Introduction to Machine Learning
Homework 2

181860066 牛铭杨

2020 年 11 月 16 日

1 [30pts] Multi-Label Logistic Regression

In multi-label problem, each instance $\boldsymbol{x}$ has a label set $\boldsymbol{y} = \{y_1, y_2, ..., y_L\}$ and each label $y_i \in \{0, 1\}, \forall 1 \le i \le L$. Assume the post probability $p(\boldsymbol{y} \mid \boldsymbol{x})$ follows the conditional independence:

$$p(\boldsymbol{y} \mid \boldsymbol{x}) = \prod_{i=1}^{L} p(y_i \mid \boldsymbol{x}). \tag{1.1}$$

Please use the logistic regression method to handle the following questions.
(1) [15pts] Please give the log-likelihood function of your logistic regression model;
给定数据集 $\{\boldsymbol{x_i}, \boldsymbol{y_i}\}_{i=1}^{m}$, 对数似然函数为

$$l(\boldsymbol{w}, b) = \sum_{i=1}^{m} \ln p(\boldsymbol{y}_i|\boldsymbol{x}_i; \boldsymbol{w}, b) = \sum_{j=1}^{L}\sum_{i=1}^{m} \ln p(y_{ij}|\boldsymbol{x}_i; \boldsymbol{w}, b) \tag{1.2}$$

其中 $\boldsymbol{y}_{ij}$ 是第 i 个样本在第 j 个标签上的属性
所以采用书上的记号，只需最小化

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{m}(-\sum_{j=1}^{L} y_{ij}\boldsymbol{\beta}^T\widehat{\boldsymbol{x}}_i + L\ln(1 + e^{\boldsymbol{\beta}^T\widehat{\boldsymbol{x}}_i})) \tag{1.3}$$

(2) [15pts] Please calculate the gradient of your log-likelihood function and show the parameters updating step using gradient descent.
似然函数的梯度为

$$\nabla l(\boldsymbol{\beta}) = \sum_{i=1}^{m} \widehat{\boldsymbol{x}}_i(-\sum_{j=1}^{L} y_{ij} + \frac{Le^{\boldsymbol{\beta}^T\widehat{\boldsymbol{x}}_i}}{1+e^{\boldsymbol{\beta}^T\widehat{\boldsymbol{x}}_i}}) \tag{1.4}$$

第 $t+1$ 轮迭代解的更新公式为

$$\boldsymbol{\beta}^{t+1} = \boldsymbol{\beta}^t - \gamma\nabla l(\boldsymbol{\beta}) \quad = \boldsymbol{\beta}^t - \gamma\sum_{i=1}^{m} \widehat{\boldsymbol{x}}_i(-\sum_{j=1}^{L} y_{ij} + \frac{Le^{\boldsymbol{\beta}^T\widehat{\boldsymbol{x}}_i}}{1+e^{\boldsymbol{\beta}^T\widehat{\boldsymbol{x}}_i}}) \tag{1.5}$$

2 [70pts] Logistic Regression from scratch

2.1 实现细节

我将这次实验分为 3 个阶段，导入训练数据，梯度下降/上升法训练出 $\boldsymbol{w} = (\boldsymbol{w}; b)$，根据测试集预测结果。

导入数据时使用两个矩阵 X, Y 来存储输入的样本和样本的标签，其中 X 每个行向量（样本）都加上一列，其值为 1，以便计算。

然后使用 OvR，训练 26 个分类器，在训练每个分类器时，作 Y 的深拷贝，并将当前识别的那类标注为正例，其他均为反例。

使用梯度下降/上升法训练出每个分类器的 $\boldsymbol{w}$，测试时识别为概率最高的那一类。测试时，可以统计每个类别的混淆矩阵，之后总计即可算出查全率和查准率以及 $F1$。

梯度下降/上升法每一步沿着梯度的方向前进一个步长的距离，经过多次迭代就可以近似最值。我们要最大化

$$l(\boldsymbol{w}) = \sum_{i=1}^{m}(y_i\boldsymbol{w}^T\widehat{\boldsymbol{x}}_i - \ln(1+e^{\boldsymbol{w}^T\widehat{\boldsymbol{x}}_i})) \tag{2.1}$$

对其求偏导，得到

$$\begin{aligned}
\frac{\partial l(\boldsymbol{w})}{\partial w_j} &= \sum_{i=1}^{m}(y_i x_{ij} - x_{ij}\frac{e^{\boldsymbol{w}^T\widehat{\boldsymbol{x}}_i}}{1+e^{\boldsymbol{w}^T\widehat{\boldsymbol{x}}_i}}) \\
&= \sum_{i=1}^{m} x_{ij}(y_i - \frac{e^{\boldsymbol{w}^T\widehat{\boldsymbol{x}}_i}}{1+e^{\boldsymbol{w}^T\widehat{\boldsymbol{x}}_i}}) \\
&= \sum_{i=1}^{m} x_{ij}(y_i - Sigmoid(\boldsymbol{w}^T\widehat{\boldsymbol{x}}_i))
\end{aligned} \tag{2.2}$$

所以有

$$\nabla l(\boldsymbol{w}) = X^T(Y - Sigmoid(X\boldsymbol{w})) \tag{2.3}$$

而第 $t+1$ 轮迭代解的更新公式为

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \gamma\nabla l(\boldsymbol{w}) = \boldsymbol{w}_t + \gamma X^T(Y - Sigmoid(X\boldsymbol{w})) \tag{2.4}$$

其中 γ 为步长，上面 (**??**) 即为我的实现参照的等式

2.2 优化与参数设置

实验的步长参数 γ 和循环次数 $loops$ 是可以调节的，我通过调节这两个参数来使实验效果更好。

一开始我设置 $\gamma = 0.001$ $loops = 1000$，效果并不好，准确率只能达到百分四十几，然后我意识到可能并不需要循环这么多次，就已经收敛了，步长太大每次都越过了最值点。所以我根据循环次数来调节步长，经过不断实践，我发现，$loops < 400$ 时，$\gamma = 0.001$，$loops < 480$ 时，$\gamma = 0.0001$，$loops < 500$ 时，$\gamma = 0.00001$，这样是较好的选择，准确率可以达到百分之六十几，而仅仅增加循环次数效果并不是很理想。

下面是实现的结果。

表 1: Performance on test set.

Performance Metric	Value (%)
accuracy	66.95
micro Precision	65.22
micro Recall	66.15
micro F_1	65.68
macro Precision	65.23
macro Recall	68.55
macro F_1	66.85