希尔伯特空间 H 的映射: $\phi : x \to K(\cdot, x)$ 由内积的定义

$$\langle K(\cdot, x), f(\cdot) \rangle = \sum_{i=1}^{m} \alpha_i K\left(x, x^{(i)}\right) = f(x) \tag{66}$$

$$\langle \phi(x), \phi(y) \rangle = \langle K(\cdot, x), K(\cdot, y) \rangle = K(x, y) \tag{67}$$

说明 K 为核函数.

另外, 对于给定的核函数 K, 特征空间 H 和映射函数 ϕ 的取法不唯一.

现在对 SVM 使用核技巧, 通过 ϕ 将数据集映射到 H. 得到主问题为:

$$\begin{aligned} \min_{w,b} \quad & \frac{1}{2} w^T w \\ \text{s.t.} \quad & y^{(i)}\left(w^T \phi\left(x_i\right) + b\right) \geq 1, \quad i = 1, 2, \ldots, m \end{aligned} \tag{68}$$

由 (56) 得到转化后的对偶问题

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y^{(i)}y^{(j)}\phi\left(x_i\right)^T\phi\left(x_j\right)-\sum_{i=1}^{N}\alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^{N}\alpha_i y^{(i)}=0, \\ & \alpha_i\geq 0, \quad i=1,2,\ldots,N. \end{aligned} \tag{69}$$

由优化函数的内积形式引入核函数 K

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y^{(i)}y^{(j)}K(x^{(i)},x^{(j)})-\sum_{i=1}^{N}\alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^{N}\alpha_i y^{(i)}=0, \\ & \alpha_i\geq 0, \quad i=1,2,\ldots,N. \end{aligned} \tag{70}$$

由此通过核方法将线性可分的 SVM 和线性不可分的 SVM 统一起来, 都可以硬间隔划分.

常用的核函数有

名称	形式	优点	缺点
线性核	$x_i^Tx_j$	有高效实现, 不易过拟合	无法解决非线性可分问题
多项式核	$\left(\beta x_i^Tx_j+\theta\right)^n$	比线性核更一般, n 直接描述了被映射空间的复杂度	参数多, 当n 很大时会导致计算不稳
高斯核	$\exp\left(-\frac{\|x_i-x_j\|^2}{2\sigma^2}\right)$	只有一个参数, 没有计算不稳定问题	计算慢, 过拟合风险大

除此之外, 我们还可以根据需要自定义核函数, 但需要满足 Mercer 条件.

Mercer 条件:$K\left(x_i,x_j\right)$ 是核函数的充要条件是其对应的矩阵 $\left[K\left(x_i,x_j\right)\right]_{m\times m}$ 是半正定的.

新的核函数还可以通过现有核函数的组合得到. 使用多个核函数的凸组合是多核学习的研究内容.

SVM 一般用于二分类, 也可以使用成对分类法 (pairwise classification) 和一类对余类 (one-against-the-rest) 等方法进行多分类.

SVM 拥有漂亮的数学推导和实际结论, 被广泛应用.《A Tutorial on Support Vector Machines for Pattern Recognition》给出的 Bunch-Kaufman 训练算法的训练计算复杂度在 $O(Nsv^3+LNsv^2+d*L*Nsv)$ 和 $O(d*L^2)$ 之间, 其中 Nsv 是支持向量的个数,L 是训练集样本的个数,d 是每个样本的维数 (原始的维数, 没有经过向高维空间映射之前的维数). 复杂度会有变化, 是因为它不光跟输入问题的规模有关（不光和样本的数量, 维数有关）, 也和问题最终的解有关（即支持向量有关）, 如果支持向量比较少, 过程会快很多, 如果支持向量很多, 接近于样本的数量, 就会产生 O(dL2) 这个十分糟糕的结果. 总的来讲,SVM 的 SMO 算法根据不同的应用场景, 其算法复杂度为 $O(N)$ 到 $O(N^{2.2})$ 之间

据原著论文的测试, 使用 SMO 算法, 在线性 SVM 上快 1000 倍, 在非线性上快 15 倍. 对于 SVM 的 SMO 算法的内存需求时线性的, 这使得其能适用比较大的训练集.

SVM 允许决策边界很复杂, 即使数据只有几个特征. 它在低维数据和高维数据（即很少特征和很多特征）上都表现都很好, 但对样本个数的缩放表现不好. 在有多达 10000 个样本的数据上运行 SVM 可能表现良好, 但如果数据量达到 100000 甚至更大, 在运行时间和内存使用方面可能会面临挑战. 所以, 如果数据量很大,SVM 的训练时间就会比较长, 如垃圾邮件的分类检测, 没有使用 SVM 分类器, 而是使用了简单的 naive bayes 分类器, 或者是使用 logistic regression 模型分类.

此外, 经典的支持向量机算法只给出了二类分类的算法, 而在数据挖掘的实际应用中, 一般要解决多类的分类问题. 可以通过多个二类支持向量机的组合来解决. 但在类别较多时训练复杂度较大, 由此有提出了一系列组合模式来克服 SVM 的这一问题.

2.1.5 支持向量回归 (Support Vector Regression,SVR)

SVR 是一种" 宽松" 的线性回归, 对于上面的线性回归而言, 一个样本只要不算正好落在作为模型的线性函数上, 就要被计算损失.SVR 在超平面的两侧制造了一个" 间隔带", 对于所有落入到间隔带内的样本, 都不计算损失；只有间隔带之外的 (即下图中带红点的样本), 才计入损失函数.

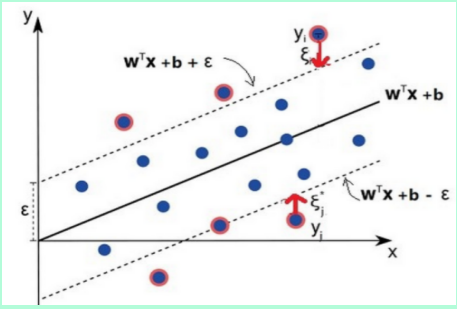


图 5: SVR 原理

$D = \left\{x^{(i)}, y^{(i)}\right\}_{i=1}^{N}$, 其中 $x^{(i)} \in \mathbb{R}^d$,$y^{(i)} \in \mathbb{R}$,N 为样本数,d 为特征数, 一般情况下我们都考虑 $N > d$ 的情况. 我们希望找到一个超平面

$$w^T x + b = 0, \quad w \in \mathbb{R}^d, b \in \mathbb{R} \tag{71}$$

对于 (一般按经验) 给定的容忍度 ε, 能使样本点尽可能落在 $y^{(i)} \in [w^T x^{(i)} + b - \varepsilon, w^T x^{(i)} + b + \varepsilon]$ 的间隔带之内. 依此来构造损失函数. 达到这一目的可以通过增大间隔带的宽度和对间隔带外的样本点施加损失.

超平面 $w^T x + b - \varepsilon = 0$ 到超平面 $w^T x + b + \varepsilon = 0$ 的距离为

$$d = 2\frac{\varepsilon}{\|w\|_2} \tag{72}$$

对于给定的 ε, 我们可以通过最小化 $\|w\|_2$ 来最大化间隔带宽度 d.

另外对于间隔带之外的样本点施加损失, 设 ξ_i 为 $w^Tx+b-\varepsilon=0$ 以下的点的损失,$\hat{\xi}_i$ 为 $w^Tx+b+\varepsilon=0$ 以上的点的损失由此得到定义

$$\begin{aligned}\xi_i &= \begin{cases}\left(w^Tx^{(i)}+b-\varepsilon\right)-y^{(i)}, & \text{if } y^{(i)}<w^Tx^{(i)}+b-\varepsilon\\ 0, & \text{otherwise}\end{cases}\\ \hat{\xi}_i &= \begin{cases}y^{(i)}-\left(w^Tx^{(i)}+b+\varepsilon\right), & \text{if } y^{(i)}>w^Tx^{(i)}+b+\varepsilon\\ 0, & \text{otherwise}\end{cases}\end{aligned} \tag{73}$$

由以上两个方面构造损失函数

$$\begin{aligned}\min_{w,b,\xi_i,\hat{\xi}_i}\quad & \frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\left(\xi_i+\hat{\xi}_i\right)\\ \text{s.t.}\quad & (w^Tx^{(i)}+b)-y^{(i)}\leqslant\varepsilon+\xi_i\\ & y^{(i)}-(w^Tx^{(i)}+b)\leqslant\varepsilon+\hat{\xi}_i\\ & \xi_i\geqslant 0,\hat{\xi}_i\geqslant 0,i=1,2,\ldots,N\end{aligned} \tag{74}$$

引入拉格朗日乘子, 构造拉格朗日函数:

$$\begin{aligned}L(w,b,\xi,\hat{\xi},\alpha,\hat{\alpha},\mu,\hat{\mu}) &= \frac{1}{2}\|w\|^2+C\sum_{i=1}^{m}\left(\xi_i+\hat{\xi}_i\right)-\sum_{i=1}^{m}\mu_i\xi_i-\sum_{i=1}^{m}\hat{\mu}_i\hat{\xi}_i\\ &+\sum_{i=1}^{m}\alpha_i(\left(w^Tx^{(i)}+b\right)-y^{(i)}-\varepsilon-\xi_i)+\sum_{i=1}^{m}\hat{\alpha}_i\left(y^{(i)}-(w^Tx^{(i)}+b)-\varepsilon-\hat{\xi}_i\right)\end{aligned}$$

分别对 $w,b,\xi,\hat{\xi}$ 求偏导, 令偏导数为 0, 得到

$$\begin{aligned}w &= \sum_{i=1}^{m}\left(\hat{\alpha}_i-\alpha_i\right)x^{(i)}\\ 0 &= \sum_{i=1}^{m}\left(\hat{\alpha}_i-\alpha_i\right)\\ C &= \alpha_i+\mu_i\\ C &= \hat{\alpha}_i+\hat{\mu}_i\end{aligned} \tag{75}$$

把上边的式子带入 L, 即可求得 SVR 的对偶问题

$$\begin{aligned}\max_{\alpha,\hat{\alpha}}\ & \sum_{i=1}^{m}y^{(i)}\left(\hat{\alpha}_i-\alpha_i\right)-\varepsilon\left(\hat{\alpha}_i+\alpha_i\right)-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\left(\hat{\alpha}_i-\alpha_i\right)\left(\hat{\alpha}_j-\alpha_j\right){x^{(i)}}^T x^{(j)}\\ \text{s.t.}\ & \sum_{i=1}^{m}\left(\hat{\alpha}_i-\alpha_i\right)=0\\ & 0\leqslant\alpha_i,\hat{\alpha}_i\leqslant C\end{aligned} \tag{76}$$

我们先假设会有间隔带之外的样本点 (这也是合理, 否则是因为给定的 ε 不合适), 则该问题满足 Slater 条件, 从而求解其对偶问题等价于求解原问题.

该问题的最优解应满足 KKT 条件, 其中互不松弛性的条件为:

$$\begin{cases} \alpha_i \left((w^T x^{(i)} + b) - y^{(i)} - \varepsilon - \xi_i \right) = 0 \\ \hat{\alpha}_i \left(y^{(i)} - (w^T x^{(i)} + b) - \varepsilon - \hat{\xi}_i \right) = 0 \\ -\mu_i \xi_i = 0 \Rightarrow \mu_i \xi_i = 0 \\ -\hat{\mu}_i \hat{\xi}_i = 0 \Rightarrow \hat{\mu}_i \hat{\xi}_i = 0 \end{cases} \tag{77}$$

由 (75) 的后两个结果可知:

$$\begin{aligned} \mu_i &= C - \alpha_i \\ \hat{\mu}_i &= C - \hat{\alpha}_i \end{aligned} \tag{78}$$

所以上述 KKT 条件可以进一步变形为:

$$\begin{cases} \alpha_i \left((w^T x^{(i)} + b) - y^{(i)} - \varepsilon - \xi_i \right) = 0 \\ \hat{\alpha}_i \left(y^{(i)} - (w^T x^{(i)} + b) - \varepsilon - \hat{\xi}_i \right) = 0 \\ (C - \alpha_i) \xi_i = 0 \\ (C - \hat{\alpha}_i) \hat{\xi}_i = 0 \end{cases} \tag{79}$$

又因为样本 $\left(x^{(i)}, y^{(i)}\right)$ 只可能处在间隔带的某一侧, 那么约束条件 $(w^T x^{(i)} + b) - y^{(i)} - \varepsilon - \xi_i = 0$ 和 $y^{(i)} - (w^T x^{(i)} + b) - \varepsilon - \hat{\xi}_i = 0$ 不可能同时成立, 所以 α_i 和 $\hat{\alpha}_i$ 中至少有一个为 0 , 也即 $\alpha_i \hat{\alpha}_i = 0$. 在此基础上再进一步分析可知, 如果 $\alpha_i = 0$ 的话, 那么根据约束 $(C - \alpha_i) \xi_i = 0$ 可知此时 $\xi_i = 0$, 同理, 如果 $\hat{\alpha}_i = 0$ 的话, 那么根据约束 $(C - \hat{\alpha}_i) \hat{\xi}_i = 0$ 可知此时 $\hat{\xi}_i = 0$, 所以 ξ_i 和 $\hat{\xi}_i$ 中也是至少有一个为 0 , 也即 $\xi_i \hat{\xi}_i = 0$. 将 $\alpha_i \hat{\alpha}_i = 0, \xi_i \hat{\xi}_i = 0$. 整合上述分析得到转化后的 KKT 条件

$$\begin{cases} \alpha_i \left((w^T x^{(i)} + b) - y^{(i)} - \varepsilon - \xi_i \right) = 0 \\ \hat{\alpha}_i \left(y^{(i)} - (w^T x^{(i)} + b) - \varepsilon - \hat{\xi}_i \right) = 0 \\ \alpha_i \hat{\alpha}_i = 0, \xi_i \hat{\xi}_i = 0 \\ (C - \alpha_i) \xi_i = 0, (C - \hat{\alpha}_i) \hat{\xi}_i = 0 \end{cases} \tag{80}$$

由此可以看出, 当且仅当 $y^{(i)} = w^T x^{(i)} + b - \varepsilon - \xi$ 时,α_i 非零; 当且仅当 $y^{(i)} = w^T x^{(i)} + b + \varepsilon + \xi$ 时,$\hat{\alpha}_i$ 非零. 也就是说当且仅当样本点在间隔带之外时,α_i 和 $\hat{\alpha}_i$ 中才有一个为 0 , 我们可以姑且认为这些间隔带之外的点就是" 支持向量".

最后, 由 (75) 可得 SVR 的解为

$$\begin{aligned} w &= \sum_{i=1}^{m} \left(\hat{\alpha}_i - \alpha_i \right) x^{(i)} \\ b &= y^{(i)} + \varepsilon - \sum_{i=1}^{m} \left(\hat{\alpha}_i - \alpha_i \right) {x^{(i)}}^T x. \end{aligned} \tag{81}$$

同样可以使用对所有 $(\hat{\alpha}_i - \alpha_i)$ 不为 0 的样本求解得到 b 的平均值.

2.2 Logistic Regresssion

logistic regression 采用概率的方法进行分类, 而不像 SVM 那样直接给出分类结果. 如果能输入样本 x, 得到 x 为各个类的概率, 那也是一种很好的分类器.logistic regression 正是基于这种思想.

逻辑斯蒂回归最早希望将回归问题映射为分类问题, 假设分类问题为二分类 $y \in \{0,1\}$ 针对线性模型

$$z = w^T x + b \tag{82}$$

映射为分类问题的话, 一般可考虑使用阶跃函数

$$y = sgn(z) = \begin{cases} 1, & z \geq 0 \\ 0, & z < 0 \end{cases} \tag{83}$$

但阶跃函数数学性质不好, 不可导也不连续. 由此考虑到了概率模型,Logistic 分布的定义: 设 X 是连续随机变量, X 服从 Logistic 分布是指 X 具有下列的分布函数和密度函数:

$$\begin{aligned} F(x) = P(X \leq x) &= \frac{1}{1 + e^{-(x-\mu)/\gamma}} \\ f(x) = F'(x) &= \frac{e^{-(x-\mu)/\gamma}}{\gamma \left(1 + e^{-(x-\mu)/\gamma}\right)^2} \end{aligned} \tag{84}$$

其中, μ 是位置参数,X 的期望; $\gamma > 0$ 为形状参数. 当 $\mu = 0, \gamma = 1$ 时, 称为标准的 Logistic 分布, 它的分布函数为

$$\sigma(x) = \frac{1}{1 + e^{-x}} \tag{85}$$

也即 Sigmoid 函数, 分布函数是一条 S 形曲线, 该曲线以点 $\left(\mu, \frac{1}{2}\right)$ 为中心对称, 即满足

$$F(-x + \mu) - \frac{1}{2} = -F(x - \mu) + \frac{1}{2} \tag{86}$$

Logistic 分布适合于分类任务, 且函数较平滑. 这里我们采用最简单的 Sigmoid 函数来定义 logistic regression 的判别标准:

$$\begin{aligned} P(y = 1 \mid x) &= \frac{e^{-(w^T x + b)}}{1 + e^{-(w^T x + b)}} \\ P(y = 0 \mid x) &= \frac{1}{1 + e^{-(w x^T + b)}} \end{aligned} \tag{87}$$

和线性回归一样, 可以简化为

$$\begin{aligned} P(y = 1 \mid x) &= \frac{e^{-w^T x}}{1 + e^{-w^T x}} \\ P(y = 0 \mid x) &= \frac{1}{1 + e^{-w^T x}} \end{aligned} \tag{88}$$

对于未知标签的数据 $\hat{x}$, 我们取 $P(y = 1 \mid \hat{x})$ 中 $P(y = 0 \mid \hat{x})$ 较大的作为预测值.

由此, 对于数据集 $\left\{x^{(i)}, y^{(i)}\right\}_{i=1}^N, \quad x^{(i)} \in \mathbb{R}^d, \quad y^{(i)} \in \{0,1\}$, 初始化参数 (w, b). 我们使用极大似然估计来估计模型参数 (w, b), 损失函数一般取 min, 所以这里用负的似然函数

$$L(w) = -\prod_{i=1}^{N} \left[P(y = 1 \mid x^{(i)})\right]^{y^{(i)}} \left[P(y = 0 \mid x^{(i)})\right]^{1 - y^{(i)}} \tag{89}$$

取对数, 得到 logistic regression 的对数损失函数

$$\begin{aligned}\mathcal{L}(w) &= -\sum_{i=1}^{N}\left[y^{(i)}\log P(y=1\mid x^{(i)}) + \left(1-y^{(i)}\right)\log\left(1-P(y=0\mid x^{(i)})\right)\right] \\ &= -\sum_{i=1}^{N}\left[y^{(i)}\log\frac{P(y=1\mid x^{(i)})}{1-P(y=0\mid x^{(i)})} + \log\left(1-P(y=0\mid x^{(i)})\right)\right] \\ &= -\sum_{i=1}^{N}\left[y^{(i)}\left(w^{T}x^{(i)}\right) - \log\left(1+\exp\left(w^{T}x^{(i)}\right)\right)\right]\end{aligned} \tag{90}$$

对损失函数求梯度得到

$$\frac{\partial}{\partial w}(\mathcal{L}(w)) = -\sum_{i=1}^{N}(y^{(i)} - \sigma(w^{T}x^{(i)}))x^{(i)} \tag{91}$$

或者将 $\mathcal{L}$ 写成矩阵形式 $\mathcal{L} = -y^{T}Xw + 1^{T}\log(1+\exp(Xw))$, 进而可以求得 $\nabla_{w}\mathcal{L} = X^{T}(\sigma(Xw) - y)$ 由此可进行梯度下降或牛顿法进行优化.

其实 logistic regression 的损失函数可以有三种取法, 但只有极大似然函数比较合适.

第一种是由对数几率引出. 事件几率 (*odds*) 定义为事件发生的概率 p 与不发生的概率 $1-p$ 的比值 $\text{odds(p)} = \frac{p}{1-p}$. 针对 logistic regression, 则有对数几率函数

$$\log\ \text{odds}(P(y=1\mid x)) = \log\frac{P(y=1\mid x)}{1-P(y=1\mid x)} = w^{T}x + b \tag{92}$$

从而 $y=1$ 的对数几率是输入 x 的线性函数, 也就是说 logistic regression 模型可以由 x 的线性函数表示, 这也是 logistic regression 的一大特点. 但若以此采用均方误差损失函数来进行线性回归,log 函数和 $y\in\{0,1\}$ 冲突, 故不可行.

第二种方法是直接以均方误差做损失函数, 即

$$L(w) = \sum_{i=1}^{N}\left(y^{(i)} - \frac{1}{1+e^{-w^{T}x^{(i)}}}\right)^{2} \tag{93}$$

其关于 w 非凸, 本质原因是 sigmoid 函数非凸. $\sigma\left(w^{T}x^{(i)}\right)$ 关于 w 的二阶导数为:

$$\sigma''\left(w^{T}x^{(i)}\right) = \left(\sigma\left(-w^{T}x^{(i)}\right) - \sigma\left(w^{T}x^{(i)}\right)\right)\sigma\left(w^{T}x^{(i)}\right)\sigma\left(-w^{T}x^{(i)}\right)x^{(i)}{x^{(i)}}^{T}$$

其符号由第一部分 $\sigma\left(-w^{T}x^{(i)}\right) - \sigma\left(w^{T}x^{(i)}\right)$ 决定, 当 $w^{T}x^{(i)} < 0$ 时, $\sigma\left(-w^{T}x^{(i)}\right) - \sigma\left(w^{T}x^{(i)}\right) > 0$, 二阶导数正定, $\sigma\left(w^{T}x^{(i)}\right)$ 为凸函数；当 $w^{T}x^{(i)} > 0$ 时, $\sigma\left(-w^{T}x^{(i)}\right) - \sigma\left(w^{T}x^{(i)}\right) < 0$, 二阶导数负定, $\sigma\left(w^{T}x^{(i)}\right)$ 为凹函数. 故均方误差函数不能保证海塞矩阵正定.

第三种方法是上文推导的极大似然损失函数

$$L(w) = \prod_{i=1}^{N}\left[P(y=1\mid x^{(i)})\right]^{y^{(i)}}\left[P(y=0\mid x^{(i)})\right]^{1-y^{(i)}} \tag{94}$$

其具有凸性

$$L(w) = \sum_{i=1}^{N}\left[y^{(i)}\left(w^{T}x^{(i)}\right) - \log\left(1+\exp\left(w^{T}x^{(i)}\right)\right)\right] \tag{95}$$

每项的第一部分是线性函数 (凸函数), 第二部分关于 w 的二阶导为: $\nabla_w^2 l = X^T diag\left(\sigma'(Xw)\right) X$ (由定义) 这是一个正定矩阵, 从而 L 是凸函数.

多分类 logistic regression 是根据 softmax 函数来描述预测为各类的概率. 假设希望将样本分为 K 类, 样本 $x^{(i)}$ 为第 r 类的概率为

$$P\left(y=r \mid x^{(i)}\right)=\frac{\exp\left(w_r^T x^{(i)}\right)}{\sum_{j=1}^{K}\exp\left(w_j^T x^{(i)}\right)} \tag{96}$$

显然满足 $\sum_{r=1}^{K} P\left(r \mid x^{(i)}\right)=1$.

这里同样采用极大似然估计先建立似然函数

$$L=-\prod_{i=1}^{N} P\left(y=r \mid x^{(i)}\right) \tag{97}$$

接着通过取对数获得损失函数表达式

$$\mathcal{L}=-\sum_{i=1}^{N}\log\left[P\left(y^{(i)} \mid x^{(i)}\right)\right]=\sum_{i=1}^{n}\left(-w_{y^{(i)}}^T x^{(i)}+\log\sum_{j=1}^{K}\exp\left(w_j^T x^{(i)}\right)\right)=\sum_{i=1}^{N}\log\mathcal{L}_i \tag{98}$$

进而推导损失函数 $\mathcal{L}$ 对于权重 w_r 的梯度表达式

$$\begin{aligned}\frac{\partial\mathcal{L}}{\partial w_r}&=\sum_{i=1}^{N}\frac{\partial\log\mathcal{L}_i}{\partial(w_r^T x^{(i)})}\frac{\partial(w_r^T x^{(i)})}{\partial w_r}\\&=\sum_{i=1}^{N}\frac{\partial\log\mathcal{L}}{\partial(w_r^T x^{(i)})}x^{(i)}\\&=\sum_{i=1}^{N}\begin{cases}x^{(i)}\left(P\left(y=r \mid x^{(i)}\right)-1\right), & \text{If}\quad y^{(i)}=r\\ x^{(i)}P\left(y=r \mid x^{(i)}\right), & \text{If}\quad y^{(i)}\neq r\end{cases}\end{aligned} \tag{99}$$

进而通过梯度下降或牛顿法进行优化.

对于选择 softmax 分类器还是个 logistic 分类器, 取决于所有类别之间是否互斥. 所有类别之间明显互斥用 softmax 分类器, 所有类别之间不互斥有交叉的情况下最好用个 logistic 分类器.

3 降维模型

3.1 主成分分析 (Principal Component Analysis,PCA)

PCA 是一种数据降维的方法, 即用较少特征地数据表达较多特征地数据 (数据压缩,PCA 属于有损压缩).PCA 推导主要基于思路: 最大化数据投影后的的方差 (让数据更分散).

假设有一组数据包含 N 个样本 $D=\left\{x^{(i)}\right\}_{i=1}^{N}$

$$X=\begin{bmatrix} {x^{(1)}}^T \\ \vdots \\ {x^{(N)}}^T \end{bmatrix}$$

每一个样本 $x^{(i)}\in\mathbb{R}^p$. 现在对样本进行降维, 希望找到一个投影矩阵, 将数据从高维空间投影到低维空间中:

$$y^{(i)}=W^Tx^{(i)} \tag{100}$$

其中 $y^{(i)}\in\mathbb{R}^q$ 且 $q<p$, W 为 $W=[w_1 \quad w_2 \quad \cdots \quad w_p]^T\in\mathbb{R}^{p\times q}$. 可以对 W 进行缩放, 所以约束使其为正交阵, 即 $W^TW=I$ (即 w_i 与 w_j 正交) 写成矩阵形式为

$$Y=XW \tag{101}$$

其中 $X=\begin{bmatrix}x^{(1)} & x^{(2)} & \cdots & x^{(N)}\end{bmatrix}^T\in\mathbb{R}^{N\times p}, Y=\begin{bmatrix}y^{(1)} & y^{(2)} & \cdots & y^{(N)}\end{bmatrix}^T\in\mathbb{R}^{N\times q}$

为了方便下面的推导, 先定义几个变量, 首先是各个维度的样本均值

$$\bar{x}=\left[\frac{1}{N}\sum_{i=1}^{N}x^{(i)}\right]^T\in\mathbb{R}^{1\times p} \tag{102}$$

样本投影后各个维度的均值

$$\mu=\left[\frac{1}{N}\sum_{i=1}^{N}x^{(i)}W\right]^T\in\mathbb{R}^{1\times q} \tag{103}$$

令 $\bar{X}=[\bar{x} \quad \bar{x} \quad \cdots \quad \bar{x}]^T\in\mathbb{R}^{N\times p}, M=[\mu \quad \mu \quad \cdots \quad \mu]^T\in\mathbb{R}^{N\times q}$, 显然有 $M=\bar{X}W$.

样本的协方差矩阵为

$$cov(X)=\Sigma=\frac{1}{N-1}\left(X-\bar{X}\right)^T\left(X-\bar{X}\right)\in\mathbb{R}^{p\times p} \tag{104}$$

接着, 对样本在 W 上的投影后的方差进行推导, 投影后的方差等于投影后的协方差矩阵的迹,

$$\begin{aligned}\sigma^2&=tr\left(\frac{1}{N-1}\left(XW-M\right)^T\left(XW-M\right)\right)\\&=tr\left(\frac{1}{N-1}\left(XW-\bar{X}W\right)^T\left(XW-\bar{X}W\right)\right)\\&=tr\left(\frac{1}{N-1}\left(\left(X-\bar{X}\right)W\right)^T\left(\left(X-\bar{X}\right)W\right)\right)\\&=tr\left(\frac{1}{N-1}W^T\left(X-\bar{X}\right)^T\left(X-\bar{X}\right)W\right)\\&=tr(W^T\Sigma W)\end{aligned} \tag{105}$$

综上优化问题为

$$\begin{aligned} &\max_{W} \quad tr(W^T\Sigma W) \\ &\text{s.t. } W^TW = I \end{aligned} \tag{106}$$

从而可以通过拉格朗日乘数法构造一个目标函数

$$L(W,\lambda) = tr(W^T\Sigma W) - \lambda\left(W^TW - I\right), \quad \lambda \in \mathbb{R}^q \tag{107}$$

令偏导为 0

$$\frac{\partial L(W,\lambda)}{\partial W} = 2\Sigma W - 2\lambda W = 0 \tag{108}$$

得到

$$\Sigma W = \lambda W \tag{109}$$

即 λ 是 Σ 的特征值, 而 W 的列向量组是对应的特征向量. 将上式代回 (105), 得到

$$\sigma^2 = tr(W^T\Sigma W) = tr(W^T\lambda W) = tr(\lambda I) \tag{110}$$

由于目标函数 $tr(W^T\Sigma W)$ 是凸函数, 故通过拉格朗日乘数法求得的极值点即为最值点. 我们只要将 Σ 所有的特征值排序, 然后选择前 q 个最大的特征值对应的特征向量放入 λ, 即可得到最大 $\sigma^2 = \sum_{i=1}^{q}\lambda_i$, 此时这些特征向量对应的 q 个单位正交向量组成所需的投影矩阵 W.PCA 的数学推导是严谨的, 但 PCA 没有考虑数据的标签.

3.2 多维尺度变换 (Multiple Dimensional Scaling,MDS)

MDS 的基本思想将高维坐标中的点投影到低维空间中, 保持点彼此之间的相似性尽可能不变.

对于数据集 $V = \left\{x^{(i)}\right\}_{i=1}^{N}, x^{(i)} \in \mathbb{R}^p$, 我们可以计算出距离矩阵 D,$D \in \mathbb{R}^{N\times N}$ 其中第 d_{ij} 表示第 i 个实例和第 j 个实例之间的距离. 我们希望将数据降维至 q 维空间中, 得到 $\left\{y^{(i)}\right\}_{i=1}^{N}, y^{(i)} \in \mathbb{R}^q$, 使得降维后任意两点之间的欧氏距离和降维前尽可能接近, 从而有目标函数

$$F = \sum_{i=1}^{N}\sum_{j=1}^{N}\left(\left\|y^{(i)} - y^{(j)}\right\| - d_{ij}\right)^2 \tag{111}$$

上述的 MDS 的目标函数可以用如下的优化理论来优化. 其基本原理是: 首先, 假设待优化的函数 $f(x)$ 有最优的最小解, 但是直接求解 $f(x)$ 的解析解是非常困难的. 因此, 我们可以找一个简单的, 更容易求解最优值的函数 $g_y(x)$ 来代替求解 $f(x)$. 其中, $g_y(x)$ 满足如下条件对于任意的 x , 有

$$g_y(x) \geq f(x)$$

其中, y 是一个固定点, 并且 $g_y(x)$ 必须" 着陆" 于 $f(x)$, 即 $f_y(x)$ 必须满足如下条件

$$g_y(y) = f(y)$$

则, 假设 x^* 是 $g_y(x)$ 的最小值, 则根据 (6), 有 $f\left(x^*\right) \leq g_y\left(x^*\right)$, 又因为 $g_y\left(x^*\right)$ 是函数 $g_y(x)$ 的最小值, 则有 $g_y\left(x^*\right) \leq g_y(y)$. 所以, 可以得到如下的一个不等式链:

$$f\left(x^*\right) \leq g_y\left(x^*\right) \leq g_y(y) = f(y)$$

从而可以发现, 经过 $g_y(x)$ 函数, $f(x)$ 朝下降的" 方向" 走去 ($f(x^*) \leq f(y)$) . 若反复迭代, 则会找到函数 $f(x)$ 的一个局部最小值, 若 $f(x)$ 是凸函数, 则会找到 $f(x)$ 的最小值. 该优化方法的迭代流程如下:

(1) 选择一个初始值 $y = y_0$, 找到一个函数, 使得 $g_y(x)$ 满足公式 (6) 和公式 (7)

(2) 第 $i+1$ 步, 求解函数 $g_y(x)$, 得到最小值 y_{i+1} .

(3) 如果 $f(y_i) - f(y_{i+1}) < \varepsilon$, 则结束, $f(x)$ 的最优值是 $f(y_{i+1})$; 否则, 另 $y = y_{i+1}$, 转到步骤 (2)

此外还需要用到柯西-施瓦茨不等式,

$$\left(\sum_{i=1}^{n} x_i y_i\right)^2 \leq \left(\sum_{i=1}^{n} x_i^2\right)\left(\sum_{i=1}^{n} y_i^2\right)$$

等号成立的条件是 $\frac{x_1}{y_1} = \frac{x_2}{y_2} = \cdots = \frac{x_n}{y_n}$

首先, 我们将 (111) 拆开

$$\begin{aligned} \sigma(Y) &= \sum_{i=1}^{n}\sum_{j=1}^{n}\left(d_{ij} - \left\|y^{(i)} - y^{(j)}\right\|\right)^2 \\ &= \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2 + \sum_{i=1}^{n}\sum_{j=1}^{n}\left\|y^{(i)} - y^{(j)}\right\|^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\left\|y^{(i)} - y^{(j)}\right\| \end{aligned} \tag{112}$$

其中,$\sum_{i=1}^{n} d_{ij}^2$ 是一个常数项, 对目标函数优化没有影响. $\sum_{i=1}^{n}\sum_{j=1}^{n}\left\|y^{(i)} - y^{(j)}\right\|^2$ 是一个凸的二次函数, 有最小值. 而第三项, 根据柯西-施瓦茨不等式, 可以得知其有界. 如下所示:

$$-2\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\left\|y^{(i)} - y^{(j)}\right\| = -2\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\sqrt{\sum_{s=1}^{p}\left(y_{is} - y_{js}\right)^2} \leq -2\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\frac{\sum_{s=1}^{p}\left(y_{is} - y_{js}\right)\left(z_{is} - z_{js}\right)}{\sqrt{\sum_{s=1}^{p}\left(z_{is} - z_{js}\right)^2}} \tag{113}$$

其中, z_{ij} 是常数. 当 $y_{ij} = z_{ij}$ 时, 上述不等式变成等式, 符合上述优化理论, 可得到:

$$\begin{aligned} \sigma(Y) &= \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2 + \sum_{i=1}^{n}\sum_{j=1}^{n}\left\|y^{(i)} - y^{(j)}\right\|^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\left\|y^{(i)} - y^{(j)}\right\| \\ &\leq \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2 + \sum_{i=1}^{n}\sum_{j=1}^{n}\left\|y^{(i)} - y^{(j)}\right\|^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\frac{\sum_{s=1}^{p}\left(y_{is} - y_{js}\right)\left(z_{is} - z_{js}\right)}{\sqrt{\sum_{s=1}^{p}\left(z_{is} - z_{js}\right)^2}} = \sigma_Z(Y) \end{aligned} \tag{114}$$

为了便于优化, 可以将拆分的目标函数 (112) 转化成矩阵的形式,

$$\begin{aligned} \sigma(Y) &= \sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}^2 + \sum_{i=1}^{n}\sum_{j=1}^{n}\left\|y^{(i)} - y^{(j)}\right\|^2 - 2\sum_{i=1}^{n}\sum_{j=1}^{n} d_{ij}\left\|y^{(i)} - y^{(j)}\right\| \\ &= \eta_\sigma^2 + \eta^2(Y) - 2\rho(Y) \end{aligned} \tag{115}$$

对于其中的 $\eta^2(Y)$, 有

$$\begin{aligned} &\left\|y^{(i)} - y^{(j)}\right\|^2 \\ =&\sum_{s=1}^{p}\left(y_{is} - y_{js}\right)^2 \\ =&tr\left(Y^T A_{ij} Y\right) \end{aligned} \tag{116}$$

其中, $A_{ij}=(e_i-e_j)(e_i-e_j)^T, a_{ii}=a_{jj}=1, a_{ij}=a_{ji}=-1$, 其他地方等于 0. 则

$$\begin{aligned}\eta^2(Y)&=\sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{s=1}^{p}(y_{is}-y_{js})^2\\&=\sum_{i=1}^{n}\sum_{j=1}^{n}tr\left(Y^TA_{ij}Y\right)\\&=tr\left(Y^T\left(\sum_{i=1}^{n}\sum_{j=1}^{n}A_{ij}\right)Y\right)\\&=tr\left(Y^TVY\right)\end{aligned}\tag{117}$$

其中, tr 是矩阵的迹. 矩阵 V 是函数 $\eta^2(Y)$ 的海森矩阵, 其形式如下:

$$V=\begin{bmatrix}2n-2 & -2 & \cdots & -2\\ -2 & 2n-2 & \cdots & -2\\ \vdots & \vdots & \ddots & \vdots\\ -2 & -2 & \cdots & 2n-2\end{bmatrix}$$

对于目标函数中的 $\rho(Y)$, 有

$$\begin{aligned}&d_{ij}\left\|y^{(i)}-y^{(j)}\right\|\\&=d_{ij}\sqrt{\textstyle\sum_{s=1}^{p}(y_{is}-y_{js})^2}\\&=d_{ij}\frac{\sum_{s=1}^{p}(y_{is}-y_{js})^2}{\sqrt{\sum_{s=1}^{p}(y_{is}-y_{js})^2}}\\&=tr\left(Y^Ts_{ij}(Y)A_{ij}Y\right)\end{aligned}\tag{118}$$

其中

$$s_{ij}(Y)=\begin{cases}d_{ij}/d_{ij}(Y) & \text{if } d_{ij}(Y)>0\\ 0 & \text{if } d_{ij}(Y)=0\end{cases}$$

则

$$\begin{aligned}\rho(Y)&=\sum_{i=1}^{n}\sum_{j=1}^{n}tr\left(Y^Ts_{ij}(Y)A_{ij}Y\right)\\&=tr\left(Y^T\left(\sum_{i=1}^{n}\sum_{j=1}^{n}s_{ij}(Y)A_{ij}\right)Y\right)\\&=tr\left(Y^TB_{ij}(Y)Y\right)\end{aligned}\tag{119}$$

综上有

$$\sigma(Y)=1+tr\left(Y^TVY\right)-2tr\left(Y^TB(Y)Y\right)\le 1+tr\left(Y^TVY\right)-2tr\left(Y^TB(Z)Z\right)=\tau_Z(Y)\tag{120}$$

至此, 我们可以优化 $\tau Z(Y)$ 来迭代求解 $\sigma(Y)$.

$$\frac{\partial\tau_Z(Y)}{\partial Y}=2VY-2B(Z)Z=0\tag{121}$$

得到

$$Y^* = V^{-1}B(Z)Z \tag{122}$$

其中, V^{-1} 是 V 的广义逆.

从另外一个角度, 我们直接要求任意两个实例在 q 维空间中的距离与原始空间的距离相同, 可以得到 MDS 目标函数的近似解.

$$d_{ij}^2 = \left\|y^{(i)} - y^{(j)}\right\|^2 = \left\|y^{(i)}\right\|^2 + \left\|y^{(j)}\right\|^2 - 2{y^{(i)}}^T y^{(j)} \tag{123}$$

因为将在 q 维空间中空间中, 点可以进行平移与旋转, 因此在 Z 维空间中会有多种分布满足要求, 不失一般性, 我们假设投影后的点是中心化的, 即: $\sum_{i=1}^{N} y^{(i)} = 0$

我们对 (123) 左右两边求和:

$$\begin{aligned}
\sum_{i=1}^{N} d_{ij}^2 &= \sum_{i=1}^{N} \left\|y^{(i)}\right\|^2 + N\left\|y^{(j)}\right\|^2 \\
\sum_{j=1}^{N} d_{ij}^2 &= \sum_{j=1}^{N} \left\|y^{(j)}\right\|^2 + N\left\|y^{(j)}\right\|^2 \\
\sum_{i=1}^{N}\sum_{j=1}^{N} d_{jj}^2 &= \sum_{i=1}^{N}\sum_{j=1}^{N} \left\|y^{(j)}\right\|^2 + N\sum_{i=1}^{N}\left\|y^{(i)}\right\|^2 = 2N\sum_{i=1}^{N}\left\|y^{(i)}\right\|^2
\end{aligned} \tag{124}$$

定义内积矩阵 $B = Y^T Y, \quad B \in \mathbb{R}^{N\times N}$. 结合上述两式, 可得:

$$b_{ij} = -\frac{1}{2}\left(\frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N} d_{ij}^2 - \frac{1}{N}\sum_{i=1}^{N} d_{ji}^2 - \frac{1}{N}\sum_{j=1}^{N} d_{ij}^2 + d_{ij}^2\right) \tag{125}$$

从而我们由距离矩阵 D 得到了 B, 进而我们可能可以反解出 $Y \in \mathbb{R}^{q\times N}$

由于矩阵 B 是一个是实对称矩阵, 因此对矩阵 B 进行特征分解可以得到:

$$B = U\Lambda U^T \tag{126}$$

其中, Λ 为特征值构成的对角阵, U 是正交阵. 由于我们将数据降维到 q 维空间中, 因此我们选择前 q 个最大的特征值以及对应的特征向量, 分别构成 Λ_q 和 U_q, 用 $U_q\Lambda_q U_q^T$ 作为矩阵 B 的近似. 从而降维之后的数据点可以近似表示为

$$Y = U_q\Lambda_q^{1/2} \tag{127}$$

如果 p 和 q 相差不过大, 那么这种近似往往很有效.

3.3 流形学习

流形学习是一类借鉴了拓扑流形概念的降维方法, 认为低维流形嵌入到高维空间之后, 直接在高维空间中计算直线距离具有误导性, 因为高维空间中的直线距离在低维嵌入流形上是不可达的.

流形是指在局部与欧氏空间同胚的空间, 即它在局部具有欧氏空间的性质, 能用欧氏距离进行距离计算. 这样即使高维空间的分布十分复杂, 但是在局部上依然满足欧式空间的性质, 基于流形学习的降维正是这种邻域保持的思想.

测地距离的计算: 利用流形在局部上与欧式空间同胚的性质, 可以使用近邻距离来逼近测地线距离, 即对于一个样本点, 它与近邻内的样本点之间是可达的, 且距离使用欧式距离计算, 这样整个样本空间就形成了一

张近邻图, 高维空间中两个样本之间的距离就转为计算近邻连接图上两点间的最短路径问题. 可采用著名的 Dijkstra 算法或 Floyd 算法计算最短距离

但是计算测地距离就需要样本密采样, 而这也是高维情形下的难点, 故流形学习的效果在实际中往往不如预期.

3.3.1 等度量映射 (Isometric Mapping,Isomap)

Isomap 就是用两点之间的测地距离代替了欧式距离, 其他与 MDS 完全一致.

3.3.2 拉普拉斯特征映射 (Laplacian Eigenmaps, LE)

首先明确拉普拉斯矩阵及其一些性质.

将数据集 $V=\left\{x^{(i)}\right\}_{i=1}^{N}$ 的每个数据看做无向图中的一个节点, 定义两点之间的相似度 (权重)w_{ij}, 即邻接矩阵 $W=(w_{ij})_{N\times N}$, 需要满足 $w_{ij}=w_{ji}$. 定义邻接矩阵 W 的方法有的是基于欧式距离的, 有的是基于流形的 (测地距离), 在这里我们一般化的将两点之间的距离表示为 s_{ij}, 其定义一般有两类:

1)ε-邻近法,

$$s_{ij}=\begin{cases}0 & \|x_i-x_j\|>\varepsilon\\ \varepsilon & \|x_i-x_j\|\leqslant\varepsilon\end{cases} \tag{128}$$

2)K 邻近法利用 KNN 算法遍历所有的样本点, 取每个样本最近的 k 个点作为近邻. 为保证邻接矩阵为对称阵, 第一种 K 邻近法是只要一个点在另一个点的 K 近邻中即非零,

$$s_{ij}=\begin{cases}0 & x_i\notin KNN\left(x_j\right)\ \text{and}\ x_j\notin KNN\left(x_i\right)\\ \|x_i-x_j\| & x_i\in KNN\left(x_j\right)\ \text{or}\ x_j\in KNN\left(x_i\right)\end{cases} \tag{129}$$

第二种 K 邻近法是必须两个点互为 K 近邻中才非零,

$$s_{ij}=\begin{cases}0 & x_i\notin KNN\left(x_j\right)\ \text{or}\ x_j\notin KNN\left(x_i\right)\\ \|x_i-x_j\| & x_i\in KNN\left(x_j\right)\ \text{and}\ x_j\in KNN\left(x_i\right)\end{cases} \tag{130}$$

进而根据距离 s_{ij} 确定 w_{ij}, 可以选择不同的核函数来定义边权重, 常用的有多项式核函数, 高斯核函数和 Sigmoid 核函数. 最常用的是高斯核函数 RBF:

$$w_{ij}=w_{ji}=\exp\left(-\frac{s_{ij}^2}{2\sigma^2}\right) \tag{131}$$

由相似度 w_{ij} 的定义, 进而定义每个节点的度,$x^{(i)}$ 的度 d_i 定义为和它相连的所有边的权重之和, 即

$$d_i=\sum_{j=1}^{N}w_{ij} \tag{132}$$

利用每个点度的定义, 我们可以得到一个 $N\times N$ 的度矩阵 D, 它是一个对角阵, 定义为:

$$D=\begin{pmatrix}d_1 & & & \\ & d_2 & & \\ & & \ddots & \\ & & & d_N\end{pmatrix} \tag{133}$$

对于点集 V 的的一个子集 $V_i \subset V$, 我们定义:

$$\begin{aligned} |V_i| &= \text{ 子集}V_i \text{ 中点的个数} \\ d(V_i) &= \sum_{i \in V_i} d_i \end{aligned} \tag{134}$$

进而定义图的拉普拉斯矩阵, $L = D - W$. 在带权无向图中, 拉普拉斯矩阵有一些很好的性质:

1) 是对称矩阵, 因为 D 和 W 都是对称阵.

2) 拉普拉斯矩阵是半正定的.

证明: 对于任意的非零向量 x,

$$\begin{aligned} x^T L x &= x^T D x - x^T W x \\ &= \sum_{i=1}^{N} d_i x_i^2 - \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} x_i x_j \\ &= \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} x_i^2 - \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} x_i x_j \\ &= \frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} x_i^2 + \sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} x_j^2 - 2\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij} x_i x_j\right) \\ &= \frac{1}{2}\left(\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}(x_i^2 + x_j^2 - 2x_i x_j)\right) \\ &= \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\left(x_i - x_j\right)^2 \geqslant 0 \end{aligned} \tag{135}$$

3)$(0,1)$ 为 L 的特征对. 验证即可.

4)L 的零特征值的几何重数等于图的连通分支数. 证明: 设图的连通分支数为 k. 首先考虑 $k = 1$, 则有 $w_{ij} > 0$, 由性质 2)

$$x^T L x = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} w_{ij}\left(x_i - x_j\right)^2 = 0 \tag{136}$$

故 $x_i = x_j$, 即 x 各个元素相同, 从而说明此时 L 的零特征空间只有一个基 $(1, 1, \cdots, 1)$.

进而考虑 $k > 1$, 我们可以通过调整节点顺序使得 L 为分块对角阵

$$L = \begin{pmatrix} L_1 & & & \\ & L_2 & & \\ & & \ddots & \\ & & & L_k \end{pmatrix} \tag{137}$$

每个小块由在一个连通分支的节点构成, 和一个连通分支一一对应. 如此可以构造线性无关的向量组

$$\begin{aligned} e_1 &= [1, 1, \cdots, 1, 0, 0, \cdots, 0]^T \in R^n \\ e_2 &= [0, 0, \cdots, 0, 1, 1, \cdots, 1, 0, 0, \cdots, 0]^T \in R^n \\ e_k &= [0, 0, \cdots, 0, 1, 1, \cdots, 1]^T \in R^n \end{aligned} \tag{138}$$

与 L 相对应, 由矩阵乘法, 这相当于对每个连通分支考虑 $k=1$ 的情况, 故 L 的零特征空间由 $e_1, e_2, \cdots, e_k$ 张成, 结论成立.

与 MDS 的保留距离不同,LE 希望原本接近的数据降维后也比较接近, 原本较远的数据降维后也较远. 考虑对数据集 $\left\{x^{(i)}\right\}_{i=1}^{N}, x^{(i)} \in \mathbb{R}^p$ 进行降维, 将其映射到 q 维空间,$f: x^{(i)} \to y^{(i)}, \quad y^{(i)} \in \mathbb{R}^q$. 与 PCA 考虑线性变换不同,LE 通过考虑映射后的性质来求得降维结果, 用局部的角度去构建数据之间的关系. 其主要思想是, 如果两个数据实例 $x^{(i)}$ 和 $x^{(j)}$ 很相似, 那么其降维后目标子空间中也应该尽量接近, 即 $y^{(i)}$ 和 $y^{(j)}$ 接近. 令 $Y=[y^{(1)}, y^{(2)}, \cdots, y^{(N)}]$, 这里我们以欧氏距离为例, 则拉普拉斯特征映射优化的目标函数可以为:

$$\min_{Y} \sum_{i=1}^{N} \sum_{j=1}^{N} \left\| y^{(i)} - y^{(j)} \right\|^2 w_{ij} \tag{139}$$

这个目标函数与 L 有关系, 即

$$\begin{aligned}
\sum_{i=1}^{N} \sum_{j=1}^{N} \left\| y^{(i)} - y^{(j)} \right\|^2 w_{ij} &= \sum_{i=1}^{N} \sum_{j=1}^{N} \left({y^{(i)}}^T y^{(i)} - 2 {y^{(i)}}^T y^{(j)} + {y^{(j)}}^T y^{(j)} \right) w_{ij} \\
&= \sum_{i=1}^{N} \left(\sum_{j=1}^{N} w_{ij} \right) {y^{(i)}}^T y^{(i)} + \sum_{j=1}^{N} \left(\sum_{i=1}^{N} w_{ij} \right) {y^{(j)}}^T y^{(j)} - 2 \sum_{i=1}^{N} \sum_{j=1}^{N} {y^{(i)}}^T y^{(j)} w_{ij} \\
&= 2 \sum_{i=1}^{N} d_i {y^{(i)}}^T y^{(i)} - 2 \sum_{i=1}^{N} \sum_{j=1}^{N} {y^{(i)}}^T y^{(j)} w_{ij} \\
&= 2 \sum_{i=1}^{N} \left(\sqrt{d_i} y^{(i)} \right)^T \left(\sqrt{d_i} y^{(i)} \right) - 2 \sum_{i=1}^{N} {y^{(i)}}^T \left(\sum_{j=1}^{N} y^{(j)} w_{ij} \right) \\
&= 2tr\left(YDY^T\right) - 2 \sum_{i=1}^{N} {y^{(i)}}^T (YW)_i \\
&= 2tr\left(YDY^T\right) - 2tr\left(Y^T YW\right) \\
&= 2tr\left(YDY^T\right) - 2tr\left(YWY^T\right) \\
&= 2tr\left[Y(D-W)Y^T\right] \\
&= 2tr\left(YLY^T\right)
\end{aligned} \tag{140}$$

此外, 这个目标函数还存在约束性的问题: 首先, 解可以退化到一个点, 即所有 $y^{(i)}$ 相同；其次,$y^{(i)}$ 可以有任意的尺度. 因此, 为该优化问题加上约束

$$YDY^T = I \tag{141}$$

这里为什么乘 D 我查阅了资料没有找到合适的解释, 一个模糊的说法是这一度量使得特征向量的数值分布更均匀, 但没有具体的证明. 简单说就是 d_i 越大,$y^{(i)}$ 越重要. 由此我们得到优化问题:

$$\begin{aligned}
&\min_{Y} tr\left(YLY^T\right) \\
&YDY^T = I
\end{aligned} \tag{142}$$

使用拉格朗日乘数法

$$L(Y,\lambda)=tr\left(YLY^T\right)+\sum_{i=1}^{N}\lambda_i\left(1-{y^{(i)}}^T Dy^{(i)}\right)=\sum_{i=1}^{N}{y^{(i)}}^T Ly^{(i)}+\lambda_i\left(1-{y^{(i)}}^T Dy^{(i)}\right) \tag{143}$$

令偏导为 0 得到,

$$Ly^{(i)}=\lambda Dy^{(i)} \tag{144}$$

由于目标函数 $tr\left(YLY^T\right)$ 是一个凸函数, 故拉格朗日乘数法求得的极值点即为最值点. 式 (144) 是个广义特征值问题, 其最优解是 $D^{-1}L$ 的 q 个最小的特征值之和, Y 是由最小的 q 个特征值对应的特征向量作为列构成的矩阵.

实际上, 由 L 的性质,L 具有特征值 0 和特征向量 1 , 如果选取 1 去构成 Y 会导致所有 y_i 的第一个分量都是 1, 退化到更低一维的空间. 因此目标函数的最优解是 $D^{-1}L$ 的 q 个最小的非零特征值之和, Y 是由最小的 q 个非零特征值对应的特征向量作为列构成的矩阵.

综上,LE 的算法流程非常简单, 对于一个数据集, 确定 w_{ij} 的取法, 求得 L, 进而计算广义特征值问题 $Lf=\lambda Df$ 的 q 个最小的非零特征值之和, 从而得到降维后的数据,Y 是由最小的 q 个非零特征值对应的特征向量作为列构成的矩阵.

3.3.3 局部保持投影 (Locality Preserving Projections,LPP)

LPP 的思想与 LE 一致,LPP 希望求得 LE 所得到的映射的线性表达, 即 LE 是非线性降维,LPP 是对 LE 的线性近似. 原文中说, 由 LPP 得到映射可以看作是由流形几何自然产生的连续映射的线性离散逼近. 考虑对 $\left\{x^{(i)}\right\}_{i=1}^{N},x^{(i)}\in\mathbb{R}^p$ 的线性降维, 设 $X=[x^{(1)},x^{(2)},\cdots,x^{(N)}]\in\mathbb{R}^{p\times N}$, 通过 $f:x^{(i)}\to y^{(i)}\in\mathbb{R}^q$ 得到 $Y=[y^{(1)},y^{(2)},\cdots,y^{(N)}]\in\mathbb{R}^{q\times N}$, 并且这种降维是线性的, 有 $Y=A^TX,\quad W\in\mathbb{R}^{p\times N}$.

将 LE 的推导过程中的 Y 替换成 A^TX 即可, 由此得到优化问题

$$\begin{aligned}&\min_W tr\left(A^TXLX^TA\right)\\&A^TXDX^TA=I\end{aligned} \tag{145}$$

通过拉格朗日乘数法, 令偏导为 0 得到广义特征值问题

$$XLX^Ta=\lambda XDX^Ta \tag{146}$$

取前 q 个最小特征值对应的特征向量构成矩阵 A.

另外, 如果使用核方法,LPP 又可以转化为 LE. 考虑映射 $\phi:\mathbb{R}^p\to\mathcal{H}$, 则通过 ϕ 将数据集全部映射到希尔伯特空间得到 $\phi(X)=[\phi(x^{(1)}),\phi(x^{(2)}),\cdots,\phi(x^{(N)})]$. 现在 LPP 在希尔伯特空间中的特征值问题由 (146) 变为

$$\phi(X)L\phi^T(X)a=\lambda\phi(X)D\phi^T(X)a \tag{147}$$

因为 $\mathcal{H}$ 是由 X 的像完备成的希尔伯特空间, 故上述广义特征值问题在 $\mathcal{H}$ 中的特征向量可由 $\phi(X)$ 的列向量线性表出, 即

$$a=\sum_{i=1}^{N}\xi_i\phi(x^{(i)})=\phi(X)\xi \tag{148}$$

其中 $\xi=[\xi_1,\xi_2,\cdots,\xi_N]$. 将 (148) 代入 (147), 再两端左乘 $\phi^T(X)$ 得到

$$KLK\xi=\lambda KDK\xi \tag{149}$$

由此求得 N 个解 $\xi^1, \xi^2, \cdots, \xi^N$. 对新数据点 x, 其在特征向量 a^k 上的投影为

$$(a^k \cdot \phi(x)) = \sum_{i=1}^{N} \xi_i^k (\phi(x^{(i)}) \cdot \phi(x)) = \sum_{i=1}^{N} \xi_i^k K\left(x^{(i)}, x\right) \tag{150}$$

而对测试集来说, 降维后的映像为 $y = K\xi$, 故 (149) 可写为

$$Ly = \lambda Dy \tag{151}$$

这和 LE 的优化结果一致, 故带有核方法的 LPP 其实和 LE 一致.

相比于 LE,LPP 给出了显式的降维方式 $Y = A^T X$, 这使得 LPP 可以直接对这个问题的新数据进行降维, 也可以将数据集分为 train data 和 test data, 而 LE 则做不到. 此外,LPP 可以通过核方法的引入得到和 LE 相同的结果.

3.3.4 局部线性嵌入 (Locally Linear Embedding,LLE)

上述的 MDS、LE 和 Isomap 倾向于在降维后保留两两数据点之间的距离,LLE 则希望保留局部多个点之间的线性关系. 所谓局部线性, 即认为在整个数据集的某个小范围内是线性的, 每个数据点可由临近的数据点的一个凸组合, 选取临近点的最直接的办法就是 KNN.

考虑对 $\left\{x^{(i)}\right\}_{i=1}^{N}, x^{(i)} \in \mathbb{R}^p$ 的降维,$f : x^{(i)} \to y^{(i)}$. 选定 k 保证数据集中所有点都是其邻近的 k 个的数据点的凸组合. 严格说必须要 p 个线性无关的数据点保证能够线性表出任一数据点, 但是用邻近的点表示可能需要少于 p 个.

更一般的做法是指定 k 值, 将每个数据点近似线性表出即可. 设对数据点 $x^{(i)}$ 选出的 KNN 集合为 S_i, 则可根据近邻关系计算出所有样本的邻域重构系数 w_i:

$$\begin{aligned} &\min_{W} \sum_{i=1}^{N} \left\| x_i - \sum_{x_j \in S_i} w_{ij} \in \mathbb{R}^{N \times N} x_j \right\|_2^2, \quad (W)_{ij} = w_{ij} \\ &\text{s.t.} \sum_{x_j \in S_i} w_{ij} = 1 \end{aligned} \tag{152}$$

对目标函数求最优,

$$\begin{aligned} \Phi(W) &= \sum_{i=1}^{N} \left\| x_i - \sum_{j=1}^{k} w_{ij} x_j \right\|^2 \\ &= \sum_{i=1}^{N} \left\| \sum_{j=1}^{k} (x_i - x_j) w_{ij} \right\|^2 \\ &= \sum_{i=1}^{N} \left\| (X_i - N_i) w_i \right\|^2, X_i = \underbrace{[x_i, \ldots, x_i]}_{k}, N_i = [x_{1i}, \ldots, x_{ki}] \\ &= \sum_{i=1}^{N} w_i^T (X_i - N_i)^T (X_i - N_i) w_i \end{aligned}$$

将 $\Sigma_i = (X_i - N_i)^T (X_i - N_i)$ 看做局部协方差矩阵, 即:

$$\Phi(W) = \sum_{i=1}^{N} w_i^T \Sigma_i w_i \tag{153}$$

运用拉格朗日乘子法:

$$L(W)=\sum_{i=1}^{N} w_i^T \Sigma_i w_i+\lambda\left(w_i^T 1_k-1\right) \tag{154}$$

求导可得:

$$\frac{\partial L(W)}{\partial w_i}=2\Sigma_i w_i+\lambda 1_k=0 \tag{155}$$

解得

$$w_i=\frac{\Sigma_i^{-1}1_k}{1_k^T\Sigma_i^{-1}1_k} \tag{156}$$

保留局部线性结构即使降维后邻域重构系数 w_{ij} 不变, 由此求解低维坐标:

$$\min_{y^{(1)},y^{(2)},\ldots,y^{(N)}} \sum_{i=1}^{m}\left\|y^{(i)}-\sum_{j\in Q_i} w_{ij}y^{(i)}\right\|_2^2,\quad Y=\left(y^{(1)},y^{(2)},\ldots,y^{(N)}\right)\in\mathbb{R}^{q\times N} \tag{157}$$

令. 设 $M=(I-W)^T(I-W)$, 这样利用矩阵 M, 再加上尺度降维后的尺度约束 $YY^T=I$ 优化问题可以重写为:

$$\begin{aligned}&\min_Y tr\left(YMY^T\right)\\&\text{s.t. } YY^T=I\end{aligned} \tag{158}$$

和之前一样,M 是实对称阵,M 的 q 个最小的特征值对应的特征向量单位化后组成的矩阵即为 Y^T

3.4 线性判别分析 (Linear Discriminant Analysis,LDA)

LDA 是对有类别的数据集进行降维, 其核心是" 最大化类间均值, 最小化类内方差"(Fisher 判别分析思想).

3.4.1 Rayleigh 商及其性质

首先说明一下瑞丽商, 设 $A\in\mathbb{R}^{n\times n}$ 为实对称阵,$x\in\mathbb{R}^n$, 那么瑞利商 $R(A,x)$ 为:

$$R(A,x)=\frac{x^TAx}{x^Tx} \tag{159}$$

设 A 的特征值与特征向量分别为 $\lambda_1,\lambda_2,\cdots,\lambda_n$ 以及 $v_1,v_2,\cdots,v_n$, 并且有: $\lambda_{\min}=\lambda_1\le\lambda_2\le\ldots\le\lambda_n=\lambda_{\max}$ 则瑞利商具有如下性质:

$$\lambda_{\min}\le\frac{x^TAx}{x^Tx}\le\lambda_{\max} \tag{160}$$

下面给出证明: 在 A 给定的情况下, 由于 A 是一个实对称阵, 所以存在正定阵 U 满足:$A=U\Sigma U^T$ 其中 $\Sigma=diag(\lambda_1,\lambda_2,\ldots,\lambda_n)$, 代入瑞利商:

$$R(A,x)=\frac{x^TUAU^Tx}{x^Tx}=\frac{\left(U^Tx\right)^TA\left(U^Tx\right)}{x^Tx} \tag{161}$$

设 $p=U^Tx$, 有:

$$R(A,x)=\frac{p^TAp}{x^Tx}=\frac{\sum_{i=1}^n\lambda_i|p_i|^2}{\sum_{i=1}^n|x_i|^2} \tag{162}$$

根据特征值的大小关系, 于是有:

$$\frac{\lambda_1 \sum_{i=1}^{n} |p_i|^2}{\sum_{i=1}^{n} |x_i|^2} \leq R(A,x) \leq \frac{\lambda_n \sum_{i=1}^{n} |p_i|^2}{\sum_{i=1}^{n} |x_i|^2} \tag{163}$$

设 U 的第 i 行, 第 j 列元素为 u_{ij}, U^T 的第 i 行, 第 j 列元素为 u_{ji} , 那么: $p_i = \sum_{j=1}^{n} u_{ji} x_j$ 于是:

$$\sum_{i=1}^{n} |p_i|^2 = \sum_{i=1}^{n} p_i^T p_i = \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{n} x_j u_{ij} u_{ki} x_k = \sum_{j=1}^{n} \sum_{k=1}^{n} \left(\sum_{i=1}^{n} u_{ki} u_{ij} \right) x_j x_k \tag{164}$$

由于 U 是正定阵, 即:$U^T U = I$, 写成展开式形式为:

$$I_{jk} = \sum_{i=1}^{n} u_{ji} u_{ik} \tag{165}$$

当 $j \neq k$ 时, $I_{jk} = 0$, 当 $j = k$ 时, $I_{jk} = 1$, 所以可以得到:

$$\sum_{i=1}^{n} |p_i|^2 = \sum_{i=1}^{n} |x_i|^2$$

代入上述不等式可得:

$$\lambda_1 \leq R(A,x) \leq \lambda_n$$

并且当 $x = v_1$ 时 $R(A,x) = \lambda_1$, 当 $x = v_n$ 时 $R(A,x) = \lambda_n$. 这就证明了前面的结论.

此外, 如果我们用 $x' = cx$ 来取代 x, 其中 c 为非零的实数, 有:

$$R(A,x') = \frac{x'^T A x'}{x'^T x'} = \frac{c x^T A x c}{c x^T x c} = R(A,x)$$

也就是说, 对 x 进行等比例缩放并不会影响瑞利商的值, 即:$R(A,cx) = R(A,x)$ 于是, 我们可以令 $x^T x = 1$, 这样就有 $R(A,x) = x^T A x$. 此时对瑞利商求极值就是在约束 $x^T x = 1$ 条件下, 对 $x^T A x$ 求极值. 下面使用拉格朗日乘子法来解, 定义拉格朗日函数:

$$L(x,\lambda) = x^T A x - \lambda \left(x^T x - 1 \right) \tag{166}$$

对 x 求梯度, 并令梯度为 0 :

$$\nabla_x L = 2Ax - 2\lambda x = 0 \tag{167}$$

即瑞利商的极值在 A 的特征向量上取得, 并且极值为特征值.

瑞利商的另一种推广形式是广义瑞利商. 定义:

$$R(A,B,x) = \frac{x^T A x}{x^T B x} \tag{168}$$

其中 B 为实对称正定矩阵, 基于同样的理由, 我们缩放 x 使得 $x^T B x = 1$, 然后利用拉格朗日乘子法求 $x^T A x$ 的极值, 定义:

$$L(x,\lambda) = x^T A x - \lambda \left(x^T B x - 1 \right) \tag{169}$$

然后求梯度取零得到驻点:

$$\begin{aligned} &\nabla_x L = 2Ax - 2\lambda B x = 0 \\ &\Leftrightarrow Ax = \lambda B x \\ &\Leftrightarrow B^{-1} A x = \lambda x \end{aligned} \tag{170}$$

也就是说, $R(A,B,x)$ 的极值在 $B^{-1}A$ 的特征向量上取得, 其极值就为 $B^{-1}A$ 的特征值. 此外和瑞丽商同理,$R(A,B,x)$ 的最大值为 $B^{-1}A$ 的最大特征值, 最小值为 $B^{-1}A$ 的最小特征值.

3.4.2 二类 LDA 与多类 LDA

一般来说, 二类 LDA 与多类 LDA 只有类间散度矩阵 S_B 的定义不同, 这里我们以多类 LDA 来描述.

对于有类别的数据集 $D=\left\{x^{(i)}\right\}_{i=1}^{N}, \quad x^{(i)} \in \mathbb{R}^p$, 可被划分成 k 个互不相交的非空子集 $D_1 \cup D_2 \cup \ldots \cup D_K = D, D_1 \cap D_2 \cap \ldots \cap D_K = \varnothing$, 设子集 D_k 的元素个数为 N_k, 则有 $\sum_{k=1}^{K} N_k = N$.

和 PCA 一样, 我们希望找到投影矩阵 $W \in \mathbb{R}^{p \times q}$, 将样本 $x^{(i)}$ 投影到

$$y^{(i)} = W^T x^{(i)} \tag{171}$$

,LDA 希望 W 能使类内方差尽可能小, 类间均值方差尽可能大.

做几个简单的定义, 子集 D_k 的均值 (中心) 为

$$\mu_k = \frac{1}{N_k} \sum_{x^{(i)} \in D_k} x^{(i)} \tag{172}$$

总体样本均值:

$$\mu = \frac{1}{N} \sum_{x^{(i)} \in D} x^{(i)} \tag{173}$$

定义各类类内散度矩阵

$$S_w(k) = \sum_{x^{(i)} \in D_k} \left(x^{(i)} - \mu_k\right)\left(x^{(i)} - \mu_k\right)^T \tag{174}$$

总体样本的类内散度矩阵定义为各类类内散度矩阵之和:

$$S_w = \sum_{k=1}^{K} S_w(k) = \sum_{k=1}^{K} \sum_{x^{(i)} \in D_k} \left(x^{(i)} - \mu_k\right)\left(x^{(i)} - \mu_k\right)^T \tag{175}$$

需要注意的是, 类间离散度矩阵的定义有些不同, 如果是二类 LDA, 类间散度矩阵定义为

$$S_b = \left(\mu_2 - \mu_1\right)\left(\mu_2 - \mu_1\right)^T \tag{176}$$

直接衡量两类中心的差异即可. 多类 LDA 则衡量多类中心的差异度, 度量的是每类均值点相对于样本中心的离散情况. 类似于将 μ_k 看作样本点, μ 看作均值的协方差矩阵, 并根据每类中的数据点个数赋权, 权重用 $\frac{N_k}{N}$ 表示,

$$S_b = \sum_{k=1}^{K} N_k \left(\mu_k - \mu\right)\left(\mu_k - \mu\right)^T \tag{177}$$

对所有数据点作用投影矩阵的线性变换后, 类内散度矩阵为 $W^T S_w W$, 最小化类内方差等价于最小化 $tr(W^T S_w W)$；类间散度矩阵为 $W^T S_b W$, 最大化类间散度矩阵等价于最大化 $tr(W^T S_b W)$

$$\arg\max_W L(W) = \frac{tr(W^T S_b W)}{tr(W^T S_w W)} \tag{178}$$

但这种目标函数不易求最值. 一个常见的 LDA 多类优化目标函数定义为:

$$\arg\max_W L(W) = \frac{\prod \operatorname{diag}\left(W^T S_b W\right)}{\prod \operatorname{diag}\left(W^T S_w W\right)} \tag{179}$$

$L(W)$ 的优化过程进而可以转化为:

$$L(W) = \frac{\prod_{j=1}^{q} w_j^T S_b w_j}{\prod_{j=1}^{q} w_j^T S_w w_j} = \prod_{j=1}^{q} \frac{w_j^T S_b w_j}{w_j^T S_w w_j} \tag{180}$$

上式右端为广义瑞利商, 其最大值是矩阵 $S_w^{-1}S_b$ 的最大特征值, 最大的 q 个值的乘积就是矩阵 $S_w^{-1}S_b$ 的最大的 q 个特征值的乘积, 此时对应的矩阵 W 为这最大的 q 个特征值对应的特征向量张成的矩阵.

由于 W 是一个利用了样本的类别得到的投影矩阵, 因此它的降维到的维度 q 最大值为 $K-1$. 因为 S_b 中每个 $\mu_k-\mu$ 的秩为 1 , 因此协方差矩阵相加后最大的秩为 K(矩阵的秩小于等于各个相加矩阵的秩的和), 但是由于如果我们知道前 $K-1$ 个 μ_k 后, 最后一个 μ_K 可以由前 $K-1$ 个 μ_k 线性表出, 因此 S_b 的秩最大为 $K-1$(与协方差矩阵的秩最大为 n-1 同理), 从而线性无关的特征向量最多有 $K-1$ 个, 也即 $q\leqslant K-1$.

此外, 大多数情况下 S_w 是不可逆的, 因此我们可以令:

$$S_w=S_w+\beta I \tag{181}$$

其中是 β 一个很小的数, 来保证 S_w 可逆.

3.5 t-分布随机近邻嵌入 (t-distributed Stochastic Neighbor Embedding,t-SNE)

t-SNE 是一种面向可视化的降维方法. 原文写道:" 之前的有的在全局的区分度上做得很好 (如 PCA 和 LDA), 有的在局部关系的保留上做得很好 (如其他方法), 但是少有能将两者兼得, 即既保留局部结构又区分整体簇类,t-SNE 对此有不错的效果."

3.5.1 随机近邻嵌入 (Stochastic Neighbor Embedding,SNE)

t-SNE 是 SNE 的变体, 我们先推导 SNE 的原理. 对于数据集 $\left\{x^{(i)}\right\}_{i=1}^N,\quad x^{(i)}\in\mathbb{R}^p$ 进行降维, 使其映射到 $\left\{y^{(i)}\right\}_{i=1}^N,\quad y^{(i)}\in\mathbb{R}^q$.SNE 与上面所有模型在方法上的最大区别就是 SNE 使用概率来衡量两点的相似度, 且这种相似度是不对称的. 设定 $x^{(j)}$ 是 $x^{(i)}$ 的近邻的概率 p_{ij} 为

$$p_{ij}=\frac{\exp\left(\frac{-\left\|x^{(i)}-x^{(j)}\right\|^2}{\left(2\sigma_i^2\right)}\right)}{\sum_{k\neq i,k=1}^N\exp\left(\frac{-\left\|x^{(i)}-x^{(k)}\right\|^2}{\left(2\sigma_i^2\right)}\right)} \tag{182}$$

或记作条件概率 $p_{j|i}$. 显然 $p_{ij}=pji$ 不一定成立, 这与两个点与其他点的距离关系有关. 我们设置 $p_{ii}=0$, 因为我们关注的是两两之间的相似度此外, 这里的有一个参数是 σ_i, 对于不同的点 σ_i 取值不一样, 这里讨论如何设置.

首先需要插入一个小内容, 求解高斯分布的熵 (积分形式), 设 $p(x)$ 为一个一元高斯分布

$$p(x)=\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\} \tag{183}$$

求解 $H[x]=-\int p(x)\ln p(x)dx$ 我们已知 $\int_{-\infty}^{\infty}e^{-x^2}dx=\sqrt{\pi}$ 将公式带入得:

$$\begin{aligned}
H[x]&=-\int\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}\ln\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}dx\\
&=-\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\int\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}\left(-\ln(\sqrt{2\pi}\sigma)-\frac{(x-\mu)^2}{2\sigma^2}\right)dx\\
&=-\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\cdot-\ln(\sqrt{2\pi}\sigma)\int\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}dx+\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\int\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}\frac{(x-\mu)^2}{2\sigma^2}dx\\
&=\frac{\ln(\sqrt{2\pi}\sigma)}{(2\pi\sigma^2)^{\frac{1}{2}}}\int\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}dx+\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\int\exp\left\{-\frac{(x-\mu)^2}{2\sigma^2}\right\}\frac{(x-\mu)^2}{2\sigma^2}dx\\
&=\frac{\ln(\sqrt{2\pi}\sigma)}{(2\pi\sigma^2)^{\frac{1}{2}}}\sqrt{2}\sigma\int\exp\left\{-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right\}d\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)+\frac{1}{(2\pi\sigma^2)^{\frac{1}{2}}}\sqrt{2}\sigma\int\exp\left\{-\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)^2\right\}\frac{(x-\mu)^2}{2\sigma^2}d\left(\frac{x-\mu}{\sqrt{2}\sigma}\right)\\
&=\frac{\ln(\sqrt{2\pi}\sigma)}{\sqrt{\pi}}\int_{-\infty}^{\infty}e^{-y^2}dy+\frac{1}{\sqrt{\pi}}\int_{-\infty}^{\infty}e^{-y^2}y^2dy\\
&=\ln(\sqrt{2\pi}\sigma)+\frac{1}{\sqrt{\pi}}\cdot-\frac{1}{2}\left(0-\int_{-\infty}^{\infty}e^{-y^2}dy\right)\\
&=\ln(\sqrt{2\pi}\sigma)+\frac{1}{2}=\frac{1}{2}\left(\ln\left(2\pi\sigma^2\right)+1\right)
\end{aligned}\tag{184}$$

由此我们得到高斯分布的熵随其方差单调递增. 设 P_i 为 p_{ij} 对所有 j 的分布, 由定义显然满足 $\sum_{j=1}^{N}p_{ij}=1$. 而 P_i 可看做离散化的高斯分布, 可以推出 P_i 的熵 $H(P_i)$ 也随 σ_i 单调递增.SNE 引入困惑度的概念,(perplexity) 的概念, 困惑度是指:

$$Perp\left(P_i\right)=2^{H(P_i)}\tag{185}$$

这里的 $H\left(P_i\right)$ 是 P_i 的熵, 困惑度可以解释为一个点附近的有效近邻点个数. 我们对每个分布 P_i 都给定困惑度 $Perp$, 之后使用二分搜索的方式寻找合适的 σ(即解 N 个 (185) 的方程).SNE 对困惑度的调整比较有鲁棒性, 通常选择 5-50 之间, .(我未能理解如何给定每个 P_i 的困惑度, 想到的一种解释是给定 Perp 比直接给定 σ_i 更有鲁棒性)

那对于降维后的 $y^{(i)}$, 我们可以指定高斯分布为方差为 $\frac{1}{2}$, 这实际上只影响最终输出图像的放缩程度. 定义降维后的 $y^{(j)}$ 是 $y^{(i)}$ 的近邻的概率 q_{ij} 为

$$q_{ij}=\frac{\exp\left(-\left\|y^{(i)}-y^{(j)}\right\|^2\right)}{\sum_{k\neq i,k=1}^{N}\exp\left(-\left\|y^{(i)}-y^{(k)}\right\|^2\right)}\tag{186}$$

同样, 设定 $q_{ii}=0$.

SNE 希望 p_{ij} 和 q_{ij} 尽可能地接近, 这样可以尽可能地保留点与点之间在概率上的相似度. 设 Q_i 为 q_{ij} 对所有 j 的分布, 也有 $\sum_{j=1}^{N}q_{ij}=1$. 我们采用 KL 散度 (具体见 (242) 附近) 来衡量 P_i 与 Q_i 之间的差异, 那么目标函数为 N 个 KL 散度之和:

$$C=\sum_{i=1}^{N}KL\left(P_i\|Q_i\right)=\sum_{i=N}^{N}\sum_{j=1}^{N}p_{ij}\log\frac{p_{ij}}{q_{ij}}\tag{187}$$

思考这个目标函数, 每一项都和 p_{ij} 和 q_{ij} 有关, 对于给定的 p_{ij}, 每个 $q_[ij]$ 都是越大越好 (使损失函数越小), 但 $\sum_{j=1}^{N}q_{ij}=1$ 又对 q_{ij} 做了限制, 故 q_{ij} 倾向于优先满足 p_{ij} 本身较大的项. 从另外一个角度考虑, 损失

函数每一项和 p_{ij} 和 $\frac{p_{ij}}{q_{ij}}$ 正相关, 对于较大的 p_{ij},$\frac{p_{ij}}{q_{ij}}$ 更需要变小, 而对于较小的 p_{ij},$\frac{p_{ij}}{q_{ij}}$ 显得不那么重要 (当然也是越小越好, 但因为有限制, 所以往往不能够被满足). 这种性质是由 KL 散度的不对称性导致的.

损失函数 (187) 对 $y^{(i)}$ 求导较复杂, 我们将其分解为链式法则, 设 $f_{ij}=\left\|y^{(i)}-y^{(j)}\right\|^2$,$w_{ij}=e^{-f_{ij}}$, 所以有

$$q_{ij}=\frac{w_{ij}}{\sum_{k\neq i,k=1}^{N}w_{ik}} \tag{188}$$

并且满足 $f_{ij}=f_{ji}$. 由此可以对损失函数关于 $y^{(i)}$ 求导

$$\begin{aligned}\frac{\partial C}{\partial y^{(i)}}&=\sum_{j=1}^{N}\sum_{l=1}^{N}\frac{\partial C}{\partial q_{ij}}\frac{\partial q_{ij}}{\partial y^{(i)}}\\&=\sum_{j=1}^{N}\left(\sum_{l=1}^{N}\frac{\partial C}{\partial q_{jl}}\frac{\partial q_{jl}}{\partial w_{ji}}\frac{\partial w_{ji}}{\partial f_{ji}}+\sum_{l=1}^{N}\frac{\partial C}{\partial q_{il}}\frac{\partial q_{il}}{\partial w_{ij}}\frac{\partial w_{ij}}{\partial f_{ij}}\right)\frac{\partial f_{ij}}{\partial y^{(i)}}\\&=\sum_{j=1}^{N}\left(u_{ji}+u_{ij}\right)\frac{\partial f_{ij}}{\partial y^{(i)}}\end{aligned} \tag{189}$$

接下来只需要求 u_{ij} 即可. 因为 $q_{ij}=\frac{w_{ij}}{\Sigma_k^N w_{ik}}=\frac{w_{ij}}{S_i}$, 所以 $\frac{q_{ij}}{w_{ij}}=\frac{1}{S_i}-\frac{q_{ij}}{S_i}$ 且 $\frac{q_{ik}}{w_{ij}}=-\frac{q_{ik}}{S_i}$ 将二者代入 u_{ij} 得:

$$u_{ij}=\frac{1}{S_i}\left[\frac{\partial C}{\partial q_{ij}}-\sum_{l=1}^{N}\frac{\partial C}{\partial q_{il}}q_{il}\right]\frac{\partial w_{ij}}{\partial f_{ij}} \tag{190}$$

$\frac{\partial C}{\partial q_{ij}}=-\frac{p_{ij}}{q_{ij}}$ $\frac{\partial w_{ij}}{\partial f_{ij}}=-w_{ij}$ 所以

$$u_{ij}=\frac{1}{S_i}\left(-\frac{p_{ij}}{q_{ij}}-\sum_{l=1}^{N}-\frac{p_{ij}}{q_{il}}q_{il}\right)(-w_{ij})=q_{ij}\left(\frac{p_{ij}}{q_{ij}}-1\right)=p_{ij}-q_{ij} \tag{191}$$

代入 $\frac{\partial C}{\partial y^{(i)}}$ 得:

$$\begin{aligned}\frac{\partial C}{\partial y^{(i)}}&=\sum_{j=1}^{N}\left(k_{ji}+k_{ij}\right)\frac{\partial f_{ij}}{\partial d_{ij}}\frac{\partial d_{ij}}{\partial y^{(i)}}\\&=\sum_{j=1}^{N}\left(p_{ij}-q_{ij}+p_{ji}-q_{ji}\right)*2\left(y^{(i)}-y^{(j)}\right)\\&=2\sum_{j=1}^{N}\left(p_{ij}-q_{ij}+p_{ji}-q_{ji}\right)\left(y^{(i)}-y^{(j)}\right)\end{aligned} \tag{192}$$

从而得到损失函数关于 $y^{(i)}$ 的偏导数:

$$\frac{\partial C}{\partial y^{(i)}}=2\sum_{j=1}^{N}\left(y^{(i)}-y^{(j)}\right)\left(p_{ij}-q_{ij}+p_{ji}-q_{ji}\right) \tag{193}$$

由此我们可以根据梯度对损失函数进行优化, 可以用较小的 σ 下的高斯分布来进行初始化. 为了加速优化过程和避免陷入局部最优解, 论文中使用了带动量的 SGD. 即参数更新中除了当前的梯度, 还要引入之前的梯度累加的指数衰减项, 如下:

$$Y^{(t)}=Y^{(t-1)}-\eta\frac{\partial C}{\partial Y}+\alpha(t)\left(Y^{(t-1)}-Y^{(t-2)}\right)$$

这里的 $Y^{(t)}$ 表示迭代 t 次的解, η 表示学习速率, $\alpha(t)$ 表示迭代 t 次的动量. 论文提到, 除了在初始优化的阶段, 每次迭代中可以引入一些高斯噪声, 之后像模拟退火一样逐渐减小该噪声, 可以用来避免陷入局部最优解. 因此, SNE 在选择高斯噪声, 以及学习速率, 何时开始衰减, 动量选择等等超参数上, 需要多次运行才找到最优解. 当然这些都是一般求解非凸优化问题的过程.

对于损失函数的梯度 (193), 作者在论文中用了一个弹簧的比喻来解释. 可以把该公式看做是所有 y_j 通过弹簧对 y_i 的合力, 类比胡克定律 $F = k\Delta x$, 这里 $(y_i - y_j)$ 表示拉伸长度, 而 $(p_{ji} - q_{ji} + p_{ij} - q_{ij})$ 表示弹性系数. 表示高维数据点和低维映射点之间点相似度的失配程度 (论文中写为 mismatch between the pairwise similarities of the data points and the map points).

3.5.2 Symmetric SNE 和 t-SNE

t-SNE 对 SNE 的两个主要问题分别提出了改进方案.

首先是 SNE 的损失函数较难优化, 在 SNE 中, $p_{ij} \neq p_{ji}$ 且 $q_{ij} \neq q_{ji}$. 如果能得出一个更加通用的联合概率分布更加合理,cook 等人在 2007 年提出了 Symmetric SNE, 分别在高维和低维空间构造联合概率分布 P 和 Q 使得对任意 i, j, 均有 $\hat{p}_{ij} = \hat{p}_{ji}, \hat{q}_{ij} = \hat{q}_{ji}$. 且有 $\hat{p}_{ii} = 0$,$\hat{q}_{ii} = 0$. 则我们可以定义联合分布 $\hat{P}$ 和 $\hat{Q}$ 分别为

$$\hat{p}_{ij} = \frac{\exp\left(-\left\|x^{(i)} - x^{(j)}\right\|^2 / 2\sigma^2\right)}{\sum_{k=1}^{N}\sum_{l=1,l\neq k}^{N}\exp\left(-\left\|x^{(k)} - x^{(l)}\right\|^2 / 2\sigma^2\right)} \tag{194}$$

$$\hat{q}_{ij} = \frac{\exp\left(-\left\|y^{(i)} - y^{(j)}\right\|^2\right)}{\sum_{k=1}^{N}\sum_{l=1,l\neq k}^{N}\exp\left(-\left\|y^{(k)} - y^{(l)}\right\|^2\right)} \tag{195}$$

然而因为所有 $\hat{p}_{ij}$ 的分母都一样, 这样设定会带来异常值的问题. 假设点 $x^{(i)}$ 是一个噪声点, 那么 $\left\|x^{(i)} - x^{(j)}\right\|^2$ 会很大, 那么对于所有的 j,p_{ij} 的值都会很小, 导致在低维映射下的 $y^{(i)}$ 对整个损失函数的影响很小, 但对于异常值, 我们需要得到一个更大的惩罚. 所以我们抛弃上面的 $\hat{P}$,($\hat{Q}$ 不改变). 将高维空间中的联合概率修正为

$$\hat{p}_{ij} = \frac{p_{ij} + p_{ji}}{2N} \tag{196}$$

其中的 p_{ij}, p_{ji} 是 SNE 中的设定, 显然这样的 $\hat{P}$ 也满足联合概率的定义和 $\hat{p}_{ji} = \hat{p}_{ji}$. 根据这样的设定, 我们得到 Symmetric SNE 的损失函数

$$\hat{C} = KL\left(\hat{P}\|\hat{Q}\right) = \sum_{i=N}^{N}\sum_{j=1}^{N}\hat{p}_{ij}\log\frac{\hat{p}_{ij}}{\hat{q}_{ij}} \tag{197}$$

求梯度的方式与 SNE 相同:

$$\frac{\delta\hat{C}}{\delta y^{(i)}} = 4\sum_{j=1}^{N}\left(p_{ij} - q_{ij}\right)\left(y^{(i)} - y^{(j)}\right) \tag{198}$$

这比 SNE 的梯度更简单, 但是实验效果却没有明显的更好.

另一个需要解决的问题就是拥挤问题,crowding problem. 即不同类别的簇挤在一起, 无法区分开来, 这就是拥挤问题. 或者说高维空间中相距一定距离的点, 在低维空间中” 塌缩” 为很近的点或者同一个点. 实际上, 拥挤问题的出现与某个特定算法无关, 而是由于高维空间距离分布和低维空间距离分布的差异造成的. 论文中举例来说, 比如一个 10 维的流形嵌入到更高维度的空间中, 现在我们的问题是把这个 10 维的流形找出来, 并且映射到二维空间上可视化. 在进行可视化时, 问题就来了, 在 10 维流形上可以存在 11 个点且两两之间距离相等. 而二维空间中最多只有三个点两两之间距离相等, 想将高维空间中的距离关系完整保留到低维空间是不

可能的. 论文通过一个实验进一步说明, 在 m(m 很大) 维空间中, 假设一个以数据点 $x^{(i)}$ 为中心, 半径为 r 的 m 球, 其体积是按 r^m 增长的, 假设数据点是在 m 维球中均匀分布的, 则大部分数据点都聚集在 m 维球的表面附近, 与点 $x^{(i)}$ 的距离分布极不均衡. 如果直接将这种距离关系保留到二维, 则会出现拥挤问题.

换句话说, 之前的降维方法希望将数据降维到它实际的维度 (或大致的维度), 而可视化的要求是强行将数据降维到 1-3 维, 所以拥挤问题才较为明显.

为解决拥挤问题, 我们引入 t 分布. 定义为: 假设 $X \sim N(0,1)$,$Y \sim \chi^2(n)$ 分布, 那么 $Z = \frac{X}{\sqrt{\frac{Y}{n}}}$ 的分布称为自由度为 n 的分布, 记为 $Z \sim t(n)$ .t 分布比正态分布要" 胖" 一些, 尤其在尾部两端较为平缓.t 分布是一种典型的长尾分布, 在处理小样本和一些异常点的时候较稳定.

在高维空间下, 在高维空间下我们使用高斯分布将距离转换为概率分布, 在低维空间下, 我们使用更加偏重长尾分布的方式来将距离转换为概率分布, 使得高维度下中低等的距离在映射后能够有一个较大的距离.

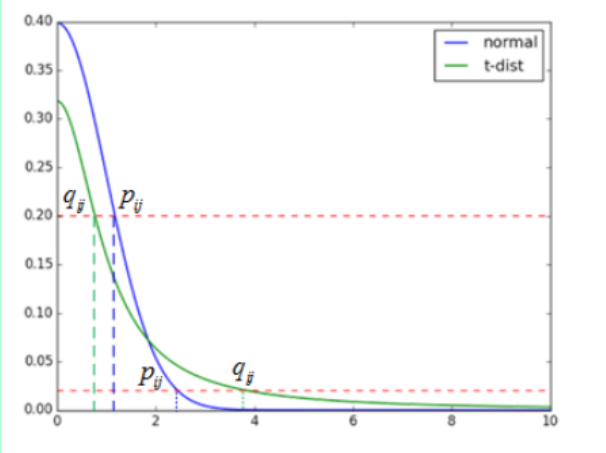


图 6: t 分布在 t-SNE 中的作用

图中有高斯分布和 t 分布两条曲线, 表示点之间的相似度与距离的关系, 横轴是 (高维或低维) 两点间距离, 高斯分布对应高维空间,t 分布对应低维空间, 那么对于高维空间中相距较近的两点, 为了满足 $p_{ij} = q_{ij}$, 如图所示, 两点在低维空间中的距离应该更小. 而对于高维空间中相距较远的两点, 为了满足 $p_{ij} = q_{ij}$, 如图所示, 两点在低维空间中的距离应该更远. 这利用了长尾分布的特性, 恰好满足了我们的需求, 使不同簇之间相互疏远, 避免了拥挤问题. 我们使用自由度为 1 的 t 分布重新定义联合概率 $\hat{Q}$

$$\hat{q}_{ij} = \frac{\left(1 + \left\|y^{(i)} - y^{(j)}\right\|^2\right)^{-1}}{\sum_{k=1}^{N}\sum_{l=1,l\neq k}^{N}\left(1 + \left\|y^{(k)} - y^{(l)}\right\|^2\right)^{-1}} \tag{199}$$

显然满足联合概率的定义.$\hat{P}$ 保持不变, 仍用 KL 散度衡量两个分布之间的相似性, 得到 t-SNE 的损失函数

$$\tilde{C} = KL\left(\hat{P}\|\hat{Q}\right) = \sum_{i=N}^{N}\sum_{j=1}^{N}\hat{p}_{ij}\log\frac{\hat{p}_{ij}}{\hat{q}_{ij}} \tag{200}$$

梯度推导和 SNE 类似, 也较复杂, 不过只需要链式法则的知识

$$\frac{\delta\tilde{C}}{\delta y^{(i)}} = 4\sum_{j=1}^{N}\left(p_{ij} - q_{ij}\right)\left(y^{(i)} - y^{(j)}\right)\left(1 + \left\|y^{(i)} - y^{(j)}\right\|^2\right)^{-1} \tag{201}$$

即 t-SNE 的梯度是 Symmetric SNE 增加了 $\left(1+\left\|y^{(i)}-y^{(j)}\right\|^2\right)^{-1}$ 一项, 之后再利用优化算法训练即可.

2014 年 Maaten 又写了一篇论文对 t-SNE 算法进行了改进, 使用了各种基于树的算法, 具体包括两部分内容: 一是采用了 KNN 图来表示高维空间中点的相似性；二是优化了梯度的求解过程, 将梯度计算分为引力和斥力两部分, 同样使用了一些优化技巧.

4 聚类模型

聚类分析计算方法主要有: 划分方法、层次方法、基于密度的方法等

4.1 K-means 聚类

4.1.1 基本 K-means

K-means 是一种基于划分的方法, 给定一个数据集:$D=\left\{x^{(i)}\right\}_{i=1}^{N}$, 希望确定一种集合的划分方式, 将集合 D 分解为 K 个互不相交的非空集合 D_i 的并集 (称为 K 个类或 K 个簇), 使其满足:

$$\begin{aligned}&\bigcup_{k=1}^{K} D_k = D\\ &D_k \bigcap D_j = \phi(k \neq j)\end{aligned} \tag{202}$$

我们在 D 的非空子集定义一个集函数 $\rho(D_k)$, 并通过该函数构造一个目标函数, 通过优化该目标函数实现聚类. 一般情况下 K-means 考虑基于欧式距离的聚类, 设集函数为

$$\rho\left(D_k\right)=\sum_{x^{(i)} \in D_k}\left(x^{(i)}-c_k\right)^T\left(x^{(i)}-c_k\right) \tag{203}$$

即求类 D_k 中的所有元素到中心的欧式距离的平方和, 其中 c_k 是 D_k 中的所有元素的中心, 即

$$c_k=\frac{1}{\left|D_k\right|} \sum_{x_i \in D_k} x_i \tag{204}$$

其中 $|D_k|$ 表示集合 D_k 中的元素个数. 由此我们得到 K-means 的损失函数:

$$L(C_m)=\sum_{k=1}^{K} \rho\left(D_k\right)=\sum_{k=1}^{K} \sum_{x^{(i)} \in D_k}\left\|x^{(i)}-c_k\right\|_2^2 \tag{205}$$

其中 C_m 为 D 的一种划分,m 为迭代次数.

直接给出 K-means 的伪代码,

---

algorithm 1 K-means

---

input: $\left\{x^{(i)}\right\}_{i=1}^{N}$,$x^{(i)} \in \mathbb{R}^n$

output: $\left\{\left(c_k,\left\{x^{(i)}\right\}_{i=1}^{N_k}\right)\right\}_{k=1}^{k=K}$

初始化 K 个中心,$\{c_k\}_{k=1}^{K}$.

repeat

(a) 计算所有样本点到中心的距离, 每个样本点归入最近的中心点所对应的类中

(b) 重新计算每个类的中心点

until

$C_m = C_{m-1}$ 或没有中心发生变化 (两者等价)

显然 K-means 算法是可行的, 下面证明其收敛性. 首先证明损失函数的单调递减性, 即随着迭代步数增多, 损失函数的值单调递减 (不一定严格递减). 设 $x_1, x_2 \ldots x_n$ 是欧式空间的 n 个向量, 则 $\sum_{i=1}^{n}\left(x_i-x\right)'\left(x_i-x\right)$

取到最小值当且仅当 x 是这 n 个向量的中心位置, 即 $x=\frac{1}{n}\sum_{i=1}^{n}x_i$ 证明很简单,

$$\frac{\partial\sum_{i=1}^{n}\left\|x^{(i)}-x\right\|_2^2}{\partial x}=-2\left(x^{(i)}-x\right)=0 \tag{206}$$

$x=\frac{1}{n}\sum_{i=1}^{n}x_i$ 是该函数的一个驻点. 显然该目标函数是严格凸的, 所以 $x=\frac{1}{n}\sum_{i=1}^{n}x_i$ 是目标函数的唯一最小值点. 这就说明, 对目标函数 (205), 单次迭代内有 (a) 到 (b) 不增 (若有某个中心位置发生了变化, 那么目标函数值严格较小). 此外, 由前一步的 (b) 到后一步的 (a), 目标函数显然也不增, 故经过一次迭代后, 目标函数关于迭代次数单调递减.

接下来证明 K-means 的收敛性, 也就是划分 C 的收敛性. 对于从任意一个初始方式开始, 不断进行迭代, 就会得到对数据集的一列划分: $C_1,C_2,C_3\ldots$, 同时对应地得到一列目标函数值 $f\left(C_1\right),f\left(C_2\right),f\left(C_3\right)\ldots$. 由前文所述, 该数列 $\{f\left(C_m\right)\}$ 单调递减且显然有 $f(C)\geq 0$., 由单调有界数列的收敛定理知, $\{f\left(C_m\right)\}$ 收敛, 即 $\lim_{n\to\infty}f\left(C_m\right)$ 存在, 这就意味着, 随着算法迭代次数增加, 目标函数一定会收敛. 对有限个数据点, 其 K-完全分类的情况是有限的, 结合目标函数的单调递减性, 我们得到结论: 经过有限次迭代后, 目标函数值与中心点位置都会恒为常量, 即该算法迭代过程不是一直无限逼近极小值点的过程, 而是经过有限步逼近后, 必然会严格等于极小值点, 此后再进行迭代, 划分方法、中心点、目标函数等都不会再改变. 并且由于对于给定的初值, 每一步的过程是完全确定的, 不含随机因素. 所以对于给定初值, 聚类结果是唯一确定的. 可以给出逼近步数一个粗略的上限是 C_N^K.

但是, 虽然对于给定的初值, 算法可以保证收敛, 但是对于不同的初值选取情况, 算法收敛到的结果可能是不一样的. 显然, 对于不同的类别数 K, 聚类结果必然不同. 所以初始中心位置与类别数 K 是该算法重要的超参数.

对于 K 的选取, 一般有如下的思想:

1) 数据的先验知识, 或者数据进行简单分析能得到 k 的大致取值.

2) 基于变化的算法: 即定义一个函数, 随着 K 的改变, 认为在正确的 K 时会产生极值. 如 Gap statistic 方法, 是通过对平均分布的参考数据集的期望值 $E(log(f(T)))$ 和观测数据集的 $log(f(T))$ 进行比较, 使得 $log(f(T))$ 下降最快的 k 值为最优聚类数.

3) 基于结构的算法: 即比较类内距离、类间距离以确定 K. 这个也是最常用的办法, 如轮廓系数法等.

4) 如 ISODATA 等变体, 根据簇内方差和簇间距离等参数对初始给定的 K 进行调整.

对于初始 K 个中心的选取, 通常采取两种方法:

1) 选择批次距离尽可能远的 K 个点. 首先随机选择一个点作为第一个初始类簇中心点, 然后选择距离该点最远的那个点作为第二个初始类簇中心点, 然后再选择距离前两个点的最近距离最大的点作为第三个初始类簇的中心点, 以此类推, 直至选出 K 个初始类簇中心点.(这实际上是 K-means 的一个变体 K-means++)

2) 先对数据用层次聚类算法进行聚类, 得到粗略的 K 个簇之后, 再使用 K-means.(如二分 K-means)

K-means 所需的存储量为 $O(N+K)n$, 聚类的时间复杂度为 $O(mKNn)$

最后, 我们对 K-means 的优缺点进行总结. 优点: 简单有效. 缺点: 不能处理非球形簇、不同尺寸的簇、不同密度的簇；且要求聚类问题在度量标准下必须有存在质心.

4.1.2 二分 K-means

二分 K-means 的聚类属于层次聚类, 是一种分裂聚类, 效果相对粗略, 不常用来做聚类. 在问题规模较大时, 我们通常用二分 K-means 的结果作为 K-means 的预处理.

二分 K-means 的过程很简单.

algorithm 2 二分 K-means

input: $\left\{x^{(i)}\right\}_{i=1}^{N}$,$x^{(i)} \in \mathbb{R}^n$

output: $\left\{\left(c_k, \left\{x^{(i)}\right\}_{i=1}^{N_k}\right)\right\}_{k=1}^{k=K}$

初始化簇表, 使之包含由所有的点组成的簇.

repeat

根据一定标准, 从簇表中选出一个簇.(选取标准可以是数据点数最多的簇, 也可以使集函数最大的簇, 或者综合评判)

对选定的簇使用 K-means, 将其分为两簇. 将这两个簇添加到簇表中.

until

簇表中有 K 个簇.

二分 K-means 只在每步用到了一次 K-means 显然是收敛的. 并且在二分 K-means 每步的 K-means 中, 由于只选取两个质心, 故较不易受质心选取的影响.

二分 K-means 得到的划分和质心一般不符合 K-means 的要求, 我们可以用二分 K-means 的 K 个质心作为初始质心, 再用 K-means 逐步求精.

4.1.3 Fuzzy K-means

Fuzzy K-means 依据模糊理论, 相较于 K-means 的硬聚类,Fuzzy K-means 提供了更加灵活的聚类结果. 因为大部分情况下, 数据集中的数据不能划分成为明显分离的簇, 指派一个对象到一个特定的簇有些生硬, 也可能会出错. 故对每个对象和每个簇赋予一个权值, 指明对象属于该簇的程度. 用隶属度代替了原来强硬的划分, 使得边缘更加平滑. 并没有改变原来迭代的思路, 还是依赖于起始点, 和距离的度量.

Fuzzy K-means 的损失函数为

$$L(C_m) = \sum_{k=1}^{K} \rho\left(D_k\right) = \sum_{k=1}^{K}\sum_{i=1}^{N} w_{ik}^{p} \|x^{(i)} - c_k\|_2^2 \tag{207}$$

其中 w_{ik} 表示数据点 $x^{(i)}$ 对于中心 c_k 的隶属度, 满足 $\forall i, \sum_{k=1}^{K} w_{ik} = 1$, 即每个数据点的隶属度之和为 1.$p > 1$ 表示聚类的模糊参数, 是事先给定的常数,m 为迭代次数.

Fuzzy K-means 使用交互式策略求解, 简单来说即给定 c_k 对 L 关于 w_{ik} 求最小, 再给定 w_{ik} 对 L 关于 c_k 求最小.

1) 对于确定的中心 $\{c_k\}_{k=1}^{K}$, 优化问题对于 N 个数据点是独立的, 对每个数据点 $x^{(i)}$, 优化其隶属度

$w_{ik}(k=1,2,\cdots,K)$, 优化问题的形式为:

$$\begin{aligned} &\min L_i(C_m) = \sum_{k=1}^{K} w_{ik}^p \|x^{(i)} - c_k\|_2^2 \\ &\text{s.t.} \sum_{k=1}^{K} w_{ik} = 1, \quad 0 \leqslant w_{ik} \leqslant 1 \end{aligned} \tag{208}$$

显然有 $L(C_m) = \sum_{i=1}^{N} L_i(C_m)$. 我们可以利用拉格朗方法求解, 设

$$F_i(w_{ik}, \lambda) = \sum_{k=1}^{K} w_{ik}^p \|x^{(i)} - c_k\|_2^2 + \lambda \left(\sum_{k=1}^{K} w_{ik} - 1 \right) \tag{209}$$

令 F 对 λ 偏导为 0, 得到:

$$\frac{\partial F_i}{\partial \lambda} = \sum_{k=1}^{K} w_{ik} - 1 = 0 \tag{210}$$

这就是约束条件. 对一个 $j=1,2,\cdots,K$, 令 F 对 w_{ij} 偏导为 0, 得到:

$$\frac{\partial F}{\partial w_{ij}} = p\left(w_{ij}\right)^{p-1} \|x^{(i)} - c_j\|_2^2 - \lambda = 0 \tag{211}$$

从而得到 K 个 w_{ij}:

$$w_{ij} = \left[\frac{\lambda}{p\|x^{(i)} - c_k\|_2^2} \right]^{\frac{1}{p-1}} \tag{212}$$

将该公式代入隶属度的约束条件: $\sum_{k=1}^{K} w_{ik} = 1$, 得到:

$$\sum_{k=1}^{K} w_{ik} = \sum_{k=1}^{K} \left[\frac{\lambda}{p\|x^{(i)} - c_k\|_2^2} \right]^{\frac{1}{p-1}} = \left(\frac{\lambda}{p} \right)^{\frac{1}{p-1}} \left\{ \sum_{k=1}^{K} \left[\frac{1}{\|x^{(i)} - c_k\|_2^2} \right]^{\frac{1}{p-1}} \right\} = 1 \tag{213}$$

即有

$$\left(\frac{\lambda}{p} \right)^{\frac{1}{p-1}} = \frac{1}{\sum_{k=1}^{K} \left[\frac{1}{\|x^{(i)} - c_k\|_2^2} \right]^{\frac{1}{p-1}}} \tag{214}$$

代回 (212) 消去 λ 得到每次更新的 w_{ij} 值:

$$w_{ij} = \frac{\left(\frac{1}{\|x^{(i)} - c_j\|_2^2} \right)^{\frac{1}{p-1}}}{\sum_{k=1}^{K} \left(\frac{1}{\|x^{(i)} - c_k\|_2^2} \right)^{\frac{1}{p-1}}} = \frac{1}{\sum_{k=1}^{K} \left(\frac{\|x^{(i)} - c_j\|_2}{\|x^{(i)} - c_k\|_2} \right)^{\frac{2}{p-1}}} \tag{215}$$

由此得到 $N \times K$ 个 w_{ij} 的更新值.

2) 对于更新的 w_{ik} 求解聚类中心 c_k: 令 $\nabla_{c_k} L(C_m) = \vec{0}$, 得到:

$$\nabla_{c_k} L(C_m) = \nabla_{c_k} \left(\sum_{k=1}^{K} \sum_{i=1}^{N} w_{ik}^p \|x^{(i)} - c_k\|_2^2 \right) = 0 \tag{216}$$

即 $\forall k=1,2,\cdots,K$, 都有 $\frac{\partial L(C_m)}{c_k}=0$, 即

$$\begin{aligned}\frac{\partial L(C_m)}{c_k}&=\sum_{i=1}^{N}(w_{ik})^p\left[-2\left(x^{(i)}-c_k\right)\right]\\&=-2\left[\sum_{i=1}^{N}(w_{ik})^p\left(x^{(i)}-c_k\right)\right]\\&=-2\left[\sum_{i=1}^{N}(w_{ik})^p x^{(i)}-\sum_{i=1}^{N}(w_{ik})^p c_k\right]=0\end{aligned}\tag{217}$$

从而更新 K 个质心位置

$$c_k=\frac{\sum_{i=1}^{N}(w_{ik})^p x^{(i)}}{\sum_{i=1}^{N}(w_{ik})^p}\tag{218}$$

这也符合模糊情况下质心的定义.

特别地, 当 (208) 中 $p=1$ 且 $w_{ik}\in\{0,1\}$ 时, 即为 K-means. 易见, 随着 p 增大, 划分变得模糊, 簇中心趋向于全局中心. 随着 p 趋向于 1, 划分趋向于 K-means.

Fuzzy K-means 的收敛性证明见论文,Hathaway R J and Bezdek J C. Recent convergence results for the fuzzy c-means clustering algorithms, Journal of Classification, 1998, 5: 237-247

组里的论文还提出基于具有流形结构的收缩模式的 FKPS 和其投影版本 PFKPS 以处理高维聚类.

4.2 谱聚类

谱聚类的思想是将样本看作顶点, 样本间的相似度看作带权的边, 从而将聚类问题转为图分割问题: 找到一种图分割的方法使得连接不同组的边的权重尽可能低 (这意味着组间相似度要尽可能低), 组内的边的权重尽可能高 (这意味着组内相似度要尽可能高), 从而达到聚类的目的.

谱聚类就是通过切割无向图来实现聚类. 对于无向图 G 的切图, 我们的目标是将图 $G(V,E)$ 切成相互没有连接的 K 个子图, 每个子图点的集合为: $V_1,V_2,\ldots V_K$, 它们满足 $V_i\cap V_j=\varnothing$, 且 $V_1\cup V_2\cup\ldots\cup V_K=V$ 对于任意两个子图点的集合 $A,B\subset V,A\cap B=\varnothing$, 我们定义 A 和 B 之间的切图权重为:

$$W(A,B)=\sum_{i\in A,j\in B}w_{ij}\tag{219}$$

那么对于我们 K 个子图点的集合: $V_1,V_2,\ldots V_K$, 我们定义切图的损失函数为:

$$Cut\left(V_1,V_2,\ldots V_K\right)=\frac{1}{2}\sum_{i=1}^{K}W\left(V_i,\bar{V_i}\right)\tag{220}$$

其中 $\bar{V_i}$ 为 V_i 的补集, 意为除 V_i 子集外其他 V 的子集的并集. 我们希望对 G 的切图的损失函数求最优, 但是 L 的定义使其容易只切割边缘单个的节点, 不满足聚类模型的需要, 故损失函数进行调整. 常用的有 RatioCut 和 NCut.

RatioCut 切图, 对每个切图, 不光考虑最小化 $L\left(V_1,V_2,\ldots V_K\right)$, 它还同时考虑最大化每个子图点的个数, 即:

$$RatioCut\left(V_1,V_2,\ldots V_K\right)=\frac{1}{2}\sum_{i=1}^{K}\frac{W\left(V_i,\bar{V_i}\right)}{|V_i|}\tag{221}$$

为解决这个最优化问题, 我们引入指示向量 $h_j \in \{h_1, h_2, \dots h_K\}, j = 1, 2, \dots K$, $h_j \in \mathbb{R}^N$. 定义 $H = (h_{ij})_{N \times K}$ 为:

$$h_{ij} = \begin{cases} 0 & v_i \notin V_j \\ \frac{1}{\sqrt{|V_j|}} & v_i \in V_j \end{cases} \tag{222}$$

由拉普拉斯矩阵的性质, 有:

$$\begin{aligned} h_i^T L h_i &= \frac{1}{2} \sum_{m=1} \sum_{n=1} w_{mn} \left(h_{im} - h_{in}\right)^2 \\ &= \frac{1}{2} \left(\sum_{m \in V_i, n \notin V_i} w_{mn} \left(\frac{1}{\sqrt{|V_i|}} - 0 \right)^2 + \sum_{m \notin V_i, n \in V_i} w_{mn} \left(0 - \frac{1}{\sqrt{|\ V_i\ |}} \right)^2 \right. \\ &= \frac{1}{2} \left(\sum_{m \in V_i, n \notin V_i} w_{mn} \frac{1}{|V_i|} + \sum_{m \notin V_i, n \in V_i} w_{mn} \frac{1}{|V_i|} \right. \\ &= \frac{1}{2} \left(Cut\left(V_i, \bar{V}_i\right) \frac{1}{|V_i|} + Cut\left(\bar{V}_i, V_i\right) \frac{1}{|V_i|} \right) \\ &= \frac{Cut\left(V_i, \bar{V}_i\right)}{|V_i|} \end{aligned} \tag{223}$$

所以 K 个子图对应的 RatioCut 函数表达式为:

$$RatioCut\left(V_1, V_2, \dots V_K\right) = \sum_{i=1}^{K} h_i^T L h_i = \sum_{i=1}^{K} \left(H^T L H\right)_{ii} = tr\left(H^T L H\right) \tag{224}$$

其中 $tr\left(H^T L H\right)$ 为矩阵的迹. 故 RatioCut 切图, 实际上就是根据 H 的定义最小化 $tr\left(H^T L H\right)$.

但由于 h_{ij} 的定义,H 有 K^N 种取法, 如果 N 不很大, 可以遍历求解. 但通常 $N > 20$, 此时这种 NP 问题不易解. 由 H 的定义注意到 $H^T H = I$, 如果忽略 H 的定义, 则我们可以简化约束条件, 由此切图优化目标为:

$$\begin{aligned} &\arg\min_{H} tr\left(H^T L H\right) \\ &\text{s.t. } H^T H = I \end{aligned} \tag{225}$$

注意到这个优化问题和 PCA 中的优化问题 (106) 基本一致, 只是目标函数分别是最小和最大. 同样使用拉格朗日乘子法, 我们通过找到 L 的最小的 K 个特征值, 可以得到对应的 K 个特征向量, 这 K 个特征向量组成一个 NxK 的矩阵, 即为 H .

此外, 还需要考虑一个问题, 由性质 3), 零特征值对应的特征向量的各分量相同, 对指示向量没有任何信息, 同时, 零特征值相当于不做切割, 不符合要求, 故我们应选择非零的最小的 K 个特征值对应的 K 个特征向量.

解决这个优化问题的复杂度大致是 $O(kn^3)$. 一般需要对得到的 H 矩阵按行做标准化, 使其列向量 h_i 都为单位向量, 即

$$h_{ij}^* = \frac{h_{ij}}{\left(\sum_{k=1}^{K} h_{ik}^2\right)^{1/2}} \tag{226}$$

由于得到的 H 和定义不符, 不能完全指示各样本的归属. 研究表明, 谱聚类得到的指示向量的矩阵 H 可以看做将数据从高维做了特征提取, 每个数据点的特征对应 H 的 k 维行向量, 因此一般在得到 NxK 维度的矩阵 H 后还需要对每一行进行一次传统的聚类, 比如使用 K-Means 聚类.

Ncut 切图和 RatioCut 切图很类似, 但是把 Ratiocut 的分母 $|Vi|$ 换成 $d(V_i)$. 由于子图样本的个数多并不一定权重就大, 切图时基于权重可能更适合我们的目标, 因此一般 Ncut 切图优于 RatioCut 切图.

$$NCut(V_1, V_2, \dots V_k) = \frac{1}{2}\sum_{i=1}^{k}\frac{W(V_i, \bar{V}_i)}{d(V_i)} \tag{227}$$

对应的, Ncut 切图对指示向量 h 做了改进. 注意到 RatioCut 切图的指示向量使用的是 $\frac{1}{\sqrt{|V_j|}}$ 标示样本归属, 而 Ncut 切图使用了子图权重 $\frac{1}{\sqrt{d(V_j)}}$ 来标示指示向量 h, 定义如下:

$$h_{ij} = \begin{cases} 0 & v_i \notin V_j \\ \frac{1}{\sqrt{vol(V_j)}} & v_i \in V_j \end{cases} \tag{228}$$

与 RatioCut 推导相同:

$$\begin{aligned} h_i^T L h_i &= \frac{1}{2}\sum_{m=1}\sum_{n=1} w_{mn}(h_{im} - h_{in})^2 \\ &= \frac{1}{2}\left(\sum_{m\in V_i, n\notin V_i} w_{mn}\left(\frac{1}{\sqrt{d(V_i)}} - 0\right)^2 + \sum_{m\notin V_i, n\in V_i} w_{mn}\left(0 - \frac{1}{\sqrt{d(V_i)}}\right)^2\right. \\ &= \frac{1}{2}\left(\sum_{m\in V_i, n\notin V_i} w_{mn}\frac{1}{d(V_i)} + \sum_{m\notin V_i, n\in V_i} w_{mn}\frac{1}{d(V_i)}\right. \\ &= \frac{1}{2}\left(cut(V_i, \bar{V}_i)\frac{1}{d(V_i)} + cut(\bar{V}_i, V_i)\frac{1}{d(V_i)}\right) \\ &= \frac{cut(V_i, \bar{V}_i)}{d(V_i)} \end{aligned} \tag{229}$$

优化目标仍然是

$$NCut(V_1, V_2, \dots V_k) = \sum_{i=1}^{k} h_i^T L h_i = \sum_{i=1}^{k}\left(H^T L H\right)_{ii} = tr\left(H^T L H\right) \tag{230}$$

但是此时我们的 $H^T H \neq I$, 而是 $H^T D H = I_\circ$. 推导如下:

$$h_i^T D h_i = \sum_{j=1}^{n} h_{ij}^2 d_j = \frac{1}{d(V_i)}\sum_{j\in V_i} d_j = \frac{1}{d(V_i)} d(V_i) = 1 \tag{231}$$

即此时优化目标为:

$$\begin{aligned} &\arg\min_H tr\left(H^T L H\right) \\ &\text{s.t. } H^T D H = I \end{aligned} \tag{232}$$

此时需要将指示向量矩阵 H 做一个变换. 我们令 $H = D^{-1/2}F$, 从而 $H^T L H = F^T D^{-1/2} L D^{-1/2} F, H^T D H = F^T F = I$, 也就是说优化目标变成:

$$\begin{aligned} &\arg\min_F tr\left(F^T D^{-1/2} L D^{-1/2} F\right) \\ &\text{s.t. } F^T F = I \end{aligned} \tag{233}$$

可以发现这个式子和 RatioCut 基本一致, 只是中间的 L 变成了 $D^{-1/2}LD^{-1/2}$. 这样可以继续安照 RatioCut 的思想, 求出 $D^{-1/2}LD^{-1/2}$ 的最小的前 k 个特征值, 然后求出对应的特征向量, 并标准化, 得到最后的特征矩阵

F, 进而由 $H = D^{-1/2}F$ 得到指示向量矩阵 H, 最后, 同样对 H 的行向量进行一次传统的聚类. $D^{-1/2}LD^{-1/2}$ 相当于对拉普拉斯矩阵 L 做了一次标准化, 即 $\frac{L_{ij}}{\sqrt{d_{i*}d_j}}$, 意味着对每个元素用数据点的度归一化.

总结一下, 谱聚类的算法流程:

1) 计算拉普拉斯矩阵 L, 其中 $L = D - W$;

2) 对 L 矩阵标准化, 即令 $L = D^{-1/2}LD^{-1/2}$;

3) 计算 L 矩阵的前 k 个最小非零特征值对应的特征向量, 按列构成矩阵 H;

4) 对数据点在新的特征空间上聚类, 即对 H 的行向量进行聚类;

综上, 谱聚类从切图的角度出发, 但实际上通过 Laplacian 特征映射的方式降维之后再做 K-means(或其他传统聚类) 的一个过程. 还有一个遗留的问题, 就是我们上面所用的 K 是我们希望将 G 切分成的子图个数, 实际上却成为了 Laplacian 特征映射降维的维数, 和最终的聚类的结果不一致. 但是我们可以采取和 LDA 类似的想法, 如果我们推测这个聚类问题可能分为 K 类, 我们可以将其降维至 $K-1$ 维, 当然这两者并没有直接关联, 降维的维数也可以作为一个可调的参数.

结合上述推导和分析, 谱聚类的主要优点有: 1) 谱聚类只需要数据之间的相似度矩阵, 因此对于处理稀疏数据的聚类很有效. 这点传统聚类算法比如 K-Means 很难做到. 2) 由于使用了降维, 因此在处理高维数据聚类时的敗杂度比传统聚类算法好. 谱聚类算法的主要缺点有: 1) 如果最终聚类的维度非常高, 则由于降维的幅度不够, 谱聚类的运行速度和最后的聚类效果均不好. 2) 聚类效果依赖于相似矩阵, 不同的相似矩阵得到的最终聚类效果可能很不同.

传统的聚类已经比较成功的解决了低维数据的聚类问题. 但是由于实际应用中数据的复杂性, 在处理许多问题时, 现有的算法经常失效, 特别是对于高维数据和大型数据的情况. 因为传统聚类方法在高维数据集中进行聚类时, 主要遇到两个问题:1. 高维数据集中存在大量无关的属性使得在所有维中存在簇的可能性几乎为零; 2. 高维空间中数据较低维空间中数据分布要稀疏, 其中数据间距离几乎相等是普遍现象, 而传统聚类方法是基于距离进行聚类的, 因此在高维空间中无法基于距离来构建簇. 高维聚类分析已成为聚类分析的一个重要研究方向. 同时高维数据聚类也是聚类技术的难点.

5 其他模型

5.1 EM 算法 (Expectation-Maximization algorithm)

5.1.1 Jensen 不等式和 KL 散度

(离散形式的 Jensen 不等式)f 为凸函数,$D = domf$, 若对于任意点集 $\{x_i\} \in D$, 对任意 M 个常数 λ_i, $\lambda_i \geq 0$ 且 $\sum_i^M \lambda_i = 1$, 满足:

$$f\left(\sum_{i=1}^{M} \lambda_i x_i\right) \leq \sum_{i=1}^{M} \lambda_i f\left(x_i\right) \tag{234}$$

证明如下: 当 M=1 或 2 时, 由凸函数的定义 (234) 成立. 假设当 $M = n$ 时,(234) 成立. 现证明则 $M = n + 1$ 时成立:

$$\begin{aligned} f\left(\sum_{i=1}^{n+1} \lambda_i x_i\right) &= f\left(\lambda_{n+1} x_{n+1} + \sum_{i=1}^{n} \lambda_i x_i\right) \\ &= f\left(\lambda_{n+1} x_{n+1} + (1-\lambda_{n+1}) \sum_{i=1}^{n} \eta_i x_i\right) \\ &\leq \lambda_{n+1} f\left(x_{n+1}\right) + (1-\lambda_{n+1}) f\left(\sum_{i=1}^{n} \eta_i x_i\right) \end{aligned} \tag{235}$$

其中 $\eta_i = \frac{\lambda_i}{1-\lambda_{n+1}}$. 注意到 λ_i 满足:$\sum_{i=1}^{n+1} \lambda_i = 1$, 因此:$\sum_{i=1}^{n} \lambda_i = 1 - \lambda_{n+1}$ 因此 η_i 也满足:$\sum_i^n \eta_i = \frac{\sum_i^n \lambda_i}{1-\lambda_{n+1}} = 1$. 由假设得到:$\sum_{i=1}^{n} f\left(\eta_i x_i\right) \leq \sum_{i=1}^{n} \eta_i f\left(x_i\right)$. 代入 (235) 得到:

$$f\left(\sum_{i=1}^{n+1} \lambda_i x_i\right) \leq \lambda_{n+1} f\left(x_{n+1}\right) + (1-\lambda_{n+1}) \sum_{i=1}^{n} \eta_i f\left(x_i\right) = \sum_{i=1}^{n+1} \lambda_i f\left(x_i\right) \tag{236}$$

因此 $M = n + 1$ 时,Jensen 不等式也成立. 由数学归纳法,Jensen 不等式成立.

在概率论中, 如果把 λ_i 看成取值为 x_i 的离散变量 x 的概率分布列, 那么 Jensen 不等式就可以写成 $f(E[x]) \leq E[f(x)]$. 接下来我们证明连续型随机变量也满足 Jensen 不等式.

(积分形式的 Jensen 不等式) 设一元函数 φ 在 $[a, b]$ 上连续, 设其上值域为 $[m, M]$.f 定义在 $[m, M]$ 上, 且是可微凸函数, p 在 $[a, b]$ 上可积, 且 $p(x) \geqslant 0$, 则:

$$f\left(\frac{\int_a^b p(x)\varphi(x)dx}{\int_a^b p(x)dx}\right) \leqslant \frac{\int_a^b p(x)f(\varphi(x))dx}{\int_a^b p(x)dx} \tag{237}$$

证明如下: 首先

$$m \leqslant \varphi(x) \leqslant M \Leftrightarrow \int_a^b p(x)mdx \leqslant \int_a^b p(x)\varphi(x)dx \leqslant \int_a^b p(x)Mdx \Leftrightarrow m \leqslant \frac{\int_a^b p(x)\varphi(x)dx}{\int_a^b p(x)dx} \leqslant M \tag{238}$$

设 $c = \frac{\int_a^b p(x)\varphi(x)dx}{\int_a^b p(x)dx} \in [m, M]$. 由 f 的凸性, 对 $\forall x \in [m, M]$, 有 $f(x) \geqslant f(c) + f^{'}(c)(x - c)$. 用 $x = \varphi(x)$

代入, 再在两边同乘以 $p(x)$, 得 $p(x)f\left(\varphi(x)\right) \geqslant \left[f(c)+f^{'}(c)(\varphi(x)-c)\right]p(x)$, 对 x 在 $[a,b]$ 上积分, 得到

$$\begin{aligned}\int_a^b p(x)f(\varphi(x))dx &\geqslant \int_a^b \left[f(c)+f'(c)(\varphi(x)-c)\right]p(x)dx \\ &= f(c)\int_a^b p(x)dx + f'(c)\int_a^b p(x)\varphi(x)dx - cf'(c)\int_a^b p(x)dx \\ &= f(c)\int_a^b p(x)dx \\ &= f\left(\frac{\int_a^b p(x)\varphi(x)dx}{\int_a^b p(x)dx}\right)\int_a^b p(x)dx\end{aligned} \tag{239}$$

从而有 (237), 得证.

进而可以推广到无穷, 令 $a=-\infty, b=+\infty$ 不影响证明过程. 由此可以推广到概率论中, 特别的, 若 x 为一维随机变量, 令 $\varphi(x)=x$,$p(x)$ 为 x 的密度函数. 从而有

$$f\left(\int_a^b p(x)\varphi(x)dx\right) \leqslant \int_a^b p(x)f(\varphi(x))dx \tag{240}$$

由此, 对凸函数 f, 不论随机变量是离散型还是连续型, 都有 Jensen 不等式:

$$f(E[x]) \leq E[f(x)] \tag{241}$$

下面介绍 KL 散度,KL 散度是两个概率分布 (probability distribution) 间差异的非对称性度量. 设 $P(x), Q(x)$ 是随机变量 X 上的两个概率分布, 则在离散型和连续型随机变量的情形下, 相对熵的定义分别为

$$\begin{aligned}KL(P\|Q) &= \sum P(x)\log\frac{P(x)}{Q(x)} \\ KL(P\|Q) &= \int P(x)\log\frac{P(x)}{Q(x)}dx\end{aligned} \tag{242}$$

利用 Jensen 不等式顺便可以证明 KL 散度的非负性:

$$KL(P\|Q) = \sum_{x\in X} P(x)\log\frac{P(x)}{Q(x)} = -E\left[\log\frac{Q(x)}{P(x)}\right] \geq -\log\left[\sum_{x\in X}P(x)\frac{Q(x)}{P(x)}\right] = -\log\left[\sum_{x\in X}Q(x)\right] = 0 \tag{243}$$

这里可看做令 $f(x)=\log\frac{Q(x)}{P(x)}$,$x$ 的密度函数为 $P(x)$, 即可证明. 只有 $P(x)=Q(x)$ 对于所有 x 都成立时, 等号才成立. 连续型的同理.

5.1.2 极大似然估计 (MLE)

最大似然估计 (MLE) 是一种参数估计的方法, 是在给定一个模型统计模型 P , 我们观测到数据 $X=\left\{x^{(i)}\right\}_{i=1}^{N}$ 时对模型 P 对应的参数 θ 进行一个估计. 大部分情况下, 定义 MLE 可以使得到参数使得观测到数据 X 发生的可能性最大, 也就是另 $P\left(\left\{x^{(i)}\right\}_{i=1}^{N} \mid \theta\right)$ 取到最大值. 另一种解释可以是: 当我们用 KL 散度来衡量差异时, 使用 MLE 得到的参数 $\hat{\theta}_{MLE}$ 可以使得 $P_{\hat{\theta}_{MLE}}$ 与真实模型 P_{θ^*} 最为接近, 此处的 θ^* 代表的是参数的真实值, 虽然未知但存在.

我们希望找到 $\hat{\theta}$ 使得 $P_{\hat{\theta}}$ 和真实模型 P_{θ^*} 最为接近, 可以转化为找到最小的 $D_{KL}\left(P_{\theta^*}, P_{\hat{\theta}}\right)$. 当我们预估参数的模型与真实分布间的 KL 散度为 0 时, 我们就得到真实参数 P_{θ^*} . 因此这个问题就被转化为了求

$D_{KL}\left(P_{\theta^*}\|P_{\hat{\theta}}\right)=\int P_{\theta^*}(x)\cdot\log\left(\frac{P_{\theta^*}(x)}{P_{\hat{\theta}}(x)}\right)d(x)$ 的最小值. KL 散度公式可以改写为:

$$\begin{aligned}D_{KL}\left(P_{\theta^*}\|P_{\hat{\theta}}\right)&=E_{\theta^*}\left[\log\frac{P_{\theta^*}(X)}{P_{\hat{\theta}}(X)}\right]\\&=E_{\theta^*}\left[\log\left(P_{\theta^*}(X)\right)\right]-E_{\theta^*}\left[\log\left(P_{\hat{\theta}}(X)\right)\right]\\&=C-E_{\theta^*}\left[\log\left(P_{\hat{\theta}}(X)\right)\right]\end{aligned}$$

因为 θ^* 就是真实参数所以 $E_{\theta^*}\left[\log\left(P_{\theta^*}(X)\right)\right]$ 的取值是固定的, 不随着 θ 的变化而变化, 因此就是常量.

因此, 最小化估计出来的分布与真实分布间的 KL 散度, 就可以转化成最大化 $E_{\theta^*}\left[\log\left(P_{\hat{\theta}}(X)\right)\right]$. 而根据大数定理 (简单来说当样本数趋近于无穷多时, 样本的均值收敛于期望 (弱大数定律)), 也就是说 n 很大时, $E[t(X)]$ 可以被 $\frac{1}{n}\sum_{i=1}^{n}t\left(X_i\right)$ 所替换. 我们在此处用大数定律对 $E_{\theta^*}\left[\log\left(P_{\hat{\theta}}(X)\right)\right]$ 进行处理, 即得到:

$$\begin{aligned}E_{\theta^*}\left[\log\left(P_{\hat{\theta}}(X)\right)\right]&=\frac{1}{n}\sum_{i=1}^{n}\log\left(P_{\hat{\theta}}\left(X_i\right)\right)\\&=\frac{1}{n}\log\prod_{i=1}^{n}P_{\hat{\theta}}\left(X_i\right)\end{aligned}$$

因此最大化 $\log\prod_{i=1}^{n}P_{\hat{\theta}}\left(X_i\right)$, 即可以使得找到 $\hat{\theta}$ 使得 $P_{\hat{\theta}}$ 和真实模型 P_{θ^*} 最为接近. 我们的似然函数的对数定义也是 $\log\prod_{i=1}^{n}P_{\hat{\theta}}\left(X_i\right)$, 因此通过 MLE 求的参数 $\hat{\theta}$, 可以使得其分布 $P_{\hat{\theta}}$ 与真实分布 P_{θ^*} 间的差异 (在 KL 散度衡量下) 最小.

MLE 的基本步骤是:

1. 首先, 写出似然函数:

样本集 $X=\left\{x^{(i)}\right\}_{i=1}^{N}$, 设概率密度是: $p\left(x^{(i)}\mid\theta\right)$ 抽到第 i 个样本的概率. 由于 N 个样本之间独立同分布, 所以同时抽到这 N 个样本的概率是它们各自概率的乘积,

$$L(\theta)=L\left(x^{(1)},x^{(2)},\ldots,x^{(N)};\theta\right)=\prod_{i=1}^{n}p\left(x^{(i)};\theta\right),\theta\in\Theta\tag{244}$$

这个联合概率反映了在该分布的参数是 θ 时, 得到 X 这组样本的概率. 我们需要找到一个参数 θ , 使得抽到 X 这组样本的概率最大, 也就是说需要其对应的似然函数 $L(\theta)$ 最大. 满足条件的 θ 叫做 θ 的最大似然估计值, 记为 $\hat{\theta}=argmaxL(\theta)$.

其次, 对似然函数取对数:

$$\mathcal{L}(\theta)=\ln L(\theta)=\ln\prod_{i=1}^{n}p\left(x^{(i)};\theta\right)=\sum_{i=1}^{n}lnp\left(x^{(i)};\theta\right)\tag{245}$$

然后, 对上式求导, 另导数为 0, 得到似然方程. 最后, 解似然方程, 得到的参数值即为所求. 多数情况下, 我们是根据已知条件来推算结果, 而极大似然估计是已知结果, 寻求使该结果出现的可能性最大的条件, 以此作为估计值.

举例, 根据样本估计高斯分布的概率. 设样本集 $X=\left\{x^{(i)}\right\}_{i=1}^{N}$ 服从高斯分布分布 $N\left(\mu,\sigma^2\right)$, 则似然函数为:

$$L\left(\mu,\sigma^2\right)=\prod_{i=1}^{N}\frac{1}{\sqrt{2\pi}\sigma}e^{\frac{\left(x^{(i)}-\mu\right)^2}{2\sigma^2}}=\left(2\pi\sigma^2\right)^{-\frac{n}{2}}e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(x^{(i)}-\mu\right)^2}\tag{246}$$

对数似然函数:

$$\mathcal{L}\left(\mu,\sigma^2\right)=\ln L\left(\mu,\sigma^2\right)=-\frac{n}{2}\ln(2\pi)-\frac{n}{2}\ln\left(\sigma^2\right)-\frac{1}{2\sigma^2}\sum_{i=1}^{n}\left(x^{(i)}-\mu\right)^2\tag{247}$$

求导, 得方程组:

$$\begin{cases} \frac{\partial \mathcal{L}\left(\mu,\sigma^2\right)}{\partial \mu} = \frac{1}{\sigma^2}\sum_{i=1}^{n}\left(x^{(i)}-\mu\right) \quad = 0 \\ \frac{\partial \mathcal{L}\left(\mu,\sigma^2\right)}{\partial \sigma^2} = -\frac{n}{2\sigma^2}+\frac{1}{2\sigma^4}\sum_{i=1}^{n}\left(x^{(i)}-\mu\right)^2 = 0 \end{cases} \tag{248}$$

联合解得;

$$\begin{cases} \mu^* = \bar{x} = \frac{1}{n}\sum_{i=1}^{n} x^{(i)} \\ \sigma^{*2} = \frac{1}{n}\sum_{i=1}^{n}\left(x^{(i)}-\bar{x}\right)^2 \end{cases} \tag{249}$$

5.1.3 EM 算法推导

EM(Expectation-Maximum) 算法也称期望最大化算法, 其基本思想是: 首先根据已经给出的观测数据, 估计出模型参数的值; 然后再依据上一步估计出的参数值估计缺失数据的值, 再根据估计出的缺失数据加上之前已经观测到的数据重新再对参数值进行估计, 然后反复迭代, 直至最后收敛, 迭代结束.

先直接给出 EM 算法的步骤,

---

algorithm 3 EM 算法

---

input: 观测变量数据 Y, 隐变量数据 Z, 联合分布 $P(Y,Z \mid \theta)$, 条件分布 $P(Z \mid Y,\theta)$

output: 模型参数 θ

选择参数的初值 $\theta^{(i)}$, 开始迭代;

repeat

E 步: 记 $\theta^{(i)}$ 为第 i 次迭代参数 θ 的估计值, 在第 $i+1$ 次迭代的 E 步, 计算

$$\begin{aligned} Q\left(\theta,\theta^{(i)}\right) &= E_z\left[\log P(Y,Z \mid \theta) \mid Y,\theta^{(i)}\right] \\ &= \sum_{Z} \log P(Y,Z \mid \theta) P\left(Z \mid Y,\theta^{(i)}\right) \end{aligned} \tag{250}$$

这里, $P\left(Z \mid Y,\theta^{(i)}\right)$ 是在给定观测数据 Y 和当前的参数估计 $\theta^{(i)}$ 下隐变量数据 Z 的条件概率分布;

M 步: 求使 $Q\left(\theta,\theta^{(i)}\right)$ 极大化的 θ, 确定第 $i+1$ 次迭代的参数的估计值 $\theta^{(i+1)}$

$$\theta^{(i+1)} = \arg\max_{\theta} Q\left(\theta,\theta^{(i)}\right) \tag{251}$$

until

$\|\theta^{(i)}-\theta^{(i-1)}\| \leqslant \varepsilon$ 或 $i >$ 最大迭代步数

对于含有隐变量 Z 的概率模型 (隐变量通常为数据与分布的对应关系, 即哪个数据以多大概率是采样与哪个分布), 我们的目标是极大化观测数据 Y 关于模型分布参数 θ 的对数似然函数:

$$\begin{aligned} \mathcal{L}(\theta) &= \log P(Y \mid \theta) = \log \sum_{Z} P(Y,Z \mid \theta) \\ &= \log\left(\sum_{Z} P(Y \mid Z,\theta) P(Z \mid \theta)\right) \end{aligned} \tag{252}$$

由于隐变量 Z 的存在, 直接 MLE 不可实现.EM 算法通过迭代逐步近似极大化 $\mathcal{L}(\theta)$, 第 i 次迭代得到的估计值是 $\theta^{(i)}$, 我们希望新估计值 θ 能使 $\mathcal{L}(\theta)$ 增加, 即 $L(\theta) > L\left(\theta^{(i)}\right)$, 并逐步达到极大值. 考虑两步迭代似然函

数的差,

$$\mathcal{L}(\theta)-\mathcal{L}\left(\theta^{(i)}\right)=\log\left(\sum_{Z}P(Y\mid Z,\theta)P(Z\mid\theta)\right)-\log P\left(Y\mid\theta^{(i)}\right)$$

利用 Jensen 不等式得到其下界:

$$\begin{aligned}\mathcal{L}(\theta)-\mathcal{L}\left(\theta^{(0)}\right)&=\log\left(\sum_{Z}P\left(Z\mid Y_t\theta^{(i)}\right)\frac{P(Y\mid Z,\theta)P(Z\mid\theta)}{P\left(ZY,\theta^{(0)}\right)}\right)-\log P\left(Y\mid\theta^{(i)}\right)\\&\geqslant\sum_{Z}P\left(Z\mid Y,\theta^{(i)}\right)\log\frac{P(Y\mid Z,\theta)P(Z\mid\theta)}{P\left(Z\mid Y,\theta^{(0)}\right)}-\log P\left(Y\mid\theta^{(i)}\right)\\&=\sum_{Z}P\left(Z\mid Y,\theta^{(i)}\right)\log\frac{P(Y\mid Z,\theta)P(Z\mid\theta)}{P\left(Z\mid Y,\theta^{(1)}\right)P\left(Y\mid\theta^{(\vartheta)}\right)}\end{aligned}\tag{253}$$

定义 $B\left(\theta,\theta^{(i)}\right)$, 令

$$B\left(\theta,\theta^{(i)}\right)=\mathcal{L}\left(\theta^{(i)}\right)+\sum_{Z}P\left(Z\mid Y,\theta^{(i)}\right)\log\frac{P(Y\mid Z,\theta)P(Z\mid\theta)}{P\left(Z\mid Y,\theta^{(i)}\right)P\left(Y\mid\theta^{(0)}\right)}\tag{254}$$

则

$$\mathcal{L}(\theta)\geqslant B\left(\theta,\theta^{(i)}\right)\tag{255}$$

即 $B\left(\theta,\theta^{(i)}\right)$ 是 $\mathcal{L}(\theta)$ 的一个下界, 且满足

$$\mathcal{L}\left(\theta^{(i)}\right)=B\left(\theta^{(i)},\theta^{(i)}\right)\tag{256}$$

因此, 任何可以使 $B\left(\theta,\theta^{(1)}\right)$ 增大的 θ, 也可以使 $\mathcal{L}(\theta)$ 增大. 为了使 $\mathcal{L}(\theta)$ 有尽可能大的增长, 选择 $\theta^{(i+1)}$ 使 $B\left(\theta,\theta^{(i)}\right)$ 达到极大, 即

$$\theta^{(i+1)}=\arg\max_{\theta}B\left(\theta,\theta^{(i)}\right)\tag{257}$$

进而有

$$\begin{aligned}\theta^{(i+1)}&=\arg\max_{\theta}\left(\mathcal{L}\left(\theta^{(i)}\right)+\sum_{z}P\left(Z\mid Y,\theta^{(i)}\right)\log\frac{P(Y\mid Z,\theta)P(Z\mid\theta)}{P\left(Z\mid Y,\theta^{(i)}\right)P\left(Y\mid\theta^{(0)}\right)}\right)\\&=\arg\max_{\theta}\left(\sum_{Z}P\left(Z\mid Y,\theta^{(i)}\right)\log(P(Y\mid Z,\theta)P(Z\mid\theta))\right)\\&=\arg\max_{\theta}\left(\sum_{Z}P\left(Z\mid Y,\theta^{(i)}\right)\log P(Y,Z\mid\theta)\right)\\&=\arg\max_{\theta}Q\left(\theta,\theta^{(i)}\right)\end{aligned}\tag{258}$$

这也就是说 EM 算法通过不断求解下界的极大化来逼近对数似然函数的极大化.

5.1.4 EM 算法的收敛性

首先我们证明 $\mathcal{L}(\theta^{(i)})$ 随迭代次数单调递增, 即 $\mathcal{L}(\theta^{(i)})<\mathcal{L}(\theta^{(i+1)})$.

由条件概率,

$$\mathcal{L}(\theta)=\ln P(X\mid\theta)=\ln\frac{P(X,Z\mid\theta)}{P(Z\mid X,\theta)}=\ln P(X,Z\mid\theta)-\ln P(Z\mid X,\theta)\tag{259}$$

由 (250)

$$H\left(\theta, \theta^{(i)}\right)=\sum_{Z} \log P(Z \mid Y, \theta) P\left(Z \mid Y, \theta^{(i)}\right) \tag{260}$$

由 (250), 对数似然函数可写成

$$\mathcal{L}(\theta)=\log P(Y \mid \theta)=Q\left(\theta, \theta^{(i)}\right)-H\left(\theta, \theta^{(i)}\right) \tag{261}$$

由此得到

$$\begin{aligned} \mathcal{L}\left(\theta^{(i+1)}\right)-\mathcal{L}\left(\theta^{(i)}\right) &= \log P\left(Y \mid \theta^{(i+1)}\right)-\log P\left(Y \mid \theta^{(i)}\right) \\ &= \left[Q\left(\theta^{(i+1)}, \theta^{(i)}\right)-Q\left(\theta^{(i)}, \theta^{(i)}\right)\right]-\left[H\left(\theta^{(i+1)}, \theta^{(i)}\right)-H\left(\theta^{(0)}, \theta^{(i)}\right)\right] \end{aligned} \tag{262}$$

上式右端第一项, 由于 $\theta^{(i+1)}$ 使 $Q\left(\theta, \theta^{(0)}\right)$ 达到极大, 所以有

$$Q\left(\theta^{(i+1)}, \theta^{(i)}\right)-Q\left(\theta^{(i)}, \theta^{(i)}\right) \geqslant 0 \tag{263}$$

右端第二项有 H 的定义有

$$\begin{aligned} H\left(\theta^{(i+1)}, \theta^{(0)}\right)-H\left(\theta^{(i)}, \theta^{(0)}\right) &= \sum_{z}\left(\log \frac{P\left(Z \mid Y, \theta^{(i+1)}\right)}{P\left(Z \mid Y, \theta^{(0)}\right)}\right) P\left(Z \mid Y, \theta^{(i)}\right) \\ &\leqslant \log\left(\sum_{Z} \frac{P\left(Z \mid Y, \theta^{(i+1)}\right)}{P\left(Z \mid Y, \theta^{(i)}\right)} P\left(Z \mid Y, \theta^{(i)}\right)\right) \\ &= \log\left(\sum_{z} P\left(Z \mid Y, \theta^{(i+1)}\right)\right)=0 \end{aligned} \tag{264}$$

综上 $\mathcal{L}\left(\theta^{(i+1)}\right)-\mathcal{L}\left(\theta^{(i)}\right) \geqslant 0$

EM 的应用有很多, 比如 GMM、聚类、HMM 、GEM 等等

5.2 神经网络 (Neural Networks)

在人工神经网络领域的数学观点中, 通用近似定理 (universal approximation theorem) 指的是: 如果一个前馈神经网络具有线性输出层和至少一层隐藏层, 只要给予网络足够数量的神经元, 便可以实现以足够高精度来逼近任意一个在 $\mathbb{R}^n$ 的紧子集上的连续函数. 由此为神经网络提供了理论保证. 而在实际中我们往往采用层数较多的网络结构而不是每层神经元较多的网络结构. 因为从 approximation error (拟合误差) 的角度, 无论是实验结果来看, 还是从理论分析来看, 大部分情况下" 深" 都比" 宽" 更有效.《Why Deep Neural Networks for Function Approximation》证明了, 如果想要达到 ε 的拟合误差, 深度为常数 (与 ε 无关) 的神经网络需要 $O(poly(1/\varepsilon))$ 个神经元, 也就是说, 浅层神经网络的神经元数量随着精度 $(1/\varepsilon)$ 的上升多项式增长. 然而, 深度为 $O(\log(1/\varepsilon))$ 的神经网络只需要 $O(poly\log(1/\varepsilon))$ 个神经元, 也就是说, 深度神经网络的神经元数量随着精度的上升对数增长. 换言之, 想要达到同样的拟合误差, 更深的神经网络需要的神经元数量远小于层数少的神经网络.

上个世纪由于计算能力的限制, 神经网络的深度和形态都受到限制,2010 年前后随着计算能力的提升和大数据的涌现, 神经网络或者说深度学习在很多任务上都都以较大优势胜过传统方法, 各种深度学习的架构和方法也大放异彩.

5.2.1 BP 神经网络

BP 神经网络是线性变换和非线性函数的不断嵌套.

神经网络中最基本的单元是神经元模型 (neuron). 在生物神经网络的原始机制中, 每个神经元通常都有多个树突 (dendrite), 一个轴突 (axon) 和一个细胞体 (cell body), 树突短而多分支, 轴突长而只有一个；在功能上, 树突用于传入其它神经元传递的神经冲动, 而轴突用于将神经冲动传出到其它神经元, 当树突或细胞体传入的神经冲动使得神经元兴奋时, 该神经元就会通过轴突向其它神经元传递兴奋.

一直沿用至今的"M-P 神经元模型" 正是对这一结构进行了抽象, 也称" 阈值逻辑单元", 其中树突对应于输入部分, 每个神经元收到 n 个其他神经元传递过来的输入信号, 这些信号通过带权重的连接传递给细胞体

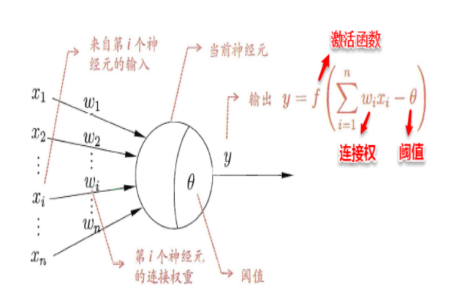


图 7: 神经网络

与线性分类十分相似, 神经元模型最理想的激活函数也是阶跃函数, 即将神经元输入值与阈值的差值映射为输出值 1 或 0, 若差值大于零输出 1, 对应兴奋；若差值小于零则输出 0, 对应抑制. 但阶跃函数不连续, 不光滑, 故在 M-P 神经元模型中, 也采用 Sigmoid 函数来近似, Sigmoid 函数将较大范围内变化的输入值挤压到 (0,1) 输出值范围内. 常见的激活函数还有 ReLU、tanh、Swish 等, 及其变体, 各具特性.

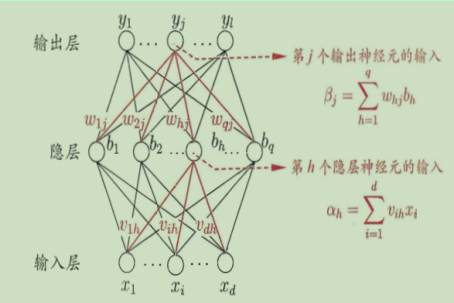


图 8: 多层神经网络

神经网络的学习主要蕴含在权重和阈值中. 给定训练集 $D = \{(x_1, y_1), , (x_2, y_2), \ldots, (x_m, y_m)\}, x_i \in \mathbb{R}^d, y_i \in \mathbb{R}^l$, 即输入示例由 d 个属性描述, 输出 l 维实值向量. 为便于讨论, 如图给出了一个拥有 d 个输入神经元、l 个输出神经元、q 个隐层神经元的多层前馈网络结构, 其中输出层第 j 个神经元的阈值用 θ_j 表示, 隐层第 h 个神经元的间值用 γ_h 表示. 输入层第 i 个神经元与隐层第 h 个神经元之间的连接权为 v_{ih}, 隐层第 h 个神经元与输出层第 j 个神经元之间的连接权为 w_{hj}. 则隐层第 h 个神经元接收到的输入为 $\alpha_h = \sum_{i=1}^{d} v_{ih} x_i$, 输出层第 j 个神经元接收到的输入为 $\beta_j = \sum_{h=1}^{q} w_{hj} b_h$, 其中 b_h 为隐层第 h 个神经元的输出.. 假设隐层和输出层神经元都使用 Sigmoid 作为激活函数.

对训练例 (x_k, y_k), 假定神经网终的输出为 $\hat{y}_k = \left(\hat{y}_1^k, \hat{y}_2^k, \ldots, \hat{y}_l^k\right)$, 其中

$$\hat{y}_j^k = f\left(\beta_j - \theta_j\right) \tag{265}$$

则网络在 (x_k, y_k) 上的均方误差为

$$E_k = \frac{1}{2} \sum_{j=1}^{l} \left(\hat{y}_j^k - y_j^k\right)^2 \tag{266}$$

图中的网络有 $(d + l + 1)q + l$ 个参数需确定: 输入层到隐层的 $d \times q$ 个权值、隐层到输出层的 $q \times l$ 个权值、q 个隐层神经元的间值、l 个输出层神经元的间值. BP 是一个迭代学习算法, 在迭代的每一轮中采用广义的感知机学习规则对参数进行更新估计, 即与式 (5.1) 类似, 任意参数 v 的更新估计式为 $v \leftarrow v + \Delta v$ 下面以隐层到输出层的连接权 w_{hj} 为例来进行推导. BP 算法基于梯度下降策略, 以目标的负梯度方向对参数进行调整. 对式 (266) 的误差 E_k, 给定学习率 η, 有

$$\Delta w_{hj} = -\eta \frac{\partial E_k}{\partial w_{hj}} \tag{267}$$

注意到 w_{hj} 先影响到第 j 个输出层神经元的输入值 β_j, 再影响到其输出值 $\hat{y}_j^k$, 然后影响到 E_k, 有

$$\frac{\partial E_k}{\partial w_{hj}} = \frac{\partial E_k}{\partial \hat{y}_j^k} \cdot \frac{\partial \hat{y}_j^k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial w_{hj}} \tag{268}$$

根据 β_j 的定义, 显然有

$$\frac{\partial \beta_j}{\partial w_{hj}} = b_h \tag{269}$$

.Sigmoid 函数的导数有很好的性质:$\sigma'(x) = \sigma(x)(1 - \sigma(x))$ 从而有

$$\begin{aligned} g_j &= -\frac{\partial E_k}{\partial y_j^k} \cdot \frac{\partial y_j^k}{\partial \beta_j} \\ &= -\left(\hat{y}_j^k - y_j^k\right) f'\left(\beta_j - \theta_j\right) \\ &= \hat{y}_j^k \left(1 - \hat{y}_j^k\right)\left(y_j^k - \hat{y}_j^k\right) \end{aligned} \tag{270}$$

将上两式代入 (268), 就得到了 BP 算法中关于 w_{hj} 的更新公式

$$\Delta w_{hj} = \eta g_j b_h \tag{271}$$

类似可得

$$\begin{aligned} \Delta\theta_j &= -\eta g_j \\ \Delta v_{ih} &= \eta e_h x_i \\ \Delta\gamma_h &= -\eta e_h \end{aligned} \tag{272}$$

其中

$$\begin{aligned} e_h &= -\frac{\partial E_k}{\partial b_h} \cdot \frac{\partial b_h}{\partial \alpha_h} \\ &= -\sum_{j=1}^{l} \frac{\partial E_k}{\partial \beta_j} \cdot \frac{\partial \beta_j}{\partial b_h} f'(\alpha_h - \gamma_h) \\ &= \sum_{j=1}^{l} w_{hj} g_j f'(\alpha_h - \gamma_h) \\ &= b_h (1 - b_h) \sum_{j=1}^{l} w_{hj} g_j \end{aligned} \tag{273}$$

学习率 $\eta \in (0,1)$ 控制着算法每一轮迭代中的更新步长, 若太大则容易振荡, 太小则收玫速度又会过慢. 有时为了做精细调节可令同层的参数采取相同的学习率.

在训练神经网络的过程中还会有很多需要考虑的参数问题.

首先是 input size 和 output size 这一般根据训练问题来确定. 如果是构造映射 $f : \mathbb{R}^d \to \mathbb{R}$ 则可以设置 input size=d 和 output size=1. 如果是分类问题则 output size 可以是类别个数, 输出代表样本为各类的概率等.

其次是网络结构, 常见的网络种类大致分为前馈神经网络、反馈神经网络、图网络. 还需要设置网络的深度、宽度等必要的结构参数.

再次, 还需要选择损失函数的种类, 激活函数的种类及其参数, 优化器的种类.

另外, 还需要设定几个重要的超参 epoch,batch size,learning rate.

最后, 还要考虑训练过程中的 tricks, 比如数据预处理, 批量归一化 BN, 提前停止, 丢弃法 dropout, 学习率衰减, 数据增强等等

5.2.2 CNN

卷积神经网络最早是主要用来处理图像信息. 如果用全连接前馈网络来处理图像时, 会存在两个问题: (1) 参数太多: 如果输入图像大小为 100×100×3. 在全连接前馈网络中, 第一个隐藏层的每个神经元到输入层都有 100×100×3 = 30,000 个相互独立的连接, 每个连接都对应一个权重参数. 随着隐藏层神经元数量的增多, 参数的规模也会急剧增加. 这会导致整个神经网络的训练效率会非常低, 也很容易出现过拟合. (2) 局部不变性特征: 自然图像中的物体都具有局部不变性特征, 比如在尺度缩放、平移、旋转等操作不影响其语义信息. 而全连接前馈网络很难提取这些局部不变特征, 一般需要进行数据增强来提高性能.

卷积神经网络是受生物学上感受野的机制而提出. 一个神经元的感受野是指视网膜上的特定区域, 只有这个区域内的刺激才能够激活该神经元. 目前的卷积神经网络一般是由卷积层、汇聚层和全连接层交叉堆叠而成的前馈神经网络, 使用反向传播算法进行训练.

CNN 重要的三个特性: 局部连接, 权重共享以及汇聚. 这些特性使得卷积神经网络具有一定程度上的平移、缩放和旋转不变性. 且和前馈神经网络相比, 卷积神经网络的参数更少.

较常用的是二维卷积, 经常用在图像处理中. 给定一个图像 $X \in \mathbb{R}^{M\times N}$, 和滤波器 $W \in \mathbb{R}^{m\times n}$, 一般 $m << M, n << N$, 其卷积为

$$y_{ij} = \sum_{u=1}^{m}\sum_{v=1}^{n} w_{uv} \cdot x_{i-u+1,j-v+1} \tag{274}$$

这个数学上的离散卷积的定义是一致的. 但这在实际操作中时不符合直觉的. 在具体实现上, 一般会以互相关操作来代替卷积, 从而会减少一些不必要的操作或开销. 互相关即一个衡量两个序列相关性的函数, 通常是用滑动窗口的点积计算来实现. 给定一个图像 $X \in \mathbb{R}^{M\times N}$ 和卷积核 $W \in \mathbb{R}^{m\times n}$, 它们的互相关为

$$y_{ij} = \sum_{u=1}^{m}\sum_{v=1}^{n} w_{uv} \cdot x_{i+u-1,j+v-1} \tag{275}$$

由卷积与互相关对比可知, 互相关和卷积的区别在于卷积核仅仅是否进行翻转. 因此互相关也可以称为不翻转卷积. 在神经网络中使用卷积是为了进行特征抽取, 卷积核是否进行翻转和其特征抽取的能力无关. 特别是当卷积核是可学习的参数时, 卷积和互相关是等价. 下文的 CNN 中直接称互相关为卷积.

通俗来说每个滤波器的作用就是将整个矩阵” 刷” 一遍, 得到一个新的矩阵 (更小的矩阵).

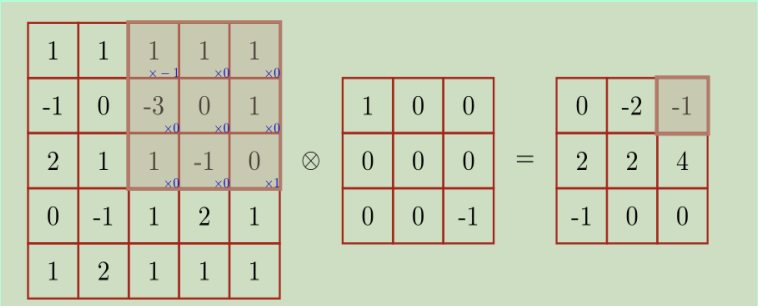
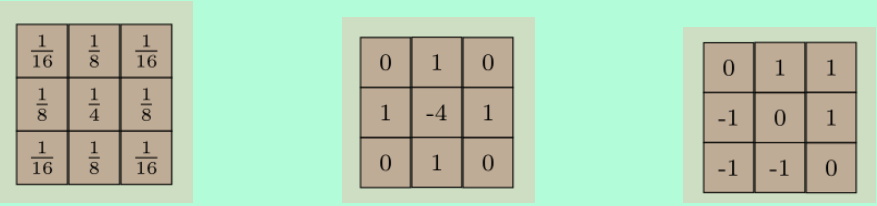


图 9: 滤波器效果

常用的均值滤波就是当前位置的像素值设为滤波器窗口中所有像素的平均值, 也就是 $f_{uv} = \frac{1}{mn}$. 在图像处理中, 卷积经常作为特征提取的有效方法. 一幅图像在经过卷积操作后得到结果称为特征映射. 下图给出在图像处理中几种常用的滤波器, 以及其对应的特征映射. 图中左边的滤波器是常用的高斯滤波器, 可以用来对图像进行平滑去噪; 中间和右边的过滤器可以用来提取边缘特征.



(a) filter1 (b) filter2 (c) filter3

图 10: 具有不同意义的 filter 举例

一般常用的卷积有以下三类: 窄卷积: 步长 $s=1$, 两端不补零 $p=0$, 卷积后输出长度为 $n-m+1$.

宽卷积: 步长 $s=1$, 两端补零 $p=m-1$, 卷积后输出长度 $n+m-1$.

等宽卷积: 步长 $s=1$, 两端补零 $p=(m-1)/2$, 卷积后输出长度 n . 卷积神经网络一般由卷积层、汇聚层和全连接层构成.

卷积层的作用是提取一个局部区域的特征, 不同的卷积核相当于不同的特征提取器. 特征映射 (Feature Map) 为一幅图像 (或其他二维张量) 在经过卷积提取到的特征, 每个特征映射可以作为一类抽取的图像特征. 为了提高卷积网络的表示能力, 可以在每一层使用多个不同的特征映射, 以更好地表示图像的特征.

在输入层, 特征映射就是图像本身. 如果是灰度图像, 就是有一个特征映射, 深度 $D=1$; 如果是彩色图像, 分别有 RGB 三个颜色通道的特征映射, 输入层深度 $D=3$. 不失一般性, 可以假设一个卷积层的结构如下:

输入特征映射组: $X \in \mathbb{R}^{M\times N\times D}$ 为三维张量, 其中每个切片 (slice) 矩阵 $X^d \in \mathbb{R}^{M\times N}$ 为一个输入特征映射, $1 \le d \le D$;

输出特征映射组: $Y \in \mathbb{R}^{M'\times N'\times P}$ 为三维张量, 其中每个切片矩阵 $Y^p \in \mathbb{R}^{M'\times N'}$ 为一个输出特征映射, $1 \le p \le P$;

卷积核: $W \in \mathbb{R}^{m\times n\times D\times P}$ 为四维张量, 其中每个切片矩阵 $W^{p,d} \in \mathbb{R}^{m\times n}$ 为一个两维卷积核, $1 \le d \le D, 1 \le p \le P$.

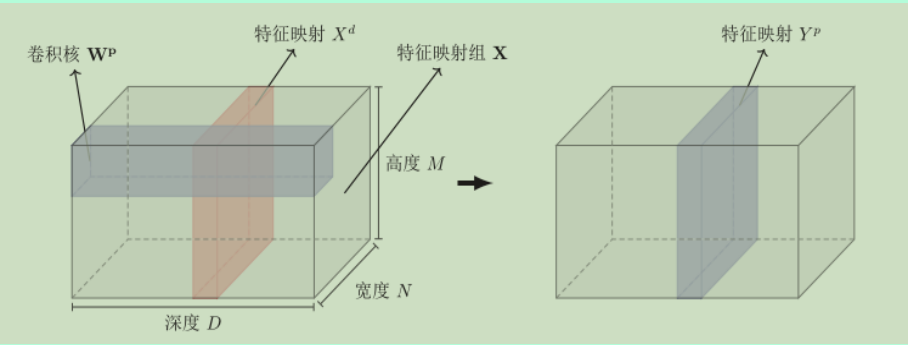


图 11: 滤波器效果

为了计算输出特征映射 Y^p, 用卷积核 $W^{p,1}, W^{p,2}, \cdots, W^{p,D}$ 分别对输入特征映射 $X^1, X^2, \cdots, X^D$ 进行卷积, 然后将卷积结果相加, 并加上一个标量偏置 b 得到卷积层的净输入 Z^p, 再经过非线性激活函数后得到输

出特征映射 Y^p .

$$\begin{aligned} Z^p &= W^p \otimes X + b^p = \sum_{d=1}^{D} W^{p,d} \otimes X^d + b^p \\ Y^p &= f\left(Z^p\right) \end{aligned} \tag{276}$$

其中 $W^p \in \mathbb{R}^{m \times n \times D}$ 为三维卷积核, $f(\cdot)$ 为非线性激活函数, 一般用 $ReLU$ 函数. 如果希望卷积层输出 P 个特征映射, 将上述过程进行 P 次, 得到 P 个输出特征映射 $Y^1, Y^2, \cdots, Y^P$. 在输入为 $X \in \mathbb{R}^{M \times N \times D}$, 输出为 $Y \in \mathbb{R}^{M' \times N' \times P}$ 的卷积层中, 每一个输入特征映射都需要 D 个滤波器以及共同的一个偏置. 假设每个滤波器的大小为 $m \times n$, 那么共需要 $P \times D \times (m \times n) + P$ 个参数.

汇聚层 (Pooling Layer), 其作用是进行特征选择, 降低特征数量, 并从而减少参数数量. 卷积层虽然可以显著减少网络中连接的数量, 但特征映射组中的神经元个数并没有显著减少. 如果后面接一个分类器, 分类器的输入维数依然很高, 很容易出现过拟合. 为了解决这个问题, 可以在卷积层之后加上一个汇聚层, 从而降低特征维数, 避免过拟合. 假设汇聚层的输入特征映射组为 $X \in \mathbb{R}^{M \times N \times D}$, 对于其中每一个特征映射 X^d, 将其划分为很多区域 $R_{m,n}^d, 1 \leq m \leq M', 1 \leq n \leq N'$, 这些区域可以重叠, 也可以不重叠. 汇聚 (Pooling) 是指对每个区域进行下采样 (Down Sampling) 得到一个值, 作为这个区域的概括. 常用的汇聚函数有两种:

1. 最大汇聚: 一般是取一个区域内所有神经元的最大值.

$$Y_{m,n}^d = \max_{i \in R_{m,n}^d} x_i \tag{277}$$

其中 x_i 为区域 R_k^d 内每个神经元的激活值.

2. 平均汇聚: 一般是取区域内所有神经元的平均值.

$$Y_{m,n}^d = \frac{1}{\left|R_{m,n}^d\right|} \sum_{i \in R_{m,n}^d} x_i \tag{278}$$

对每一个输入特征映射 X^d 的 $M' \times N'$ 个区域进行子采样, 得到汇聚层的输出特征映射 $Y^d = \left\{Y_{m,n}^d\right\}, 1 \leq m \leq M', 1 \leq n \leq N'$. 图 5.8 给出了采样最大汇聚进行子采样操作的示例. 可以看出, 汇聚层不但可以有效地减少神经元的数量, 还可以使得网络对一些小的局部形态改变保持不变性, 并拥有更大的感受野.

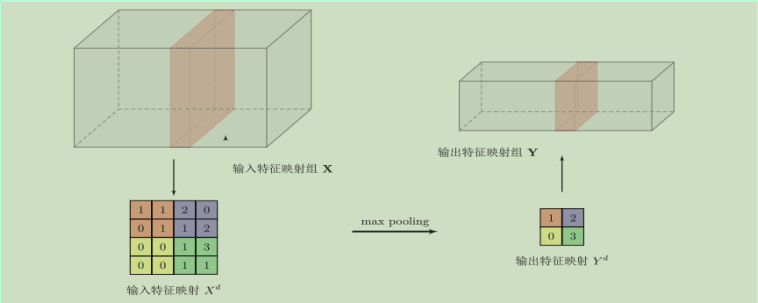


图 12: 滤波器效果

目前主流的卷积网络中, 汇聚层仅包含下采样操作, 在早期的一些卷积网络中, 有时也会使用非线性激活函数, 比如

$$Y'^d = f\left(w^d \cdot Y^d + b^d\right) \tag{279}$$

其中 Y'^d 为汇聚层的输出, $f(\cdot)$ 为非线性激活函数, w^d 和 b^d 为可学习的标量权重和偏置.

典型的汇聚层是将每个特征映射划分为 2×2 (或 3×3) 大小的不重叠区域, 然后使用某种汇聚的方式进行下采样. 汇聚层也可以看做是一个特殊的卷积层, 卷积核大小为 $m \times m$, 步长为 $s \times s$, 卷积核为 max 函数或 mean 函数. 过大的采样区域会急剧减少神经元的数量, 会造成过多的信息损失. 在实际训练过程中都是可调的参数.

5.2.4 典型的卷积网络结构通常卷积网络是由卷积层、汇聚层交叉堆叠而成, 最后在接全连接层来输出结果. 如下图所示所示. 一个卷积块为连续 M 个卷积层和 b 个汇聚层 (M 通常设置为 $2 \sim 5, b$ 为 0 或 1). 一个卷积网络中可以堆叠 N 个连续的卷积块, 然后在接着 K 个全连接层 (N 的取值区间比较大, 比如 $1 \sim 100$ 或者更大; K 一般为 $0 \sim 2$).

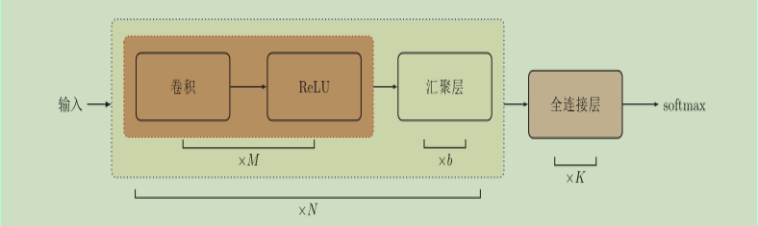


图 13: 滤波器效果

目前, 整个网络结构趋向于使用更小的卷积核 (比如 1×1 和 3×3) 以及更深的结构 (比如层数大于 50). 此外, 由于卷积的操作性越来越灵活 (比如不同的步长), 汇聚层的作用变得也越来越小, 因此目前比较流行的卷积网络中, 汇聚层的比例也逐渐降低, 趋向于全卷积网络.

在卷积网络中, 参数为卷积核中权重以及偏置. 和全连接前馈网络类似, 卷积网络也可以通过误差反向传播算法来进行参数学习. 在全连接前馈神经网络中, 梯度主要通过每一层的误差项 δ 进行反向传播, 并进一步计算每层参数的梯度. 在卷积神经网络中, 主要有两种不同功能的神经层: 卷积层和汇聚层. 而参数为卷积核以及偏置, 因此只需要计算卷积层中参数的梯度.

我们先考虑二维卷积运算的导数, 假设 $Y = W \otimes X$, 其中 $X \in \mathbb{R}^{M \times N}, W \in \mathbb{R}^{m \times n}, Y \in \mathbb{R}^{(M-m+1) \times (N-n+1)}$, 函数 $f(Y) \in \mathbb{R}$ 为一个标量函数, 则

$$\begin{aligned} \frac{\partial f(Y)}{\partial w_{uv}} &= \sum_{i=1}^{M-m+1} \sum_{j=1}^{N-n+1} \frac{\partial y_{ij}}{\partial w_{uv}} \frac{\partial f(Y)}{\partial y_{ij}} \\ &= \sum_{i=1}^{M-m+1} \sum_{j=1}^{N-n+1} x_{i+u-1, j+v-1} \frac{\partial f(Y)}{\partial y_{ij}} \\ &= \sum_{i=1}^{M-m+1} \sum_{j=1}^{N-n+1} \frac{\partial f(Y)}{\partial y_{ij}} x_{u+i-1, v+j-1} \end{aligned} \tag{280}$$

由此可以看出, $f(Y)$ 关于 W 的偏导数为 X 和 $\frac{\partial f(Y)}{\partial Y}$ 的卷积

$$\frac{\partial f(Y)}{\partial W} = \frac{\partial f(Y)}{\partial Y} \otimes X \tag{281}$$

同理得到,

$$\begin{aligned}\frac{\partial f(Y)}{\partial x_{st}} &= \sum_{i=1}^{M-m+1}\sum_{j=1}^{N-n+1}\frac{\partial y_{ij}}{\partial x_{st}}\frac{\partial f(Y)}{\partial y_{ij}} \\ &= \sum_{i=1}^{M-m+1}\sum_{j=1}^{N-n+1} w_{s-i+1,t-j+1}\frac{\partial f(Y)}{\partial y_{ij}}\end{aligned} \tag{282}$$

其中当 $(s-i+1)<1$, 或 $(s-i+1)>m$, 或 $(t-j+1)<1$, 或 $(t-j+1)>n$ 时, $w_{s-i+1,t-j+1}=0$. 即相当于对 W 进行了 $p=(M-m,N-n)$ 的零填充.

从上式可以看出, $f(Y)$ 关于 X 的偏导数为 W 和 $\frac{\partial f(Y)}{\partial Y}$ 的宽卷积, 其中的卷积是真正的卷积而不是互相关, 为了一致性, 我们用互相关的" 卷积", 即

$$\begin{aligned}\frac{\partial f(Y)}{\partial X} &= rot180\left(\frac{\partial f(Y)}{\partial Y}\right)\tilde{\otimes}W \\ &= rot180(W)\tilde{\otimes}\frac{\partial f(Y)}{\partial Y}\end{aligned} \tag{283}$$

其中 $rot180(\cdot)$ 表示旋转 180 度. 不失一般性, 对第 l 层为卷积层, 第 $l-1$ 层的输入特征映射为 $X^{(l-1)}\in\mathbb{R}^{M\times N\times D}$, 通过卷积计算得到第 l 层的特征映射净输入 $Z^{(l)}\in\mathbb{R}^{M'\times N'\times P}$. 第 l 层的第 $p(1\le p\le P)$ 个特征映射净输入

$$Z^{(l,p)}=\sum_{d=1}^{D}W^{(l,p,d)}\otimes X^{(l-1,d)}+b^{(l,p)} \tag{284}$$

其中 $W^{(l,p,d)}$ 和 $b^{(l,p)}$ 为卷积核以及偏置. 第 l 层中共有 $P\times D$ 个卷积核和 P 个偏置, 可以分别使用链式法则来计算其梯度. 根据上面卷积导数的推导, 损失函数关于第 l 层的卷积核 $W^{(l,p,d)}$ 的偏导数为

$$\begin{aligned}\frac{\partial\mathcal{L}(Y,\hat{Y})}{\partial W^{(l,p,d)}} &= \frac{\partial\mathcal{L}(Y,\hat{Y})}{\partial Z^{(l,p)}}\otimes X^{(l-1,d)} \\ &= \delta^{(l,p)}\otimes X^{(l-1,d)}\end{aligned} \tag{285}$$

其中 $\delta^{(l,p)}=\frac{\partial\mathcal{L}(Y,\hat{Y})}{\partial Z^{(l,p)}}$ 为损失函数关于第 l 层的第 p 个特征映射净输入 $Z^{(l,p)}$ 的偏导数.

同理可得, 损失函数关于第 l 层的第 p 个偏置 $b^{(l,p)}$ 的偏导数为

$$\frac{\partial\mathcal{L}(Y,\hat{Y})}{\partial b^{(l,p)}}=\sum_{i,j}\left[\delta^{(l,p)}\right]_{i,j} \tag{286}$$

在卷积网络中, 每层参数的梯度依赖其所在层的误差项 $\delta^{(l,p)}$. 5.3.1 误差项的计算卷积层和汇聚层中, 误差项的计算有所不同, 因此我们分别计算其误差项. 汇聚层当第 $l+1$ 层为汇聚层时, 因为汇聚层是下采样操作, $l+1$ 层的每个神经元的误差项 δ 对应于第 l 层的相应特征映射的一个区域. l 层的第 p 个特征映射中的每个神经元都有一条边和 $l+1$ 层的第 p 个特征映射中的一个神经元相连. 根据链式法则, 第 l 层的一个特征映射的误差项 $\delta^{(l,p)}$, 只需要将 $l+1$ 层对应特征映射的误差项 $\delta^{(l+1,p)}$ 进行上采样操作 (和第 l 层的大小一样), 再和 l 层特征映射的激活值偏导数逐元素相乘, 就得到了 $\delta^{(l,p)}$. 则第 l 层的第 p 个特征映射的误差项 $\delta^{(l,p)}$ 为:

$$\begin{aligned}\delta^{(l,p)} &\triangleq \frac{\partial\mathcal{L}(Y,\hat{Y})}{\partial Z^{(l,p)}} \\ &= \frac{\partial X^{(l,p)}}{\partial Z^{(l,p)}}\cdot\frac{\partial Z^{(l+1,p)}}{\partial X^{(l,p)}}\cdot\frac{\partial\mathcal{L}(Y,\hat{Y})}{\partial Z^{(l+1,p)}} \\ &= f_l'\left(Z^{(l,p)}\right)\odot up\left(\delta^{(l+1,p)}\right)\end{aligned} \tag{287}$$

其中 $f_l'(\cdot)$ 为第 l 层使用的激活函数导数, up 为上采样函数, 与汇聚层中使用的下采样操作刚好相反. 如果下采样是最大汇聚, 误差项 $\delta^{(l+1,p)}$ 中每个值会直接传递到上一层对应区域中的最大值所对应的神经元, 该区域中其它神经元的误差项的都设为 0 . 如果下采样是平均汇聚, 误差项 $\delta^{(l+1,p)}$ 中每个值会被平均分配到上一层对应区域中的所有神经元上. 卷积层当 $l+1$ 层为卷积层时, 假设特征映射净输入 $Z^{(l+1)} \in \mathbb{R}^{M' \times N' \times P}$, 其中第 $p(1 \leq p \leq P)$ 个特征映射净输入

$$Z^{(l+1,p)} = \sum_{d=1}^{D} W^{(l+1,p,d)} \otimes X^{(l,d)} + b^{(l+1,p)} \tag{288}$$

其中 $W^{(l+1,p,d)}$ 和 $b^{(l+1,p)}$ 为第 $l+1$ 层的卷积核以及偏置. 第 $l+1$ 层中共有 $P \times D$ 个卷积核和 P 个偏置. 进而第 l 层的第 d 个特征映射的误差项 $\delta^{(l,d)}$ 为:

$$\begin{aligned}
\delta^{(l,d)} &\triangleq \frac{\partial \mathcal{L}(Y,\hat{Y})}{\partial Z^{(l,d)}} \\
&= \frac{\partial X^{(l,d)}}{\partial Z^{(l,d)}} \cdot \frac{\partial \mathcal{L}(Y,\hat{Y})}{\partial X^{(l,d)}} \\
&= f_l'\left(Z^{(l)}\right) \odot \sum_{p=1}^{P} \left(rot180\left(W^{(l+1,p,d)}\right) \tilde{\otimes} \frac{\partial \mathcal{L}(Y,\hat{Y})}{\partial Z^{(l+1,p)}} \right) \\
&= f_l'\left(Z^{(l)}\right) \odot \sum_{p=1}^{P} \left(rot180\left(W^{(l+1,p,d)}\right) \tilde{\otimes} \delta^{(l+1,p)} \right)
\end{aligned} \tag{289}$$

其中 $\tilde{\otimes}$ 为宽卷积.

5.2.3 集成学习,bagging 与 boosting 与 stacking

集成学习是先通过现有学习算法产生一组" 个体学习器"(基学习器), 再通过某种策略将他们结合起来.

给定一个学习任务, 假设输入 x 和输出 y 的真实关系为 $y = h(x)$. 对于 M 个不同的模型 $f_1(x), \cdots, f_M(x)$, 每个模型的期望错误为

$$\begin{aligned}
\mathcal{R}\left(f_m\right) &= \mathbb{E}_x\left[\left(f_m(x) - h(x)\right)^2\right] \\
&= \mathbb{E}_x\left[\varepsilon_m(x)^2\right]
\end{aligned} \tag{290}$$

其中 $\varepsilon_m(x) = f_m(x) - h(x)$ 为模型 m 在样本 x 上的错误. 那么所有的模型的平均错误为

$$\overline{\mathcal{R}}(f) = \frac{1}{M} \sum_{m=1}^{M} \mathbb{E}_x\left[\varepsilon_m(x)^2\right] \tag{291}$$

最直接的集成学习策略就是直接平均, 即" 投票". 基于投票的集成模型 $F(x)$ 为

$$F(x) = \frac{1}{M} \sum_{m=1}^{M} f_m(x) \tag{292}$$

对于 M 个不同的模型 $f_1(x), \cdots, f_M(x)$, 其平均期望错误为 $\overline{\mathcal{R}}(f)$. 基于简单投票机制的集成模型 $F(x) = \frac{1}{M}\sum_{m=1}^{M} f_m(x)$, $F(x)$ 的期望错误在 $\frac{1}{M}\overline{\mathcal{R}}(f)$ 和 $\overline{\mathcal{R}}(f)$ 之间.

证明: 根据定义, 集成模型的期望错误为

$$\begin{aligned}\mathcal{R}(F) &= \mathbb{E}_x\left[\left(\frac{1}{M}\sum_{m=1}^{M} f_m(x) - h(x)\right)^2\right] \\ &= \frac{1}{M^2}\mathbb{E}_x\left[\left(\sum_{m=1}^{M}\varepsilon_m(x)\right)^2\right] \\ &= \frac{1}{M^2}\mathbb{E}_x\left[\sum_{m=1}^{M}\sum_{n=1}^{M}\varepsilon_m(x)\varepsilon_n(x)\right] \\ &= \frac{1}{M^2}\sum_{m=1}^{M}\sum_{n=1}^{M}\mathbb{E}_x\left[\varepsilon_m(x)\varepsilon_n(x)\right]\end{aligned} \tag{293}$$

其中 $\mathbb{E}_x\left[\varepsilon_m(x)\varepsilon_n(x)\right]$ 为两个不同模型错误的相关性. 如果每个模型的错误不相关, 即 $\forall m \neq n, \mathbb{E}_x\left[\varepsilon_m(x)\varepsilon_n(x)\right] = 0$. 如果每个模型的错误都是相同的, 则 $\forall m \neq n, \varepsilon_m(x) = \varepsilon_n(x)$. 并且由于 $\varepsilon_m(x) \geq 0, \forall m$, 可以得到

$$\overline{\mathcal{R}}(f) \geq \mathcal{R}(F) \geq \frac{1}{M}\overline{\mathcal{R}}(f) \tag{294}$$

即集成模型的期望错误是大于等于所有模型的平均期望错误的 $1/M$, 小于等于所有模型的平均期望错误. 由此可以看出集成学习往往是很有效的.

但是一个有效的集成需要各个基模型的差异尽可能大. 为了增加模型之间的差异性, 通常可以采取 Bagging 类和 Boosting 类两类方法.

Bagging 类方法是通过随机构造训练样本、随机选择特征等方法来提高每个基模型的独立性, 代表性方法有 Bagging 和随机森林等.

首先介绍一种采样方法, 自助采样法. 给定包含 n 个样本的数据集 D, 我们对它进行采样产生数据集 D' : 每次随机从 D 中挑选一个样本, 将其拷贝放入 D', 然后再将该样本放回初始数据集 D 中, 使得该样本在下次采样时仍有可能被采到; 这个过程重复执行 n 次后, 我们就得到了包含 n 个样本的数据集 D', 这就是自助采样的结果. 显然, D 中有一部分样本会在 D' 中多次出现, 而另一部分样本不出现. 可以做一个简单的估计, 样本在 n 次采样中始终不被采到的概率是 $\left(1 - \frac{1}{m}\right)^m$, 取极限得到

$$\lim_{m \mapsto \infty}\left(1 - \frac{1}{m}\right)^m \mapsto \frac{1}{e} \approx 0.368 \tag{295}$$

即通过自助采样, 初始数据集 D 中约有 36.8% 的样本未出现在采样数据集 D' 中. 于是我们可将 D' 用作训练集, $D \backslash D'$ 用作测试集; 这样, 实际评估的模型与期望评估的模型都使用 n 个训练样本, 而我们仍有数据总量约 1/3 的、没在训练集中出现的样本用于测试.

Bagging 是一个通过不同模型的训练数据集的独立性来提高不同模型之间的独立性, 直接基于自助采样法. 给定包含 n 个样本的数据集, 我们采用自助采样法采样, 训练集中有的样本在采样集里多次出现, 有的则从未出现. 由式 (295) 可知, 初始训练集中约有 63.2% 的样本出现在采样集中. 照这样, 我们可采样出 m 个含 n 个训练样本的采样集, 然后基于每个采样集训练出一个基学习器, 再将这些基学习器进行结合. 这就是 Bagging 的基本流程. 在对预测输出进行结合时, Bagging 通常对分类任务使用简单投票法, 对回归任务使用简单平均法. 若分类预测时出现两个类收到同样票数的情形, 则最简单的做法是随机选择一个, 也可进一步考察学习器投票的置信度来确定最终结果. 值得注意的是, 一般来说 bagging 中的基学习器的类型是一致的

随机森林是在 Bagging 的基础上再引入了随机特征, 进一步提高每个基模型之间的独立性. 在随机森林中, 每个基模型都是一棵决策树.

Boosting 类方法是按照一定的顺序来先后训练不同的基模型, 每个模型都针对前序模型的错误进行专门训练. 根据前序模型的结果, 来调整训练训练样本的权重, 从而增加不同基模型之间的差异性.Boosting 类方法是一种非常强大的集成方法, 只要基模型的准确率比随机猜测好, 就可以通过集成方法来显著地提高集成模型的准确率.Boosting 类方法的代表性方法有 AdaBoost 等.

AdaBoost 是迭代的方法来学习每个弱分类器, 即按照一定的顺序依次训练每个弱分类器. 在学习了第 m 个弱分类器后, 增加其分错样本的权重, 使得第 $m+1$ 个弱分类器" 更关注" 于前面弱分类器分错的样本. 这样增加每个弱分类器的差异, 最终提升的集成分类器的准确率.

以两类分类为例, 先后训练弱分类器 $f_m(x) \in \{+1, -1\}$. 最初赋予每个样本同样的权重. 在每一轮迭代中, 根据当前的样本权重训练一个新的弱分类器. 然后根据这个弱分类器的错误率来计算其集成权重, 并调整样本权重.

algorithm 4 二分类的 AdaBoost

input: $\left\{x^{(i)}, y^{(i)}\right\}_{i=1}^{N}$, 迭代次数 M

output: $F(x) = sign\left(\sum_{m=1}^{M} \alpha_m f_m(x)\right)$

初始样本权重:$w_1^{(i)} = \frac{1}{N}$

for $m = 1 \cdots M$ **do**

1. 按照样本权重 $w_m^{(1)}, \cdots, w_m^{(N)}$, 学习弱分类器 f_m;
2. 计算弱分类器 f_m 在数据集上的加权错误为 ε_m;
3. 计算分类器的集成权重:
   $\alpha_m \leftarrow \frac{1}{2} \log \frac{1-\varepsilon_m}{\varepsilon_m}$;
4. 调整样本权重:
   $w_{m+1}^{(i)} \leftarrow w_m^{(i)} \exp\left(-\alpha_m y^{(i)} f_m\left(x^{(i)}\right)\right), \forall i \in [1, N]$;

end for

AdaBoost 算法可以看做是一种分步优化的加性模型,

$$F(x) = \sum_{m=1}^{M} \alpha_m f_m(x) h_t(x) \tag{296}$$

AdaBoost 是通过基学习器的线性组合来最小化指数损失函数, 损失函数定义为

$$\mathcal{L}(F(x)) = \mathcal{L}\left(\alpha_m, f_m(x)\right) = \mathbb{E}_{x \sim \mathcal{D}}\left[\exp(-yF(x))\right] = \mathbb{E}_{x \sim \mathcal{D}}\left[\exp\left(-y \sum_{m=1}^{M} \alpha_m f_m(x)\right)\right] \tag{297}$$

其中 $y, f_m(x) \in \{+1, -1\}$.

若 $F(x)$ 能令指数损失函数最小化, 则考虑式损失函数对 $F(x)$ 的偏导

$$\frac{\partial \mathcal{L}(F(x))}{\partial F(x)} = -e^{-F(x)} P(y = 1 \mid x) + e^{F(x)} P(y = -1 \mid x) \tag{298}$$

令偏导数为 0 得到

$$F(x) = \frac{1}{2} \ln \frac{P(y = 1 \mid x)}{P(y = -1 \mid x)} \tag{299}$$

因此, 有

$$sign(F(x)) = sign\left(\frac{1}{2}\ln\frac{P(y=1\mid x)}{P(y=-1\mid x)}\right) \quad = \begin{cases} 1, & P(y=1\mid x) > P(y=-1\mid x) \\ -1, & P(y=1\mid x) < P(y=-1\mid x) \end{cases} = \underset{y\in\{-1,1\}}{\arg\max} P(f(x)=y\mid x) \tag{300}$$

即 $sign(F(x))$ 达到了贝叶斯最优错误率. 也就是说, 若指数损失函数最小化, 则分类错误率也将最小化.

假设经过 $m-1$ 次迭代, 得到

$$F_{m-1}(x) = \sum_{t=1}^{m-1}\alpha_t f_t(x) \tag{301}$$

则第 m 次迭代的目标是找一个 α_m 和 $f_m(x)$ 使得下面的损失函数最小.

$$\mathcal{L}\left(\alpha_m, f_m(x)\right) = \sum_{n=1}^{N}\exp\left(-y^{(i)}\left(F_{m-1}\left(x^{(i)}\right) + \alpha_m f_m\left(x^{(i)}\right)\right)\right) \tag{302}$$

令 $w_m^{(i)} = \exp\left(-y^{(i)}F_{m-1}\left(x^{(i)}\right)\right)$, 则损失函数可以写为

$$\mathcal{L}\left(\alpha_m, f_m(x)\right) = \sum_{n=1}^{N} w_m^{(i)}\exp\left(-\alpha_m y^{(i)} f_m\left(x^{(i)}\right)\right) \tag{303}$$

因为 $y, f_m(x) \in \{+1, -1\}$, 有

$$yf_m(x) = 1 - 2I\left(y \neq f_m(x)\right) \tag{304}$$

其中 $I(x)$ 为指示函数. 将损失函数在 $-\alpha_m y^{(i)} f_m\left(x^{(i)}\right) = 0$ 处进行二阶泰勒展开, 有

$$\mathcal{L}\left(\alpha_m, f_m(x)\right) = \sum_{n=1}^{N} w_m^{(i)}\left(1 - \alpha_m y^{(i)} f_m\left(x^{(i)}\right) + \frac{1}{2}\alpha_m^2\right) \propto \alpha_m \sum_{n=1}^{N} w_m^{(i)} I\left(y^{(i)} \neq f_m\left(x^{(i)}\right)\right) \tag{305}$$

从上式可以看出, 当 $\alpha_m > 0$ 时, 最优的分类器 $f_m(x)$ 为使得在样本权重为 $w_m^{(i)}, 1 \leq n \leq N$ 时的加权错误率最小的分类器. 在求解出 $f_m(x)$ 之后, 损失函数可写为

$$\mathcal{L}\left(\alpha_m, f_m(x)\right) = \sum_{y^{(i)}=f_m\left(x^{(i)}\right)} w_m^{(i)}\exp\left(-\alpha_m\right) + \sum_{y^{(i)}\neq f_m\left(x^{(i)}\right)} w_m^{(i)}\exp\left(\alpha_m\right) \propto \left(1-\varepsilon_m\right)\exp\left(-\alpha_m\right) + \varepsilon_m\exp\left(\alpha_m\right) \tag{306}$$

其中 ε_m 为分类器 $f_m(x)$ 的加权错误率, 其中 ε_m 为分类器 $f_m(x)$ 的加权错误率,

$$\varepsilon_m = \frac{\sum_{y^{(i)}\neq f_m\left(x^{(i)}\right)} w_m^{(i)}}{\sum_n w_m^{(i)}} \tag{307}$$

求上式关于 α_m 的导数并令其为 0 , 得到

$$\alpha_m = \frac{1}{2}\log\frac{1-\varepsilon_m}{\varepsilon_m} \tag{308}$$

stacking 与 bagging 和 boosting 主要存在两方面的差异. 首先,Stacking 通常考虑的是不同类型的弱学习器（不同的学习算法被组合在一起）, 而 bagging 和 boosting 主要考虑的是相同类型的弱学习器. 其次,stacking 学习用元模型 (一个机器学习模型) 来组合基础模型, 而 bagging 和 boosting 则根据确定性算法组合弱学习器.

5.2.4 梯度提升决策树 (Gradient Boosting Decision Tree,GBDT)

介绍 GBDT 之前首先要介绍 CART 树. Classification And Regression Tree(CART) 是决策树的一种.CART 是一棵二叉树, 每一次分裂会产生两个子节点.CART 树分为分类树和回归树. 两者的区别在于输出值是离散值 (如标签) 还是连续值. 如果是分类树, 将选择能够最小化分裂后节点 GINI 值的分裂属性；如果是回归树, 选择能够最小化两个节点样本方差的分裂属性.CART 跟其他决策树算法一样, 需要进行剪枝, 才能防止算法过拟合从而保证算法的泛化性能.

首先是 CART 分类树,CART 分类树预测分类离散型数据, 采用基尼指数选择最优特征, 同时决定该特征的最优二值切分点. 分类过程中, 假设有 K 个类, 样本点属于第 k 个类的概率为 p_k, 则概率分布的基尼指数定义为

$$Gini(p) = \sum_{k=1}^{m} p_k \left(1 - p_k\right) = 1 - \sum_{k=1}^{K} p_k^2 \tag{309}$$

直观来说反映了从数据集中随机抽取两个样本其标签不一致的概率. 根据基尼指数定义, 可以得到样本集合 D 的基尼指数, 其中 C_k 表示数据集 D 中属于第 k 类的样本子集.

$$Gini(D) = 1 - \sum_{k=1}^{K} \left(\frac{|C_k|}{|D|}\right)^2 \tag{310}$$

如果数据集 D 根据特征 A 在某一取值 a 上进行分割, 得到 D1,D2 两部分后, 那么在特征 A 下集合 D 的基尼系数如下所示. 其中基尼系数 Gini(D) 表示集合 D 的不确定性, 基尼系数 Gini (D, A) 表示 $A =$ a 分割后集合 D 的不确定性. 基尼指数越大, 样本集合的不确定性越大.

$$GainGini(D, A) = \frac{|D1|}{|D|} Gini\left(D_1\right) + \frac{|D1|}{|D|} Gini\left(D_2\right) \tag{311}$$

对于属性 A, 分别计算任意属性值将数据集划分为两部分之后的 GainGini, 选取其中的最小值, 作为属性 A 得到的最优二分方案. 然后对于训练集 S, 计算所有属性的最优二分方案, 选取其中的最小值, 作为样本及 S 的最优二分方案.

$$\min_{A \in Attribute} \left(\min_{i \in A} \left(GainGini(D, A)\right)\right) \tag{312}$$

接下来是 CART 回归树,CART 回归树预测回归连续型数据, 假设 X 与 Y 分别是输入和输出变量, 并且 Y 是连续变量. 在训练数据集所在的输入空间中, 递归的将每个区域划分为两个子区域并决定每个子区域上的输出值, 构建二叉决策树, 所以实际上回归树算是不断递归的分类树, 递归至划分足够细.

选择最优切分特征 j 与切分点 s : 遍历特征 j , 对规定的切分变量扫描切分点 s , 选择使下式得到最小值时的 (j, s) 对.

$$L(j, s) = \min_{j, s} \left[\min_{c_1} \sum_{x^{(i)} \in R_i(j, s)} \left(y_i - c_1\right)^2 + \min_{c_2} \sum_{x^{(i)} \in R_i(j, s)} \left(y_i - c_2\right)^2\right] \tag{313}$$

其中 R_m 是被划分的输入空间, c_m 是空间 R_m 对应的固定输出值, 用选定的 (j, s) 对, 划分区域并决定相应的输出值

$$\begin{aligned} R_1(j, s) = \left\{x \mid x^{(j)} \leq s\right\}, R_2(j, s) = \left\{x \mid x^{(j)} > s\right\} \\ \hat{c}_m = \frac{1}{N_m} \sum_{x^{(i)} \in R_m(j, s)} y_i, m = 1, 2 \end{aligned} \tag{314}$$

继续对两个子区域调用上述步骤, 将输入空间划分为 M 个区域 R1,R2,..., Rm , 生成决策树.

$$f(x)=\sum_{m=1}^{M}\hat{c}_m I\left(x\in R_m\right) \tag{315}$$

当输入空间划分确定时, 可以用平方误差来表示回归树对于训练数据的预测方法, 用平方误差最小的准则求解每个单元上的最优输出值.

$$\sum_{x^{(i)}\in R_m}\left(y_i-f\left(x^{(i)}\right)\right)^2 \tag{316}$$

接下来是提升树 (boosting tree), 提升方法实际采用加法模型（即基函数的线性组合）与前向分步算法. 以决策树为基函数的提升方法称为提升树 (boosting tree). 对分类问题决策树是二叉分类树, 对回归问题决策树是二叉回归树. 在例 8.1 中看到的基本分类器 $x<v$ 或 $x>v$, 可以看作是由一个根结点直接连接两个叶结点的简单决策树, 即所谓的决策树桩 (decision stump). 提升树模型可以表示为决策树的加法模型:

$$f_M(x)=\sum_{m=1}^{M}T\left(x;\Theta_m\right) \tag{317}$$

其中, $T\left(x;\Theta_m\right)$ 表示决策树; Θ_m 为决策树的参数; M 为树的个数.

提升树算法采用前向分步算法. 首先确定初始提升树 $f_0(x)=0$, 第 m 步的模型是

$$f_m(x)=f_{m-1}(x)+T\left(x;\Theta_m\right) \tag{318}$$

其中, $f_{m-1}(x)$ 为当前模型, 通过经验风险极小化确定下一棵决策树的参数 Θ_m,

$$\hat{\Theta}_m=\arg\min_{\Theta_m}\sum_{i=1}^{N}L\left(y_i,f_{m-1}\left(x^{(i)}\right)+T\left(x^{(i)};\Theta_m\right)\right) \tag{319}$$

由于树的线性组合可以很好地拟合训练数据, 即使数据中的输入与输出之间的关系很复杂也是如此, 所以提升树是一个高功能的学习算法.

提升树利用加法模型与前向分步算法实现学习的优化过程. 当损失函数是平方损失和指数损失函数时, 每一步优化是很简单的. 但对一般损失函数而言, 往往每一步优化并不那么容易. 针对这一问题, Freidman 提出了梯度提升 (gradient boosting）算法. 这是利用最速下降法的近似方法, 其关键是利用损失函数的负梯度在当前模型的值

$$r_{mi}=-\left[\frac{\partial L\left(y,f\left(x^{(i)}\right)\right)}{\partial f\left(x^{(i)}\right)}\right]_{f(x)=f_{x-1}(x)} \tag{320}$$

作为回归问题提升树算法中的残差的近似值, 拟合一个回归树.

algorithm 5 GBDT

input: 训练数据集 $T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}, x^{(i)}\in\mathcal{X}\subseteq\mathbf{R}^n, y_i\in\mathcal{Y}\subseteq\mathbf{R}$; 损失函数 $L(y,f(x))$;

output: 回归树 $\hat{f}(x)$.

初始化

$$f_0(x)=\arg\min_c\sum_{i=1}^{N}L(y_i,c) \tag{321}$$

for $m=1\cdots M$ **do**

(a) 对 $i=1,2,\cdots,N$, 计算

$$r_{mi}=-\left[\frac{\partial L(y_i,f(x_t))}{\partial f(x_t)}\right]_{f(x)=f_{m-1}(x)}$$

(b) 对 r_{mi} 拟合一个回归树, 得到第 m 棵树的叶结点区域 $R_{mj},j=1,2,\cdots,J$

(c) 对 $j=1,2,\cdots,J$, 计算

$$c_{mj}=\arg\min_c\sum_{x^{(i)}\in R_{mj}}L\left(y_i,f_{m-1}\left(x^{(i)}\right)+c\right) \tag{322}$$

(d) 更新 $f_m(x)=f_{m-1}(x)+\sum_{j=1}^{J}c_{mj}I(x\in R_{mj})$

end for

得到回归树

$$\hat{f}(x)=f_M(x)=\sum_{m=1}^{M}\sum_{j=1}^{J}c_{mj}I(x\in R_{mj}) \tag{323}$$

算法第 1 步初始化, 估计使损失函数极小化的常数值, 它是只有一个根结点的树. 迭代中的 (a) 步计算损失函数的负梯度在当前模型的值, 将它作为残差的估计. 对于平方损失函数, 它就是通常所说的残差; 对于一般损失函数, 它就是残差的近似值. 第 (b) 步估计回归树叶结点区域, 以拟合残差的近似值. 第 (c) 步利用线性搜索估计叶结点区域的值, 使损失函数极小化. 第 (d) 步更新回归树. 第 3 步得到输出的最终模型 $\hat{f}(x)$.

对于分类算法, 其损失函数一般有对数损失函数和指数损失函数两种: a) 如果是指数损失函数, 则损失函数表达式为

$$L(y,f(x))=\exp(-yf(x)) \tag{324}$$

b) 如果是对数损失函数, 假设类别数为 K, 则损失函数为:

$$L(y,f(x))=-\sum_{k=1}^{K}y_k\log p_k(x) \tag{325}$$

对于回归算法, 常用损失函数有如下 4 种:

a) 均方差, 这个是最常见的回归损失函数了

$$L(y,f(x))=(y-f(x))^2 \tag{326}$$

b) 绝对损失, 这个损失函数切很常见

$$L(y,f(x))=-\sum_{k=1}^{K}y_k\log p_k(x) \tag{327}$$

对应负梯度误差为:

$$sign\left(y_i - f\left(x^{(i)}\right)\right) \tag{328}$$

c) Huber 损失, 它是均方差和绝对损失的折中产物, 对于远离中心的异常点, 采用绝对损失, 而中心附近的点采用均方差. 这个界限一般用分位数点度量. 损失函数为:

$$L(y, f(x)) = \begin{cases} \frac{1}{2}(y - f(x))^2 & |y - f(x)| \leq \delta \\ \delta\left(|y - f(x)| - \frac{\delta}{2}\right) & |y - f(x)| > \delta \end{cases} \tag{329}$$

对应的负梯度误差为:

$$r\left(y_i, f\left(x^{(i)}\right)\right) = \begin{cases} y_i - f\left(x^{(i)}\right) & \left|y_i - f\left(x^{(i)}\right)\right| \leq \delta \\ \delta sign\left(y_i - f\left(x^{(i)}\right)\right) & \left|y_i - f\left(x^{(i)}\right)\right| > \delta \end{cases} \tag{330}$$

d) 分位数损失. 它对应的是分位数回归的损失函数, 表达式为

$$L(y, f(x)) = \sum_{y \geq f(x)} \theta |y - f(x)| + \sum_{y < f(x)} (1 - \theta)|y - f(x)| \tag{331}$$

其中 θ 为分位数, 需要我们在回归前指定. 对应的负梯度误差为:

$$r\left(y_i, f\left(x^{(i)}\right)\right) = \begin{cases} \theta & y_i \geq f\left(x^{(i)}\right) \\ \theta - 1 & y_i < f\left(x^{(i)}\right) \end{cases} \tag{332}$$

对于 Huber 损失和分位数损失, 主要用于健壮回归, 也就是减少异常点对损失函数的影响.

GBDT 主要的优点有:

1) 可以灵活处理各种类型的数据, 包括连续值和离散值.

2) 在相对少的调参时间情况下, 预测的准确率也可以比较高. 这个是相对 SVM 来说的.

3）使用一些健壮的损失函数, 对异常值的鲁棒性非常强. 比如 Huber 损失函数和 Quantile 损失函数.

GBDT 的主要缺点有:

1) 由于弱学习器之间存在依赖关系, 难以并行训练数据. 不过可以通过自采样的 SGBT 来达到部分并行.

参考文献

[1] 周志华. 机器学习

[2] 李航. 统计学习方法

[3] 范明, 范宏建. 数据挖掘导论

[4] 邱锡鹏. 神经网络与深度学习

[5] Van Der Maaten L, Postma E, Van den Herik J. Dimensionality reduction: a comparative[J]. J Mach Learn Res, 2009, 10(66-71): 13.

[6] He X, Niyogi P. Locality preserving projections[J]. Advances in neural information processing systems, 2003, 16: 153-160.

[7] Van der Maaten L, Hinton G. Visualizing data using t-SNE[J]. Journal of machine learning research, 2008, 9(11).

[8] Hinton G, Roweis S T. Stochastic neighbor embedding[C]//NIPS. 2002, 15: 833-840.

[9] Bezdek J C, Ehrlich R, Full W. FCM: The fuzzy c-means clustering algorithm[J]. Computers geosciences, 1984, 10(2-3): 191-203.

[10] Zhao X, Nie F, Wang R, et al. Robust Fuzzy K-Means Clustering With Shrunk Patterns Learning[J]. IEEE Transactions on Knowledge and Data Engineering, 2021.

[11] Belkin M, Niyogi P. Laplacian eigenmaps and spectral techniques for embedding and clustering[C]//Nips. 2001, 14(14): 585-591.

[12] Liang S, Srikant R. Why deep neural networks for function approximation?[J]. arXiv preprint arXiv:1610.04161, 2016.