基于k-prototypes算法的游戏玩家细分系统的设计与实现

摘要

随着互联网以及移动互联网的发展，多人对战游戏渐渐成为人们茶余饭后的主要娱乐方式。各类竞技游戏通过自己的游戏匹配系统对玩家进行在线匹配。然而传统的游戏匹配系统往往基于游戏玩家的实际水平而进行相应的匹配，而对于玩家的游戏体验，付费信息，游戏素质等因素缺乏考虑。

本文的主要研究内容是希望游戏匹配系统在线匹配过程中，能够综合考虑到除了游戏玩家竞技水平之外的一些其它的因素。因而通过对遇到的问题进行分析，发现通过采用K-prototypes算法可以较好的实现对玩家类型的合理的分配。接着对游戏匹配系统所需要的框架模块进行了整体分析，最后本文利用Python实现了模拟的游戏匹配系统分类。

【关键词】K-prototypes算法，游戏匹配系统，在线匹配，聚类分析

Abstract

With the development of the Internet and the mobile Internet, multiplayer games have gradually become the main form of entertainment for people after their gossips. All kinds of competitive games through their own game matching system to match online players. However, the traditional game matching system is often based on the actual level of the game player to make the corresponding match, but for the player's game experience, payment information, game quality and other factors lacking consideration.  
The main research content of this paper is to hope that in the process of online matching of game matching system, we can consider some other factors except game player's competition level. Therefore, through the analysis of the problems encountered, I found that by using K-prototypes algorithm, the reasonable allocation of player types can be achieved well. Then, the whole framework of the game matching system module is analyzed. Finally, this paper uses Python to simulate the game matching system classification.

[Keywords] K-prototypes algorithm, game matching system, online matching, clustering analysis

第一章：绪论

本章首先阐明了本文所选的课题的研究背景以及其所具有的研究价值意义，接下来介绍相关领域里常见的游戏/体育匹配系统的匹配算法。最后介绍本论文的主要研究工作和本论文内容安排。

* 1. 本选题的背景与研究意义

近年来，随着人们物质生活水平的提高，相应的精神生活需求也在增长。并且游戏已经在人们的精神生活体验中占据了一定的地位，闲暇之余体验游戏的快感不失为一种娱乐放松的方式，特别是近年来随着计算机与智能手机的普及，以它们为传播媒介的电子游戏更是有了长足的发展。而好的游戏匹配系统不仅可以给玩家带来极好的体验，还可以延长游戏的寿命，增加游戏公司的收益，促进社会的经济发展。

同时，随着竞技游戏玩家对于游戏体验的需求不断提高，游戏运营商也面临着如何为玩家提供一局高质量的游戏竞技的问题。在竞技游戏中，或称基于比赛的游戏(match-based game)，游戏内容通常是固定的，拥有良好的平衡性(会定期发布补丁来调整和添加游戏内容)，而玩家的游戏体验主要是通过游戏过程中与其他玩家的交互来体现(合作和对抗)。 因此，游戏运营商想要满足玩家的需求，就必须为玩家匹配合适的队友和对手，使得玩家有一个紧张激烈的游戏体验。 然而，传统的匹配系统(例如随机匹配和玩家手动寻找)只关心游戏能够顺利开始，并没有考虑玩家的游戏体验，无法满足上述需求。 随之而来的问题则是越来越多的游戏中的不当行为(inappropriate behaviors)，例如挂机和恶意言语等，降低玩家的游戏体验，最终造成玩家流失。

在这样的背景之下，本文所研究的基于k-prototypes算法的游戏玩家细分系统对游戏玩家细分的目的是通过对玩家合理的分类与公平的匹配使玩家能够更加友好地进行游戏，并获得更好的游戏体验。同时游戏公司也能更加清楚地了解玩家对游戏的体验结果和消费水平，从而对游戏发展的整体走向有一个明确的认识以便及时对游戏更新维护，优化体验，创造更加可观的收益，也可以在一定程度上净化游戏环境，减少游戏玩家的流失。

* 1. 相关游戏匹配系统简介

匹配系统是伴随着在线竞技游戏的发展而出现的，随着玩家数量的激增，游戏匹配系统渐渐成为游戏和玩家不可或缺的重要工具。目前，游戏匹配系统大致分为根据游戏玩家的体验的玩家匹配系统和根据游戏玩家水平的玩家匹配系统以及对两者相结合的游戏匹配系统几个类别。本节主要简单介绍一下根据玩家水平的游戏匹配系统。

几乎每个竞技游戏玩家都追求紧张激烈， 犹如站在悬崖边上的游戏体验。 这种游戏体验的一个必要条件就是游戏中玩家必须大致位于同一水平线上。玩家的水平或者说技术在竞技游戏中扮演着重要的角色，直接影响玩家之间的交互和游戏体验。

基于玩家水平(Skill-Based Approach， 简称 SBA)的游戏匹配系统负责根据玩家的水平来推荐平衡的比赛(Balanced Match)。其体系结构主要由两部分组成：水平评级系统(skill-rating sys-tem)和游戏匹配系统(matchmaking system)。

水平评级系统可以看作是玩家水平模型，它会根据玩家的历史表现(例如胜率)为他们分配一个数值来反映玩家的水平。而匹配系统则会根据玩家池中的玩家当前的水平值和一些约束条件(例如匹配时间)，为玩家匹配一局游戏。基于玩家游戏水平的匹配系统的工作流程通常为： 在游戏中记录水平评级系统所需的数据(通常为游戏结果，也有系统会考虑玩家在游戏中的表现)，每局比赛结束后，都会由水平评级系统为玩家更新水平值，这个值作为临时的水平值为下局游戏的匹配系统所用，当玩家玩的局数足够多时，这个值会趋于稳定。匹配系统则每次都根据玩家的水平值尽量为玩家匹配水平接近的对手。可以看出，水平评级系统是 SBA 的核心，只有准确有效地为玩家分配水平值，才能保证玩家拥有平衡的游戏体验。 而水平评级系统本身作为一个独立的研究领域，已经有了较为成熟的方法体系，并且在体育运动中有着广泛的应用。接下来我们简单讨论一下水平评级系统以及介绍两个在游戏中广泛应用的典型的水平评级系统。

不论游戏还是体育，评估选手或者队伍水平的问题有着很长的历史。通常水平评级系统 (skill-ratingsystem)用来帮助完成选手或者玩家的排位(ranking)、匹配(matchmaking)和结果预测(outcome prediction)。有很多算法被应用于在线游戏中的玩家排位和评级，从简单的根据比赛结果的玩家位置交换(position swapping)，到复杂的基于统计的系统(statistics based system)。不论是基于比赛结果的统计系统，或者是基于多种因素和情境的评级系统，还有根据玩家游戏中表现的评级系统，或者是玩家声誉(player's reputation)，甚至是基于模糊逻辑(fuzzy logic)的评级系统，这些系统虽然在方法上大相径庭，但是都有一个共同的特点，即都是以比赛为单位来评估选手的水平。这种基于比赛的结构恰好与在线竞技游戏的特点相符合，正因为如此，水平评级系统才被广泛应用于竞技游戏中， 并扮演着越来越重要的作用。

ELOELO 评级系统 (ELO rating system)， 最初由它的创始人 Arpad Elo5 设计来改进国际象棋的评级系统，并被国际棋联(FIDE)采用至今。它的改进和扩展被广泛应用于竞技游戏中的水平评估(dota、lol、hon、wow)。 ELO评级算法是一个基于统计学的方法，系统把比赛结果的概率看作是两名选手水平差异的函数。这个函数在原始的系统中为高斯密度函数，但实践表明选手的表现并非呈正态分布，所以改进后的 ELO 排名系统通常使用的是逻辑函数。选手的等级分计算过程如下：在每局比赛之前，根据参赛选手的当前分数，比赛结果的期望被计算出来，这个期望值对之后选手的加分和丢分有着重大的影响，简单说来，排名更高的选手被认为应该赢得比赛，因此如果高排位选手赢得了比赛，他只会加较少的分数(相对于对手获胜的情况)，同时如果他输掉了比赛，他会损失很多的分数。ELO 排名系统选手的数量和水平间呈现正态分布，如图 2 所示，从图中可以看出，绝大部分玩家都位于普通水平段，而顶尖玩家和初学者则位于分布的两端。根据大数定律，一个玩家或者队伍所完成的比赛越多，系统对他的评级也就越精确。统计证据显示，系统的评级分数会在有效分数周围最多 200 分范围内波动。分数反映了玩家相对于其他参与者的相对位置，由于大家都使用统一的评分模型，在玩家比赛足够多时， 这个分数作为玩家的水平值是可行的。

True Skill 系统是基于贝叶斯推断 (Bayesian infer-ence) 的评分系统 ，由微软研究院 (Microsoft research) 开发以替代传统的 ELO 评分，并成功应用于 Xbox Live7自动匹配系统。 True Skill 评分系统是 Glicko 评分系统的衍伸，主要应用于多人游戏中(gears of war2 and halo3)。True Skill评分系统考虑到了玩家水平的不确定性 ，综合考虑了玩家的胜率和可能的水平涨落。 当玩家进行了更多的游戏后，即使你的胜率不变，系统也会因为对玩家的水平更加了解而改变评分。 在 True Skill 系统中，玩家的水平被看作一个正态分布，如图 3 所示，它被两个参数描述：1、平均值 μ(代表着玩家实际的水平值)；2、标准差 σ(代表玩家水平的不确定性)。 不确定性(uncertainty) 指的是游戏中的运气成分 ， 例如关键时刻出现的暴击使得你杀死了对手。 选手的表现稳定会降低这个值，反之这个值会上升，这个值的大小代表着系统对于你的水平值的信心。 而你的实际水平则通过胜负来衡量，True Skill 不会考虑玩家在游戏中的表现数据(你的击杀/死亡率，headshot count，等等)。True Skill 只会记录玩家的 μ 值和 σ 值。 在 True Skill 系统下，你的队伍会匹配到大致一个水平线的对手， 系统会预测比赛的结果(类似 ELO 系统)，如果你应当赢得比赛而你输了， 你会丢失水平值； 如果你应当输掉比赛而你赢了，你会增加水平值。 如果应当赢的情况下赢了或者应当输的情况下输了， 你会增加/丢失小部分的水平值。以上只是简单而基本的概览，True Skill 中有许多技术性的细节，想要全面了解可以阅读相关文献。

匹配系统在建立了可靠的玩家水平评级系统后，游戏的匹配过程就变得简单明了，即寻找“最平衡”的游戏。下面介绍 3 种常用的匹配方法：最常用也是最简单的匹配算法是保证玩家间的分数差距不超过某个值(例如100分)，同时两队间玩家的分数总和差距也不超过 100 分。另一个当前流行的方法(starcraft2 and league of legends)是将玩家进行分组。根据玩家的分数，将玩家划分到水平不同的小组中，玩家只会匹配到同组中的玩家作为对手，若玩家在某组中表现出色，分数超过了临界值，就会升组，反之则会降组。分组系统的引入缓解了匹配时间的约束下匹配质量过低的问题。 最后一个更为复杂的方法是， 将匹配系统看作一个推荐系统(recommendation system)， 并根据一个相似度公式来计算一局比赛玩家间的相似度。 一旦相似度被计算出来，各个比赛会根据这个相似度排序， 最优的那局比赛会被推荐给玩家。

1.3本论文的主要研究工作

虽然现有的游戏匹配系统已经相当强大，然而其往往过分的专注于游戏玩家的水平而进行相应的分类。缺少对单纯的玩家竞技水平之外的一些因素的考虑（例如，玩家对不同游戏角色的喜好，玩家的付费状态，玩家的游戏素质）。因而本文的主要研究内容是在已获得玩家水平模型数据以及相关外部因素数据基础之上，利用K-prototype算法实现一种更加合理的游戏匹配分类系统。

1.4本论文的内容安排

本论文首先介绍本文的研究课题背景与研究意义，以及已存在的游戏匹配系统的现状类别，接下来介绍聚类分析的相关概念和本次设计采用的K-prototypes算法。然后，开始进行系统模块设计和搭建，最后给出基于Python的游戏匹配系统。

第二章 聚类分析

本章首先介绍聚类分析的大概念，接下来具体介绍了K-prototypes算法的具体内容与算法流程。

**2.1 聚类分析的概念**

　 聚类分析（Cluster Analysis）又称为群分析，是根据“物以类聚”的道理，对样品或指标进行分类的一种多元统计分析方法，其讨论的对象是大量的样品，要求能合理地按各自的特性来进行合理的分类，没有任何模式可供参考或依循，即是在没有先验知识的情况下进行的。聚类分析起源于分类学，在古老的分类学中，人们主要依靠经验和专业知识来实现分类，很少利用数学工具进行定量的分类。随着人类科学技术的发展，对分类的要求越来越高，以致有时仅凭经验和专业知识难以确切地进行分类，于是人们逐渐地把数学工具引用到了分类学中，形成了数值分类学，之后又将多元分析的技术引入到数值分类学形成了聚类分析。聚类分析被应用于很多方面，在商业上，聚类分析被用来发现不同的客户群，并且通过购买模式刻画不同的客户群的特征；在生物上，聚类分析被用来动植物分类和对基因进行分类，获取对种群固有结构的认识；在地理上，聚类能够帮助在地球中被观察的数据库商趋于的相似性；在保险行业上，聚类分析通过一个高的平均消费来鉴定汽车保险单持有者的分组，同时根据住宅类型，价值，地理位置来鉴定一个城市的房产分组；在因特网应用上，聚类分析被用来在网上进行文档归类来修复信息。聚类是将数据分类到不同的类或者簇这样的一个过程，所以同一个簇中的对象有很大的相似性，而不同簇间的对象有很大的相异性。聚类分析的目标就是在相似的基础上收集数据来分类。聚类分析可以应用于很多领域，包括数学，计算机科学，统计学，生物学和经济学。在不同的应用领域，很多聚类技术都得到了发展，这些技术方法被用作描述数据，衡量不同数据源间的相似性，以及把数据源分类到不同的簇中。如今，聚类分析作为一种比较重要的数据挖掘技术，被广泛应用于模式识别、空间数据分析、图像处理以及市场营销等领域。聚类分析同时作为数据挖掘和机器学习等领域有力的研究工具受到越来越多的研究者的重视，在文献中也涌现出了大量的研究结果。各种各样的算法，其算法的选择取决于数据的类型、聚类的目的和应用。按不同的分类性质，大致可以把目前的聚类算法分成：（1）层次的方法（hierarchical method）；（2）划分方法（partitioning method）；（3）基于密度的方法（density-based method）；（4）基于网格的方法（grid-based method）；（5）基于模型的方法（model-based method）。

**2.2 K-prototypes算法**

K-prototype算法作为聚类分析中的一种典型算法，是处理混合属性聚类的典型算法。其不仅继承了K-means算法和K-modes算法的思想。而且还加入了描述数据簇的原型和混合属性数据之间相异度的计算公式。

其常规定义为：设X={，，……}为一个数据集合（包含n个数据），每个数据包含m个数据属性，即：={,,,……}。其中表示数据i的第j个属性。设前p个属性为数值型属性，后m-p个属性为分类属性。K-prototypes算法的目的是设定一个目标函数，其类似于K-means算法的误差平方和，自己不断地迭代，一直到目标函数值不再变化。

K-prototypes算法所处理的数据是混合型数据，既有数值型数据又有分类属性数据。K-prototypes算法对混合型数据处理的办法就是，采用混合属性簇原型。混合属性簇原型就是其数值型原型用属性中所有属性取值的均值作为数值型簇聚类的质心，分类属性原型则是分类属性中属性值取值频率最高的属性作为离散分类型聚类的质心。两者合起来就是混合型数据簇的原型。

对于分类属性：采用海明威距离，即属性值相同，为0；属性值不同，为1。

对于分类属性：

对于数值型数据属性：采用欧式距离。

则数据和簇的相异度计算公式为：

其中分别为数值型和分类属性数据。

K-prototypes算法的具体步骤是：

步骤 1 确定类的个数 k，并为每个类选择 k 个初始的聚类原型，，…，;

步骤 2 按照式(2.2)计算样本到各原型的距离，并将其划分到离它最近的聚类原型所代表的聚类中;

步骤 3 对于每个聚类,重新计算聚类原型;

步骤 4 重复步骤 3,直到各个聚类中数据对象稳定，迭代目标函数值 F( X，V) 不变，结束。

2.3 本章小结

本章对数据挖掘中经常用到的聚类分析概念进行了大致的介绍，并且接着介绍了本论文研究的核心算法K-prototypes算法，并介绍了K-prototypes算法的具体实现流程。

三．系统模块架构

本章根据所将要研究的游戏匹配系统，首先对问题进行分析，接下来对具体的匹配系统架构进行简单的介绍。

3.1 问题分析

本论文主要研究的是基于玩家的游戏水平，对参与游戏的玩家进行分类的问题。每一个游戏玩家的个体属性在这里分为两类，一类是游戏玩家本身的游戏竞技水平。此类属性一般情况下是随着游戏玩家对游戏次数增加而呈现一个对数级分布，在这里一般利用游戏玩家的胜率来表示，不过同时也要考虑到游戏玩家的游戏场数。所以在这里采用暴雪游戏的星际争霸的匹配系统机制即计算游戏玩家的MMR来评判玩家的真实的游戏水平。另一类游戏玩家属性，则包括付费玩家，游戏素质高低，游戏角色喜好程度。根据这些性质对每个具体属性离散量化为0-9一共10个等级，根据这些离散属性和游戏玩家的竞技水平的连续属性，来对游戏玩家进行具体的分类。这样设每个游戏玩家为变量=[,,,]，这样,,都是离散型变量，而,则为连续型变量。接下来根据K-prototypes算法，计算其相异性测度。

则对应于分类数据的初始聚类的数值属性和分类属性。则是聚类的分类属性的权值，用来调整两类分类属性在目标函数中的具体的比例，以避免偏向任何一方特征。对数值属性的数据采用欧几里得距离平方，对类属性差异度定义为：σ（,）=,（3.2），则接下来分类的目的就是使得3.1式值最小。从而分出非常相似的几个类别。已达到对相近游戏玩家进行匹配分类的目的。

然后游戏匹配系统，在相近的游戏玩家组成的范围内对游戏玩家随机组队，这样既保证了游戏玩家竞技水平相近，又保证了游戏玩家的类别一致。从而在保证游戏质量的同时，大大提升了游戏玩家的游戏体验。

3.2 系统结构模块

通过对上一节问题的具体分析，本论文研究的游戏匹配系统可由为三个模块组成，分别是数据采集与预处理模块、基于K-prototype的游戏数据挖掘和游戏玩家的匹配规则。游戏匹配系统的模块结构如图1所示。

游戏匹配系统

游戏匹配规则

基于K-prototypes算法的游戏数据挖掘

匹配游戏数据采集与预处理

图1：游戏匹配系统模块结构框图

游戏数据采集与预处理由采集玩家游戏数据与对玩家游戏行为分类组成，玩家的游戏数据主要是指玩家进行游戏的过程中所赢得游戏场数与游戏时长以及一局游戏的KDA。将其胜率量化为连续型数值。而游戏玩家的行为则是指玩家的消费水平，队友对其评价，敌方对自己的游戏行为评价，自己本身游戏素质。将这些行为属性量化为10个等级，作为游戏玩家的离散属性行为。

游戏数据挖掘模块则是基于K-prototypes算法，将采集到的游戏数据进行聚类的过程。根据3.1节所描述的算法流程，对经过处理的游戏玩家的数据进行分类。

游戏玩家匹配规则则是在数据分类的基础之上，在一局游戏匹配阶段，先从当前游戏玩家所属类别相近的人群中优先组队，如果没有属性相近的，则尽量选择属性相同的相似游戏玩家进行组队。

3.3 本章小结

本章主要通过对游戏匹配问题的分析，建立了游戏匹配系统的大致模块结构。即主要包括游戏数据采集与预处理模块，基于K-prototypes算法的游戏数据挖掘模块以及得到处理过后的数据从而进行游戏匹配的游戏匹配系统模块。

四．游戏匹配系统的结构与实现

本章对此游戏匹配系统的具体细节，将使用Python语言进行模拟实现。首先介绍Python的历史和发展，接下来介绍游戏玩家数据的采集与预处理并给出游戏数据挖掘的算法注释。文末附录将列出核心代码。

4.1 Python简介

Python 是由 Guido van Rossum 在八十年代末和九十年代初，在荷兰国家数学和计算机科学研究所设计出来的。Python 本身是一个高层次的结合了解释性、编译性、互动性和面向对象的脚本语言。像 Perl 语言一样，Python 源代码同样遵循 GPL(GNU General Public License)协议。

Python具有丰富和强大的库。而且其还被称为胶水语言，能够把用其他语言制作的各种模块（尤其是C/C++）很轻松地联结在一起。常见的一种应用情形是，使用Python快速生成程序的原型（有时甚至是程序的最终界面），然后对其中有特别要求的部分，用更合适的语言改写，比如3D游戏中的图形渲染模块，性能要求特别高，就可以用C/C++重写，而后封装为Python可以调用的扩展类库。需要注意的是在使用扩展类库时可能需要考虑平台问题，某些可能不提供跨平台的实现。

Python拥有一个强大的标准库。Python语言的核心只包含数字、字符串、列表、字典、文件等常见类型和函数，而由Python标准库提供了系统管理、网络通信、文本处理、数据库接口、图形系统、XML处理等额外的功能。Python标准库命名接口清晰、文档良好，很容易学习和使用。

基于以上特点，Python因而有着非常广泛的应用，广泛的适用于图形处理、数学处理、文本处理、数据库编程、网络编程、Web编程、多媒体应用、pymo引擎和黑客编程等等方面。

4.2 游戏玩家数据采集与预处理

对游戏玩家的游戏对局情况进行采集与预处理是整个匹配系统最重要的环节之一，本论文设计的游戏匹配系统，所采集的数据包括两个方面。一方面是反应游戏玩家竞技水平的竞技数据。另一个方面是游戏玩家的游戏行为分类属性。

对于第一类数据类型的数据采集，本论文参照ELO评级系统来进行设计，ELO评级算法是一个基于统计学的方法，系统把比赛结果的概率看作是两名选手的水平差异的函数。此函数在原始系统中为高斯密度函数。选手的等级分数计算按照如下步骤来进行：在每局比赛之前，根据参赛选手的当前分数，比赛结果的期望被计算出来，这个期望值对之后选手的加分和丢分有着重大的影响，简单说来，排名更高的选手被认为应该赢得比赛，因此如果高排位选手赢得了比赛，他只会加较少的分数(相对于对手获胜的情况)，同时如果他输掉了比赛，他会损失很多的分数。 ELO 排名系统选手的数量和水平间呈现正态分布，如图 2 所示，从图中可以看出，绝大部分玩家都位于普通水平段，而顶尖玩家和初学者则位于分布的两端。 根据大数定律，一个玩家或者队伍所完成的比赛越多，系统对他的评级也就越精确。 统计证据显示，系统的评级分数会在有效分数周围最多 200 分范围内波动。分数反映了玩家相对于其他参与者的相对位置，由于大家都使用统一的评分模型，在玩家比赛足够多时， 这个分数作为玩家的水平值是可行的。

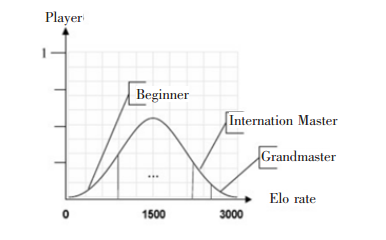


图2： ELO评级系统的玩家水平分布

另一方面，对玩家游戏行为的类别属性数据分析则可以采用离散型数据进行统计分类属性变量。对于消费玩家的具体属性，可根据玩家的具体消费信息行为分成【0-9】十个等级类别。对于玩家喜欢的游戏角色，结合相应游戏角色数目进行归一化处理，也分成【0-9】十个类别。另外，对于游戏玩家素质的数据采集，则要通过游戏玩家在游戏之中的具体表现以及游戏结束后队友对游戏玩家的相应的评价来确定，同时也利用归一化处理，将游戏行为分为【0-9】十个类别来进行处理。

综合这两个方面，就得到了每个游戏玩家的数据模型=[]。其中为连续性数值变量即为ELO评分，另外三个为离散型分类属性变量。

4.3 游戏数据挖掘

由于本数据集中既有连续型变量，又有类别变量，因此考虑使用K-prototype算法来进行聚类。K-prototype的本质就是用K-means处理连续型变量，

用K-modes处理类别变量，然后两类结果合起来即为prototype.在计算混合类型数据簇中心点时，连续型变量的均值即为连续型数据原型的中心，类别变量中取值频率最高的属性即为类别型数据原型的中心。两者相结合即为簇的中心。在计算相似度量时，K-prototype使用相异度（Dissimilarity）来衡量数据点之间的相似程度。

在K-prototype算法中混合属性的相异度分为数值属性和分类属性，两者分开求然后相加。

对于数值属性:使用欧几里得距离

对于分类属性:使用海明威距离，即属性值相同，为0 ；属性值不同，为1。

因此数据点和某一簇之间的相异度为：

其中γ为分类属性的权重因子

要最小化的目标函数为

其中 = 1 是在centroid（l）里面， 反之则= 0

以上为K-prototype计算距离，中心点以及优化目标函数的基本原理，本论文中直接调用Python中kmode包里的Kprototypes函数进行计算，其计算原理和以上基本相同。在具体作业中，首先将连续型变量和类别变量分开，然后对连续型变量进行标准化统一化，然后再将两类数据合起来放入Kprototype函数中进行计算。函数的输入为样本数据和聚类数，输出为样本类别标签、各个类别样本数以及聚类中心。

具体步骤是：

输入：聚类簇的个数k，权重因子γ

输出：产生好的k个聚类。

步骤：从数据集中随机选取k个对象作为初始的k个簇的原型，遍历数据集中的每一个数据，计算数据与k个簇的相异度。然后将该数据分配到相异度最小的对应的簇中，每次分配完毕后，更新簇的原型。然后计算目标函数，对比目标函数值是否改变，循环直到目标函数值不再变化为止。

典型函数说明：

def kprototype(filename, num\_clusters)：#输入数据

num\_data = 3000#若输入完整数据库30000个数据项可能计算时间会过久，故先以3000个数据点作为例子。

X\_original = np.genfromtxt(filename, dtype=object, delimiter=',')[1:num\_data,:]

#normalize连续型变量

X\_categorical = X\_original[:,0:10]

X\_numerical = normalize(X\_original[:,11:], norm='l2')

#对于连续型变量，如果数量较多的话可以考虑使用PCA降维

#X\_numerical = PCA(n\_components=1).fit\_transform(X\_numerical)

X = np.concatenate((X\_categorical, X\_numerical), axis=1)

#开始训练，默认权重u为0.5 连续型变量值的标准差

kproto = kprototypes.KPrototypes(n\_clusters=num\_clusters, init='Cao', verbose = 2)

clusters = kproto.fit\_predict(X, categorical=[0,1,2,3,4,5,6,7,8,9])

print '\n'

#输出每个数据点所属的聚类标签

print 'Labels of each data point: \n', kproto.labels\_, '\n'

#输出各个类别的样本数

for i in range(num\_clusters):

num\_sample = 0

for n in kproto.labels\_:

if i == n:

num\_sample += 1

print 'numbers of samples in the', i, 'cluster: ', num\_sample

print '\n'

print 'Clusters: \n', kproto.cluster\_centroids\_#输出聚类中心

print 'Cost: ', kproto.cost\_#输出目标函数成本

4.4 规则库建立及系统实现

根据采集得到的游戏玩家数据以及对玩家游戏数据处理和数据挖掘后得到的分类信息根据相应的规则对游戏玩家的匹配做出判断。当一个玩家进入游戏，开始进行游戏匹配的时候，系统首先获取所有在线等待匹配玩家的个人游戏数据模型信息。对当前等待游戏玩家进行数据预处理和数据挖掘，将在线玩家根据竞技水平实力和行为模式进行分类。得到相近性的玩家类别。然后，从类别中采用距离近优先原则选取最近的几个玩家组成一局游戏。这样，就可以实现在保证游戏质量的同时，也使得游戏玩家有着更好的游戏体验。

4.4 本章小结

本章是基于第三章对问题的建模而对问题解决所构建的模块进一步的细化与代码实现，在介绍了Python语言的相关背景之后，首先对数据采集与预处理模块进行了深入的更近一步的分析，接着利用Python构建了一个简单基于K-prototypes算法的简单demo，最后对聚类之后的到数据进行规制库的建立和系统实现。

五．总结与展望

5.1 总结

本文通过对当前多人对局游戏匹配系统特点与原理的分析，发现当前游戏匹配系统存在的不足之处。并且接着通过对模拟的游戏匹配系统进行分析，发现玩家游戏数据信息呈现混合数据簇类型，从而设计出基于K-prototypes算法的游戏匹配系统。并且由于Python语言的简单易用性，所以采用Python语言，结合K-prototypes算法库，最后给出了游戏匹配系统模拟的简单Python程序实现。

5.2展望

到目前为止，游戏匹，配系统正在不断迈向新的台阶。随着一批又一批大型好玩的游戏不断上线，自动匹配系统在游戏中的地位也逐渐变得重要。一个好的游戏匹配系统几乎可以左右游戏玩家的竞技游戏体验。所以对自动匹配系统的探索还应该继续下去，不断地去探索去实践，从而开发出更多让玩家满意的游戏自动匹配系统。

虽然如今游戏自动匹配系统日益完善，但是其仍然面临许多的问题，诸如游戏匹配时间不稳定（LOL或DOTA中5个人一起匹配的时候，要经过漫长的等待），游戏匹配对手不稳定（菜鸟玩家匹配到高玩），完美匹配游戏的巨大计算开销，玩家行为对匹配系统的破坏等等问题。因而未来对于游戏匹配系统的研究还有许多难题亟待解决。

例如，如何取舍匹配时间和匹配质量，如何降低匹配过程的计算开销，如何评价一场对局游戏的质量，如何将玩家模型应用到游戏匹配中等等问题。

总而言之，这些问题往往涉及到方方面面，需要来自人工智能、游戏设计、人机交互、心理学等等多个研究领域的研究，同时这也是一个多学科的交叉研究平台。解决这样的问题收获所带来的效益无疑是巨大的：在为玩家提供高质量的游戏对局的同时，避免了游戏玩家因为游戏本身之外的原因而放弃游戏，并且通过精准的匹配系统，带给玩家更好的游戏体验。对人类交互与娱乐机制的更深层次理解都将成为可能。

**参考文献**

[1] 杨博等. pt - 复杂网络聚类方法[J]. Journal of Software 软件学报, 2009,1,：5-10.

[2] 乔树文，陈剑军.基于竞技排名模型实证研究与分析[J].吉林师范大学学报（自然科学版）.2010,8(3):77-81.

[3] Everit B.Cluster Analysis[C],In:Proceedings of I2nd ed,Halsted Press,1981

[4] 李广斌，梁久祯.分级聚类与K-均值核聚类相结合的聚类算法[J].计算机科学，2006.10.31

[5] Guo Y,Shen S,Visser O,et al.An Analysis of Online Match Based Games[C].In:Haptic Audio Visual Environments and Games(HAVE),2012 IEEE International Workshop.Munich:IEEE,2012:134~1393

[6] Butcher C.E pluribus unum:Matchmaking in Halo 3[R].Presented at the Game Developers Conference.San Francisco,CA,Feb.19~22,2008

[7] Kaiser E, Feng W. Player Rating: a Reputation System for Multiplayer Online Games [C]. In: Proceedings of the 8th Annual Workshop on Network and Systems Support for Games: NetGames 2009. Paris: IEEE Press, 2009:8

[8] Pedersen C, Togelius J, Yannakakis G N. Modeling Player Experience in Super Mario Bros[C]. In: Proceedings of the IEEE 2009 Symposium on Computational Intelligence and Games: CIG'09. Milan, Italy: IEEE, 2009:132~139

[9] 徐钟济. 蒙特卡罗方法. 上海科学技术出版社, 1985.

[10] 邢文训，谢金星. 现代优化计算方法. 清华大学出版社, 1999.

致谢

逝者如斯，不舍昼夜，四次春去春又来，岁月稍纵即逝。此时，回头想想这段求学之路，时而喜悦，时而惆怅。人生处处是驿站，已是挥手作别之时，在此，向所有帮助过我的人献上我最诚挚的谢意！饮其流时思其源，成吾学时念吾师。”至此论文完成之际，谨向我尊敬的导师致以诚挚的谢意。非常幸运能够成为您的学生，在这短短的三年里，聆听着您孜孜不倦的教诲，感受着您严谨进取的治学精神和乐观向上的生活态度，我不仅体会到知识与研究的魅力，也学会了许多做人的道理。毕业在即，在此谨向您表示我最衷心的感谢，同时，祝您工作顺利，合家欢乐！“何当共剪西窗烛，却话巴山夜雨时。”本科期间与我朝夕相处的同学是我最值得珍惜的宝贵财富，感谢所有的同学，很幸运能够认识你们，但愿我们的友情长久，祝福我们的未来美好！最后，衷心的感谢在百忙之中抽出时间审阅论文的专家教授老师，你们辛苦了！

附录

K-prototypes算法的Python实现：

import pandas as pd

import numpy as np

from kmodes import kmodes

from kmodes import kprototypes

from sklearn.preprocessing import normalize

from sklearn.decomposition import PCA

def kprototype(filename, num\_clusters):

#输入数据

num\_data = 3000

X\_original = np.genfromtxt(filename, dtype=object, delimiter=',')[1:num\_data,:]

#normalize连续型变量

X\_categorical = X\_original[:,0:10]

X\_numerical = normalize(X\_original[:,11:], norm='l2')

#对于连续型变量，如果数量较多的话可以考虑使用PCA降维

#X\_numerical = PCA(n\_components=1).fit\_transform(X\_numerical)

X = np.concatenate((X\_categorical, X\_numerical), axis=1)

#开始训练，默认权重u为0.5 \* 连续型变量值的标准差

kproto = kprototypes.KPrototypes(n\_clusters=num\_clusters, init='Cao', verbose = 2)

clusters = kproto.fit\_predict(X, categorical=[0,1,2,3,4,5,6,7,8,9])

print '\n'

#输出每个数据点所属的聚类标签

print 'Labels of each data point: \n', kproto.labels\_, '\n'

#输出各个类别的样本数

for i in range(num\_clusters):

num\_sample = 0

for n in kproto.labels\_:

if i == n:

num\_sample += 1

print 'numbers of samples in the', i, 'cluster: ', num\_sample

print '\n'

#输出聚类中心

print 'Clusters: \n', kproto.cluster\_centroids\_

#输出目标函数成本

print 'Cost: ', kproto.cost\_

#输入数据集和类别数，开始计算K-prototypes

filename = 'samples.csv'

num\_cluster = 3

kprototype(filename, num\_cluster)