第2章　线性模型

刘家锋

哈尔滨工业大学

第2章 线性模型

2.1 线性模型的基本形式

线性模型

- **基本形式**
  - 线性模型学习一个通过属性线性组合进行预测的函数：

$$f(\mathbf{x}) = w_1x_1 + w_2x_2 + \cdots + w_dx_d + b = \sum_{i=1}^{d} w_ix_i + b$$

  - 写成向量形式：

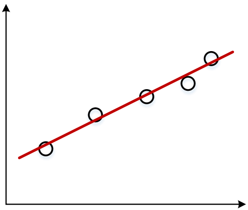
$$f(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + b, \qquad \mathbf{w} = (w_1, w_2, \cdots, w_d)^t$$

    其中，权向量$\mathbf{w}$和偏置b是需要学习的模型参数；
  - 例如西瓜问题学到的线性模型：

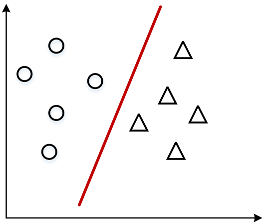
$$f_{\text{好瓜}}(\mathbf{x}) = 0.2 \cdot x_{\text{色泽}} + 0.5 \cdot x_{\text{根蒂}} + 0.3 \cdot x_{\text{敲声}} + 1$$

线性模型

- **线性模型的特点**
  - 线性模型形式简单，易于学习和建模；
  - 得到的模型易于理解，权值大小表征了属性的重要程度；
  - 线性模型的预测能力不强，但是很多非线性模型的基础；



线性回归



线性分类

2.2 线性回归

线性回归

- **线性回归任务**
  - 给定数据集$D = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}$，其中$y_i \in \mathbb{R}$;
  - 一般要求属性$\mathbf{x}_i = (x_{i1}, \cdots, x_{id})^t$为连续向量，离散的属性需要连续化;
  - 线性回归试图学习一个线性模型，尽可能准确地预测实值输出标记:

$$\hat{y} = f(\mathbf{x}) = \mathbf{w}^t\mathbf{x} + b$$

一元线性回归

- **一元线性回归问题**
  - 输入x是一元的标量，回归问题希望学习一组参数w, b，使得输入训练数据x_i，线性模型的输出尽量接近标记y_i：

$$f(x_i) = wx_i + b \quad \rightarrow \quad y_i$$

  - 线性回归可以通过最小化训练集上的平方误差，学习一组最优的模型参数：

$$(w^*, b^*) = \arg\min_{w,b} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$

  这种模型求解的方法称为“最小二乘法”；

最小二乘法求解

优化的目标函数：

$$E(w,b)=\sum_{i=1}^{m}\left(f(x_i)-y_i\right)^2=\sum_{i=1}^{m}\left(wx_i+b-y_i\right)^2$$

分别对w和b求导数，并求极值：

$$\frac{\partial E(w,b)}{\partial w}=2\left(w\sum_{i=1}^{m}x_i^2-\sum_{i=1}^{m}(y_i-b)x_i\right)=0$$

$$\frac{\partial E(w,b)}{\partial b}=2\left(mb-\sum_{i=1}^{m}(y_i-wx_i)\right)=0$$

得到：

$$w^*=\frac{\sum_{i=1}^{m}y_i(x_i-\bar{x})}{\sum_{i=1}^{m}x_i^2-\frac{1}{m}\left(\sum_{i=1}^{m}x_i\right)^2}\qquad b^*=\frac{1}{m}\sum_{i=1}^{m}(y_i-wx_i)$$

多元线性回归

- **多元线性回归问题**
  - 输入$\mathbf{x}$是d维的向量，回归问题希望学习一组参数$\mathbf{w}, b$，使得输入训练数据$\mathbf{x}_i$，线性模型的输出尽量接近标记y_i：

$$f(\mathbf{x}_i) = \mathbf{w}^t\mathbf{x}_i + b \quad \to \quad y_i$$

  - 最小二乘问题：

$$\begin{aligned}(\mathbf{w}^*, b^*) &= \arg\min_{\mathbf{w},b} \sum_{i=1}^{m} (f(\mathbf{x}_i) - y_i)^2 \\ &= \arg\min_{\mathbf{w},b} \sum_{i=1}^{m} (\mathbf{w}^t\mathbf{x}_i + b - y_i)^2\end{aligned}$$

向量和矩阵的导数

- **向量的导数**
  - 令$f(w_1,\cdots,w_d)$是一个d元函数，其中$\mathbf{w}=(w_1,\cdots,w_d)^t$；
  - 函数$f(\mathbf{w})$关于向量$\mathbf{w}$的导数称为梯度向量：

$$\nabla f(\mathbf{w})=\frac{\partial f(\mathbf{w})}{\partial \mathbf{w}}=\begin{pmatrix}\frac{\partial f}{\partial w_1}\\ \vdots\\ \frac{\partial f}{\partial w_d}\end{pmatrix}$$

- **矩阵的导数**
  - 令W为$d\times d$维矩阵，$f(W)$为定义在W元素上的d^2元函数；
  - 函数$f(W)$关于矩阵W的导数：

$$\frac{\partial f(W)}{\partial W}=\begin{pmatrix}\partial f/\partial w_{11} & \cdots & \partial f/\partial w_{1d}\\ \vdots & \ddots & \vdots\\ \partial f/\partial w_{d1} & \cdots & \partial f/\partial w_{dd}\end{pmatrix}$$

常用的微分公式

- **线性函数**
  - $f(\mathbf{x}) = \mathbf{x}^t\mathbf{y} = \sum_{i=1}^{d} x_i y_i$

$$\frac{\partial f}{\partial x_j} = y_j, \quad \frac{\partial f}{\partial \mathbf{x}} = \begin{pmatrix} \frac{\partial f}{\partial x_1} \\ \vdots \\ \frac{\partial f}{\partial x_d} \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_d \end{pmatrix} = \mathbf{y}$$

- **二次函数**
  - $f(\mathbf{x}) = \mathbf{x}^t A\mathbf{x}$

$$\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = (A + A^t)\mathbf{x}$$

当A为对称矩阵时，$\frac{\partial f(\mathbf{x})}{\partial \mathbf{x}} = 2A\mathbf{x}$

向量形式求解

优化的目标函数：

$$E(\mathbf{w},b)=\sum_{i=1}^{m}(f(\mathbf{x}_i)-y_i)^2=\sum_{i=1}^{m}(\mathbf{w}^t\mathbf{x}_i+b-y_i)^2$$

分别对$\mathbf{w}$和b求导数，并求极值：

$$\frac{\partial E(\mathbf{w},b)}{\partial \mathbf{w}}=2\sum_{i=1}^{m}(\mathbf{w}^t\mathbf{x}_i+b-y_i)\mathbf{x}_i=0$$

$$\frac{\partial E(\mathbf{w},b)}{\partial b}=2\sum_{i=1}^{m}(\mathbf{w}^t\mathbf{x}_i+b-y_i)=0$$

很难直接得到解析解的表达式；

矩阵形式求解

将权值向量$\mathbf{w}$和偏置b表示为一个向量：

$$\hat{\mathbf{w}} = \begin{pmatrix} \mathbf{w} \\ b \end{pmatrix}$$

输入数据和输出数据分别表示为矩阵和向量：

$$X = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1 \\ x_{21} & x_{22} & \cdots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{md} & 1 \end{pmatrix} = \begin{pmatrix} \mathbf{x}_1^t & 1 \\ \mathbf{x}_2^t & 1 \\ \vdots & \vdots \\ \mathbf{x}_m^t & 1 \end{pmatrix}, \quad \mathbf{y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_m \end{pmatrix}$$

优化的目标函数可以写成矩阵形式：

$$E(\hat{\mathbf{w}}) = \|X\hat{\mathbf{w}} - \mathbf{y}\|^2 = \sum_{i=1}^{m} (\mathbf{w}^t \mathbf{x}_i + b - y_i)^2$$

矩阵形式求解

优化的目标函数：

$$\begin{aligned} E(\hat{\mathbf{w}}) = \|X\hat{\mathbf{w}} - \mathbf{y}\|^2 &= (X\hat{\mathbf{w}} - \mathbf{y})^t (X\hat{\mathbf{w}} - \mathbf{y}) \\ &= \hat{\mathbf{w}}^t X^t X \hat{\mathbf{w}} - 2\hat{\mathbf{w}}^t X^t \mathbf{y} + \mathbf{y}^t \mathbf{y} \end{aligned}$$

对$\hat{\mathbf{w}}$求导数，并求极值：

$$\frac{\partial E(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}}} = 2X^t X\hat{\mathbf{w}} - 2X^t\mathbf{y} = 0$$

当X^tX为满秩矩阵时：($(X^tX)^{-1}X^t$称为X的伪逆矩阵)

$$\hat{\mathbf{w}}^* = (X^tX)^{-1}X^t\mathbf{y}$$

当X^tX不可逆时，可以正则化计算：(ϵ为小的正数，I为单位矩阵)

$$\hat{\mathbf{w}}^* = (X^tX + \epsilon I)^{-1}X^t\mathbf{y}$$

2.3 线性二分类

二分类问题

- **训练数据集**
  - 输入数据为向量$\mathbf{x}$，输出y标记类别：

$$y = \begin{cases} 0, & \mathbf{x}\text{是反例} \\ 1, & \mathbf{x}\text{是正例} \end{cases}$$

  - 训练数据集$D = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}$，$y_i \in \{0, 1\}$；
- **二分类线性判别**
  - 定义单位阶跃函数：

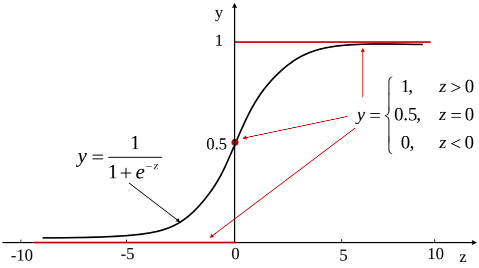
$$y = g(\mathbf{x}) = \begin{cases} 0, & \mathbf{w}^t\mathbf{x} + b < 0 \\ 0.5, & \mathbf{w}^t\mathbf{x} + b = 0 \\ 1, & \mathbf{w}^t\mathbf{x} + b > 0 \end{cases}$$

  - $y = 0.5$表示临界的预测值，类别可以任意判别；

二分类问题

- **阶跃函数与Sigmoid函数**
  - 阶跃函数不连续，可以使用Sigmoid函数近似阶跃函数：

$$y = \frac{1}{1+e^{-z}} = \frac{1}{1+e^{-(\mathbf{w}^t\mathbf{x}+b)}}$$



最小二乘求解

- **将分类问题视为回归问题**
  - 将类别标记y_i作为输入$\mathbf{x}_i$时的“期望”输出；
  - 按照线性回归的方式，学习权值向量$\mathbf{w}$和偏置b；
  - 分类时，根据$z = \mathbf{w}^t\mathbf{x} + b \lesseqgtr 0.5$判别；
- **最小二乘求解**
  - 平方误差损失函数优化：

$$\hat{\mathbf{w}}^* = \arg\min_{\hat{\mathbf{w}}} \|X\hat{\mathbf{w}} - \mathbf{y}\|^2$$

  - 伪逆解：

$$\hat{\mathbf{w}}^* = (X^tX)^{-1}X^t\mathbf{y}$$

对数几率回归

- **Logistic Regression**
  - 使用Sigmoid函数代替阶跃函数，并将其视为正例类别的“后验概率”：

$$P(y=1|\mathbf{x}) = \frac{1}{1+e^{-(\mathbf{w}^t\mathbf{x}+b)}} = \frac{e^{\mathbf{w}^t\mathbf{x}+b}}{1+e^{\mathbf{w}^t\mathbf{x}+b}}$$

  - 显然，反例类别的后验概率：

$$P(y=0|\mathbf{x}) = 1-P(y=1|\mathbf{x}) = \frac{1}{1+e^{\mathbf{w}^t\mathbf{x}+b}}$$

对数几率回归

- **判别方法**
  - 如果正例类别的后验概率大于反例类别，应该判别为正例，否则应该判别为反例：

$$y = \begin{cases} 0, & P(y=0|\mathbf{x}) > P(y=1|\mathbf{x}) \\ 1, & P(y=0|\mathbf{x}) < P(y=1|\mathbf{x}) \end{cases}$$

  - 计算后验概率之比的对数：

$$\ln \frac{P(y=1|\mathbf{x})}{P(y=0|\mathbf{x})} = \mathbf{w}^t\mathbf{x} + b \begin{cases} < 0, & y=0 \\ > 0, & y=1 \end{cases}$$

    得到与阶跃函数相同的判别条件；

对数几率回归

- **参数的学习**
  - 给定数据集$D = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$，学习模型的参数$\mathbf{w}, b$;
  - 希望学习到的参数使得数据集D中样本总的后验概率最大;
  - 极大似然估计，优化对数似然函数(详细内容见第5章):

$$(\mathbf{w}^*, b^*) = \arg\max_{\mathbf{w},b} \sum_{i=1}^{m} \ln P(y = y_i|\mathbf{x}_i; \mathbf{w}, b)$$

- **类别的后验概率**
  - 类别的后验概率满足Bernoulli分布:

$$P(y = y_i|\mathbf{x}_i) = P(y = 1|\mathbf{x}_i)^{y_i} \times P(y = 0|\mathbf{x}_i)^{1-y_i}$$

极大似然估计

简化符号，令：

$$\hat{\mathbf{w}} = \begin{pmatrix} \mathbf{w} \\ b \end{pmatrix}, \qquad \hat{\mathbf{x}_i} = \begin{pmatrix} \mathbf{x}_i \\ 1 \end{pmatrix}$$

将Logistic Regression代入后验概率的对数：

$$\begin{aligned} \ln P(y = y_i|\hat{\mathbf{x}}_i; \hat{\mathbf{w}}) &= y_i \ln P(y = 1|\hat{\mathbf{x}}_i; \hat{\mathbf{w}}) + (1 - y_i) \ln P(y = 0|\hat{\mathbf{x}}_i; \hat{\mathbf{w}}) \\ &= y_i \left[\hat{\mathbf{w}}^t \hat{\mathbf{x}}_i - \ln\left(1 + e^{\hat{\mathbf{w}}^t \hat{\mathbf{x}}_i}\right) \right] - (1 - y_i) \ln\left(1 + e^{\hat{\mathbf{w}}^t \hat{\mathbf{x}}_i}\right) \\ &= y_i \hat{\mathbf{w}}^t \hat{\mathbf{x}}_i - \ln\left(1 + e^{\hat{\mathbf{w}}^t \hat{\mathbf{x}}_i}\right) \end{aligned}$$

极大似然估计等价于最小化：

$$\hat{\mathbf{w}}^* = \arg\min_{\hat{\mathbf{w}}} \sum_{i=1}^{m} \left[-y_i \hat{\mathbf{w}}^t \hat{\mathbf{x}}_i + \ln\left(1 + e^{\hat{\mathbf{w}}^t \hat{\mathbf{x}}_i}\right) \right]$$

无法直接根据极值点条件得到$\hat{\mathbf{w}}^*$的解析解，需要采用梯度法迭代优化；

梯度法优化

- **目标函数的一阶近似**
  - 当$\Delta\mathbf{w}$充分小时，在$\mathbf{w}$点附近可以用一阶Taylor展开式近似目标函数$J(\mathbf{w})$：

$$J(\mathbf{w}+\Delta\mathbf{w}) \approx J(\mathbf{w}) + \nabla J^t(\mathbf{w})\Delta\mathbf{w}$$

  - 目标函数的梯度：

$$\nabla J(\mathbf{w}) = \frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \begin{pmatrix} \frac{\partial J}{\partial w_1} \\ \vdots \\ \frac{\partial J}{\partial w_d} \end{pmatrix}$$

- **梯度法**
  - 梯度法优化从一个初始值$\mathbf{w}_0$开始迭代；
  - 每一轮迭代，希望在当前值$\mathbf{w}_t$的附近，找到一个新的值$\mathbf{w}_{t+1}=\mathbf{w}_t+\Delta\mathbf{w}$，使得目标函数$J(\mathbf{w}_{t+1})$最小；

梯度法优化

- **梯度法迭代**
  - 限定增量向量$\Delta\mathbf{w}$的长度为1，在$\mathbf{w}$附近寻找一点$\mathbf{w}+\Delta\mathbf{w}$，使得目标函数$J(\mathbf{w}+\Delta\mathbf{w})$最小；
  - 显然，当$\Delta\mathbf{w}=-\nabla J(\mathbf{w})/\|\nabla J(\mathbf{w})\|$时，$J(\mathbf{w}+\Delta\mathbf{w})$取得最小值；
  - 考虑到一阶展开式的近似精度需要$\Delta\mathbf{w}$充分小，一般梯度法采用如下方式迭代：

$$\mathbf{w}_{t+1}=\mathbf{w}_t-\eta\nabla J(\mathbf{w}_t)$$

学习率η需要设置一个比较小的值，才能够保证收敛；

对数几率回归

对数似然的优化目标函数：

$$l(\hat{\mathbf{w}}) = \sum_{i=1}^{m}\left[-y_i\hat{\mathbf{w}}^t\hat{\mathbf{x}}_i + \ln\left(1 + e^{\hat{\mathbf{w}}^t\hat{\mathbf{x}}_i}\right)\right]$$

计算梯度：

$$\begin{aligned}\nabla l(\hat{\mathbf{w}}) &= \sum_{i=1}^{m}\left[-y_i\hat{\mathbf{x}}_i + \frac{e^{\hat{\mathbf{w}}^t\hat{\mathbf{x}}_i}}{1 + e^{\hat{\mathbf{w}}^t\hat{\mathbf{x}}_i}}\hat{\mathbf{x}}_i\right] \\ &= \sum_{i=1}^{m}\left(P(y=1|\hat{\mathbf{x}}_i;\hat{\mathbf{w}}) - y_i\right)\hat{\mathbf{x}}_i\end{aligned}$$

对数几率回归

Algorithm 1 Logistic Regression的梯度学习

Input: 数据集$D = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$，学习率$\eta$，收敛精度$\theta$

Output: 模型的参数$\hat{\mathbf{w}}^* = (\mathbf{w}^*; b^*)$

1: 随机初始化$\hat{\mathbf{w}}_0$，$k = 0$，$l_0 = 0$;

2: **repeat**

3: 计算后验概率：$P(y = 1|\hat{\mathbf{x}}_i; \hat{\mathbf{w}}_k), \quad i = 1, \cdots, m$

4: 学习参数：

$$\hat{\mathbf{w}}_{k+1} \leftarrow \hat{\mathbf{w}}_k - \eta \sum_{i=1}^m \left(P(y = 1|\hat{\mathbf{x}}_i; \hat{\mathbf{w}}_k) - y_i\right) \hat{\mathbf{x}}_i$$

5: 计算对数似然：

$$l_{k+1} = \sum_{i=1}^m \ln P(y = y_i|\hat{\mathbf{x}}_i; \hat{\mathbf{w}}_{k+1})$$

6: $k \leftarrow k + 1$

7: **until** $|l_k - l_{k-1}| < \theta$

8: **return** $\hat{\mathbf{w}}^* = \hat{\mathbf{w}}_k$;

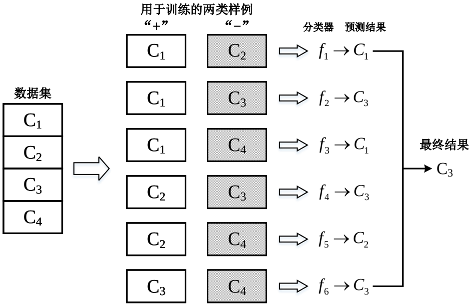
2.4 线性多分类

多分类问题

- **多分类的学习**
  - 将二分类扩展到多分类问题，有N个类别$C_1, \cdots, C_N$；
  - 给定数据集$D = \{(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_m, y_m)\}, y_i \in \{1, \cdots, N\}$
  - 学习一组线性模型，用于区分N个类别；
- **学习方法**
  - 拆分法：多分类转化为多个二分类问题
  - 输出编码：用向量作为类别标记，同时学习多个线性模型

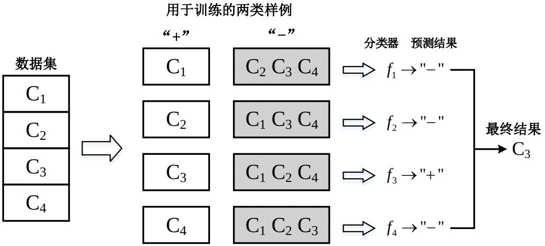
拆分方式一

- "一对一"方式：**One vs. One**
  - 学习$N(N-1)/2$个分类器，每个分类器区分两个类别；
  - 测试阶段，产生$N(N-1)/2$个二分类结果，投票获得多分类结果；



拆分方式二

- “一对其余”方式：**One vs. Rest**
  - 每次将一类作为正例，其余的作为反例，学习N个分类器；
  - 测试阶段，如果仅有一个分类器预测为正例，则得到最终分类结果；



输出编码

- **One-Hot编码**
  - 使用一个One-Hot的向量$\mathbf{y}_i$，表示类别标记y_i：

$$y_i = k \qquad \Longrightarrow \qquad \mathbf{y}_i = (\overbrace{0,\cdots,1,\cdots,0}^{N})^t$$

  - 测试样例$\mathbf{x}$的输出$\mathbf{y}$，以$\mathbf{y}$中最大元素作为类别预测结果：

$$\mathbf{y} = W^t\hat{\mathbf{x}}, \qquad y = \arg\max_{j=1,\cdots,N} y_j$$

其中：

$$W = \begin{pmatrix} \mathbf{w}_1 & \cdots & \mathbf{w}_N \\ b_1 & \cdots & b_n \end{pmatrix}, \qquad \hat{\mathbf{x}} = \begin{pmatrix} \mathbf{x} \\ 1 \end{pmatrix}$$

输出编码

- 参数的学习
  - 将数据集D写成矩阵形式：

$$X = \begin{pmatrix} \mathbf{x}_1^t & 1 \\ \vdots & \vdots \\ \mathbf{x}_m^t & 1 \end{pmatrix}, \quad Y = \begin{pmatrix} \mathbf{y}_1^t \\ \vdots \\ \mathbf{y}_m^t \end{pmatrix}$$

  - 优化平方误差损失函数：

$$W^* = \arg\min_W \|XW - Y\|_F^2$$

  - 伪逆解：

$$W^* = (X^tX)^{-1}X^tY$$