基于 XGBOOST 的游戏用户留存度分析及预测

摘　要

在游戏市场竞争日益激烈的背景下，提升用户留存度成为游戏公司实现盈利和可持续发展的关键。本文基于 SAS 高校数据分析大赛提供的数据集，深入分析游戏用户留存度的影响因素，并构建预测模型以优化用户留存策略。研究分为三个主要部分：

第一部分，针对用户留存度的分群分析。通过**聚类分析方法**，将用户分为高、中、低三个留存度群体，并对各群体的客户画像进行对比分析。结果显示，高留存度用户具有较高的游戏参与频率、更长的游戏时长以及更高的成就解锁率，而低留存度用户则表现出较低的参与度和游戏投入。此外，不同留存度群体在性别、年龄、地域和游戏类型偏好上也存在差异。

第二部分，建立用户留存度预测模型。本文构建了多个分类模型，包括判别分析、KNN、随机森林和 **XGBoost**，并通过准确率、灵敏性和特异性等指标对模型进行评估。经过对比分析，XGBoost 模型表现最优，其准确率达到 **0.9118**，灵敏性为 **0.9023**，特异性为 **0.9513**，能够有效预测用户是否留存。

第三部分，基于最优模型的预测结果，对 1 万个样本的留存率进行计算，并提出针对性的用户留存提升策略。研究发现，增加每周游戏会话次数、优化游戏内成就系统、提升游戏体验的**个性化设计**等措施，能够显著提高用户留存度。此外，结合不同留存度群体的特征，建议游戏公司针对核心玩家群体提供更具挑战性的内容，针对休闲玩家群体优化游戏的轻松性和趣味性，以满足不同用户的需求。

综上所述，本文通过数据分析和建模，为游戏公司提供了基于数据驱动的用户留存提升策略，旨在通过**精细化运营**改善游戏体验，降低用户流失率，提升用户生命周期价值，最终助力游戏公司实现更高的盈利和**可持续发展**。

关键词：用户留存度 XGBoost 聚类分析 分类模型 游戏数据分析 精细化运营

一、 问题重述

1.1 背景资料

随着游戏行业的竞争日益激烈，游戏公司面临着高额的开发成本和迅速实现盈利的压力。在这种背景下，如何提升游戏的销量和用户留存度变得尤为重要。《黑神话：悟空》自正式发售以来，曾一度达到300万的最高同时在线人数，目前在线人数稳定在数十万级别，显示出强劲的市场需求。用户留存度高，意味着用户对游戏的粘性强，能够有效降低流失率，提升用户生命周期价值，进而推动口碑传播，为公司带来持续的盈利和发展。因此，游戏公司需要深入分析用户的游戏行为，通过数据分析和精细化运营，改善游戏体验并提升用户留存度。

1.2 问题描述：

我们通过分析相关数据，运用数学思想建立模型，来研究游戏用户留存度相关的下列问题：

（1）基于用户的留存度，进行用户分群，并对不同群体的客户画像进行对比分析，挖掘各个群体的特点。

（2）建立一个分类模型，预测用户是否会响应（即是否留存），并对模型的效果进行评估。如果建立了多个模型，需要进行模型比较，并选择最优的预测模型。

（3）基于最优模型，对提供的1万个样本进行留存预测，计算预测的留存率。同时，结合分析结果，制定相应的策略以提高用户留存度，并向游戏公司提出改善游戏整体体验的建议。

通过以上分析和建模工作，游戏公司可以根据不同的用户群体特点和留存预测，调整产品和运营策略，从而提高用户粘性、减少流失、优化用户体验，最终提升公司的盈利能力。

二、 问题分析

2.1 问题一的分析

问题一的核心在于通过对用户留存度的分析，将用户分为不同群体，并深入理解各个群体的行为特征。通过聚类分析或分类模型，得出不同留存度群体的客户画像，并为后续优化用户留存和提升整体游戏体验奠定基础。具体思路如下：

首先基于用户的留存度将用户分为高中低三个群体。为了进一步深入理解不同群体的特征，对每个留存度群体进行聚类分析，采用 PAM（Partitioning Around Medoids）方法。PAM 方法适用于处理混合类型数据，能够根据用户的行为数据（如活跃度、付费情况等）将其划分为多个子群体。在聚类分析后，我们将对每个群体进行数值型特征分析和分类变量分析，具体包括对每个群体的行为模式、人口统计特征等维度进行描述统计，分析各聚类之间的差异性，并通过统计方法（如方差分析、卡方检验等）比较不同留存度群体（High、Medium、Low）在聚类后的统计特征上的显著差异。最终，我们将通过对不同群体的客户画像进行对比分析，帮助公司识别出高留存度群体的核心特征，从而为后续的精细化运营和用户留存策略提供数据支持。

2.2 问题二的分析

问题二的核心在于，将不同类别的玩家群体提取特征以便后续预测，困难之处在于多变量的数据可能其在几个变量之间存在某种相加、拮抗、增强效应等一系列合成影响，需要寻找更合适的方法把握其中线性与非线性的关系。

数据分区：将 DS_train 数据集以 7:3 划分为训练集和验证集，并进行分层抽样以保持 Target 列的分布。

使用以下分类模型进行训练：

判别分析：其可以根据某一研究对象的各种特征值判别其类型归属，适用于多变量统计分析，可作为基准模型。

KNN 算法：其本身具有物以类聚的思想，将指定数量的空间度量上最为靠近的数据点进行归纳，将群体内出现最多的标签赋予该样本，实现归类。

随机森林（Random Forest）：处理非线性关系并评估特征重要性。

XGBoost ：高性能分类模型，用于处理较复杂的特征关系，且由于其存在对误差迭代的过程，往往准确率和有效性比较好。

评估指标：分类模型性能使用 **Accuracy**（准确率）、**sensitivity**（灵敏性）、**specificity**（特异性）比较所有模型的评估指标，选择最优模型。

2.3 问题三的分析

沿用问题二中选择的最优模型，并结合变量重要性条形图，为游戏开发企业提供有序、有效、性价比高的发展意见，以促进买断制游戏的购入率以及内购游戏的游戏寿命，间接促进可持续收益。

三、 模型假设

1. 信息收集的准确度得到保证。
2. 信息收集的范围足够全面，不存在偏向收集与排斥收集。
3. 留存度定义与主流定义相近。
4. 成就解锁条件绝大多数并不和游玩时间硬性挂钩

四、 符号说明

符号	说明
SPK	每周进行游戏会话的次数
ASD	每周每个游戏会话的平均持续时间
ACHV	用户解锁的成就数量
level	用户的游戏等级
C_k	第k个类别的用户留存度

五、 问题一模型的建立与求解

5.1 数据预处理

首先进行数据预处理，检查并剔除数据中的异常值，缺失值。同时，为了确保在建模过程中，将所有设计分列变量的列都转为因子，进行哑变量编码，使其可以在模型中作为分类特征输入。

5.2 模型的建立

通过“Target”字段，对用户进行高、中、低三个级别的分群后，做出各个字段的多变量分组描述性统计，如下**表 1** 所示：

表 1：分组特征描述性统计表

Name	Levels	High (N=7036)	Low (N=7072)	Medium (N=13310)	p
Age	Mean ± SD	32.0 ± 10.1	32.0 ± 10.0	32.1 ± 10.1	.687
Sex	Female	2837 (40.3%)	2798 (39.6%)	5358 (40.3%)	.571
	Male	4199 (59.7%)	4274 (60.4%)	7952 (59.7%)	

Name	Levels	High (N=7036)	Low (N=7072)	Medium (N=13310)	p
Loc	Asia	1441 (20.5%)	1401 (19.8%)	2675 (20.1%)	.898
	Europe	2093 (29.7%)	2150 (30.4%)	3968 (29.8%)	
	Other	675 (9.6%)	705 (10%)	1305 (9.8%)	
	USA	2827 (40.2%)	2816 (39.8%)	5362 (40.3%)	
GameType	Action	1439 (20.5%)	1404 (19.9%)	2693 (20.2%)	.333
	RPG	1315 (18.7%)	1432 (20.2%)	2657 (20%)	
	Simulation	1461 (20.8%)	1444 (20.4%)	2627 (19.7%)	
	Sports	1409 (20%)	1394 (19.7%)	2681 (20.1%)	
	Strategy	1412 (20.1%)	1398 (19.8%)	2652 (19.9%)	
PlayTimes	Mean ± SD	12.1 ± 6.9	12.1 ± 6.9	11.9 ± 6.9	.140
InGame	0	5554 (78.9%)	5703 (80.6%)	10652 (80%)	.036
	1	1482 (21.1%)	1369 (19.4%)	2658 (20%)	
DD	Easy	3458 (49.1%)	3524 (49.8%)	6640 (49.9%)	.648
	Hard	1454 (20.7%)	1414 (20%)	2631 (19.8%)	
	Medium	2124 (30.2%)	2134 (30.2%)	4039 (30.3%)	
SPK	Mean ± SD	14.3 ± 3.9	4.6 ± 4.9	9.6 ± 4.8	<.001
ASD	Mean ± SD	131.9 ± 34.2	66.4 ± 48.6	89.7 ± 43.8	<.001
Level	Mean ± SD	50.8 ± 28.7	46.1 ± 28.3	50.7 ± 28.5	<.001
ACHV	Mean ± SD	25.0 ± 14.4	22.5 ± 14.1	25.3 ± 14.4	<.001

通过描述性统计发现，在仅考虑留存度的情况下，除了 SPK（每周进行游戏会话的次数）、ASD（每周每个游戏会话的平均持续时间）、level（用户的游戏等级）以及 ACHV（用户解锁的成就数量）这几个游戏行为相关的变量以外，其他字段（如性别、年龄等）对游戏用户的留存率影响及相关性不大。其中高留存度群体的游戏参与频率显著高于低留存度群体和中留存度群体（$p<0.001$）；同时高留存度群体的每次游戏持续时间也明显高于低留存度群体和中留存度群体（$p<0.001$）；高留存度和中留存度用户的游戏等级较高，低留存度群体的游戏等级较低；高留存度和中留存度用户的成就解锁数量较高，低留存度群体的成就解锁数量略低（$p<0.001$）。

接下来对用户分群进行进一步的聚类分析。首先需要选择可能对识别和理解数据中不同观测值分组有重要影响的变量，并将变量标准化。

由于 K 均值聚类方法是基于均值的，对异常值较为敏感。我们选择了更稳健的 PAM，即基于中心点的划分方法。与其用质心（变量均值向量）表示类，用一个最有代表性的观测值来表示（称为中心点）。PAM 可以容纳混合数据类型，并且不仅限于连续变量。

PAM 的算法如下：

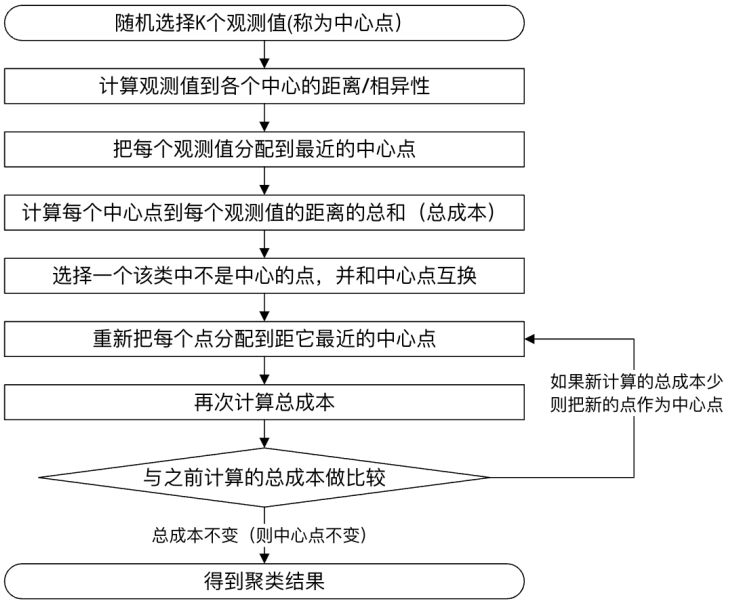

图 1：聚类分析流程图

5.2.1 对于高留存度用户群体的聚类分析结果

表 2　高留存度用户统计描述表

Name	Levels	1 (N=3593)	2 (N=1362)	3 (N=2081)	p
Age	Mean ± SD	31.9 ± 10.0	32.5 ± 10.0	31.8 ± 10.2	.066
Sex	Female	769 (21.4%)	451 (33.1%)	1617 (77.7%)	<.001
	Male	2824 (78.6%)	911 (66.9%)	464 (22.3%)	
Loc	Asia	552 (15.4%)	288 (21.1%)	601 (28.9%)	<.001
	Europe	1010 (28.1%)	397 (29.1%)	686 (33%)	
	Other	353 (9.8%)	137 (10.1%)	185 (8.9%)	
	USA	1678 (46.7%)	540 (39.6%)	609 (29.3%)	
GameType	Action	719 (20%)	278 (20.4%)	442 (21.2%)	.510
	RPG	690 (19.2%)	235 (17.3%)	390 (18.7%)	
	Simulation	719 (20%)	308 (22.6%)	434 (20.9%)	
	Sports	730 (20.3%)	268 (19.7%)	411 (19.8%)	
	Strategy	735 (20.5%)	273 (20%)	404 (19.4%)	
PlayTimes	Mean ± SD	12.0 ± 7.0	12.1 ± 6.8	12.1 ± 6.9	.869

Name	Levels	1 (N=3593)	2 (N=1362)	3 (N=2081)	p
InGame	0	3593 (100%)	0 (0%)	1961 (94.2%)	<.001
	1	0 (0%)	1362 (100%)	120 (5.8%)	
DD	Easy	1425 (39.7%)	620 (45.5%)	1413 (67.9%)	<.001
	Hard	799 (22.2%)	294 (21.6%)	361 (17.3%)	
	Medium	1369 (38.1%)	448 (32.9%)	307 (14.8%)	
SPK	Mean ± SD	15.1 ± 3.5	14.5 ± 3.9	12.9 ± 4.1	<.001
ASD	Mean ± SD	126.0 ± 35.1	129.2 ± 34.7	144.0 ± 29.0	<.001
Level	Mean ± SD	51.9 ± 28.7	52.9 ± 28.3	47.6 ± 28.6	<.001
ACHV	Mean ± SD	21.7 ± 13.9	24.1 ± 14.4	31.4 ± 13.0	<.001

首先群体间的年龄差异无显著性（$p = 0.066$），年龄分布相对均匀。总体来看，这些高留存度用户的年龄分布较为广泛，表明不同年龄段的用户都可能对游戏保持较高的参与度和忠诚度。同时不同群体对游戏类型的偏好差异不显著。

- 群体 1：核心深度玩家群体

  性别：男性占主导（78.6%）

  地域：主要来自美国（46.7%）

  游戏行为：偏好较平衡的游戏类型（动作、角色扮演、策略等）。此类用户对竞技和有更高的需求，喜欢通过游戏积累经验和解锁成就。SPK 和 ASD 都较高，表明这些用户频繁参与游戏且每次游戏时长适中。他们的 Level 和 ACHV 也表现出较高的游戏进程和成就解锁数量，注重任务、成就系统和社交互动。

- 群体 2：高留存度均衡游戏群体

  性别：性别均衡。

  地域：美国和欧洲用户较多。

  游戏行为：多样化，但偏好中等难度的游戏设置，活跃度相对较低，但是 Level 相对其他群体较高。

- 群体 3：轻松休闲玩家群体

  性别：女性占主导（77.7%）

  地域：亚洲和欧洲用户较为突出。

  游戏行为：Level 较低但是成就解锁数量最多（均值为 31.4），偏好轻松、有趣且富有成就感的游戏内容。大多选择较低难度的游戏（67.9%选择易。这些用户可能更倾向于享受轻松、愉悦的游戏体验。

5.2.2 对于中留存度用户群体的聚类分析结果

表 3　中留存度用户统计描述表

Name	Levels	1 (N=6092)	2 (N=2560)	3 (N=4658)	p
Age	Mean ± SD	31.9 ± 10.1	32.2 ± 10.1	32.3 ± 10.0	.239
Sex	Female	774 (12.7%)	986 (38.5%)	3598 (77.2%)	<.001
	Male	5318 (87.3%)	1574 (61.5%)	1060 (22.8%)	
Loc	Asia	1579 (25.9%)	518 (20.2%)	578 (12.4%)	<.001
	Europe	2026 (33.3%)	811 (31.7%)	1131 (24.3%)	
	Other	560 (9.2%)	227 (8.9%)	518 (11.1%)	
	USA	1927 (31.6%)	1004 (39.2%)	2431 (52.2%)	
GameType	Action	1269 (20.8%)	508 (19.8%)	916 (19.7%)	.277
	RPG	1199 (19.7%)	507 (19.8%)	951 (20.4%)	
	Simulation	1218 (20%)	510 (19.9%)	899 (19.3%)	
	Sports	1224 (20.1%)	487 (19%)	970 (20.8%)	
	Strategy	1182 (19.4%)	548 (21.4%)	922 (19.8%)	
PlayTimes	Mean ± SD	13.3 ± 6.8	11.9 ± 7.0	10.2 ± 6.7	<.001
InGame	0	6088 (99.9%)	2 (0.1%)	4562 (97.9%)	<.001
	1	4 (0.1%)	2558 (99.9%)	96 (2.1%)	
DD	Easy	2977 (48.9%)	1309 (51.1%)	2354 (50.5%)	.201
	Hard	1211 (19.9%)	507 (19.8%)	913 (19.6%)	
	Medium	1904 (31.3%)	744 (29.1%)	1391 (29.9%)	
SPK	Mean ± SD	9.6 ± 4.8	9.7 ± 4.8	9.4 ± 4.6	.074
ASD	Mean ± SD	88.1 ± 44.2	87.8 ± 42.6	92.9 ± 43.8	<.001
Level	Mean ± SD	58.0 ± 27.2	51.2 ± 28.6	41.0 ± 27.3	<.001
ACHV	Mean ± SD	29.2 ± 13.7	25.4 ± 14.4	20.1 ± 13.8	<.001

- 群体 1：中等核心玩家群体

  性别：男性占主导（87.3%）。

  地域：主要来自美国、欧洲，全球化特征明显。

  游戏行为：该群体倾向于深入体验游戏，具有较高的参与频率（SPK）和较长的游戏时长（ASD）。他们在游戏中的等级较高，解锁的成就较多，表明这些用户注重挑战、进阶以及完成目标。偏好多元化的游戏类型，没有明显偏向某一类型。

- 群体 2：休闲玩家群体

  性别：性别均衡（男性 61.5%，女性 38.5%）。

  地域：偏向美国和欧洲，地域分布较为均匀。

游戏行为：该群体的游戏参与频率适中，游戏时长略低，表现出轻度的游戏参与度。他们的游戏等级和成就解锁数量处于中等水平，表明他们对游戏有一定兴趣，但不会投入过多的时间和精力。游戏类型偏好多样，可能寻找不太复杂且富有娱乐性的体验。

- 群体 3：中等频次玩家群体

性别：女性占主导（77.2%）。

地域：主要来自美国，较少来自亚洲和欧洲。

游戏行为：群体 3 的参与频率最低，但每次游戏的时长较长，表明他们可能更倾向于在每次游戏时投入更多的时间，进行深度体验。解锁的成就较少，游戏等级较低，表明这些用户可能对游戏内容的深度有兴趣，而不是游戏的竞争性或进度性。他们可能更喜欢沉浸式的游戏体验，较少关注游戏的任务或目标。

5.2.3 对于低留存度用户群体的聚类分析结果

表 4　低留存度用户统计描述表

Name	Levels	1 (N=3642)	2 (N=2140)	3 (N=1290)	p
Age	Mean ± SD	31.9 ± 10.1	32.2 ± 10.0	31.8 ± 10.1	.446
Sex	Female	683 (18.8%)	1553 (72.6%)	562 (43.6%)	<.001
	Male	2959 (81.2%)	587 (27.4%)	728 (56.4%)	
Loc	Asia	533 (14.6%)	586 (27.4%)	282 (21.9%)	<.001
	Europe	1001 (27.5%)	742 (34.7%)	407 (31.6%)	
	Other	386 (10.6%)	190 (8.9%)	129 (10%)	
	USA	1722 (47.3%)	622 (29.1%)	472 (36.6%)	
GameType	Action	559 (15.3%)	591 (27.6%)	254 (19.7%)	<.001
	RPG	647 (17.8%)	532 (24.9%)	253 (19.6%)	
	Simulation	756 (20.8%)	440 (20.6%)	248 (19.2%)	
	Sports	800 (22%)	333 (15.6%)	261 (20.2%)	
	Strategy	880 (24.2%)	244 (11.4%)	274 (21.2%)	
PlayTimes	Mean ± SD	11.0 ± 6.8	13.8 ± 6.7	12.2 ± 6.8	<.001
InGame	0	3569 (98%)	2134 (99.7%)	0 (0%)	<.001
	1	73 (2%)	6 (0.3%)	1290 (100%)	
DD	Easy	1779 (48.8%)	1064 (49.7%)	681 (52.8%)	.002
	Hard	695 (19.1%)	456 (21.3%)	263 (20.4%)	
	Medium	1168 (32.1%)	620 (29%)	346 (26.8%)	
SPK	Mean ± SD	3.2 ± 3.8	6.9 ± 5.6	4.5 ± 4.8	<.001

Name	Levels	1 (N=3642)	2 (N=2140)	3 (N=1290)	p
ASD	Mean ± SD	75.3 ± 49.6	51.7 ± 43.1	66.0 ± 48.1	<.001
Level	Mean ± SD	46.2 ± 28.2	45.1 ± 28.0	47.7 ± 28.8	.029
ACHV	Mean ± SD	26.2 ± 13.9	16.2 ± 12.3	22.6 ± 14.1	<.001

- 群体 1：轻度娱乐玩家群体

  性别：男性占比 81.2%。

  地域：主要来自美国（47.3%）和欧洲（27.5%），具有全球化特征。

  游戏行为：群体 1 的参与频率较低，且游戏时长较长，可能偏向于有挑战性和策略性的游戏，注重游戏的深度和策略。成就解锁数量较多，表现出较强的游戏投入和长时间的参与。偏好策略类和运动类游戏。

- 群体 2：女性轻松玩家群体

  性别：女性占比 72.6%。

  地域：亚洲用户比例较高（27.4%），这群体更可能来自亚洲地区。

  游戏行为：群体 2 的游戏频率较高，游戏时长较短，表现出典型的休闲娱乐型玩家特征。成就解锁较少，表明他们对游戏的投入较低。偏好动作类和角色扮演类游戏，倾向于娱乐性更强的内容。

- 群体 3：轻度玩家群体

  性别：男性占比 56.4%。

  地域：美国用户占比最高（36.6%）。

  游戏行为：群体 3 的游戏时长适中，参与频率也在群体之间处于中间水平，表明他们偏好偶尔但较为深度的游戏体验。游戏等级和成就解锁数量较高，显示出他们有一定的游戏投入。偏好均衡的游戏类型，不偏向某一类。

六、 问题二模型的建立与求解

6.1 数据预处理

在面对分类问题时。类别不平衡问题是指数据集中的某一类别的样本数量显著多于其他类别，导致模型在训练时更偏向于预测样本数较多的类别，从而忽略少数类。此时模型会对少数类的预测表现较差，导致其无法有效识别少数类的样本。与此同时传统的评估指标（如准确率）可能会受到数据不平衡的影响，导致评估结果

不真实。例如，若数据集中 95%的样本为负类，模型只预测负类，即使准确率达到 95%，仍然无法有效识别少数类。

在本文中，由于中留存度的用户数量为高留存度和低留存度用户数量的两倍，因此我们利用 SMOTE 方法对数据进行不平衡性的处理通过，通过插值生成新的少数类样本来增加少数类的数量，使得其与多数类的样本数量接近，减少模型在构建时产生的不当倾向性。

6.2 判别分析模型

判别分析（Discriminant Analysis）是一种用于分类的统计方法，主要目标是通过已知类别的数据（训练集），建立一个判别模型来预测新的观察值属于哪个类别。其基本思路是通过构造一个分类规则，最大化类间差异，同时最小化类内差异，从而达到将不同类别的数据区分开的目的。

6.2.1 Fisher 判别分析（LDA）

LDA 的目标是将数据投影到一个低维空间（通常是 1 维或 2 维），使得不同类别之间的距离尽可能大，而类别内部的距离尽可能小。LDA 属于监督学习方法，需要标记数据集。LDA 的主要思想是寻找最能区分不同类别的数据线性组合。

具体步骤如下：

（1）**计算每个类别的均值向量**：对每个类别，计算其特征的均值。

（2）**计算类内散度矩阵（Within-Class Scatter Matrix）**衡量类别内部的散布程度。

（3）**计算类间散度矩阵（Between-Class Scatter Matrix）**衡量各类别均值之间的散布程度。

（4）**求解特征值和特征向量**：通过类间和类内散度矩阵求解特征值和特征向量。特征向量对应了最能区分类别的方向，特征值则是衡量该方向的重要性。

（5）**投影到最优子空间**：根据得到的特征向量将数据投影到新的子空间，从而完成类别区分。

LDA 的模型建立过程如下：

我们的目标是通过给定数据集的不同字段，建立对于游戏用户的分类模型，通过其特征及游戏行为来判断其留存度。因此 LDA 模型的目标变量为 Target 字段，其目标是通过预测目标变量（如分类标签）来判断样本属于哪个类别。

在问题一中，对于总体数据的特征描述性分析，我们知道在仅考虑留存度的情况下，对游戏用户留存率有显著性影响的字段分别为：SPK（每周进行游戏会话的次

数）、ASD（每周每个游戏会话的平均持续时间）、level（用户的游戏等级）以及ACHV（用户解锁的成就数量）这几个游戏行为相关的变量。因此我们将其纳入LDA 模型，作为特征变量区分不同类别。LDA 通过线性函数来预测类别，该函数是特征变量的加权和。

首先，选择对留存度（目标变量 **Target**）有显著影响的特征变量。由于不同特征可能具有不同的量纲（如 **SPK** 是次数，**ASD** 是时间），需要对数据进行标准化处理，以消除特征量纲的影响，确保每个特征对模型的贡献均等。LDA 的目标是通过最大化类间差异和最小化类内差异来找到最佳的分割超平面。为此，需要计算两种散度矩阵：

类内散度矩阵（S_w）：表示每个类别内数据点的散布情况，即类别内部的差异性。对于每个类别，计算其特征变量的方差-协方差矩阵。

$$S_w = \sum_{i=1}^{c} \sum_{x_j \in C_i} (x_i - \mu_i)\left(x_j - \mu_j\right)^T \tag{1}$$

其中：

c是类别数，C_i是第i个类别，μ_i是其对应的均值向量，x_i是类别C_i中的样本。

类间散度矩阵（S_b）：表示类别之间的差异，即各类别均值之间的差异。通过计算每个类别均值和整体均值之间的散布，来衡量类别之间的差异性。

$$S_b = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T \tag{2}$$

其中：

N_i是第i类的样本数量，μ_i是第i类的均值向量，μ是所有样本的整体均值向量。

LDA 的核心在于通过线性组合，寻找一个新的特征空间使得类间差异最大，类内差异最小。为此，需要求解一个判别矩阵W，其目标是找到一个投影方向，使得数据在该方向上的类间差异最大。

求解线性判别矩阵W的标准方法是通过最大化类间散度矩阵和类内散度矩阵的比值。

$$W = agrmax_W \frac{W^T S_b W}{W^T S_w W} \tag{3}$$

可以通过对上式进行特征值分解，得到最优的投影矩阵 W，该矩阵能够将原始数据映射到一个低维空间。

在得到了最优的判别矩阵W之后，我们可以将原始数据投影到该低维空间上.每个样本都会有一个投影值。LDA 通过比较不同类别的投影值，来判断一个新样本的类别。

6.2.2 贝叶斯判别分析（Naive Bayes）

贝叶斯判别分析是一种基于贝叶斯理论的分类方法，作为基于概率的分类方法，Naive Bayes 假设各特征变量之间是条件独立的，这个假设简化了模型的复杂度，使其在实际应用中非常高效。它通过计算每个类别的后验概率来进行分类决策。贝叶斯判别分析与 LDA（线性判别分析）相似，但 BDA 不要求数据满足某些严格的假设（如正态分布）。其基本思想是基于已知类别的先验概率和特征的条件概率，通过贝叶斯定理计算出每个类别的后验概率，最终选择具有最大后验概率的类别作为预测结果

其主要步骤为：

（1）**计算每个类别的先验概率**：即不同类别的出现概率。

（2）**计算每个特征的条件概率**：即在给定类别的情况下特征变量取特定值的概率。

（3）**应用贝叶斯定理**：通过先验概率和条件概率计算后验概率，选择最大后验概率的类别。

贝叶斯判别分析的主要建立过程如下：

与上述 LDA 相似，我们以 Target 字段为目标向量，选择对用户留存度有显著影响的变量作为特征变量纳入模型。贝叶斯定理是贝叶斯判别分析的核心，用来计算类别的后验概率。贝叶斯定理的公式如下：

$$P(C_k|x) = \frac{P(C_k)P(x|C_k)}{P(x)} \tag{4}$$

其中，$P(C_k|x)$是给定特征x时，样本属于类别C_k的后验概率；

$P(C_k)$是类别C_k的先验概率，表示样本属于类别C_k的概率，通过训练数据中的类别概率来估计，可以表示为：

$$P(C_k) = \frac{\text{类别}\mathrm{C_k}\text{的样本数}}{\text{总样本数}}$$

$P(x|C_k)$是在类别C_k下特征x的似然函数，通常情况下，贝叶斯判别分析假设在每个类别下，特征变量符合某种特定的分布（如正态分布）；

$P(x)$是特征x的边际概率，通过对所有类别的似然函数加权求和得到，即：

$$P(x) = \sum_{k} P(C_k)P(x|C_k)$$

通过贝叶斯定理计算得到的后验概率用于确定一个样本属于哪个类别。具体地，对于一个新的样本x，计算其属于每个类别C_k的后验概率，并选择具有最大后验概率的类别作为预测类别。

$$\hat{C} = argmax_{C_k} P(C_k|x) = agrmax_{C_k} \frac{P(C_k)P(x|C_k)}{P(x)} \quad (5)$$

因为$P(x)$对所有类别相同，可以忽略不计，所以简化为：

$$\hat{C} = argmax_{C_k} P(C_k)P(x|C_k) \quad (6)$$

贝叶斯判别分析的决策边界是通过比较不同类别的后验概率来决定的。假设类别有k个，当我们计算出每个类别的后验概率后，贝叶斯判别分析将样本分配给后验概率最大的类别。

6. 3 KNN 模型

KNN 算法（K 近邻）的核心思想是：如果一个样本在特征空间中存在 k 个最相似（即特征空间中最邻近）的样本中的大多数属于某一个类别，则该样本也属于这个类别。具体步骤如下：

(1) 确定 k 值：选择一个正整数 k，表示在特征空间中查找最近的 k 个邻居。

(2) 计算距离：计算待分类样本与训练集中每个样本之间的距离。常用的距离度量方法有欧氏距离、曼哈顿距离和明可夫斯基距离等。由于数据本身具有的连续性，本次选用的欧氏距离。

(3) 查找最近邻：根据计算出的距离，找出 k 个最近的邻居。

(4) 投票决策：在分类问题中，根据 k 个最近邻的标签进行投票，多数类别的标签作为预测结果。

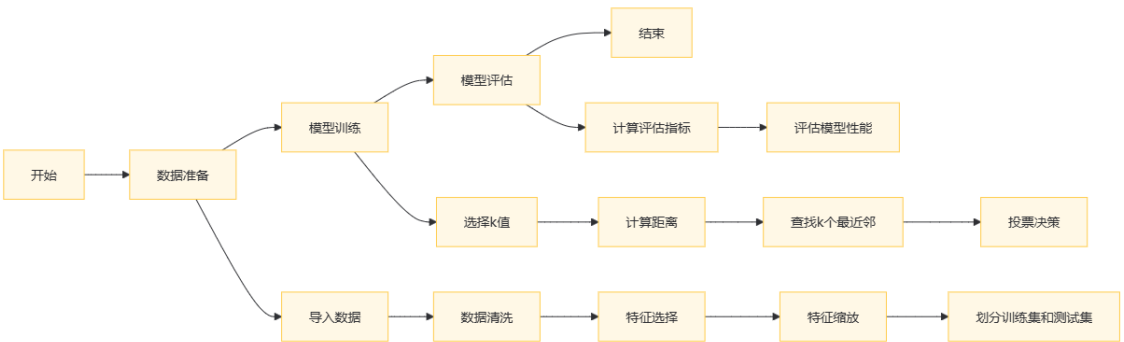

图 2　KNN 流程图

6.4 随机森林模型

随机森林（random forest）是一种组成式的有监督学习方法。在随机森林中，我们同时生成多个预测模型，并将模型的结果汇总以提升分类准确率。

随机森林的算法涉及对样本单元和变量进行抽样，从而生成大量决策树。对每个样本单元来说，所有决策树依次对其进行分类。所有决策树预测类别中的众数类别即为随机森林所预测的这一样本单元的类别。

假设训练集中共有 N 个样本单元，M 个变量，则随机森林算法如下：

(1) 从训练集中随机有放回地抽取 N 个样本单元，生成大量决策树。

(2) 在每一个节点随机抽取 m<M 个变量，将其作为分割该节点的候选变量。每一个节点处的

变量数应一致。

(3) 完整生成所有决策树，无需剪枝（最小节点为 1）。

(4) 终端节点的所属类别由节点对应的众数类别决定。

(5) 对于新的观测点，用所有的树对其进行分类，其类别由多数决定原则生成。

生成树时没有用到的样本点所对应的类别可由生成的树估计，与其真实类别比较即可得到袋外预测（out-of-bag，OOB）误差。无法获得验证集时，这是随机森林的一大优势。

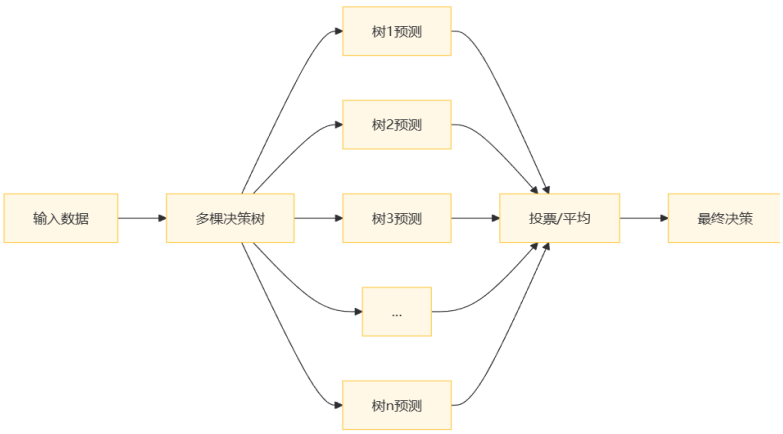

图 3　随机森林流程图

6.5 XGboost 模型

XGBoost（eXtreme Gradient Boosting）是一种基于梯度提升决策树（GBDT）的集成学习算法，其核心思想是在前一轮模型的基础上，通过拟合当前残差（预测误差）来构建新的决策树，从而逐步提升模型的预测能力。

XGBoost 的优化目标是最小化一个正则化的损失函数，该函数由两个部分组成：一部分是模型在训练数据上的预测误差，另一部分是模型复杂度的正则化项。具体来说，对于一个给定的训练集$D = \{(\mathrm{x}_1, \mathrm{y}_1), (\mathrm{x}_2, \mathrm{y}_2), \ldots, (\mathrm{x}_\mathrm{n}, \mathrm{y}_\mathrm{n})\}$，其中 x_i 是特征向量，y_i 是对应的目标值，XGBoost 的目标是最小化以下目标函数：

$$Obj(\phi) = \sum_{i=1}^{n} (y_{ihet}, y_i) + \sum_{i=1}^{K} \Omega(f_k) \tag{7}$$

其中$y_{ihet} = \sum_{i=1}^{K} f_k(x_i)$

这里，前部分是损失函数，用于衡量模型预测y_{ihet}与真实值y_i之间的差异；f_k 是第 k 棵树的预测函数；Ω 是正则化项，用于控制模型的复杂度，防止过拟合。正则化项通常包括树的深度和叶子节点的数量等

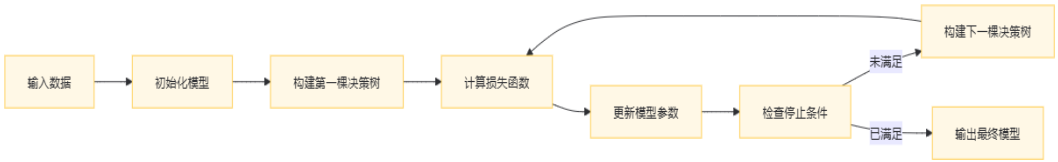

图 4　XGBoost 流程图

6.6 模型检验

在评估分类模型的性能时，除了准确率（Accuracy）之外，我们还需要考虑其他统计量，如特异性（Specificity）和灵敏性（Sensitivity），也称为召回率（Recall）。

为了描述分类模型的性能，我们使用混淆矩阵来展示了模型预测的类别与真实类别之间的关系，其中：

- **True Positives (TP)**：模型正确预测为正类的数量。
- **True Negatives (TN)**：模型正确预测为负类的数量。
- **False Positives (FP)**：模型错误预测为正类的数量。
- **False Negatives (FN)**：模型错误预测为负类的数量。

以下是几个常用的分类性能指标：

准确率表示分类模型正确预测的样本数占总样本数的比例。其计算公式为:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

特异性是指负类样本被正确识别的比例，其计算公式为

$$Precison = \frac{TP}{TP + FP}$$

灵敏性或召回率是指正类样本被正确识别的比例，可以用来评估模型在正类样本中的性能。其计算公式为:

$$Sensitivity = \frac{TP}{TP + FN}$$

综合考虑准确率、特异性和灵敏性，我们可以更全面地评估分类模型的有效性。

为确保不存在偶然性导致的差异，本次研究使用 100 次交叉验证得到 100 次相应指标，并对其求均值得到稳定指标。

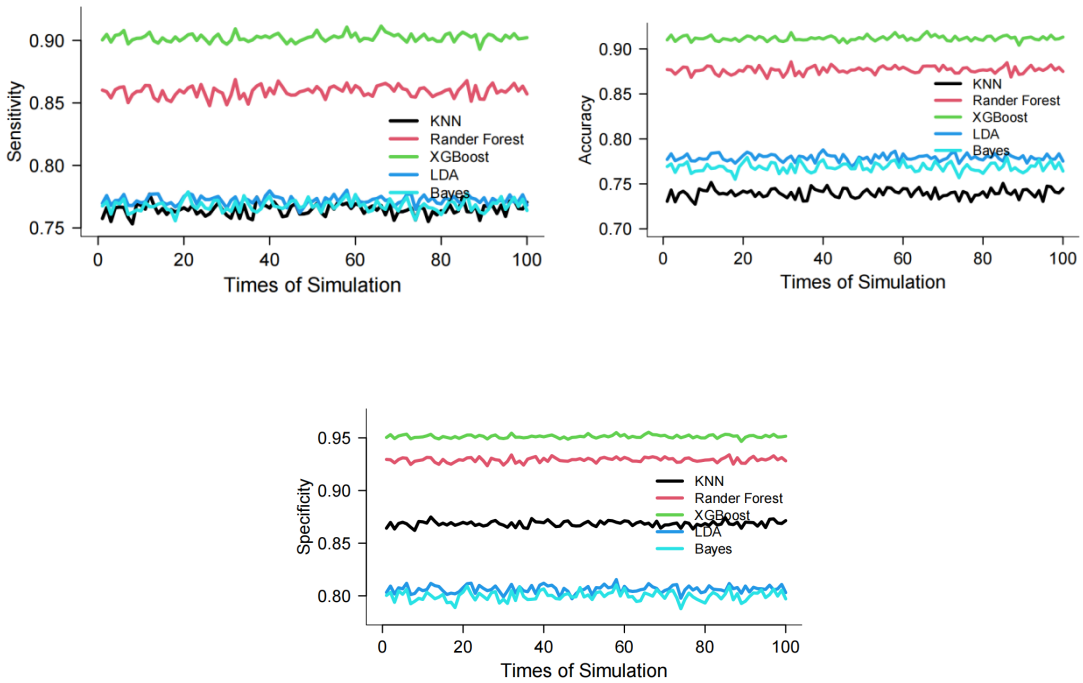

图 5　各评价指标 100 次评价图

表 5　模型指标对比表

Model	Accuracy	Sensitivity	Specificity
KNN	0.739762947	0.764646587	0.868508165
RF	0.876613178	0.859249323	0.929163603

XGBOOST	0.911821055	0.902276879	0.951315374
Bayes	0.768619013	0.767869079	0.80000862
LDA	0.779229273	0.772227255	0.806348354

在准确率（Accuracy）方面，XGBOOST 模型表现最佳，达到了 0.9118211，其次是随机森林（RF）模型，准确率为 0.8766132，而 KNN 模型的准确率最低，为 0.7397629。

在灵敏度（Sensitivity）或召回率方面，XGBOOST 模型同样表现最佳，达到了 0.9022769，随机森林模型次之，为 0.8592493，KNN 模型的灵敏度最低，为 0.7646466。

在特异性（Specificity）方面，XGBOOST 模型再次领先，达到了 0.9513154，随机森林模型为 0.9291636，而 Bayes 模型的特异性最低，为 0.8000086。

综合考虑准确率、灵敏度和特异性三个指标，XGBOOST 模型不仅在准确率上领先，而且在灵敏度和特异性上也显著优于其他模型，因此可以认为是最优选择。

七、 问题三模型的建立与求解

问题三求解上沿用了问题二中最为优异的 XGBOOST 模型。

同时，依据其得到的变量重要性制定提高用户留存度策略，为企业效益提升提供整体思路。

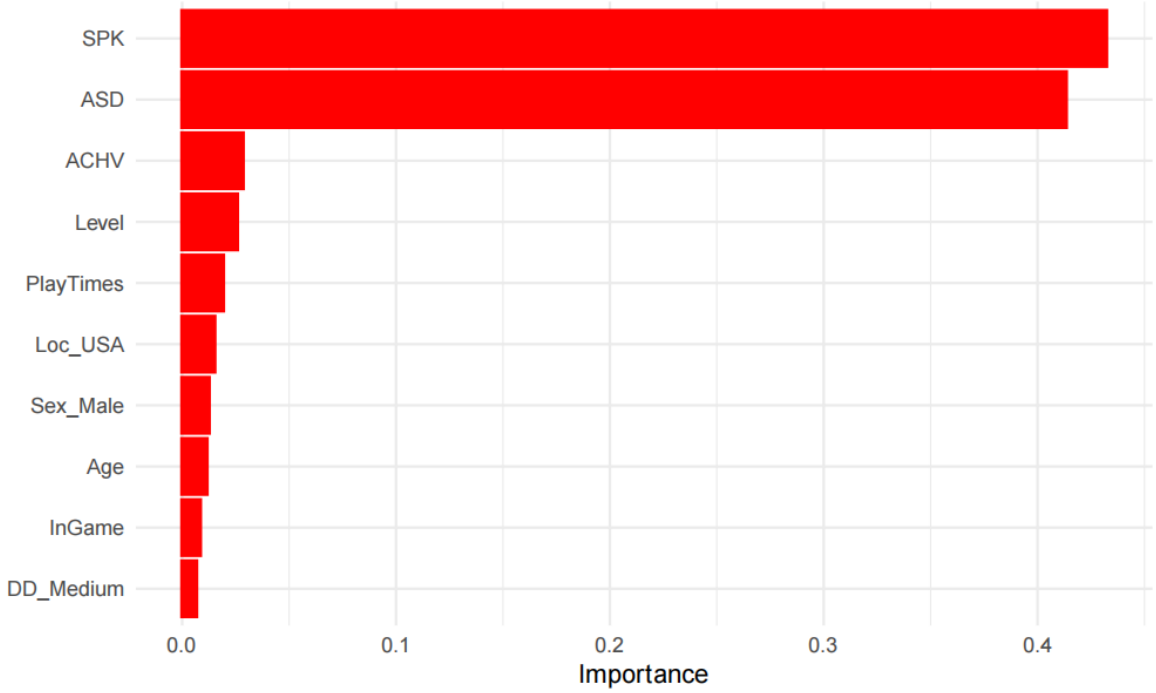

图 6　变量重要性排序图

由图中可以得知，用户的每周会话次数及其平均持续时间与游戏用户的留存度有较大相关性，虽然这并不完全代表其因果效应的指向，但也提示各位企业管理者提升此类指标对用户留存度的提升具有一定好处。

提升开启游戏会话次数，可以利用用户“来都来了”的沉没成本心理，间接提升用户留存度，过去十年内，网络游戏厂商常常使用定时定点领取特定奖励[4]（如：“游戏内体力”“游戏内货币”“游戏内抽奖次数”“定时参与获取抽奖资格”等）的方式提升用户进入游戏的次数。不过近年来由于单机游戏的异军突起，以及大量同质化应用带来的吸引力免疫效果，这套方法已经不适用于整个游戏制作行业了，根据本团队的切身体会以及网络意见的收集来看，限时不同的游戏剧情往往会比传统的奖励方法更吸引玩家。市面上已经出现游戏内时间与现实时间同步，NPC随时间不同调整游戏文本内容的产品，虽然内容体量并不算大，但买断价格也不高昂，仅在部分软件内用户购买数量已达十万有余，或许更加真实的交互设计，未来会成为游戏行业的新风向，无论是在游戏画面上，亦或者是游戏玩法上。推荐游戏制作企业可以小规模跟进类似系统，因为小体量游戏的修改复杂性不高，其在小体量游戏上颇具优势，也可成为企业后续产品的“探路石”。

除提升开启游戏会话次数以外，提升平均持续时间以及每周每会话持续时间也尤为重要，过去十年内，游戏制作厂商也尝试与提升开启游戏会话次数结合提升平均持续时间，这也对应了上文中“游戏内体力”“游戏内货币”这两类奖励的设置意图，部分厂商会尝试为游戏体力（即游戏内游玩部分内容所需入门资格）的上限进行限制，导致用户会急于将体力使用，以避免体力恢复至上限带来的资源损失。与“游戏内货币”奖励联动，部分产品也会在内部设置“拍卖行”，其内部由玩家组成，提供道具交换场所，其内部交换价格均由玩家设定，因此会出现动态变化，为以最为合适的价格购买游戏道具，部分玩家会长期处于在线状态，也可有效提升玩家留存度。但由于大量产品运用此类方法，导致多个时间段用户为完成任务疲于奔命，不得不对其中部分游戏进行舍弃，其中游戏流畅度低，玩家竞争过于激烈以及游戏内容过于复杂的游戏会被率先舍弃，已经有部分产品为保证留存度，极致简化游戏流程，专注于剧情描绘，将游戏任务挂机化操作，并且得到了良性回响。也可以通过反其道而行之的思维，将类似奖励取消，用户在疲于奔命完成其余竞品的任务后必然存在一定空窗期，此刻其大概率不会选择刚刚“上班”过的游戏，此时

我方产品就会具有低竞争环境、高竞争力的良好局面，同时也能通过竞品对比，让玩家更青睐于我方产品。

其余指标并非不重要，而是由于本身留存度计算就需要运用上文两类指标，因此其它指标作用被弱化。

成就解锁数量也会对用户的留存度进行影响，在之前的分析里也得以看出，高留存度用户 ACHV 也表现出较高成就解锁数量，注重任务、成就系统和社交互动。其中的深度核心玩家群体对应喜好的动作类、策略类游戏也与之对应，这类游戏往往以单机为主要游玩方式，给予玩家更加充足的思考时间，增加游戏可控性以助力游戏解锁成就，这无疑对游戏机制的深化进行了加固，会引导玩家更加频繁的在游戏外思考游戏内容，增加游玩兴趣。而往往多人联机对抗类游戏往往不会得到此类反馈，因为其大多数以即时对抗为主，游戏用户并不愿意在与他人对抗时分心完成成就，此类游戏设置成就往往会出现无作用甚至副作用，部分用户热爱将其游玩的游戏“白金”化（玩家定义：全成就），但由于“爱而不得”的心理，长时间如此反而会让其厌烦此机制，从而导致其寻找专人帮助其完成成就，减少了可转换收益的游玩时间，同时降低游戏理解速度，部分游戏会将完成成就或者任务解锁数量作为匹配玩家依据，导致此类玩家在获取成就后与更高游戏理解的玩家对局，降低游戏体验，但此类玩家往往是非常有价值的，对于含有内购的游戏产品，玩家内购行为往往是主要收入来源，很明显[1]这部分玩家有足够的经济实力（成就解锁服务往往需要付费购买），应尽可能留存其，更新成就解锁方式，降低解锁偶然性，联机类游戏将成就设置为对局次数一类更可控的数据往往对留存度提升有帮助。

游戏内等级的提升也会带给玩家一定幸福感，从而间接提升留存率，等级往往与下面两点挂钩，一类是成长数值体验[3]：角色的成长、数值的反馈等，另一类是成长目标感，第二类适用性更高，因为不需要任何外界因素，部分玩家本身就注重等级为目标感，但第一类激励效果更加明显，因为角色的成长往往代表着解锁新能力，这也可以解释为何往往游戏初期玩家拥有更高留存度，玩家的探索欲望会加强留存度，而到了后期无法解锁新的能力之后，数值提升就可以代替其作用继续激励玩家了，因为这代表着玩家可以缓解之前遭到的不良体验，甚至实现“报仇”，这对留存度的提升有不小的作用，而且这种幸福感也会促进玩家转向内购，延长游戏寿命。

同时，在之前的研究中也发现，在不同留存度玩家的群体中性别以及地区分布都有一定规律，往往男性会尝试深度体验类游戏，而女性会更喜欢轻松休闲类以及中等深度游戏，这也意味着游戏厂商需要根据游戏类型与内容修改至其更符合用户画像，而不要力求全面，因为这往往会遭到混杂玩家群体的排斥，与其增加低活跃玩家数量，不如提升收益转化率，提升符合画像的玩家留存度。同样，亚洲玩家会更青睐与休闲类游戏，或许与地缘环境有关，企业可尝试细分地区，创建更详细的用户画像以提高留存度。

八、 参考文献

[1] 王冠群.基于满意度分析的 B 站客户留存度提升策略研究[D].湖北大学,2024.DOI:10.27130/d.cnki.ghubu.2024.002114.
[2] https://www.docin.com/p-4786452912.html
[3] https://bbs.gameres.com/thread_908311_1_1.html
[4] https://www.gameres.com/855499.html

附录

附录 1

介绍：软件版本 R4.2.2，用于问题一的建立与求解

```
##############################################
# 项目名称: 问题一——基于留存度的客户画像
# 软件版本: R4.4.2
# 攥写日期: 2025.1.12
##############################################

# 安装并导入相关包
#install.packages("rio");install.packages("tidyverse")
#install.packages("themis");install.packages("autoReg")
#install.packages("flexclust");install.packages("factoextra")
#install.packages("compareGroups");install.packages("tidymodels")
library(rio);library(tidyverse);library(tidymodels);library(themis)
library(autoReg);library(dplyr);library(NbClust);library(factoextra)
library(cluster);library(compareGroups)

#-----------------------------
# Part 1
#
# 数据录入与预处理
#-----------------------------

# 数据导入
datatrain <- import("DS_train.csv")
# 剔除缺失值
datatrain <- na.omit(datatrain)
datatrain <- datatrain %>% filter(DD != "") %>% select(-ID)
# 数据预处理
datatrain <- datatrain %>% mutate(Sex=factor(Sex),Loc=factor(Loc),
                                  GameType=factor(GameType),
                                  InGame=factor(InGame),DD=factor(DD),
                                  Target=factor(Target))

#-------------------------------------------
# Part 2
#
# 问题一:基于留存度的客户画像
#-------------------------------------------

#基线统计描述
data_all <-    gaze(Target ~ .,data=datatrain) %>% myft()
```

```
data_all

#基于留存度分类
datatrain_high <- datatrain%>%filter(Target=="High")
datatrain_medium <- datatrain%>%filter(Target=="Medium")
datatrain_low <- datatrain%>%filter(Target=="Low")


classify <- function(data) {

  set.seed(123)
  fit.pam <- pam(data[,-1], k=3 #  聚为 3 类
                 , stand = T #  聚类前进行标准化
  )
  group <- as.factor(fit.pam$clustering)
  output <- cbind(data,group)
  table <-    gaze(group ~ .-ID,data=output) %>% myft()
  return(table)
}

#绘制基线统计表
high_group <- classify(datatrain_high)
high_group
medium_group <- classify(datatrain_medium)
medium_group
low_group <- classify(datatrain_low)
low_group
```

附录 2

介绍：软件版本 R4.2.2，用于问题二的建立与求解

```
#######################################
# 项目名称: 问题二——分类模型的选择
# 软件版本: R4.4.2
# 攥写日期: 2025.1.12
#######################################

# 安装并导入相关包
#install.packages("rio");install.packages("tidyverse")
#install.packages("themis");install.packages("tidymodels")
#install.packages("parallel");install.packages("ranger")
#install.packages("kknn");install.packages("themis")
#install.packages("xgboost")
#install.packages("MASS");install.packages("klaR")
library(rio);library(tidyverse);library(tidymodels);library(themis)
```

```
library(parallel)
library(MASS)#LDA 判别分析
library(klaR)#贝叶斯判别分析

#----------------------------
# Part 1
#
# 数据录入与预处理
#----------------------------

# 数据导入
datatrain <- import("DS_train.csv")
# 剔除缺失值
datatrain <- na.omit(datatrain)
datatrain <- datatrain %>% filter(DD != "") %>% dplyr::select(-ID) %>%
  mutate(Target=factor(Target))


#-------------------------------------------------
# Part 2
#
# KNN,随机森林,XGBoost 模型的建立
#-------------------------------------------------

KNN <- data.frame(array(NA,dim=c(100,3)))
RanderForest <- data.frame(array(NA,dim=c(100,3)))
XGBoost <- data.frame(array(NA,dim=c(100,3)))
LDA <- data.frame(array(NA,dim=c(100,3)))
Bayes <- data.frame(array(NA,dim=c(100,3)))
Metrics <- metric_set(accuracy,sensitivity,specificity)
# 100 次交叉验证
# 划分训练集和测试集(7:3)
for(i in c(1:100)){
  set.seed(20250112+i*10)
  split <- initial_split(datatrain,prop = 0.7,strata = Target)
  Train <- training(split)
  Test <- testing(split)

  # 处理训练集中的不平衡问题且分类变量哑变量
  Recipe <- recipe(Target ~.,data = Train) %>%
    step_dummy(c(Sex,Loc,GameType,DD)) %>%
    step_smote(Target)
  Training <- Recipe %>% prep() %>%juice()
```

```
#LDA 模型的建立
#拟合模型
LDA_fit <- lda(Target ~ SPK+ASD+Level+ACHV, data = Training)
# 利用验证集测试模型
LDA_pred <- predict(LDA_fit,Test)$class
L_conf_matrix <- table(Test$Target, LDA_pred)
# 输出准确性,灵敏性以及特异性三类指标
LDA[i,1]<- sum(diag(L_conf_matrix)) / sum(L_conf_matrix)
LDA[i,2] <-mean(diag(L_conf_matrix) / colSums(L_conf_matrix))
LDA[i,3] <- mean(diag(L_conf_matrix) / rowSums(L_conf_matrix))

#Bayses 判别模型的建立
#拟合模型
Bayes_fit <- NaiveBayes(Target ~ SPK+ASD+Level+ACHV, data = Training)
# 利用验证集测试模型
Bayes_pred <- predict(Bayes_fit,Test)$class
B_conf_matrix <- table(Test$Target, Bayes_pred)
# 输出准确性,灵敏性以及特异性三类指标
Bayes[i,1]<- sum(diag(B_conf_matrix)) / sum(B_conf_matrix)
Bayes[i,2] <-mean(diag(B_conf_matrix) / colSums(B_conf_matrix))
Bayes[i,3] <- mean(diag(B_conf_matrix) / rowSums(B_conf_matrix))

# KNN 模型的建立
# 训练模型
ModelDesignKNN <- nearest_neighbor(
  neighbors = 125,
  weight_func = 'rectangular') %>%
  set_engine('kknn') %>%
  set_mode('classification')
WfModelKNN <- workflow() %>%
  add_recipe(Recipe) %>%
  add_model(ModelDesignKNN) %>%
  fit(Train)

# 利用验证集测试模型
DataTestWithPredKNN <- augment(WfModelKNN,Test)

# 输出准确性,灵敏性以及特异性三类指标
KNN[i,1] <- Metrics(
  DataTestWithPredKNN,truth = Target,estimate = .pred_class)[1,3]
KNN[i,2] <- Metrics(
  DataTestWithPredKNN,truth = Target,estimate = .pred_class)[2,3]
KNN[i,3] <- Metrics(
  DataTestWithPredKNN,truth = Target,estimate = .pred_class)[3,3]
```

```
# 随机森林模型的建立
# 训练模型
ModelDesignRandFor=rand_forest(min_n=5, mtry=2, trees=2000) %>%
  set_engine("ranger",importance="impurity",
                  num.threads=detectCores()) %>%
  set_mode("classification")
WfModelRF <- workflow() %>%
  add_recipe(Recipe) %>%
  add_model(ModelDesignRandFor) %>%
  fit(Train)

# 利用验证集测试模型
DataTestWithPredRF <- augment(WfModelRF,Test)

# 输出准确性,灵敏性以及特异性三类指标
RanderForest[i,1] <- Metrics(
  DataTestWithPredRF,truth = Target,estimate = .pred_class)[1,3]
RanderForest[i,2] <- Metrics(
  DataTestWithPredRF,truth = Target,estimate = .pred_class)[2,3]
RanderForest[i,3] <- Metrics(
  DataTestWithPredRF,truth = Target,estimate = .pred_class)[3,3]


# XGboost 模型的建立
# 训练模型
ModelDesignBoostTrees <- boost_tree(trees = 100,tree_depth = 5) %>%
  set_engine("xgboost") %>%
  set_mode("classification")
WfModelBT <- workflow() %>%
  add_recipe(Recipe) %>%
  add_model(ModelDesignBoostTrees) %>%
  fit(Train)

# 利用验证集测试模型
DataTestWithPredBT <- augment(WfModelBT,Test)

# 输出准确性,灵敏性以及特异性三类指标
XGBoost[i,1] <- Metrics(
  DataTestWithPredBT,truth = Target,estimate = .pred_class)[1,3]
XGBoost[i,2] <- Metrics(
  DataTestWithPredBT,truth = Target,estimate = .pred_class)[2,3]
XGBoost[i,3] <- Metrics(
```

```
      DataTestWithPredBT,truth = Target,estimate = .pred_class)[3,3]
}


#-----------------------------------------------------
# Part 3
#
# 绘制各模型 100 次交叉验证评价指标图
#-----------------------------------------------------

# 准确性
par(mar=c(5,5,2,2))
plot(c(1:100),KNN[,1],lwd=4,type="l",bty="l",xlim=c(0,100),lty=1,xaxt="n",
       yaxt="n",xlab="",ylab="",ylim=c(0.7,0.92),col=1)
axis(1,las=1,cex.axis=1.3,lwd=1.6)
axis(2,las=1,cex.axis=1.3,lwd=1.6)
title(xlab="Times of Simulation",font.lab=7,cex.lab=1.5,line=2.6)
title(ylab="Accuracy",font.lab=7,cex.lab=1.3,line=3.5)
lines(c(1:100),RanderForest[,1],lwd=4,lty=1,col=2)
lines(c(1:100),XGBoost[,1],lwd=4,lty=1,col=3)
lines(c(1:100),LDA[,1],lwd=4,lty=1,col=4)
lines(c(1:100),Bayes[,1],lwd=4,lty=1,col=5)
legend(65,0.88,c("KNN","Rander Forest","XGBoost","LDA","Bayes"),lty=1,lwd=4,
           bty="n",xpd=TRUE,cex=1.1,col=c(1,2,3,4,5))

# 灵敏性
par(mar=c(5,5,2,2))
plot(c(1:100),KNN[,2],lwd=4,type="l",bty="l",xlim=c(0,100),lty=1,xaxt="n",
       yaxt="n",xlab="",ylab="",ylim=c(0.75,0.92),col=1)
axis(1,las=1,cex.axis=1.3,lwd=1.6)
axis(2,las=1,cex.axis=1.3,lwd=1.6)
title(xlab="Times of Simulation",font.lab=7,cex.lab=1.5,line=2.6)
title(ylab="Sensitivity",font.lab=7,cex.lab=1.3,line=3.5)
lines(c(1:100),RanderForest[,2],lwd=4,lty=1,col=2)
lines(c(1:100),XGBoost[,2],lwd=4,lty=1,col=3)
lines(c(1:100),LDA[,2],lwd=4,lty=1,col=4)
lines(c(1:100),Bayes[,2],lwd=4,lty=1,col=5)
legend(65,0.85,c("KNN","Rander Forest","XGBoost","LDA","Bayes"),lty=1,lwd=4,
           bty="n",xpd=TRUE,cex=1.1,col=c(1,2,3,4,5))

# 特异性
par(mar=c(5,5,2,2))
plot(c(1:100),KNN[,3],lwd=4,type="l",bty="l",xlim=c(0,100),lty=1,xaxt="n",
       yaxt="n",xlab="",ylab="",ylim=c(0.78,0.97),col=1)
```

```
axis(1,las=1,cex.axis=1.3,lwd=1.6)
axis(2,las=1,cex.axis=1.3,lwd=1.6)
title(xlab="Times of Simulation",font.lab=7,cex.lab=1.5,line=2.6)
title(ylab="Specificity",font.lab=7,cex.lab=1.3,line=3.5)
lines(c(1:100),RanderForest[,3],lwd=4,lty=1,col=2)
lines(c(1:100),XGBoost[,3],lwd=4,lty=1,col=3)
lines(c(1:100),LDA[,3],lwd=4,lty=1,col=4)
lines(c(1:100),Bayes[,3],lwd=4,lty=1,col=5)
legend(65,0.925,c("KNN","Rander Forest","XGBoost","LDA","Bayes"),lty=1,lwd=4,
       bty="n",xpd=TRUE,cex=1.1,col=c(1,2,3,4,5))
```

附录 3

介绍：软件版本 R4.2.2，用于问题三的建立与求解

```
########################################
# 项目名称: 问题三——分类模型的预测
# 软件版本: R4.4.2
# 攥写日期: 2025.1.12
########################################

# 安装并导入相关包
#install.packages("rio");install.packages("tidyverse")
#install.packages("themis");install.packages("tidymodels")
#install.packages("parallel");install.packages("ranger")
#install.packages("kknn");install.packages("themis")
#install.packages("xgboost")
library(rio);library(tidyverse);library(tidymodels);library(themis)

#-----------------------------
# Part 1
#
# 数据录入与预处理
#-----------------------------

# 数据导入
datatrain <- import("DS_train.csv")
# 剔除缺失值
datatrain <- na.omit(datatrain)
datatrain <- datatrain %>% filter(DD != "") %>% select(-ID) %>%
  mutate(Target=factor(Target))

# 划分训练集和测试集(7:3)
set.seed(20250112)
split <- initial_split(datatrain,prop = 0.7,strata = Target)
Train <- training(split)
```

```
Test <- testing(split)

# 检测训练集中预测变量是否存在数据不平衡问题
count(Train,Target)

# 处理训练集中的不平衡问题且分类变量哑变量
Recipe <- recipe(Target ~.,data = Train) %>%
  step_dummy(c(Sex,Loc,GameType,DD)) %>%
  step_smote(Target)

# 输出处理后的训练集
Training <- Recipe %>% prep() %>% juice()
# 再次检测训练集中预测变量是否存在数据不平衡问题
count(Training,Target)


#------------------------------
# Part 2
#
# XGboost 模型的建立
#------------------------------

# 训练模型
ModelDesignBoostTrees <- boost_tree(trees = 100,tree_depth = 5) %>%
  set_engine("xgboost") %>%
  set_mode("classification")
WfModelBT <- workflow() %>%
  add_recipe(Recipe) %>%
  add_model(ModelDesignBoostTrees) %>%
  fit(Train)

# 利用验证集测试模型
DataTestWithPredBT <- augment(WfModelBT,Test)

# 输出准确性,灵敏性以及特异性三类指标
Metrics <- metric_set(accuracy,sensitivity,specificity)
Metrics(DataTestWithPredBT,truth = Target,estimate = .pred_class)


#------------------
# Part 3
#
# 模型的预测
#------------------
```

```
# 导入待预测的数据集
datapred <- import("DS_test.csv")
# 利用模型进行预测
DataPred <- augment(WfModelBT,datapred)

# 导出预测结果
Pred <- DataPred %>% select(c(ID,.pred_class)) %>%
   mutate(target = .pred_class) %>% select(-.pred_class)
write.csv(Pred,file = "问题三预测结果.csv",row.names = FALSE)

# 可视化预测变量变化影响
library(vip)
vip(extract_fit_engine(WfModelBT),
       aesthetics = list(fill = "red",color = "red", size = 0.8)) +
   theme_minimal()
```