数据分析与数据挖掘第三次作业第一大题 – b问分析文档

数据分析与数据挖掘

DAM COURSE, SPRING 2018

BY

1552674 李　源

Tongji University
School of Software Engineering

1 代码运行结果

这里我参照了文档要求，选择了预测结果转为经纬度后的中位误差作为评判分类器预测结果的指标。并且做出了各自的 CDF 曲线。

七种分类器在修正前后的 CDF 曲线如下：

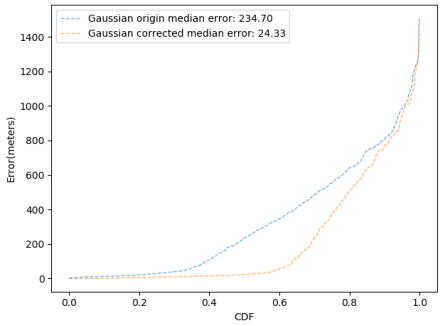


Figure 1.1: Gaussian

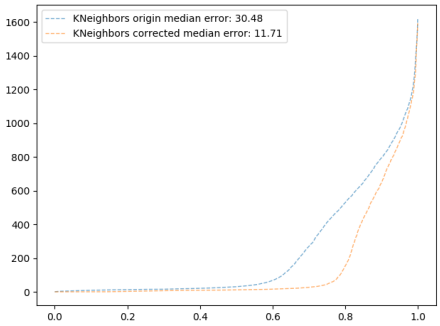


Figure 1.2: KNeighbors

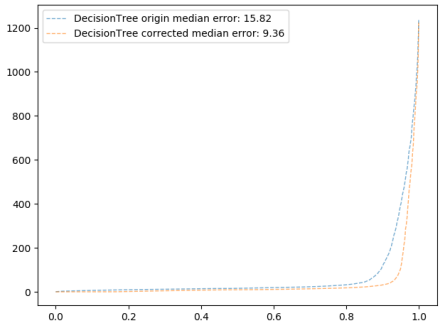


Figure 1.3: DecisionTree

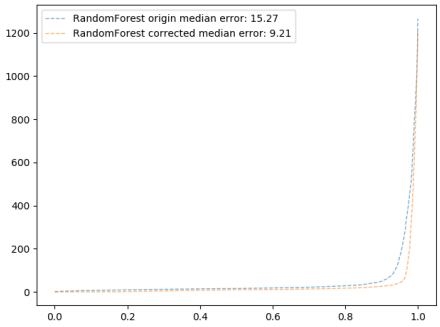


Figure 1.4: RandomForest

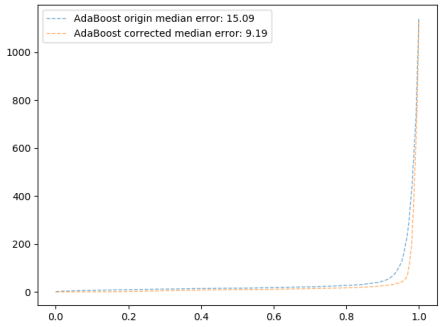


Figure 1.5: AdaBoost

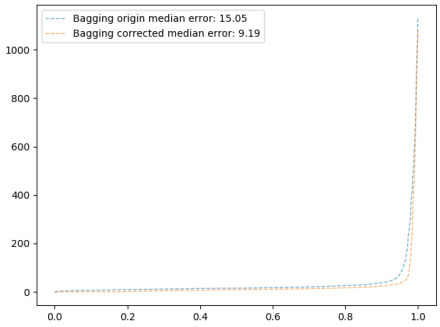


Figure 1.6: Bagging

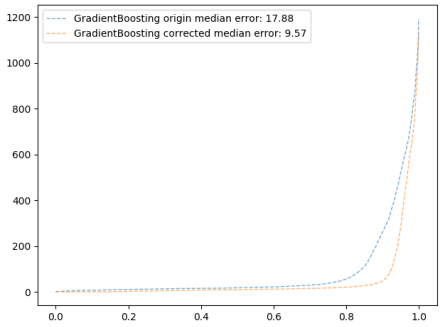


Figure 1.7: GradientBoosting

2 分析讨论

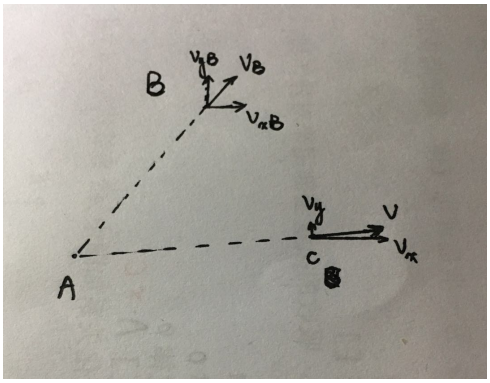
这里我主要对我的修正方法和修正后的结果进行讨论。

2.1 修正方法

我在这里的修正方法，主要是考虑了将单一的 MR 设备的所有点按照时间排序后，是否存在异常点，即**跳点**的情况。

2.1.1 跳点的定义

在给出跳点的定义前，首先给出一些速率的定义。这是我做的一个描述图：



图中的 A、B、C 三点为按照时间戳排序后，连续的三个点。v 代表的是 A、C 之间的速度，v_x 和 v_y 是 v 分解到 x 轴和 y 轴后的速率。v_B 代表的是 A、C 之间的速度，v_{xB} 和 v_{yB} 是 v_B 分解到 x 轴和 y 轴后的速率。那么我可以定义如下几种速率：

我认为从 A 点到 C 点的速率为真实速率，即：

（1）x 轴方向的真实速率：$v_{xt} = v_x$
（2）y 轴方向的真实速率：$v_{yt} = v_y$

我认为从 A 点到 B 点的速率为预测速率，即：

（3）x 轴方向的预测速率：$v_{xp} = v_{xB}$
（4）y 轴方向的预测速率：$v_{yp} = v_{yB}$

对于 B 点和 C 点来说，在 x 轴和 y 轴方向速率改变值可以定义为：

（5）x 轴方向速率改变值为：$v_x^{'} = v_{xt} - v_{xp}$
（6）y 轴方向速率改变值为：$v_y^{'} = v_{yt} - v_{yp}$

那么跳点的定义如下：

对于按照时间戳排序后的连续三点，如果其在 x 轴和 y 轴方向上任意一个速率改变值大于了其方向上的真实速率的 20 倍，那么三点中的第二个点为**跳点**。即：

$$v_x^{'} > v_{xt} * 20 \quad or \quad v_y^{'} > v_{yt} * 20$$

此时，我认为 x 轴和 y 轴方向上预测速率是不正确的，需要进行修正。

2.1.2 具体的修正方法

首先给出修正方法的伪代码：

Algorithm 1 修正方法

while $v_x^{'} > v_{xt} * 20 \quad and \quad v_y^{'} > v_{yt} * 20$ **do**
  $v_{xp} = v_{xp}/10$
  $v_{yp} = v_{yp}/10$
end while
if $v_x^{'} > v_{xt} * 20 \quad and \quad v_y^{'} < v_{yt} * 20$ **then**
  $v_{xp} = v_y^{'}/v_{yt} * v_{xt} + v_{xp}$
end if
if $v_x^{'} < v_{xt} * 20 \quad and \quad v_y^{'} > v_{yt} * 20$ **then**
  $v_{yp} = v_x^{'}/v_{xt} * v_{yt} + v_{yp}$
end if

修正方法可以分为三种情况分别考虑，其步骤说明如下：

（1） 如果 x 轴和 y 轴方向的速率改变值均大于真实速率的 20 倍，那么将 x 轴和 y 轴方向预测速率除以 10。

（2）重复（1）的操作直到有一个方向上的速率改变值小于真实速率的 20 倍，或者（1）已经进行了 5 次以上。

（3）此时如果仍然有一个方向上的速率改变值过大，如 x 轴，那么此时我认为 y 轴方向上的速率改变值是正确的，那么我基于如下的公式：

$$v_y^{'}/v_{yt} = v_x^{'}/v_{xt}$$

来修正 x 轴方向上的速率改变值。并通过修正后的速率改变值来求得新的预测速率。

2.2 修正后的结果

查看前面列出的结果，可以和明显的发现，针对所有的分类器，在经过 b 问的修正过后，其中位误差都有了明显的下降。同时，查看 CDF 曲线可以发现，修正最为明显的是误差刚刚开始提升的时候。

而对于十分极端的误差，我的方法并没有很好的修正，这应该和我限制了修正中步骤（1）的次数有关。

分类器	评价标准		
	原始中位误差	修正后的中位误差	修正率
Gaussian	234.70	24.33	89.63%
KNeighbors	30.48	11.71	60.83%
DecisionTree	15.82	9.36	40.83%
RandomForest	15.27	9.21	39.69%
AdaBoost	15.09	9.19	39.10%
Bagging	15.05	9.19	39.09%
GradientBoosting	17.88	9.57	47.04%

3 性能分析

很明显的可以发现，如果原始误差相对较大，在进行我的修正方法后，能够很明显地提升准确率，比如 Gaussian 分类器其修正率达到了 89.63%。

而如果原始误差本来就较小的话，我的修正方法也能够在一定程度上修正误差，不过效果略差，这也符合逻辑。因为对于这种分类器获得的结果，其需要进行修正的跳点相对就要少一些了。

七种分类器的耗时如下：

分类器	耗时（循环10次结果、a问）	耗时（循环10次结果、b问）
Gaussian	0:00:20.24	0:00:21.54
KNeighbors	0:00:13.12	0:00:13.42
DecisionTree	0:00:16.76	0:00:17.74
RandomForest	0:00:26.09	0:00:28.18
AdaBoost	0:06:21.42	0:06:20.42
Bagging	0:01:28.11	0:01:22.54
GradientBoosting	11:14:17.90	11:17:17.65

Table 1: 分类器耗时

与 a 问相比，耗时并没有出现明显地增加。