逻辑回归 Logistic Regression

概述

逻辑回归是应用非常广泛的一个分类机器学习算法，它将数据拟合到一个logit函数(或者叫做logistic函数)中，从而能够完成对事件发生的概率进行预测。逻辑回归是一种极其高效的概率计算机制，与线性回归分析有很多相同之处。

模型

考虑二分类问题，模型为

$$P(Y=1|X)=g(z)=\frac{1}{1+e^{-z}}$$

$$P(Y=0|X)=1-g(z)=1-\frac{1}{1+e^{-z}}$$

函数g(z)又称为logistic function，一般取$z=x^T\theta$，实际上，也有其他取值在[0,1]的函数可以替代g(z)，在此暂且取用logistic function，之后再讨论这个模型与广义线性模型的关系

模型求解

g的一阶导数为

$$\frac{\partial g(z)}{\partial z}=\frac{\partial}{\partial z}\frac{1}{1+e^{-z}}=\frac{1}{(1+e^{-z})^2}e^{-z}=\frac{1}{(1+e^{-z})}\left(1-\frac{1}{(1+e^{-z})}\right)=g(z)[1-g(z)]$$

把概率密度函数改写为

$$p(y|x;\theta)=g(x^T\theta)^y(1-g(x^T\theta))^{1-y}$$

假设样本量为n，那么似然函数为

$$L(\theta)=\prod_{i=1}^{n}p(y_i|x_i;\theta)=\prod_{i=1}^{n}g(x_i^T\theta)^{y_i}[1-g(x_i^T\theta)]^{1-y_i}$$

得到对数似然函数

$$l(\theta)=\sum_{i=1}^{n}y_i\log g(x_i^T\theta)+(1-y_i)\log[1-g(x_i^T\theta)]$$

求极大似然估计可以直接用梯度下降法求$-l(\theta)$的极小值点

$$
\begin{aligned}
\frac{\partial l(\theta)}{\partial \theta_j} &= \sum_{i=1}^{n}\left[y_i \frac{1}{g(x_i^T\theta)} - (1-y_i)\frac{1}{1-g(x_i^T\theta)}\right]\frac{\partial g(x_i^T\theta)}{\partial \theta_j} \\
&= \sum_{i=1}^{n}\left[y_i \frac{1}{g(x_i^T\theta)} - (1-y_i)\frac{1}{1-g(x_i^T\theta)}\right] g(x_i^T\theta)[1-g(x_i^T\theta)]x_{ij} \\
&= \sum_{i=1}^{n}\left[y_i - g(x_i^T\theta)\right] x_{ij}
\end{aligned}
$$

求$-l(\theta)$的极小值点的梯度下降更新方程为

$$
\theta_j := \theta_j + \alpha \sum_{i=1}^{n}\left[y_i - g(x_i^T\theta)\right] x_{ij}
$$

用逻辑回归搭建多分类器

1 vs other

假设是个K分类问题，那么用1对其他的方法需要建立K个分类器，第i个分类器C_i对应的模型就是

$$
P(Y=i|X) = \frac{1}{1+e^{-x^T\theta}}
$$

$$
P(Y \neq i|X) = 1 - \frac{1}{1+e^{-x^T\theta}}
$$

1 vs 1

假设是个K分类问题，那么用1对其他的方法需要建立$K(K-1)/2$ 个分类器,第 $(i,j), i<j$ 个分类其对应的模型是

$$
P(Y=i|X) = \frac{1}{1+e^{-x^T\theta}}
$$

$$
P(Y=j|X) = 1 - \frac{1}{1+e^{-x^T\theta}}
$$

拓展模型(softmax回归，multinomial假设)

考虑m分类问题，模型为

$$P(Y=1|X)=g(z_1)=e^{-x^T\theta_1}\,/\,\sum_{k=1}^{m}e^{-x^T\theta_k}$$

$$P(Y=2|X)=g(z_2)=e^{-x^T\theta_2}\,/\,\sum_{k=1}^{m}e^{-x^T\theta_k}$$

$$\ldots\ldots$$

$$P(Y=m|X)=g(z_m)=e^{-x^T\theta_m}\,/\,\sum_{k=1}^{m}e^{-x^T\theta_k}$$

则对数似然函数为

$$l(\theta)=\sum_{i=1}^{n}\sum_{k=1}^{m}\mathbf{1}_{(y_i=k)}\log g(x_i^T\theta_k)$$

解法与原逻辑回归模型一样，对$l(\theta)$求导

$$\begin{aligned}\frac{\partial l(\theta)}{\partial\theta_{kj}} &= \sum_{i=1}^{n}\sum_{k=1}^{m}\mathbf{1}_{(y_i=k)}\frac{1}{g(x_i^T\theta_k)}\frac{\partial g(x_i^T\theta_k)}{\partial\theta_{kj}}\\ &= \sum_{i=1}^{n}\sum_{k=1}^{m}\mathbf{1}_{(y_i=k)}\frac{1}{g(x_i^T\theta_k)}g(x_i^T\theta_k)[1-g(x_i^T\theta)]x_{ij}\\ &= \sum_{i=1}^{n}\sum_{k=1}^{m}\mathbf{1}_{(y_i=k)}[1-g(x_i^T\theta_k)]x_{ij}\end{aligned}$$

求$-l(\theta)$的极小值点的梯度下降更新方程为

$$\theta_{kj}:=\theta_{kj}+\alpha\sum_{i=1}^{n}\left[y_i-g(x_i^T\theta_k)\right]x_{ij}$$

θ_{kj} 为 θ_k 的第 j 个分量，注意到这个梯度下降更新方程的形式与前面是相同的，实际上，由于multinomial属于指数族，在广义线性模型中都可以获得类似的梯度下降更新方程。

惩罚项

将$-l(\theta)$视为损失函数的话，就可以在上面添加惩罚项(正则化项)防止过拟合了。

用scikit-learn实现

数据说明

使用scikit-learn内建的wine数据集

模型说明

class sklearn.linear_model.LogisticRegression(penalty=’l2’, dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver=’liblinear’, max_iter=100, multi_class=’ovr’, verbose=0, warm_start=False, n_jobs=1)

参数选择

惩罚项

可选l1,l2惩罚

多分类的方法

multi_class : str, {‘ovr’, ‘multinomial’}, default: ‘ovr’

'ovr': 用 1 vs rest 的方法构建多分类其

'multinomial': 基于multinomial假设的softmax回归

求解方法

solver : {‘newton-cg’, ‘lbfgs’, ‘liblinear’, ‘sag’, ‘saga’}, default: ‘liblinear’ Algorithm to use in the optimization problem.

‘multinomial’做多分类问题只能用: ‘newton-cg’, ‘sag’, ‘saga’ , ‘lbfgs’，'ovr'则全都可以用

‘newton-cg’, ‘lbfgs’ and ‘sag’ 只能处理L2惩罚，而、‘liblinear’ and ‘saga’ 可以处理L1惩罚

小细节：用sag或者saga求解时，预处理输入数据把量纲调整一致有利于快速收敛。