多分类问题

科研方法与论文写作2024

范雯怡

南京邮电大学 计算机学院、软件学院、网络空间安全学院

汇报提纲

- 应用场景
- 问题定义
- 模型假设
- Softmax函数优缺
- Softmax与Logistic
- 学习准则
- 优化算法
- 代码实例

应用场景——多分类问题

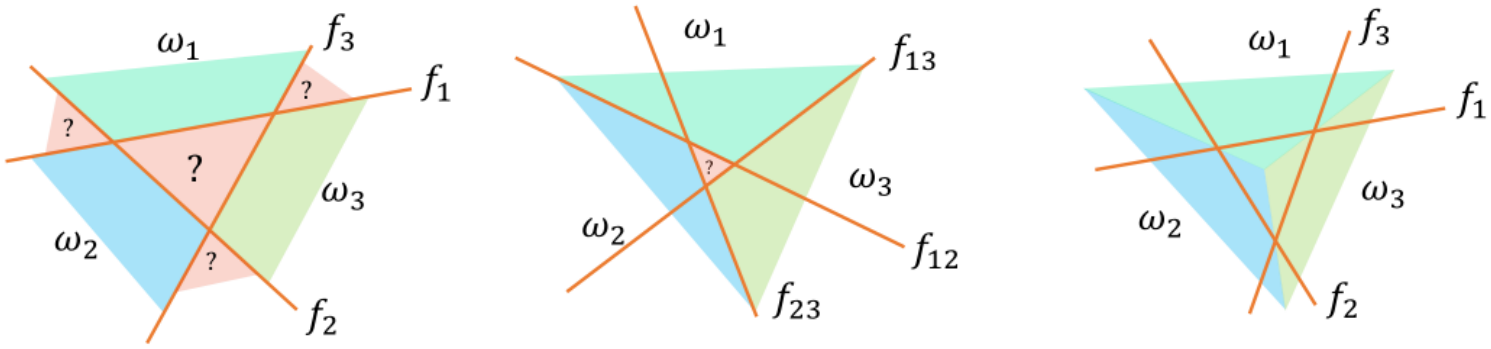
- 某个电子邮件属于垃圾邮件文件夹、工作文件夹还是资讯文件夹?
- 某个声音是儿童、成人还是老人?
- 某个图像描绘的是驴、狗、猫、还是鸡?
- 某个电影属于科幻片还是古装片还是现代片?

问题定义

- 给定一个包含N个训练样本的训练集 $\{(x^{(n)}, y^{(n)})\}_{n=1}^{N}$ ，其中 $x \epsilon \mathbb{R}^D$ ，$y \epsilon \{1, 2, ..., C\}$ ，C是一个常数。

  也就是需要对这些样本x进行分类，这些样本分别属于不同的C个类别

- 多分类问题有3种常见的设计判别函数的方式，而Softmax回归采用第三种方式



(a) “一对其余”方式　(b) “一对一”方式　(c) “argmax”方式

问题定义

- Softmax回归基于argmax方式，需要区分C个类别就使用C个判别函数

“*argmax*”方式：这是一种改进的“一对其余”方式，共需要C个判别函数

$$f_c(\mathbf{x};\mathbf{w}_c)=\mathbf{w}_c^{\mathrm{T}}\mathbf{x}+b_c,\qquad c=[1,\cdots,C] \tag{3.10}$$

如果存在类别c,对于所有的其他类别$\tilde{c}(\tilde{c}\neq c)$都满足$f_c(\mathbf{x};\mathbf{w}_c)>f_{\tilde{c}}(\mathbf{x},\mathbf{w}_{\tilde{c}})$，那么$\mathbf{x}$属于类别$c$。即

$$y=\arg\max_{c=1}^{C} f_c(\mathbf{x};\mathbf{w}_c). \tag{3.11}$$

模型假设

- 首先假设有C个线性模型 $f_c(x;w_c)=w_c^T x$ $c\epsilon\{1,...,C\}$
  即C个判别函数
  这样一个样本x作为输入，可以得到C个结果。
  其中w_c是第c类的权重向量

- 将分类问题看作条件概率估计问题

  但是判别函数值域为R，需要建模成条件概率的形式 $p_\vartheta(y=c|x)$
  即给定x后y=c的条件概率，满足值域在[0，1]之间，并且c的所有取值对应的条件概率和为1.
  所以现在要把一个实数区间的值转换成一个概率分布就依靠Softmax函数

- Softmax函数

  对于K个标量$x_1,...,x_K$，转换成值域[0,1],和为1的值

$$Soft\max(x_k)=\frac{\exp(x_k)}{\sum_{i=1}^{K}\exp(x_i)}$$

  使用Softmax函数就可以将K个标量转换成具有K个取值的分布

模型假设

- 综上，给定一个样本$\boldsymbol{x}$， Softmax回归预测的属于类别c的条件概率为：

$$p(y=c|\boldsymbol{x}) = \text{softmax}(\boldsymbol{w}_c^\top \boldsymbol{x}) = \frac{\exp(\boldsymbol{w}_c^\top \boldsymbol{x})}{\sum_{c'=1}^{C} \exp(\boldsymbol{w}_{c'}^\top \boldsymbol{x})}$$

因为用Softmax函数构建了类别的条件概率，所以这个模型称为Softmax回归

- 还可以写成向量形式

$$\hat{\boldsymbol{y}} = \text{softmax}(\boldsymbol{W}^\top \boldsymbol{x}) = \frac{\exp(\boldsymbol{W}^\top \boldsymbol{x})}{1_C^\top \exp(\boldsymbol{W}^\top \boldsymbol{x})}$$

$W = [w_1, ..., w_C]$

1_C为C维的全一向量

这里左乘全一向量相当于在做求和操作

这个$\hat{y} \in \mathbb{R}^C$,是所有类别的预测条件概率组成的向量，第c维的值是第c类的预测条件概率

模型假设

在上面基础上决策函数可以表示为：

$$\hat{y} = \arg\max_{c=1}^{C} p(y = c|\boldsymbol{x})$$
$$= \arg\max_{c=1}^{C} \boldsymbol{w}_c^{\mathrm{T}} \boldsymbol{x}.$$

选择拥有最大条件概率的类别，作为样本x的最终预测类别。

结果是一个标量，表示样本x对应的类别索引。

Softmax函数优缺

使用Softmax函数完成对判别函数的值域约束，使得结果能以概率的形式出现具有一定的好处：

1. 函数中引入了指数函数，因为指数函数曲线呈现递增趋势，最重要的是斜率逐渐增大，所以在x轴上一个很小的变化，可以导致y轴上很大的变化。这种函数曲线能够将输出的数值拉开距离。
2. 而且在深度学习中通常使用反向传播求解梯度进而使用梯度下降进行参数更新的过程，而指数函数在求导的时候比较方便。

同时也带来了缺点：

因为使用了指数函数，所以若值非常大的话，计算得到的数值也会变的非常大，数值可能会溢出。

有两种解决方法：

1. 所有输入送入Softmax函数之前，先减去所有输入中最大的元素
2. 在经验风险函数中添加正则化项，并不会影响最终结果，而且可以避免数值溢出和过拟合

Softmax与logistic

如果用Softmax回归解决二分类问题，与Logistic回归有什么区别?

1. 判别函数个数。Logistic回归用1个，Softmax回归用2个
2. 概率化时选择的函数不同，一个用Logistic函数，一个用Softmax函数
3. 决策函数有不同。

Softmax的决策函数:

$$\begin{aligned}\hat{y} &= \arg\max_{y\in\{0,1\}} \boldsymbol{w}_y^{\mathsf{T}}\boldsymbol{x} \\ &= I\left(\boldsymbol{w}_1^{\mathsf{T}}\boldsymbol{x} - \boldsymbol{w}_0^{\mathsf{T}}\boldsymbol{x} > 0\right) \\ &= I\left((\boldsymbol{w}_1 - \boldsymbol{w}_0)^{\mathsf{T}}\boldsymbol{x} > 0\right)\end{aligned}$$

Logistic的决策函数:

$$\hat{y} = I(w^T x > 0)$$

Softmax与logistic

解决多分类问题用多个Logistic回归好，还是用Softmax回归？

- 需要判断类别之间是否有互斥

  例如，如果有四个类别的音乐，分别为：古典音乐、乡村音乐、摇滚乐和爵士乐，那么可以假设每个训练样本只会被打上一个标签（即：一首歌只能属于这四种音乐类型的其中一种），此时使用Softmax 回归比较好。

  如果四个类别如下：人声音乐、舞曲、影视原声、流行歌曲，那么这些类别之间并不是互斥的。例如：一首歌曲可以来源于影视原声，同时也包含人声 。这种情况下，使用 4 个二分类的 Logistic 回归分类器更为合适。这样，对于每个新的音乐作品 ，我们的算法可以分别判断它属于哪些类别。

学习准则

因为要学习，所以先要构建损失函数。这里建立参数化的条件概率和真实条件概率之间的交叉熵，公式如下：

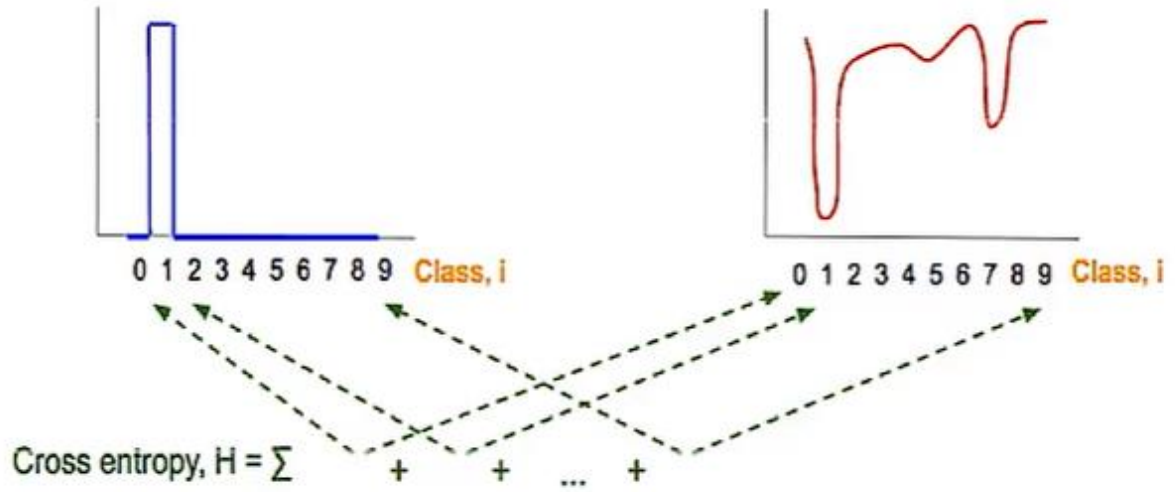
针对某个样本x的交叉熵损失函数：

$$-\sum_{c=1}^{C} y_c^{(n)} \log \hat{y}_c^{(n)}$$

其中$y_c^{(n)}$表示样本x属于类别c的真实概率

其中$\hat{y}_c^{(n)}$表示样本x预测到属于类别c的概率

即存在一个真实概率分布和一个预测概率的负对数分布，将他们对应相乘后再相加而得



学习准则

以某一个样本的标签和预测结果为例，演示交叉熵损失函数的具体计算公式和结果：

- 对于一个三类分类问题，类别为[0,0,1]，预测类别概率为[0.3,0.3,0.4]，则

$$\begin{aligned}\mathcal{L}(\theta) &= -\big(0 \times \log(0.3) + 0 \times \log(0.3) + 1 \times \log(0.4)\big) \\ &= -\log(0.4).\end{aligned}$$

学习准则

- 为了方便，下面会借助C维的one-hot向量$\boldsymbol{y} \in \{0,1\}^C$来表示类别标签．如果标签属于类别$c$，标签y向量表示为：

$$\boldsymbol{y} = [I(1=c), I(2=c), \cdots, I(C=c)]^{\mathrm{T}}$$

- 采用交叉熵损失函数，Softmax回归模型的经验风险函数为：

$$\mathcal{R}(\boldsymbol{W}) = -\frac{1}{N}\sum_{n=1}^{N}\sum_{c=1}^{C}\boldsymbol{y}_c^{(n)}\log\hat{\boldsymbol{y}}_c^{(n)}$$

$$= -\frac{1}{N}\sum_{n=1}^{N}(\boldsymbol{y}^{(n)})^{\mathrm{T}}\log\hat{\boldsymbol{y}}^{(n)},$$

其中$\hat{\boldsymbol{y}}^{(n)} = \mathrm{softmax}(\boldsymbol{W}^{\mathrm{T}}\boldsymbol{x}^{(n)})$为样本$\boldsymbol{x}^{(n)}$在每个类别的后验概率

$y^{(n)}$是样本$x^{(n)}$对应的$onehot$类别标签

优化算法

风险函数有了之后就可以使用优化算法进行参数学习，这里主要采用梯度下降

首先计算得到风险函数$\mathcal{R}(\boldsymbol{W})$关于$\boldsymbol{W}$的梯度为：

$$\frac{\partial \mathcal{R}(\boldsymbol{W})}{\partial \boldsymbol{W}} = -\frac{1}{N}\sum_{n=1}^{N} \boldsymbol{x}^{(n)} \left(\boldsymbol{y}^{(n)} - \hat{\boldsymbol{y}}^{(n)}\right)^{\mathrm{T}}$$

梯度公式的推导过程见下页

$y^{(n)}$是样本$x^{(n)}$对应的$onehot$类别标签

$\hat{y}^{(n)}$是样本$x^{(n)}$在每个类别预测的条件概率

优化算法

先计算每个样本对应的损失函数关于参数W的梯度，稍作更改后就可以得到上述风险函数关于W的梯度值

前提假设：① $\hat{y}=\mathrm{softmax}(z)$ 那么 $\frac{\partial \hat{y}}{\partial z}=\mathrm{diag}(\hat{y})-\hat{y}\hat{y}^T$

② $z=W^Tx=[W_1^Tx, W_2^Tx, \cdots, W_C^Tx]^T$ 那么 $\frac{\partial z}{\partial W_c}=\left[\frac{\partial W_1^Tx}{\partial W_c}, \frac{\partial W_2^Tx}{\partial W_c}, \cdots, \frac{\partial W_C^Tx}{\partial W_c}\right]$

$=[0,0,\cdots,x,\cdots,0]$

$\triangleq \mathbb{M}_c(x)$ 是个第c列为x，其余为0的矩阵

$\frac{\partial z^{(n)}}{\partial W_c}=\mathbb{M}_c(x^{(n)})$ $\frac{\partial \hat{y}^{(n)}}{\partial z^{(n)}}=\mathrm{diag}(\hat{y}^{(n)})-\hat{y}^{(n)}\hat{y}^{(n)T}$

$\therefore \frac{\partial \mathcal{L}^{(n)}(W)}{\partial W_c}=-\frac{\partial (y^{(n)T}\log \hat{y}^{(n)})}{\partial W_c}$

$=-\frac{\partial z^{(n)}}{\partial W_c}\frac{\partial \hat{y}^{(n)}}{\partial z^{(n)}}\frac{\partial \log \hat{y}^{(n)}}{\partial \hat{y}^{(n)}}y^{(n)}$ 上下同乘 $\partial z^{(n)}$、$\partial \hat{y}^{(n)}$ ∵ $y^{(n)}$ 是标签与 W_c 无关，∴可以直接从∂中提出来

$=-\mathbb{M}_c(x^{(n)})(\mathrm{diag}(\hat{y}^{(n)})-\hat{y}^{(n)}(\hat{y}^{(n)})^T)(\mathrm{diag}(\hat{y}^{(n)}))^{-1}y^{(n)}$

$=-\mathbb{M}_c(x^{(n)})(I-\hat{y}^{(n)}(\hat{y}^{(n)})^T(\mathrm{diag}(\hat{y}^{(n)}))^{-1})y^{(n)}$ ↓ ∵ $y^T\mathrm{diag}(y)^{-1}=1_c^T$ 全1行向量

$=-\mathbb{M}_c(x^{(n)})(I-\hat{y}^{(n)}1_c^T)y^{(n)}$

$=-\mathbb{M}_c(x^{(n)})(y^{(n)}-\hat{y}^{(n)}1_c^Ty^{(n)})$ ↓ ∵ $y^{(n)}$ 是one-hot列向量 ∴ $1_c^Ty^{(n)}=1$

$=-\mathbb{M}_c(x^{(n)})(y^{(n)}-\hat{y}^{(n)})$

$=-x^{(n)}[y^{(n)}-\hat{y}^{(n)}]_c$

因此 $\frac{\partial \mathcal{L}^{(n)}(W)}{\partial W}=-x^{(n)}(y^{(n)}-\hat{y}^{(n)})^T$

优化算法

采用梯度下降算法时，Softmax回归的训练过程为，先初始化W_0，然后用下面的式子进行迭代更新：

$$\boldsymbol{W}_{t+1} \leftarrow \boldsymbol{W}_t + \alpha\left(\frac{1}{N}\sum_{n=1}^{N}\boldsymbol{x}^{(n)}\left(\boldsymbol{y}^{(n)} - \hat{\boldsymbol{y}}_{\boldsymbol{W}_t}^{(n)}\right)^{\mathrm{T}}\right)$$

其中α为学习率，$\hat{y}_{W_t}^{(n)}$是参数为W_t时，$Soft\max$回归模型的输出

代码实例

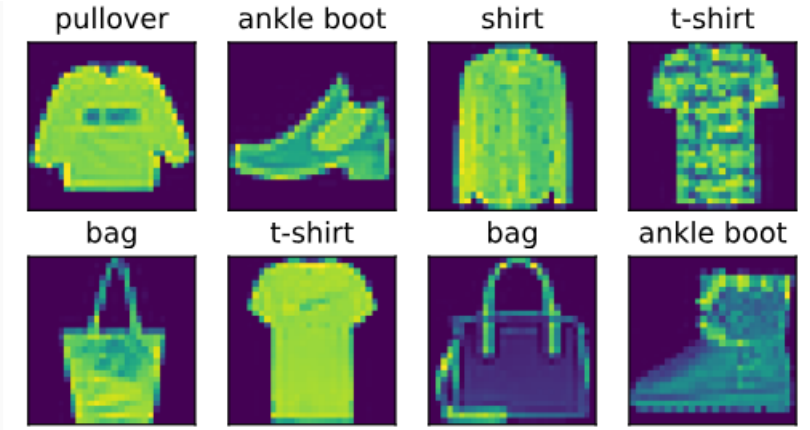
利用Fashion-MNIST服装分类数据集，训练一个多分类模型，数据集包含的10个类别，分别为t-shirt、trouser、pullover、dress、coat、sandal、shirt、sneaker、bag和ankle boot

```
# 读取训练集和测试集
batch_size = 256
train_iter, test_iter = d2l.load_data_fashion_mnist(batch_size)
```

```
# softmax函数
def softmax(X):
    X_exp = torch.exp(X)
    partition = X_exp.sum(1, keepdim=True)  # 矩阵的每一行求和，重新生成新的矩阵
    return X_exp / partition  # 结果中每一行代表一个样本，行中的每个数据代表在该类别的概率，
```

```
# 定义模型，
W = torch.normal(0, 0.01, size=(num_inputs, num_outputs), requires_grad=True)
b = torch.zeros(num_outputs, requires_grad=True)
def net(X):
    return softmax(torch.matmul(X.reshape((-1, W.shape[0])), W) + b)
```

```
# 交叉熵损失函数
def cross_entropy(y_hat, y):
    return - torch.log(y_hat[range(len(y_hat)), y])
```



代码实例

```python
# 优化算法
def updater(batch_size):
    return d2l.sgd([W, b], lr, batch_size)
```

```python
# 计算分类精度:正确预测数量与总预测数量之比，作为评价标准
"""..."""
def accuracy(y_hat, y):
    """计算预测正确的数量"""
    if len(y_hat.shape) > 1 and y_hat.shape[1] > 1:
        y_hat = y_hat.argmax(axis=1)  #   每一行中元素最大的那个下标存在
    cmp = y_hat.type(y.dtype) == y  # #将y_hat转换为y的数据类型然后作比
    return float(cmp.type(y.dtype).sum())
```

训练好后，对一些样本的预测结果如下：