模式识别与机器学习 作业1 报告

数据集设置

数据集中共三种高斯分布，标签分别标为0,1,2。分别以$0.1, 0.5, 0.4$的概率从这三种分布中采样构成数据集。

有：

$$\mathcal{N}_1 = \mathcal{N}(0, 1) \tag{1}$$
$$\mathcal{N}_2 = \mathcal{N}(1, 1) \tag{2}$$
$$\mathcal{N}_2 = \mathcal{N}(-1, 1) \tag{3}$$

然后：

$$n = 1000 \quad \text{表示采样的数据量} \tag{4}$$
$$P_{\text{select}} = \{0 : 0.4, 1 : 0.5, 2 : 0.1\} \quad \text{表示不同标签的先验分布} \tag{5}$$
$$D = \{(\mathbf{x_i}, y_i) | x_{ij} \sim \mathcal{N}_{y_i}, y_i \sim \mathcal{P}_{select}, 0 \le i < n\} \quad \text{是采样得到的数据} \tag{6}$$

其中$\mathbf{x_i}$是d维向量。这里取$d = 8$。

数据集被预先按$0.1/0.1/0.8$的比例划分成验证集、测试集和训练集。

模型

判别式模型

这是一个线性模型，即模型为$f(x) = Wx + b$。为了公式的简洁，可以预先在x加入一个常量维度，然后就可以把bias放到W中。因此有：

$$f(\mathbf{x}) = W\mathbf{x}$$

其中$\mathbf{x} \in R^{d \times 1}$是$d$维列向量，$W \in R^{m \times d}$是模型参数。
注意这里的$\mathbf{x}$是添加了常量维度之后的，因此这里的d比数据集中的d大1。另外$m = 3$是标签种类数量。

$$p(y|\mathbf{x}) = \text{softmax}(W\mathbf{x})$$

其中

$$\text{softmax}(\mathbf{x})_i = \frac{e^{x_i}}{\sum_j e^{x_j}}$$

训练

使用交叉熵损失函数，采用随机梯度法(SG)训练：
1. 采样一个样本$(\mathbf{x}, y)$
2. 算出损失函数$L = -\log p(y|\mathbf{x})$
3. 算出梯度估计值$\mathbf{g} = \frac{\partial L}{\partial W}$
4. 更新参数$W \leftarrow W - \alpha\mathbf{g}$。其中$\alpha$是学习率。

接下来考虑具体如何算$\mathbf{g}$：
设

$$\hat{\mathbf{y}} = f(\mathbf{x}) = W\mathbf{x}$$
$$\mathbf{p} = \text{softmax}(\hat{\mathbf{y}})_i = \frac{e^{\hat{y_i}}}{\sum_j e^{\hat{y_j}}}$$

那么有：

$$\mathbf{g} = \frac{\partial L}{\partial W} \tag{7}$$
$$= \frac{\partial L}{\partial \hat{\mathbf{y}}} \frac{\partial \hat{\mathbf{y}}}{\partial W} \tag{8}$$

为了防止混淆，记标签$y = \mathrm{k}$，就有：

$$\frac{\partial L}{\partial \hat{\mathbf{y}}_i} = \frac{\partial - \log\left(\frac{e^{\hat{y_k}}}{\sum_j e^{\hat{y_j}}}\right)}{\partial \hat{y}_i} \tag{9}$$
$$= \frac{\partial \log\left(\sum_j e^{\hat{y_j}}\right) - \hat{y_k}}{\partial \hat{y_i}} \tag{10}$$
$$= \frac{e^{\hat{y_i}}}{\sum_j e^{\hat{y_j}}} - [k = i] \tag{11}$$
$$= p_i - [k = i] \tag{12}$$

其中$[\ ]$表示艾佛森记号。$[\mathrm{True}] = 1$，$[\mathrm{False}] = 0$。
所以就有：

$$\frac{\partial L}{\partial \hat{\mathbf{y}}} = \mathbf{p} - \mathbf{1}_k$$

其中$\mathbf{1}_k$表示一个只有第k个分量为1，其他分量为0的向量。
又：

$$\frac{\partial \hat{\mathbf{y}}}{\partial W} = \frac{\partial W\hat{\mathbf{x}}}{\partial W} = \hat{\mathbf{x}}^T$$

所以：

$$g = \frac{\partial L}{\partial \hat{\mathbf{y}}} \frac{\partial \hat{\mathbf{y}}}{\partial W} = (\mathbf{p} - \mathbf{1}_k)\hat{\mathbf{x}}^T$$

接下来的事情就很简单了。

生成式模型

生成式模型中直接训练求$p(\mathbf{x}, y)$，而预测时输出为

$$y = \arg\max_z p(\mathbf{x}|z) = \arg\max_z \frac{p(\mathbf{x}, z)}{p(z)}$$

模型和之前一样还是个线性模型，最后的输出为：

$$p(\mathbf{x}, z) = \mathrm{sigmoid}(W^{(z)}\mathbf{x})$$

表示模型对标签为z的把握程度。 其中$W^{(z)} \in R^{1\times d}, x \in R^{d\times 1}$，

$$\mathrm{simoid}(x) = \frac{e^x}{1 + e^x}$$

注意这里实际上有3个模型。第z个模型负责预测$p(\mathbf{x}, z)$。

训练

在训练时，对$z = 0, 1, 2$都训练一次。为了清晰，接下来还是记$k = y$。
对$z = k$，把输出往1的方向训练，目标函数为$L = -\log(p(\mathbf{x}, z))$。
对$z \neq k$，把输出往0的方向训练，目标函数为$L = -\log(1 - p(\mathbf{x}, z))$。

对于求梯度。还是设$\hat{y} = W^{(z)}\mathbf{x}$。依然有：

$$\frac{\partial L}{\partial W^{(z)}} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial W^{(z)}}$$

而

$$\frac{\partial L}{\partial \hat{y}} = \frac{\partial \log(1+e^{\hat{y}}) - [z=k]\hat{y}}{\partial \hat{y}} \tag{13}$$

$$= \frac{e^{\hat{y}}}{1+e^{\hat{y}}} - [z=k] \tag{14}$$

$$= p(\mathbf{x}, z) - [z=k] \tag{15}$$

又：

$$\frac{\partial \hat{y}}{\partial W^{(z)}} = \frac{\partial W^{(z)}\mathbf{x}}{W^{(z)}} = \mathbf{x}^T$$

所以第z个模型的梯度就是：

$$g^{(z)} = \frac{\partial L}{\partial W^{(z)}} = \frac{\partial L}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial W^{(z)}} = (p(\mathbf{x}, z) - [z=k])\mathbf{x}^T$$

实验设置

1. 学习率$\alpha = 10^{-2}$
2. 训练步数$s = 10000$。每一步从训练集中随机挑选一个样本进行更新。
3. 数据集大小为1000，验证集/测试集/训练集比例为$0.1/0.1/0.8$。

结果

模型	Dev Acc	Test Acc
判别式	91%	88%
生成式	88%	86%

讨论：SG method vs Batch method

还有另一种常用的优化策略，通常称为batch方法。

之前用的SG方法中，因为每一步只使用一个样本来算梯度。这样得到的实际上是对训练集梯度的一个估计。虽然这是一个无偏估计，但是方差比较大，相当于会在估计梯度时引入一些噪声。因此可能损害优化效果。

batch方法中每一步用上所有的数据样本，把它们的梯度取平均，因此可以把方差减少n倍，使梯度估计更准确。
但是batch方法的问题在于因为每一步都要在整个训练集上计算梯度，所以十分慢，虽然其可以更好地利用并行化。

因为batch method太慢了，我只测试了判别式模型在两种优化方法下的效果：

模型	优化方法	用时	Dev Acc	Test Acc
判别式	SG	21.18s	91%	88%
判别式	Batch	1187.15s	90%	88%

可以看到Batch方法并没有表现出明显的优势。
这个结果有可能是因为学习率不是很合适。不过Batch方法十分慢所以非常难调参。

实践上或许还是SG方法更合适。

讨论：更难的数据

默认的数据中，三个高斯分布的均值都不同，所以模型基本上可以直接通过求均值来大概区分。
接下来我尝试把两个高斯分布的均值设置为一样的，仅方差有区别。具体来说：

$$\mathcal{N}_1 = \mathcal{N}(0, 1) \tag{16}$$

$$\mathcal{N}_2 = \mathcal{N}(0, 0.1) \tag{17}$$

$$\mathcal{N}_2 = \mathcal{N}(1, 1) \tag{18}$$

称这个数据集为D2，原来的为D1。

结果：

模型	数据集	Dev Acc	Test Acc
判别式	D1	91%	88%
生成式	D1	88%	86%
判别式	D2	81%	65%
生成式	D2	78%	65%

可见这个模型确实不太有能力分辨方法。我个人认为这是因为模型能力比较小。方差是二阶特征，仅一个线性模变换应该不太能很好地处理方差。

代码使用

```
python source.py
```

因为数据文件比较大（127KB），我就没有传到github上了。调用source.py会自动生成数据。
模型会在log.log中生成详细的日志文件，在report.log中生成一个简短的结果报告。