Softmax Regression 应用——手写数字识别

1. Softmax Regression 介绍

在 softmax回归中，我们解决的是多分类问题（相对于 logistic 回归解决的二分类问题），类标 y 可以取 k 个不同的值（而不是 2 个）。因此，对于训练集 $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$，我们有 $y^{(i)} \in \{1, 2, \ldots, k\}$。（注意此处的类别下标从 1 开始，而不是 0）。例如，在 MNIST 数字识别任务中，我们有 $k = 10$ 个不同的类别。

对于给定的测试输入 x，我们想用假设函数针对每一个类别估算出概率值 $p(y = j|x)$。也就是说，我们想估计 x 的每一种分类结果出现的概率。因此，我们的假设函数将要输出一个 k 维的向量（向量元素的和为1）来表示这 k 个估计的概率值。 具体地说，我们的假设函数 $h_\theta(x)$ 形式如下：

$$h_\theta(x^{(i)}) = \begin{bmatrix} p(y^{(i)} = 1|x^{(i)}; \theta) \\ p(y^{(i)} = 2|x^{(i)}; \theta) \\ \vdots \\ p(y^{(i)} = k|x^{(i)}; \theta) \end{bmatrix} = \frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}} \begin{bmatrix} e^{\theta_1^T x^{(i)}} \\ e^{\theta_2^T x^{(i)}} \\ \vdots \\ e^{\theta_k^T x^{(i)}} \end{bmatrix}$$

其中 $\theta_1, \theta_2, \ldots, \theta_k \in \Re^{n+1}$ 是模型的参数。请注意 $\frac{1}{\sum_{j=1}^{k} e^{\theta_j^T x^{(i)}}}$这一项对概率分布进行归一化，使得所有概率之和为 1 。

为了方便起见，我们同样使用符号 θ 来表示全部的模型参数。在实现Softmax回归时，将 θ 用一个 $k \times (n+1)$ 的矩阵来表示会很方便，该矩阵是将 $\theta_1, \theta_2, \ldots, \theta_k$ 按行罗列起来得到的，如下所示：

$$\theta = \begin{bmatrix} —\theta_1^T— \\ —\theta_2^T— \\ \vdots \\ —\theta_k^T— \end{bmatrix}$$

2. Softmax Regression 的损失函数

$$J(\theta) = -\frac{1}{m} \left[\sum_{i=1}^{m} \sum_{j=1}^{k} 1 \left\{ y^{(i)} = j \right\} \log \frac{e^{\theta_j^T x^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T x^{(i)}}} \right]$$

通过取 log 可以把 softmax regression 的最大似然估计的惩罚转化为加法，但是 softmax 存在一个问题，就是存在求值冗余，存在多组最优解，理由如下:

在Softmax回归中存在着参数冗余的问题。简单来讲就是参数中有些参数是没有任何用的，为了证明这点，假设从参数向量θ_j中减去向量ψ，假设函数为：

$$\begin{aligned} p\left(y^{(i)} = j \mid \mathbf{x}^{(i)}; \theta\right) &= \frac{e^{(\theta_j - \psi)^T \mathbf{x}^{(i)}}}{\sum_{l=1}^{k} e^{(\theta_l - \psi)^T \mathbf{x}^{(i)}}} \\ &= \frac{e^{\theta_j^T \mathbf{x}^{(i)}} \cdot e^{-\psi^T \mathbf{x}^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T \mathbf{x}^{(i)}} \cdot e^{-\psi^T \mathbf{x}^{(i)}}} \\ &= \frac{e^{\theta_j^T \mathbf{x}^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T \mathbf{x}^{(i)}}} \end{aligned}$$

从上面可以看出从参数向量θ_j中减去向量ψ对预测结果并没有任何的影响，也就是说在模型中，存在着多组的最优解。

因此要么就是把某一个类别作为基准，即 theta 为 0，或者加上惩罚项：

$$\frac{\lambda}{2} \sum_{i=1}^{k} \sum_{j=0}^{n} \theta_{ij}^2$$

可以使得在函数求解的时候取到最优解，

本文采用惩罚项即上述的 L2 正则约束，所以最后损失函数如下：

$$J(\theta) = -\frac{1}{m}\left[\sum_{i=1}^{m}\sum_{j=1}^{k} I\left\{y^{(i)}=j\right\} log\frac{e^{\theta_j^T \mathbf{x}^{(i)}}}{\sum_{l=1}^{k} e^{\theta_l^T \mathbf{x}^{(i)}}}\right] + \frac{\lambda}{2}\sum_{i=1}^{k}\sum_{j=0}^{n}\theta_{ij}^2$$

3. Softmax Regression 求导

求损失函数的最小值，需要对参数求导：

$$\nabla_{\theta_j} J(\theta) = -\frac{1}{m}\sum_{i=1}^{m}\left[x^{(i)}(1\{y^{(i)}=j\} - p(y^{(i)}=j|x^{(i)};\theta))\right] + \lambda\theta_j$$

根据上述公式，编写代码，代码见附件。

4. MNIST 数据集介绍

MNIST 数据集来自美国国家标准与技术研究所, National Institute of Standards and Technology (NIST). 训练集 (training set) 由来自 250 个不同人手写的数字构成，其中 50% 是高中学生, 50% 来自人口普查局 (the Census Bureau) 的工作人员. 测试集(test set) 也是同样比例的手写数字数据.

MNIST 数据集可在 http://yann.lecun.com/exdb/mnist/ 获取，它包含了四个部分:

Training set images: train-images-idx3-ubyte.gz (9.9 MB，解压后 47 MB，包含 60,000 个样本)

Training set labels: train-labels-idx1-ubyte.gz (29 KB，解压后 60 KB，包含 60,000 个标签)

Test set images: t10k-images-idx3-ubyte.gz (1.6 MB，解压后 7.8 MB，包含 10,000 个样本)

Test set labels: t10k-labels-idx1-ubyte.gz (5KB，解压后 10 KB，包含 10,000 个标签)

5. 备注

1）代码中对像素点进行了处理，因为手写数字主要是 0-1 是否有像素的区别，而不是像素值的问题。

2）Softmax 与上课说的 multinominal regression 其实是一模一样的，在 R 里面是用采用第一种做法，把某类作为参考，即 theta 定义为 0（我之前是用 R 写的，但是实在不知道为什么和 R 包装好的程序最优值不一样，最后改成了 python，用 python 的 softmax 做了一遍），softmax 采用了 L2 正则约束。

3）最后 MNIST 的测试集准确率：91.75%

具体迭代情况：

```
损失函数 : 0.327688793225  错误率 : 0.08165
opt_solution:        fun: 0.32768879322510652
 hess_inv: <7840x7840 LbfgsInvHessProduct with dtype=float64>
      jac: array([  1.10376133e-07,  -2.78936728e-08,  -1.16377286e-07, ...,
        -1.02875820e-07,   6.42885200e-08,   2.21852812e-08])
  message: b'CONVERGENCE: NORM_OF_PROJECTED_GRADIENT_<=_PGTOL'
     nfev: 166
      nit: 155
   status: 0
  success: True
        x: array([  1.10376133e-04,  -2.78936728e-05,  -1.16377286e-04, ...,
        -1.02875820e-04,   6.42885200e-05,   2.21852812e-05])
测试集准确率 : 0.9175
```

在训练集上迭代了 155 次后，函数收敛到最小值，损失函数值最后为 0.32768 错误率为 8.165%。

在测试集上，测试集准确率：91.75%。x 表示 theta 优化解。

参考教程：http://ufldl.stanford.edu/wiki/index.php/Softmax%E5%9B%9E%E5%BD%92

https://github.com/siddharth-agrawal/Softmax-Regression/blob/master/softmaxRegression.py

https://github.com/sudk1896/Softmax/blob/master/softmax.py