如何写一个简单的最大熵分类器

余南 *

黑龙江大学

2016年5月

*电子邮件: 20124826@s.hlju.edu.cn

分类器能将待分类实体划归于某一类。它能完成各种各样的分类任务，比如预测天气，识别数字等等。本文主要讲如何DIY一个简单的最大熵分类器对句子进行情感分类。关于句子的情感分类例子，见表1

句子	情感分类结果
发动机 比较 平顺	正向
方向盘 又 大 又 重	负向

Table 1: 句子情感分类表

最大熵分类器中有一些参数来进行分类操作，它可以只有一组训练参数，用一层函数完成分类。y=Wx 就是这样的一层 函数，其中 W是训练参数，需要进行训练；x是从每个句子中抽取的特征；y是计算得到的分类结果。比如这个y=Wx对表 1的两个句子进行分类，从分类结果y中得到了以下的概率分布，见表2。参数训练的目的是要使得式1趋于0， 那么为了达成这个目的，需要不断去调整更新W。

$$Ł = -\log p_{11} - \log p_{22} \tag{1}$$

句子	正向概率	负向概率
发动机 比较 平顺	$p_{11} = 0.4$	$p_{12} = 0.6$
方向盘 又 大 又 重	$p_{21} = 0.3$	$p_{22} = 0.7$

Table 2: 分类概率分布表

在上述的y=Wx函数中x，y都是向量，我们来看看文本是如何一步步转化成向量形式的。首先，这些句子和其对应的 情感都会写入两个文件中，一个是训练集，另一个是测试集。训练集用于训练参数，测试集用于测试参数性能。训练集和测 试集的例子见表3。表中的句子标签也就是上文的情感分类结果。

句子标签	句子
positive	发动机 比较 平顺
negative	方向盘 又 大 又 重

Table 3: 训练集和测试集的文件格式

用vector<Instance >train 和 vector <Instance >test来存训练集和测试集。单个句子的存储Instance如下：

```
class Instance // 存储一个句子信息的类
{
string label; // 句子标签，也就是句子的分类结果。
```

vector<string >words; // 句子的分词。

};

下一步应抽取句子的特征。句子的特征可以是句子的长度，句子包含的字词等等一切和句子有关的东西。如果简单的抽 取句子特征，可以把句子分词中所有出现过的字词作为特征进行抽取，特别注意句子特征可重复出现，见表4。其中每个 特征前加入“uni=”以此来区分使用了何种特征抽取方式（即特征模板）。

句子	句子特征
发动机 比较 平顺	uni=发动机，uni=比较，uni=平顺
方向盘 又 大 又 重	uni=方向盘，uni=又，uni=大，uni=又，uni=重

Table 4: 句子的特征

在训练参数之前需要解决Instance类无法直接参与计算的问题。于是需要一套编码规则把句子的特征和标签转化成可 以进行计算的格式。把训练集的所有句子特征写入一个特征库中，特征库中的特征各不相同。以上文两个句子为例，其形成 的特征库为F=uni=发动机，uni=比较，uni=平顺，uni=方向盘，uni=又，uni=大，uni=重，对于句子每个特征以其 在特征库中出现的次数进行编码。但为了解决特征的稀疏问题（大量的0出现在编码中），对每个句子特征以在特征库的下 标进行编码。为了顺利进行后续的矩阵乘法，标签以ONE-HOT 方式编码。这套编码规则的存储:

class Alphabet

{

map<string,int>features; // 所有特征和其出现顺序，用其对句子特征进行编码。

vector<string>labels; // 所有类型的标签。用其对句子标签进行ONE-HOT编码。

}

关于句子特征和标签的编码例子见表5和表6。

句子特征	理论上的编码	解决稀疏问题的编码
uni=发动机，uni=比较, uni=平顺	（1，1，1，0，0，0，0）	0,1,2
uni=方向盘，uni=又，uni=大，uni=又，uni=重	（0，0，0，1，2，1，1）	3，4，4，5，6

Table 5: 句子特征编码

句子标签	句子标签的编码
positive	(1,0)
negative	(0,1)

Table 6: 句子标签的编码

vector<Example>train ， vector<Example>test分别用来存储以上编码。单个Example如下：

class Example

{

vector<int>dLabel; // 句子标签编码 这是一个货真价实的向量

vector<int>dFeatures; // 句子特征编码 这是向量为了解决稀疏问题的编码方式

};

完成文本到向量的转化后便可进行参数训练。最大熵分类器的参数训练过程由多次迭代构成，一次迭代是将所有的训练 集的句子特征编码x逐个和当前的W计算分类结果，然后进行W的参数更新。一次y=Wx 更新W的伪代码见算法1

另外有四个值得注意的地方。

其一，训练参数中的y=Wx做的是矩阵乘法，而伪代码中将会看到是加法。这和前文提到的句子稀疏特征在程序中的实 际编码格式有关。比如表1 第二行的句子特征编码如果进行矩阵乘法的话，如式2。可以看出这个矩阵乘法是可以写作 W对应列上的加法。

$$\begin{pmatrix} 0 & 0 & 0 & 1 & 2 & 1 & 1 \end{pmatrix} \begin{pmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \\ W_{31} & W_{32} \\ W_{41} & W_{42} \\ W_{51} & W_{52} \\ W_{61} & W_{62} \\ W_{71} & W_{72} \end{pmatrix} = \begin{pmatrix} W_{41} + 2W_{51} + W_{61} + W_{71} \\ W_{42} + 2W_{52} + W_{62} + W_{72} \end{pmatrix}^T \tag{2}$$

其二，关于y的概率化，令式1的计算结果为(a,b)，其概率化后的结果见式3。

$$\begin{pmatrix} \frac{e^a}{e^a+e^b} & \frac{e^b}{e^a+e^b} \end{pmatrix} \tag{3}$$

Input: $label$,$x[0:xSize]$,$W[0:fSize][0:lSize]$,$EPS = 1e-6$,$ALPA = 0.01$,
$eg2W[x[j]] = 0$,$sqrtEg2w = 0$

Output: $W[0:fSize][0:lSize]$

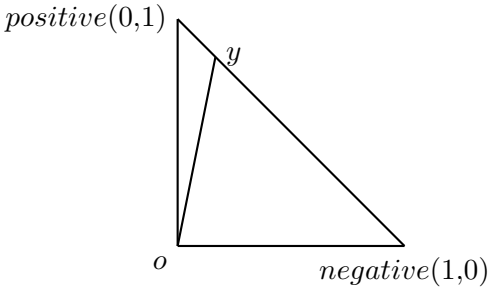
for $i : xSize$ **do**
  for $j : fSize$ **do**
    $s = W[x[j]]$;
  end
  $y = softMax(s)$;
  $ly = y - label$;
  for $j : fSize$ **do**
    $gradW[x[j]] += ly$;
  end
  for $j : fSize$ **do**
    $eg2W[x[j]] += gradW[x[j]] * gradW[x[j]]$;
    $sqrtEg2w = sqrt(eg2W[x[j]] + EPS)$;
    $W[x[j]] =$
    $(W[x[j]] * sqrtEg2W - gradW[x[j]] * ALPA)/(ALPA + sqrtEg2W)$;
  end
  $gradW = 0$;
end

Algorithm 1: 更新训练参数W

不过为了防止溢出，在每一个计算结果向量的分量都会先减去其最大的分量再进行概率化见式4，公式中的 max，max = a>b ？ a ： b;计算结果在2维以上的以此类推。

$$\left(\frac{e^{a-max}}{e^{a-max}+e^{b-max}} \quad \frac{e^{b-max}}{e^{a-max}+e^{b-max}} \right) \tag{4}$$

其三，简要提一下关于分类结果y的损失ly。为了更加直观地看出ly，见坐标图。图中的$\overrightarrow{y, negative}$向量 是y的损失ly，这里实际上是一个“错误驱动”的方式来更新训练参数W，为了使得之后的训练参数W得出的分类结果y更加地接近正确答案。



其四，关于计算结果向量如何转化为程序的分类结果。令式2的计算结果为（a，b）。分类结果见表7。

计算结果的大小	分类结果
a >b	positive
a <b	negative

Table 7: 计算结果-分类结果表

以下内容和实现程序没有什么关系，也就是说下面内容就算看不明白也不影响上文的编程实现。题目中提到最大熵分类 器，这里将说明一下程序是如何使用最大熵的原理。交叉熵的公式见式5，yi 是si概率化后的结果，见式6。

$$H(\xi) = -\sum_{i=1}^{n} a_i \log y_i \tag{5}$$

$$y_i = \frac{e^i}{\sum\limits_{j=1}^{n} e^{s_j}} \tag{6}$$

首先将式6代入式5得到式7。

$$H(\xi) = -\sum_{i=1}^{n} a_i (s_i - \log \sum_{j=1}^{n} e^{s_j}) \tag{7}$$

整理得到式8

$$H(\xi) = \sum_{i=1}^{n} a_i \log \sum_{j=1}^{n} e_j - \sum_{i=1}^{n} a_i s_i \tag{8}$$

由于式9，这里的a向量就是正确答案的编码。

$$\sum_{i=1}^{n} a_i = 1 \tag{9}$$

所以式8 可以继续整理得式10

$$H(\xi) = \log \sum_{i=1}^{n} e_i - \sum_{i=1}^{n} a_i s_i \tag{10}$$

对其求偏导，见式11

$$\frac{\partial H}{\partial s_i} = \frac{e^{s_i}}{\sum\limits_{i=1}^{n} e^{s_i}} - a_i \tag{11}$$

式6代入式11，得式12

$$\frac{\partial H}{\partial s_i} = y_i - a_i \tag{12}$$

其核心思想就是梯度下降法。