实验1. 度量学习实验报告

MF1733037，刘鑫鑫，liuxx@lamda.nju.edu.cn

2017 年 10 月 31 日

综述

度量学习用于学习一个度量样本间距离的距离函数。以马氏距离为例:

$$dist^2_{mah}(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i - \boldsymbol{x}_j)^T \boldsymbol{M} (\boldsymbol{x}_i - \boldsymbol{x}_j) \tag{1}$$

其中$\boldsymbol{x}_i, \boldsymbol{x}_j$是两个样本，$dist_{mah}(\boldsymbol{x}_i, \boldsymbol{x}_j)$则是这两个样本间的距离。对马氏距离的度量学习实际上就是学习矩阵$\boldsymbol{M}$。对$\boldsymbol{M}$的学习往往可以将其嵌入到某些分类器中。如近邻成分分析(Neighbourhood Component Analysis,NCA)就是将其嵌入到近邻分类器中。

任务1

度量函数学习目标

在任务1中使用近邻成分分析法。其优化目标为:

$$P = \arg\min_P 1 - \sum_{i=1}^{m} \sum_{j\in\Omega_i} \frac{exp(-dist^2_{mah}(\boldsymbol{x}_i, \boldsymbol{x}_j))}{\sum_l exp(-dist^2_{mah}(\boldsymbol{x}_i, \boldsymbol{x}_l))} \tag{2}$$

其中Ω_i表示和样本$\boldsymbol{x}_i$标签一致的样本集合。考虑到矩阵$\boldsymbol{M}$是半正定对称矩阵，即存在矩阵$\boldsymbol{P}$使得$\boldsymbol{M} = \boldsymbol{P}^T\boldsymbol{P}$，于是优化目标可以写为：

$$P = \arg\min_P 1 - \sum_{i=1}^{m} \sum_{j\in\Omega_i} \frac{exp(-\|\boldsymbol{P}^T\boldsymbol{x}_i - \boldsymbol{P}^T\boldsymbol{x}_j\|_2^2)}{\sum_l exp(-\|\boldsymbol{P}^T\boldsymbol{x}_i - \boldsymbol{P}^T\boldsymbol{x}_l\|_2^2)} \tag{3}$$

优化算法

对于式（3）所示的优化目标，可以采用随机梯度下降法。

设数据集为$D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$。即对每个样本$\boldsymbol{x}_i$，设其标签为$y_i$。那么对于样本$\boldsymbol{x}_i$，我们要去最小化如下函数：

$$f(\boldsymbol{x}_i) = -\sum_{j\in\Omega_i} \frac{exp(-\|\boldsymbol{P}^T\boldsymbol{x}_i - \boldsymbol{P}^T\boldsymbol{x}_j\|_2^2)}{\sum_l exp(-\|\boldsymbol{P}^T\boldsymbol{x}_i - \boldsymbol{P}^T\boldsymbol{x}_l\|_2^2)} \tag{4}$$

该函数是可导的，求出其对参数矩阵$\boldsymbol{P}$的导函数为：

$$\frac{\partial f}{\partial \boldsymbol{P}} = -2\boldsymbol{P}(p_i \sum_k p_{ik}\boldsymbol{x}_{ik}\boldsymbol{x}_{ik}^T - \sum_{j\in\Omega_i} p_{ij}\boldsymbol{x}_{ij}\boldsymbol{x}_{ij}^T) \tag{5}$$

其中：

$$p_{ij} = \frac{exp(-\|\boldsymbol{P}^T\boldsymbol{x}_i - \boldsymbol{P}^T\boldsymbol{x}_j\|_2^2)}{\sum_l exp(-\|\boldsymbol{P}^T\boldsymbol{x}_i - \boldsymbol{P}^T\boldsymbol{x}_l\|_2^2)}$$

$$p_i = \sum_{j\in\Omega_i} p_{ij}$$

$$\boldsymbol{x}_{ij} = \boldsymbol{x}_i - \boldsymbol{x}_j$$

接下来，就可以应用随机梯度下降法来优化NCA的目标式（3）了：

Algorithm 1 SGD for NCA

Input:

DataSet $D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_n, y_n)\}$

learning rate for SGD: lr

maximum iteration for SGD:max_iter

Output: Matrix $\boldsymbol{P}$

```
initialize P
initialize max_iter
for iter ∈ (1 : max_iter) do
    i = iter% n
    calculate ∂f/∂P
    update P: P ← P − lr ∗ ∂f/∂P
end for
```

具体代码见附件中myDML.py。其中myDML.train()函数用于训练样本，myDML.distance()函数用于计算样本间距离。使用时注意要先训练样本再调用距离函数，否则会出现错误。主要的矩阵运算均调用numpy包里的相关函数。

对于任务1数据集，笔者推荐的超参数为$lr = 0.01, max_iter = 20000$。

任务2

在该任务中，笔者仍然使用任务1中的NCA方法去训练样本并计算距离，优化方法也是SGD。在设定合适的超参数后，该方法错误率低于欧几里得距离的错误率约$6\% - 8\%$。具体测试结果见附件evaluation.txt

对于任务2数据集，笔者推荐的超参数为$lr = 0.005, max_iter = 182000$