

《数据挖掘课程设计》报告

**2020/2021(2)**



报告题目 基于多元线性回归和K均值聚类分析的

足球赛事输赢预测

学生班级 数据应用1901

**计算机科学与技术学院**

**摘要**

本文从澳客网爬取了10年内英超足球赛事数据，包括包括但不限于个足球队，每一年总得分情况净胜球情况的数据，以及每一场比赛中参赛队伍的各种基本信息，包括比分队员身价，队员平均身高，队员平均年龄等信息，以及每一场比赛的综合99家平均赔率和个别具有代表性的赔率等信息；由于澳客网由复杂的反爬机制和ip屏蔽技术，花了很长时间，尝试了多种方法才完成的数据爬取，数据爬取有一定难度和工作量。综合所有数据，提出了两个假设，从两个方向解决问题。

对所有数据进行了浏览和预处理，在回归中对球队数据进行特征选择和预处理。在对球队实力的多元线性回归中，采用了置信区间分析等方法，在降维后得到了较好的结果，基本能够根据基本已知信息模拟球队实力。在取得队伍实力参考值后利用k最近邻对比赛结果进行了预测。

基于分析结果尝试了不同预处理方法和降维，在通过聚类分析预测具体样本比赛结果的过程中，利用交叉验证和留出法的方法，分别确定了k均值中k的取值，并根据簇中样本的分布，尝试了连续化处理。在此基础上提出了一些改进意见和展望。算法并调参后达到了较好的分类效果。

1. **问题分析与描述**

如今，足球博彩的形式多种多样，参与博彩的平台和公司也各有不同。竞猜的内容包括足球比赛中主队的输赢，或者平局，以及具体的比分，不一而足。

考虑到博彩的性质，我们主要考虑在资金总量一定的情况下，如何通过数据分析的方法尽可能的多获利或减少亏损。考虑到我们的主要是基于澳客网的数据进行分析。该网站主要提供的博彩形式是对足球赛输赢的预测，而不关注具体的比分，因此我的，预测主要着眼于对输赢赢结果的预测，并在此基础上尽可能的多获利。本文以英超足球比赛作为研究对象。众所周知，球队存在实力差距，因此，博彩公司给双方提供的赔率也是不一样的，如果仅考虑预测胜率的话，显然并不能做到最大盈利。在热门和冷门中做出选择，这是本文研究的重点。

比赛的赔率是由博彩公司给定的，博彩公司已经预先对各支队伍的实力进行了判断，但我们可以根据往年的数据和今年的实际情况，对队伍的实力做出我们的判断，并根据两队的闲的实力对比和种种厂商因素对输赢作出自己的判断，结合赔率，也就是庄家的判断，制定我们的投资方案。因此，如何借助往年数据和今年实际情况，以及场上的因素，对队伍实力和队伍间的实力对比做出判断，这也是本文要研究的主题之一。

考虑到队伍实力和队伍间的实力对比和输赢并不是有着直接关系的，还存在种种概率性的因素，队伍实力和队伍间的实力对比与比赛胜负结果的映射关系，也是本文要研究的内容之一。

1. **研究背景**

体育赛事不仅仅是一项运动的项目，还蕴含了一个国家的体育文化。近年来，体育赛事的统计工作也越来越受到重视。然而，体育赛事的比赛结果具有不确定性，且影响比赛胜负的因素也复杂而繁多，尽管球队的实力是影响比赛结果的重要因素，但球员疲劳和伤病、临场发挥、主客场条件、心理因素，甚至是运气等因素都会对比赛结果产生相当程度上的影响，这些因素的不可预测性，大大增加了体育比赛结果的预测难度。

如今，对体育赛事的分析和预测已经成为了一个重要的研究领域。一方面，球队的建设离不开经济，对一个球队实力的有效分析和评估能够更好地反映投入的成效，从而帮助经营者对资源的投入和分配进行合理的规划；另一方面，体育赛事，尤其是足球比赛的结果、进球得分等，可供彩民们进行博彩，从而使彩民有机会获得一定的收益。体育赛事的分析对社会、对个人的观念都具有一定的现实指导意义。

1. **研究现状**

体育赛事的竞争性和观赏性，引起了国内外各界的广泛关注。同时，体育赛事的分析工作也越来越受到重视。例如，Maher[2]应用泊松分布分析了每支球队的得分情况与球队的进攻和防守的优势有关。Dixon和Coles[3]对泊松分布进行了调整，介绍了结果为0-0，0-1，1-0，和1-1的概率模型。Karlis和Ntzoufras[4]应用二元泊松分布和相关参数分析了球队的进球数目，并扩展了该模型从而提高了平局的概率。Rue和Salvesen[5]应用动态扩展模型对球队实力进行了分析。Knorr-Held[6]应用Bayes估计分析了每支球队的攻击和防守强度参数。Crowder等人[7]提出了球队攻击能力和防守能力的自回归模型。Clarke和Norman[8]提出了两支球队的实力差异之间的线性回归。Goddar和Asimakopoulos[9]使用有序的概率模型来确定协变量，例如影响本场比赛的结果，比赛的重要性、犯规、黄牌和红牌。Koning[10]采用有序概率模型对比赛进行了分析。Kuk[11]介绍了球队的主场实力参数和客场实力参数。Barry和Hartigan[12]提出了球队实力参数的动态扩展，他们应用先验分布的选择模型对适时变化的球队实力进行了分析。Fahrmeir和Tutz[13]应用经验Bayes估计分析了球队实力变化的可能性。RDBaker和IGMchale[14]应用了动态的成对比较模型，对1968年到2014年男子职业网球公开赛的冠军进行了排名。McHale和Scarf[1]将二元离散分布运用到球队射门次数，并用边际值解释了球队射中和失球的负相关。

Mark J.Dixon 和 Stuart G.Coles[15]通过考虑数据结构的复杂性和球队表现的动态性，利用 Poisson 回归模型分析 1992 年到1995 年英超足球联赛和杯赛数据．通过极大似然估计来估计分析如何获得一个更有利的足球博彩策略．Ian McHale 和 Phil Scarf[4]构造一个新的双变量离散分布即双变量 Poisson联合分布来拟合英超足球数据，将协变量和足球比赛中的传球数和拦截数等解释变量联结起来，其结果显示更多的传球和拦截有助于球队更好的发挥，能有更多的机会射门．Ke Shen[16]考虑数据集的相关性和可变性，提出了利用双变量 Poisson 模型处理二维离散数据，并预测 2013 至 2014 中国 CBA 和 2012 至 2013 美国 NBA 的常规赛和季后赛各球队能力．Desmond Mwembe,Lizwe Sibanda 和 Ndava Constantine[17]利用双变量 Poisson 模型分析和计算津巴布韦总理足球联赛数据，通过蒙特卡洛模拟和对数线性模型估计相关系数和得分强度．构建一个更有利的足球博彩策略。

1. **数据获取**

**4.1 数据选择**

首先考虑需要获取的数据。如图，我们需要近五到十年来英超比赛每一场的结果，阵容和赔率的情况，以及网站提供的各个队伍本年度的总体情况。



图1 具体比分和赔率情况



图2 阵容情况



图3 各队年度总体情况1



图4 各队年度总体情况2

4.2 数据获取

本文利用爬虫获取数据。首先尝试采用浏览器代理的url请求方式爬取数据。

运行爬虫时遇到问题，无法获取信息。详细信息见附录1。

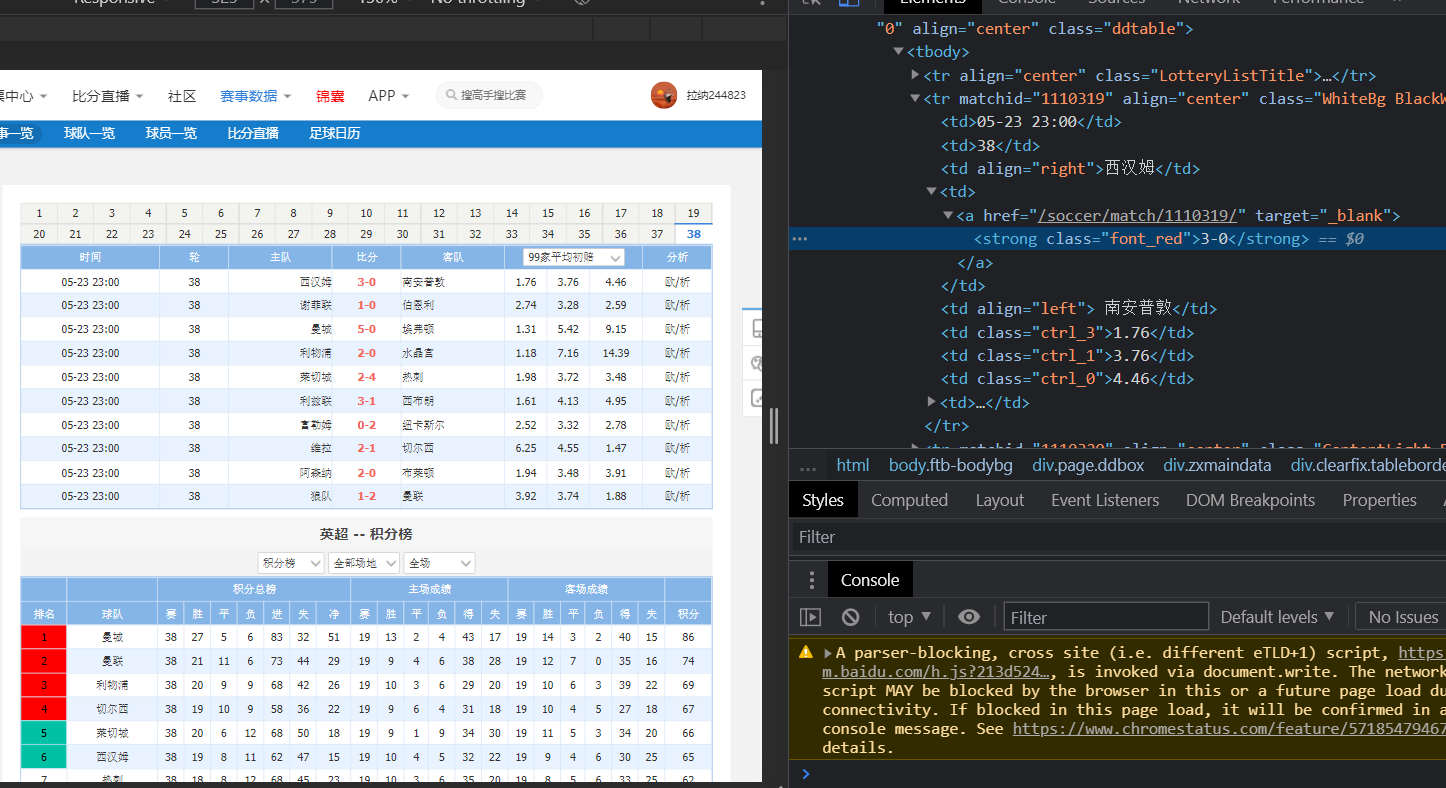


图5 网页检查详细信息



图6 网页爬取失败

猜想是由于网站采用了反爬虫机制，尝试利用超链网页数据，对超链网页数据进行解码，以验证猜想。详细信息见附录2。

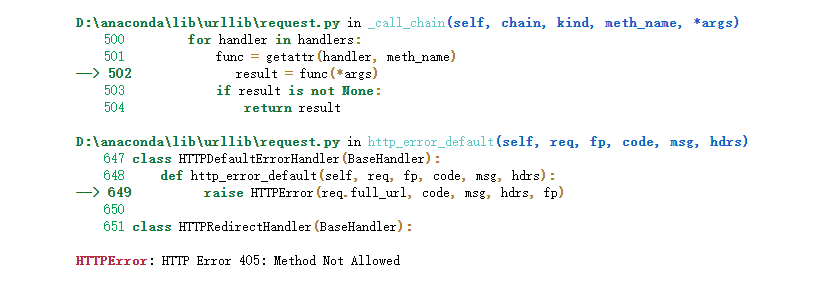


图7 网页爬取失败的研究

因此，爬取方法改为get\_table，并用模拟人操作的方式避免被网站屏蔽ip地址。详细代码见附录3-7。其中保存表格的方法采用的是模拟鼠标点击的动作。



图8 网页爬取成功

值得注意的是，本方法使用时需要人为点击网页进行刷新，并且在校园网网络环境下无法实现。

1. **数据预处理**

**5.1 异常分析**

首先观察数据的大体情况。



图9 数据大体情况

观察可以发现，1930个样本量不符合循环赛实际情况，20支队伍进行循环赛，理论上讲应该有1900场比赛。因此先删去由于时间异常导致的错误样本第二不对，没有比分等具体信息的样本进行筛选，发现有两条样本属于异常情况，对异常情况的样本，采取删除处理。最终获得1898个样本。

具体代码见附录7。



图10 完成预处理的样本

**5.1 数据有效性分析**

对比赛输赢有直接影响的因素，是队伍的实力和队伍间实力的对比。考虑到获取到的数据中和队伍实力有关系的数据包括了以往比赛的实际表现，队伍当前的信息以及当前比赛队伍的赔率，本文从这里着手展开分析。

首先考虑全部购买热门时能够赚到的钱。



图 11 全部购买热门时能够赚到的钱

上图数据中，一共购买了1898张热票，每张价值一元，如果预测正确，将获利的金额等于赔率。总共消耗本金1898元。1056表示全部购买热门时，能够预测正确的数量；图片中1844.06表示全部购买热门时能够获得的利润（不包括本金）。可以发现，即使全部购买热门能够命中的数量，也只有大约60%，如果再考虑到热门的队伍，赔率通常比较低，最终可以发现，如果全部购买热门，一定是亏本的。这也符合事实规律。一方面出名简单的购买热门是不可行的方案，另一方面也说明即使是庄家做出的预测也不够准确，我们需要采用更加优秀的方法去预测队伍的实力和比赛的胜负情况。具体代码见附录8。

为了方便后续处理，本文将对赔率的三个数值进行进一步的规范化，将盈利指数也就是公司的凯利指数规格化到一，保存为home,draw,away三个值。具体来说，将进行如下处理：

其中k为凯利指数。x,y,z为源数据中主场赔率，平局赔率和客场赔率。x’,y’,z’为规格化后的量，记为ome,draw,away。也即使下市成立：

此外，由于数据源中比分为字符串形式，为方便后续处理，对其进行数字分离，分别将两队的得分保存为score1和score2。



图 12 预处理最终结果

**六、系统构架**

**6.1 假设提出**

前文已经发现，即使是博彩公司，也不能对单场比赛的胜负做出极其精准的预测。博彩公司只能队两个队伍的实力做出基本的预测，具体的比赛胜负仍然存在非常大的随机性。因此，博彩公司的盈利依赖于起其庞大的规模带来的大数效应。即在凯利指数小于1的情况下，只要比赛数量足够多，参与博彩的人数和资金足够多，博彩公司就一定会盈利。

博彩公司只能队两个队伍的实力做出基本的预测，具体的比赛胜负仍然存在非常大的随机性。可以认为，即使是拥有大量专业分析人员和过往比赛数据的博彩公司也不能精准的预测比赛结果。对于这一问题，我们对其原因进行分析，并提出以下两个假设。

1. 博彩公司对于足球队的实力的算法不够优秀，不能准确的反应球队实力，其开出的赔率不能很好的很精准的反映两支球队间的实力对比。

2. 博彩公司对于足球队的实力的算法相当优秀，但是足球比赛存在巨大的不确定因素，以弱胜强的概率并不小。

以上两种情况需要不同的处理手段。

1. 对于假设一，我们考虑利用现有的数据，回归分析得出比赛前，每支球队的相对实力值的计算公式；如此以来，我们将能够得到每一场比赛前两只队伍的实力对比值；对于每一场结果未知的比赛，我们借助已知的实力对比值，利用K最近邻算法获得队该场比赛结果的预测。

2. 对于假设二，我们大可以相信博彩公司的赔率，利用聚类的方法，直接得到对比赛的预测。

**6.2 对于假设一的模型建立**

对于假设一，本文的处理流程如下：我们考虑利用现有的数据，回归分析得出比赛前，每支球队的相对实力值的计算公式；如此以来，我们将能够得到每一场比赛前两只队伍的实力对比值；对于每一场结果未知的比赛，我们借助已知的实力对比值，利用K最近邻算法获得队该场比赛结果的预测。

具体流程如下图所示：

图 13 假设一的模型

**6.3 对于假设一的模型的求解**

按照上述流程，在完成数据集的分割后，我们首先需要找出每一个队伍的具体信息进行主成分分析。然而，由于数据爬取的局限性，我们取“队伍平均身价” “平均年龄”“平均身高”三个量作为主成分开始分析，具体代码间附录9。

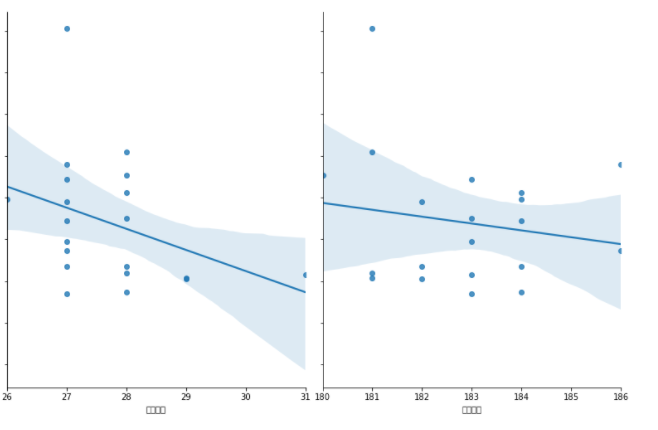


图14 回归中的置信度分析

本文得出的系数矩阵如下：

B-1=[ 0.00077575 -0.00745624 0.00244002]

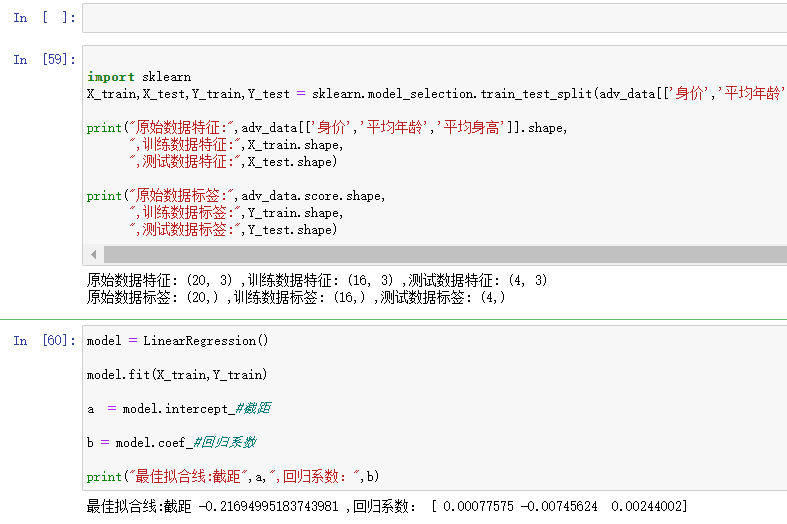


图15 回归结果分析

则每个队伍的相对实力公式如下：

其中X为由该队伍的“队伍平均身价” “平均年龄”“平均身高”三个量组成的向量。

随后，本文用交叉验证法，确定K最近邻的取K=13时结果最佳，具体代码间附录10。（实际结果显示，K取11或13或15或17，结果相差不大，可能还和训练集本身的大小和数据的密度有关）

结果显示，在测试集的190个样本中，命中72个，盈利173.28元，净利润-16.72元。可以说，结果并不理想。

**6.4 对于假设一的模型的评价**

由于数据源的限制，本文获得的目标函数（队伍相对实力）并不理想，可以说和博彩公司的结果还有一定的距离，因此，本文认为，假设一无法成立，即使成立，在笔者的水平下也无法妥善处理。因此着重考虑假设2。

**6.5 对于假设二的模型建立**

对于假设二，我们大可以相信博彩公司的赔率，利用聚类的方法，得到对比赛的预测。

考虑到样本的数据特征，我们这里采用余弦距离作为衡量样本与样本间的距离，以及与簇心间距离的标准。由于在完成聚类后仍然有许多种方案去代表每一个簇的结果，因此，本文采用循序渐进的方法，对模型进行改进。

在假设一的验证中，我们已经可以发现博彩公司对足球，对实力的预测是比较准确的，锦屏县有的数据无法得到比国彩公司更准确的对实力的预测，因此，我们将直接采用样本中和博彩公司直接相关的胜负及平局的赔率，作为参考因素，也就是构成余弦距离的向量。

若设样本中每一场比赛的x,y,z为源数据中主场赔率，平局赔率和客场赔率。x’,y’,z’为规格化后的量，记为ome,draw,away。

则两个样本的余弦距离为：

其中

我们采用K均值聚类的方法找出样本的簇。

实验流程如下。

图16 假设二实验流程

利用前文中已经整理和切分好的数据，我们开始实验。

首先执行步骤1，对于每一个测试样例，首先找出离他最近的簇，在簇中找到占最大比例的结果（胜，平或败），以此作为对该样本的结果预测。例如，在某一个样本所在簇中，胜的结果占多数，则以此作为对该样本的预测。具体代码详见附录11。结果如下：



图17 简单聚类结果

图中的数字752表示，在1898次比赛中预测对了752次，总收入1742.42元，净亏损百余元。

可以发现，即使是在全集上，结果也并不理想，甚至不如全部购买热门时的收入。下图为其交叉验证的结果：



图18 简单聚类交叉验证结果

图中的数字350表示，在1898次比赛中预测对了350次，总收入1330.28元，净亏损五百余元。

显然这个结果无法令人满意，下面进行第二步。取消条件3，用交叉验证的方法，找出最佳的K取值。

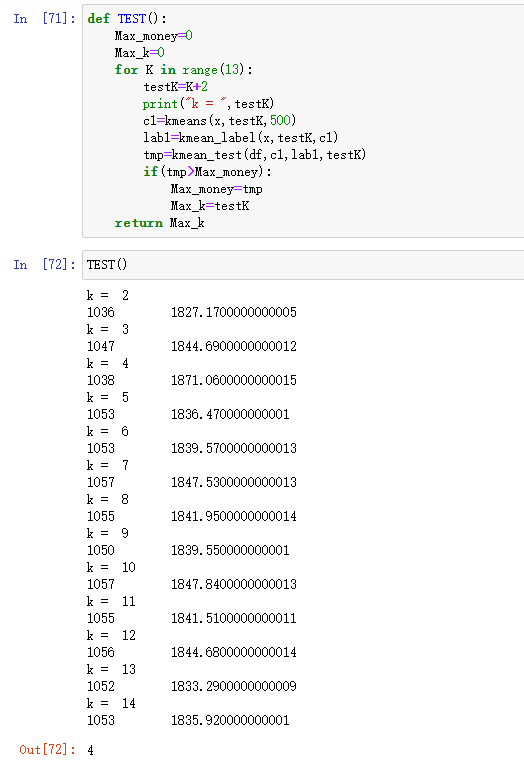


图19 不同K的取值结果

可以发现在这种情况下得到的结果比上文要好，当k取4时命中率已经能够超过最初全选热门时的命中率了，并且收入也高于全选热门时的收入了。

以下是交叉验证的结果。具体代码见附录13。

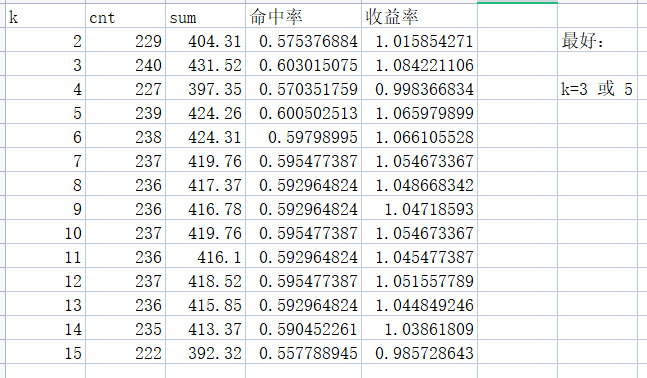


图20 交叉验证条件下不同K的取值结果

接着进行下一步，例，首先找出离他最近的簇，再按簇中三种结果的分布情况，做出预测。具体代码见附录14。

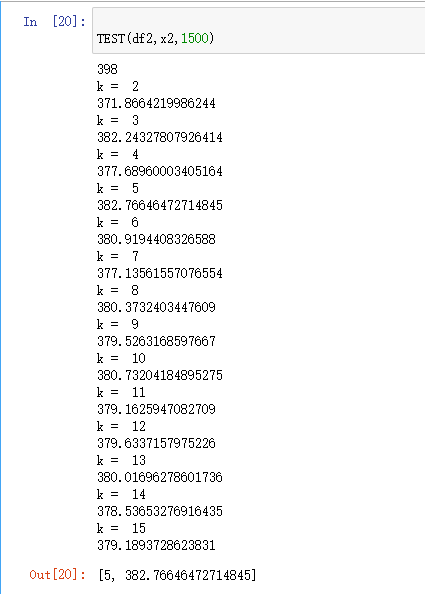


图21 交叉验证条件下连续化的结果

具体来说，对于某一样本我们首先找到他所在的簇，在初中找到每一种结果的占比，例如主场胜利的样本占比为40%，平局的样本占比为20%，客场胜利的样本为40%，那么对于该测试样本，我们的投资金额分配为4：2：4。

可以发现在连续化的条件下，测试结果非但没有变好，反而变得更加不良。

**七、结果对比**

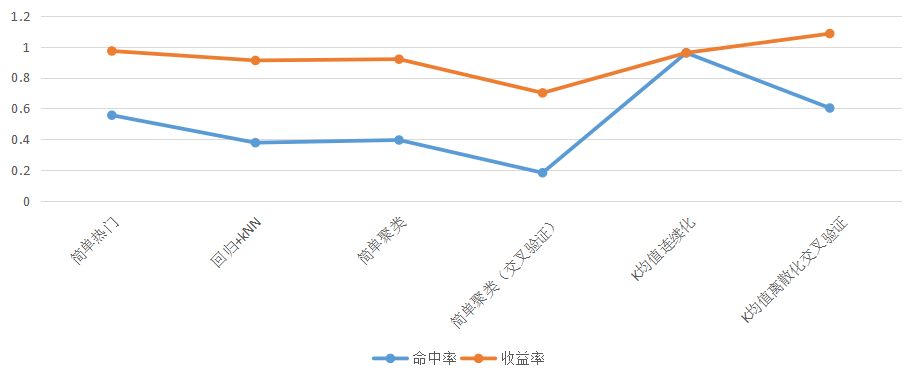


图22 总体结果呈现

本文采用了两个方向的思路，分别对数据进行了处理分析。

如上图所示，在回归分析的结果上进行预测，利用k最近邻算法得到的结果并不理想，因此舍弃第一种假设，尝试第二种模型。

在第二种假设中，利用k均值算法进行聚类。通过调整聚类后的决策，并利用交叉验证的方式，选择最佳的k取值，最终使正确率达到了60%以上，并实现在测试集中盈利赢利，利润达到8%，相当理想。

**八、总结与展望**

在对队伍实力进行多元线性的回归的模型中，样本属性值偏少，或许可以增加爬取的样本属性种类，再通过主成分分析后重新进行线性回归，或许会有更好的结果。此外，对于回归目标函数值的选择也有待改进。

在k均值聚类的距离选择中，可以进行更多尝试，可以尝试选择欧式距离，或许会有更好的结果。

在k均值聚类分析的连续化处理过程后，可以增加一个风险参数s，该参数决定投资的重点是偏向于低风险，高稳定度，但回报率低的样本或是投资集中于高风险，高收益，低稳定性的样本，具体算法模型还有待改进。

**九、参考文献**

* [1] Mchale I, Scarf P. Modelling soccer matches using bivariate discrete distributions
* with general dependence structure[J]. Statistica Neerlandica, 2007, 61(61):432-445
* [2] Maher M J. Modelling association football scores[J]. Statistica Neerlandica, 1982, 36(36):109-118.
* [3] Dixon M J, Coles S G. Modelling Association Football Scores and Inefficiencies in the Football Betting Market[J]. Journal of the Royal Statistical Society, 1997, 46(2):265-280.
* [4] Karlis D. Analysis of Sports Data by Using Bivariate Poisson Models[J]. Journal of the Royal Statistical Society, 2003, 52(3):381-393.
* [5] Rue H, Øyvind Salvesen. Prediction and Retrospective Analysis of Soccer Matches in a League[J]. Journal of the Royal Statistical Society, 1998, 49(3):399–418.
* [6] Clarke S R, Norman J M. Home Ground Advantage of Individual Clubs in English Soccer[J]. Journal of the Royal Statistical Society, 1995, 44(4):509.
* [7] Crowder M, Dixon M, Ledford A, et al. Dynamic Modelling and Prediction of English
* Football League Matches for Betting[J]. Journal of the Royal Statistical Society, 2002,51(2):157–168.
* [8] Goddard J, Asimakopoulos I. Forecasting football results and the efficiency of

fixed-odds betting[J]. Journal of Forecasting, 2004, 23(1):51–66.

* [9] Koning R H. Balance in Competition in Dutch Soccer[J]. Journal of the Royal StatisticalSociety, 2000, 49(3):419–431.
* [10] Kuk A Y C. Modelling Paired Comparison Data with Large Numbers of Draws and Large Variability of Draw Percentages Among Players[J]. Journal of the Royal Statistical Society,1995, 44(4):523-528.
* [11] Barry D, Hartigan J A. Choice Models for Predicting Divisional Winners in Major League Baseball[J]. Journal of the American Statistical Association, 1993, 88(423):766-774.
* [12] Ludwig Fahrmeir, Gerhard Tutz. Dynamic Stochastic Models for Time-Dependent Ordered Paired Comparison Systems[J]. Journal of the American Statistical Association, 1994,89(428):1438-1449.
* [13] Baker R D, Mchale I G. A dynamic paired comparisons model: Who is the greatest tennis player?[J]. European Journal of Operational Research, 2014, 236(2):677–684.
* [14] Held L, Vollnhals R. Dynamic rating of European football teams[J]. Ima Journal of Management Mathematics, 2005, 16(2):121-130.
* [15] Mark J.Dixon,Stuart G.Coles. Modelling Association Football Scores and inefficiencies in the Football Betting Market[J]. Applied Statistics,1997,46(2):265-280.
* [16] Ian McHale,Phil Scarf. Modelling soccer matchea using bivariate discrete distributions with general dependence structure[J]. Statistical Neerlandica,2007,61(61): 432-445.
* [17] Ken Shen. Data ananlysis of basketball game performance based on bivariate poisson regression models[J]. Computer modelling & New technologies,2014,18(12):474-479.
* [18] D Mwembe.Application of a Bivariate Poisson model in devising a profitable betting strategy of the zimbabwe premier scoccer league match results[J]. American journal of theoretial and applied statistics,2015,4(3):99-111.

**附录、源代码**

附录1：英超 爬虫1代码

# -\*- codeing = utf-8 -\*-

from bs4 import BeautifulSoup # 网页解析，获取数据

import re # 正则表达式，进行文字匹配`

import urllib.request, urllib.error # 制定URL，获取网页数据

import xlwt # 进行excel操作

#import sqlite3 # 进行SQLite数据库操作

findTime\_info = re.compile(r'<font style="vertical-align: inherit;">(.\*?)</font>')

#

def main():

baseurl = "http://app.gooooal.com/resultschedule.do?lid=4&sid=2020&roundNum=1&lang=tr" #要爬取的网页链接

# 1.爬取网页

datalist = getData(baseurl)

savepath = "英超1.xls" #当前目录新建XLS，存储进去

# 3.保存数据

saveData(datalist,savepath) #2种存储方式可以只选择一种

#

# 爬取网页

def getData(baseurl):

datalist = [] #用来存储爬取的网页信息

url = baseurl

html = askURL(url) # 保存获取到的网页源码

# 2.逐一解析数据

soup = BeautifulSoup(html, "html.parser")

for item in soup.find\_all('tr', class\_="alt"): # 查找符合要求的字符串

data = [] # 保存 所有信息

item = str(item)

Time\_info = re.findall(findTime\_info, item)[0] # 通过正则表达式查找

print(Time\_info)

data.append(Time\_info)

datalist.append(data)

return datalist

#

#

def askURL(url):

head = { # 模拟浏览器头部信息，向豆瓣服务器发送消息

"User-Agent": "Mozilla/5.0 (Linux; Android 6.0; Nexus 5 Build/MRA58N) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.77 Mobile Safari/537.36"

}

# 用户代理

request = urllib.request.Request(url, headers=head)

html = ""

try:

response = urllib.request.urlopen(request)

html = response.read().decode("utf-8")

except urllib.error.URLError as e:

if hasattr(e, "code"):

print(e.code)

if hasattr(e, "reason"):

print(e.reason)

return html

#

# 保存数据到表格

def saveData(datalist,savepath):

print("save.......")

book = xlwt.Workbook(encoding="utf-8",style\_compression=0) #创建workbook对象

sheet = book.add\_sheet('英超20-21', cell\_overwrite\_ok=True) #创建工作表

col = ("队伍1")

for i in range(0,1):

sheet.write(0,i,col[i]) #列名

for i in range(0,1):

# print("第%d条" %(i+1)) #输出语句，用来测试

data = datalist[i]

for j in range(0,1):

sheet.write(i+1,j,data[j]) #数据

book.save(savepath) #保存

#

#

if \_\_name\_\_ == "\_\_main\_\_": # 当程序执行时

# 调用函数

main()

# init\_db("movietest.db")

print("爬取完毕！")

#

运行结果：

OK

---------------------------------------------------------------------------

IndexError Traceback (most recent call last)

<ipython-input-29-e61e72086727> in <module>

76 if \_\_name\_\_ == "\_\_main\_\_": # 当程序执行时

77 # 调用函数

---> 78 main()

79 # init\_db("movietest.db")

80 print("爬取完毕！")

<ipython-input-29-e61e72086727> in main()

12 baseurl = "http://app.gooooal.com/resultschedule.do?lid=4&sid=2020&roundNum=1&lang=tr" #要爬取的网页链接

13 # 1.爬取网页

---> 14 datalist = getData(baseurl)

15 savepath = "英超1.xls" #当前目录新建XLS，存储进去

16 # 3.保存数据

<ipython-input-29-e61e72086727> in getData(baseurl)

30 data = [] # 保存 所有信息

31 item = str(item)

---> 32 Time\_info = re.findall(findTime\_info, item)[0] # 通过正则表达式查找

33 print(Time\_info)

34 data.append(Time\_info)

IndexError: list index out of range

附录2：英超 爬虫2代码

from urllib.request import urlopen#用于获取网页

from bs4 import BeautifulSoup#用于解析网页

import urllib.request as req

import re

import xlsxwriter

Lable = ["Num", "url", "doi", "info"]

url = 'https://www.okooo.com/soccer/league/17/schedule/14243/1-1-1/'

webpage = req.urlopen(url) # 根据超链访问链接的网页

data = webpage.read() # 读取超链网页数据

data = data.decode('utf-8') # byte类型解码为字符串

table = re.findall(r'<p.\*?>(.\*?)</p>', data, re.S)

addr = re.findall(r'<a.\*?>(.\*?)</a>', data, re.S)

addr.reverse()

for i in range(14):

addr.pop()

addr.reverse()

#创建一个列表用于存储href属性值

hrefs = []

# todo 创建excel文件

xl = xlsxwriter.Workbook(r'D:/newfolder/test.xlsx')

# todo 添加sheet

sheet = xl.add\_worksheet('sheet1')

html = urlopen('http://www.cata-wang.com/case.php')

bsObj = BeautifulSoup(html, 'html.parser')

t1 = bsObj.find\_all('a')

for t2 in t1:

t3 = t2.get('href')

hrefs.append(t3)

for row in range(4):

sheet.write\_string(0, row, Lable[row])

for item in range(len(table)):

sheet.write\_string(item+1, 0, f"{addr[2 \* item]}")

sheet.write\_string(item + 1, 1, f"{hrefs[item \* 2 + 15]}")

sheet.write\_string(item + 1, 2, f"{addr[2 \* item + 1]}")

sheet.write\_string(item+1, 3, f"{table[item]}")

# todo 关闭文件

xl.close()

运行结果：

---------------------------------------------------------------------------

HTTPError Traceback (most recent call last)

<ipython-input-3-480127aab057> in <module>

12 url = 'https://www.okooo.com/soccer/league/17/schedule/14243/1-1-1/'

13

---> 14 webpage = req.urlopen(url) # 根据超链访问链接的网页

15

16 data = webpage.read() # 读取超链网页数据

D:\anaconda\lib\urllib\request.py in urlopen(url, data, timeout, cafile, capath, cadefault, context)

220 else:

221 opener = \_opener

--> 222 return opener.open(url, data, timeout)

223

224 def install\_opener(opener):

D:\anaconda\lib\urllib\request.py in open(self, fullurl, data, timeout)

529 for processor in self.process\_response.get(protocol, []):

530 meth = getattr(processor, meth\_name)

--> 531 response = meth(req, response)

532

533 return response

D:\anaconda\lib\urllib\request.py in http\_response(self, request, response)

638 # request was successfully received, understood, and accepted.

639 if not (200 <= code < 300):

--> 640 response = self.parent.error(

641 'http', request, response, code, msg, hdrs)

642

D:\anaconda\lib\urllib\request.py in error(self, proto, \*args)

567 if http\_err:

568 args = (dict, 'default', 'http\_error\_default') + orig\_args

--> 569 return self.\_call\_chain(\*args)

570

571 # XXX probably also want an abstract factory that knows when it makes

D:\anaconda\lib\urllib\request.py in \_call\_chain(self, chain, kind, meth\_name, \*args)

500 for handler in handlers:

501 func = getattr(handler, meth\_name)

--> 502 result = func(\*args)

503 if result is not None:

504 return result

D:\anaconda\lib\urllib\request.py in http\_error\_default(self, req, fp, code, msg, hdrs)

647 class HTTPDefaultErrorHandler(BaseHandler):

648 def http\_error\_default(self, req, fp, code, msg, hdrs):

--> 649 raise HTTPError(req.full\_url, code, msg, hdrs, fp)

650

651 class HTTPRedirectHandler(BaseHandler):

HTTPError: HTTP Error 405: Method Not Allowed

附录3：login\_spider

import pandas as pd

def get\_table(year,i):

flag=0

mlist=[]

temp=[]

if(year==2021):

f = open("D:/澳客网/{0}/text{1}.txt".format(str(year),str(i)), "r", encoding='utf-8')

else:

f=open("D:/澳客网/{0}/{1}text{2}.txt".format(str(year),str(year),str(i)), "r",encoding='utf-8')

ml=list(filter(None,f.read().splitlines()))

for item in ml:

if(item=='分析'):

flag=1

if(item=='积分总榜'):

flag=0

if(flag==1):

temp.append(item)

if(item=='欧/析'):

mlist.append(temp)

temp=[]

mlist[0].remove('分析')

football=pd.DataFrame(mlist)

return football

table2021=[]

for i in range(1,39):

table2021.append(get\_table(2021,i))

final\_table2021=pd.concat(table2021)

#print(final\_table)

final\_table2021.columns=["日期","轮次","主队","比分","客队","主队指数","平指数","客队指数","分析","empty"]

final\_table2021['year']="20-21"

#print(final\_table2021)

#final\_table.to\_csv("D:/澳客网/table1.csv")

table1920=[]

for i in range(1,39):

table1920.append(get\_table(1920,i))

final\_table1920=pd.concat(table1920)

#print(final\_table)

final\_table1920.columns=["日期","轮次","主队","比分","客队","主队指数","平指数","客队指数","分析","empty"]

final\_table1920['year']="19-20"

#print(final\_table1920)

table1819=[]

for i in range(1,39):

table1819.append(get\_table(1819,i))

final\_table1819=pd.concat(table1819)

#print(final\_table)

final\_table1819.columns=["日期","轮次","主队","比分","客队","主队指数","平指数","客队指数","分析","empty"]

final\_table1819["year"]="18-19"

#print(final\_table1819)

table1718=[]

for i in range(1,39):

table1718.append(get\_table(1718,i))

final\_table1718=pd.concat(table1718)

#print(final\_table)

final\_table1718.columns=["日期","轮次","主队","比分","客队","主队指数","平指数","客队指数","分析","empty"]

final\_table1718["year"]="17-18"

#print(final\_table1718)

table1617=[]

for i in range(1,39):

table1617.append(get\_table(1617,i))

final\_table1617=pd.concat(table1617)

#print(final\_table)

final\_table1617.columns=["日期","轮次","主队","比分","客队","主队指数","平指数","客队指数","分析","empty"]

final\_table1617["year"]="16-17"

#print(final\_table1617)

table1516=[]

for i in range(1,39):

table1516.append(get\_table(1516,i))

final\_table1516=pd.concat(table1516)

#print(final\_table)

final\_table1516.columns=["日期","轮次","主队","比分","客队","主队指数","平指数","客队指数","分析","empty"]

final\_table1516["year"]="15-16"

#print(final\_table1617)

fiveyear=pd.concat([final\_table2021,final\_table1920,final\_table1819,final\_table1718,final\_table1617,final\_table1516])

#print(fiveyear)

fiveyear.to\_csv("D:/澳客网/6year.csv")

附录4：playerinformation

import re

import pandas as pd

f1=open('D:/澳客网/1819soccer1-32.txt','r', encoding='utf-8')

f2=open('D:/澳客网/1819soccer33-38.txt','r', encoding='utf-8')

def get\_table(f):

host=[]

away=[]

for item in list(filter(None,f.read().splitlines())):

change=item.strip()

#print(change[:4])

if(change[:4]==1920):

continue

if(change[:2]=='主队'):

#print(re.findall(r"\d+",change))

temp=re.findall(r"\d+", change)

host.append(list(map(int,temp)))

if(change[:2]=='客队'):

temp1 = re.findall(r"\d+", change)

away.append(list(map(int, temp1)))

all=[]

all=host

for i in range(len(host)):

for j in range(3):

all[i].append(away[i][j])

return pd.DataFrame(all)

#print("all:",all)

d1=get\_table(f1)

print(d1)

print(d1.shape[0])

d2=get\_table(f2)

print(d2.shape[0])

df=pd.concat([d1,d2])

print(df)

df.columns=['主队身价','主队评价年龄','主队平均身高','客队身价','客队平均年龄','客队平均身高']

#df=df.iloc[:,0:-1]

#print(df)

df.to\_csv('D:/澳客网/1819player.csv')

print(df)

附录5：spider

from bs4 import BeautifulSoup

from lxml import html

import xml

import requests

import time

#url = "https://sjz.lianjia.com/ershoufang/hy1f2f5sf1l3l2l4a2a3a4/"

def get\_info(i):

#i represents ith page of the information

url="https://www.okooo.com/soccer/league/17/schedule/14243/1-1-{0}/".format(str(i))

f = requests.get(url,headers={'user-agent':'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.77 Safari/537.36',

})

soup = BeautifulSoup(f.content, "lxml") #用lxml解析器解析该网页的内容, 好像f.text也是返回的html

print(soup)

#print(f.content.decode()) #尝试打印出网页内容,看是否获取成功

content = soup.find\_all('div',class\_="clearfix tableborder" ) #尝试获取节点，因为calss和关键字冲突，所以改名class\_

#print(content)

fo=open("D:/澳客网/testtext{0}.txt".format(str(i)),'w',encoding='utf-8')

for content\_div in content:

ha=content\_div.find\_all('tr')

#print(ha)

for h in ha:

#print(h.find\_all('td'))

#print(h.get\_text())

fo.writelines(h.get\_text())

#print(h)

#print(type(h))

#temp=str(h)

#item=temp.find('target')

#print(temp[item:])

get\_info(1)

#print(housename)

#for k in soup.find\_all('div',class\_='pl2'):#,找到div并且class为pl2的标签

# a = k.find\_all('span') #在每个对应div标签下找span标签，会发现，一个a里面有四组span

# print(a[0].string) #取第一组的span中的字符串

附录6：spider2

import urllib.request

import ssl

import bs4

import lxml

import requests

import xml

import random

from urllib import request

import pandas as pd

from lxml import etree

from selenium import webdriver

import time

my\_header={"User-Agent":"Mozilla/5.0 (Windows NT 6.1; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.77 Safari/537.36"}

opt=webdriver.ChromeOptions()

driver=webdriver.Chrome(options=opt)

#proxies={'https':'116.149.194.222:8888'}

soccer=[]

with open('D:/澳客网/1819soccer33-38.txt', 'w',encoding='utf-8') as f:

for i in range(33,39):

f.writelines('1819 {0}轮次\n'.format(str(i)))

page\_url="https://www.okooo.com/soccer/league/17/schedule/13782/1-1-{0}/".format(str(i))

driver.get(page\_url)

time.sleep(5)

driver.refresh()

time.sleep(2)

req\_page=requests.get(page\_url,headers=my\_header)

soup\_page=bs4.BeautifulSoup(req\_page.content,'lxml')

#print(soup\_page)

mtable=soup\_page.find('table',id='team\_fight\_table')

nums=mtable.find\_all('tr')

arr=[]

#soup\_page.has\_attr()

for i in range(len(nums)):

if(i==0):

continue

score=nums[i].find\_all('td')[3].get\_text().strip()

if(score=='延期'):

continue

#print(i,":",nums[i])

infom=nums[i].find('a')

print("infom:",infom)

result=infom.attrs

arr.append(result['href'])

for item in arr:

url="https://www.okooo.com"+item+"formation/"

#url="https://www.okooo.com/soccer/match/1109952/formation/"

#req=requests.get(url,headers=my\_header)

driver.get(url)

time.sleep(5)

driver.refresh()

time.sleep(3)

req=requests.get(url,headers=my\_header)

soup=bs4.BeautifulSoup(req.content,'lxml')

#print(soup)

ff=soup.find('div',class\_="matchboxbg")

if(ff==None):

print('{0} is missing'.format(item))

continue

#print(soup)

infos=ff.find('span',class\_="text")

print("info:",infos.get\_text())

f.writelines(infos.get\_text())

#attack=first.find('div',class\_="match-battle-item")

#print("attack:",attack)

#attable=attack.find('div',class\_="item-box")

#print("attable:",attable)

#html=etree.HTML(attable)

#print("information:",html)

附录7：模拟鼠标点击

import os

import time

import pyautogui as pg

try:

while True:

sW, sH = pg.size() #获取屏幕的尺寸（像素）screenWidth，screenHeight

print("屏幕分辨率：\n"+str(sW)+','+str(sH)+'\n') #打印屏幕分辨率

x,y = pg.position() #获取当前鼠标的坐标（像素）

print("鼠标坐标:\n" + str(x).rjust(4)+','+str(y).rjust(4)) #打印鼠标坐标值

time.sleep(2) #等待1秒

os.system('cls') #清屏

except KeyboardInterrupt:

print('\n结束')

附录7： 预处理1

import pandas as pd

df = pd.read\_csv("5year.csv")

df.head(5)

df["score1"]=df["比分"].str[0:1]

df["score2"]=df["比分"].str[2:3]

df["judge"]=df["score1"]

index=[]

for i in range(1930):

if(df['比分'][i]==' 延期' or df['主队指数'][i]=='-'):

index.append(i)

print(i)

df=df.drop(index)

def change(x):

if(x=='-'):

return None

else:

return float(x)

df['主队指数']=df['主队指数'].apply(lambda x:change(x))

df['客队指数']=df['客队指数'].apply(lambda x:change(x))

df['平指数']=df['平指数'].apply(lambda x:change(x))

df.to\_csv("5year2.csv",index=False)

附录 8： 预处理2

import pandas as pd

df = pd.read\_csv("5year2.csv")

df.head(5)

N=1898

df[0:]

df[["主队指数","客队指数","平指数"]]= df[["主队指数","客队指数","平指数"]].values.astype(float)

# 先测试只买最小赔率（即博彩公司认为胜率最高的）

cnt=0

mony=0

for i in range(N):

if (df["主队指数"][i]<=df["客队指数"][i] and df["主队指数"][i]<=df["平指数"][i] and df["score1"][i]>df["score2"][i]):

cnt+=1

mony+=df["主队指数"][i]

if (df["主队指数"][i]>=df["客队指数"][i] and df["客队指数"][i]<=df["平指数"][i] and df["score1"][i]<df["score2"][i]):

cnt+=1

mony+=df["客队指数"][i]

if (df["主队指数"][i]>=df["平指数"][i] and df["客队指数"][i]>=df["平指数"][i] and df["score1"][i]==df["score2"][i]):

cnt+=1

mony+df["平指数"][i]

print(cnt)

print(mony)

# 测试博彩公司的盈利水平

# 归一化 修正公司盈利

# home 主场 draw平局 away 客场

df["home"]=df["主队指数"]

df["draw"]=df["平指数"]

df["away"]=df["客队指数"]

suma=0

for i in range(N):

a=1/( (1/df["主队指数"][i]) + (1/df["客队指数"][i]) + (1/df["平指数"][i]) )

df["home"][i]=df["home"][i]/a

df["draw"][i]=df["draw"][i]/a

df["away"][i]=df["away"][i]/a

suma+=a

print(suma/N)

df[0:]

df=df.drop('Unnamed: 0', 1)

df=df.drop('empty', 1)

df[0:]

df=df.drop('judge', 1)

df.to\_csv("5year3.csv",index=False)

附录 9：队伍实力回归分析

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

from pandas import DataFrame,Series

from sklearn.linear\_model import LinearRegression

adv\_data = pd.read\_excel("re.xlsx")

print('head:',adv\_data.head(),'\nShape:',adv\_data.shape)

#数据描述

print(adv\_data.describe())

#缺失值检验

print(adv\_data[adv\_data.isnull()==True].count())

def data\_norm(df,\*cols):

df\_n = df.copy()

for col in cols:

ma = df\_n[col].max()

mi = df\_n[col].min()

df\_n[col + '\_n'] = (df\_n[col] - mi) / (ma - mi)

return(df\_n)

# 创建函数，标准化数据

adv\_data = data\_norm(adv\_data,'身价')

adv\_data = data\_norm(adv\_data,'平均年龄')

adv\_data = data\_norm(adv\_data,'平均身高')

adv\_data.head(5)

df=adv\_data[['身价\_n','平均年龄\_n','平均身高\_n','score']]

df.boxplot()

plt.savefig("boxplot.jpg")

plt.show()

##相关系数矩阵 r(相关系数) = x和y的协方差/(x的标准差\*y的标准差) == cov（x,y）/σx\*σy

#相关系数0~0.3弱相关0.3~0.6中等程度相关0.6~1强相关

print(df.corr())

from \_\_future\_\_ import division

import seaborn as sns

# 通过加入一个参数kind='reg'，seaborn可以添加一条最佳拟合直线和95%的置信带。

sns.pairplot(df, x\_vars=['身价\_n','平均年龄\_n','平均身高\_n'], y\_vars='score', size=7, aspect=0.8,kind = 'reg')

plt.savefig("pairplot.jpg")

plt.show()

adv\_data.boxplot()

plt.savefig("boxplot1.jpg")

plt.show()

##相关系数矩阵 r(相关系数) = x和y的协方差/(x的标准差\*y的标准差) == cov（x,y）/σx\*σy

#相关系数0~0.3弱相关0.3~0.6中等程度相关0.6~1强相关

print(adv\_data.corr())

# 通过加入一个参数kind='reg'，seaborn可以添加一条最佳拟合直线和95%的置信带。

sns.pairplot(adv\_data, x\_vars=['身价','平均年龄','平均身高'], y\_vars='score', size=7, aspect=0.8,kind = 'reg')

plt.savefig("pairplot2.jpg")

plt.show()

import sklearn

X\_train,X\_test,Y\_train,Y\_test = sklearn.model\_selection.train\_test\_split(adv\_data[['身价','平均年龄','平均身高']],adv\_data.score,train\_size=.80)

print("原始数据特征:",adv\_data[['身价','平均年龄','平均身高']].shape,

",训练数据特征:",X\_train.shape,

",测试数据特征:",X\_test.shape)

print("原始数据标签:",adv\_data.score.shape,

",训练数据标签:",Y\_train.shape,

",测试数据标签:",Y\_test.shape)

model = LinearRegression()

model.fit(X\_train,Y\_train)

a = model.intercept\_#截距

b = model.coef\_#回归系数

print("最佳拟合线:截距",a,",回归系数：",b)

附件10：利用回归的结果 knn

a=[ 0.00077575,-0.00745624,0.00244002]

from sklearn import datasets

import numpy as np

import pandas as pd

from sklearn.model\_selection import KFold

pd.set\_option('display.max\_columns',None)

pd.set\_option('display.max\_rows', None)

data=datasets.load\_wine()

wine=pd.DataFrame(data.data)

wine=wine.sub(wine.mean(),axis=1).div(wine.std(),axis=1)

target=data.target

wine['target']=target

test\_index=[2,17,21,30,68,77,81,89,98,111,121,126,144,145,158,160,172,176]

train\_index=list(set(wine.index)-set(test\_index))

test=wine.iloc[test\_index].reset\_index(drop=True)

train=wine.iloc[train\_index].reset\_index(drop=True)

def acc(test, predict):

return sum(test == predict) / test.size

def knn(train,test,k):

predict=[]

for te in np.array(test):

sub=train.iloc[:,:-1].sub(te[:-1])

d=np.sqrt((sub\*\*2).sum(axis=1))

d\_index=d.sort\_values(ascending=True).index[:k]

pred=train['target'].iloc[d\_index].value\_counts().index[0]

predict.append(pred)

return predict

#1)对训练集 10 折交叉验证的结果

x=[1,3,5,7,9]

kf = KFold(n\_splits=10,shuffle=True)

maxacc=0

for k in x:

accuracy=0

for train\_index, test\_index in kf.split(train):

tra=train.iloc[train\_index].reset\_index(drop=True)

validation=train.iloc[test\_index].reset\_index(drop=True)

predict=knn(tra,validation,k)

accuracy+=acc(validation['target'],predict)

accuracy/=10

print('k=%d:'%k,accuracy)

# 基于整个训练集对测试集样本进行预测并计算 accuracy

for kk in x:

predict=knn(train,test,kk)

print('k=%d:'%kk, acc(test['target'],predict))

附件11： 聚类(简单的分成3类)

import pandas as pd

df = pd.read\_csv("5year3.csv")

N=1898

df.head(5)

import numpy as np

from sklearn import metrics

import math

x=df[['home','draw','away']]

# x=np.array(x)

from sklearn.metrics.pairwise import cosine\_similarity

def distance(a,b):

tmp=[a,b]

tmp1=cosine\_similarity(tmp)

return tmp1[1][0]

import matplotlib.pyplot as plt

import numpy as np

from sklearn.cluster import KMeans

estimator = KMeans(n\_clusters=3)#构造聚类器

estimator.fit(x)#聚类

label\_pred = estimator.labels\_ #获取聚类标签

df['cluster']=label\_pred

df

cnt=0

s=0

for i in range(N):

if (df['cluster'][i]==2 and df['score1'][i]<df['score2'][i]):

cnt+=1

s+=df['客队指数'][i]

if (df['cluster'][i]==1 and df['score1'][i]>df['score2'][i]):

cnt+=1

s+=df['主队指数'][i]

if (df['cluster'][i]==0 and df['score1'][i]==df['score2'][i]):

cnt+=1

s+=df['平指数'][i]

print(cnt)

print(s)

附录 12： kmean聚类(简单余弦)-1

import pandas as pd

df = pd.read\_csv("5year3.csv")

N=1898

df.head(5)

df['ans']=0

for i in range(N):

if (df['score1'][i]>df['score2'][i]):

df['ans'][i]=1

if (df['score1'][i]==df['score2'][i]):

df['ans'][i]=0

if (df['score1'][i]<df['score2'][i]):

df['ans'][i]=-1

import numpy as np

from sklearn import metrics

import math

x=np.array(df[['home','draw','away']])

# 生成随机数列 k-mean 用

RAM=np.random.permutation(N)

RAM

def distance(a,b):

if (np.linalg.norm(b)==0) or (np.linalg.norm(a)==0):

return 0

return np.dot(a,b)/np.linalg.norm(a)/np.linalg.norm(b)

import copy

def kmeans(X,k,limitation):

# 返回中心坐标

# initialize

cent=X[0:k]

newcent=X[0:k]

clust=np.zeros(N)

newclust=np.zeros(N)

RAM=np.random.permutation(N)

for i in range(k):

cent[i]=X[RAM[i]]

cnt=0

# print(cent)

while cnt<limitation:

cnt+=1

# renew cluster belonging

for i in range(N):

min\_dis=0

for j in range(k):

if (distance(X[i],cent[j])-min\_dis>0):

min\_dis=distance(X[i],cent[j])

newclust[i]=j

# renew center

for j in range(k):

newcent[j]=[0,0,0]

c=0

for i in range(N):

if (newclust[i]==j):

newcent[j]=newcent[j]+X[i]

c+=1

if (c!=0):

newcent[j]=newcent[j]/c

# print(newcent)

# test ultimation

flag=1

# print("old clust")

# print(clust[0:50])

# print("new")

# print(newclust[0:50])

for i in range(N):

if clust[i]!=newclust[i]:

flag=0

break

if (flag==1):

break

cent=copy.deepcopy(newcent)

clust=copy.deepcopy(newclust)

return cent

c1=kmeans(x,3,500)

def kmean\_label(X,k,center):

cnt=np.zeros(3\*k).reshape(k,-1)

ans=np.zeros(k)

# 算出每个簇的选择（简单多数法）

for i in range(N):

min\_dis=0

ind=-1

for j in range(k):

if (distance(X[i],center[j])-min\_dis>0):

min\_dis=distance(X[i],center[j])

ind=j

if(df['score1'][i]>df['score2'][i]):

cnt[ind][0]+=1

if(df['score1'][i]==df['score2'][i]):

cnt[ind][1]+=1

if(df['score1'][i]<df['score2'][i]):

cnt[ind][2]+=1

# ans 1 表示主队赢， 2表示平，3 客队赢

for j in range(k):

if(cnt[j][0]>=cnt[j][1])and(cnt[j][0]>=cnt[j][2]):

ans[j]=1

if(cnt[j][1]>cnt[j][0])and(cnt[j][1]>=cnt[j][2]):

ans[j]=2

if(cnt[j][2]>cnt[j][0])and(cnt[j][2]>cnt[j][1]):

ans[j]=3

return ans

lab1=kmean\_label(x,3,c1)

def kmean\_test(DF,center,label,k):

cnt=0

sum=0

for i in range(DF.iloc[:,0].size):

min\_dis=0

ind=-1

X=np.array([ df['主队指数'][i] , df['平指数'][i] , df['客队指数'][i] ])

for j in range(k):

if (distance(center[j],X)-min\_dis>0):

min\_dis=distance(X,center[j])

ind=j

if ind==1 and df['score1'][i]>df['score2'][i]:

cnt+=1

sum+=df['主队指数'][i]

if ind==2 and df['score1'][i]==df['score2'][i]:

cnt+=1

sum+=df['平指数'][i]

if ind==3 and df['score1'][i]<df['score2'][i]:

cnt+=1

sum+=df['客队指数'][i]

print(cnt)

print(sum)

return sum

kmean\_test(df,c1,lab1,3)

附件13：聚类(余弦)整合-2

import pandas as pd

df = pd.read\_csv("5year3.csv")

N=1898

df.head(5)

df['ans']=0

for i in range(N):

if (df['score1'][i]>df['score2'][i]):

df['ans'][i]=1

if (df['score1'][i]==df['score2'][i]):

df['ans'][i]=0

if (df['score1'][i]<df['score2'][i]):

df['ans'][i]=-1

import numpy as np

from sklearn import metrics

import math

x=np.array(df[['home','draw','away']])

# 生成随机数列 k-mean 用

RAM=np.random.permutation(N)

RAM

def distance(a,b):

if (np.linalg.norm(b)==0) or (np.linalg.norm(a)==0):

return 0

return np.dot(a,b)/np.linalg.norm(a)/np.linalg.norm(b)

import copy

def kmeans(X,k,limitation):

# 返回中心坐标

# initialize

cent=X[0:k]

newcent=X[0:k]

clust=np.zeros(N)

newclust=np.zeros(N)

RAM=np.random.permutation(N)

for i in range(k):

cent[i]=X[RAM[i]]

cnt=0

# print(cent)

while cnt<limitation:

cnt+=1

# renew cluster belonging

for i in range(N):

min\_dis=0

for j in range(k):

if (distance(X[i],cent[j])-min\_dis>0):

min\_dis=distance(X[i],cent[j])

newclust[i]=j

# renew center

for j in range(k):

newcent[j]=[0,0,0]

c=0

for i in range(N):

if (newclust[i]==j):

newcent[j]=newcent[j]+X[i]

c+=1

if (c!=0):

newcent[j]=newcent[j]/c

# print(newcent)

# test ultimation

flag=1

# print("old clust")

# print(clust[0:50])

# print("new")

# print(newclust[0:50])

for i in range(N):

if clust[i]!=newclust[i]:

flag=0

break

if (flag==1):

break

cent=copy.deepcopy(newcent)

clust=copy.deepcopy(newclust)

return cent

def kmean\_label(X,k,center):

cnt=np.zeros(3\*k).reshape(k,-1)

ans=np.zeros(k)

# 算出每个簇的选择（简单多数法）

for i in range(N):

min\_dis=0

ind=-1

for j in range(k):

if (distance(X[i],center[j])-min\_dis>0):

min\_dis=distance(X[i],center[j])

ind=j

if(df['score1'][i]>df['score2'][i]):

cnt[ind][0]+=1

if(df['score1'][i]==df['score2'][i]):

cnt[ind][1]+=1

if(df['score1'][i]<df['score2'][i]):

cnt[ind][2]+=1

# ans 1 表示主队赢， 2表示平，3 客队赢

for j in range(k):

if(cnt[j][0]>=cnt[j][1])and(cnt[j][0]>=cnt[j][2]):

ans[j]=1

if(cnt[j][1]>cnt[j][0])and(cnt[j][1]>=cnt[j][2]):

ans[j]=2

if(cnt[j][2]>cnt[j][0])and(cnt[j][2]>cnt[j][1]):

ans[j]=3

return ans

def kmean\_test(DF,center,label,k):

cnt=0

sum=0

for i in range(DF.iloc[:,0].size):

min\_dis=0

ind=-1

X=np.array([ df['主队指数'][i] , df['平指数'][i] , df['客队指数'][i] ])

for j in range(k):

if (distance(center[j],X)-min\_dis>0):

min\_dis=distance(X,center[j])

ind=j

if label[ind]==1 and df['score1'][i]>df['score2'][i]:

cnt+=1

sum+=df['主队指数'][i]

if label[ind]==2 and df['score1'][i]==df['score2'][i]:

cnt+=1

sum+=df['平指数'][i]

if label[ind]==3 and df['score1'][i]<df['score2'][i]:

cnt+=1

sum+=df['客队指数'][i]

print(cnt,' ',sum)

return sum

def TEST():

Max\_money=0

Max\_k=0

for K in range(13):

testK=K+2

print("k = ",testK)

c1=kmeans(x,testK,500)

lab1=kmean\_label(x,testK,c1)

tmp=kmean\_test(df,c1,lab1,testK)

if(tmp>Max\_money):

Max\_money=tmp

Max\_k=testK

return Max\_k

TEST()

附件14： 聚类(余弦) 连续化 流出法-4

import pandas as pd

df = pd.read\_csv("5year3.csv")

N=1898

df.head(5)

df['ans']=0

for i in range(N):

if (df['score1'][i]>df['score2'][i]):

df['ans'][i]=1

if (df['score1'][i]==df['score2'][i]):

df['ans'][i]=0

if (df['score1'][i]<df['score2'][i]):

df['ans'][i]=-1

import numpy as np

from sklearn import metrics

import math

x=np.array(df[['home','draw','away']])

# 生成随机数列 k-mean 用

RAM=np.random.permutation(N)

RAM

def distance(a,b):

if (np.linalg.norm(b)==0) or (np.linalg.norm(a)==0):

return 0

return np.dot(a,b)/np.linalg.norm(a)/np.linalg.norm(b)

import copy

def kmeans(X,k,limitation):

# 返回中心坐标

# initialize

cent=X[0:k]

newcent=X[0:k]

clust=np.zeros(N)

newclust=np.zeros(N)

N1=np.shape(X)[0]

RAM=np.random.permutation(N1)

for i in range(k):

cent[i]=X[RAM[i]]

cnt=0

# print(cent)

while cnt<limitation:

cnt+=1

# renew cluster belonging

for i in range(N1):

min\_dis=0

for j in range(k):

if (distance(X[i],cent[j])-min\_dis>0):

min\_dis=distance(X[i],cent[j])

newclust[i]=j

# renew center

for j in range(k):

newcent[j]=[0,0,0]

c=0

for i in range(N1):

if (newclust[i]==j):

newcent[j]=newcent[j]+X[i]

c+=1

if (c!=0):

newcent[j]=newcent[j]/c

# print(newcent)

# test ultimation

flag=1

# print("old clust")

# print(clust[0:50])

# print("new")

# print(newclust[0:50])

for i in range(N1):

if clust[i]!=newclust[i]:

flag=0

break

if (flag==1):

break

cent=copy.deepcopy(newcent)

clust=copy.deepcopy(newclust)

return cent

def kmean\_label(X,k,center):

cnt=np.zeros(3\*k).reshape(k,-1)

ans=np.zeros(k)

# 算出每个簇的选择（简单多数法）

N1=np.shape(X)[0]

for i in range(N1):

min\_dis=0

ind=-1

for j in range(k):

if (distance(X[i],center[j])-min\_dis>0):

min\_dis=distance(X[i],center[j])

ind=j

if(df1['score1'][i]>df1['score2'][i]):

cnt[ind][0]+=1

if(df1['score1'][i]==df1['score2'][i]):

cnt[ind][1]+=1

if(df1['score1'][i]<df1['score2'][i]):

cnt[ind][2]+=1

return cnt

def kmean\_test(DF,DF\_head,center,label,k):

cnt=0

sum=0

for ii in range(DF.iloc[:,0].size):

i=ii+DF\_head

min\_dis=0

ind=-1

X=np.array([ DF['主队指数'][i] , DF['平指数'][i] , DF['客队指数'][i] ])

for j in range(k):

if (distance(center[j],X)-min\_dis>0):

min\_dis=distance(X,center[j])

ind=j

if DF['score1'][i]>DF['score2'][i]:

sum+=DF['主队指数'][i]\*label[ind][0]/(label[ind][0]+label[ind][1]+label[ind][2])

if DF['score1'][i]==DF['score2'][i]:

sum+=DF['平指数'][i]\*label[ind][1]/(label[ind][0]+label[ind][1]+label[ind][2])

if DF['score1'][i]<DF['score2'][i]:

sum+=DF['客队指数'][i]\*label[ind][2]/(label[ind][0]+label[ind][1]+label[ind][2])

print(sum)

return sum

def TEST(DF,X,DF\_head):

print(DF.iloc[:,0].size)

Max\_money=0

Max\_k=0

for K in range(14):

testK=K+2

print("k = ",testK)

c1=kmeans(X,testK,500)

lab1=kmean\_label(X,testK,c1)

tmp=kmean\_test(DF,DF\_head,c1,lab1,testK)

if(tmp>Max\_money):

Max\_money=tmp

Max\_k=testK

return [Max\_k,Max\_money]

df2=df1[1500:1898]

x2=x1[0:1500]

TEST(df2,x2,1500)

附录15： 处理回归函数的目标函数值

#include<bits/stdc++.h>

using namespace std;

typedef long long ll;

const int INF=0x3f3f3f3f;

int main(){

int n=20;

freopen("in1.txt","r",stdin);

freopen("out.txt","w",stdout);

double a[30];

for(int i=0;i<n;i++)cin>>a[i];

double b[100]={1.0001,1.0002,1.0003,1.0004,1.0005,1.0006,1.0007,1.0008,1.0009,1.001,

1.0011,1.0012,1.0013,1.0014,1.0015,1.0016,1.0017,1.0018,1.0019,1.002,

1.0025,1.003,1.0035,1.004,1.005,1.006,1.007,1.008,1.009,1.01,

1.011,1.012,1.013,1.014,1.015,1.016,1.017,1.018,1.019,1.02

,1.025,1.03,1.035,1.04,1.045,1.05,1.06,1.07,1.08,1.09,1.1

,1.11,1.12,1.13,1.14,1.15,1.16,1.17,1.18,1.19,1.2

,1.25,1.3,1.35,1.4,1.45,1.5,1.6,1.7,1.8,1.9,2

};

for(int i=0;i<n;i++){

for(int j=0;b[j]!=0;j++)printf("%lf\t",pow(b[j],a[i]);

printf("\n");

}

return 0;

}