最大熵与对数线性模型（用于分类问题）讲课要点

李正华

2015 年 12 月 6 日

目录

1 从最大熵原理（maximum entropy）到对数线性的形式（log-linear）

主要参考http://www.cs.cmu.edu/afs/cs/user/aberger/www/html/tutorial/tutorial.html，建议看到 outline 小节即可。同时也阅读了李航老师的《机器学习方法》相关部分。

我觉得这两个参考文献，在用拉格朗日乘子求解时，都有一些小的瑕疵。所以再写一遍（还有个问题没有完全理解）。

1.1 符号定义

$\mathcal{D} = \{x^j, y^j\}_{j=1}^N$：表示一个数据集，包含 N 个实例 x^j 和对应的 N 个人工标注类别标记 y^j。

$\mathcal{T}$：表示类别集合，$y^j \in \mathcal{T}$。

$\tilde{p}(x) = \frac{\texttt{Count}(x,\mathcal{D})}{N}$：表示实例 x 在数据集 $\mathcal{D}$ 出现的概率，满足 $\sum_{x\in\mathcal{D}} \tilde{p}(x) = 1$

$\tilde{p}(x,y) = \frac{\texttt{Count}(x,y,\mathcal{D})}{N}$：表示实例 x 及对应答案 y 在数据集 $\mathcal{D}$ 出现的概率，满足 $\sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y) = 1$

$\tilde{p}(x)$ 和 $\tilde{p}(x,y)$ 一般称为经验概率，即从数据中通过数数直接可以得到的概率。这两个概率符号的引入，主要是为了简化下面的推导。

1.2 最大熵

在数据集 $\mathcal{D}$ 上，对一个模型（概率分布）$p(y|x)$ 的熵的定义为：

$$\begin{aligned} H(p,\mathcal{D}) &= -\frac{1}{N} \times \sum_{j=1}^{N} \sum_{y\in\mathcal{T}} p(y|x^j) \log p(y|x^j) \\ &= -\sum_{\substack{x\in\mathcal{D} \\ y\in\mathcal{T}}} \tilde{p}(x) p(y|x) \log p(y|x) \end{aligned} \tag{1}$$

接下来，我们希望找一个模型（概率分布）$p(y|x)$，使得 $H(p,\mathcal{D})$ 最大，即：（注意 $p(y|x)$ 的参数形式我们目前并不知道，接下来通过推导会得到其参数形式其实就是 Log-linear 模型）。

$$p^* = \arg\max_{p} H(p,\mathcal{D}) \tag{2}$$

然而，这个最优化问题还有几个约束需要满足。第一个约束是每一个特征 $f_k(.)$ 的 empirical count 需要等于 model expectation（为什么需要这个约束需要大家思考），即：

$$\begin{aligned} \sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y) f_k(x,y) &= \sum_{x\in\mathcal{D}} \left(\tilde{p}(x) \times \sum_{y\in\mathcal{T}} p(y|x) f_k(x,y) \right) \\ &= \sum_{\substack{x\in\mathcal{D} \\ y\in\mathcal{T}}} \tilde{p}(x) p(y|x) f_k(x,y) \end{aligned} \tag{3}$$

第二个约束是对于 $\forall x \in \mathcal{D}$，条件概率之和为 1：

$$1 = \sum_{y\in\mathcal{T}} p(y|x) \tag{4}$$

第三个约束是对于 $\forall x \in \mathcal{D}, y \in \mathcal{T}$，$p(y|x)$ 是一个概率：

$$1 \le p(y|x) \le 0 \tag{5}$$

1.3 求解最大熵模型

利用拉格朗日乘数法，转化为无约束的最优化问题：

$$\begin{aligned}\mathcal{E}(p,\lambda_k,\delta_x,\mathcal{D}) = &-\sum_{\substack{x\in\mathcal{D}\\y\in\mathcal{T}}}\tilde{p}(x)p(y|x)\log p(y|x)\\ &+\sum_k\left(\lambda_k\times\left(\sum_{\substack{x\in\mathcal{D}\\y\in\mathcal{T}}}\tilde{p}(x)p(y|x)f_k(x,y)-\sum_{(x,y)\in\mathcal{D}}\tilde{p}(x,y)f_k(x,y)\right)\right)\\ &+\sum_{x\in\mathcal{D}}\left(\delta_x\times(\sum_{y\in\mathcal{T}}p(y|x)-1)\right)\end{aligned} \tag{6}$$

优化目标为：

$$(p^*,\lambda_k^*,\delta_x^*) = \arg\max_{p,\lambda_k,\delta_x}\mathcal{E}(p,\lambda_k,\delta_x,\mathcal{D}) \tag{7}$$

进而对 $\mathcal{E}(p,\lambda_k,\delta_x,\mathcal{D})$ 求 $p(y|x)$ 的偏导：

$$\frac{\partial\mathcal{E}(p,\lambda_k,\delta_x,\mathcal{D})}{\partial p(y|x)} = -\tilde{p}(x)\times(\log p(y|x)+1)+\left(\sum_k\lambda_k\tilde{p}(x)f_k(x,y)\right)+\delta_x \tag{8}$$

令偏导为 0，可以得到：

$$\begin{aligned}p(y|x) &= e^{\sum_k\lambda_k f_k(x,y)}\times e^{\frac{\delta_x}{\tilde{p}(x)}-1}\\ &= \frac{e^{\sum_k\lambda_k f_k(x,y)}}{e^{1-\frac{\delta_x}{\tilde{p}(x)}}}\end{aligned} \tag{9}$$

结合第二个约束，即公式 (4)，可以得到：

$$\begin{aligned}Z(x) &= e^{1-\frac{\delta_x}{\tilde{p}(x)}}\\ &= \sum_{y\in\mathcal{T}}e^{\sum_k\lambda_k f_k(x,y)}\end{aligned} \tag{10}$$

进而可以求解 δ_x(但是没什么必要了)。$Z(x)$ 一般称为 normalization factor，保证概率和为 1。

目前为止，通过推导，我们确定了最大熵模型的参数形式（同时也得到了 δ_x），即：

$$p(y|x) = \frac{e^{\sum_k\lambda_k f_k(x,y)}}{Z(x)} \tag{11}$$

对这个参数形式取对数，那么我们可以得到：

$$\log p(y|x) = \left(\sum_k\lambda_k f_k(x,y)\right)-\log Z(x) \tag{12}$$

不考虑 $\log Z(x)$ 这一项，可以看到模型是关于特征向量的线性形式，因此称为**对数线性模型**。

回到公式 (7)，我们确定了 δ_x 的值和 $p(y|x)$ 的参数形式，并且都和 λ_k 相关，因此重新定义一个函数：

$$\begin{aligned}\mathcal{F}(p,\lambda_k,\mathcal{D}) = &-\sum_{\substack{x\in\mathcal{D}\\y\in\mathcal{T}}}\tilde{p}(x)p(y|x)\log p(y|x)\\ &+\sum_k\left(\lambda_k\times\left(\sum_{\substack{x\in\mathcal{D}\\y\in\mathcal{T}}}\tilde{p}(x)p(y|x)f_k(x,y)-\sum_{(x,y)\in\mathcal{D}}\tilde{p}(x,y)f_k(x,y)\right)\right)\end{aligned} \tag{13}$$

最优化问题转化为：

$$
\begin{aligned}
\lambda_k^* &= \arg\max_{\lambda_k} \mathcal{F}(p, \lambda_k, \mathcal{D}) \\
&= \arg\max_{\lambda_k} -\sum_{\substack{x\in\mathcal{D}\\ y\in\mathcal{T}}} \tilde{p}(x)p(y|x)\log p(y|x) \\
&\quad + \sum_k \left(\lambda_k \times \left(\sum_{\substack{x\in\mathcal{D}\\ y\in\mathcal{T}}} \tilde{p}(x)p(y|x)f_k(x,y) - \sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y)f_k(x,y) \right) \right) \\
&= \arg\max_{\lambda_k} -\sum_{\substack{x\in\mathcal{D}\\ y\in\mathcal{T}}} \tilde{p}(x)p(y|x) \left(\sum_k \lambda_k f_k(x,y) - \log Z(x) \right) \\
&\quad + \sum_{\substack{x\in\mathcal{D}\\ y\in\mathcal{T}}} \tilde{p}(x)p(y|x) \sum_k \lambda_k f_k(x,y) - \sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y) \sum_k \lambda_k f_k(x,y) \\
&= \arg\max_{\lambda_k} -\left(\sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y) \sum_k \lambda_k f_k(x,y) - \sum_{\substack{x\in\mathcal{D}\\ y\in\mathcal{T}}} \tilde{p}(x)p(y|x)\log Z(x) \right) \\
&= \arg\max_{\lambda_k} -\left(\sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y) \sum_k \lambda_k f_k(x,y) - \sum_{x\in\mathcal{D}} \tilde{p}(x)\log Z(x) \sum_{y\in\mathcal{T}} p(y|x) \right) \qquad (14) \\
&= \arg\max_{\lambda_k} -\left(\sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y) \sum_k \lambda_k f_k(x,y) - \sum_{x\in\mathcal{D}} \tilde{p}(x)\log Z(x) \right) \\
&= \arg\max_{\lambda_k} -\left(\sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y) \sum_k \lambda_k f_k(x,y) - \sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y)\log Z(x) \right) \\
&= \arg\max_{\lambda_k} -\sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y) \left(\sum_k \lambda_k f_k(x,y) - \log Z(x) \right) \\
&= \arg\max_{\lambda_k} -\sum_{(x,y)\in\mathcal{D}} \tilde{p}(x,y)\log p(y|x) \\
&= \arg\max_{\lambda_k} -\frac{1}{N}\sum_{j=1}^{N} \log p(y^j|x^j) \\
&= \arg\max_{\lambda_k} -\frac{1}{N}\mathcal{LL}(\mathcal{D};\lambda_k) \\
&= \arg\max_{\lambda_k} -\mathcal{LL}(\mathcal{D};\lambda_k)
\end{aligned}
$$

其中，$\mathcal{LL}(\mathcal{D};\lambda_k)$ 表示数据的对数似然。可以看到，最大熵和最大似然估计有一定的联系。应该是完全等价才对，不知道我哪儿推导错了!! 或许，此处需要用对偶问题的方式求解，可以继续研究李航老师的《统计机器学习》。应该是对拉格朗日乘数理解不到位，拉格朗日乘数应该只能求极值，而不能保证最大值或最小值，所以符号其实没那么重要？

2 Log-linear model 的似然函数及梯度

从上面推导可以看到，最大熵原理和最大化 log-linear 模型的似然是等价的。数据的似然为：

$$\mathcal{LL}(\mathcal{D};\lambda_k)=\sum_{j=1}^{N}\log p(y^j|x^j) \tag{15}$$

求解的问题是最大化数据的似然：

$$\lambda_k^*=\arg\max_{\lambda_k}\mathcal{LL}(\mathcal{D};\lambda_k) \tag{16}$$

公式 (16) 无法得到解析解，因此只能通过迭代的方式，梯度下降求解。梯度为：

$$\begin{aligned}
&\frac{\partial\mathcal{LL}(\mathcal{D};\lambda_k}{\partial\lambda_k})\\
=&\frac{\partial\sum_{j=1}^{N}\log p(y^j|x^j)}{\partial\lambda_k}\\
=&\frac{\partial\sum_{j=1}^{N}\sum_{k}\lambda_k f_k(x,y)-\log Z(x)}{\partial\lambda_k}\\
=&\sum_{j=1}^{N}\left(f_k(x,y)-\frac{\sum_{y\in\mathcal{T}}f_k(x,y)\times e^{\sum_k\lambda_k f_k(x,y)}}{Z(x)}\right)\\
=&\sum_{j=1}^{N}\left(f_k(x,y)-\sum_{y\in\mathcal{T}}p(y|x)f_k(x,y)\right)
\end{aligned} \tag{17}$$

前一项称为 empirical counts，后一项称为 model expectation。

3 以词性标注为例，实现一个最大熵模型

符号定义、特征模板等内容，和 Linear Model 完全相同。给定句子 S^j，其第 i 个词，标为词性 y 的概率为：

$$p(y|S^j,i;\mathbf{w})=\frac{e^{\mathbf{w}\cdot\mathbf{f}(S^j,i,y)}}{\sum_{y'\in\mathcal{T}}e^{\mathbf{w}\cdot\mathbf{f}(S^j,i,y')}}$$

3.1 Stochastic Gradient Descent (SGD) 训练，basic 版本

训练和测试阶段和 Linear Model 基本相同。算法1给出了一个最基本的 SGD 训练算法。其中，第 (7) 行中，y_i^j 为正确答案，括号内是是用当前模型 $\mathbf{w}^k$，对当前实例计算梯度，参见公式17。

4 编程作业

编写一个最大熵模型，包括训练、测试、评价等程序。数据和 HMM 模型使用的数据（训练数据 train.conll、开发数据 dev.conll）相同。满分 15 分。

Algorithm 1 SGD training (basic).

1: **Input:** Labeled data $\mathcal{D}$, and Feature space $\mathcal{E}$
2: **Output:** Feature weights $\mathbf{w}$（$\mathbf{w}$ 和上面公式中的 λ_k 是相同的）。
3: **Initialization:** $\mathbf{w}^0 = \mathbf{0}$, 梯度向量 $\mathbf{g} = \mathbf{0}$, $k = 0$, batch size 设为 $B = 50$, $b = 0$;
4: **for** $m = 1$ **to** M **do** {*iterations*}
5: **for** $j = 1$ **to** N **do** {*for sentence* S^j}
6: **for** $i = 1$ **to** n_j **do** {*for word* w_i^j}
7: $\mathbf{g} = \mathbf{g} + \left(\mathbf{f}(S^j, i, y_i^j) - \sum_{y \in \mathcal{T}} p(y|S^j, i; \mathbf{w}^k) \mathbf{f}(S^j, i, y) \right)$
8: $b = b + 1$
9: **if** $B = b$ **then** {*batch is over, update weights*}
10: $\mathbf{w}^{k+1} = \mathbf{w}^k + \mathbf{g}$
11: $k = k + 1, b = 0, \mathbf{g} = \mathbf{0}$
12: **end if**
13: **end for**
14: **end for**
15: Here, after each iteration, we can evaluate the current model $\mathbf{w}$ on some validation (development) dataset.
16: **end for**
17: Output $\mathbf{w}$

5 SGD 训练，L2 regularization

以公式（15）为优化目标，训练得到的模型参数 λ_k 很容易 overfitting 到 training data 上。因此，一般会增加一个正则项，约束模型参数。此处增加一个二阶范式（norm 2）https://en.wikipedia.org/wiki/Norm_(mathematics)#p-norm。

$$\begin{aligned} \mathcal{LL}'(\mathcal{D}; \lambda_k) &= \mathcal{LL}(\mathcal{D}; \lambda_k) - \frac{C}{2} \sum_k \lambda_k^2 \\ \lambda_k^* &= \arg\max_{\lambda_k} \mathcal{LL}'(\mathcal{D}; \lambda_k) \end{aligned} \tag{18}$$

其中，$0 \le C \le 1$ 为正则项的权重。即在最大化数据似然的同时，让权重尽可能小。

此时，梯度为：

$$\frac{\partial \mathcal{LL}'(\mathcal{D}; \lambda_k}{\partial \lambda_k}) = \frac{\partial \mathcal{LL}(\mathcal{D}; \lambda_k}{\partial \lambda_k}) - C\lambda_k \tag{19}$$

可以尝试在算法1中增加正则项（试试 $C = 0.01$）。

6 模拟退火

为了让梯度下降训练算法更好更快的收敛到一个比较好的局部最优解，一般用模拟退火的方式，开始时更新步长 η 较大，之后逐渐减小步长（微调）。即算法1第10行修改为：

$$\mathbf{w}^{k+1} = \mathbf{w}^k + \eta \mathbf{g}$$

7 一段真实的 C++ 代码，同时使用 L2 regularization 和模拟退火，供参考

```
batch_size = 50;
init_eta = 0.0125; // initial update step
C = 0.1; // regularization term

_lambda = 2 * C / N;
// N: number of instances (not sentence in the case of word-wise POS tagging)

_t0 = 1 / (_lambda * init_eta);
_t = 0;

int epoch = 1;
for (; epoch <= num_epochs; ++epoch) { // iterations
    _decay = 1.;
    while (i < N) {
        /* Update various factors. */
        _eta = 1 / (_lambda * (_t0 + _t));
        _decay *= (1.0 - _eta * _lambda);
        _gain = _eta / _decay;

        m_param.set_scale(_decay);

        // store the updates in _g
        _g.resize(K);
        fill(_g.begin(), _g.end(), 0.);

        int b = 0;
        for (; b < batch_size; ++b,++i) {
            if (i >= N) break;

            // compute gradient and add to _g
            // codes skiped

        }
        _t += b;
        // do the updates
        for (int k = 0; k < K; ++k) m_param.c_buf()[k] += _g[k];
    }

    /* Scale the feature weights. */
    vecscale(m_param.c_buf(), _decay, K);
}
```

8 Adwait Ratnaparkhi: A Simple Introduction to Maximum Entropy Models for Natural Language Processing (1997)

另外，我仔细阅读了 Adwait Ratnaparkhi 1997 的关于最大熵的技术报告，也很受启发，有兴趣的同学可以在网上找一下。下面是阅读中的一些理解：

$a \in \mathcal{A}$, $\mathcal{A}$ 表示类别集合
$b \in \mathcal{B}$, $\mathcal{B}$ 表示 context 的集合

context 刚开始读不容易理解，应该表示一个实例除了类别外的所有信息，从 context 可以根据特征模板抽取各种具体的特征。

Theorem 1 证明中：
为什么 uniform distribution $u \in Q$?
可以让所有特征对应的 α_j 都相同，对于每一个类别，通过特征模板实例化出来的特征数是一样的（特征空间不要只考虑正特征），这样每个类别的概率都相同。

Definition 2，$H(p)$ 定义少一个负号

所有证明中，都是使用联合概率 $p(x) = p(a, b)$，似乎便于证明。而我的讲义中，都使用条件概率 $p(a|b)$