线性模型

2023年11月14日　8:33

基本形式

- **问题描述**：给定由d个属性描述的示例$x=(x_1;x_2;\dots;x_d)$其中xi是x在第i个属性上的取值，线性模型试图学得一个通过属性的线性组合来进行预测的函数
- **函数形式**：$f(x)=w_1x_1+w_2x_2+\dots w_dx_d+b$
- **向量形式**：$f(x)=w^Tx+b$
  其中$w=(w_1;w_2;\dots w_d)$，w和b学得后 模型就得以确定

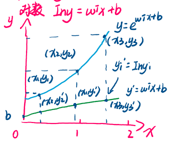
线性回归

- **问题描述**
  - 给定数据集 $D=\{(x_1,y_1),(x_2,y_2),\dots(x_m,y_m)\}$
  - 其中$x_i=(x_{i1};x_{i2};x_{i3};\dots;x_{id})$，$y_i\in R$
  - 线性回归试图学得一个线性模型以尽可能准确地预测实值输出标记
  - 矩阵表示法：$\begin{bmatrix} x_1=(x_{11},x_{12},x_{13},\dots x_{1n}) \\ \dots \\ x_m \end{bmatrix}_{m\times d}[w_1\ w_2\ \dots\ w_d]=\begin{bmatrix} y_1\\ y_2\\ y_3\\ \dots\\ y_m\end{bmatrix}_{m\times 1}$

- **一元线性回归**
  - 问题描述
    - 只有一个属性，即d=1，x是一个标量数
    - w,b为单个的数
  - 目标函数
    - 确定w和b的关键在于如何衡量f(x)与y之间的差别。而均方误差是回归任务中最常用的性能度量，因此我们可以试图让均方误差最小化
    - $(w^*,b^*)=\underset{(w,b)}{\arg\min}\sum_{i=1}^{m}(f(x_i)-y_i)^2=\underset{(w,b)}{\arg\min}\sum_{i=1}^{m}(y_i-wx_i-b)^2$ 也叫最小二乘法(就是试图找到一条直线 使所有样本到直线上的欧氏距离最小)
    - w^*代表w的解
  - 目标函数求解
    - 该过程也称为线性回归模型的对最小二乘“参数估计”
    - 对w求导 $\frac{\partial E(w,b)}{\partial w}=2\left(w\sum_{i=1}^{m}x_i-\sum_{i=1}^{n}(y_i-b)x_i\right)$（欧几里德距离）
      $E(w,b)=\arg\min\sum_{i=1}^{m}(f(x_i)-y_i)^2$ 这其中有2个参数，所以求偏导 $\begin{cases}\frac{\partial E}{\partial w}\\ \frac{\partial E}{\partial b}\end{cases}$
    - 对b求导 $\frac{\partial E(w,b)}{\partial b}=2\left(mb-\sum_{i=1}^{m}(y_i-wx_i)\right)$
    - 导数为0求得最优解
      - $w=\frac{\sum_{i=1}^{m}y_i(x_i-\bar{x})}{\sum_{i=1}^{m}x_i^2-\frac{1}{m}\left(\sum_{i=1}^{m}x_i\right)^2}$
      - $b=\frac{1}{m}\sum_{i=1}^{m}(y_i-wx_i)$
      - 其中$\bar{x}=\frac{1}{m}\sum_{i=1}^{m}x_i$，为x的均值

- **多元线性回归**
  - 关系式 $f(x_i)=w^Tx_i+b$，使得$f(x_i)\simeq y_i$（w,x为向量）
  - 目标函数
    - 类似地，可利用最小二乘法来对w和b进行估计。为便于讨论，我们把w和b吸收入向量形式$\hat{w}=(w;b)$ 相应的，把数据集D表示为一个$m\times(d+1)$ 大小的矩阵X，其中每一行对应于一个示例，该行前d个元素对应于示例的d个属性值，最后一个元素恒置为1
    - $X=\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1\\ x_{21} & x_{22} & \cdots & x_{2d} & 1\\ \vdots & \vdots & \ddots & & \vdots\\ x_{m1} & x_{m2} & \cdots & x_{md} & 1\end{pmatrix}=\begin{pmatrix} x_1^T & 1\\ x_2^T & 1\\ \vdots & \vdots\\ x_m^T & 1\end{pmatrix}$ 这里的$x_m=(x_{m1};x_{m2};\dots x_{md})$
    - 再把标记也写成向量形式$y=(y_1;y_2;\dots y_m)$
    - $\hat{w}^*=\underset{\hat{w}}{\arg\min}(y-X\hat{w})^T(y-X\hat{w})$

- **对数线性回归**
  - 原型 $y=w^Tx+b$
  - 对数 $\ln y=w^Tx+b$



对数线性回归示意图

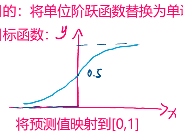
- **广义线性模型**
  - 考虑更一般的情况，有link function联系函数g()
  - $y=g^{-1}(w^Tx+b)$
  - 当y与x不是线性关系，而g(y)与x是线性关系时，即$g(y)=wx+b\Rightarrow y=g^{-1}(wx+b)$（g的反函数$g^{-1}$）

对数几率回归（逻辑回归）

- **问题描述**
  - 分类问题，y的取值个数有限
  - 考虑广义线性模型，只需要用一个单调可微的关系函数就可将回归转换成分类任务
- **二分类任务**
  - $y=\{0,1\}$
  - 关系函数
    - 单位阶跃函数 $y=\begin{cases}0, & z<0\\ 0.5, & z=0\\ 1, & z>0\end{cases}$

不连续，所以不可导

    - sigmoid函数
      - 目的：将单位阶跃函数替换为单调可微函数
      - 目标函数：



将预测值映射到[0,1]

      - 要求
        - $x\to+\infty$，$y\to1$，$y'\to0$
        - $x\to-\infty$，$y\to0$，$y'\to0$
        - $x=0$，$y=0.5$　$y'max=y'(0)$
        - 有$y'=y(1-y)$满足要求
      - 目标问题：已知y'，求y
        - 通式：将y,x改为P,t，有$P=P(t)$，则$\frac{dP}{dt}=kP(1-\frac{P}{k})$（k是常数）
        - 移项，分离变量，积分 $\int\frac{dP}{P(1-\frac{P}{k})}=\int k\,dt$
        - 左侧简化分式
          - 上下通式乘k $\frac{1}{P(1-\frac{P}{k})}=\frac{k}{P(k-P)}$
          - 偏分式求解 $\frac{k}{P(k-P)}=\frac{1}{P}+\frac{1}{k-P}$
        - 原式变为$\int(\frac{1}{P}+\frac{1}{k-P})dP=\int k\,dt$ 已知导数求原函数
        - 微分的逆运算 $\ln|P|-\ln|k-P|=kt+C\Leftrightarrow\ln|k-P|-\ln|P|=-kt-C\Leftrightarrow\ln\left|\frac{k-P}{k}\right|=-kt-C$
        - 反函数 $\left|\frac{k-P}{k}\right|=e^{-kt-C}\Leftrightarrow\left|\frac{k-P}{k}\right|=e^{-C}e^{-kt}\Leftrightarrow\frac{k-P}{k}=\pm e^{-C}e^{-kt}\Leftrightarrow\frac{k-P}{k}=Ae^{-kt}(A=\pm e^{-C})$
          $\Leftrightarrow\frac{k}{P}-1=Ae^{-kt}\Leftrightarrow\frac{k}{P}=1+Ae^{-kt}\Leftrightarrow\frac{P}{k}=\frac{1}{1+Ae^{-kt}}\Leftrightarrow P=\frac{k}{1+Ae^{-kt}}$
        - 即P的通式为 $P(t)=\frac{k}{1+Ae^{-kt}}$ 为了求出A，令t=0 ……
  - 函数式
    - $y=\frac{1}{1+e^{-(w^Tx+b)}}$
    - $\ln\frac{y}{1-y}=w^Tx+b$
    - 几率：x取1的相对可能性 $\frac{y}{1-y}$（x=1[即x为正例]的可能性为y，x=0的可能性为1-y）

$y=\frac{1}{1+e^{-(w^Tx+b)}}$ $x=0,\ y=0.5\quad y'_{max}=y'(0)$

反函数 $\left|\frac{k-p}{k}\right|=e^{-kt-c} \Leftrightarrow \left|\frac{k-p}{k}\right|=e^{c}e^{-kt} \Leftrightarrow \frac{k-p}{k}=\pm e^{c}e^{-kt} \Leftrightarrow \frac{k-p}{k}=Ae^{-kt}(A=\pm e^{c})$

$\Leftrightarrow \frac{k}{p}-1=Ae^{-kt} \Leftrightarrow \frac{k}{p}=1+Ae^{-kt} \Leftrightarrow \frac{p}{k}=\frac{1}{1+Ae^{-kt}} \Leftrightarrow p=\frac{k}{1+Ae^{-kt}}$

即P的通式为 $p(t)=\frac{k}{1+Ae^{-kt}}$ 为了求出A，令t=0 $p_0=\frac{k}{1+Ae^{0}} \Rightarrow A=\frac{k-p_0}{p_0}$

对应目标函数，x=0时y=0.5，即P(0)=0.5,又因y'=y(1-y)，即k=1，可计算A=1

至此得到 $y=\frac{1}{1+e^{-x}}$，sigmoid

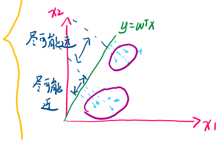
函数式

- $\ln\frac{y}{1-y}=w^Tx+b$
- 几率：x取1的相对可能性 $\frac{y}{1-y}$（x=1[即x为正例]的可能性为y，x=0的可能性为1-y）
- 对数几率：$\ln\frac{y}{1-y}$

本质：用线性回归模型的结果去逼近真实标记的对数几率

LDA线性判断分析

- 基本思想
  - 给定训练样例集，设法将样例投影到一条直线上，使得同类样例的投影点尽可能接近、异类样例的投影点尽可能远离；在对新样本进行分类时，将其投影到同样的这条直线上，再根据投影点来确定新样本的类别



LDA的二维示意图，"+","-"分别代表正例和反例，椭圆表示数据簇的外轮廓，虚线表示投影

多分类学习---拆解法

拆分为若干个二分类

类别不平衡问题

处理方法（以逻辑回归为例）

- 当正反例可能性相同 若 $\frac{y}{1-y}>1$，则预测为正例
- 假设训练集是样本总体的无偏采样
  - 总体样本是什么比例，采集过来就是什么比例
  - 当正例数量m+，反例数量m- 若 $\frac{y}{1-y}>\frac{m^+}{m^-}$，则预测为正例
  - 再缩放
    - 若分类器默认基于正反例可能性相同预测，只需 $\frac{y'}{1-y'}=\frac{y}{1-y}\times\frac{m^-}{m^+}$
- 若上面假设不成立，则无法基于训练集类别数量推断真实几率
  - 欠采样
    - 删除一些样本，使得正反例数目接近
    - 但也可能因此丢失一些重要信息
    - 解决方法：EasyEnsemble算法——将反例划分为若干个集合供不同的学习器使用，每个学习器都进行了欠采样，全局来看却不会丢失重要信息
  - 过采样
    - 增加一些样本
    - 不能简单地对初始样本进行重复采样，否则会招致严重的过拟合
  - 阈值移动
    - ”再缩放“也是”代价敏感学习“的基础，在代价敏感学习中m-/m+用cost+/cost-代替即可，其中cost+是将正例误分为反例的代价，cost-则是将反例误分为正例的代价

梯度下降

基本思想

己知樣本集x，預測函數 $f(x)=wx+b=\hat{y}$

對預次值和真實值作差得到損失函數 $loss=|Y-\hat{y}|$

若把f(x)表现为矩阵形式（表示为θ），则有 $loss=h(\theta)$

欲求loss的最小值，即對 $h'(\theta)$ 求導

当 $h'(\theta)=0$ 时，θ 就是要找的极值

但對于高維空間求導比较困難，可考慮以下方法

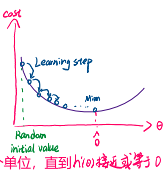
任意给定 θ 一个初始值，解出该值下的 loss 和 $h'(\theta)$

若 $h'(\theta)>0$，說明函數此时在上升，所以需要将 θ 回退几个单位，直到 $h'(\theta)$ 接近或等于 0

如当 $h'(\theta)>0$ 时，回退后的 $\theta'=\theta-h'(\theta)\cdot\eta$ 学习率，可自定

回退后再次求的loss和 $h'(\theta)$ 并循环往复

直到loss值小到我們可接受的范圍或 $h'(\theta)$ 越接近0



批量梯度下降

将上述问题扩展到多维时，就需要对每一个方向都求偏导（梯度）

$$\frac{\partial}{\partial\theta_j}MSE(\theta)=\frac{2}{m}\sum_{i=1}^{m}(\theta^T x^{(i)}-y^{(i)})x_j^{(i)}$$

损失函数对参数的偏导 $$\nabla_\theta MSE(\theta)=\begin{pmatrix}\frac{\partial}{\partial\theta_0}MSE(\theta)\\ \frac{\partial}{\partial\theta_1}MSE(\theta)\\ \vdots\\ \frac{\partial}{\partial\theta_n}MSE(\theta)\end{pmatrix}=\frac{2}{m}X^T(X\theta-y)$$

参数更新 $\theta^{(next\ step)}=\theta-\eta\nabla_\theta MSE(\theta)$

要求所有的方向都求出偏导后，才算走了一步，所以处理速度较慢

随机梯度下降

每次随机随机选择一个方向，每次迭代处理的数据很少，处理速度高

达到最小值不会稳定下来，会继续弹跳，因此算法停止时，最终的参数是一个比较好的参数，但不是最佳的

学习率(Learning Rate)

- 参数更新时，需要选择合适的学习率。若学习率太低，处理数据速度会变慢；如果太高，容易掠过最低点
- 寻找合适的学习率
  - 网格搜索
    - 使用网格搜索，设定迭代次数，尝试几种可能的参数，比较结果，选择最优
  - 梯度限制
    - 设置大量的迭代，当梯度向量变小于一个值是停止

局部最优和全局最优

有些时候我们找到的点可能只是局部最优，还需要添加其他参数辅助查找全局最优。但一般情况下，只要该局部最优点的损失是在可控范围内就可以了