算法预测 NBA 中的杰出球员预测与比赛胜负

智能科学与技术 1913416 阎丝雨

- ## Introduction - Motivation

由题可见，本次作业实现了 Project A 的要求。由于本人平时关注篮球比赛动态，在基于对 NBA 比赛机制、评比规则有一定的了解上选择了第一题，希望在兼顾客观的数据处理中努力贴合人们对 NBA 的认知，并选择恰当的算法得出符合事实的结果。

- ## Problem definition

1、数据分析:

1.1 数据集介绍

所有数据集可以分为三类：球员数据、比赛信息、教练数据。包括常规赛（regular_season）、季后赛（playoffs）、全明星赛（allstar）。

其中，常规赛为 NBA30 支球队之间进行的轮回赛。季后赛是在东部和西部前八名之间进行的，最终获胜者获得 NBA 总冠军的比赛。全明星赛是由东西部两个地区的最佳球员组成两支球队举行一场比赛决定胜负的比赛。

1.2 数据解释

以 player_regular_season 数据集为例，该数据集收录了 1946-2004 年之间所有的球员数据。球员的基本数据包括工会、数据采样赛季、名、姓、所效力俱乐部、联赛种类六种，技术指标数据则包括篮板、助攻、得分等多项数据指标。

1.3 新添数据

outliers_MVP 中得到了每年的 outstanding_player 球员名单，将此汇总作为 MVP 预测的数据。

在预测比赛结果部分，题目中给出的数据中只有每个队伍每个赛季末的成绩汇总，但无法体现出两队伍之间的具体胜负关系。因此我们将队伍视为本问题中的个体，将队伍球员的指标处理为队伍的整体指标，为此引入了队伍胜负关系的数据集。

2、问题分析

问题一：选出杰出球员

根据给出的球员比赛数据，在所有球员中选出杰出球员。考虑到此种选拔应面对全体球员且各项数据在同一赛制下得出，因此选择常规赛球员数据作为训练数据，首先利用上场时间进行初步筛选，比赛时间不足的替补球员或者因伤中途推出的球员无此资格参与。之后挑选出每一赛季的球员，利用孤立森林算法选出 MVP 的预备队即 outstanding player 球员。

之后是 MVP 的选拔，用前一个程序得出的 outstanding player 球员数据作为 train_data，选用随机森林算法得出年度 MVP。此外，本程序中还额外加入了 2005-2006 年比赛数据，用以检验此程序的好坏。

问题二：预测比赛结果

出于简化问题考虑，此程序根据各场比赛的比分结果，并综合考虑主客场等不可避免的影响因素，利用已有的比赛结果的比赛本身的数据（球员实力、比赛分数数据等），选择 logistic 回归以及决策树算法得出结果。

- **Proposed method**

1、outstanding player&MVP

1.1 算法选择

孤立森林(Isolation Forest)算法选出与众不同的 outstanding player，随机森林(Random Forest)算法在选出的 outstanding player 球员中选拔出 MVP。

选出 outstanding_player，我们可以将杰出球员理解为，与绝大多数球员不同的更优秀的球员，故我们的目的就是找出这些少数的、与其他人相差较大的人，离群点检测恰好是选择出"与预期对象的行为差异较大的对象"，故选用离群点检测算法。

之所以抛弃了绝大多数人会选择的聚类算法，是因为本人认为，此数据维度过高，且个人战术、打球风格个性化程度较高，难以形成较典型的聚类，且不排除最终形成的聚类标准不是个人的优秀与否，而是球员的位置，例如中锋和中锋聚在一起，前锋和前锋聚在一起（因为数据相似度很高）。

关于 MVP 预测，是建立在已经选拔出的 outstanding player 上，直接根据选拔出的 MVP 候选人的数据来得出最后的 MVP，而并非选择离群点作为 MVP，这样，我们就可以在不适用离群点检测的小规模样本中较为精确地评选出适当的 MVP。

1.2 数据处理

首先要将各个球员的数据以赛季为指标进行合并。去除掉冗余数据，例如将球员的姓氏和名字连接到一起后去除掉其中一列。同时考虑到球员上场场次不同，导致最终的总分也可能不在一个可比的档次，于是将球员的各项数据指标处理为平均指标。例如将得分转化为分均得分、命中次数转化为命中率等。对数据进行标准化处理。

我们选用离群点检测中得到的 outstanding_player 数据集作为训练集，同时加入了 2005-2006 年赛季的年之后的小规模优秀球员的数据将这些数据作为测试集，以此来评选当年度的 MVP 球员。

在数据收集完成后，我们进行了手动算出在评选 MVP 过程中信息熵最大的五个因素，并根据这五个因素来构建随机森林评选 MVP。

1.3 算法介绍

1.3.1 孤立森林(Isolation Forest)

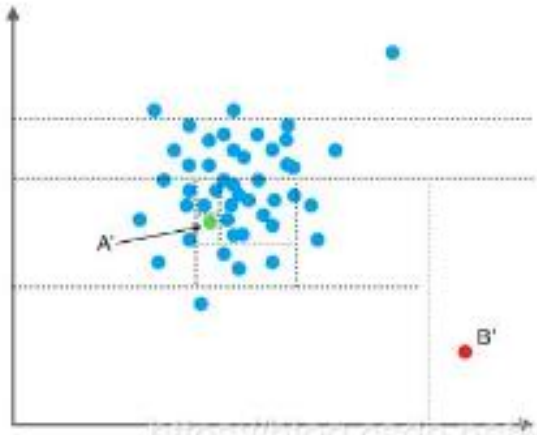
首先来基本了解孤立森林算法

Isolation Forest 孤立森林是一个基于 Ensemble 的快速异常检测方法，具有线性时间复杂度和高精准度，是符合大数据处理要求的 state-of-the-art 算法。

Isolation Forest 即不用定义数学模型也不需要有标记的训练。对于如何查找哪些点是否容易被孤立，Isolation Forest 使用了一套非常高效的策略。

先用一个简单的例子来说明 Isolation Forest 的基本想法。假设现在有两维数据。我们沿着两个坐标轴进行随机切分，尝试把下图中的点 A'和点 B'分别切分出来。我们先随机选择一个特征维度，在这个特征的最大值和最小值之间随机选择一个值，按照跟特征值的大小关系将数据进行左右切分。然后，在左右两组数据中，我们重复上述步骤，再随机的按某个特征维度的取值把数据进行细分，直到无法细分，即：只剩下一个数据点，或者剩下的数据全部相同。跟先前的例子类似，直观上，点 B'跟其他数据点比较疏离，可能只需要很少的几次操作

就可以将它细分出来；点 A'需要的切分次数可能会更多一些。



1.3.2 被选择的孤立森林算法在本程序中相比其他算法的优点

首先，观察数据特点：

1. 本数据都是无标签数据，在得出结论之前不知道球员是否为杰出球员
2. 球员数据具有个体特异性，随机性，因此无法为其建立起有效的数据模型，即数据不会按照某些固定的规律分布
3. 并非所有维度的数据都会对最终的评比结果产生影响，因此需要不考虑其中的一些重要性不大的数据

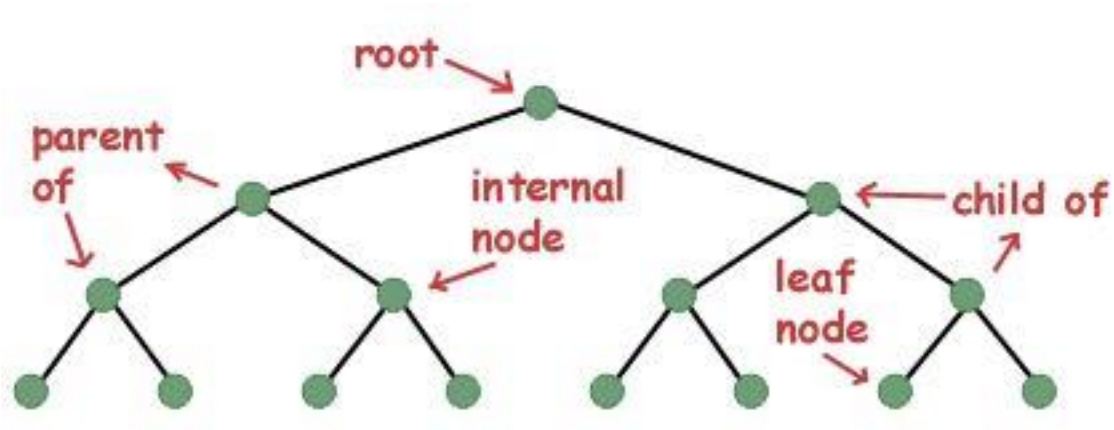
   首要选择的是无监督学习算法，其次在无监督学习算法中可供选择的算法中排除掉需要数据服从特定的概率分布的基于统计的异常检测算法，排除掉聚类算法，以及其他的基于密度的算法选择了孤立森林算法。

1.3.2 随机森林(Random Forest)

首先介绍随机森林算法

在介绍随机森林之前，先简要介绍决策树。

决策树是一种树形结构，其中每个内部节点表示一个属性上的测试，每个分支代表一个测试输出，每个叶节点代表一种类别。



随机森林就是通过集成学习的思想将多棵树集成的一种算法，它的基本单元是决策树，而它的本质属于机器学习的一大分支——集成学习（Ensemble Learning）方法。随机森林的名称中有两个关键词，一个是"随机"，一个就是"森林"。"森林"我们很好理解，一棵叫做树，那么成百上千棵就可以叫做森林了，这样的比喻还是很贴切的，其实这也是随机森林的主要思想--集成思想的体现。"随机"的含义我们会在下边部分讲到。

随机森林中有许多的分类树。要将一个输入样本进行分类，需要将输入样本输入到每棵树中进行分类。每棵决策树都是一个分类器，那么对于一个输入样本，N 棵树会有 N 个分类结果。而随机森林集成了所有的分类投票结果，将投票次数最多的类别指定为最终的输出。

2、Predict Game

2.1 算法选择

同时选择了 logitic 和决策树算法进行预估，比较两种算法的结果发现差距不大。

预测实际比赛的胜负需要用到大量的信息，包括一些难以用数据体现的球队的精神面貌，士气，比赛氛围，训练程度。为简化模型，此程序忽略这些因素，仅仅从比赛数据本身出发进行预测。

比赛结果只有两种，赢或输，因此容易想到我们本学期学过的 logistic 回归与决策树，因此同时采用这两种算法进行处理。

2.2 数据处理

在比赛结果预测的算法实现中，由于给出的数据集不够直观可靠，因此对其进行了较多处理。

将比赛结果抽象为了两支队伍之间的数据比较。队伍内球员在本赛季的表现即为球队的基本属性。在两个队伍比赛结果预测预测中，两个队伍不同数据的差值构成了一个 n 维的比赛特征向量，比赛结果（0 或 1）即为标签。那么我们就可以将比赛预测抽象为分类器的问题。

本例中的数据处理方式大体与 MVP 预测的数据处理方式相差不大。但是有一点值得提及。在 NBA 的球赛中，两队交战有主场客场之分，主场球队对球场设施以及观众氛围上有所加持，胜率会因此稍微提高。为了考虑到这个因素，我们将主场胜利队伍的数据赋上数相应的优势数据。

2.3 算法介绍

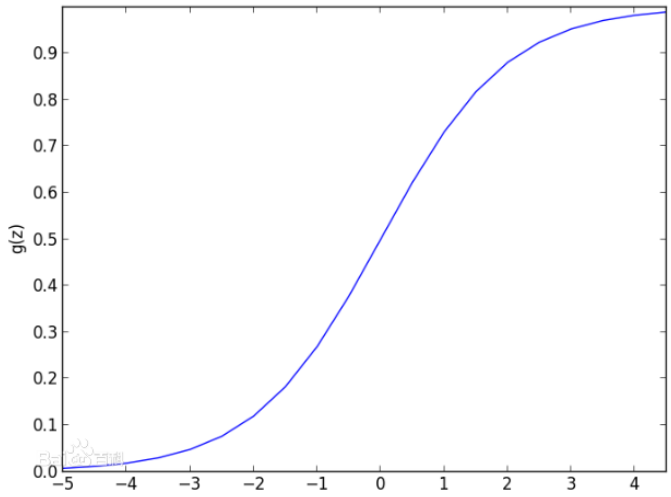
Logistic 算法介绍

logistic 回归又称罗杰斯蒂克回归分析，或逻辑回归分析。

Logistic 回归为概率型非线性回归模型，是研究分类观察结果 y 与一些影响因素 x（单变量，多变量都可以）之间关系的一种多变量的分析方法 。

$$y = \frac{1}{1 + e^{\theta x}}$$

函数图像如下：



它的分布曲线，在中心附近增长很快，而在两端增长很慢。这就是说，若以概率 0.5（中心点 μ 处的分布概率）为分界点，大于 μ 的点 Z 为一类，小于 μ 的点为另一类，那么，我们能很好很快地把中心点附近的数据分类。

决策树算法介绍

见上文

- ## 机器学习平台

本次的算法实现除了代码环节之外，还在机器学习平台上进行了框图的搭建模拟。

由于平台仍有待完善，且报错诸多，因此我选择了编写代码后再按照代码中的逻辑关系搭建框图，一共搭建了三个框图，基本实现了题目所需的功能。导出的 gph 文件也一起放在了压缩包中。

对于离散点我选用了孤立森林算法，由于平台上并无此节点，因此自己构建了实现鼓励森林算法的节点，除此之外，数据处理与检测的节点也均为自己构建。

我在测试过程中遇到了很多问题，例如容器构建陷入长时间死循环没有任何反应。能不能正常的运行只靠多刷新几次页面。例如不能实现节点内部的函数调用，因此

只能将代码先进行整合，写进一个大函数中。此外，在编辑一个节点后，不能进行二次编辑甚至不能查看源代码内容，造成了编译过程的一些不便。

总体而言，如果此平台的功能能够更加完善，性能得到更好的提升，将为机器学习的学习与应用带来极大遍历，但目前仍需进一步的提升与完善。

- ## Result

  预测出的 2004-2005 赛季 outstanding_player 球员名单如下：

  Tim Duncan
  Kevin Garnett
  Steven Hunter
  Zydrunas Ilgauskas
  Allen Iverson
  LeBron James
  Yao Ming
  Shaquille O'Neal
  Kurt Thomas
  Ben Wallace

  选取离群点检测中得到的历年 outstanding_player 数据集作为训练集，手动算出在评选 MVP 过程中信息熵最大的五个因素，并根据这五个因素来构建随机森林评选当年年度常规赛 MVP。

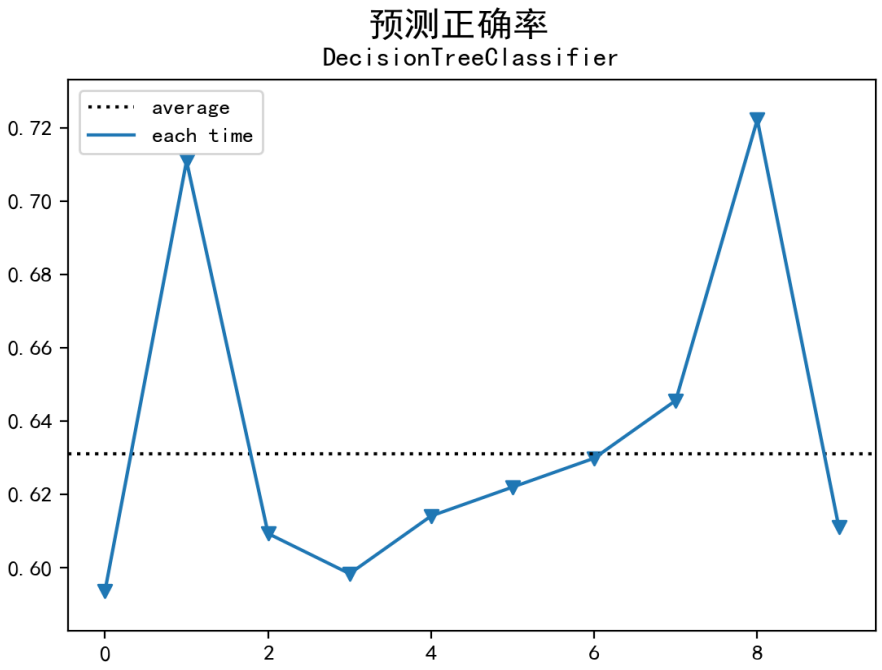
  预测出的 2004-2005 赛季常规赛 MVP 为

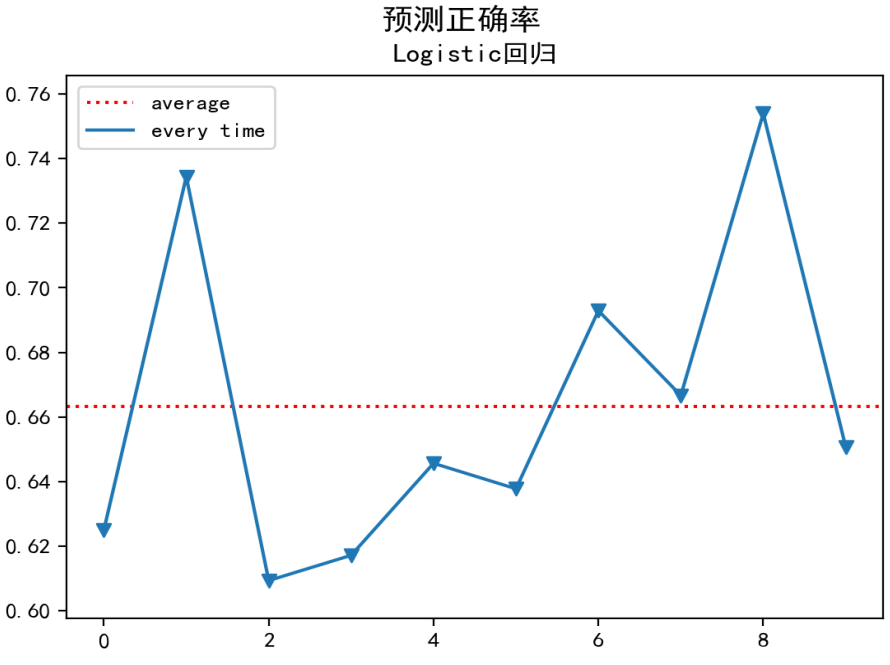
  Shaquille O'Neal

  同时，收集了 2005-2006 年球员的数据，预测出的 MVP 为

  Dirk Nowitzki

  比赛结果预测：





可以看出两者结果相差不大。

- ## Conclusions

1、不足之处

本程序仅在数据的角度出发，没有综合考虑到其他因素，例如所在球员打的位置，球队赢的场次，个人打球风格，是否进场打黑球导致风评不好等等，导致最终的结果正确率不高，得出的结论也和当年的 MVP 获选者不同，但是可以肯定的是选出的球员都是十分优秀的球员，即便不是 MVP 也是 NBA 中的佼佼者。

在预测比赛中，将球员的作用一以概之，没有考虑到各个位置的球员的作用的不同以及对比赛影响结果比例因素的分布，更重要的是没有考虑到教练的指导作用，因此最后的结果不如人意。

总之，尽管此程序有诸多不足，但基本上在能利用的数据集的基础上达到了最佳，我们期待将其进一步改善，考虑到上述的干扰因素。