高级机器学习
作业一

2018 年 12 月 14 日

1 [25pts] Multi-Class Logistic Regression

教材的章节 3.3 介绍了对数几率回归解决二分类问题的具体做法。假定现在的任务不再是二分类问题，而是多分类问题，其中 $y \in \{1, 2 \ldots, K\}$。请将对数几率回归算法拓展到该多分类问题。

(1) **[15pts]** 给出该对率回归模型的“对数似然”(log-likelihood);

(2) **[5pts]** 计算出该“对数似然”的梯度。

提示 1：假设该多分类问题满足如下 $K-1$ 个对数几率，

$$\begin{aligned}
\ln \frac{p(y=1|\mathbf{x})}{p(y=K|\mathbf{x})} &= \mathbf{w}_1^{\mathrm{T}}\mathbf{x} + b_1 \\
\ln \frac{p(y=2|\mathbf{x})}{p(y=K|\mathbf{x})} &= \mathbf{w}_2^{\mathrm{T}}\mathbf{x} + b_2 \\
&\cdots \\
\ln \frac{p(y=K-1|\mathbf{x})}{p(y=K|\mathbf{x})} &= \mathbf{w}_{K-1}^{\mathrm{T}}\mathbf{x} + b_{K-1} \frac{\partial l(\beta)}{\partial \beta_i} = -\sum_{j=0}^{m} (\hat{x}(\P(y_i = K) - 1) - p_i(\hat{x}; \beta))
\end{aligned}$$

提示 2：定义指示函数 $\mathbb{I}(\cdot)$，

$$\mathbb{I}(y=j) = \begin{cases} 1 & \text{若 } y \text{ 等于 } j \\ 0 & \text{若 } y \text{ 不等于 } j \end{cases}$$

Solution. 此处用于写解答 *(中英文均可)*

(1) 解：

由提示 *1* 中多项式可推出 $p(y = i|x)$:

因为

$$\ln \frac{p(y=i|\mathbf{x})}{p(y=K|\mathbf{x})} = \mathbf{w}_i^{\mathrm{T}}\mathbf{x} + b_i (i \neq K)$$

所以

$$p(y=i|\mathbf{x}) = p(y=K|\mathbf{x}) e^{\mathbf{w}_i^T \mathbf{x} + b_i} (i \neq K) \tag{1.1}$$

又因为

$$\sum_{i=1}^{K} p(y=i|\mathbf{x}) = \sum_{i=1}^{K-1} p(y=K|\mathbf{x}) e^{\mathbf{w}_i^T \mathbf{x}+b_i} + p(y=K|\mathbf{x}) = 1$$

由上式可得

$$p(y=K|\mathbf{x}) = \frac{1}{1+\sum_{i=1}^{K-1} e^{\mathbf{w}_i^T \mathbf{x}+b_i}} \tag{1.2}$$

将式 *(1.2)* 代入 *(1.1)* 可得

$$p(y=i|\mathbf{x}) = \frac{e^{\mathbf{w}_i^T \mathbf{x}+b_i}}{1+\sum_{i=1}^{K-1} e^{\mathbf{w}_i^T \mathbf{x}+b_i}}$$

推导得到 $p(y=i|x)$，然后对样本数据 $\{\{x_i, y_i\}\}_1^m$ 构造似然函数

$$l(\mathbf{w}, \mathbf{b}) = \sum_{i=1}^{m} \ln p(y=y_i|\mathbf{x}_i; \mathbf{w}, b) \tag{1.3}$$

令 $\beta = (\mathbf{w}; \mathbf{b}), \hat{\mathbf{x}} = (\mathbf{x}; 1), p_i(\hat{\mathbf{x}}|\beta) = p(y=i|\hat{\mathbf{x}}; \beta), p(y=y_i|\mathbf{x}_i; \mathbf{w}, b)$ 可重写为

$$\begin{aligned} p(y_i|\mathbf{x}_i; \beta) &= \sum_{j=1}^{K-1} \mathbb{I}(y_i=j) p_i(\hat{\mathbf{x}}_i; \beta) + \mathbb{I}(y_i=K) p_K(\hat{\mathbf{x}}_i; \beta) \\ &= \frac{\sum_{j=1}^{K-1} \mathbb{I}(y_i=j) e^{\beta_j^T \hat{\mathbf{x}}_i} + \mathbb{I}(y_i=K)}{1+\sum_{j=1}^{K-1} e^{\beta_j^T \hat{\mathbf{x}}_i}} \end{aligned}$$

所以

$$\ln p(y_i|x_i; \beta) = \ln(\sum_{j=1}^{K-1} (\mathbb{I}(y_i=j) e^{\beta_j^T \hat{\mathbf{x}}_i}) + \mathbb{I}(y_i=K)) - \ln(1+\sum_{j=1}^{K-1} e^{\beta_j^T \hat{\mathbf{x}}_i})$$

又因为 $\ln(\sum_{j=1}^{K-1} (\mathbb{I}(y_i=j) e^{\beta_j^T \hat{\mathbf{x}}_i}) + \mathbb{I}(y_i=K))$ 可化简为

$$\begin{cases} \beta_{y_i}^T \hat{\mathbf{x}}_i & y_i \neq K \\ 0 & y_i = K \end{cases}$$

因此原似然函数 *(1.3)* 可重写为：

$$\begin{cases} l(\beta) = \beta_{y_i}^T \hat{\mathbf{x}}_i - \ln(1+\sum_{j=1}^{K-1} e^{\beta_j^T \hat{\mathbf{x}}_i}) & y_i \neq K \\ l(\beta) = -\ln(1+\sum_{j=1}^{K-1} e^{\beta_j^T \hat{\mathbf{x}}_i}) & y_i = K \end{cases}$$

(2) 解：

求偏导得到“对数似然”的梯度：

$$\begin{aligned} \frac{\partial l(\beta)}{\partial \beta_t} &= \sum_{i=1}^{m} (\mathbb{I}(y_i=t) \hat{\mathbf{x}}_i - \frac{\hat{\mathbf{x}}_i e^{\beta_t^T \hat{\mathbf{x}}_i}}{1+\sum_{j=1}^{K-1} \beta_j^T \hat{\mathbf{x}}_i}) \\ &= \sum_{i=1}^{m} ((\mathbb{I}(y_i=t) - p(y_i=t|\hat{\mathbf{x}}_i; \beta)) \hat{\mathbf{x}}_i) \quad t \in [1, K-1] \end{aligned}$$

2 [15pts] Semi-Supervised Learning

我们希望使用半监督学习的方法来对文本文档进行分类。假设我们使用二进制指示符的词袋模型描述各个文档，在这里，我们的词库有 10000 个单词，因此每个文档由长度为 10000 的二进制向量表示。

对于以下提出的分类器，说明其是否可以用于改进学习性能并提供简要说明。

1. **[5pts]** 使用 EM 的朴素贝叶斯；
2. **[5pts]** 使用协同训练的朴素贝叶斯；
3. **[5pts]** 使用基于特征选择的朴素贝叶斯；

Solution. *此处用于写解答 (中英文均可)*

1. *使用 EM 的朴素贝叶斯可以改进学习性能，因为样本的一些特征可能因为某些因素而不可观测或者在训练数据中没有体现，对于这种隐变量，就可以使用 EM 算法来对它进行学习，从而提升学习的性能。*
2. *使用协同训练的朴素贝叶斯可以改进学习性能，从训练数据的两个不同视图去学习数据的特征，能学习到更多的数据相关特征，从而分类也能更精准。*
3. *使用基于特征选择的朴素贝叶斯可以改进学习性能，特征选择可以从数据中选择出重要的特征，去除“无关特征”与“冗余特征”后可以大大降低算法学习的难度，提升学习的性能。*

3 [60pts] Dimensionality Reduction

请实现三种降维方法：PCA，SVD 和 ISOMAP，并在降维后的空间上用 1-NN 方法分类。

1. 数据：我们给出了两个数据集，都是二分类的数据。可以从`https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`找到，同时也可以在提交作业的目录文件夹中找名为“two datasets”的压缩文件下载使用。每个数据集都由训练集和测试集组成。

2. 格式：再每个数据集中，每一行表示一个带标记的样例，即每行最后一列表示对应样例的标记，其余列表示对应样例的特征。

具体任务描述如下：

1. **[20pts]** 请实现 PCA 完成降维（方法可在参考书`http://www.charuaggarwal.net/Data-Mining.htm` 中 Section 2.4.3.1 中找到）

   首先，仅使用训练数据学习投影矩阵；

   其次，用学得投影矩阵将训练数据与测试数据投影到 k-维空间 $(k = 10, 20, 30)$；

   最后，在降维后空间上用 1-NN 预测降维后 k 维数据对应的标记 $(k = 10, 20, 30)$，并汇报准确率。注意，测试数据集中的真实标记仅用来计算准确率。

2. **[20pts]** 请实现 SVD 完成降维（方法在上述参考书 Section 2.4.3.2 中找到）

   首先，仅使用训练数据学习投影矩阵；

   其次，用学得投影矩阵将训练数据与测试数据投影到 k-维空间 $(k = 10, 20, 30)$；

   最后，在降维后空间上用 1-NN 预测降维后 k 维数据对应的标记 $(k = 10, 20, 30)$，并汇报准确率。注意，测试数据集中的真实标记仅用来计算准确率。

3. **[20pts]** 请实现 ISOMAP 完成降维（方法在参考书 Section 3.2.1.7 中找到）

   首先，使用训练数据与测试数据学习投影矩阵。在这一步中，请用 4-NN 来构建权重图。（请注意此处 4 仅仅是用来举例的，可以使用其他 k-NN, $k \geq 4$ 并给出你选择的 k。如果发现构建的权重图不连通，请查找可以解决该问题的方法并汇报你使用的方法）

   其次，用学得投影矩阵将训练数据与测试数据投影到 k-维空间 $(k = 10, 20, 30)$。

   最后，在降维后空间上用 1-NN 预测降维后 k 维数据对应的标记 $(k = 10, 20, 30)$，并汇报准确率。注意，测试数据集中的真实标记仅用来计算准确率。

可以使用已有的工具、库、函数等直接计算特征值和特征向量，执行矩阵的 SVD 分解，计算 graph 上两个节点之间的最短路。PCA/SVD/ISOMAP 和 1-NN 中的其他步骤必须由自己实现。

报告中需要包含三个方法的伪代码和最终的结果。最终结果请以表格形式呈现，表中包含三种方法在两个数据集中，不同 $k = 10, 20, 30$ 下的准确率。

Solution. *此处用于写解答 (中英文均可)*

1. 解：中心化，未标准化

算法 1 PCA

输入: $traindataMat$ 训练数据矩阵，$traintagMat$ 训练数据标签矩阵, $testdataMat$ 测试数据矩阵, $testtagMat$ 测试数据标签矩阵, $topK$ 降维的维度

输出: 准确率

$import\ numpy\ as\ np$
$meanVals \leftarrow np.mean(traindataMat,\ axis = 0)$
$centralizedMat \leftarrow (traindataMat\ -\ manVals)$
$convMat \leftarrow np.conv(centralizedMat, rowvar = 0)$
$eigVals,\ eigVects \leftarrow np.linalg.eig(conMat)$
$eigValsIdx\_topK \leftarrow argsort(eigVals)[: -(topK + 1) : -1]$
$projectionMat \leftarrow eigVects[:,\ eigValsIdx\_topK]$
$traindataMat\_proj \leftarrow traindataMat\ *\ projectionMat$
$testdataMat\_proj \leftarrow testdataMat\ *\ projectionMat$
return $1NN(traindataMat\_proj,\ testdataMat\_proj,\ traintagMat,\ testtagMat)$

2. 解：未中心化，未标准化

算法 2 SVD

输入: $traindataMat$ 训练数据矩阵，$traintagMat$ 训练数据标签矩阵, $testdataMat$ 测试数据矩阵, $testtagMat$ 测试数据标签矩阵, $topK$ 降维的维度

输出: 准确率

$import\ numpy\ as\ np$
$\_,\ S,\ VT \leftarrow np.linalg.svd(traindataMat)$
$SIdx\_topK \leftarrow argsort(S)[: -(topK + 1) : -1]$
$VT\_dimK \leftarrow VT[SIdx\_topK, :]$
$projectionMat \leftarrow VT\_dimK.T$
$traindataMat\_proj \leftarrow traindataMat\ *\ projectionMat$
$testdataMat\_proj \leftarrow testdataMat\ *\ projectionMat$
return $1NN(traindataMat\_proj,\ testdataMat\_proj,\ traintagMat,\ testtagMat)$

3. 解：未中心化，未标准化

算法 3 ISOMAP

输入: $traindataMat$ 训练数据矩阵，$traintagMat$ 训练数据标签矩阵, $testdataMat$ 测试数据矩阵, $testtagMat$ 测试数据标签矩阵, $topK$ 降维的维度, KNN_K K 近邻参数

输出: 准确率

$import\ numpy\ as\ np$
$dataMat \leftarrow np.vstack(traindataMat,\ testdataMat)$
$dataSize \leftarrow len(dataMat)$
$Euc\_distanceMat \leftarrow calc\_distance(dataMat)$
// 建立 KNN 的图
$knn\_distanceMat \leftarrow np.ones([dataSize,\ dataSize],\ float32) * inf$
for $i = 0 \rightarrow dataSize - 1$ **do**
    $knnIdx \leftarrow np.argpartition(Euc\_distanceMat[i],\ KNN\_K)[: KNN\_K + 1]$
    $knn\_distanceMat[i][knnIdx] \leftarrow Euc\_distanceMat[i][knnIdx]$
    for $j\ in\ knnIdx$ **do**
        $knn\_distanceMat[j][i] \leftarrow knn\_distanceMat[i][j]$
    end for
end for
$adjacencyTable \leftarrow build\_adjancecy\_table(knn\_distanceMat)$
$dist \leftarrow np.ones([dataSize,\ dataSize],\ float32) * inf$
for $i = 0 \rightarrow dataSize - 1$ **do**
    $dist[i][i] = 0.0$
    $dijkstra(dist,\ adjacencyTable,\ i)$
end for
$data\_dimK \leftarrow mds(dist,\ topK)$
return $1NN(data\_dimK,\ trantagMat,\ testtagMat)$

最终结果：

Accuracy *Dim* / *Method*	$k = 10$	$k = 20$	$k = 30$	*Dataset*
PCA	0.5825242718446602	0.5631067961165048	0.5631067961165048	*sonar*
PCA	0.7581609195402299	0.7627586206896552	0.735632183908046	*splice*
SVD	0.5922330097087378	0.5825242718446602	0.5631067961165048	*sonar*
SVD	0.7586206896551724	0.7641379310344828	0.7480459770114942	*splice*
$ISOMAP(6NN)$	0.4174757281553398	0.4174757281553398	0.4368932038834951	*sonar*
$ISOMAP(4NN)$	0.6809195402298851	0.6901149425287356	0.6919540229885057	*splice*