1 分类的线性方法

1.1 线性拟合与感知器

既然分类的输出是离散值, 它当然也属于连续值. 于是, 我们可以简单地沿用线性回归方法, 即

$$y = g(\boldsymbol{x}; \boldsymbol{w}) = w_0 x_0 + \cdots + w_M x_M = \boldsymbol{w}^{\mathrm{t}} \boldsymbol{x}$$

回归模型输出的是连续值, 因此需要离散化. 例如, 当输出值为 $\{0, 1\}$ 时, 可以使用

$$g(\boldsymbol{x}) = \begin{cases} 1, & f(\boldsymbol{x}; \boldsymbol{w}) \geqslant \dfrac{1}{2} \\ 0, & f(\boldsymbol{x}; \boldsymbol{w}) < \dfrac{1}{2} \end{cases}$$

将结果离散化. 然而, 如果出现远离数据聚集处的点, 那么线性回归可能会因为迎合这些点而产生较大偏差. 因此, 我们可以直接用阶跃函数进行拟合, 即

$$y = g(\boldsymbol{x}; \boldsymbol{w}) = f\left(\boldsymbol{w}^{\mathrm{t}} \boldsymbol{x}\right) = H\left(\boldsymbol{w}^{\mathrm{t}} \boldsymbol{x}\right), \quad \text{where } H(x) = \begin{cases} 1, & x \geqslant 0 \\ 0, & x < 0 \end{cases}$$

其中 $f(z)$ 为激活函数, 对输入的 z 进行非线性变换得到结果. 这里采用的激活函数为阶跃函数 $H(z)$. 直接用上述定义的 $g(\boldsymbol{x}; \boldsymbol{w})$ 进行拟合 (例如进行最大似然法估计等), 这就是**感知器 (perceptron)** 模型.

定义 1.1 感知器 使用阶跃函数 $H(z)$ 作为激活函数的线性分类模型, 即

$$y = g(\boldsymbol{x}; \boldsymbol{w}) = H\left(\boldsymbol{w}^{\mathrm{t}} \boldsymbol{x}\right)$$

称为**感知器 (perceptron)** 模型.

1.2 逻辑回归

1.2.1 逻辑回归的模型

感知器模型的激活函数是阶跃函数, 它在 $x = 0$ 处不可导, 因此无法使用梯度下降法等涉及导数的方法进行优化. 为了替换成光滑的函数以便优化, 我们可以用 sigmoid 函数 $\sigma(z)$ 替换前面的阶跃函数:

$$y = g\left(\boldsymbol{x}; \boldsymbol{w}\right) = \sigma\left(\boldsymbol{w}^{\mathrm{t}} \boldsymbol{x}\right), \quad \text{where } \sigma(z) = \frac{1}{1 + e^{-z}}$$

这就是**逻辑回归 (logistic regression)** 模型.

定义 1.2 逻辑回归 使用 sigmoid 函数 $\sigma(z)$ 作为激活函数的线性分类模型, 即

$$y = g\left(\boldsymbol{x}; \boldsymbol{w}\right) = \sigma\left(\boldsymbol{w}^{\mathrm{t}} \boldsymbol{x}\right), \quad \text{where } \sigma(z) = \frac{1}{1 + e^{-z}}$$

称为**逻辑回归 (logistic regression)** 模型.

上述函数返回一个 $(0,1)$ 间的连续值. 因此, 我们需要将结果解读为样本属于某一类别的概率.

当 $\sigma\left(\boldsymbol{w}^{\mathrm{t}}\boldsymbol{x}\right)>0.5$ 时, 我们预测样本属于类别 1 的概率更大; 否则预测样本属于类别 0 的概率更大. 所预测的两种类别的边界称作**决策面 (Decision Boundary)**, 由下式描述:

$$\boldsymbol{w}^{\mathrm{t}}\boldsymbol{x}=0$$

这是 $\boldsymbol{x}$ 的空间中的一个线性平面.

1.2.2 逻辑回归的求解

与前面一样, 逻辑回归也可以归结为一个概率优化问题, 使用极大似然法即可. 仍然假定数据集为 $\mathcal{D}=\{\boldsymbol{x}_n,t_n\}_{n=1}^N$. 当 $t_n=1$ 时, 模型预测结果与 t_n 一致的概率时 $y_n=\sigma(\boldsymbol{w}^{\mathrm{t}}\boldsymbol{x}_n)$, 反之则为 $1-y_n$. 综合而言, 模型预测正确的概率为

$$p(t_n|\boldsymbol{w})=y_n^{t_n}(1-y_n)^{1-t_n}$$

假定数据点之间是独立的, 似然函数可以写成

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

定义交叉熵误差函数

$$E(\boldsymbol{w})=-\ln p(\boldsymbol{t}|\boldsymbol{w})=-\sum_{n=1}^{N}\left[t_n\ln y_n+(1-t_n)\ln(1-y_n)\right]$$

这样做是因为取对数后的函数是凸函数, 可以更好地进行求解. 为了利用梯度下降法求解 $\boldsymbol{w}$ 的最佳值, 我们先来推导误差函数的梯度 $\nabla_{\boldsymbol{w}}E(\boldsymbol{w})$. 首先有

$$\frac{\mathrm{d}\sigma(z)}{\mathrm{d}z}=\frac{-\mathrm{e}^{-z}}{-(1+\mathrm{e}^{-z})^2}=\sigma(z)[1-\sigma(z)]$$

于是

$$\begin{aligned}\nabla_{\boldsymbol{w}}E(\boldsymbol{w})&=-\sum_{n=1}^{N}\left(\frac{t_n}{y_n}-\frac{1-t_n}{1-y_n}\right)\frac{\mathrm{d}y_n}{\mathrm{d}\boldsymbol{w}}\\&=-\sum_{n=1}^{N}\left(\frac{t_n}{y_n}-\frac{1-t_n}{1-y_n}\right)\frac{\mathrm{d}\sigma(\boldsymbol{w}^{\mathrm{t}}\boldsymbol{x}_n)}{\mathrm{d}\boldsymbol{w}}\\&=-\sum_{n=1}^{N}\left(\frac{t_n}{y_n}-\frac{1-t_n}{1-y_n}\right)y_n(1-y_n)\boldsymbol{x}_n\\&=\sum_{n=1}^{N}(y_n-t_n)\boldsymbol{x}_n\end{aligned}$$

这一矢量给出了 $E(\boldsymbol{w})$ 随 $\boldsymbol{w}$ 增加最快的方向和速度. 因此, 梯度下降的迭代求解可以写为

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

当数据个数太多时, 梯度计算公式中的求和将耗费大量时间. 考虑到迭代过程本身就是一个逐步逼近最优解的过程, 每次的移动方向不一定要很精确, 因此可以利用部分数据而非全部数据计算梯度. 在极端条件下, 每次只利用一个训练数据点, 即

$$\boldsymbol{w}^{(\tau+1)} = \boldsymbol{w}^{(\tau)} - \eta \nabla_{\boldsymbol{w}} E_n(\boldsymbol{w}), \quad E_n(\boldsymbol{w}) = (y_n - t_n)\boldsymbol{x}_n$$

1.2.3 基函数和正则化的使用

基函数和正则化的使用与在线性回归中是类似的, 这里不多赘述. 引入基函数可以更好地描述决策面的形状, 它在基函数的线性空间中是一个平面, 但在样本数据空间中则可以表现地更为复杂.

1.2.4 多分类问题

处理多分类问题常有以下两种方法.

定义 1.3 一对余方法 **一对余方法 (one-versus-the-rest)** 的基本想法是每次考虑将样本分为一个类别和其余类别, 从而将多分类问题转化为多个二分类问题.

一对余方法适用于分类不互斥的方法.

定义 1.4 softmax 方法 **softmax** 方法对每个类别 i 定义

$$z_i(\boldsymbol{x}) = \boldsymbol{w}_{(i)}^{\mathrm{t}} \boldsymbol{x}$$

然后利用如下 softmax 函数计算 $\boldsymbol{x}$ 属于类别 i 的概率

$$y^{(i)}(\boldsymbol{x}) = \frac{\mathrm{e}^{z_i}}{\sum\limits_i \mathrm{e}^{z_i}} = \frac{\mathrm{e}^{\boldsymbol{w}_{(i)}^{\mathrm{t}} \boldsymbol{x}}}{\sum\limits_i \mathrm{e}^{\boldsymbol{w}_{(i)}^{\mathrm{t}} \boldsymbol{x}}}$$

softmax 方法满足各类别概率之和等于 1 的要求, 因此特别适用于类别互斥的情形.