Logistic Regression 的理解和实现

汪加林

2016 年 12 月 22 日

1 Introduction

本文主要是介绍 Logistic Regression，同时还讲解了基于 L1 正则项的模型求解方法 Coordinate Descent(坐标下降方法), 基于 L2 正则项的模型求解方法 Gradient Descent 和 newton BFGS 方法。代码是用 python 实现的。

下面是本说明文档的内容组织结构:

- 代码说明
- 逻辑回归
- 模型更新方法
- 实验
- 类的说明

2 代码说明

本文实现了加了不同正则项的 logistic regression，加了平方项正则项和一次范数正则项。并且还实现了梯度下降法、坐标下降法等进行模型参数的更新求解。其中 Lasso 回归只能使用 coordinate descent 方法，Ridge 回归有两种参数求解方法，gradient descent 和 BFGS newton 方法。

3 逻辑回归

前面讲过回归模型目标变量取值范围为整个实数空间，那么在用于分类时，以二分类为例，通过将回归模型中的值再次映射到 [0,1] 区间内，表示类别为 1 的概率，就得

到了逻辑回归模型。采用 sigmoid 函数进行映射：

$$y = \frac{1}{1+e^{-x}}$$

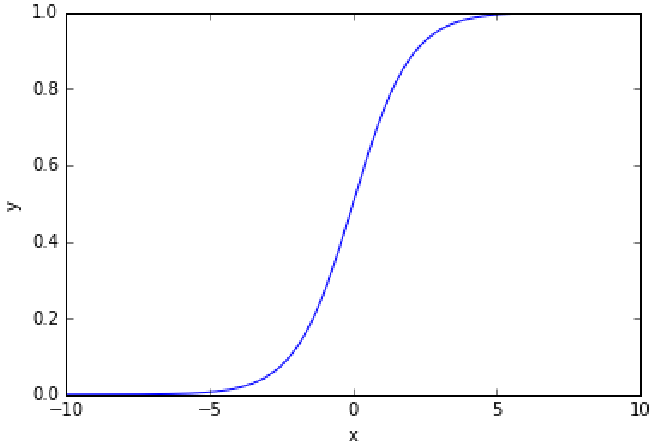

Figure 1: sigmoid 函数

将 x 替换成回归模型中的 $w^T x$, 得：

$$prob(y=1|w,x) = \frac{1}{1+e^{-w^T x}}$$

$$prob(y=0|w,x) = \frac{e^{-w^T x}}{1+e^{-w^T x}}$$

3.1 目标函数

显然，我们想使我们的模型使分类正确的概率越大越好。因此有：

$$Prob = \prod_{i=1}^{N}(prob(y=1|w,x)^{y_i} + prob(y=0|w,x)^{1-y_i})$$

通常，我们会对这种连乘的取对数，将乘法转化为求和。

$$\begin{aligned}
ln(Prob) &= \sum_{i=1}^{N} ln(prob(y=1|w,x_i)^{y_i} + prob(y=0|w,x_i)^{1-y_i}) && (1)\\
&= \sum_{i=1}^{N} [y_i lnprob(y=1|w,x_i) + (1-y_i) lnprob(y=0|w,x_i)] && (2)\\
&= \sum_{i=1}^{N} [y_i ln\frac{1}{1+e^{-w^T x_i}} + (1-y_i) ln\frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}] && (3)\\
&= \sum_{i=1}^{N} [-y_i ln(1+e^{-w^T x_i}) + (1-y_i)(-w^T x_i - ln(1+e^{-w^T x_i})] && (4)\\
&= -\sum_{i=1}^{N} [y_i ln(1+e^{-w^T x_i}) + (1-y_i)(w^T x_i + ln(1+e^{-w^T x_i})] && (5)\\
&= -\sum_{i=1}^{N} [ln(1+e^{-w^T x_i}) + (1-y_i) w^T x_i] && (6)\\
& && (7)
\end{aligned}$$

因此，$max\ ln(Prob) = min\ \ -ln(Prob)$

3.1.1 L1 正则项

$$\begin{aligned}
Obj_{L1} &= -ln(Prob) + C||w|| && (8)\\
&= \sum_{i=1}^{N} [ln(1+e^{-w^T x_i}) + (1-y_i) w^T x_i] + C||W|| && (9)
\end{aligned}$$

3.1.2 L2 正则项

$$\begin{aligned}
Obj_{L2} &= -ln(Prob) + C||w|| && (10)\\
&= \sum_{i=1}^{N} [ln(1+e^{-w^T x_i}) + (1-y_i) w^T x_i] + C||W||^2 && (11)
\end{aligned}$$

现在我们的目标就变成了最小化目标函数 Obj_{L1} 和 Obj_{L2} 了。

4 模型更新

在求解模型的参数时，通常会针对不同的模型和不同的数据集大小采取不同的模型更新方法。在这里，我们将讲解并实现梯度下降法、坐标下降法、牛顿法。

4.1 梯度下降法

首先来看看梯度下降法。梯度下降法的原理大家可以看看这里。梯度下降法的关键就在于求解梯度，对于 Obj_{L2}:

$$\frac{\partial Obj_{L2}}{\partial w_j} = \sum_{i=1}^{N} \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}(-x_{ij}) + 2Cw_j \tag{12}$$

$$= \sum_{i=1}^{N}(1 - \frac{1}{1+e^{-w^T x_i}})(-x_{ij}) + 2Cw_j \tag{13}$$

因此，参数更新方式为；

$$w_j = w_j - \frac{\partial Obj_{L2}}{\partial w_j} \tag{14}$$

$$= w_j - \{\sum_{i=1}^{N}(1 - \frac{1}{1+e^{-w^T x_i}})(-x_{ij}) + 2Cw_j\} \tag{15}$$

$$= (1-2C)w_j - \sum_{i=1}^{N}(1 - \frac{1}{1+e^{-w^T x_i}})(-x_{ij}) \tag{16}$$

4.2 坐标下降法

坐标下降法则是对于存在 L1 正则项的模型进行求解的。依然需要求导，可是不难发现，在求导过程中，我们需要对绝对值进行求导。因此在这里子梯度 (subgradient) 的概念很重要：对于在点 a 处的子梯度 $V \in \partial g(x)$,$g(x)$ 满足：

$$g(b) \geq g(a) + V(b-a)$$

例如，对于 $|x|$,$V \in [-1, 1]$。

$$\frac{\partial Obj_{L1}}{\partial w_j} = \sum_{i=1}^{N} \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}(-x_{ij}) + 2C\frac{\partial |w|}{\partial w_j} \tag{17}$$

$$\tag{18}$$

$$
\frac{\partial Obj_{L1}}{\partial w_j} = \begin{cases} \sum_{i=1}^{N} \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}(-x_{ij}) + 2C, \ w_j > 0 \\ \sum_{i=1}^{N} \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}(-x_{ij}) - 2C, \ w_j < 0 \\ \sum_{i=1}^{N} \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}(-x_{ij}) + 2C * \frac{\partial |w|}{\partial w_j}, \ w_j = 0, \frac{\partial |w|}{\partial w_j} \in [-1, 1] \end{cases} \tag{19}
$$

对于 $w_j = 0$，不妨取，$\frac{\partial |w|}{\partial w_j} = 0$，

$$
\frac{\partial Obj_{L1}}{\partial w_j} = \begin{cases} \sum_{i=1}^{N} \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}(-x_{ij}) + 2C, \ w_j > 0 \\ \sum_{i=1}^{N} \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}(-x_{ij}) - 2C, \ w_j < 0 \\ \sum_{i=1}^{N} \frac{e^{-w^T x_i}}{1+e^{-w^T x_i}}(-x_{ij}), \ w_j = 0 \end{cases} \tag{20}
$$

因此，参数更新方式为；

$$
w_j = w_j - \frac{\partial Obj_{L1}}{\partial w_j}
$$

4.3 牛顿法及 BFGS 算法

最初看 BFGS 是在李航老师的统计学习方法这本书的附录里，大体而言讲的很清楚了。不过在更新海森矩阵的时候的推导有些细节没有讲好。

4.4 牛顿法

$$
f(x) = f(x_k) + g_k^T (x - x_k) + \frac{1}{2}(x - x_k)^T H_(k+1)(x - x_k)
$$

其中，$g_k = \nabla f(x_k)$, $H_{k+1} = [\frac{\partial f(x_k) \partial f(x_k)}{\partial x_{k,i} \partial x_{k,j}}]_{n*n}$

$$
\nabla f(x) = g_k + H_{k+1}(x - x_k)
$$

优化时，假设 x_{k+1} 最优点满足 $\nabla f(x_{k+1}) = 0$，因此有：$0 = g_k + H_{k+1}(x_{k+1} - x_k)$ 得：

$$
x_{k+1} = x_k - H_{k+1}^{-1} g_k
$$

$$
\begin{aligned}
f(x_{k+1}) &= f(x_k) + g_k^T (x_{k+1} - x_k) + \frac{1}{2}(x - x_k)^T H_{k+1}(x - x_k) && (21) \\
&= f(x_k) - g_k^T H_{k+1}^{-1} g_k + \frac{1}{2}(H_k^{-1} g_k)^T H_k (H_{k+1}^{-1} g_k) && (22) \\
&= f(x_k) - \frac{1}{2} g_k^T H_{k+1}^{-1} g_k && (23)
\end{aligned}
$$

当 H_{k+1} 正定时，$f(x_{k+1}) < f(x_k)$。

4.5 BFGS 拟牛顿法

由于需要对 H_k 求逆，这是一个非常耗时的操作，因此人们提出了各种来更新 H_k 的方法。

$$\nabla f(x_{k+1}) = g_k + H_{k+1}(x_{k+1} - x_k) \tag{24}$$

$$g_{k+1} - g_k = H_{k+1}(x_{k+1} - x_k) \tag{25}$$

令 $y_k = g_{k+1} - g_k, \delta_k = x_{k+1} - x_k$，得：

$$y_k = H_{k+1}\delta_k$$

不妨取 $H_{k+1} = H_k + \alpha uu^T + \beta vv^T$, 得:

$$y_k = (H_k + \alpha uu^T + \beta vv^T)\delta_k$$

易得 $\alpha u^T\delta_k$, $\beta v^T\delta_k$ 是两个实数，可以单独提到前面。并且不妨取 $\alpha u^T\delta_k = 1$, $\beta v^T\delta_k = -1$，那么有:

$$y_k = (H_k + u - v)\delta_k \tag{26}$$

$$u - v = y_k - H_k\delta_k \tag{27}$$

又不妨取：$u = y_k, v = H_k\delta_k$，代入之前的公式可得：

$$\alpha = \frac{1}{y_k^T\delta_k} \tag{28}$$

$$\beta = \frac{-1}{\delta_k^T H_k^T \delta_k} \tag{29}$$

代入，即得 H_k 的更新公式：

$$H_{k+1} = H_k + \frac{y_k y_k^T}{y_k^T\delta_k} - \frac{H_k\delta_k\delta_k^T H_k^T}{\delta_k^T H_k^T \delta_k}$$

4.6 梯度下降法和坐标下降法的异同

1. 1. 坐标下降法是梯度下降法的特殊情况

2. 2. 每次方向是不同的

Algorithm 1 BFGS

$B_0 = I, x_0 = [0, .., 0]$

$k = 1$, 计算 g_k,x_{k+1}

while not convergent **do**

更新 $x_{k+1}, x_{k+1} = x_k - H_{k+1}^{-1} g_k$

计算 $g_{k+1}, y_k = g_{k+1} - g_k, \delta_k = x_{k+1} - x_k$

更新 H_{k+1},$H_{k+1} = H_k + \frac{y_k y_k^T}{y_k^T \delta_k} - \frac{H_k \delta_k \delta_k^T H_k^T}{\delta_k^T H_k^T \delta_k}$

$k = k + 1$

end while

4.7 为什么牛顿法比梯度下降法收敛的更快

先摆结论:

梯度下降法 线性收敛，在最优点处容易震荡，难以收敛

牛顿法 平方收敛，在最优点处很易收敛

5 实验

为了方便，实验数据直接从 sklearn 中导出。设置的最大迭代次数为 5000，数据大小 1797*64。运行结果如下所示：

惩罚项	求解方法	运行迭代次数	准确率	运行时间
L1 正则项,Lasso	Coordinate Descent	5000	0.9076	32.51
L2 正则项,Ridge	Gradient Descent	5000	0.8920	0.538
L2 正则项,Ridge	BFGS(Newton)	798	0.90818	0.243

Table 1: 实验比较

从实验结果不难看出，newton 方法确实收敛的更快。

6 类的说明

LogitRegression ：逻辑回归类，定义了不同模型的参数，预测等功能

Update ：主要实现的是梯度下降、坐标下降、牛顿法等实现模型参数的求解的类

7 参考文献

1. BFGS 推导,http://blog.csdn.net/itplus/article/details/21897443
2. 最速下降法, 凸优化,Stephen Broyd