软件工程结课报告

组49

整体系统和数据分析部分

## 一、项目名称：

NBAlabala数据分析系统

## 二、项目目标：

开发功能实用可靠、人机交互友好，能够真正交付现实用户使用的NBA数据分析系统。

## 三、主要功能：

迭代一、迭代二、迭代三所要求的全部功能。

基于球员、球队、比赛数据的多角度查询和展示。

对球员数据的多角度分析和展现。

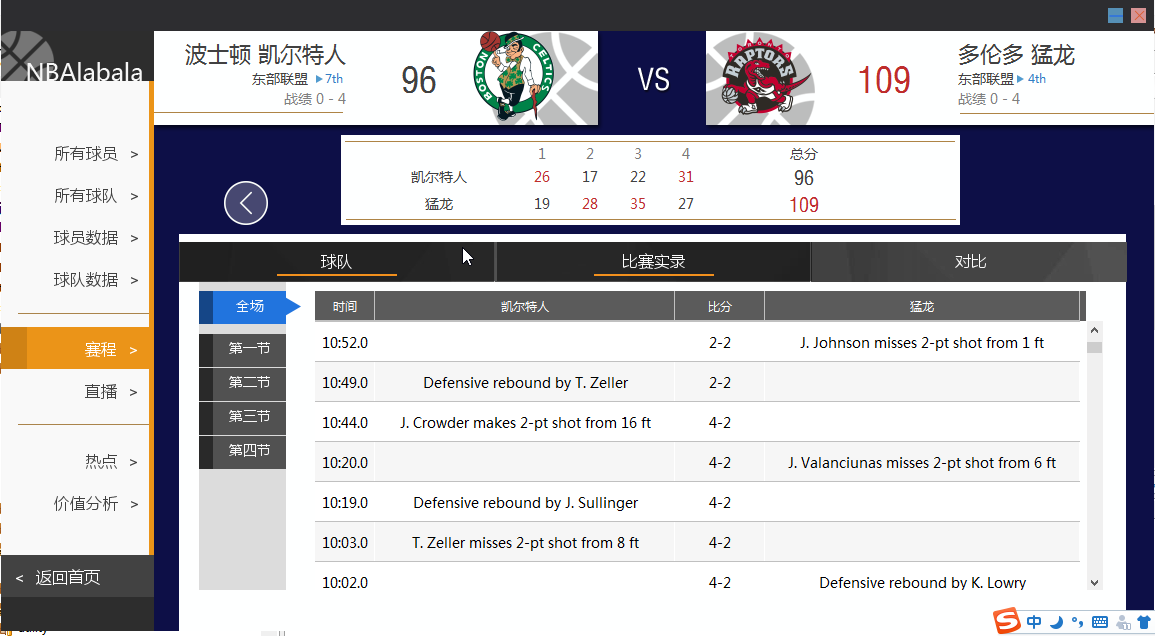
数据同步和实时直播。

用户可以根据需要进行个性化设置。

## 四、主要亮点：

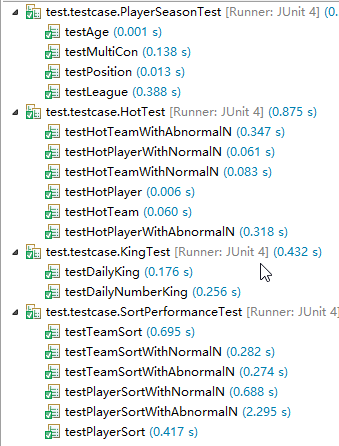
### 在网络环境下获取历史比赛数据文字实录

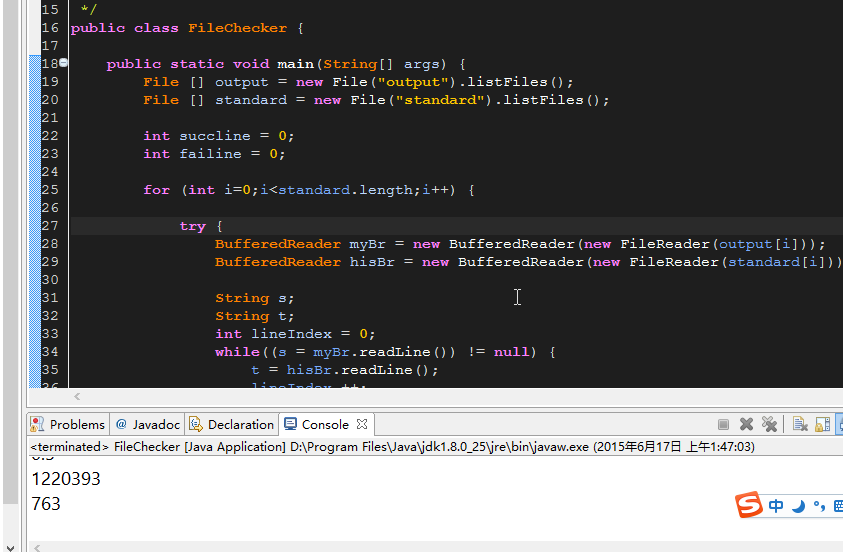
NBA比赛每一场的文字实录大概在30KB左右，每赛季将有40MB的文字实录，将几十个赛季数GB的文字数据扔给用户是不现实的，所以我们提供了网络环境下获取历史比赛数据文字实录的功能，用户边浏览过往比赛，程序边爬取该场比赛的文字实录并展现给用户。



### 数据正确性的严苛把控

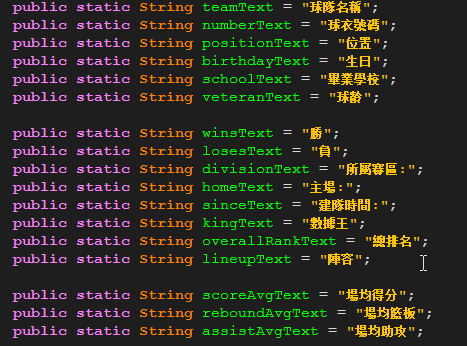
对于球员效率、真实命中率、球队进攻防守效率等数据，我们的获取方法是：从网站上获取球员和球队的原始比赛数据，使用公开的公式进行计算，核对计算结果，保证计算逻辑正确性，保留测试用例和标准数据。之后每次变更数据源或者改变逻辑，均执行回归测试以保证其正确性。（如图，回归测试结果表明：122万行输出结果中有763行不相符，考虑到浮点误差，这说明没有出错）





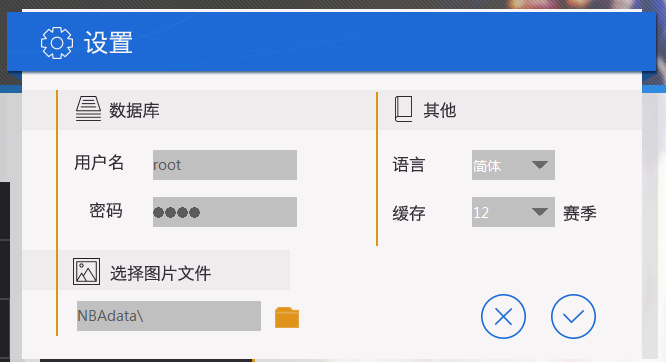
### 多语言和良好的可扩展性

程序内置简体中文、繁体中文、英文，方便不同用户。事实上，如果要加入法语或者西班牙语的话，只要用谷歌翻译，修改一个类中的字符串，就能在几分钟之内增加一个拙劣的语言版本。如果请一位粗通该语种的同学来增加一个老外可用的语言版本，大概用不了一小时的工作量。——真切体会到了常量集中管理的好处。



### 性能和缓存的个性化设置

尽管这只是一个小程序，我们依然认为考虑性能是一种好的态度。我们假定当一个用户读取了一个赛季的相关数据的时候，他很有可能在未来一段时间内多次访问这个赛季的数据。所以我们设置了缓存机制，并允许用户根据自身对程序内存消耗和响应时间的要求设置缓存赛季数量。当新的赛季被读入时，最长时间没被访问过的赛季将会被内存中淘汰。



## 五、数据分析

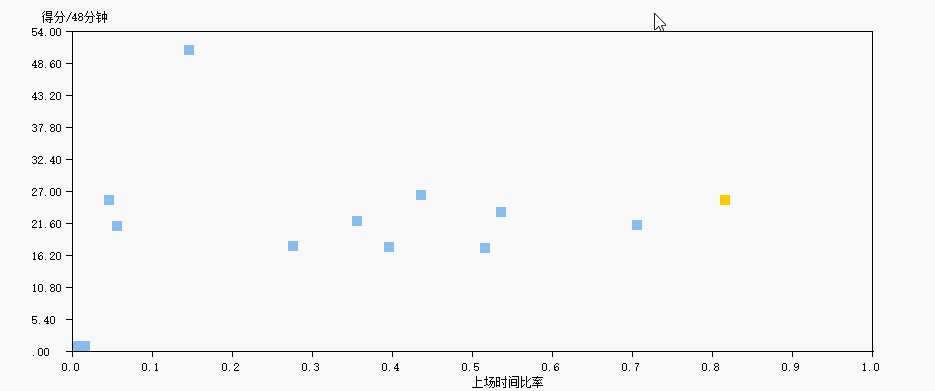
### 决胜时刻

Clutch Time指的是第四小节最后五分钟或加时赛且两队比分相差不超过5分时，如果一个球员在这种时刻上场时间多，说明他得到了整个球队的信任。如果一个球员此时得分多，可以认为他没有辜负球队的期望。

使用二维表，是因为我们希望在一张图上详细的展示出球员与他的队友横向对比的情况，避免球队整体强弱因素对我们判断一个球员的价值造成干扰。

横坐标表示球员上场时间比率（假设一个球员横坐标值为1，就表示该球队所有Clutch Time他都上场了），纵坐标表示得分（为了更便于感受得分的多少，换算成了每48分钟的数值）。

这样展示的效果是：用户一眼看上去，就知道自己选择的球员在球队中是属于力挽狂澜的人物还是板凳选手。

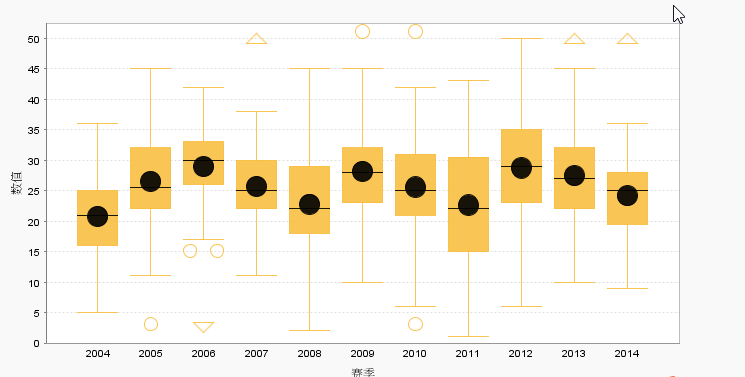


### 2、生涯数据

我们试图从一个较为广阔的视角展现一个球员的生涯数据，使用户对这个球员的数据水平快速得到一个大致的认识。

由于这种需要，箱须图就成了我们的一种自然的选择。因为箱须图既能表现出不同组数据之间的差异，又能表现出同组数据内部的离散情况。

这种图表的效果是：用户可以从这张图上得知，用户的某项数据均值是随时间变高还是变低，也可以得知一个赛季内他的发挥的稳定情况。

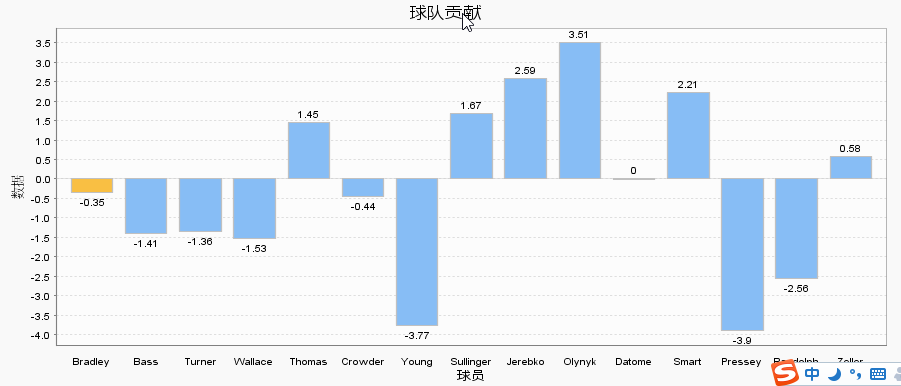


### 球队贡献

这个部分展现的是球员对所属球队的表现作出了什么样的贡献，使用的数据有参赛场数、参赛时间、ORPM、DRPM、RPM、WAR。以下是ESPN对这四项数据的解释。关于WAR的详细内容可见References、

* **ORPM:** Player's estimated on-court impact on team offensive performance, measured in points scored per 100 offensive possessions
* **DRPM:** Player's estimated on-court impact on team defensive performance, measured in points allowed per 100 defensive possessions
* **RPM:** Player's estimated on-court impact on team performance, measured in net point differential per 100 offensive and defensive possessions. RPM takes into account teammates, opponents and additional factors
* **WAR:** The estimated number of team wins attributable to each player, based on RPM

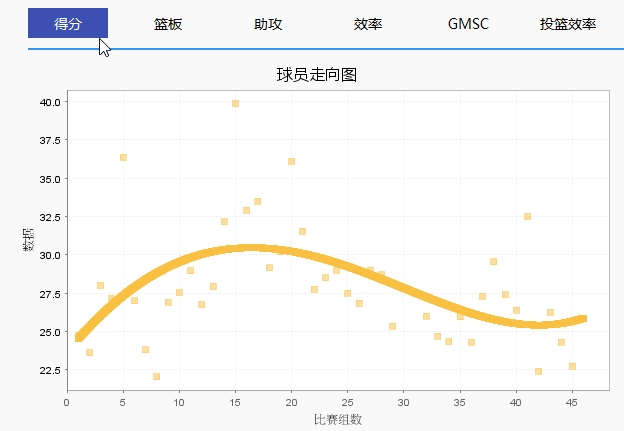
我们选用条形图，是为了既显示出数据的数值大小，又便于展现该球员和他的队友的对比，判断谁是球队的中坚力量。

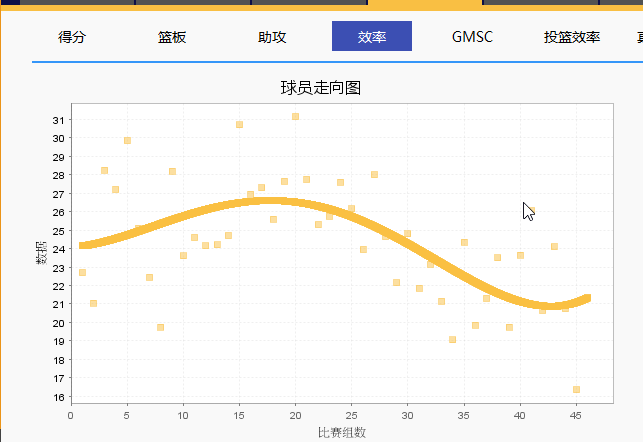


### 前程展望

如果我是一名球队教练的话，我更希望知道球员的未来发展前程如何。我们使用基于矩阵运算（Jama）的外部包（Jamlib）进行多项式拟合分析，对球员未来的数据进行预测。（详情可见References）

从 拟合效果来看是比较不错的：

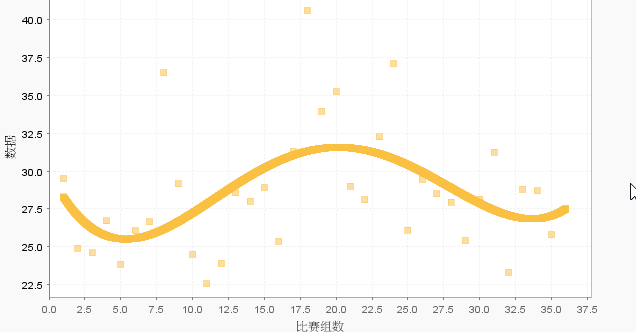




从预测效果上来看，以Kobe Bryant为例：

我们以2010-2011赛季之前的数据为样本，对Kobe的未来得分进行预测：





预测结果是27.5分。

Kobe2011-12赛季场均得分为27.9分：

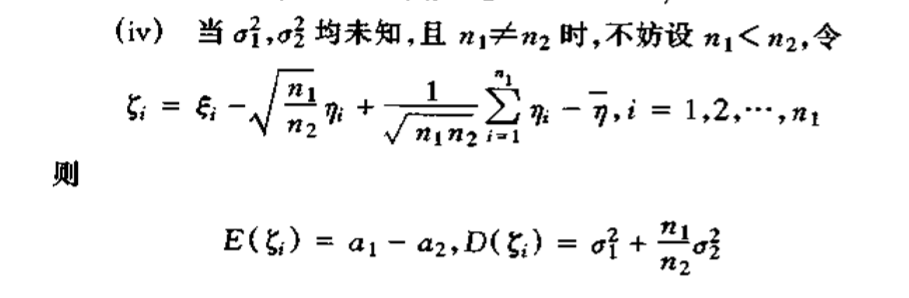


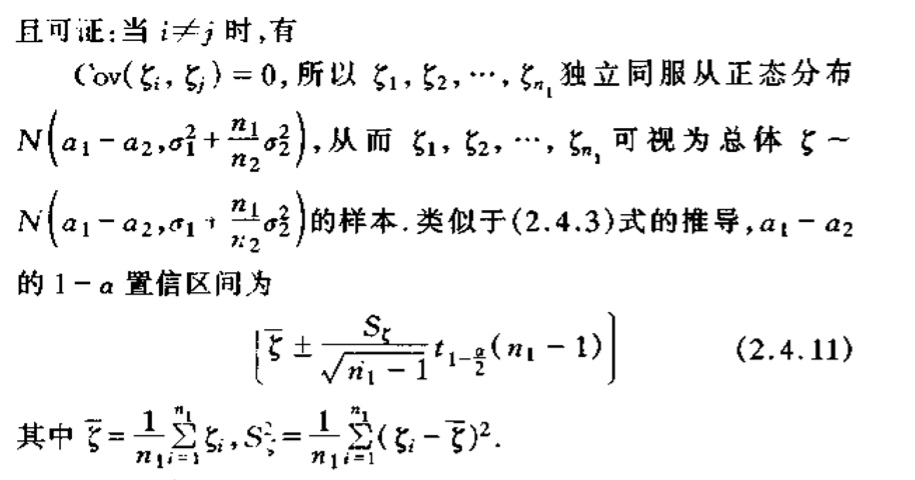
看起来我们的预测是有一定可靠性的。

### 转会分析

我们期望得知球员转会后发生了什么，并不是每个人都能像哈登一样一转成名的。所以我们试图运用t分布来检验球员转会前后数据的期望有无显著差异。但是在这一应用场景下，我们既不能假设样本容量足够大，又不能假设转会前后方差相等，所以软统教材上并没有提供可用的检验方法。

幸运的是，我们在孙荣桓《应用数理统计》上找到了解决方案。





有了理论指导以后，剩下的事情就变得简单了。

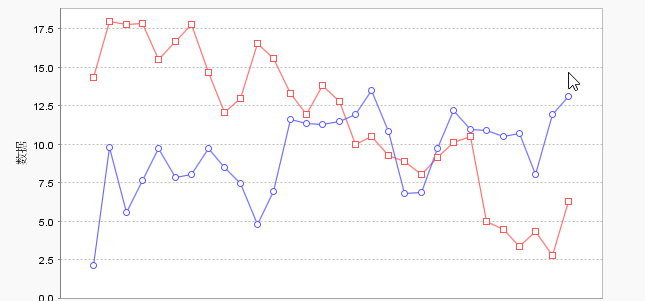
我们选用折线图，是因为我们希望展示同一球员的某项数据随时间的变化，而且样本容量可能较大也可能较小，这种情况下用折线图是自然的选择。

最终效果是不错的，至少我们的程序也认为詹姆斯哈登转会火箭后的大爆发是惊人的。

### 球员对比

我们衡量一个球员有多出色，经常拿其他球员跟他对比。然而球员的发挥是不稳定的，我们肯定不能拿两个均值做一个减法就声称球员A比球员B优秀。

我们采用了求置信区间的方法，来估计两个球员某项数据到底有没有明显区别，谁比谁优秀多少。

由于要展现的数据是随时间变化的，且样本容量可能较大也可能较小，所以使用折线图就成了我们自然的选择。

上图是一位处于上升期的新秀球员和一位正在下滑的球员的比较，程序告诉我们，老将依然是比较厉害的。

## REFERENCES

1、《应用数理统计》 孙荣桓

2、两个正态总体均值之差的区间估计解析 邱瑾 浙江财经学院 数学与统计学院

3、Java in Science: Data Interpolation and Extrapolation Using Numerical Methods of Polynomial Fittings <http://www.developer.com/tech/article.php/762441/Java-in-Science-Data-Interpolation-and-Extrapolation-Using-Numerical-Methods-of-Polynomial-Fittings-Part-1.htm>

4、Wins Above Replacement John Wiley & Sons