上海交通大學

SHANGHAI JIAO TONG UNIVERSITY

数学科学学院

数据挖掘实验报告

班　　级：应用统计

姓　　名：陈　妍　119071910037

刘东航　119071910048

张雯婷　119071910075

赵文喆　119071910076

指导老师：高振国

2019~2020 第一学期

使用 LaTeX 撰写于 2019 年 12 月 13 日

摘　要

本次报告以 2019 年 11 月 9 日的 Google Play Store 官网各类别主页的 App 为对象，研究其属性以及各属性之间是否存在显著关系。基于原始的包含 3661 条记录和 25 个属性的数据，经过恰当的数据清洗，本文进行了充分的探索性分析。进一步地，我们希望知道 App 最核心的评分 (rating) 属性能否通过其他重要属性进行解释和预测。本报告分别采用 Logistic 回归、随机森林、Xgboost 和神经网络来建模分析，进而探究帮助 App 获得高评分的属性，并且比较上述四种模型交叉验证后的平均预测准确率，从而找到最适合的预测模型。

研究发现：首先，包括 reviews(评论数)，updateDays(更新日期) 等属性在预测评分 (rating) 时较为重要。其次，四种模型中，Xgboost 模型的预测准确率更高。

关键字：　数据清洗　Logistic 回归　随机森林　XGBoost　神经网络

Abstract

This article takes the application of each category homepage on the official website of the Google Play Store on November 9, 2019 as an object, and studies its attributes and whether there is a significant relationship between these attributes. Based on the original data containing 3,661 records and 25 attributes, and after proper data cleaning, this paper has conducted a comprehensive exploratory analysis. In addition, we want to know if the core rating attributes of the application can be explained and predicted by other important attributes. This article uses Logistic Regression, Random Forest, Xgboost, and Neural Networks for modeling and analysis, and then explores attributes that help the App get high scores, and compares the average prediction accuracy rate after cross-validation of the above four models to find the most suitable forecasting model.

The study found: First, attributes such as review and updateDays are more important for predicting ratings. Secondly, of the four models, the classification accuracy of the Xgboost model is the highest.

Keywords：Data clean Logistic regression Random forest XGBoost Neural networks

目录

一 项目介绍

1.1 研究背景

近年来，以移动设备技术逐渐成熟为契机，承借移动通信技术的发展的东风，以及人们对移动业务需求变得更加多样化、精细化和个性化，各种各样的应用软件呈现井喷式发展的状态。但是，应用软件市场准入门槛较低，所以导致目前市面上的应用软件风格各异、功能多样且质量不一，没有统一的行业评价标准供各方参考。从开发者角度来看，值得关心的问题是其开发的应用软件在市场上的竞争力，以及怎样有针对性地进行改进和吸引使用者的目光。目前已经有一些相关的研究可以供开发者参考，但是总体上来说数量还比较少，仍然有值得探索挖掘的地方。对数据挖掘的目标数据集 Google Play Store 数据集进行爬取。

1.2 研究问题

移动网络和智能手机的普及推动了应用商店的快速发展，而 Google Play Store 是目前全球最大的应用程序发布平台之一，所以我们选择其平台下应用程序作为研究对象。针对应用程序，评分是用户对该产品的体验反馈，在一定程度上能够反映其在市场上的竞争力，并且对于开发者而言有重要意义。因此，我们针对应用程序的特征进行具体的分析和建模，以探究对评分有显著性影响的因素。为了方便评估，我们将评分以其中位数为界进行均衡的二分类，并应用相关算法建模预测。

1.3 数据介绍

1.3.1 数据获取

为了获取 Google Play（见图 1.1）应用程序数据信息，我们首先通过 Python 的 requests 和 BeautifulSoup 库爬取了官网各类别主页 App 的 URL 链接。之后，通过修改和调用 play-scraper 库 (https://github.com/danieliu/play-scraper) 相关函数，根据已获得的链

接爬取 App 的详细数据信息。

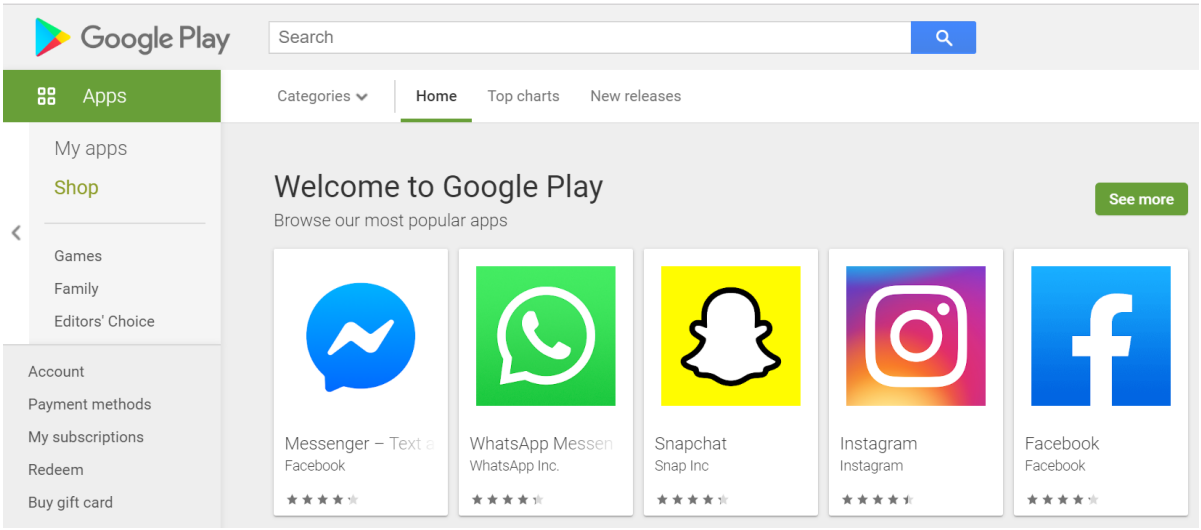


图 1.1: Google Play 网站截图

1.3.2 数据描述

初始数据集包含 3661 条记录，25 个属性，其中包含数值型变量，类型变量，字符型变量和布尔型变量。25 个属性分别为：title(名称)，icon(图标网址)，category(类别)，score(评分均分)，histogram(评分分布)，reviews(评论数)，description(描述)，price(价格)，free(是否免费)，iap(是否有应用内收费项目)，developer id(开发者 ID), updated(最新更新日期)，size(大小)，installs(下载量)，current version(版本号)，required android version(安卓版本要求)，content rating(内容级别)，iap range(应用内收费项目价格区间)，interactive elements(交互元素)，developer(开发商)，developer email(开发商邮箱)，developer url(开发商网址)，developer address(开发商地址)，app id(ID), url(链接)。基于具体研究问题，我们将剔除掉数据集中删除了开发者 ID、APP 版本号、开发商邮箱等与小组研究目的不太相关、且研究价值不高的 9 个属性，从 25 个属性初步筛选出 16 个属性作为重点研究对象，具体属性介绍见表 1.1。

1.4 研究方法

本文选取 Google Play 3661 条 APP 记录作为研究对象，目标是完成评分高低二分类的预测任务。针对该任务，我们主要采用了逻辑回归，随机森林，XGBoost 和神经网

表 1.1: 清洗前 Google Play 数据集重要属性介绍

名称	描述	例
title	应用程序名称	TikTok - Make Your Day
icon	图标网址	https:// lh3.googleuserconte -nt.com/ z5nin1RdQ4UZh v6fa1FNG7VE33imGqPgC4 kKZIUjgf_up7E- Pj3AaojlMPwNNXaeGA
category	类别	['SOCIAL']
score	评分	4.4
histogram	评分分布	{5: 9891605, 4: 989160, 3: 494580, 2: 197832, 1: 1285908}
description	描述	TikTok is THE destination for mobile videos. On TikTok, short-formed videos are...
installs	下载量	500,000,000+
reviews	评论数	12859087
size	大小	79M
content rating	内容级别	['Teen']
free	是否免费	TRUE
price	价格	0
iap	是否有应用内收费项目	TRUE
iap range	应用内收费项目价格区间	('$0.99','$99.99')
updated	最新更新日期	7-Nov-19
required android version	安卓版本要求	4.1 and up

络算法建立对应分类模型，并进行预测评估及模型对比。

1.5 组织结构

根据研究内容的需要，本文共分为 5 个章节详细展开：

第 1 章为引言部分，主要介绍了项目的研究背景，数据描述，研究问题及方法，并且简述论文的整体框架。

第 2 章为数据清洗，包括属性处理和缺失值处理两个部分，清洗后的数据将用于之后的数据挖掘。

第 3 章是数据的探索性分析，采取将数据可视化的方式，以期发现数据规律、为后文研究提供思路。

第 4 章分别建立了逻辑回归、随机森林、XGBoost、神经网络分类模型，并对以上模型分类效果进行评估与对比。

第 5 章为全文总结、延伸以及对未来工作的展望。

二 数据清洗

2.1 数据集属性选择

Google Play 数据集包含 25 个原始属性，我们的研究重点关注其中 16 个属性。在属性的筛选中，我们删除了 APP 图标网址、开发者 ID、APP 版本号、交互元素、开发商邮箱、开发商网址、开发商地址、应用程序网站和 APP 网址等与我们小组研究目的不太相关、且研究价值不高的 9 个属性，保留了反映应用程序的性质及特点的属性，并将选择后的属性应用于后续的数据挖掘与分析。选择的 16 个属性具体见。

2.2 数据集属性处理

在 2.1 的基础上，对于筛选出的 16 个属性分别进行精细化的预处理。预处理的目的是为了让杂乱无序的数据集变得尽量规整，利于后续的数据分析与研究。

title,description 属性

title,description 属性下是文本数据，使用了包括中文、英文、拉丁语等多种语言类型。我们的处理分为检测和翻译两个步骤，在 Python 中调用模块 googletrans: Free Google Translate API for Python（https://pypi.org/project/googletrans/）。首先使用 detect 函数，检测出 title 和 description 下所有内容对应的语言类型。然后采用 translate 函数，将两个属性下非英文的内容翻译为英文，最终将检测出的 222 条非英文内容翻译为了英文内容。以 title 属性为例展示了我们所做的翻译工作，见图 2.1。

index	title	detect	translate
39	8891汽車 - 買車,	Detected(lang=zh-CN, confidence=1.0)	8891 car - a car, first on
40	Билеты ПДД 2019	Detected(lang=ru, confidence=1.0)	Tickets SDA 2019 Exami
46	중고차 등록 대수 :	Detected(lang=ko, confidence=1.0)	Used car registration nu

图 2.1: 基于 title 属性的翻译工作截图

图 2.1中将汉语、俄语与韩语均成功译为英语。该处理的目的是将所有内容处理为统一语言，便于在后面的研究中直接使用。

category 属性

在原始数据中，每一个对象对应着多个类别，例如：[‘ART AND DESIGN’, ‘FAMILY CREATE‘]。category 列表第一个元素是主要的所属类别，第二个元素是辅助的类别信息。在第一个位置上，类别有可能被细分为更小的分类，例如‘GAME XX ‘，表示该对象属于 GAME 大类下的某一小类。我们首先提取列表的第一个元素，对应内容为主要的所属类别，然后去掉小分类、统一归为大类。例如将 [‘GAME XX’, ‘FAMILY CREATE’] 处理为 [‘GAME’]。处理之后 category 属性下的类别减少至 33 个，并将处理后的数据保存在新属性 main category 下。

score(rating) 属性

原始数据为 float 型，用于研究分析已经足够，所以不做处理，仅将 score 属性更名为 rating 属性。但需要注意存在缺失值，在下一节中进行处理。

histogram 属性

原始数据为字符串类型，每一个对象包含了 5 分、4 分、3 分、2 分和 1 分的评分分别对应的条数。我们认为这一属性中包含了五个信息，所以将字符串拆分为五个新的属性，将提取分割后的五个信息保存在新属性’rating#5’, ’rating#4’, ’rating#3’, ’rating#2’, ’rating#1’ 下，分别表示各级评分所占比例，数据格式为 float 型。

installs 属性

原始数据为字符串类型，部分数据末尾含有符号“+”，代表下载量大于某个值。我们所做的处理是将末尾的“+”符号删除，并且将数据改为 int 型，便于在后面的研究中进行计算。

reviews 属性

原始数据为 int 型，可在研究中直接进行计算处理，已经满足分析需求，故不做处理。

size 属性

原始数据单位不统一，表示为“M”或者“k”。首先统一数据单位。采用换算公式将“k”转换为“M”，统一数据单位为“M”。然后将全部数据单位一同去掉，数值类型为 float 型。若其值为'Varies with device'，则记为空值，这部分数据将在下一节中作为缺失值进行处理。

content ratings 属性

在原始数据中，每一个对象包含了多个内容分级，例如 ['Teen', 'Fantasy Violence, Mild Blood, Mild Language, Suggestive Themes']。我们认为，在本分析中只需用到与年龄相关的分级内容，即第一个位置上的数据信息，其他分级内容不在本研究涉及范围内，故进行舍弃。数据处理后，保留了“Everyone”、“Teen”、“Mature 17+”、“Everyone 10+”、“Adults only 18+”五个年龄分级。

free 属性

原始数据为 bool 型，取值为 False 和 True，不作进一步处理。

price 属性

原始数据为字符型，在字符串首有“$”符号。将“$”删去，并将数据更改为 float 型，以便后续处理。

iap 属性

iap 属性含义为是否有应用内收费项目，原始数据为 bool 型，取值 False 和 True，可以满足分析需求，故不处理。需要说明的是，iap 取值 False 指的是应用内无收费项目，共有 2159 个 APP，iap 取值 True 指的是应用内存在收费项目，共有 1502 个 APP。

iap range 属性

原始数据含义为应用内收费项目价格区间，例如 ('$0.99', '$99.99')。对于有收费项目的 2159 个 APP, 提取出价格区间中的上限与下限，将原始属性 iap range 拆分为两个新属性 iap max 和 iap min，分别表示应用内商品最低价格和最高价格，并将数据类型更

改为 float 型。对于无收费项目的 1502 个 APP，我们的处理是把 iap max 和 iap min 均记为零值。

updated 属性

原始数据为字符串型，表示应用软件最新更新日期。首先将数据类型由字符串改为日期型，然后进行处理：updated-max(updated)，即该用应用软件的更新日期减去所有软件中的最大更新日期。添加新属性 updateDays，并将处理后的数据保存在新列中，用于计量最近更新时间，数据类型为 int 型。

required android version 属性

原始数据类型为字符串，并且表现形式不一，例如“2.1”和“4.3.1.1”。首先去除多余字符，统一取前 3 个字符，例如 4.1，将数据长度统一。其次，将数据类型改为 float 型，方便后续分析。将处理后的数据保存在新属性 android version 下。若其值为’Varies with device’，记为缺失值，在下一节中进行缺失值处理。

2.3 数据集缺失值处理

对 Google Play 数据集进行识别之后，发现 rating score 属性存在 32 个缺失值，required android version 属性存在 6 个缺失值。通过与原始网页数据信息对比，确定缺失值是由于输入时的遗漏所造成的。从总体上来看，缺失值数量较少，并未丢掉过多有用信息，并且数据的缺失是完全随机的，因此不会对样本的无偏性产生影响。

值得注意的是，在 required android version 属性和 size 属性中，存在着形如“Varies with device”的数据 (required android version:733 个，size:897 个)，意为 APP 面向的安卓版本以及大小会根据不同的机型有所不同。这部分数据虽然没有表现为缺失值，但从实际意义上来看，并未提供任何有效信息，因此在本研究中将其认定为缺失值一并处理。综上，缺失值在各属性下的分布情况由表 2.1给出。

由于缺失值的存在会给数据挖掘带来不便，进而对研究产生影响，所以需要进行处理，以减少数据挖掘算法与实际应用之间的差距。在本研究中，缺失值的处理方法主要有直接删除、众数填充和平均值填充。

(1) rating 属性

表 2.1: 缺失值处理表

NaN Attributes	Total	Percent
rating	32	0.87%
required android version	739(6+733)	20.19%
size	897	24.5%

在 rating 属性下，共计有 32 个缺失值，不包含“Varies with device”的情况。与总体数据量相比，缺失值所占比例较低，直接删除对研究分析影响不大，所以选择删除整条记录的方式进行处理。

(2) required android version 属性

在 required android version 属性下，共计有 739 个缺失值，其中包括 6 个原有的缺失值和 733 个“Varies with device”。原有的 6 个缺失值在上一步骤中删去 rating 缺失行后消失，不需要另作处理。考虑到“Varies with device”数据量比较大，为了避免严重的数据偏离，不能采用直接删除的方法，需要进行填充处理。从实际角度考虑，在应用软件市场上，同样类型的应用软件对安卓版本的要求大多数是相同的。因此，在本研究中利用 main category(33 类) 将应用软件按照所属类别分组，并用各组的众数替代该组下“Varies with device”的缺失值。

(3) size 属性

size 属性的缺失值均为“Varies with device”，缺失值总数占比高达 24.50%，与 android version 情况类似，不能予以直接删除。所以选用与 required android version 类似的方法进行填充，不同的地方是改为以已有的样本数据的中心来代表缺失的数据，具体做法是根据该属性在其他所有对象中的取值的平均值来填充缺失值。

2.4 数据集清洗结果

经过属性处理及缺失值处理，清洗后的 Google Play 数据集重要属性见表 2.2。清洗后的数据集将用于后续的探索性研究及数据分析。

表 2.2: 清洗前 Google Play 数据集重要属性介绍

名称	描述	例
title	应用程序名称	TikTok - Make Your Day
icon	图标网址	https:// lh3.googleusercontent-nt.com/ z5nin1RdQ4UZhv6fa1FNG7VE33imGqPgC4kKZIUjgf_up7E-Pj3AaojlMPwNNXaeGA
main category	类别	['SOCIAL']
rating	评分	4.4
description	描述	TikTok is THE destination for mobile videos. On TikTok, short-formed videos are...
installs	下载量	500,000,000
reviews	评论数	12859087
size	大小	79M
content rating	内容级别	['Teen']
free	是否免费	TRUE
price	价格	0
iap	是否有应用内收费项目	TRUE
iap max	应用内商品最高价格	$99.99
iap min	应用内商品最低价格	$0.99
updateDays	更新天数	-1
android version	安卓版本要求	4.1

三　数据探索性分析

在第二部分对 Google Play 数据集清洗后，我们为了对数据集有更直观清晰的了解，展开了针对数据集的探索性分析工作，以期在探索性分析的过程中发现数据集存在的某些规律，并为后续研究方案的制定提供一些思路。

探索性分析主要分为对 Google Play 数据集中单变量的分析，双变量的分析以及多变量的相关性分析。

3.1　单变量的探索性分析

单变量的分析我们选取了比较有研究价值的 rating 属性与 category 属性，此处的 category 属性指的是数据预处理之后的 33 大类的 main category 属性。

3.1.1　对 rating 属性的分析

为了对 rating 进行分析，首先画出其分布图，以清晰地展示出 rating 的分布情况。然后再计算其基本的数字特征，并画出相应的箱线图，以便能够更为直观简洁地获取 rating 的特征，见图 3.1。

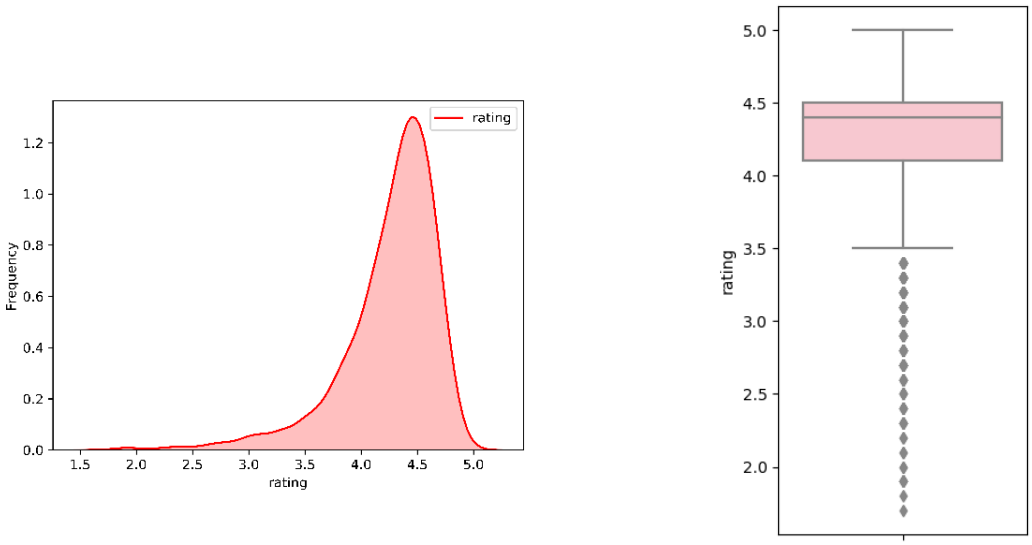


图 3.1: Google Play 数据集中 APP 评分分布图以及评分箱线图

根据图 3.1，rating 的分布明显呈左偏形态，其均值为 4.26，中位数为 4.4，众数为 4.5，可见在满分为 5 分的条件下，用户对应用软件评分普遍较高。同时，可以观察到下四分位数为 4.1，表明大部分用户的评分在 4 分以上，说明多数用户更倾向于给应用软件打出高分，对应用软件的宽容度较高。

3.1.2 对 category 属性的分析

对样本的 category 进行探索分析。以 category 为横轴、频数为纵轴，按频数从高到低排列，可以画出条形图 3.2。

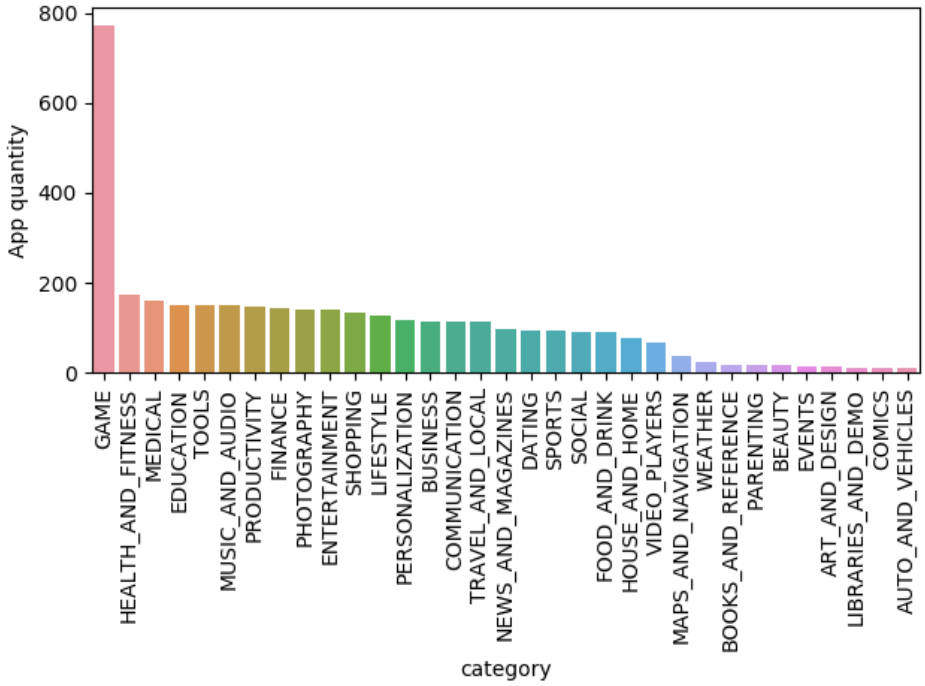


图 3.2: Google Play 数据集的 APP 种类分布条形图

从图 3.2中可以看出，“GAME”类型和其他类型的应用软件的数量出现了断层，说明市场上大部分软件是属于“GAME”类别的。可以推测“GAME”类型的应用软件可以给软件开发商带来更好的经济效益。通过探索性分析，我们也发现现有的 category 还存在着一些不足，可能也因此部分地影响了其他类型应用软件的数量远远小于“GAME”类型。部分 category 包含的应用软件范围重叠部分较多，可以进行合并处理。例如，“HEALTH AND FITNESS”和“MEDICAL”包含的都是与“医疗”、“健康”等关键词有关的应用软件，存在着很大范围的重合部分，可以合并为“HEALTH”类别。其他的一些分类也存在着同样的问题，在后续的分析中可以进行进一步的处理。

3.2 双变量的探索性分析

双变量的探索性分析主要将两变量间的关系可视化并挖掘出一些基于底层信息的二级信息。

3.2.1 rating 属性与 content rating 属性

content rating 属性前面已经介绍过是描述 APP 内容级别 (面向年龄段) 的变量。为了探索评分和用户年龄之间的关系，我们以横轴代表用户年龄、纵轴代表评分做出了下图 3.3。

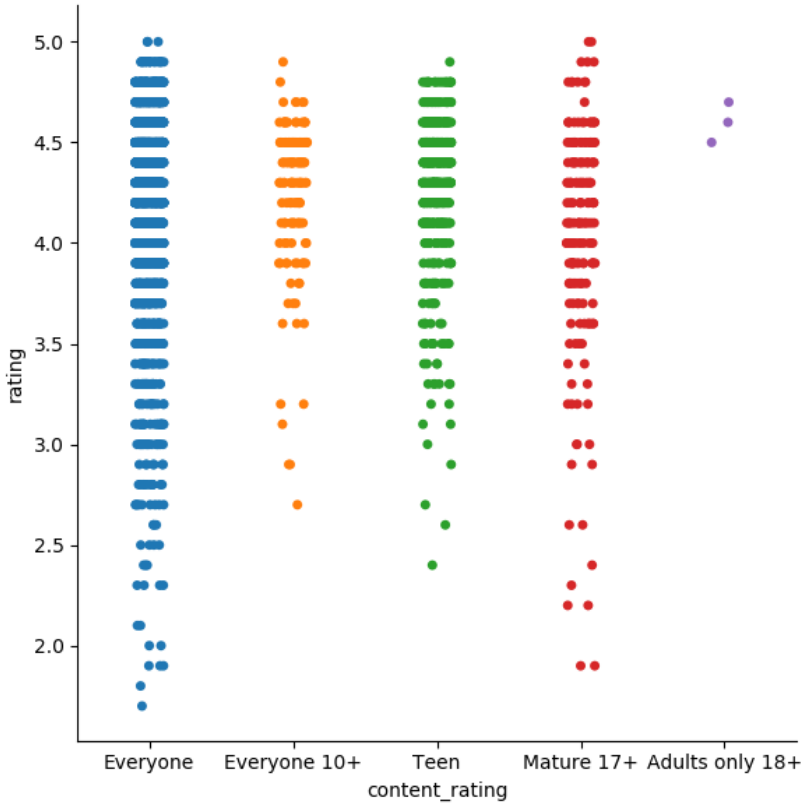


图 3.3: Google Play 数据集中 APP 评分与内容级别因素的关系图

对于图 3.3从纵向来看,每一列代表了不同年龄的用户的评分情况。首先对“Everyone 10+”、“Teen”和“Mature 17+”的用户进行分析。从图形上看，“Adults only 18+”的用户评分集中在高分部分，但是由于这部分用户数据较少，为避免分析的有偏性，在此不做讨论。“Everyone 10+”和“Teen”的用户评分分布情况相似，可以合并讨论。“Teen”和“Everyone 10+”的用户相对于“Mature 17+”的用户来说，评分更集中于 [3.5,4.5] 的

区间内，且极端低分更少，说明这个年龄段的用户评分更为宽容，应用软件使用感更佳。“Mature 17+”的用户评分更为分散，并且对极端低分的贡献率最大，展现出这一年龄组内应用软件使用感差异较大，更容易出现低分情况。软件开发商可以针对这一年龄段的用户需求进行有针对性的分析，以便更好地改进软件。

第一列对应的年龄为“Everyone”，包含了大部分样本数据，并且在其对象中含有其他年龄段的用户，所以可以将其他列与第一列进行对比分析。对比“Everyone”和“Everyone 10+”可以更为明显的看出，相对于其他年龄段的用户，“Everyone 10+”评分较高也更为集中，与前文的分析结论相符合。在此对比分析中，“Teen”和“Everyone 10+”的用户情况类似，结论也基本一致。对比“Everyone”和“Mature 17+”可以发现，两列数据的分布在所有对比分析中最为相似，特别是在低分部分表现得更为明显，这说明在“Everyone”中的极端低分贡献最大者也是“Mature 17+”的用户。

从横向来看，每一行代表了不同评分的数量。“Everyone 10+”、“Teen”和“Mature 17+”的评分大都集中在 [3.5,5.0] 的区间范围内，并且集中的区间范围逐渐扩大，即“Everyone 10+”的评分最为集中，而“Mature 17+”的评分最为分散。

3.2.2 rating 属性与 update year 属性

对 rating 和 update year 之间的关系进行探索性分析。其中，update year 表示软件最后更新的年份，例如：update year 为 2017，表示该应用软件最后一次更新是在 2017 年，在 2018 年和 2019 年均不再更新。以 update year 为横轴、rating 为纵轴画出箱线图 3.4。

从共性角度来看，四个箱线图的箱体部分大多集中于 [4.0,4.5] 的区间范围内，并且大部分都出现了极端低值的情况。这说明所有年份的评分数据大多数集中在 [4.0,4,5] 的区间范围内，总体评分都比较高，但是都会有极端低分的情况出现，这说明大部分应用软件都能给用户带来良好的使用体验，但是仍然会有少部分用户的需求未能得到满足，这一分析也符合现实情况。

对比四个年份的箱线图，可以看出平均评分逐年升高。可以合理地推测，随着应用软件市场的逐渐成熟，出现了更多性能和使用感更好的应用软件，用户也能够更便捷地找到满足自身需求的应用软件，可以得出结论，大趋势上应用软件的使用体验变得越来越好。

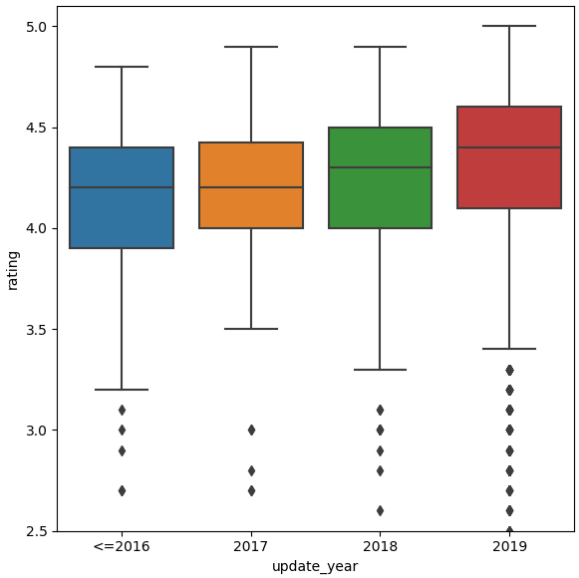


图 3.4: Google Play 数据集中 APP 最后更新年限的箱线图

3.2.3 rating 属性与 free 属性

为了探索应用软件是否付费与评分之间的关系，首先对市场上付费软件与免费软件的数量进行对比，做出环形图以直观地看出数量对比情况。然后分别对付费和免费软件画出 rating 的箱线图，便于进行对比分析，具体见图 3.5。

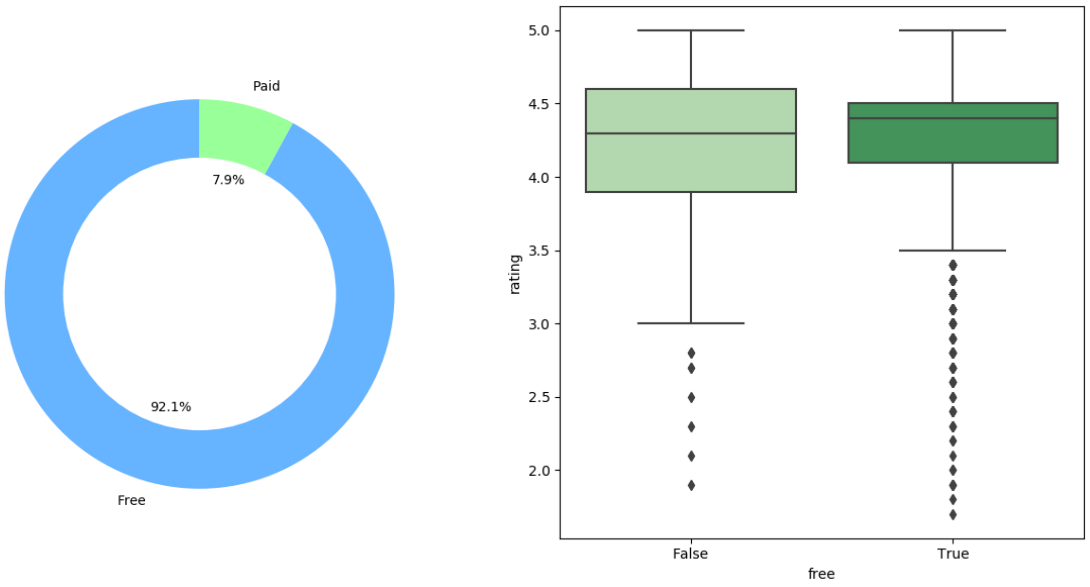


图 3.5: Google Play 数据集中 APP 是否付费环形图以及评分基于是否付费的箱线图

从环形图可以看出，在市场上现有的应用软件中，大多数为免费软件，对应的占比

超过了 90%，仅有不足 10% 的应用软件为付费软件。

与免费的应用软件相比，在付费软件的箱线图中，箱体部分更短，下边缘更高，并且中位数对应的评分也更高。这说明相较于免费软件，付费软件的评分更为集中，总体评分更高，而且更不容易出现极端低分的情况。可以分析得到，市场上的付费软件虽然数量较少，但是能够提供更好的服务以更好地满足用户需求，进而为用户带来更佳的软件使用体验。

3.3 多变量的相关性分析

在探索性分析中，属性之间的相关性分析是非常重要的，甚至能够影响后续分析的走向。在处理后 Google Play 数据集中有 12 个属性是以数值型数据及布尔型数据的形式呈现的。我们对这 12 个属性进行了相关性分析，并且画出了相关性图 3.6。

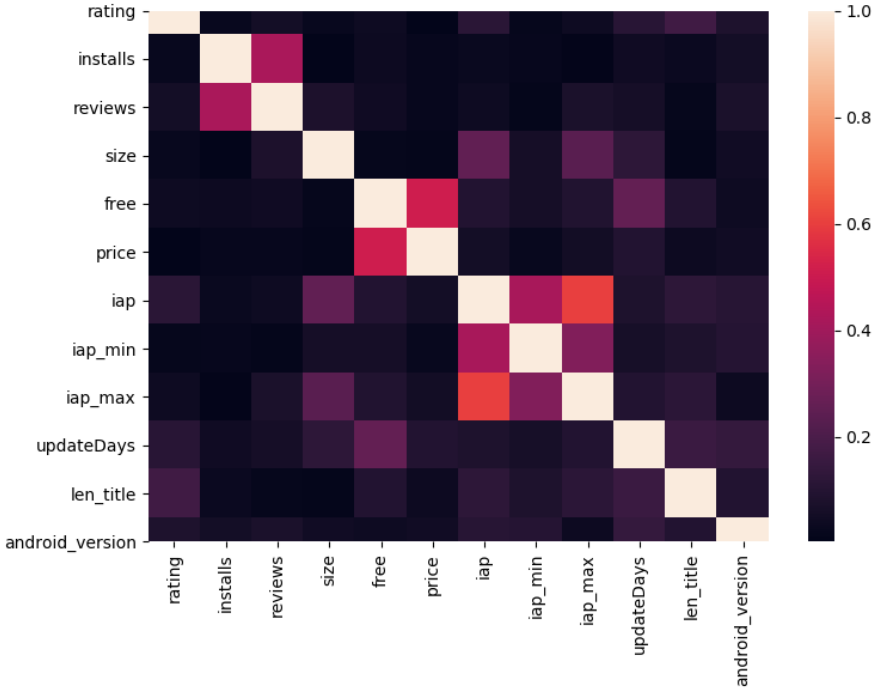


图 3.6: Google Play 数据集中 12 个属性的相关性图

在相关性图中，颜色越深代表相关性越弱，颜色越浅则代表相关性越强。从相关性图中可以看出，“iap max”和“iap”的相关系数处于 [0.6,0.8] 的区间范围内，具有强相关关系；“price”和“free”、“installs”和“reviews”的相关系数处于 [0.4,0.6 的区间范围内，且前者的相关性强于后者，具有中等程度的相关性；其他属性之间为弱相关或不相关关系。从现实角度考虑，“installs”和“reviews”的中等相关关系更具现实意义，

与实际情况相符合。可以合理地分析，当软件的下载数量较多时，使用该软件的用户基数更大，可能收到的评论数也会更多；当软件的评论数量较多时，说明可能有更多的使用用户，从而表现出下载量较多的现象。该相关性分析可以更好地为后文的分析提供研究方向和思路。

四 基于数据集的研究方案

将 rating 二分类后得到的新属性定义为 rating class，作为响应变量。自变量的选择则根据参考文献，选取了 installs，reviews，size，content rating，free，price，iap，android version，len title(title 属性的字符长度)，main category，iap min，iap max，updateDays 总共 13 个属性作为基础自变量。本节所展示的各类算法如无特殊说明，都以上述 13 个基础自变量对二分类响应变量 rating class 进行解释和预测，并在此基础上通过变量变换，删减变量等方法试图对模型进行优化。

4.1 Logistic 分类

4.1.1 Sigmoid，一种跃迁的概率估计函数

我们想要的函数应该是，能接受所有的输入然后预测出类别。例如，在两个类的情况下，上述函数输出 0 或 1。或许你之前接触过具有这种性质的函数，该函数称为 Sigmoid 函数。Sigmoid 函数具体的计算公式如下：

$$\sigma(z) = \frac{1}{1 + e^{-z}} \tag{4.1}$$

图 4.1给出了 Sigmoid 函数在不同坐标尺度下的两条曲线图。当 x 为 0 时，Sigmoid 函数值为 0.5。随着 x 的增大，对应的 Sigmoid 值将逼近于 1；而随着 x 的减小，Sigmoid 值将逼近于 0。如果横坐标刻度足够大（图 4.1右图），Sigmoid 函数看起来很像一个阶跃函数。因此，为了实现 Logistic 回归分类器，我们可以在每个特征上都乘以一个回归系数，然后把所有的结果值相加，将这个总和代入 Sigmoid 函数中，进而得到一个范围在 0 1 之间的数值。任何大于 0.5 的数据被分入 1 类，小于 0.5 即被归入 0 类。所以，Logistic 回归也可以被看成是一种概率估计。

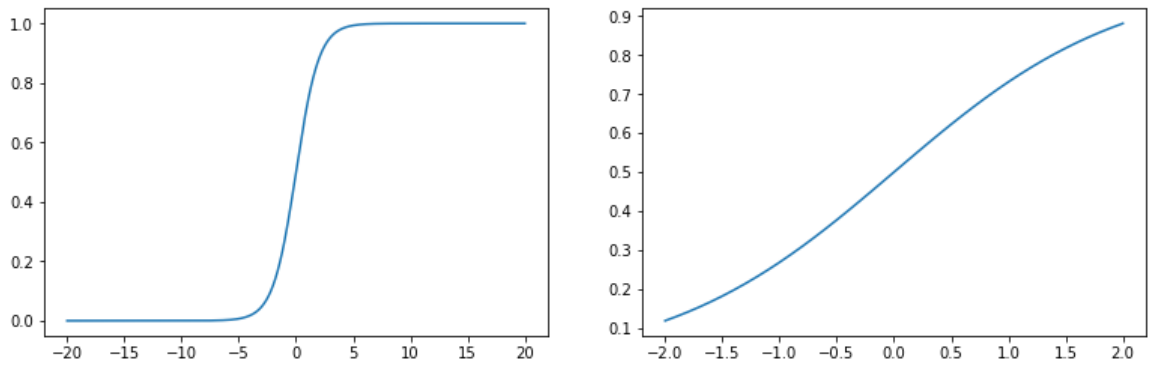


图 4.1: 两种坐标尺度下的 Sigmoid 函数图。左图的横坐标范围为-2 到 2，这时数据变化较为平滑; 右图横坐标范围为-20 到 20，可以看到，此时在 x=0 附近看起来很像跃迁函数

4.1.2 基于 sklearn 的最佳回归系数确定

Sigmoid 函数输入为 z，z 的表达式为:

$$z = w_0 x_0 + w_1 x_1 + ... + w_n x_n \tag{4.2}$$

如果采用向量的写法，上述公式可以写成 $z = w^T x$ 的形式，其中 x 是分类器输入的数据，向量 w 也就是我们寻找的最佳参数，从而使分类器尽可能精确。我们可以使用 Python 的 sklearn 模块来进行 Logistic 回归的建模和交叉验证 (cv=10)。

rating class 的二分类问题的 Logistic 模型在测试集中的交叉验证结果显示，预测准确率为 52.85%。

4.1.3 虚拟变量转换

由于在基础自变量中，分类变量 content rating 和 main category 都被赋予连续的整数来分类，而不是建立多个虚拟变量，这可能是 Logic 回归准确率受限的原因。因此，我们考虑转换成虚拟变量的方法进行回归，并观察模型是否得到了优化。

考虑到 content rating 分为 5 类而 main category 分为 33 类，为避免过高的维度，仅将 content rating 转换成虚拟变量。在删去 content rating 的基础上，添加了 4 个虚拟变量，进行 rating class 对上述 16 个新自变量的 Logistic 回归和交叉验证 (cv=10)，最终得到的预测准确率仍为 52.85%。

两次结果相比较并没有显著差异，由此认为虚拟变量转换没能实现对模型的优化，

预测准确率依然处于较低的水平，这启发我们开始寻找其他潜在的更适合的模型来解释和预测 rating class 二分类问题。

4.1.4 自变量删减

考虑到回归效果并不理想，我们猜测基础自变量中存在很多对 rating class 没有贡献的自变量，并尝试简化模型。经多次尝试，交叉验证 (cv=10) 的结果显示，我们发现仅选取 reviews 和 updateDays 作为自变量时，预测准确率增加到了 57.31%。这可能是由于其他自变量在 Logistic 回归中属于无效特征，导致过拟合，所以在预测集中减少自变量个数预测准确率反而上升了。

4.2 随机森林算法

4.2.1 算法简介

随机森林算法是集成学习最具代表性的算法之一，是一种常用的分类和回归算法。它的原理是利用多个决策树模型各自独立地学习并做出预测，之后集成预测结果以提升模型的泛化能力，因此效果一般优于单个决策树模型。随机森林可同时处理数值型特征和类别型特征，并且有较高的稳定性。而它的一个主要缺点是由于其算法本身的复杂性，可能耗费大量的计算资源，但本文采用的数据量较小，不受该问题影响。因此，我们采用该模型对 APP 评分进行二分类，并评估其准确率。

4.2.2 模型与结果

在以 80% 数据量作为训练集，20% 为测试集的 train-test split 方法下，通过调整相应参数 criterion=’entropy’，n estimators=200，max depth=9，max features=10，输出的混淆矩阵和分类报告结果见表 4.1和表 4.2，其中分类正确率 65.70%，而对于高评分分类的正确率达 71%。经过交叉验证 (cv=10)，输出结果平均正确率为 63.77%。另外，关于各特征重要性的排序结果展示见图 4.2，评论数和最近更新日期的对于分类的重要性较高，而价格类特征 (price, free, iap) 的重要性较低。由此可见，评论数和最近更新日期对评分影响较高，价格对评分影响较低。

表 4.1: 随机森林的混淆矩阵

Accurate/Predict	Rating low	Rating high
Rating low	219	107
Rating high	142	258

表 4.2: 随机森林的分类报告

Result	precision	recall	f1 score	support
Unsuccessful	0.61	0.67	0.64	326
Successful	0.71	0.65	0.67	400

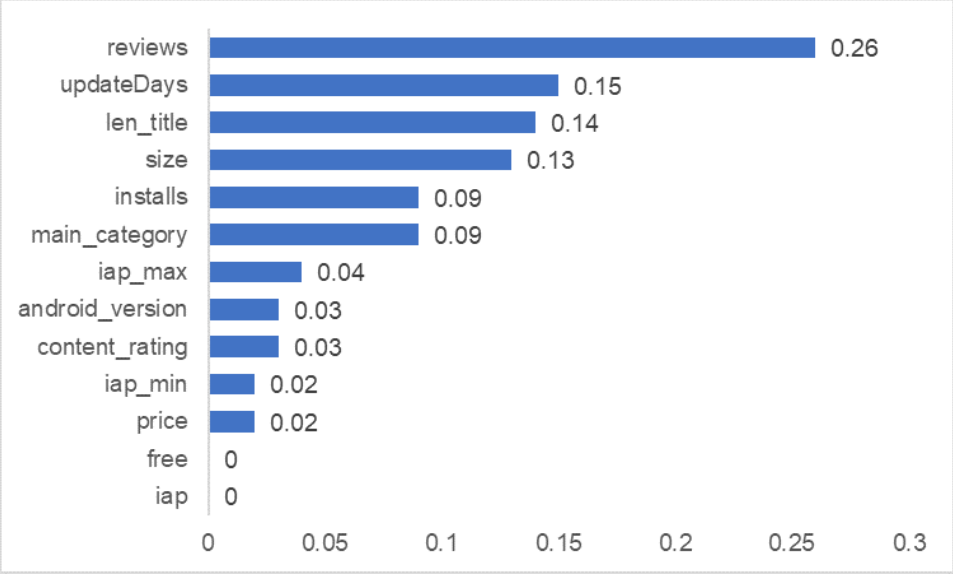


图 4.2: 随机森林的特征重要性结果

4.3 XGBoost 分类

4.3.1 算法简介

XGBoost 算法同样利用集成学习方法，它通过结合多个学习器的预测能力来得出最终结果。基础学习器一般是偏差较高的弱学习器，其预测能力仅略好于随机猜测。但每一个弱学习器都会添加一些重要的预测信息，通过有效地组合这些弱学习器，从而降低偏差和方差，形成一个强学习器。

4.3.2 模型与结果

在以 80% 数据量作为训练集，20% 为测试集的 train-test split 方法下，通过调整参数 max depth=3，objective='binary:logistic'，num round=12，eta=1，输出的混淆矩阵和分类报告结果见表 4.3和表 4.4，其中分类正确率 68.46%，高评分分类的正确率达 73%。图 4.3展示了调参后 XGBoost 模型中一个决策树的分类情况。关于各特征重要性的排序结果展示见图 4.4，可见评论数是对于评分分类最重要的特征。XGBoost 分类模型在交叉验证 (cv=10) 下，平均正确率为 64.62%，分类效果良好。

表 4.3: XGBoost 的混淆矩阵

Accurate/Predict	Rating low	Rating high
Rating low	225	101
Rating high	128	272

表 4.4: XGBoost 的分类报告

Result	precision	recall	f1 score	support
Unsuccessful	0.64	0.69	0.66	326
Successful	0.73	0.68	0.70	400

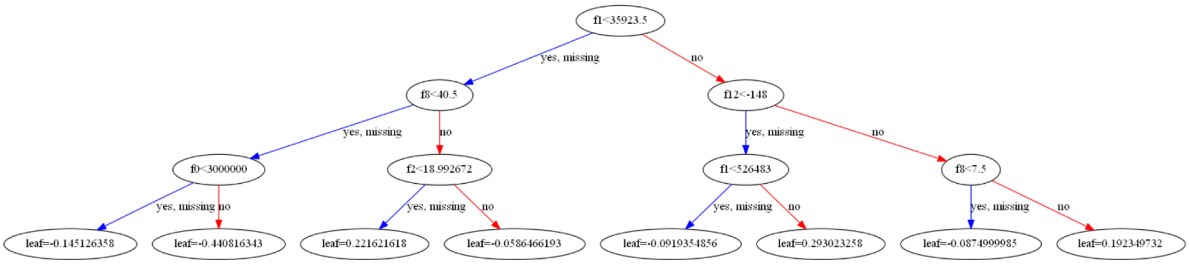


图 4.3: 调参后 XGBoost 算法的一个决策树

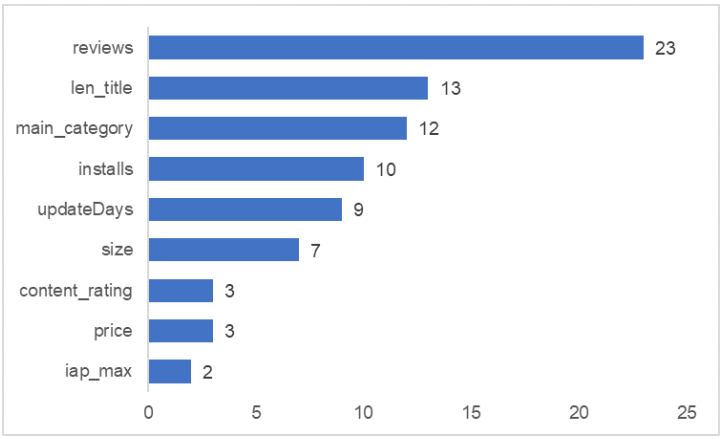


图 4.4: XGBoost 的特征重要性结果

4.4 神经网络分类

4.4.1 神经网络原理简介

神经网络算法是基于生物学人类大脑工作原理建立起来的算法，其本质是用不同权重的线性函数将神经层连接起来，在输出层设置损失函数进行训练，过程中用到前向与反向传播更新参数，最终使得网络的输出可最大限度地逼近训练集的结果，并可以稳健地预测测试集结果。

4.4.2 神经网络实验的具体设置

神经网络输入与输出的设置

基于前面算法：Logistic 分类、随机森林分类以及 XGBoost 分类中用到的同样标准，将 Google APP 数据集中的 APP 评分以 4.4 分（中位数）为截断，把其分为两类：低于 4.4 分的为“0”类，高于 4.4 分的为“1”类。采用的自变量数目仍与前面的算法一致：

共包含 installs, reviews 等 12 个维度，这 12 个维度上的数据类型为整数和浮点数类型。

训练集、测试集的划分与神经网络模型的设置

将 3629 个 APP 数据以 7:3 的比例随机划分为训练集和测试集，可以得到训练集有 2540 个数据，测试集有 1089 个 APP。在训练集上训练神经网络，并在测试集上检验训练出的神经网络。为了方便结果的复现，划分过程用到了随机种子。

神经网络的层数共有三层：包含两层隐藏层：分别包含 256 个神经元和 128 个神经元；输出层用到 softmax 函数，输出一个二元的向量，用于预测 APP 的分数 0,1 类别。模型的框架采用 tensorflow 中的 Keras 模块。由于 Adam 优化器在该数据集上的效果并不好，优化器采用最简单的 SGD，模型的参数随机初始化，在训练集上训练 500 个周期后停下。

4.4.3 神经网络的训练结果

神经网络在训练集上训练 500 个周期后的正确率在测试集达到了 59.96%，在测试集达到了 57.21%。神经网络训练过程中正确率的变化见图 4.5。

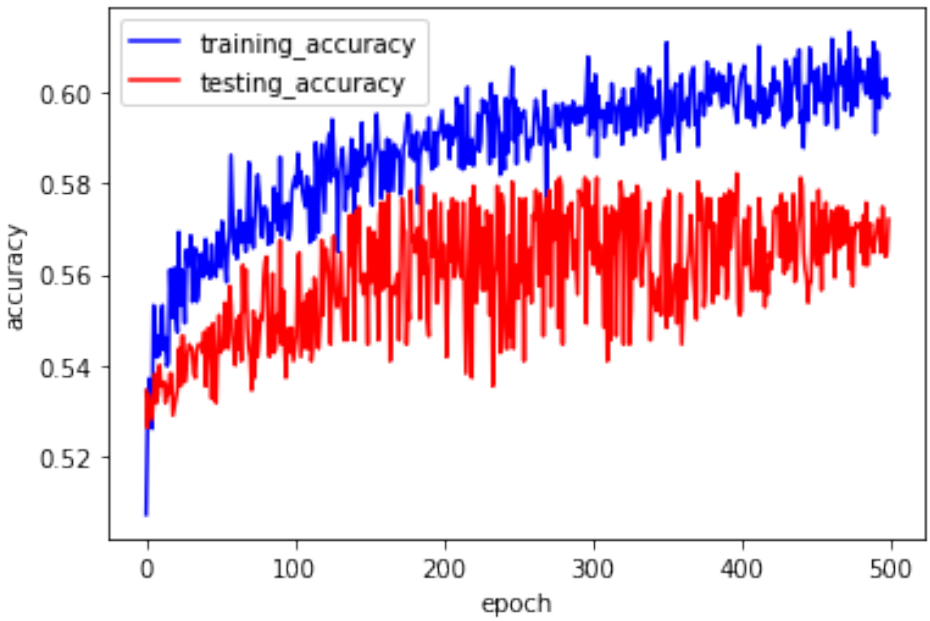


图 4.5: 神经网络训练 500 个周期训练集和测试集上的正确率展示图

由上图可知：神经网络的训练过程中测试集上正确率的波动较大，但总体趋势是上升的，且在 300 个周期之后正确率的波动逐渐变小，稳定在 57% 上下。

4.4.4 神经网络模型的预测情况

将训练好的模型应用在个别 APP 的类别预测上。推特 (Twitter) 的真实分数是 4.5 分，因此其真实类别为 1 类，神经网络模型对其的预测也是 1 类。Instagram 的真实分数是 4.4 分 (1 类)，经检验，模型能够将其正确预测为 1 类。此外，还有愤怒的小鸟、谷歌、声破天 (spotify)APP 都可被正确预测为 1 类。

但是，对于微信 (真实评分 3.8 分，属于 0 类)，模型将其错误预测为 1 类，说明微信拥有一些高分 APP 的属性。此外，QQ(真实评分 2.9 分，属于 0 类)，模型可以将其正确预测为 0 类，说明对于低分 APP, 分数离中位数越远，越容易被判断正确。

总的来说，神经网络用于 Google APP 的二分类预测正确率较低，需要进一步的主观调参工作来提高正确率，该算法可能不太适合用于该数据集。

4.4.5 模型对比

根据以上建立的 Logistic 回归、随机森林、XGBoost 和神经网络模型，交叉验证 (cv=10) 正确率结果如图 4.6。通过对比，可发现 Logistic 回归和神经网络的正确率在 58% 左右，而基于树模型的随机森林和 XGBoost 相较于其他模型效果更好，XGBoost 模型效果最优，其分类正确率为 65%。

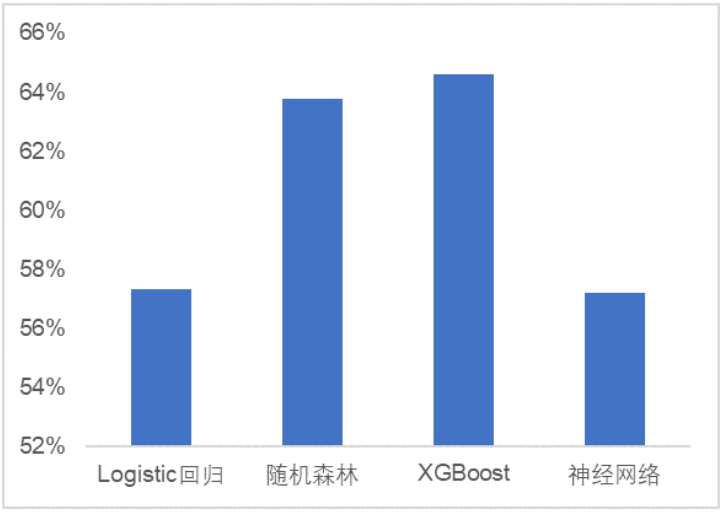


图 4.6: 各模型正确率比较

五 延伸、总结及展望

至此，我们完成了对 Google APP 数据集从爬取到清洗、再到算法实验的全部流程，但是数据集中包含的 APP 的 ikon 属性 (APP 的 logo 图片所在的网址) 并没有被我们用到，为了挖掘更多数据集背后的信息，我们决定对每个 APP 的 logo 图片进行处理与剖析，并期望得到一些有价值的信息。

这一部分还会对数据挖掘报告做一整体评价与总结，并对后续的工作作一展望。

5.1 延伸：对 logo 图片信息的挖掘

5.1.1 logo 图片的介绍

对于处理后的数据集共 3629 个 logo 图片网址，采用 skimage 模块直接通过网址读取图片，logo 图片的格式都标准化为 RGB 三通道形式 (黑白图片将其单通道的值重复赋给三个通道)，读取过程中有两张 logo 图片 (logo[236],logo[1010]) 文件破碎，因此将其剔除。最终可以得到 3627 张格式为 $512 \times 512 \times 3$ 的图片。读取的图片以音乐 APP spotify 为例作一展示，见图 5.1。

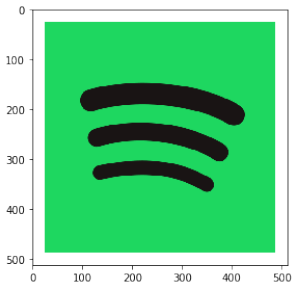


图 5.1: Spotify 的 logo 图片

5.1.2 对 logo 图片的特征提取

提取图片特征的主流网络是包含卷积层的 CNN，它们可以高效稳健地提取出图片潜在的某些特征。我们采用 VGG16 模型来提取图片的特征，模型的初始参数下载自已经在 ImageNet 分类任务上训练好的参数，我们认为这些参数直接用于 logo 图片的特征

提取是存在一定道理的。$512 \times 512 \times 3$ 的图片经过 VGG16 模型后将变为 $16 \times 16 \times 512$ 的高维特征张量。

5.1.3 对图片高维特征进行聚类

基于 VGG16 网络提取的高维特征，我们对 3627 张 logo 图片以 K=2 进行 K 均值聚类，遍历 3627 次后就能得到每个图片的聚类类别 (label 为 0 或 1)。由于特征的维数高达 $16 \times 16 \times 512 = 131072$ 维，程序运行速度较慢，为了提升速度，把代码放在 Google Colab 的 GPU 上运行。最终的部分聚类结果展示在图 5.2, 图 5.3, 图 5.4中。

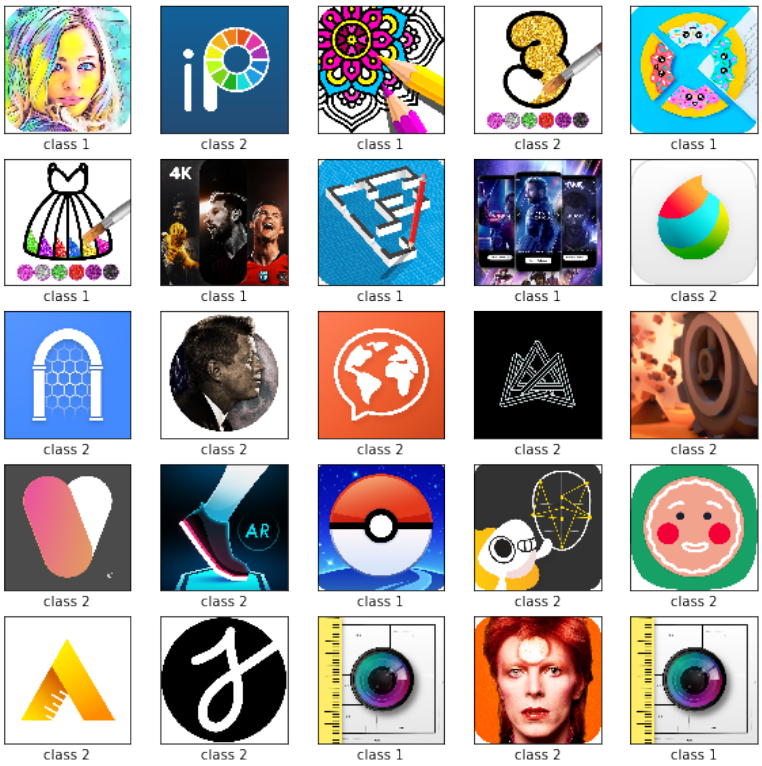


图 5.2: 第 1-25 张 logo 图片的聚类情况

从这些聚类结果中可以看出，第一类 (class 1) 中的 logo 图片大概率包含更复杂的图案与细碎的纹理，第二类 (class 2) 中的 logo 图片倾向于由一些简单的几何图形组成。这说明以 VGG16 提取出的特征做聚类并不是无规律的行为，是存在一定道理的。

5.1.4 探讨图片类别与 APP 评分、APP 类型的关系

在将 APP 以 logo 图片聚为两类的基础上，我们进一步想了解 logo 图片的类别是否对 APP 类型以及 APP 评分有一定效应的影响。

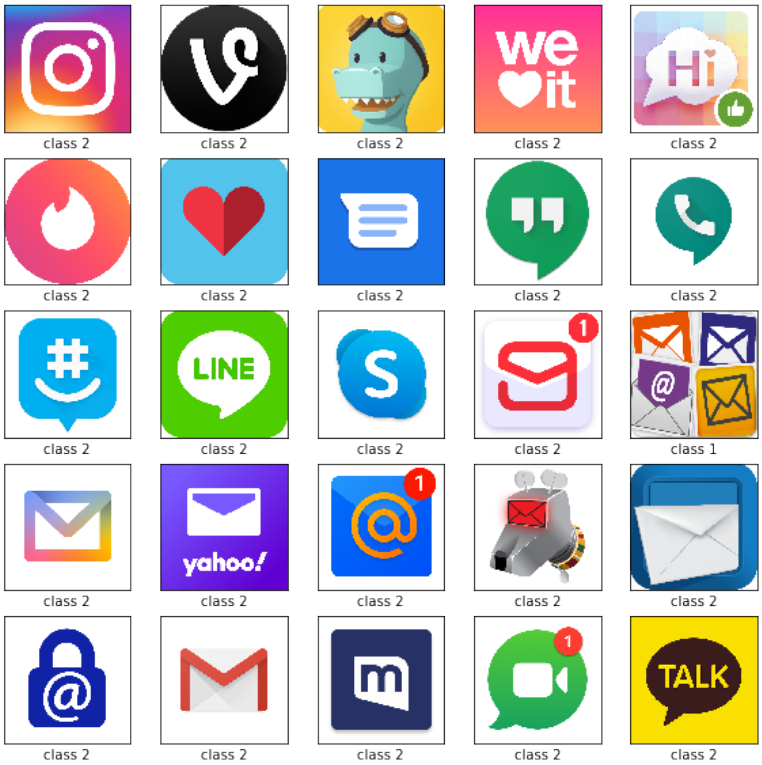


图 5.3: 第 176-200 张 logo 图片的聚类情况

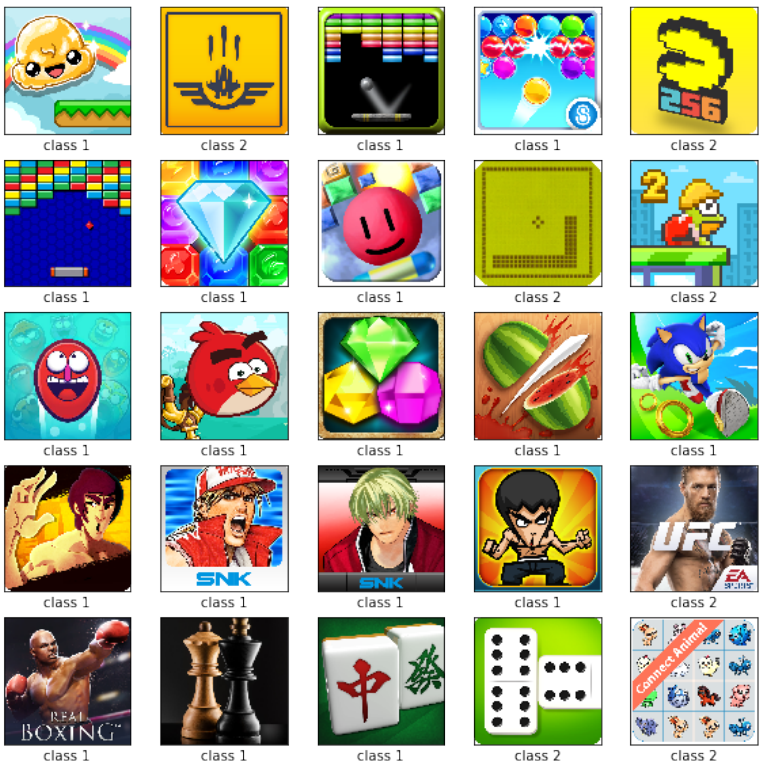


图 5.4: 第 1526-1530 张 logo 图片的聚类情况

图片类别与 APP 评分

经计算，class 1 的 APP 的均分为 4.33 分，class 2 的 APP 均分为 4.23 分，两者绝对差异为 0.1 分，相对差异约 2.3%, 画出两种图片类对应的 APP 评分分布见图 5.5。

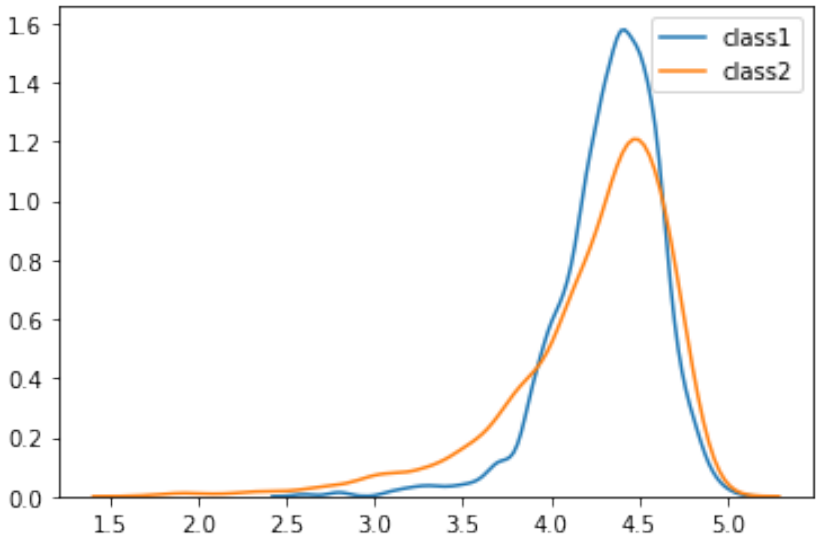


图 5.5: 两类 logo 图片对应的 APP 评分分布

可以看出第二类相较于第一类在低分区更加厚尾，这说明第二类的分数更为分散，且低分的比例更多。因此，APP 的 logo 设计可能与它的评分存在一定关系。也许做得更复杂精美一点的 logo（第一类：图 5.5中的蓝色曲线），评分会更高一点。

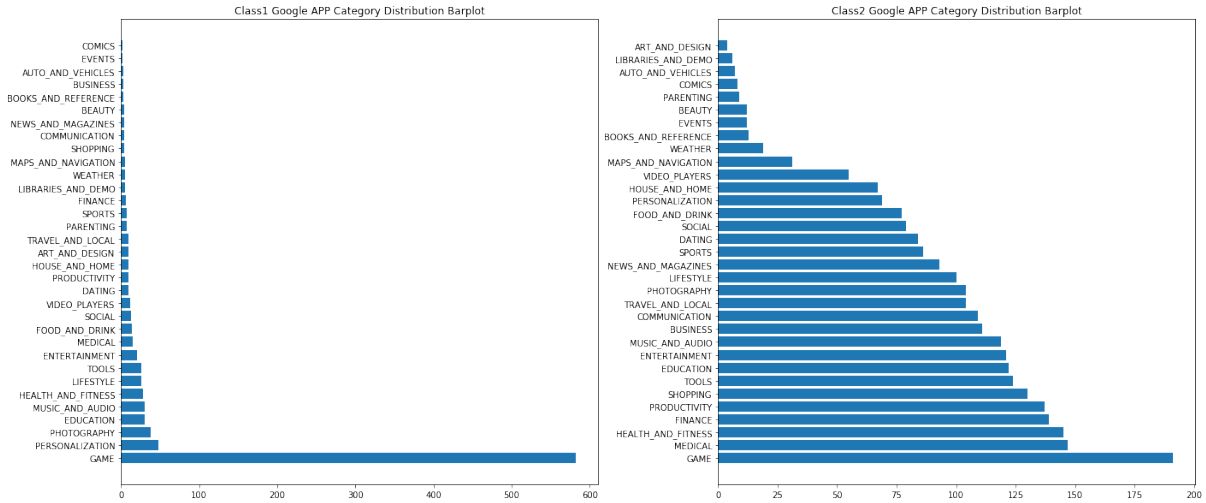
图片类别与 APP 类型

进一步地，想要探索 logo 图片类别是否与 APP 的类型有某些关系。可知 APP 数据在经初步处理后：3629 个 APP 共有游戏、运动、教育、艺术设计等共 33 个类型。在不按图片分类时，所有 APP 的类型分布见图 5.6c。

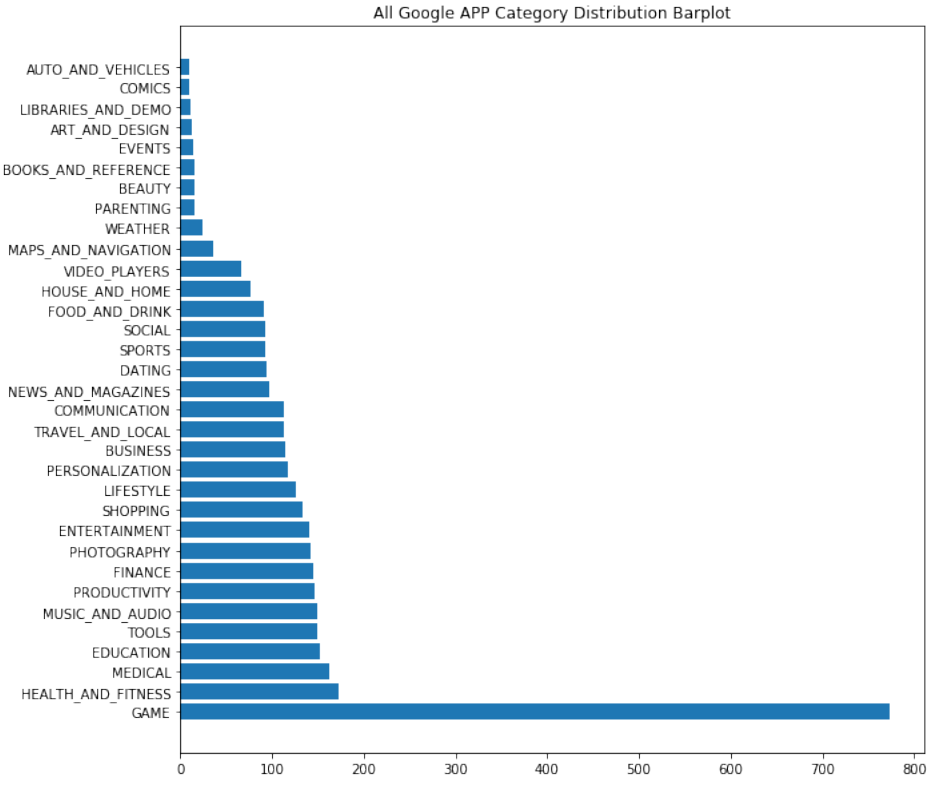
隶属于第一类 logo 图片类的 APP 类型分布图见图 5.6a。

隶属于第二类 logo 图片类的 APP 类型分布图见图 5.6b。

由图 5.6可以看出：相较于不分类的整体，第一类（logo 图片设计复杂）的 APP 类型中游戏类的主导地位变得更加明显，因此我们可以说游戏类 APP 更喜欢复杂的 logo 设计。而第二类（logo 图片设计简单）中 APP 类型的分布变得更均衡，说明简洁的设计几乎是所有类型 APP 设计 logo 的普适准则。



(a) (b)



(c)

图 5.6: 图片类别一、类别二的 APP 类型分布图以及整体的 APP 类型分布图

5.2 报告的评价与总结

本次数据挖掘报告把重点放在了数据清洗以及具体算法模型在数据集的套用上。数据清洗本着尽量不减少数据集本身信息的目的将数据集的维度、属性做了一定的调整，方便后续的探索性分析以及算法实验。

实验聚焦逻辑回归、随机森林、XGBoost 以及神经网络四种算法、对 APP 的评分高低进行了建模与检验、预测，并横向对比了这些算法在 Google play APP 数据集上的效果。在实验的延伸部分，加入了 APP 的图片属性，用 K 均值算法对 logo 图片进行了聚类，并挖掘到一些有价值的信息。

5.3 思考与展望

在对报告做了总结的基础上，不难发现本次报告仍有一些不足和可以改进的地方。

在数据清洗的过程中，我们完全基于人工对数据集的理解，背后缺乏客观的清洗准则和理论支撑，因此清洗过程可能会造成原数据集的可研究成分一定程度的流失。

在四种算法的横向对比中，我们不能解释算法在 Google play APP 数据集上表现优劣的本质原因，比如：为什么逻辑回归、神经网络的效果要比 XGBoost 差。此外，我们也无法说清提升各算法表现的调参方法的具体指标。

在延伸的部分：对 logo 图片的处理上：提取图片特征的过程中，下载的训练好的 VGG16 模型参数其实可以基于 logo 图片做进一步的训练和更新 (迁移学习)，但由于计算资源不足，对网络参数的更新耗时费力，因此直接使用了 ImageNet 上已经训练好的参数。

综上，Google play APP 数据集还有很多信息有待进一步的探索与挖掘，本次报告忠于研究目标进行了发散性的研究与分析，仍有很大进步和改善的空间。

参 考 文 献

[1] Hastie, Trevor, Tibshirani, Robert, Friedman, Jerome. [Springer Series in Statistics] The Elements of Statistical Learning || Boosting and Additive Trees[J]. 10.1007/978-0-387-84858-7(Chapter 10):337-387.

[2] Burr T . Pattern Recognition and Machine Learning. Christopher M. Bishop[J]. Journal of the American Statistical Association, 2008, 103(June):886-887.