链滴

优化版 JAVA 最大熵模型（GIS 训练）

作者：bestjimmy
原文链接：https://ld246.com/article/1534473498372
来源网站：链滴

网上现有的最大熵模型，如：https://blog.csdn.net/nwpuwyk/article/details/37500371
该代码在训练环节性能较差，特征函数存储的结构也涉及较简单。
我在该版本基础上进行了改进，优化了特征函数的数据结构和训练代码。
<pre><code class="highlight-chroma">   /**
  * 样本数据集
  */
  List<Instance
gt; instanceList = new ArrayList<Instance>();
  /**
  * 特征列表，来
所有事件的统计结果
  */
</code></pre>
//   Map<String,Feature> featureMap=new HashMap<>();
/**
* 每个特征的出现次数
*/
//Map<String,Integer> featureCountMap=new HashMap<>();
/**
* 事件（类别）集
*/
List labels = new ArrayList();
/**
* 每个特征函数的权重
*/
// double[] weight;
Map<String,Weight> weightMap=new HashMap<>();
/**
* 一个事件最多一共有多少种特征
*/
double learningRate=10;
int C;
Map<String,List> testInstance;
/**
* 样本数据集 */ List instanceList = new ArrayList();
/**
* 特征列表，来自所有事件的统计结果 ///   Map featureMap=new HashMap<>();
/**
* 每个特征的出现次数 */ //Map featureCountMap=new HashMap<>();
/**
* 事件（类别）集 */ List labels = new ArrayList();
/**
* 每个特征函数的权重 */ // double[] weight;
Map<String,Weight> weightMap=new HashMap<>();
/**
* 一个事件最多一共有多少种特征 */ double learningRate=10;
int C;
<pre><code class="highlight-chroma"> * 训练模型 * @param maxIt 最大迭代次数
  */public void tran(int maxIt,String savePath) throws IOException {

```
    Map,Double&t; empiricalE = new HashMap&lt;&gt;(); // 经验期望
    Map,Double&gt modelE = new HashMap&lt;&gt;(); // 模型期望

    for (Map.EntryWeight&gt; e:weightMap.entrySet())
    {
        double rati=(double) e.getValue().getCnt() / instanceList.size();
    empiricalE.put(egetKey(),ratio);
    }
    Map,Double&t; lastWeight=new HashMap&lt;&gt;();
   for (int i = 0; i &l; maxIt; ++i)
    {
        System.out.rintln("iter:"+i);
    computeModeEmodelE);//计算模型期望
    System.out.printn("model finish.updating...");
   for (Map.Entry,eight&gt; e:weightMap.entrySet())
        {
          //lastWeiht[w] = weight[w];
    lastWeight.put(.getKey(),e.getValue().getWeight());
    String f=e.getKy();
   double delta=lerningRate / C * Math.log(empiricalE.get(f)/ modelE.get(f));
    weightMap.get().addWeight(delta);
    }
        System.out.rintln("saving iter:"+i);
    learningRate*=099;
    learningRate=larningRate&lt;10?10:learningRate;
    saveParam(savPath+"ent_insopt.par"+i);
   if (checkConverence(lastWeight, weightMap)) break;

    }

}

 /**
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">   * 预测类别 * @p
ram fieldList
</span></span><span class="highlight-line"><span class="highlight-cl">    * @return
</span></span><span class="highlight-line"><span class="highlight-cl">    */
</span></span><span class="highlight-line"><span class="highlight-cl">  public Pair, Doub
e&amp;gt;[] predict(Map,Integer&amp;gt; fieldList)
</span></span><span class="highlight-line"><span class="highlight-cl">  {
</span></span><span class="highlight-line"><span class="highlight-cl">      double[] prob
= calProb(fieldList);
</span></span><span class="highlight-line"><span class="highlight-cl">    Pair, Double&gt
[] pairResult = new Pair[prob.length];
</span></span><span class="highlight-line"><span class="highlight-cl">    for (int i = 0; i &l
; prob.length; ++i)
</span></span><span class="highlight-line"><span class="highlight-cl">      {
</span></span><span class="highlight-line"><span class="highlight-cl">          pairResult[i]
= new Pair, Double&gt;(labels.get(i), prob[i]);
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">      return pairRes
lt;
</span></span><span class="highlight-line"><span class="highlight-cl">  }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">  /**
</span></span><span class="highlight-line"><span class="highlight-cl">    * 检查是否收敛 *
@param w1
</span></span><span class="highlight-line"><span class="highlight-cl">    * @param w2
</span></span><span class="highlight-line"><span class="highlight-cl">    * @return 是否
敛
</span></span><span class="highlight-line"><span class="highlight-cl">    */public boolean
checkConvergence(Map,Double&amp;gt; w1, Map,Weight&amp;gt; w2)
</span></span><span class="highlight-line"><span class="highlight-cl">  {
</span></span><span class="highlight-line"><span class="highlight-cl">      System.out.pri
tln("w1 size:"+w1.size());
</span></span><span class="highlight-line"><span class="highlight-cl">    boolean flag=tr
e;
</span></span><span class="highlight-line"><span class="highlight-cl">    for (Map.Entry,
ouble&amp;gt; e1:w1.entrySet())
</span></span><span class="highlight-line"><span class="highlight-cl">      {
</span></span><span class="highlight-line"><span class="highlight-cl">          //System.ou
.println("thread:"+Math.abs(e1.getValue() - w2.get(e1.getKey())) );
</span></span><span class="highlight-line"><span class="highlight-cl">     if (Math.abs(e1.
etValue() - w2.get(e1.getKey()).getWeight()) &amp;gt;= 1e-4)    // 收敛阀值0.01可自行调整
</span></span><span class="highlight-line"><span class="highlight-cl">     flag=false;
</span></span><span class="highlight-line"><span class="highlight-cl">     }
</span></span><span class="highlight-line"><span class="highlight-cl">      return flag;
</span></span><span class="highlight-line"><span class="highlight-cl">  }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">  /**
</span></span><span class="highlight-line"><span class="highlight-cl">    * 计算模型期望
即在当前的特征函数的权重下，计算特征函数的模型期望值。 * @param modelE 储存空间，应当事
分配好内存（之所以不return一个modelE是为了避免重复分配内存）
</span></span><span class="highlight-line"><span class="highlight-cl">    */public void c
mputeModeE(Map,Double&amp;gt; modelE)
</span></span><span class="highlight-line"><span class="highlight-cl">  {
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">    modelE.clear();
</span></span><span class="highlight-line"><span class="highlight-cl">  double rate=1.0
/ instanceList.size();
</span></span><span class="highlight-line"><span class="highlight-cl">  for (int i = 0; i &l
; instanceList.size(); ++i)
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">      Map,Intege
&gt; fieldMap = instanceList.get(i).fieldList;//no labels
</span></span><span class="highlight-line"><span class="highlight-cl">  //计算当前样本
对应所有类别的概率  double[] pro = calProb(fieldMap);
</span></span><span class="highlight-line"><span class="highlight-cl">  for (Map.Entry,In
eger&gt; e:fieldMap.entrySet())
</span></span><span class="highlight-line"><span class="highlight-cl">      {
</span></span><span class="highlight-line"><span class="highlight-cl">        String ins
eature=e.getKey();/**
</span></span><span class="highlight-line"><span class="highlight-cl">   * 训练模型
</span></span><span class="highlight-line"><span class="highlight-cl">   * @param maxI
 最大迭代次数
</span></span><span class="highlight-line"><span class="highlight-cl">   */
</span></span><span class="highlight-line"><span class="highlight-cl">  public void train
int maxIt,String savePath) throws IOException {
</span></span><span class="highlight-line"><span class="highlight-cl">    Map&lt;Strin
,Double&gt; empiricalE = new HashMap&lt;&gt;();  // 经验期望
</span></span><span class="highlight-line"><span class="highlight-cl">    Map&lt;Strin
,Double&gt; modelE = new HashMap&lt;&gt;();      // 模型期望
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    for (Map.Entr
&lt;String,Weight&gt; e:weightMap.entrySet())
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">      double rat
o=(double) e.getValue().getCnt() / instanceList.size();
</span></span><span class="highlight-line"><span class="highlight-cl">      empiricalE
put(e.getKey(),ratio);
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">    Map&lt;Strin
,Double&gt; lastWeight=new HashMap&lt;&gt;();
</span></span><span class="highlight-line"><span class="highlight-cl">    for (int i = 0; i
&lt; maxIt; ++i)
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">      System.out
println("iter:"+i);
</span></span><span class="highlight-line"><span class="highlight-cl">      compute
odeE(modelE);//计算模型期望
</span></span><span class="highlight-line"><span class="highlight-cl">      System.out
println("model finish.updating...");
</span></span><span class="highlight-line"><span class="highlight-cl">      for (Map.E
try&lt;String,Weight&gt; e:weightMap.entrySet())
</span></span><span class="highlight-line"><span class="highlight-cl">      {
</span></span><span class="highlight-line"><span class="highlight-cl">        //lastWe
ght[w] = weight[w];
</span></span><span class="highlight-line"><span class="highlight-cl">        lastWei
ht.put(e.getKey(),e.getValue().getWeight());
</span></span><span class="highlight-line"><span class="highlight-cl">        String f
```

```
e.getKey();
</span></span><span class="highlight-line"><span class="highlight-cl">                double
elta=learningRate / C * Math.log(empiricalE.get(f)/ modelE.get(f));
</span></span><span class="highlight-line"><span class="highlight-cl">                weight
ap.get(f).addWeight(delta);
</span></span><span class="highlight-line"><span class="highlight-cl">            }
</span></span><span class="highlight-line"><span class="highlight-cl">            System.out
println("saving iter:"+i);
</span></span><span class="highlight-line"><span class="highlight-cl">            learningRa
e*=0.99;
</span></span><span class="highlight-line"><span class="highlight-cl">            learningRa
e=learningRate&amp;lt;10?10:learningRate;
</span></span><span class="highlight-line"><span class="highlight-cl">            saveParam
savePath+"ent_insopt.par"+i);
</span></span><span class="highlight-line"><span class="highlight-cl">            if (checkC
nvergence(lastWeight, weightMap)) break;
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    /**
</span></span><span class="highlight-line"><span class="highlight-cl">     * 预测类别
</span></span><span class="highlight-line"><span class="highlight-cl">     * @param field
ist
</span></span><span class="highlight-line"><span class="highlight-cl">     * @return
</span></span><span class="highlight-line"><span class="highlight-cl">     */
</span></span><span class="highlight-line"><span class="highlight-cl">    public Pair&amp;lt;S
ring, Double&amp;gt;[] predict(Map&amp;lt;String,Integer&amp;gt; fieldList)
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">        double[] prob
= calProb(fieldList);
</span></span><span class="highlight-line"><span class="highlight-cl">        Pair&amp;lt;String
 Double&amp;gt;[] pairResult = new Pair[prob.length];
</span></span><span class="highlight-line"><span class="highlight-cl">        for (int i = 0; i
&amp;lt; prob.length; ++i)
</span></span><span class="highlight-line"><span class="highlight-cl">        {
</span></span><span class="highlight-line"><span class="highlight-cl">            pairResult[i
 = new Pair&amp;lt;String, Double&amp;gt;(labels.get(i), prob[i]);
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">        return pairRe
ult;
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    /**
</span></span><span class="highlight-line"><span class="highlight-cl">     * 检查是否收敛
</span></span><span class="highlight-line"><span class="highlight-cl">     * @param w1
</span></span><span class="highlight-line"><span class="highlight-cl">     * @param w2
</span></span><span class="highlight-line"><span class="highlight-cl">     * @return 是
收敛
</span></span><span class="highlight-line"><span class="highlight-cl">     */
</span></span><span class="highlight-line"><span class="highlight-cl">    public boolean
heckConvergence(Map&amp;lt;String,Double&amp;gt; w1, Map&amp;lt;String,Weight&amp;gt; w2)
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">        System.out.println("w1 size:"+w1.size());
</span></span><span class="highlight-line"><span class="highlight-cl">        boolean flag=true;
</span></span><span class="highlight-line"><span class="highlight-cl">        for (Map.Entry&lt;String,Double&gt; e1:w1.entrySet())
</span></span><span class="highlight-line"><span class="highlight-cl">        {
</span></span><span class="highlight-line"><span class="highlight-cl">            //System.out.println("thread:"+Math.abs(e1.getValue() - w2.get(e1.getKey())) );
</span></span><span class="highlight-line"><span class="highlight-cl">            if (Math.abs(e1.getValue() - w2.get(e1.getKey()).getWeight()) &gt;= 1e-4)    // 收敛阀值0.01可自行调整
</span></span><span class="highlight-line"><span class="highlight-cl">                flag=false;
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">        return flag;
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    /**
</span></span><span class="highlight-line"><span class="highlight-cl">     * 计算模型期望即在当前的特征函数的权重下，计算特征函数的模型期望值。
</span></span><span class="highlight-line"><span class="highlight-cl">     * @param modelE 储存空间，应当事先分配好内存（之所以不return一个modelE是为了避免重复分配内存）
</span></span><span class="highlight-line"><span class="highlight-cl">     */
</span></span><span class="highlight-line"><span class="highlight-cl">    public void computeModeE(Map&lt;String,Double&gt; modelE)
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">        modelE.clear();
</span></span><span class="highlight-line"><span class="highlight-cl">        double rate=1.0 / instanceList.size();
</span></span><span class="highlight-line"><span class="highlight-cl">        for (int i = 0; i &lt; instanceList.size(); ++i)
</span></span><span class="highlight-line"><span class="highlight-cl">        {
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">            Map&lt;String,Integer&gt; fieldMap = instanceList.get(i).fieldList;//no labels
</span></span><span class="highlight-line"><span class="highlight-cl">            //计算当样本X对应所有类别的概率
</span></span><span class="highlight-line"><span class="highlight-cl">            double[] po = calProb(fieldMap);
</span></span><span class="highlight-line"><span class="highlight-cl">            for (Map.Entry&lt;String,Integer&gt; e:fieldMap.entrySet())
</span></span><span class="highlight-line"><span class="highlight-cl">            {
</span></span><span class="highlight-line"><span class="highlight-cl">                String inFeature=e.getKey();
</span></span><span class="highlight-line"><span class="highlight-cl">                int cnt=.getValue();
</span></span><span class="highlight-line"><span class="highlight-cl">                for (int k = 0; k &lt; labels.size(); k++)
</span></span><span class="highlight-line"><span class="highlight-cl">                {
</span></span><span class="highlight-line"><span class="highlight-cl">                    String feature=labels.get(k)+":"+insFeature;
</span></span><span class="highlight-line"><span class="highlight-cl">                    if (w
```

```
ightMap.containsKey(feature)) {
                    double  delta=pro[k] * rate*cnt;
                    modelE.put(feature, modelE.containsKey(feature) ? modelE.get(feature) + delta : delta);
                }
            }
        }
    }
//    public class ode implements Runnable
//    {
//        ConcurrentLinkedQueue&lt;Integer&gt; insQueue=new ConcurrentLinkedQueue&lt;&gt;();
//        boolean fla=true;
//        List&lt;Instance&gt; iə
//        public void addIns(int i)
//        {
//
//        }
//
//        @Override
//        public void un() {
//            while(fla)
//            {
//                int ins insQueue.poll();
//            }
//        }
//    }
    /**
     * 计算p(y|x),此的x指的是instance里的field
     * @param fieldist 实例的特征列表
     * @return 该例属于每个类别的概率
     */
    public double[] alProb(Map&lt;String,Integer&gt; fieldList)
    {
        double[] p = ew double[labels.size()];
        double sum 0;  // 正则化因子，保证概率和为1
        for (int i = 0; i &lt; labels.size(); ++i)
        {
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">            double we
ghtSum = 0;
</span></span><span class="highlight-line"><span class="highlight-cl">            String labe
=labels.get(i);
</span></span><span class="highlight-line"><span class="highlight-cl">            for (String
ield : fieldList.keySet())
</span></span><span class="highlight-line"><span class="highlight-cl">            {
</span></span><span class="highlight-line"><span class="highlight-cl">                String f
ature=label+":"+field;
</span></span><span class="highlight-line"><span class="highlight-cl">                 if (weig
tMap.containsKey(feature)) {
</span></span><span class="highlight-line"><span class="highlight-cl">                    weigh
Sum += weightMap.get(feature).getWeight()*fieldList.get(field);
</span></span><span class="highlight-line"><span class="highlight-cl">                }
</span></span><span class="highlight-line"><span class="highlight-cl">            }
</span></span><span class="highlight-line"><span class="highlight-cl">            if(weightS
m&amp;gt;15)
</span></span><span class="highlight-line"><span class="highlight-cl">            {
</span></span><span class="highlight-line"><span class="highlight-cl">                weightS
m=15;
</span></span><span class="highlight-line"><span class="highlight-cl">            }
</span></span><span class="highlight-line"><span class="highlight-cl">            p[i] = Math
exp(weightSum);
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">            sum += p[i
;
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">        //System.out
println();
</span></span><span class="highlight-line"><span class="highlight-cl">        for (int i = 0; i
&lt; p.length; ++i)
</span></span><span class="highlight-line"><span class="highlight-cl">        {
</span></span><span class="highlight-line"><span class="highlight-cl">            p[i] /= sum

</span></span><span class="highlight-line"><span class="highlight-cl">//            if(Double
isNaN(p[i]))
</span></span><span class="highlight-line"><span class="highlight-cl">//            {
</span></span><span class="highlight-line"><span class="highlight-cl">//                System
out.println(p[i]);
</span></span><span class="highlight-line"><span class="highlight-cl">//            }
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">        return p;
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    /**
</span></span><span class="highlight-line"><span class="highlight-cl">     * 一个观测实例
包含事件和时间发生的环境
</span></span><span class="highlight-line"><span class="highlight-cl">     */
</span></span><span class="highlight-line"><span class="highlight-cl">    class Instance i
plements Serializable
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">        /**
</span></span><span class="highlight-line"><span class="highlight-cl">         * 事件（类
），如Outdoor
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">        */
</span></span><span class="highlight-line"><span class="highlight-cl">        String label;
</span></span><span class="highlight-line"><span class="highlight-cl">        /**
</span></span><span class="highlight-line"><span class="highlight-cl">         * 事件发生
环境集合，如[Sunny, Happy]
</span></span><span class="highlight-line"><span class="highlight-cl">         */
</span></span><span class="highlight-line"><span class="highlight-cl">        Map&amp;lt;Strin
,Integer&amp;gt; fieldList = new HashMap&amp;lt;&amp;gt;();
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">        public Instan
e(String label, Map&amp;lt;String,Integer&amp;gt;fieldList)
</span></span><span class="highlight-line"><span class="highlight-cl">        {
</span></span><span class="highlight-line"><span class="highlight-cl">            this.label =
label;
</span></span><span class="highlight-line"><span class="highlight-cl">            this.fieldLis
 = fieldList;
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">    /**
</span></span><span class="highlight-line"><span class="highlight-cl">     * 特征(二值函数)
</span></span><span class="highlight-line"><span class="highlight-cl">     */
</span></span><span class="highlight-line"><span class="highlight-cl">    class Weight
</span></span><span class="highlight-line"><span class="highlight-cl">    {
</span></span><span class="highlight-line"><span class="highlight-cl">        double weigh
=0;
</span></span><span class="highlight-line"><span class="highlight-cl">        int cnt=0;
</span></span><span class="highlight-line"><span class="highlight-cl">        public void a
dWeight(double w)
</span></span><span class="highlight-line"><span class="highlight-cl">        {
</span></span><span class="highlight-line"><span class="highlight-cl">            weight+=
w);
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">        public double
getWeight() {
</span></span><span class="highlight-line"><span class="highlight-cl">            return wei
ht;
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">        public void a
dCnt(int c)
</span></span><span class="highlight-line"><span class="highlight-cl">        {
</span></span><span class="highlight-line"><span class="highlight-cl">            cnt+=c;
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">        public void s
tWeight(double weight) {
</span></span><span class="highlight-line"><span class="highlight-cl">            this.weight
= weight;
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">
</span></span><span class="highlight-line"><span class="highlight-cl">        public int ge
Cnt() {
</span></span><span class="highlight-line"><span class="highlight-cl">            return cnt;
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">
```

```
</span></span><span class="highlight-line"><span class="highlight-cl">        public void s
tCnt(int cnt) {
</span></span><span class="highlight-line"><span class="highlight-cl">            this.cnt =
nt;
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">    int cnt=e.getVal
e();
</span></span><span class="highlight-line"><span class="highlight-cl">    for (int k = 0; k
lt; labels.size(); k++)
</span></span><span class="highlight-line"><span class="highlight-cl">            {
</span></span><span class="highlight-line"><span class="highlight-cl">                String
eature=labels.get(k)+":"+insFeature;
</span></span><span class="highlight-line"><span class="highlight-cl">    if (weightMap.c
ntainsKey(feature)) {
</span></span><span class="highlight-line"><span class="highlight-cl">                    dou
le delta=pro[k] * rate*cnt;
</span></span><span class="highlight-line"><span class="highlight-cl">    modelE.put(fea
ure, modelE.containsKey(feature) ? modelE.get(feature) + delta : delta);
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">            }
</span></span><span class="highlight-line"><span class="highlight-cl">        }
</span></span><span class="highlight-line"><span class="highlight-cl">    }
</span></span><span class="highlight-line"><span class="highlight-cl">}
</span></span></code></pre>
```