三、7.2 最大熵模型----代码讲解

2016年3月19日 星期六　　下午1:48

一、简介

在github找了一份开源的最大熵代码，是google检索"maxent github"结果排名第一的。

读了代码主要部分，并做了代码注释，下面写出代码的思路和代码中需要注意的细节。

下面只讲解lbfgs的解法。

二、原理

三、思路、执行过程

训练之前初始化

构造param

param作用是记录f_k（x_a，y_b）中，记录x_a，y_b->k的映射。

这个变量主要的作用是在训练过程中，计算梯度的时候用到，可以先看看那部分。我们在计算梯度的过程中，知道a、b可以通过param知道k。

训练初始化

计算观测数据上f的期望E（f）

这个期望是在观测数据上，f的期望，不需要训练就可以得到，仅仅是A事件观测数据上的概率*f（A），也就是A事件的频度*f（A），统计可得。

详见LBFGSTrainer::init_trainer()

对于过每一轮的数据

对于每个样本

计算概率分布、预测概率最大的类，相当于前馈

计算后验概率的公式（见统计学习方法）

$$P_w(y|x)=\frac{1}{Z_w(x)}\exp\left(\sum_{i=1}^{n}w_i f_i(x,y)\right)$$

首先，计算未归一化概率P'（y | x）= Σ w_i * f（x，y）在代码中，w_i变量名是m_theta（或者是x），f变量名是fval，fval = context[i].second，这里f（x，y）应该是随训练数据一起输入的已知量，而不是参数，其实f（x，y）就是输入样本的特征值！！！。在作者的实现中，是和训练样本一起读入的，是只和x有关的，也就是∀y， f（x）= f（x，y）。

```
for (size_t i = 0;i < len; ++i) {
    vector<pair<size_t, size_t> >& param = (*m_params)[context[i].first];
    float fval = context[i].second; // sample' weight / input's prior probability / f_i(a, b)
    for (size_t j = 0; j < param.size(); ++j) {
        size_t oid = param[j].first;  // oid is class label (k-th in K class)
        probs[oid] += m_theta[param[j].second] * fval; // param[j].second is the index of W, m_theta is weight matrix W
        // P = sum of w_i * f_i(x, y)   P is no softmax probability
        // P' = exp(P) / Z(x)     P' is after softmax
    }
}
```

然后，用softmax进行归一化，这部分不再赘述。

对于fval（输入的特征值）部分代码可以看maxent.cpp load_events() 和maxent.cpp get_sample()

计算梯度

计算梯度的公式（见统计学习方法）

$$\frac{\partial f(w)}{\partial w_i}=\sum_{x,y}\tilde{P}(x)P_w(y|x)f_i(x,y)-E_{\tilde{p}}(f_i),\quad i=1,2,\cdots,n$$

从公式，我们看出是模型学出的参数的期望 和观测数据期望的差。而且计算的梯度是参数w的梯度。

首先，在初始化阶段计算出的观测数据上期望

E_p（f_i）可以在本轮训练数据训练前就减去。

其次，计算模型学出的参数的期望。

对于样本中的每个特征（m_context[i]）

我们能找到他在公式f_k（x_a，y_b）中的x_a，代码中是pid = it->m_context[i].first。

我们能找到f（x，y），这是和训练样本一起输入的，而不是训练的参数，代码中是fval = it->m_context[i].second。

对于特征x情况下，输出y有K种选择，K是多分类的类别数，代码中是param.size()。

我们可以找到每个类id，oid = param[j].first

也可以找到此f_k（x_a，y_b），对应的权重参数W对应维度的下标fid = param[j].second

这里注意，权重参数W_k和f_k（x_a，y_b）是一一对应的，我们确定了（x_a，y_b）也就确定了w

W在k维的梯度被加上了P（y|x）* f（x，y）

g[fid] += it->m_count * q[oid] * fval;

这里作者不考虑P~（x）了，也可能是把P~（x）当做样本x的先验加在f（x，y）中

```
// [begin] get grad
// what's the meaning of pid, m_params, fid. see at MaxentModel::build_params
for (size_t i = 0; i < it->context_size(); ++i) {
    size_t pid = it->m_context[i].first;  // context_precidate_id, the x_a of (x_a, y_b)pair
                                          // one context_precidate_id is not one (x, y)pair.
                                          // x_a is feature, y_b is ground truth value (which class is right)
    float fval = it->m_context[i].second; // feature value of f_k(x_a, y_b)) (for every b).
                                          // feature_value is the weight between input feature(x_a) and output class(y_b)
                                          // feature_value is from input sample, but not auto product. see at maxent.cpp load_events() and maxent.cpp get_sample()

    vector<pair<size_t, size_t> >& param = (*m_params)[pid]; //pid is x_a, is context_precidate_id
    for (size_t j = 0; j < param.size(); ++j) { // param.size() is outcome_size (total class of classify)
        size_t oid = param[j].first;  // outcome_id (k-th of K-class)  first dim of weight W and f_k(x, y), is y's index
        size_t fid = param[j].second; // feature_id, index of W, g[fid] is the grad of Wk
                                      // fid is the "k" in f_k(x_a, y_b)

        //[get grad] step 2: sum of p(x_a) * p(y_b|x_a) * f_k(x_a, y_b)
        g[fid] += it->m_count * q[oid] * fval; // count * p(y|x) * f_k(x_a, y_b)   where is P(x) (x's proir probability)
    }
}
//[end] get grad
```

```
// [begin] get grad
// what's the meaning of pid, m_params, fid. see at MaxentModel::build_params
for (size_t i = 0; i < it->context_size(); ++i) {
    size_t pid = it->m_context[i].first;  // context_precidate_id, the x_a of (x_a, y_b)pair
                                          // one context_precidate_id is not one (x, y)pair.
                                          // x_a is feature, y_b is ground truth value (which class is right)
    float fval = it->m_context[i].second; // feature value of f_k(x_a, y_b)) (for every b).
                                          // feature_value is the weight between input feature(x_a) and output class(y_b)

    vector<pair<size_t, size_t> >& param = (*m_params)[pid]; //pid is x_a, is context_precidate_id
    for (size_t j = 0; j < param.size(); ++j) { // param.size() is outcome_size (total class of classify)
        size_t oid = param[j].first;  // outcome_id (k-th of K-class)  first dim of weight W and f_k(x, y), is y's index
        size_t fid = param[j].second; // feature_id, index of W, g[fid] is the grad of Wk
                                      // fid is the "k" in f_k(x_a, y_b)

        //[get grad] step 2: sum of p(x_a) * p(y_b|x_a) * f_k(x_a, y_b)
        g[fid] += it->m_count * q[oid] * fval; // count * p(y|x) * f_k(x_a, y_b)   where is P(x) (x's proir probability)
    }
}
//[end] get grad
```

使用LBFGS计算需要更新的梯度

需要权重参数W（代码中是x，也是m_theta），一阶的梯度g（上一步算出了），目前目标函数的值，代码中是f。

用这些计算二阶导，具体计算的过程还需要看看。

三、回顾

输入：

样本，包含直接作为输入的特征值。

特征key-特征value的映射

标准，label，包含K类

输出：

训练好的模型w，可以计算后验概率的接口

未知参数：

权重参数w

可以直接求的变量：

观测数据的期望 E~（f_p）

训练过程

1、前馈，计算后验概率

2、根据现有参数W和标注，计算W的梯度（一阶导）

3、把目标函数的结果、梯度、目前的参数W值，作为输入进行反馈更新

四、注释版完整代码

tianzhiliang@cp01-rdqa-dev371.cp01.baidu.com:/home/users/tianzhiliang/tools/code_ml_dnn/maxent/maxent