**基于Scikit-learn的电影票房预测系统的设计与实现**

作者姓名：

学科专业：软件工程

校内导师：

企业导师：

完成时间：二○一六年八月十五日

**Limited**

University of Science and Technology of China

A dissertation for master’s degree

of engineering



**The Design and Implementation of Movie Box OfficePrediction System Based on Scikit-learn**

Author’s Name：Mu Yang

Speciality：Software Engineering

Supervisor： Prof. Baoqun Yin

Advisor： Dr. Minlong Lin

Finished time: Aug 15th, 2016

中国科学技术大学学位论文原创性声明

本人声明所呈交的学位论文,是本人在导师指导下进行研究工作所取得的成果。除已特别加以标注和致谢的地方外，论文中不包含任何他人已经发表或撰写过的研究成果。与我一同工作的同志对本研究所做的贡献均已在论文中作了明确的说明。

作者签名：\_\_\_\_\_\_\_\_\_\_\_ 签字日期：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

中国科学技术大学学位论文授权使用声明

作为申请学位的条件之一，学位论文著作权拥有者授权中国科学技术大学拥有学位论文的部分使用权，即：学校有权按有关规定向国家有关部门或机构送交论文的复印件和电子版，允许论文被查阅和借阅，可以将学位论文编入有关数据库进行检索，可以采用影印、缩印或扫描等复制手段保存、汇编学位论文。本人提交的电子文档的内容和纸质论文的内容相一致。

保密的学位论文在解密后也遵守此规定。

□公开 □保密（\_\_\_\_年）

作者签名：\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 签字日期：\_\_\_\_\_\_\_\_\_\_\_\_\_\_

# 摘要

近年来，随着电影行业迅猛发展，年度电影总票房持续快速增长。对电影票房进行预测可以给电影的制片方和宣发方提供票房数据支持，并在此基础上制更加完善的营销和宣传策方案。传统的电影票房预测往往由各大院线的相关工作人员完成。这种由人工完成的票房预测大多存在预测精度低、预测主观性强、人力资源成本高等问题，因此不能较好地满足实际需求。

针对人工进行票房预测存在的种种问题，本文设计了一个智能化的电影票房预测系统。该系统使用Python爬虫技术对外网电影相关数据进行采集，并对这些数据进行解析、过滤、转换等预处理操作，然后使用HDFS（Hadoop Distributed File System）文件系统结合Mysql数据库对完成预处理操作后的数据进行存储。系统的特征抽取模块利用特征工程相关技术和算法对每部电影的特征进行量化计算和数值修正。最后，系统使用普通最小二乘法、局部加权线性回归算法、随机森林算法和随机梯度回归树四种机器学习算法进行单模型学习训练，并对每个算法最好的模型进行算法融合，得到最终电影票房预测结果。

本文的最后使用2013年1月到2016年4月总共947部电影进行算法模型的训练与评测，并以判定系数R2值作为评判标准。系统最终的R2值达到0.9301，表明系统具有较高准确率。

**关键字：**电影票房，预测，大数据，Scikit-learn，机器学习

# Abstract

In recent years, with the rapid development of movie industry, total annual box office continued to grow rapidly. To predict box office can provide data to help the movie's producer and propagator to formulate better marketing and promotion strategies. Traditional movie box office prediction are usually done by the relevant staff of major theaters. Box office prediction completed by artificial way is of low accuracy, strong prediction subjectivity and high human resources cost, it can not satisfy the actual needs well.

In order to change the situationof artifical box office prediction, we design an intelligent movie box office prediction system. The system uses Python crawler technology to collect network movie data, and process these data by parsing, filtering and conversion. Then using Hadoop Distributed File System and Mysql database to storage the preprocessed data. The feature extraction module of the system uses feature engineering technology to calculate and correct features of each movie. Finally, the system uses oridinary least square,locally weighted linear regression, random forest andgradient boosting regression treemachine learning algorithms to train four different models of each algorithm and combine these four models to calculate the final movie box office prediction value.

At the end of this paper, we use 947 movies from January 2013 to April 2016 to train and test algorithm models. We use coefficient of determination R2 as the criterion. The R2 of the system is 0.9301, indicating that the system has a high precision rate.

**Key words:**movie box office，prediction，big data，Scikit-learn，machine learning

**目录**

[摘要 I](#_Toc459326582)

[Abstract II](#_Toc459326583)

[目录 III](#_Toc459326584)

[第1章绪论 1](#_Toc459326585)

[1.1 课题的背景与意义 1](#_Toc459326586)

[1.1.1 选题的依据 1](#_Toc459326587)

[1.1.2 选题的意义 2](#_Toc459326588)

[1.2 国内外文献资料综述 3](#_Toc459326589)

[1.2.1 电影票房预测研究进展 3](#_Toc459326590)

[1.2.2 大数据处理系统调研 4](#_Toc459326591)

[1.2.3 机器学习算法库调研 5](#_Toc459326592)

[1.3 本人的主要工作 6](#_Toc459326593)

[1.4 本文的组织结构 7](#_Toc459326594)

[第2章相关技术与理论分析 8](#_Toc459326595)

[2.1 Scikit-learn机器学习算法库 8](#_Toc459326596)

[2.2 Apache Hadoop 9](#_Toc459326597)

[2.2.1 Hadoop HDFS 9](#_Toc459326598)

[2.2.2 MapReduce计算框架 10](#_Toc459326599)

[2.3 Python爬虫技术 11](#_Toc459326600)

[2.4 Beautiful Soup 12](#_Toc459326601)

[2.5 特征选择技术 12](#_Toc459326602)

[2.6 预测算法 14](#_Toc459326603)

[2.7 本章小结 17](#_Toc459326604)

[第3章需求分析 18](#_Toc459326605)

[3.1 功能性需求 18](#_Toc459326606)

[3.1.1 需求概述 18](#_Toc459326607)

[3.1.2 需求建模 18](#_Toc459326608)

[3.2 非功能性需求 22](#_Toc459326609)

[3.2.1 准确性需求 22](#_Toc459326610)

[3.2.2 同步性需求 23](#_Toc459326611)

[3.2.3 健壮性需求 23](#_Toc459326612)

[3.2.4 可扩展性需求 23](#_Toc459326613)

[3.2.5 其它需求 23](#_Toc459326614)

[3.3 可能性分析 23](#_Toc459326615)

[3.4 本章小结 24](#_Toc459326616)

[第4章概要设计 25](#_Toc459326617)

[4.1 系统整体架构 25](#_Toc459326618)

[4.2 系统功能模块划分 27](#_Toc459326619)

[4.3 本章小结 31](#_Toc459326620)

[第5章详细设计与实现 32](#_Toc459326621)

[5.1 用户登录模块详细设计与实现 32](#_Toc459326622)

[5.1.1 用户登录整体流程 32](#_Toc459326623)

[5.1.2 静态类设计和核心接口实现 33](#_Toc459326625)

[5.2 电影数据查询模块 35](#_Toc459326626)

[5.2.1 静态类设计和核心接口实现 35](#_Toc459326627)

[5.3 数据抓取模块 36](#_Toc459326628)

[5.4 数据预处理模块 38](#_Toc459326629)

[5.4.1 数据解析 38](#_Toc459326630)

[5.4.2 数据存储 39](#_Toc459326631)

[5.4.3 数据过滤 39](#_Toc459326632)

[5.5 特征抽取模块 39](#_Toc459326633)

[5.5.1 离线特征计算 40](#_Toc459326634)

[5.5.2 特征数值修正 44](#_Toc459326635)

[5.6 算法预测模块 45](#_Toc459326636)

[5.6.1 数据集划分 46](#_Toc459326637)

[5.6.2 普通最小二乘法 46](#_Toc459326638)

[5.6.3 局部加权线性回归 46](#_Toc459326639)

[5.6.4 随机梯度回归树 47](#_Toc459326640)

[5.6.5 随机森林 48](#_Toc459326641)

[5.6.6 算法融合 48](#_Toc459326642)

[5.7 本章小结 50](#_Toc459326643)

[第6章系统部署与测试 51](#_Toc459326644)

[6.1 测试环境 51](#_Toc459326645)

[6.1.1 硬件环境 51](#_Toc459326646)

[6.1.2 软件环境 51](#_Toc459326647)

[6.2 系统部署安装 52](#_Toc459326648)

[6.3 功能测试 54](#_Toc459326649)

[6.3.1 用户登录功能测试 54](#_Toc459326650)

[6.3.2 电影数据查询功能测试 54](#_Toc459326651)

[6.3.3 数据抓取功能测试 56](#_Toc459326652)

[6.3.4 数据预处理功能测试 57](#_Toc459326653)

[6.3.5 特征抽取功能测试 57](#_Toc459326654)

[6.3.6 算法预测结果评估 60](#_Toc459326655)

[6.4 性能测试 63](#_Toc459326656)

[6.5 本章小结 64](#_Toc459326657)

[第7章总结与展望 65](#_Toc459326659)

[7.1 论文工作总结 65](#_Toc459326661)

[7.2 不足与展望 66](#_Toc459326662)

[参考文献 67](#_Toc459326663)

[致谢 69](#_Toc459326664)

# 第1章绪论

## 1.1 课题的背景与意义

### 1.1.1 选题的依据

随着生活水平逐渐提高，电影成了丰富人们日常生活和精神世界的不可或缺的元素之一。电影市场在近些年发展得十分迅猛。据统计，2009年电影总票房为62亿元，此后每年保持40%左右的票房增长速度。2014年的年度总票房为296.3亿，而2015年的年度总票房增长到440亿，与前一年相比，涨幅高到48.73%，其中国产电影在总票房中的占比达到60%。面对如此巨大的票房市场，国内互联网三大巨头百度、腾讯和阿里巴巴在去年相继成立了自己的影视子集团，由此可见电影行业具有极大发展潜力。

传统的电影票房预测往往由各大院线工作人员（一般是各个院线的经理）根据历史数据和个人经验完成。但近年来，由于影片量级的快速增长、观影人数的急剧扩张以及宣传营销方式和社交网络的多样化发展，影响影片最终票房的因素变得越来越复杂，例如演员知名度、影片剧情、排片数、宣传力度、社交舆情、传播热度等[1]。根据个人经验和历史数据预估出的电影票房大多存在以下问题：第一，影响影片票房的因素的数据量庞大且数据差异性显著，个人不能准确地量化这些影响因素；第二，个人预测电影票房经常存在由于个人主观因素而导致非客观预测的现象；第三，近年来电影数量急剧增加，通过人工进行票房预测的工作量巨大，需要较多人力资源。上述问题最终会使得人工预测出的电影票房与票房实际值相差较大，即准确率比较低。而精准的电影票房预测值能在电影上映早期给电影的投资方和制片方带来营销和宣传上的数据支持，从而能在电影上映阶段获取更多观影票房。

随着大数据相关技术[2-3]和工具的快速发展，与之相关的统计学理论[4]和机器学习算法的应用越来越成熟，例如使用混合模型[5]进行推荐系统[6]的研发、利用朴素贝叶斯算法进行垃圾邮件的识别、利用随机森林算法对特征进行选择[7]、利用深度学习算法进行语音识别[8]等都是业界较为实际的应用。为了解决人工进行电影票房预测存在的种种问题，急需构建一个基于人工智能技术，例如数据挖掘和机器学习算法，并融合当下优秀的大数据处理工具，例如Scikit-learn以及Hadoop等进行电影票房预测的智能系统。

### 1.1.2 选题的意义

根据以上介绍的背景，伴随着网络通讯设备如智能手机、平板电脑、智能手表等的迅速普及，网络数据暴增，我们进入了大数据时代。大数据具有整体量级庞大、增长速度极快、类型繁杂等诸多特点。当前我国以及世界上其它部分国家已经发起了大数据战略。科研工作者和软件开发工程师可以从海量数据中提取出与实际需求相关的数据形式，并对这些数据进行数据清洗、数据转换、特征分析等操作得到目标数据，然后使用数据挖掘相关技术和算法对目标数据进行挖掘和分析，最后将这些数据应用于实际决策之中。在大数据相关技术不断创新的背景之下，业界出现了越来越多的高效大数据处理框架、工具和算法。如何充分利用这些大数据技术来解决电影票房预测问题是本文的意义所在。具体包含三个方面：第一，工程研究意义，GFS（Google File System）[9]和HDFS [10]等分布式文件系统在大数据技术的发展下应运而生，其中GFS由谷歌公司研发并在其内部广泛应用，而HDFS则在GFS原理的基础上研发并与Hadoop MapReduce[11]计算框架一同被业界广泛使用。Hadoop MapReduce作为一种分布式处理框架，拥有分布式计算能力强、容灾性强以及可扩展性好等优点，因此被大量应用于大规模离线计算任务。为了解决数据计算时重复I/O操作而产生的计算资源大量浪费的问题，基于内存计算的大数据框架的研究得到突破性进展，并诞生了以Spark[12]为代表的基于内存RDD（Resilient Distributed Datasets）算子的计算平台。Spark对计算过程中产生的结果进行缓存，而不是存储在磁盘上，因此具有计算效率高、I/O操作次数少等优点。此外，由于与影片相关的数据种类较多，且大部分数据无规律的存储在外部Web网站上，为了收集这些数据需要使用Python爬虫[13]技术进行大规模数据采集和存储。本文研究如何利用这些大数据工具进行电影票房数据的收集、存储、处理以及计算，具有工程研究意义。第二，理论研究意义，在大数据技术十分火热的同时，如何利用合适的理论算法挖掘出数据背后的价值是工业界和学术界研究的一个重要方向，与此相关的学科如数据挖掘[14]、机器学习[15]、模式识别等正受到计算机相关工作者越来越高的重视。在上一小节中，简要介绍了目前机器学习算法在业界的一些典型实际应用。当前，业界具有众多机器学习算法开源库，如Scikit-learn，它具有丰富的API接口、文档详细、使用简单等优点。本文研究如何利用机器学习算法进行电影相关数据的特征处理、数据计算以及准确票房预测，因此具有理论研究意义。第三，实际应用意义，本文的落脚点是设计一个智能化的电影票房预测系统，从而解决传统人工预测存在的的种种问题，并释放人力资源，具有很强的实际应用意义。

## 1.2 国内外文献资料综述

### 1.2.1电影票房预测研究进展

电影票房预测系统是一种智能的自动化系统，融合了数据收集、数据清洗、数据存储、数据计算、以及票房预测等较多功能，旨在弥补人工预测存在的诸多问题，提高票房预测的准确率和效率，为电影制片方和发行方提供准确的票房预测数据支持，从而进行更好的电影宣传和营销。

对电影票房进行预测起源于上世纪八十年代中后期，美国密歇根州立大学经济学家Barry Litman对当时最新的七百多部电影进行分析建模，并在此基础之上对影片票房进行预测[16]，他把决定电影最终票房的因素分成三类：影片创意、上映时间和影片营销方式，然后他根据这三类变量得出一个回归方程并进行票房预测。2006年，美国的两位学者Ramesh Sharda以及Dursun Delen使用神经网络算法对电影票房进行分档预测[17]，但是其准确率只有36.9%。此外，国外的Sitaram Asur等人在2010年使用推特的社交数据对票房进行了预测[18]。Berman等人提出使用反馈神经网络对票房预测盈利与否[19]，而不是预测票房实际值，Berman等人描述的神经网络只有一个隐藏层导致模型过于简单，实际使用价值不高。利用计算机技术进行电影票房预测真正进入人们的视野，是谷歌公司在2013年发表的一篇名为《Quantifying Movie Magic with Google Search》[20]的文章。从该文可知，谷歌主要使用影片排片数和相关搜索数据对票房进行预测。谷歌对外宣称他们预测的影片票房值在测试集上的判定系数R2为94%（其中R2取值区间为0到1，越大表示预测结果与实际票房越吻合，即预测效果越好），但他们一直没有公布对即将上映的电影的预测结果，因此无法对其准确率进行验证。

在国外对电影票房进行预测研究的同时，国内学术界和工业界也在这一领域展开了探索。百度投资子集团爱奇艺2014年对电影票房进行了预测，但其系统并未对外公开。而华影BRP近年来一直在对外公布单个影片的预测报告，但其系统的整体情况从未公开说明。重庆大学郑坚等2013年利用反馈神经网络对电影票房进行了分档预测[21]，其实验分类预测准确率高达93.69%，但其团队并未对票房的分布区间和具体值进行预测，因此不能满足实际业务的需求。国内的罗捷等人使用电影的评价数据，包括电影评分和电影评论数据对进口电影票房进行了预测[22]，但这种预测是基于后验数据而言，并不具备大规模实际生产要求，并且其实验对象只是针对国外电影，忽略了目前日益火爆的国内电影。

根据以上分析，电影票房预测经历了从人工预测到使用计算机技术进行预测这一过程。对电影票房预测系统进行进一步的研究十分必要，需要结合实际业务需求，并不断提高系统的预测准确率和预测效率。

### 1.2.2 大数据处理系统调研

近年来，与大数据技术相关的应用已经扩散到各个领域之中，例如医疗、金融、智能交通、智慧城市、信息安全等领域，将大数据与云计算相结合已成为了工业界的一个趋势。面对如此海量的数据，如何存储和高效地挖掘出人们关心的那一小部分数据是学术界和工业界一直在研究的问题。随着硬件计算性能地提升，诞生了许多不同的大数据处理系统。目前，大数据的处理任务主要分为批处理任务和实时处理任务，其中批处理任务是指实时性要求不高，可离线计算的任务。而实时处理任务则需要较高的响应速度。此外，大数据的存储主要基于键值对形式的海量分布式文件系统[23]。

大数据系统真正被人们所认识是谷歌公司2003年研发的GFS文件系统以及2004年研发的MapReduce分布式并行计算框架。但当时谷歌公司只公布了设计文档，并没有开源这两个系统。2006年，Hadoop发明者Doug Cutting根据谷歌公司的相关文档研发出了HDFS文件系统和MapReduce计算框架。Hadoop MapReduce主要用来处理离线大规模分布式并行计算任务，而HDFS则是一个能够对数据进行分布式并行存储的文件系统。在此基础之上，诸如Hbase，Cassandra[24]等诸多开源工具逐渐进入人们的视野。随着Hadoop的广泛应用，我国的互联网公司相继研发出自己的云计算应用平台。腾讯于2009年推出独立研发的云平台，目前其集群规模已达到上万台服务器；百度推出大数据处理引擎，支持TB级别数据查询，在此基础上研发的百度云盘是当下应用最广泛的云盘之一；另一互联网巨头阿里巴巴研发并推广的云平台ODPS[25]是国内先进的云平台之一，并已大规模投入商用和学术研究使用。阿里云在Sort Benchmark 2015年排序竞赛中对100TB数据进行排序仅仅花费了377秒，打破历史记录。除此之外，国内的七牛云，新浪云等也被越来越广泛的应用。

由于批处理系统在处理实时性要求高、响应速度要求快的任务上不具备优势，实时处理成了大数据技术实际生产中的一个重要研究问题。实时处理系统一般分为两类，一类是一次处理一批数据，这类任务一般吞吐量较高；另一类是一次处理一条数据，即流式计算，这类任务吞吐量较小。实时处理系统在谷歌公司于2010年推出Dremel[26]之后得到迅速发展。2011年推特公司对外发布其内部使用的Storm计算平台，其具有流式处理数据的功能。2009年诞生于伯克利大学的大数据计算系统Spark经过若干年的积累，目前已是工业界主流的实时处理平台之一，Spark的具有批处理、实时处理、交互式查询、图计算等功能，是一种基于内存RDD数据结构的计算框架。Spark对计算过程中产生的临时结果进行缓存，减少了中间计算过程的I/O开销，因此具有计算速度快的优点。由于Spark的计算是在内存中进行的，因此需要集群具有大容量的内存设备，而本文研究的电影票房预测系统大多数计算任务是离线计算形式，因此本系统没有采用Spark平台而使用了Hadoop计算平台。

### 1.2.3机器学习算法库调研

在大数据技术迅猛发展的同时，机器学习理论也越来越成熟。机机器学习是人工智能的核心，其主要内容是如何让计算机根据历史数据学习得到一个算法模型，并用该模型对实际数据进行计算得到一个预测结果。机器学习算法按照学习方式不同可以分为四大类：第一，监督式学习算法，该类算法的所使用的数据都有一个固定的标签，当算法学习完一个模型后对实际数据进行预测得到该数据的标签，主要应用包括回归和分类问题。第二，非监督式学习算法，该类算法的实际数据不具备明确的标签，因此算法需要学习数据潜在的关系，主要的应用包括关联规则以及聚类等问题。第三，半监督式学习，该类学习算法的训练数据往往部分有标签而剩余部分没有被标识。第四，强化学习，在这种学习方式下，算法模型需要立即对输入数据做出相应调整，使得奖励函数值最大，常见的算法包括Q-Learning[27]等。机器学习的算法根据实际应用又可以划分为回归算法、分类算法、聚类算法、关联算法等类别。

出于实际应用效率考虑，业界已经有众多机器学习算法库，这些算法库对各类算法进行封装并提供API接口，使用者只需要按照API接口要求进行调用并调整参数即可，调参是使用机器学习算法较为重要的一个步骤，好的参数组合往往能在一定程度上提升算法最终效果。Libsvm[28]是由国立台湾大学的Chin-Jen Lin博士于2001年使用C语言研发，Libsvm主要提供了非线性的的svm分类器算法，由于Libsvm具有高效、算法实现好和调参便捷等优点，其一经发布便受到广泛的关注和应用，Chin-Jen Lin博士于2008年研发并发布了用于大规模线性计算的Liblinear[29]。Liblinear提供了逻辑回归、线性SVM等线性算法接口，与上一代产品Libsvm一样具许多优点并提供了诸如C语言、Java语言以及Python等语言的API接口，遗憾的是实现的算法种类过少。Spark Mlib是Spark生态系统中的一个机器学习算法库，Spark Mlib实现了大多数常用的机器学习算法并与Spark RDD完美结合。随着深度学习理论的不断完善，2015年谷歌公司开源了其内部研发的机器学习算法库Tensorflow，与Theano、Caffe和Torch类似，它主要应用于人工神经网络和深度学习相关的问题，但需要大量的硬件计算资源和运行时间。由于Python具有开发速度快、高效以及开源库较多等优点，近年来，在数据处理上具有越来越显著的优势。基于Python研发的开源机器学习算法库Scikit-learn提供了常用机器学习算法和工具，具有算法实现好、文档齐全、算法种类多等优点，已经是该领域最受欢迎的机器学习算法库之一。由于Scikit-learn的这些优点，本文着重讨论基于Scikit-learn的电影票房预测系统的设计与实现。

## 1.3本人的主要工作

本人在基于Scikit-learn的电影票房预测系统的设计与开发中，参与了大量工作，具体包括如下内容：

1、分析系统需求与架构，调研实现系统应采用的理论和技术；

2、设计票房预测系统的特征工程，结合业务逻辑，从多维度挖掘特征；

3、实现数据抓取模块，从外部网站抓取与电影相关的数据；

4、实现自动化数据预处理和特征抽取模块；

5、实现票房预测模块，使用不同算法对历史数据训练得出的模型进行交叉

验证，得出每个算法的最好模型，并进行算法融合计算最终票房预测值；

6、对系统进行测试，确保系统的功能和性能符合要求；

7、分析系统存在的不足，并给出解决方案。

## 1.4 本文的组织结构

本论文共分为七个章节，每章主要讨论内容如下：

第一章为本文的绪论部分，主要介绍了本文的选题背景和研究意义，同时讨论了国内外相关领域的发展趋势，并对本人的主要工作进行了简要的说明；

第二章为相关理论原理与技术的介绍，主要介绍了整个系统开发过程中用到的工具，如Scikit-learn、Hadoop MapReduce计算框架、HDFS文件系统等，并分析了为什么要使用它们，同时详细描述了系统所需要使用的算法理论；

第三章对系统了行了需求分析，主要讨论了系统的功能性需求和非功能性需求，并对系统的可行性进行了讨论；

第四章阐述了系统的概要设计，主要分析了系统的整体架构和系统模块划分，并对每个模块的功能和数据接口设计进行了描述；

第五章主要讨论了票房预测系统的详细设计与实现，深入阐述了每个功能模块的具体技术细节以及相关算法；

第六章是本文的测试部分，主要介绍了系统的部署方法并对系统的功能和性能进行了测试分析；

第七章是工作的总结与展望，对已经完成的工作进行总结，同时提出了本系统进一步优化的方向。

# 第2章相关技术与理论分析

本章主要对本系统设计与开发的过程中使用到的相关理论与技术进行分析，包括机器学习算法库Scikit-learn、大数据平台Hadoop（主要分析Hadoop生态圈中的文件系统HDFS和并行计算框架MapReduce）、Python网络爬虫技术以及机器学习算法中解决回归预测问题的普通最小二乘法、局部加权线性回归、随机森林等算法。

## 2.1 Scikit-learn机器学习算法库

Scikit-learn是一个Python机器学习算法库，由David Cournapeau在2007年发起的一个Google Summer of Code项目衍生而出。从发起之日开始，该项目便在Github上开源，目前该项目已经拥有600多名源码贡献者，并由一群志愿者维护。由于Python在进行数据处理时具有高效、简洁、第三方工具包多等优点，在数据挖掘和机器学习领域，Python备受科学研究者和工程师的青睐。Scikit-learn基于Python开发，并具有执行效率高、使用简单、无访问限制、BSD开源协议等优点，因此在数据挖掘和机器学习领域是当下最受欢迎的算法库之一。

具体来说，与其它算法库相比，Scikit-learn拥有下述优点：

1. 基于Python开发，建立在SciPy科学计算算法工具包之上，并与数值计

算工具包NumPy以及绘图框架MatPlotlib完美融合，方便使用者完成从数值处理到结果图形化展示等工作；

2、Scikit-learn对机器学习算法实现高效，API接口使用方便，并且实现的

算法种类多；

3、Scikit-learn提供的官方文档详细、丰富，方便使用者阅读；

4、跨平台性，Scikit-learn可在Windows、Linux以及Mac OS等诸多平台上

使用。

正是由于Scikit-learn与Python的完美结合并具有上叙述的诸多优点，在本系统的开发与设计过程中，选择其提供的相应预测算法对目标影片抽取出的特征进行票房预测。

## 2.2 Apache Hadoop

Hadoop由Doug Cutting在2006年根据谷歌公司内部的文件系统GFS和MapReudce计算框架研发并对外公布源代码，其整体架构由Java实现。Hadoop解决了实际应用中大数据领域存在的如何存储数据和如何对数据进行计算等问题。目前，Hadoop系统已经拥有了诸如HDFS、HBase、Hive、Pig等开源子项目，本系统主要用到了Hadoop的存储和计算能力，因此本章着重介绍HDFS和Mapreduce。

### 2.2.1 Hadoop HDFS

HDFS是一个主/从体系结构的分布式文件系统。HDFS可以对文件进行CRUD（Create、Read、Update以及Delete）常用文件操作。所有的文件分布式存储在集群的各台计算机之上。集群由一个主节点（NameNode）和许多个从节点（DataNode）构成。主节点主要负责系统元数据的文件管理，具体包括明确数据文件在各个从节点上的位置和信息、管理文件系统的整体命名空间以及调度用户接口对数据的操作。从节点主要负责该节点所在机器上的文件管理。HDFS主从节点间使用TCP/IP套接字通信，而客户端使用RPC远程过程调用进行通信。HDFS上的文件被切分成相同大小的数据块，每个数据块通过Hash函数计算后分散存储在不同的从节点之上。HDFS架构如图2-1所示。



图2-1 HDFS结构示意图

HDFS的主节点利用事务日志来更新记录文件元数据的变化，同时利用映像文件来存储系统的文件属性等信息。HDFS同时也是一个可靠的文件系统，它具有优秀的数据备份和异常故障恢复机制。为了处理异常情况，HDFS的所有数据块都具有副本文件，副本的数量默认为3，可以人工配置，数据存储在三个节点之上，其中两个在同一个机架，另一个在其它机架之上以应对突发的硬件故障问题。同时主节点周期性的对各个从节点进行心跳检测，根据收到的报告数据判断从节点是否宕机，如果宕机，则启动新的从节点，并在需要的时候从副本中恢复

数据至新的从节点。此外，HDFS的快照机制可以在文件受损时，回滚到过去数据正确的一个时间点。综上所述，HDFS是一个可分布式存储大量数据的文件系统，它拥有健壮可靠等特点。

### 2.2.2 MapReduce计算框架

MapReudce是位于HDFS之上的分布式计算框架。MapReduce是一种高度抽象的编码方式，它的核心思想是“分而治之”，即把原始对海量数据整体的计算任务切割并分发到不同机器进行并行计算，子任务完成后对各台机器的中间结果进行整合得到最终的计算结果。一般由Map和Reduce两个计算步骤完成。Map阶段负责把原始计算任务分解成多个子计算任务，Reduce阶段负责把各个子计算任务的计算结果综合起来得到最终结果。MapReduce计算框架将底层模块的计算、存储、数据备份、资源分配等问题很好地封装处理，编程人员只需要按照该编程模式完成业务逻辑编码即可。

MapReduce计算流程如图2-2所示。Map阶段将输入数据切分成固定大小的数据块，并将各个数据块根据实际业务需求分解成键值对key-value的格式。Hadoop为每个数据块创建一个Map任务，并对计算得到的中间过程结果按照键值对的键的大小进排序，将键值相同的中间结果分发到相同的Reduce任务进行下一步计算。Reduce阶段将从不同Map任务分发来的中间计算结果合并起来并对合并后的数据排序，接着调用编程人员根据实际需求完成的reduce方法对中间结果进行计算，并将最终计算结果写入HDFS。

MapReduce的调度过程主要由一个JobTracker和若干个TaskTracker节点控制完成。所有TaskTracker的调度和管理工作由JobTracker完成。JobTracker通常运行在主节点之上，TaskTracker必须在从节点上执行，因此从节点既是集群的存储节点也是计算节点。JobTracker将Map和Reduce阶段产生的计算任务分配给资源充足的TaskTracker后，由后者并行完成计算任务。除了调度和管理之外，JobTracker还对任务的运行状况进行监控，如果集群上的某个TaskTracker出现了故障，JobTracker会立刻将它的计算任务分发到集群上其它空闲的TaskTracker执行。



图2-2 Hadoop MapReduce计算流程

## 2.3 Python爬虫技术

随着大数据时代的到来，越来越多的数据出现在万维网之上，然而大多数网站只是将这部分数据通过前端网页展示出来，并没有提供获取数据的接口。为了得到这部分目标数据，网络爬虫技术随之诞生。爬虫技术是获取外部数据的主要途径之一，其目标是将互联网上的相关数据抓取到本地并组织成一个数据文件，编程工作者只需在此基础上解析数据文件得到目标数据即可。Python网络爬虫技术是业界使用较多的技术，由于Python对网络工具库Urllib的完美封装以及第三方丰富的网页解析工具如Beautiful Soup库等，此外针对大数据量所面临的并行化抓取，服务器代理等需求Python也能很好地解决。因此Pyhton在进行爬虫工作上具有得天独厚的优势。

Python网络爬虫基本工作流程是：

1. 设置好目标种子URL，种子URL是要抓取数据的网站，由这些种子URL

扩散可以得到其它URL；

2、将上述所有种子URL加入到待抓取队列中；

3、从待抓取队列中选择URL进行网络访问，获取目标数据对应的IP地址

并将相应的网页数据下载到本地文件系统。

4、分析已经抓取的URL链接中是否存在其他子链接，若有，则按照一定的

遍历策略对子链接进行抓取。常见的遍历策略包括深度优先遍历和广度优先遍历。

以本系统为例，大量与电影相关的信息，如电影情节、上映档期、剧组人员、社交舆情、传播力度、排片数等数据存在诸如时光网、豆瓣网、百度指数、猫眼电影、预告片等网站上，因此本系统需要使用Python爬虫技术抓取与电影相关的外部数据。

## 2.4Beautiful Soup

当前，Web上的数据大多以网页数据的形式展示给用户，使用爬虫技术获取的数据均是网页形式的文件。为了从HTML、XML等网页文件中解析出目标数据，我们需要相应的解析工具。Beautiful Soup[30]是一个用Pyhton实现的开源网页解析函数库。它提供了一些Python式的函数接口来处理网页文件，同时，Beautiful Soup解决了其它解析器存在的编码问题，它将输入文件统一转换成Unicode编码形式，并以UTF-8编码的形式输出结果文件。此外，Beautiful Soup还具有简单、轻便、高效等特点。综上，在本系统的开发过程中，使用Beautiful Soup对抓取的网页数据进行解析。

## 2.5特征选择技术

特征选择[31-33]是数据挖掘和机器学习等领域重要的研究方向之一，特征质量的高低对算法模型的最终效果有着显著的影响。特征选择指利用某些方法从原始数据集合中选出一个最优特征子集，在这个过程中利用某种特定的标准判断一个特征是否保留下来，而其它冗余和不相关的特征被过滤。特征选择流程如图2-3所示。电影票房预测系统中的特征选择是指，从Python爬虫技术抓取的与电影相关的数据中，提取出能使模型效果最优的特征子集。在抽取出的特征的基础上进行算法模型的学习训练，通过交叉验证选择最优模型，并对即将上映的影片的特征数据进行预测，得出电影预测票房值。

图2-3 特征选择流程示意图



特征选择算法主要包含Filter、Wrapper[34]以及Embedded三大类。其中Filter是指使用设定的评价准则来判断特征与目标值之间的相关性，评价标准分为距离度量、信息度量以及依懒性度量等几类。具体可以通过卡方检验、相关系数、信息增益以及互信息等进行判断。Wrapper是指通过一定的迭代搜索策略来判定一个特征是否加入到特征子集中，例如完全搜索策略在原始特征集合中寻找对于所涉及的可分性判断标准而言最优的特征子集合。但是这种策略的搜索范围巨大，为0(2n)，其中n为原始特征维数，因此该算法的时间复杂度较高。而启发式搜索策略则每次单独判断一个特征对当前特征子集的影响，从而决定是否将该特征添加到特征子集中，具体包括序列前向选择方法（SFS）、序列前后选择方法（SBS）、广义序列前向选择方法（GSFS）以及广义序列后向选择方法（GSBS）。此外，Wrapper方法还包括随机搜索策略的选择方法，这种方法与模拟退火算法类似，具有一定随机性。Embedded则侧重于使用算法模型自动选择特征的方式，例如采用正则化的方式进行特征选择，具体实例有在线性回归中加入L1范式之后正则化为Lasso回归[35]，加入L2范式之后正则化为岭回归。Lasso回归和岭回归都具有自动进行特征选择的功能。除正则化外，还有利用模型进行特征选择的方式，比如利用随机森林进行特征选择，近年来随着深度学习应用的大力推广，基于深度学习[36]的特征学习方式也逐渐被业界广泛使用。

除了上述三类特征选择方式外，当数据特征维度过高甚至出现维度灾难的时候，基于降维算法的特征选择方法往往具有显著的作用。主成分分析（PCA）是一种常见的特征降维算法，它是一种无监督学习算法。其主要原理是借助正交变换对高维特征集合进行降维。首先求出原始特征向量矩阵的协方差矩阵，然后计算出该协方差矩阵特征值所对应的特征向量，最后取特征值较大的若干个特征向量组成新的矩阵，并与原始特征向量矩阵相乘即可以得到降维后的特征向量矩阵，该算法对原有数据的信息损失较小。除了主成分分析之外，常见的特征降维算法还有奇异值分解（SVD）、线性判别分析（LDA）、局部性嵌入（LLE）等算法。

以上所描述的特征选择方法确实可以在高维度的原始特征数据的基础上选出所需的隐性特征。此外，为了最大程度的构造最优特征子集，在进行特征选择之后，结合电影票房预测的实际业务需求对特征子集进行特征值修正处理，生成新的特征集合，并在此基础上展开预测工作。

## 2.6预测算法

常见的机器学习算法根据实际问题不同可以分为回归算法、分类算法、聚类算法、关联关系挖掘等算法。其中回归算法主要用于解决连续值的预测问题，例如房价走势预测、金融趋势预测、流感性疾病预测等。常见的回归算法包括普通最小二乘法（Oridinary Least Square）、局部加权线性回归（Locally Weighted Linear Regression）[37]、岭回归（Ridge Regression）以及Lasso等算法。分类算法主要用于解决二分类或者多分类问题，例如垃圾邮件分类、用户是否购买商品预测、手写识别预测等。常见的分类算法包括逻辑回归（Logistic Regression）、支持向量机（Support Vector Machines，SVM）、AdaBoost、决策树（Decision Tree）、神经网络（Neural Networks）等，此外集成学习（Ensemble Learning）模型算法如随机森林（Random Forest）[38]以及随机梯度回归树（Gradient Boosting Regression Tree）[39]既可以用于分类问题也可以解决回归问题。聚类算法主要用于解决在数据集没有标签的情况下将一个数据集划分成子集的问题。常见的聚类算法有k-均值算法、k-medodis算法、BIRCH算法、DBSCAN等算法。而关联关系挖掘算法则主要用于多个组合数据之间的内在关系，比较著名的频繁项集挖掘的实际案例有啤酒与纸尿布、面包与牛奶。在关联关系算法中，Apriori算法、FP-growth以及基于Apriori改进的Eclat等算法是常见的关联关系算法。

票房预测是一个回归问题。本文介绍的票房预测系统在设计和开发的时候主要使用了线性回归中的普通最小二乘法、局部加权线性回归算法、随机森林算法以及随机梯度回归树算法。

1、普通最小二乘法

普通最小二乘法旨在从训练数据中学习一组回归系数，并根据该组回归系数建立一个拟合训练数据集的回归方程，在进行回归预测时只需要将目标特征值代入回归方程即可得到一个预测结果。

在进行模型训练时，算法的目标是使数据集上数据的真实值和预测值的平方误差和最小，也就是损失函数达到最小值。损失函数表达式为：

（2-1）

其中m为训练样本的个数，yi为样本的实际值，xi为该样本的特征向量，w为回归系数，xiTw为样本的预测值。为了使损失函数达到最小值，首先求其关于w的偏导数：

（2-2）

然后建立方程，令损失函数的偏导数为零，求得w如下：

（2-3）

2、局部加权线性回归

局部加权线性回归与普通最小二乘回归的不同之处在于，局部加权线性回归在对每一个样本进行预测的时候，对该样本点附近的样本集合中的每一个样本赋予一定的权值，并在这个样本子集上进行回归训练。结合普通最小二乘法的推导过程，局部加权线性回归法的损失函数为：

（2-4）

其中di是权值，其表达式为：

（2-5）

式2-5中k为权值变化速率，是一个可调参数，xi是数据集中的点，x是要预测的点。因此xi离x越近则di越大，xi离x越远则di越小。

令损失函数J(w)对w的偏导数为零，可得局部加权线性回归回归系数为：

（2-5）

3、随机森林

随机森林是一种集成模型。在进行模型训练的时候，算法首先对数据集进行装袋操作，即对数据集进行有放回的采样从而得到k个不同的数据集。然后在这k个不同的数据集上分别训练得到k个决策树模型。值得注意的是，在训练决策树模型的过程中，每一个节点随机选择m个属性作为该节点的候选属性，其中m小于可用属性数M，通常令m等于log(M+1)。由于上述进行的数据采样和特征采样两个随机步骤，随机森林一般不容易出现过拟合现象。

对于分类问题，随机森林算法综合考虑各个分类器的预测结果，数目最多的结果就是是算法的最终结果。

对于回归问题，随机森林算法的最终结果是训练得到的所有树的预测值的平均值。

4、随机梯度回归树

随机梯度回归树算法的主要思想是依次迭代训练多棵决策树，第一棵决策树模型拟合的数据是训练样本的目标值，此后每一棵树拟合的数据都是前面两轮迭代预测的值的差，也就是残差。最后将每一轮预测的值相加得到实际预测结果。具体实例如图2-4所示。



图2-4随机梯度回归树实例

在图2-4中有四个用户A、B、C、D，现在使用随机梯度回归树算法根据用户是否拥有信用卡和历史购物金额是否大于2000来预测用户的年龄。为了更形象地描述算法执行过程，假设最多有两棵树，且每棵树只拥有两个叶子节点。首先通过是否拥有信用卡这个特征训练得到第一棵数，得到两个叶子节点的均值分别为15和25，计算残差，例如A的残差等于真实值14减去预测值15，结果为-1。同理，B、C、D的残差分别为1、-1、1。用同样的方式以购物金额是否大于2000为特征对上述残差进行拟合，得到第二棵树。将两棵树的预测结果相加得到每个用户的实际年龄。

## 2.7 本章小结

本章依次介绍了Scikit-learn、Apache Hadoop、Python爬虫、Beatiiful Soup、特征选择算法和机器学习预测算法等相关技术，并对这些技术的原理和优点进行了分析，说明了在本系统开发和设计的过程中选用这些技术的原因。在后面章节的论述过程中，会详细阐述如何使用这些技术实现电影票房预测系统的各项功能。

# 第3章需求分析

需求是系统特征功能的一个描述。需求分析是软件开发过程中极为重要的一个过程，是后续进行软件系统的设计、编码、部署、测试等步骤的基础。需求分析工作完成质量的好坏直接影响到整个软件最终的效果。本章主要论述了电影票房预测系统的实际需求，对系统的功能性需求、非功能性需求和可行性进行了分析。

## 3.1功能性需求

### 3.1.1 需求概述

文本所设计的电影票房预测系统是一个智能化的、多功能的系统。从系统角度出发，系统的最终目的是提供对各部即将上映的电影的票房预测值。为了实现这个最终目的，系统需要有一套从数据抓取、数据预处理、数据存储、特征转换、特征抽取到最终算法预测的自动化流程。从用户角度出发，系统除了提供常见的登录功能外，还需提供电影信息查询、不同状态电影列表查看等功能。而本系统的侧重点主要是使用机器学习算法结合大数据工具对电影票房进行预测，这也是本人的在整个系统设计开发过程中的主要工作之一。

### 3.1.2需求建模

#### 3.1.2.1 用户需求分析

用户作为最终使用该系统的直接对象，对其而言最重要的功能就是查询电影的票房预测值和电影相关信息。本系统的用户主要是公司内部人员，因此不需要进行用户注册，直接使用员工内部信息即可登录系统。用户可以分为普通用户和管理员用户，普通用户是系统使用率最高的对象，具有登录、搜索电影、查询票房、检索不同状态电影列表等功能；管理员主要负责对系统进行相关管理。具体的用户与系统的交互如下：

1、用户、管理员登陆系统；

2、用户、管理员搜索查询电影票房；

3、用户、管理员查看电影基本信息；

4、用户、管理员查看不同状态电影列表；

5、管理员对系统进行管理。

以上交互信息说明了用户和管理员如何使用本系统，同时对系统支持的用户功能进行了相关描述。本系统的用例如图3-1所示。



图3-1 系统整体用例

#### 3.1.2.2 系统需求分析

电影票房预测系统的核心功能是利用相关数据，结合预测算法对指定电影的总票房进行预测。系统除了对用户提供上一小节讨论的交互功能外，在后台应具备网络数据自动抓取、数据预处理、数据存储、特征提取、票房预测等功能。这些功能构成了系统的主体框架。

1、数据抓取功能

数据抓取是整个系统的第一步，是整个票房预测系统的数据来源。系统采用Python爬虫技术对网络上与电影相关的资源进行抓取。具体需求描述如表3-1所示。

表3-1 数据抓取需求模板

|  |  |
| --- | --- |
| 相关需求 | Python爬虫进行数据抓取 |
| 目标 | 周期性抓取电影相关数据 |
| 前置条件 | 系统成功启动 |
| 执行成功结果 | 成功抓取目标数据并更新存储于HDFS中 |
| 执行失败结果 | 不更新电影相关数据并记录错误日志 |
| 主要角色 | 系统管理员 |
| 从属角色 | 电影票房预测系统 |
| 触发条件 | 每天凌晨1点更新一次 |
| 主要流程 | 1）管理员周期性启动整个系统； 2）根据预先设定好的网站URL地址、每隔一个固定时间  片进行数据抓取； 3）将成功抓取的数据存储于HDFS之中。 |
| 扩展 | 无 |

2、数据预处理功能

数据预处理是在数据已经成功抓取的基础上进行的处理，主要包括数据检测、数据过滤、目标数据抽取几个步骤。预处理完的数据存储于HDFS之上。具体需求描述如表3-2所示。

表3-2数据预处理需求模板

|  |  |
| --- | --- |
| 相关需求 | 数据预处理 |
| 目标 | 对网络爬虫抓取的网页文件进行预处理操作 |
| 前置条件 | 相关数据抓取成功 |
| 执行成功结果 | 数据预处理完成并保存在HDFS之上 |
| 执行失败结果 | 数据预处理失败并记录错误日志 |
| 主要角色 | 系统管理员 |

续表3-2数据预处理需求模板

|  |  |
| --- | --- |
| 从属角色 | 电影票房预测系统 |
| 触发条件 | 数据抓取成功后触发 |
| 主要流程 | 1）从HDFS中读取已经成功抓取的数据；  2）对上述数据进行数据检测、异常数据过滤以及网页目  标数据解析等操作； 3）将处理完的数据写入HDFS。 |
|
| 扩展 | 无 |

3、数据持久化功能

在整个系统的设计与开发过程中，由于系统前端展示需求，部分数据如电影基本信息、不同状态电影列表、最终预测出的电影票房值等数据需要持久化到Mysql数据库中以便前端页面快速访问。具体需求描述如表3-3所示。

表3-3数据持久化需求模板

|  |  |
| --- | --- |
| 相关需求 | 数据持久化 |
| 目标 | 将需要前端展示的数据持久化到Mysql数据库中 |
| 前置条件 | 数据预处理成功 |
| 执行成功结果 | 数据完成持久化并存储在Mysql中 |
| 执行失败结果 | 数据持久化异常并记录错误日志 |
| 主要角色 | 系统管理员 |
| 从属角色 | 电影票房预测系统 |
| 触发条件 | 数据预处理成功后触发 |
| 主要流程 | 1）预先设定好需要持久化的数据对象；  2）从HDFS将上述数据读取出并存储到Mysql数据库中。 |
| 扩展 | 无 |

4、特征抽取功能

特征抽取是整个票房预测流程中极为重要的一部，特征抽取质量的好坏关系到系统最终进行票房预测的准确率。本功能点结合2.5节中讨论的特征选择技术完成特征抽取工作。具体需求描述如表3-4所示。

表3-4特征抽取需求模板

|  |  |
| --- | --- |
| 相关需求 | 特征抽取 |
| 目标 | 根据预处理的数据进行特征抽取工作 |
| 前置条件 | 数据预处理成功 |
| 执行成功结果 | 特征抽取成功并将特征文件存储于HDFS中 |
| 执行失败结果 | 特征抽取异常并记录错误日志 |
| 主要角色 | 系统管理员 |

续表3-4特征抽取需求模板

|  |  |
| --- | --- |
| 从属角色 | 电影票房预测系统 |
| 触发条件 | 数据预处理成功后触发 |
| 主要流程 | 1）从HDFS读取预处理成功的数据；2）利用特征选择技术进行特征选择，并将处理完成的  特征文件写入HDFS。 |
| 扩展 | 无 |

5、票房预测功能

票房预测是系统最终对用户的直接反馈，用户最关心的也是票房预测值。本功能点在特征抽取完成后，利用算法模型对电影票房进行预测。具体需求描述如表3-5所示。

表3-5票房预测需求模板

|  |  |
| --- | --- |
| 相关需求 | 票房预测 |
| 目标 | 基于特征文件对票房进行最终预测 |
| 前置条件 | 特征抽取成功 |
| 执行成功结果 | 票房预测成功并将结果存储在Mysql数据库中 |
| 执行失败结果 | 票房预测出现异常并记录错误日志 |
| 主要角色 | 系统管理员 |
| 从属角色 | 电影票房预测系统 |
| 触发条件 | 特征抽取成功后触发 |
| 主要流程 | 1）从HDFS中读取电影特征文件并进行票房预测；  2）将成功预测的电影票房值写入Mysql数据库中。 |
| 扩展 | 支持多模型算法融合 |

## 3.2 非功能性需求

上一节详细讨论了系统的功能性需求，功能性需求描述了系统应具备的功能，而非功能性需求是系统潜在的性能需求。它们一起构成了系统的整体需求。本节主要对系统的非功能性需求进行讨论，着重阐述系统的准确性需求、同步性需求、健壮性需求和可扩展性需求。

### 3.2.1 准确性需求

票房预测系统的核心竞争力就是要对电影进行精准的票房预测，精准的票房预测可以在电影上映的早期给电影制片方和宣发方提供营销策略的数据支持，以便对电影进行更好的宣传和推广。对于回归预测问题的准确性度量可以用判定系数R2，区间预测准确率等，本系统采用相关系数作为评判标准。

### 3.2.2 同步性需求

由于电影和电影相关的数据是不断更新的，因此系统需要具有较强的同步性。系统的同步性需求主要表现在：第一，实时更新最新将要上映的影片列表信息；第二，对将上映的电影，实时抓取与电影相关的数据；第三，在影片上映前两周同步预测该影片的总票房值。

### 3.2.3健壮性需求

集群往往会出现宕机等风险，而系统也会出现例如数据抓取异常、数据存储失败等问题。面对这些突发状况，系统本身需要有一定的容错机制和应对措施，如数据备份、异常情况自动重启、错误日志报表等。

### 3.2.4 可扩展性需求

由于数据和业务的变化，系统需要应对不同的实际扩展性需求，例如针对数据对象的增加，系统需要提供从数据抓取到最终预测的可扩展能力优秀的接口，又如系统的底层存储设备需要有足够的扩展能力应对数据量增加的情况。

### 3.2.5 其它需求

除了上述讨论的四个非功能性需求外，为了使系统性能更加出色，还需具备易用性、稳定性、美观性和低延迟性。

## 3.3可能性分析

1、技术可行性

本系统基于Linux平台开发，系统的数据抓取使用Python爬虫技术实现。从上世纪九十年代开始，网络爬虫技术逐渐被业界广泛研究和使用，当前Python爬虫技术具备相当成熟的技术架构，是目前最受欢迎和运用最广泛的爬虫技术；系统使用Hadoop 的HDFS文件系统对数据进行存储，Hadoop相关技术已经发展了近十年之久，而HDFS作为Hadoop的核心存储系统，在工程技术层面已经十分成熟，并在百度、阿里巴巴、腾讯、亚马逊等许多国内外著名互联网公司大规模应用；系统的机器学习算法库使用Scikit-learn，它是一个免费开源的并且由一群志愿者共同维护的机器学习算法库，从2007年发行至今，已经迭代了众多版本，并被工业界和学术界广泛使用；系统Web开发使用Java SSH2框架开发，SSH2是Struts2、Spring和Hibernate的集成框架，该框架经过上十年的发展已经十分成熟；系统的底层数据库采用Mysql，它是一个跨平台的可扩展性好的数据库，同时它具有较强的网络适应性。与昂贵繁重的Oracle数据库相比，Mysql数据库具有免费开源、轻便、高效等诸多特点，因此深受广大用户喜爱。综上，本系统在设计和开发的过程中使用的技术都是被学术界和工业界证实的有效的技术，因此本系统的开发具有技术可行性。

2、资源可行性

本项目为腾讯影视大数据联合子项目，拥有足够的计算存储设备资源和经济资源。同时，本系统的开发团队由两名高级工程师和三名工程师组成，开发团队均是全职开发且具有丰富的工程经验，故本项目具有资源可行性。

## 3.4 本章小结

本章对票房预测系统进行了详细的需求分析，主要论述了系统的功能性需求和非功能性需求，并对系统的可行性进行了探讨。其中功能性需求是系统应具备的直接功能需求，本章结合系统UML用例图，分别从用户角度和系统角度进行了逐一分析。而非功能性需求是系统应具备的潜在性需求，本章主要从准确性、同步性、健壮性和可扩展性四个方面对其进行了分析。在本章的最后对系统的可行性进行了探究，结果表明系统具备可行性。在本文的剩余章节，将根据本章探讨的需求进行系统的设计和实现。

# 第4章概要设计

上一章对票房预测系统的进行了需求分析，着重讨论了系统的功能性需求和非功能性需求。在完成系统的需求分析之后，我们需要思考如下问题：第一，如何设计系统的整体架构；第二如何对系统进行模块划分；第三，各个模块的的数据结构和数据库表如何设计。本章围绕上述三个问题对系统进行概要设计，同时为后续的详细设计做好准备。

## 4.1系统整体架构

系统的整体架构设计应围绕它具备的功能展开。通过第3章的讨论，本系统分别从用户和系统后台角度具备不同的功能。从用户角度，用户应具备登录、查看电影信息、查看不同状态的电影列表以及查询电影票房等功能。从系统角度，系统应具备数据抓取、数据预处理、数据持久化、特征抽取以及票房预测等功能。

系统整体时序如图4-1所示，用户首先使用用户名和密码进行电影票房预测系统登录（1.1操作）。登录到系统首页后，进行电影搜索并对电影票房值进行查询（1.2操作）。系统收到用户请求后，立即对用户请求进行分解，判断用户请求类型，并将相应的票房预测结果通过前端页面对用户作出反馈（1.3操作）。前端展示页面获取的票房预测值通过后台离线计算模块得到（2.1操作）。在后台流程方面，管理员对系统进行管理和维护，管理员周期性重新启动系统（3.1操作）。系统每天凌晨1点开始进行相应电影数据抓取，并对抓取成功的数据进行数据预处理以及数据持久化（3.2操作）。系统对存储在HDFS上的经过预处理后的数据利用特征工程技术进行特征抽取（3.3操作）。在特征抽取任务完成之后，利用相关预测算法对电影票房值进行最终预测，由于预测产生的票房值需要实时反馈给用户，将这部分数据持久化到Mysql数据库中（3.4操作）。

根据上述时序分析，系统的整体架构如图4-2所示。系统的最底层提供数据抓取功能，对于外网数据使用Python爬虫技术进行抓取，对于内部已经拥有的数据可以直接使用。数据抓取的上一层提供数据预处理功能，主要对底层数据进行清洗、过滤和解析等操作，保证数据的纯净性、可用性和时效性。再上一层提

供了数据的存储以及并行计算功能，其中HDFS和Mysql提供对数据的存储功能，MapReduce和Scikit-learn提供了对数据的计算能力，具体来说MapReduce进行大规模数据的并行计算，Scikit-learn则提供了相应计算算法库。架构的最顶层利用特征选择算法对数据数据进行特征抽取，并使用机器学习算法进行最终的票房预测计算。对于最终预测的票房值，该架构提供网站数据访问接口，方便用户和管理员进行电影票房查询和其它相关操作。



图4-1 票房预测系统整体时序



图4-2票房预测系统整体架构

## 4.2 系统功能模块划分

根据上一节对系统的整体架构和整体时序的探讨，可将系统划分为用户登录模块、数据查询模块、数据抓取模块、数据预处理模块、特征抽取模块和预测算法模块。下面将对各个模块进行具体描述。

1、用户登录模块

该模块是用户和管理员进行电影票房预测系统登录的接口。用户在成功登录本系统后可以根据个人需求访问系统。用户和管理员的个人信息从公司员工信息总表中导出部分字段。用户具有用户ID、用户名、密码、性别、头像、手机号、电子邮件等个人属性，具体数据库设计如表4-1所示。

表4-1 用户属性表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 是否为空 | 是否主键 | 字段说明 |
| user\_id | int | N | Y | 用户ID |
| user\_name | varchar(20) | Y | N | 用户名 |
| user\_password | varchar(20) | N | N | 密码 |
| user\_gender | int | Y | N | 性别 |
| user\_photo | text | Y | N | 头像 |
| user\_phone | varchar(20) | Y | N | 手机号 |
| user\_email | varchar(20) | Y | N | 电子邮件 |

表4-2管理员属性表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 是否为空 | 是否主键 | 字段说明 |
| admin \_id | int | N | Y | 用户ID |
| admin \_name | varchar(20) | Y | N | 用户名 |
| admin \_password | varchar(20) | N | N | 密码 |
| admin \_gender | int | Y | N | 性别 |
| admin \_photo | text | Y | N | 头像 |
| admin \_phone | varchar(20) | Y | N | 手机号 |
| admin \_email | varchar(20) | Y | N | 电子邮件 |
| admin\_level | int | N | N | 级别 |

2、数据查询模块

该模块主要提供用户对电影相关数据查询的功能。当用户成功登录票房预测系统之后，用户具有搜索查询影片票房值、查询影片基本信息、查询不同状态电影列表等功能。电影对象具有电影名、电影类型、演员列表、导演、上映日期、电影简介等基本属性，具体的数据库表设计如表4-3所示。

表4-3 电影基本属性表

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 字段名 | 字段类型 | 是否为空 | 是否主键 | 字段说明 |
| movie\_name | varchar(50) | N | Y | 电影名称 |
| movie\_type | varchar(100) | Y | N | 电影类型 |
| release\_date | varchar(20) | Y | N | 首映日期 |
| movie\_actors | varchar(200) | Y | N | 演员列表 |
| movie\_director | varchar(50) | Y | N | 导演名称 |
| movie\_alias | varchar(200) | Y | N | 电影别名 |
| movie\_nation | varchar(50) | Y | N | 拍摄国家 |
| movie\_time | varchar(50) | Y | N | 电影时长 |
| movie\_introduction | text | Y | N | 电影简介 |
| company | text | Y | N | 制片公司 |
| screenwriter | text | Y | N | 电影编剧 |
| releaseCountry | varchar(50) | Y | N | 上映国家 |
| boxOfficeShowDays | text | Y | N | 票房累计天数 |
| boxOfficeTotal | text | Y | N | 总票房 |
| boxOfficePredict | text | Y | N | 预测票房 |
| movieNumberSameTime | int | Y | N | 同档期电影数 |
| scheduleNum | int | Y | N | 首映排片数 |
| scheduleMovieList | text | Y | N | 同期电影列表 |
| movie\_search\_index | int | Y | N | 搜索热度 |
| movie\_social\_index | int | Y | N | 社交热度 |
| movie\_sent\_index | int | Y | N | 舆情指数 |

查看不同状态的电影列表为用户提供了一个整体的把握全局电影的功能。电影列表是不同状态的电影的一个集合，电影的状态包括即将上映、正在上映、已下映以及全部电影，其中全部电影是其它三个状态的电影的集合。电影状态规范如表4-4所示。

表4-4电影列表状态规范

|  |  |  |
| --- | --- | --- |
| 状态字段 | 状态说明 | 其它说明 |
| 1 | 即将上映的电影 | 无 |
| 2 | 正在上映的电影 | 无 |
| 3 | 已下映的电影 | 无 |
| 4 | 全部电影 | 全部电影是所有状态的电影的集合 |

3、数据抓取模块

该模块的主要功能是使用Python爬虫技术对互联网上与电影相关数据进行抓取。该功能是进行票房预测的第一步，也是较为重要的一个步骤。在互联网迅速普及的今天，数据中所蕴含的价值是无穷的。面对如此巨大的数据，有两个问题值得思考：第一，应该抓取哪些类型的数据；第二，如何有效的抓取数据。针对第一个问题，经过大量实际调研工作后，我们应该更多的抓取与电影相关的数据，具体分为电影基本信息、电影排片信息、电影社交热度信息、电影舆情走势信息等几类数据，而这些数据大多存在于电影相关垂直网站，如猫眼、豆瓣、时光网等，同时也存在于各大互联网社交平台之上，如百度指数、微博指数、各大网站的网友评论等，具体的说明如表4-5所示。而上述第二个问题是一个工程化问题，在面对大数据抓取任务时，较为有效的抓取方式为分布式并行抓取，在多台机器上采用多进程的方式抓取能较快的收集目标数据。

表4-5电影抓取数据类型说明

|  |  |  |
| --- | --- | --- |
| 数据种类 | 数据来源 | 数据价值 |
| 电影基本信息 | 时光网、豆瓣电影、百度百科 | 电影的基本信息是对其基本面的一个详细描述，从直观上展现了电影的直接影响力 |
| 电影社交热度 | 朋友圈、百度指数、微博指数、QQ空间 | 电影社交热度反应了电影在互联网上的知名程度，社交热度越高，知名度越大 |
| 电影相关舆情 | 朋友圈、QQ空间、新浪微博、百度贴吧 | 电影舆情指数反应了网友对电影的看好程度，正面舆情越多，网友越看好该电影，反之，负面舆情越多则电影越不被看好 |
| 电影预告片  热度 | 预告片网、微博电影 | 电影预告片被点击、播放、评论的次数与电影观影成正相关关系，预告片点击、播放、评论次数越多，则电影会拥有更多的观影人数 |
| 电影排片数 | 猫眼电影 | 各大院线对电影的期望从侧面反映了实体院线对电影的看好程度，电影排片数越多则电影越被实体院线看好 |
| 电影竞品 | 预告片网 | 影片竞品数反映了同一档期内其它影片的竞争力度，竞品数越多则影片面对的票房竞争越大 |

4、数据预处理模块

该模块主要对数据抓取模块爬取的电影相关数据进行预处理操作。由于使用Python爬虫技术获取的数据是网页数据形式，不具备直接可用性，因此模块首先需要对网页数据文件进行解析。对网页形式的目标数据的解析可以使用Beautiful Soup库与正则解析相结合的处理方式。从网页文件中解析出目标数据后需要对数据进行清洗、过滤、转换等操作，最后将数据写入存储系统。

电影的基本信息如电影名、电影类型、导演名以及演员列表等可以稳定地从相应电影网站获取且数据量比较小，因此这部分信息可以选择存储在Mysql数据库上，而电影的其他信息，如各大电影垂直网站的影片评分信息、网友社交评论信息、百度指数、新浪微博社交数据等由于数据种类的繁杂且数据量较大，因此将这部分数据存储在Hadoop集群上。

5、特征抽取模块

该模块的功能是从底层数据中抽取出电影票房的相关特征供算法模型进行计算。该模块主要包含特征计算和数值修正两个步骤。特征抽取模块架构如图4-3所示。



图4-3特征抽取模块架构

特征计算步骤主要完特征的量化计算工作，分析系统需求和实际拥有的数据，应对以下类型的特征进行量化：

1）电影相关人物影响力

2）电影上映档期

3）电影社交热度

4）电影舆情指数

5）电影搜索装换率

6）电影排片数

7）电影预告片热度

8）电影类型

9）电影同档期竞品

在完成特征量和抽取任务之后，还需对特征数据进行数值归一化、连续特征离散化等数值修正处理，从而减少特征数据异常，提升特征可用性。

6、预测算法模块

票房预测系统主要涉及到的预测算法包括普通最小二乘法、局部加权线性回归算法、随机森林算法以及随机梯度决策树算法。该模块的主要功能是使用上述四种算法分别训练得到不同的模型，然后利用交叉验证选出最优模型和最优模型对应的算法参数，最后对这四个最优模型进行算法融合得到最终的电影票房预测结果。

## 4.3本章小结

本章主要讨论了票房预测系统的概要设计，首先给出了系统的整体时序和整体架构，然后对系统进行模块划分，详细分析了系统应该具有的子模块和每个模块的具体功能，具体来说，系统分为用户登陆模块、数据查询模块、数据抓取模块、数据预处理模块、特征抽取模块和预测算法模块。此外，本章还对讨论了各个模块的数据结构、数据库表设计以及相关规范，每个模块之间满足高内聚低耦合的设计思想。

# 第5章详细设计与实现

上一章讨论了系统的概要设计，着重分析了系统的整体架构和模块划分。详细设计是对系统概要设计的进一步精化设计，明确每一个子模块的内部组件、算法实现和逻辑流程。本章在概要设计的基础上对系统进行详细设计，并对实现细节进行说明，包括重点算法分析以及如何使用第二章介绍的相关技术和工具进行编码实现。

## 5.1用户登录模块详细设计与实现

### 5.1.1 用户登录整体流程

普通用户和管理员需要输入正确的账号和密码才能登录电影票房预测系统，若账号或者密码输入错误则需要重新登录直到输入正确为止。用户登录流程如图5-1所示。



图5-1用户登录注册流程图

### 5.1.2 静态类设计和核心接口实现

根据用户登录流程图，结合第四章概要设计中的表4-1和表4-2所定义的用户和管理员基本属性，系统用户登录模块相关类设计如图5-2所示。这里具有三个主要的类，分别是用户类、管理员类和登录类。其中User类和Admin类分别描述用户和管理员的基本属性和方法。LoginAction是系统的登录类，该类主要提供用户和管理员进行系统登录的接口。下面将逐一介绍该模块的核心接口设计与相关实现。



图5-2 用户登录模块类图

LoginAction类提供了用户和管理员的登录方法。用户使用userLoginCheck方法进行系统登录，输入账号和密码后调用业务逻辑层userService类的checkUser方法检测用户输入信息是否正确，如果正确则返回该用户的后台信息不为空，同时切换到电影票房预测系统的主页，如果错误则用户需要重新输入正确的账号和密码才能登录。管理员的的登录流程与普通用户类似，主要调用adminLoginCheck和checkAdmin两个核心方法进行系统登录。userLoginCheck和checkUser方法的实现如代码片段5-1和5-2所示。

|  |
| --- |
| public String userLoginCheck(){  String page = "fail";  User user = new User();  user = userService.checkUser(userName, userPassword);  if(user != null)  {  page = "success";  HttpServletRequest request = ServletActionContext.getRequest();  HttpSession session = request.getSession();  if(session.getAttribute("loginUser") != null)  {  session.removeAttribute("loginUser");  }  session.setAttribute("loginUser", user);  }  return page;  } |

代码片段5-1 userLoginCheck方法实现

|  |
| --- |
| public User checkUser(String userName,String userPassword) {  Session session = sessionFactory.getCurrentSession();  String hql = "from User as user where user.userName = '"+userName+"' and user.userPassword = '"+userPassword+"'";  User user = null;  List<User> userList = session.createQuery(hql).list();  if(userList != null &&userList.size() > 0) { user = userList.get(0); }  return user;  } |

代码片段5-2checkUser方法实现

## 5.2 电影数据查询模块

电影票房预测系统底层算法预测模块预测出的电影票房值需要有一个展示接口将系统的核心计算结果展示出来，同时票房预测值也是系统用户最关注的数据，因此本模块需要提供用于对票房预测值查询的功能。此外，系统的用户大多是与电影相关的从业者，为了将本系统打造为一个更全面的电影数据展示平台，方便用户使用，本系统还需要提供电影基本信息的查询和不同状态电影列表查询的接口。以上讨论的数据查询功能是本模块的主要功能。

### 5.2.1 静态类设计和核心接口实现

在4.2节中我们讨论了电影具备的基础属性，并给出了相应数据库表设计。电影类作为一个独立的类，应具有提供各种属性的查询方法。除了电影类以外，还应该设计一个电影列表类，该类的主要功能是提供不同状态电影列表的查询方法。

1、不同状态电影列表查询

MovieSet类提供了用户对即将上映、正在上映、已经下映和全部电影这四种不同状态的电影列表的的获取方法，这四个方法分别是getComingMovieList、getShowingMovieList、getOffLineMovieList和getAllMovieList。MovieSet类的定义如代码片段5-3所示。

|  |
| --- |
| public class MovieSet{  private List<String> comingMovieList;  private List<String> showingMovieList;  private List<String> offLineMovieList;  private List<String> allMovieList;  ...  public List<String> getComingMovieList();  public List<String> getShowingMovieList();  public List<String> getOffLineMovieList();  public List<String> getAllMovieList();  ...  } |

代码片段5-3 MovieSet类定义

2、电影票房及基本信息查询

电影票房预测值和电影其它信息均作为电影类Movie的基本属性。这些属性信息持久化存储在后台Mysql数据库中。Movie类提供了对这些数据的访问方法，其中getMovieBoxOffice方法用来查询电影票房预测值，getMovieBasicInfo方法对其它基本方法进行封装，可以用来获取电影基本信息。Movie类的定义如代码片段5-4所示。

|  |
| --- |
| public class Movie{  private String movieName;  private String movieType;  private String releaseDate;  private String movieBoxOffice;  private String movieDirector;  private String movieActors;  private String movieTime;  private String movieIntroduction;  ...  public String getMovieBoxOffice() throws Exception;  public String getMovieBasicInfo()throws Exception;  public String getMovieName();  public String getMovieType();  public String getMovieReleaseDate();  public String getMovieDirector();  public String getMovieActors();  ...  } |

码片段5-4 Movie类定义

## 5.3数据抓取模块

进行数据抓取时，首先需要确定抓取数据对象的种类、类型以及数据来源，在上一章中，我们对这个问题进行了详细分析，讨论结果如表4-5所示。抓取的数据对象主要分为电影基本信息、电影社交热度、电影舆情信息、电影预告片热度、电影同期竞品数据和电影排片数。以上这些数据均存储在各大互联网站以及电影相关垂直网站之中。其中朋友圈和QQ空间数据属于公司内部数据、这部分数据不用抓取可以直接获取。而其它站点的数据存储于外部互联网站，这部分数据必须使用爬虫技术进行抓取。

对于外网数据的抓取我们使用Python爬虫技术编码实现抓取功能，在设计与实现爬虫功能时，会遇到如下问题：

1、使用同一机器对某些站点访问频率过高或者访问次数过多时，这些站点

的反爬虫机制认为该IP地址的机器存在作弊情况，并屏蔽该IP地址继续对站点的访问；

2、由于本系统的目标数据量较大，并且需要持续抓取。如果只是用单台服

务器进行抓取，则抓取效率过低。

针对上述在数据抓取的过程中可能会遇到的代表性问题，本文使用基于代理技术的分布式爬虫方式，其架构如图5-3所示。系统首先确定需要抓取数据站点的种子URL链接，并将这些链接保存到多个文件之中。然后，系统将这些种子URL文件分发到不同机器之上，并结合代理技术并行爬取数据。最后，将抓取成功的数据写入HDFS之中进行存储。



图5-3 系统爬虫架构

Python爬虫主要用urllib2库进行数据抓取。首先利用urllib2的ProxyHandler方法设置一个代理，然后调用build\_opener和install\_opener方法建立代理连接，接着使用urlopen方法对目标站点进行访问的同时调用read方法将网页内容保存至内存变量，最后使用自定义的write2HDFS方法将该内存变量中保存的临时网页数据写入HDFS文件系统。数据抓取功能的核心实现如代码片段5-5所示。

|  |
| --- |
| def getData(URL):  proxyPot = urllib2.ProxyHandler({'http': proxyURL})  opener = urllib2.build\_opener(proxyPot, urllib2.HTTPHandler)  urllib2.install\_opener(opener)  request = urllib2.urlopen(URL)  webFile = request.read()  //写入HDFS文件系统  write2HDFS(webFile) |

代码片段5-5 数据爬虫实现

## 5.4数据预处理模块

### 5.4.1 数据解析

数据解析是爬虫模块成功抓取数据后进行的第一步处理。爬虫模块抓取的数据都是以网页HTML或者XML的格式存储在HDFS文件系统上，数据解析的主要任务就是从HDFS中将网页数据读取出来，并解析出其中的关键目标数据。网页解析使用Beautiful Soup函数库。网页数据以树状的形式存储，并使用不同的标签标识每一部分数据，我们需要做的就是要到目标标签，然后提取该标签中的数据。以解析网页数据中的电影名为例，使用Beautiful Soup进行网页文件解析，首先需要生成一个新的soup对象用来参与后续处理，然后使用findAll函数找到目标标签中包含show-movtip所对应的数据项，最后可以直接从该数据项中解析出目标数据电影名称并调用write函数将电影名称列表写入存储系统。使用Beautiful Soup解析数据的核心实现如代码片段5-6所示。

|  |
| --- |
| def parseWebData(file):  movieList = []  soup = BeautifulSoup(file)  data = soup.findAll('li',{'class': 'show-movtip'})  for mdata in data:  mname = mdata['data-name']  movieList.append(mname)  write(movieList) |

代码片段5-6 解析网页数据核心实现

### 5.4.2数据存储

从网页文件解析出来的目标数据以及内网提供的与电影相关的数据根据业务需求不同，应分为两种不同的存储方式：第一，对于需要实时前台展示且数据量较小的数据应存储在Mysql数据库上，这类数据包括电影基本信息和电影预测票房值；第二，对于数据量大且计算任务较为繁重的数据应存储在HDFS文件系统之中，这样既可以分布式存储又方便上层计算任务并行计算。这类数据包括社交舆情数据、网络热度数据、百度微博指数、预告片数据、竞品数据等。在存储和读取数据时可以使用HDFS命令进行操作，具体命令如表5-1所示。

表5-1常用HDFS文件系统命令

|  |  |  |
| --- | --- | --- |
| 具体命令 | 使用方法 | 功能说明 |
| cat | hadoop fs -cat URI | 输出指定路径的文件到标准流 |
| mkdir | hadoop fs -mkdir <paths> | 创建指定路径目录 |
| ls | hadoop fs -ls <args> | 返回集群上文件和路径信息 |
| get | hadoop fs -get <src><localdst> | 复制集群上的文件到本地文件 |
| put | hadoop fs -put <localsrc> ... <dst> | 上传本地文件到集群指定路径 |
| rm | hadoop fs -rm URI [URI …] | 删除集群上指定路径文件 |
| cp | hadoop fs -cp URI [URI …] <dest> | 对集群上的文件进行拷贝操作 |
| mv | hadoop fs -mv URI [URI …] <dest> | 对集群上的文件进行移动操作 |

### 5.4.3 数据过滤

对于某些异常电影数据需要进行过滤处理。例如电影《人间小团圆》由于其演员杜汶泽的个人原因，导致一部历时五年打造的电影的最终票房极其惨淡；又如由某些知名度并不高的导演和演员拍摄的电影最终不会在各大院线上映，这些电影的票房值几乎为零。对于上述这些类型的电影，它们的数据是存在问题的，因此在进行数据处理的时候将这些电影的数据过滤掉。

## 5.5 特征抽取模块

特征抽取模块是整个票房预测系统中较核心的部分之一。本模块在前述模块的基础上对系统整体特征进行量化计算。基于电影相关数据进行特征抽取主要包括离线特征抽取和特征数值修正两步。其中离线特征计算步骤的主要任务是对各类数据进行量化分析，并利用相关算法计算出最终特征值；特征数值修正步骤主要对已经计算出的特征值进行归一化、连续特征离散化等处理得到新的特征值。

### 5.5.1 离线特征计算

1、人物影响力

人物影响力包括演员影响力、导演影响力和编剧影响力。人物影响力是指电

影参与者对电影票房的号召力，可以根据该人物对历史票房的贡献度来计算。

在计算演员影响力的时候，由于同一部电影中不同演员对票房的贡献度不同，大牌明星演员对电影票房的贡献度往往高于一般演员，因此在计算演员影响力的时候需要根据该电影演员列表中演员的出现顺序进行加权，演员列表中越靠前的演员权值越大。演员影响力的具体计算公式为：

（5-1）

其中m代表演员i参演的历史影片数，rk代表演员i在其参演的第k部影片中的演员权重，boxofficek表示该第k部电影的历史票房值，t为可调参数。

同理可得，导演影响力计算公式为：

（5-2）

编剧影响力计算公式为：

（5-3）

2、上映档期

电影档期指电影上映的时间段，选择合适的档期往往有利于促进电影票房的增长。例如大多数爱情片选择在七夕或者情人节档期上映；贺岁片大多在农历春节期间上映；动画电影往往选择在学生课程压力较小的寒假和暑假上映。

在处理电影上映档期的时候，首先建立如表5-2所示的档期分类表。然后根据每部电影的上映日期具体时间对应不同的档期，并将该档期特征值设置为1，其它档期特征值设置为0。

表5-2档期分类

|  |  |
| --- | --- |
| 序号 | 档期名称 |
| 1 | 元旦 |

续表5-2档期分类

|  |  |
| --- | --- |
| 2 | 春节 |
| 3 | 清明节 |
| 4 | 劳动节 |
| 5 | 儿童节 |
| 6 | 端午节 |
| 7 | 中秋节 |
| 8 | 七夕节 |
| 9 | 国庆节 |
| 10 | 暑假 |
| 11 | 寒假 |
| 12 | 圣诞节 |
| 13 | 情人节 |

3、社交热度

电影的社交热度反映了该电影在社交网络上的传播度，社交热度越高则电影的传播度越广。电影每一天的社交热度使用当天的百度指数、微博指数、朋友圈指数和QQ空间指数求平均值得出，计算公式为：

（5-4）

其中baiduIndex、weiboIndex、wxIndex和qqIndex分别表示该电影当天的百度指数、微博指数、朋友圈指数和QQ空间指数的值。

由于社交热度存在时效性，距离当前日期较近的社交指数往往更能反映电影近期在社交网络上的传播度。因此在计算电影社交热度特征时需要在每日社交热度的基础上乘以一个惩罚因子。惩罚因子k的表达式为：

（5-5）

其中x为时间距离，时间越近惩罚因子k值越小，时间越远惩罚因子k值越大。w为可以调参的衰减因子，默认设置为1。选择最近90天的社交热度进行特征计算，计算公式为：

（5-6）

4、舆情指数

电影的舆情指数反应了人们对电影的支持程度，正面舆情越多则电影拥有更多的正面支持。电影每一天的舆情指数分为正面指数和负面指数，正负面舆情指数使用当天的朋友圈舆情指数、QQ空间舆情指数、百度贴吧舆情指数和微博舆情指数求平均值得出，每日正面舆情指数的计算公式为：

（5-7）

其中posSent1到posSent4依次表示朋友圈、QQ空间、百度贴吧和微博的正面舆情指数。

同理可得，每日负面舆情指数的表达式为：

（5-8）

其中negSent1到negSent4依次表示朋友圈、QQ空间、百度贴吧和微博的负面舆情指数。

由于舆情指数存在与社交热度同样的时效性问题，在计算最终的舆情特征时需要加入式5-5中的惩罚因子k。选择最近90天的舆情指数，加入惩罚因子后正负面舆情特征计算公式为：

（5-9）

（5-10）

其中posSentIndex表示正面舆情特征，negSentIndex表示负面舆情特征，t为可调参数。

5、预告片热度

预告片热度从侧面反映了电影受欢迎程度，同时预告片热度往往和电影票房存在正相关的关系。预告片热度计算公式为：

（5-11）

其中playNum、clickNum和commentNum分别是是预告片的播放数、点击数和评论数，预告片热度为这三者之和，t为可调整参数。

6、电影类型

电影类型作为每部影片的基本属性之一，对影片总票房具有一定的影响力。经统计，动作片、科幻片、喜剧等类型影片的票房值往往高于恐怖、纪录片等类型的影片。

对电影类型进行特征抽取时，先建立如表5-3所示的影片类型分类表。然后依据每部电影所对应的类型将对应值置为1其它类型置为0。

表5-3影片类型

|  |  |
| --- | --- |
| 序号 | 档期名称 |
| 1 | 喜剧 |
| 2 | 冒险 |
| 3 | 科幻片 |
| 4 | 惊悚悬疑 |
| 5 | 恐怖片 |
| 6 | 记录片 |
| 7 | 战争片 |
| 8 | 动作片 |
| 9 | 家庭片 |
| 10 | 爱情片 |
| 11 | 动画电影 |
| 12 | 音乐电影 |

7、电影搜索转换率

电影的搜索转换率指电影相关资源的点击率与电影关键字搜索次数的比率。电影每日搜索转换率的表达式为：

（5-12）

其中clickNum是电影相关链接的点击次数，searchNum是电影关键字搜索次数。在计算最终电影的转化率特征时需要考虑时效性的影响，因此加入式5-5中的惩罚因子k。选择最近90天的搜索数据，加入惩罚因子后电影搜索装换率特征的计算公式为：

（5-13）

8、电影排片数

电影排片数是指电影在各大院线当日的播放场次，该值由各大院线设定。经统计，影片排片数与最终影片票房往往存在一定的正相关的关系。

9、同档期竞品数

在电影上映期间，与该电影处于同一档期的同类电影会与该电影形成竞争关系，同时在一定程度上对该电影的最终票房产生影响。同档期电影是指上映时间在目标电影上映期间的电影，同档期竞品数是这些电影的总和。

### 5.5.2 特征数值修正

上一节讨论了电影特征抽取过程，对每一类特征的计算方法和意义给出了详细说明。但是这样构建的数据特征会存在一些数据异常，例如某部电影的社交热度可能存在异常值，这些异常值较正常值可能极大或者极小。又如，由于不同年份的年度整体票房存在较大差异，票房数值不在同一数量级会影响最终预测效果。针对上述问题，需要对特征数值进行相关修正。

#### 5.5.2.1特征归一化处理

对特征进行归一化处理的目的是消除特征数据量级上的差异性，使得数据特征属于同一数量级。这样处理的不仅消除了了异常值的影响，同时也能促进模型在训练的过程中更快的收敛到最佳状态。常用的归一化方法有最大最小归一化和z分数归一化。最大最小归一化的计算公式为：

（5-14）

其中max和min分别代表该维度特征值中的最大和最小值。

z分数归一化的计算公式为：

（5-15）

其中μ和σ分别表示该维度特征的均值和标准差。

对于此前讨论的特征类型，需要做归一化处理的特征有人物影响力、社交热度、舆情指数、电影排片数以及预告片热度。

#### 5.5.2.2连续特征离散化

连续特征离散化是指将连续型特征值离散化为一系列离散的值如0，1。例如，根据竞品数的大小将同档期竞品数分为十个等级，每个等级对应一个竞品数，当电影的同档期竞品数落在某个等级上，将该竞品数所对应的等级的值置为1，其它等级值置为0。

连续特征离散化具有诸多优点：第一，离散特征使得向量内积计算更快，计算结果也易与存储；第二，离散化的特征对异常数据具有极强的容错性；第三，特征离散化降低了模型过拟合的风险。电影特征中需要进行离散化处理的特征有同档期竞品数。

#### 5.5.2.3电影票房大盘同步化

2009年至2015年的电影票房大盘走势如图5-4所示。票房的增长曲线近似拟合一个多项式函数y=9.3263x2-17.262x+85.033，可见电影票房整体大盘的趋势是非线性的。因此在进行票房预测时需要根据大盘的增长率对票房做归一化处理，相应计算公式如下：

（5-16）

其中ratio为后一年大盘票房值较前一年的增长率。boxOffice为电影原始票房值，boxOfficeadjust为调整后的电影票房值。

图5-4 电影票房大盘走势

## 5.6 算法预测模块

票房预测算法主要包括普通最小二乘法、局部加权线性回归算法、随机森林算法和随机梯度回归树算法。第2章对这些算法的思想和原理进行了描述，本节不再赘述。本模块对特征抽取模块计算的电影特征进行划分，分为训练集和测试集数据。然后利用上述算法分别训练得到四个预测模型，其中普通最小二乘法和局部加权线性回归由本人编码实现，随机森林和随机梯度回归树使用Scikit-learn的开源算法库。最后对这四个模型进行算法融合得到电影票房的最终预测值。

### 5.6.1数据集划分

为了对算法模型的进行学习训练和效果评估，需要将原始特征数据划分为两部分，分别是训练集和测试集。这两类数据集的作用如表5-4所示。

表5-4 数据集划分

|  |  |
| --- | --- |
| 数据集 | 作用 |
| 训练集 | 进行算法学习训练，得到模型M |
| 测试集 | 将测试集特征数据带入模型M，对模型效果进行最终评估 |

### 5.6.2 普通最小二乘法

普通最小二乘法使用Python语言实现的核心代码如代码片段5-7所示。其中xArray和yArray分别是特征向量和目标值向量，这里均以数组的形式存储。首先将这两个数组转化为矩阵xMatrix和yMatrix，接着计算xMatrix的转置矩阵和xMatrix自身的乘积xTx，最后计算得到回归系数向量wVector。

|  |
| --- |
| def ols(xArray,yArray):  xMatrix = mat(xArray); yMatrix = mat(yArray).T  xTx = xMatrix.T\*xMatrix  if not linalg.det(xTx) == 0.0:  wVector = xTx.I \* (xMatrix.T\*yArray)  return wVector |

代码片段5-7 普通最小二乘法核心实现

### 5.6.3 局部加权线性回归

局部加权线性回归算法使用Python语言实现的核心代码如代码片段5-8所示。由于局部加权线性回归算法是一种非参数学习算法，因此针对预测集合中的每一个样本target，算法都需要重新学习一组新的权重系数和回归系数。与普通最小二乘法类似，算法首先将特征向量和目标值向量转化为矩阵xMatrix和yMatrix，接着创建每个点的权重矩阵weight并初始化对角元素值为1，然后根据数据集中每一个点与目标测试点的距离计算并更新权重矩阵，这里权值变化速率k的值缺省为1.0，最后根据权重矩阵计算得到回归系数向量wVector。

|  |
| --- |
| def lwlr(target,xArray,yArray,k=1.0):  xMatrix = mat(xArray); yMatrix = mat(yArray).T  m = shape(xMatrix)[0]  weight = mat(eye((m)))  for j in range(m):  diffMatrix = target - xMatrix[j,:]  weight[j,j] = exp(diffMatrix\*diffMatrix.T/(-2.0\*k\*\*2))  xTx = xMatrix.T \* (weight \* xMatrix)  if not linalg.det(xTx) == 0.0:  wVector = xTx.I \* (xMatrix.T \* (weight \* yMatrix))  return wVector |

代码片段5-8 局部加权线性回归算法核心实现

### 5.6.4 随机梯度回归树

随机梯度回归树算法使用Scikit-learn算法库中提供的算法接口，使用该算法进行预测的核心代码如代码片段5-9所示。其中x\_train、y\_train、x\_test分别代表训练集特征数据、训练集目标值、测试集特征数据。算法首先调用GradientBoostingRegressor方法初始化一个预测器对象est，然后调用fit方法训练得到最终模型predictor，最后调用predict方法对测试集特征数据进行预测，得到最终预测结果result。

|  |
| --- |
| def gbrt(x\_train, y\_train, x\_test):  est = GradientBoostingRegressor(learning\_rate=0.01, n\_estimators=100 ,  max\_depth=5, loss='ls')  predictor = est.fit(x\_train, y\_train)  result = predictor.predict(x\_test))  return result |

代码片段5-9 随机梯度回归树使用示例

在初始化预测器对象时，Scikit-learn提供了一系列参数，常用参数如表5-5所示。

表5-5 随机梯度回归树常用参数说明

|  |  |
| --- | --- |
| 参数 | 说明 |
| loss | 算法选用的损失函数，有ls、lad、huber、quantile四种类型 |
| learning\_rate | 学习速率 |

续表5-5 随机梯度回归树常用参数说明

|  |  |
| --- | --- |
| n\_estimators | 迭代次数 |
| max\_depth | 每棵子树的最大高度 |
| max\_leaf\_nodes | 叶子节点最大个数 |
| min\_samples\_split | 最小分裂节点样本数 |

### 5.6.5 随机森林

随机森林算法也使用Scikit-learn算法库中提供的接口，使用该算法进行预测的核心代码如代码片段5-10所示。与随机梯度回归树不同的是，算法一开始使用RandomForestRegressor方法初始化一个预测期对象est，算法的其它步骤与使用随机梯度回归树算法类似，最终的result为预测结果。

|  |
| --- |
| def rf(x\_train, y\_train, x\_test):  est = RandomForestRegressor(n\_estimators=10, criterion='mse',  max\_depth =5, max\_features = 'sqrt')  predictor = est.fit(X\_train, y\_train)  result = predictor.predict(X\_test))  return result |

代码片段5-10 随机森林使用示例

需要注意的是Scikit-learn中随机森林算法的参数与随机梯度决策树参数不一样，随机森林常用参数如表5-6所示。

表5-6 随机森林常用参数说明

|  |  |
| --- | --- |
| 参数 | 说明 |
| criterion | 每一个节点进行分裂时的判别条件，缺省值为mse |
| max\_features | 每一次节点进行分裂时使用的特征个数 |
| n\_estimators | 树的总数 |
| max\_depth | 每棵子树的最大高度 |
| max\_leaf\_nodes | 叶子节点最大个数 |
| oob\_score | 是否使用oob\_score评估泛华误差 |

### 5.6.6 算法融合

算法预测模块首先使用交叉验证对上述四种算法训练出的模型进行评估，选出各个算法的最优模型。然后对这四个模型的票房预测结果进行算法融合，并由此得到最终预测结果，算法最终融合的架构如图5-5所示。算法融合的具体步骤如下：

1）将特征抽取模块抽取出的特征数据划分成训练集和测试集两个不同的

集和；

2）在训练集上分别对普通最小二乘法、局部加权线性回归、随机森林以

及随机梯度回归树算法使用交叉验证得到各个算法最好的参数组合；

3）使用上述最佳参数组合分别训练各个算法的最优模型；

4）对第3步中预测效果最好的的四个模型的输出结果按照一定的比例进

行加权融合，计算公式为：

（5-17）

其中si为第i个模型的票房预测结果，wi为其相对应的权重。score为最终的票房预测结果。



图5-5 预测算法融合

## 5.7 本章小结

本章在概要设计的基础上对系统进行了详细设计与实现。具体来说，主要对用户登录模块、电影数据查询模块、数据抓取模块、数据预处理模块、特征抽取模块以及算法预测模块进行了详细设计和说明，在进行详细设计的过程中结合第2章介绍的各种工具和算法对各个模块的实现进行了论述，同时给出了每个模块功能的核心实现代码。

# 第6章系统部署与测试

本章主要描述系统的测试与部署。首先介绍测试的硬件配置和使用的软件版本，并对系统的软件安装部署进行了阐述，然后对系统的功能和性能进行测试。测试主要集中在用户登录、电影数据查询、数据抓取、数据预处理、特征抽取和电影票房预测这六个方面。最后根据测试的结果对系统的功能和性能进行分析。

## 6.1测试环境

### 6.1.1 硬件环境

本系统的开发环境包含10个服务器节点，每个节点的详细配置如表6-1所示。

表6-1 服务器节点硬件配置

|  |  |
| --- | --- |
| 名称 | 配置 |
| 硬盘 | 6TB机械硬盘 |
| 内存 | 16GB |
| 处理器 | 8 CORES |
| 操作系统 | Centos6.5 |

### 6.1.2 软件环境

本系统使用的软件工具包括Hadoop、Java语言、Python语言、Beautifu Soup、Scikit-learn、Mysql等软件，具体的版本信息如表6-2所示。

表6-2 系统软件配置

|  |  |
| --- | --- |
| 名称 | 版本 |
| Java | 1.7 |
| Python | 2.7 |
| Hadoop | 2.4.1 |
| Beautiful Soup | 4.4 |
| Scikit-learn | 0.17 |

续表6-2 系统软件配置

|  |  |
| --- | --- |
| Mysql | 5.7 |
| Spring | 4.3.2 |
| Struts | 2.3.7 |
| Hibernate | 5.1.2 |

## 6.2系统部署安装

本节主要介绍Hadoop的安装。在部署Hadoop之前集群已经安装JDK并实现各服务器间SSH免密码登录。Hadoop的安装配置步骤如下：

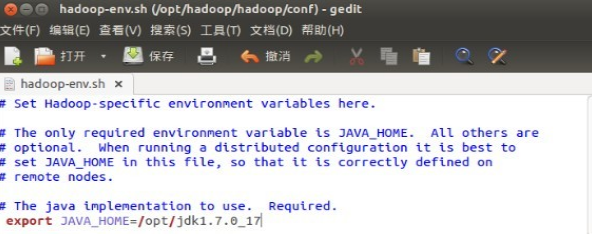
1、配置hadoop-env.sh文件。在其中添加JDK路径。配置如图6-1所示。

图6-1hadoop-env.sh文件配置

2、配置core-site.xml文件。该文件是Hadoop的核心配置文件，目的是设置

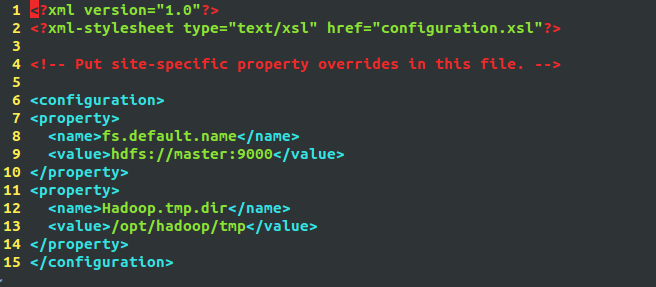
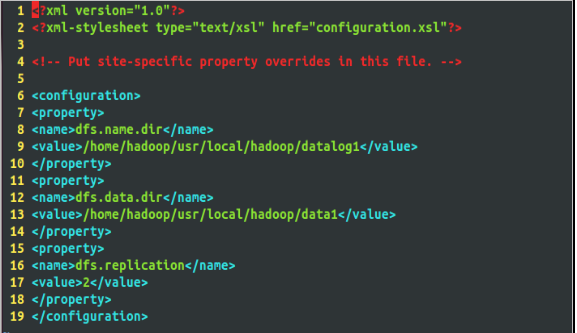
HDFS的端口和地址。具体配置如图6-2所示。

图6-2 core-site.xml文件配置

3、配置hdfs-site.xml文件。该文件主要用来设置HDFS上文件的副本数。在发生数据异常时，可以从副本中恢复出原始数据。在这里将副本数设置为2，具体配置如图6-3所示。

图6-3 hdfs-site.xml文件配置

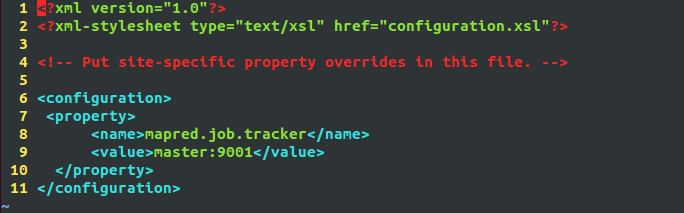
4、配置mapred-site.xml文件。该文件主要设置JobTracker的地址和端口，具体配置如图6-4所示。

图6-4 mapred-site.xml文件配置

5、配置masters和slaves文件，添加节点IP地址。

上述配置步骤完成后，运行hadoop namenode –formate进行HDFS文件系统格式化，然后运行start-all.sh脚本即可成功启动Hadoop系统。

## 6.3 功能测试

### 6.3.1 用户登录功能测试

1、测试用例

相应测试用例如表6-3所示。

表6-3系统登录测试用例

|  |  |  |
| --- | --- | --- |
| 测试用例描述 | 用户登录功能测试 | |
| 测试目的 | 测试系统的登录功能是否正确 | |
| 前提条件 | 系统正常运行 | |
| 测试内容 | 测试方法 | 预期结果 |
| 用户登录 | 使用正确和错的用户名和密码进行登录测试 | 正确的用户名和密码登录成功，错误的则登录失败 |
| 备足 | 无 | |

2、详细描述

错误的用户名和密码在登录系统时登录失败。当输入正确的用户名和密码时用户成功登录系统，与预期结果相同，该功能正确。用户登录界面如图6-5所示。

图6-5用户登录界面

### 6.3.2 电影数据查询功能测试

1、测试用例

相应测试用例如表6-4所示。

表6-4电影查询功能测试用例

|  |  |  |
| --- | --- | --- |
| 测试用例描述 | 数据查询功能测试 | |
| 测试目的 | 测试电影基本信息查、电影票房查询以及不同状态电影列表查询等功能的正确性 | |
| 前提条件 | 用户成功登录本系统 | |
| 测试内容 | 测试方法 | 预期结果 |
| 不同状态电影列表查询 | 用户登录系统后，选择不同状态的电影列表选项卡 | 根据用户选择的不同状态，返回对应的电影列表 |
| 电影基本信息与电影票房查询 | 用户登录系统后，搜索具体电影名 | 返回搜索电影的实际票房和预测票房，同时返回电影的演员、导演等基本信息 |
| 备注 | 电影拥有即将上映、正在上映、已经下映和全部电影四个状态 | |

2、详细描述

1）不同状态电影列表查询

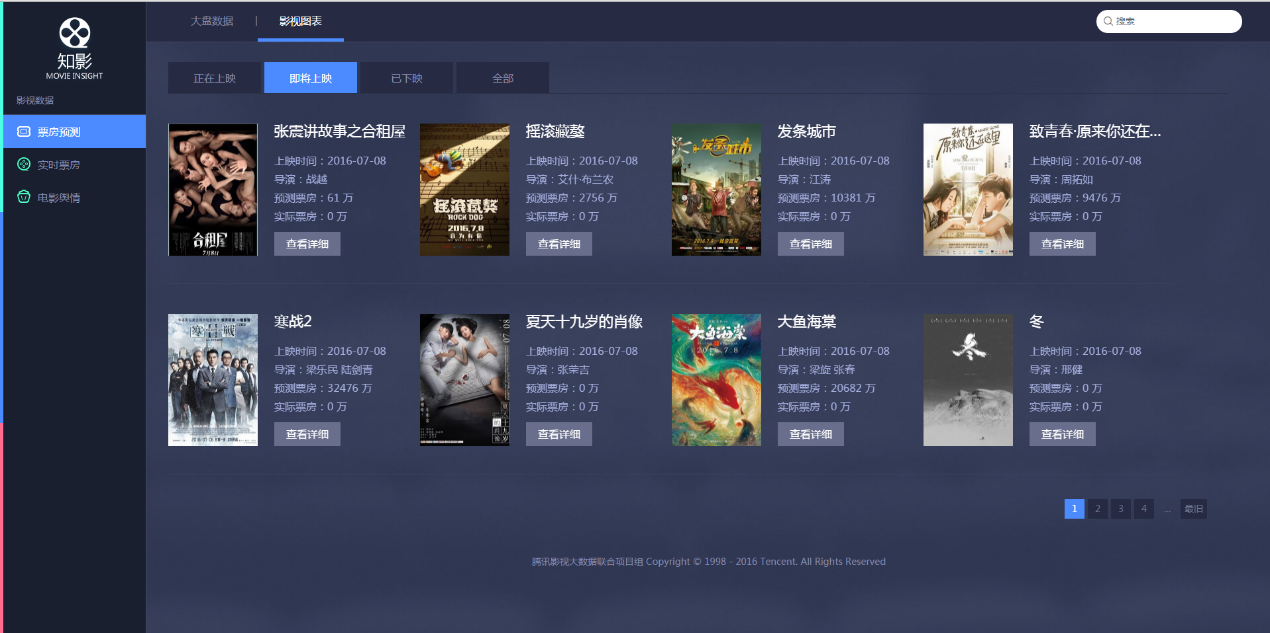
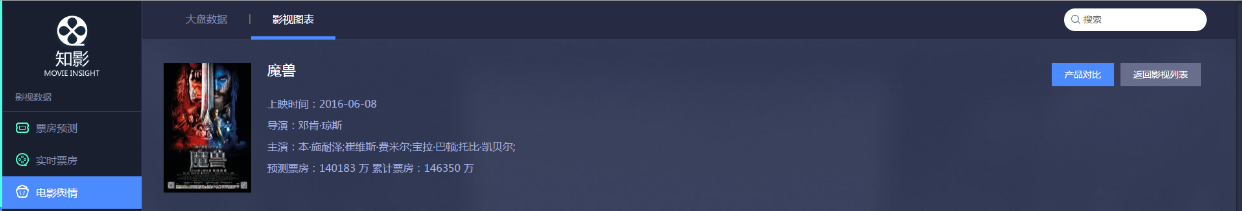
用户选择不同状态选项卡，返回不同电影列表。这里以即将上映电影为例，系统返回结果如图6-6所示。

图6-6不同状态电影列表

2）电影基本信息与电影票房查询

以魔兽为目标测试电影，系统的返回结果如图6-7所示。系统返回的数据包括电影上映时间，导演和演员名称，预测票房为140183万，实际票房为146350万。

图6-7 电影基本信息和票房数据

综上，电影数据查询功能测试用例的实际结果与预期结果一致，故该功能正确。

### 6.3.3 数据抓取功能测试

1、测试用例

相应测试用例如表6-5所示。

表6-5电影插叙功能测试用例

|  |  |  |
| --- | --- | --- |
| 测试用例描述 | 数据抓取功能测试 | |
| 测试目的 | 检测系统能否正常抓取数据 | |
| 前提 | 系统正常启动 | |
| 测试内容 | 测试方法 | 预期结果 |
| 指定网页链接抓取 | 对指定的网页数据进行爬取 | 成功抓取目标网页数据并存储至HDFS |
| 备注 | 无 | |

2、详细描述

测试的目标网页是预告片网站中的最近更新页面，该页面主要展示了近期上映的电影。对该网页进行数据爬取，爬取结果的数据片段如图6-8所示。网页数据抓取后，保存至HDFS。该功能的实际结果与预期结果一致，功能正确。

图6-8数据抓取结果

### 6.3.4 数据预处理功能测试

1、测试用例

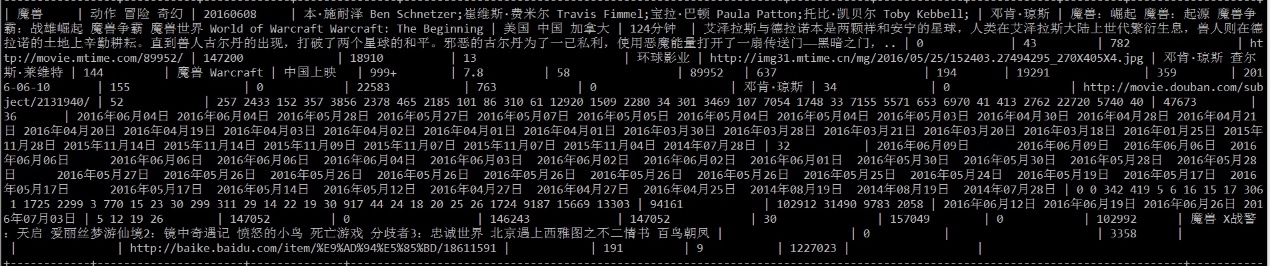
相应测试用例如表6-6所示。

表6-6数据预处理功能测试用例

|  |  |  |
| --- | --- | --- |
| 测试用例描述 | 数据预处理功能测试 | |
| 测试目的 | 检测系统能否正常进行数据解析、清洗、存储电影数据 | |
| 前提 | 网络数据成功抓取 | |
| 测试内容 | 测试方法 | 预期结果 |
| 指定具体电影 | 对指定的电影存储的最终数据和原始数据进行对比 | 电影数据正常 |
| 备注 | 无 | |

2、详细说明

数据预处理要经过数据解析、数据过滤和数据存储等处理步骤。这里以魔兽电影为例，数据经过预处理后，最终存储在Mysql数据库中的数据如图6-9所示。与原始数据字段对比，数据预处理得到的最终数据无异常，因此，该功能正确。

图6-9数据预处理结果

### 6.3.5 特征抽取功能测试

1、测试用例

相应测试用例如表6-7所示。

表6-7特征抽取功能测试用例

|  |  |
| --- | --- |
| 测试用例描述 | 特征抽取功能测试 |
| 测试目的 | 检测特征量化计算和特征数值修正是否正确 |
| 前提 | 数据预处理成功 |

续表6-7特征抽取功能测试用例

|  |  |  |
| --- | --- | --- |
| 测试内容 | 测试方法 | 预期结果 |
| 特征量化计算 | 对指定的电影的特征进行计算 | 特征计算结果正确 |
| 特征数值修正 | 对上述特征进行数值修正处理 | 处理得到的最终特征数据正确 |
| 备注 | 无 | |

2、详细描述

根据5.5.1节构造的电影特征体系，得到电影特征如表6-8所示。

表6-8电影特征表

|  |  |
| --- | --- |
| director\_influence\_1 | 第1导演影响力 |
| director\_influence\_2 | 第2导演影响力 |
| actor\_influence\_1 | 第1主演影响力 |
| actor\_influence\_2 | 第2主演影响力 |
| actor\_influence\_3 | 第3主演影响力 |
| actor\_influence\_4 | 第4主演影响力 |
| script\_writer\_influence\_1 | 第1编剧影响力 |
| script\_writer\_influence\_2 | 第2编剧影响力 |
| new\_year | 元旦 |
| sprint\_festival | 春节 |
| tomb\_sweeping\_day | 清明节 |
| labor\_day | 劳动节 |
| children\_day | 儿童节 |
| dragon\_boat\_festival | 端午节 |
| mid\_autumn\_festival | 中秋节 |
| tanabata\_festival | 七夕节 |
| nation\_day | 国庆节 |
| summer\_vacation | 暑假 |
| winter\_vacation | 寒假 |
| christmas | 圣诞节 |
| valentine\_day | 情人节 |
| social\_hot\_index | 社交热度 |
| pos\_sent\_index | 正面舆情数 |
| neg\_sent\_index | 负面舆情数 |
| traile\_hot\_index | 预告片热度 |
| comedy | 喜剧 |
| adventure\_movie | 冒险 |
| science\_fiction | 科幻片 |

续表6-8电影特征表

|  |  |
| --- | --- |
| thriller | 惊悚悬疑 |
| dracula\_movie | 恐怖片 |
| documentary | 记录片 |
| war\_movie | 战争片 |
| action\_movie | 动作片 |
| family\_movie | 家庭片 |
| affectional\_movie | 爱情片 |
| cartoon\_movie | 动画电影 |
| music\_movie | 音乐电影 |
| search\_rate | 搜索转化率 |
| movie\_play\_number | 电影排片数 |
| competitor\_number | 同档期竞品数 |

1）特征量化计算

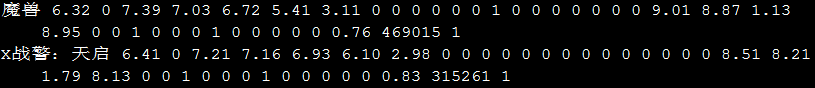
以魔兽和x战警：天启两部同档期的电影为例，对它们进行特征量化计算。计算得到的特征结果如图6-10所示，结合特征表和电影数据，电影特征计算结果正确。

图6-10电影特征量化计算结果

2）特征数值修正

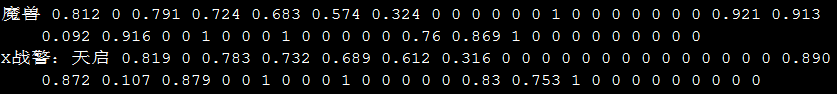
对魔兽和x战警：天启进行特征量化得到的影片特征值加以数值修正处理，修正后的特征数据如图6-11所示。可见，在特征量化的基础上，系统成功的对人物影响力、社交热度、舆情指数、预告片热度以及排片数进行了归一化处理，同时对同档期电影竞品数进行了连续特征离散化处理，离散化为10个特征，处理结果正确。

图6-11电影特征数值修正结果

### 6.3.6 算法预测结果评估

1、数据集划分说明

本系统对2013年1月到2016年4月总共947部电影进行算法模型的训练与评测。对数据集进行划分，选取其中最近上映的的247部电影作为测试集，剩余700部电影作为训练集。

2、评价指标

由于本系统进行票房预测实质上是一个回归问题，因此本系统采用的预测结果评判标准是R2值，R2的计算公式为：

（6-1）

其中fi和yi分别是样本i的预测值和真实值，ym是所有样本目标值的平均值。因此，R2反映了估计值与实际值的相关程度，R2值越小相关程度越低，R2值越大相关程度越高，R2值的大小在0到1之间。

3、交叉验证

交叉验证是是一种模型选择方法，该方法随机地将数据集切分成n份大小相同的互不相交的子集，然后利用其中的n-1份作为模型训练集，剩余数据作为模型测试集。对n种不同的数据组合重复进行上述操作，选出在测试集上表现最佳的模型。

4、算法最佳模型参数

对本系统涉及到的预测算法进行模型系数选择，其中普通最小二乘法不需要进行模型调参。随机森林、随机梯度回归树以及局部加权线性回归算法的模型系数选择如表6-9、表6-10和表6-11所示。

表6-9随机森林最优参数

|  |  |  |  |
| --- | --- | --- | --- |
| max\_features | n\_estimators | max\_depth | R2 |
| 7 | 500 | 5 | 0.9209 |
| 7 | 400 | 5 | 0.9226 |
| 7 | 350 | 5 | 0.9199 |
| 7 | 300 | 5 | 0.9201 |
| 7 | 200 | 5 | 0.9123 |
| 6 | 400 | 5 | 0.9103 |
| 8 | 400 | 5 | 0.9185 |
| 5 | 400 | 5 | 0.9033 |

续表6-9随机森林最优参数

|  |  |  |  |
| --- | --- | --- | --- |
| 9 | 400 | 5 | 0.9207 |
| 7 | 400 | 4 | 0.9057 |
| 7 | 400 | 6 | 0.9182 |
| 7 | 400 | 7 | 0.9149 |

表6-10随机梯度回归树模型参数选择

|  |  |  |  |
| --- | --- | --- | --- |
| learning\_rate | n\_estimators | max\_depth | R2 |
| 0.1 | 200 | 6 | 0.9125 |
| 0.1 | 300 | 6 | 0.9179 |
| 0.1 | 350 | 6 | 0.9214 |
| 0.1 | 400 | 6 | 0.9181 |
| 0.1 | 500 | 6 | 0.9173 |
| 0.1 | 350 | 5 | 0.9193 |
| 0.1 | 350 | 7 | 0.9201 |
| 0.1 | 350 | 8 | 0.9197 |
| 0.01 | 350 | 6 | 0.9203 |
| 0.05 | 350 | 6 | 0.9198 |
| 0.5 | 350 | 6 | 0.9136 |
| 1 | 350 | 6 | 0.9103 |

表6-11 局部加权线性回归参数选择

|  |  |
| --- | --- |
| 权值变换速率k | R2 |
| 0.01 | 0.8912 |
| 0.03 | 0.9012 |
| 0.05 | 0.8964 |
| 0.07 | 0.9046 |
| 0.09 | 0.9053 |
| 0.1 | 0.9128 |
| 0.3 | 0.9117 |
| 0.5 | 0.9136 |
| 0.7 | 0.9123 |
| 0.9 | 0.9031 |
| 1 | 0.9012 |
| 5 | 0.8834 |

结合表6-9、表6-10和表6-11，随机森林的最佳参数组合是max\_features取7，n\_estimators取400，max\_depth取5。随机梯度回归树的最佳参数组合是learning\_rate取0.1，n\_estimators取350，max\_depth取6。局部加权线性回归权重变化速率的最佳值为0.5。

5、算法融合结果

对各算法的最优模型在测试集上的预测结果进行融合，融合的权重系数和最终预测结果如表6-12所示。从实验结果可以得出，当普通最小二乘法、局部加权线性回归算法、随机森林和随机梯度回归算法的权重系数分别为0.11、0.15、0.31和0.43时R2值最高为0.9301，预测值和实际值的线性相关度如图6-12所示。

表6-12模型融合系数

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 普通最小二乘法 | 局部加权线性回归 | 随机森林 | 随机梯度回归树 | R2 |
| 0.25 | 0.25 | 0.25 | 0.25 | 0.9126 |
| 0.12 | 0.31 | 0.35 | 0.22 | 0.9238 |
| 0.17 | 0.23 | 0.31 | 0.29 | 0.9241 |
| 0.21 | 0.26 | 0.29 | 0.24 | 0.9217 |
| 0.36 | 0.17 | 0.21 | 0.26 | 0.9118 |
| 0.41 | 0.13 | 0.15 | 0.31 | 0.9049 |
| 0.18 | 0.31 | 0.05 | 0.46 | 0.9211 |
| 0.15 | 0.21 | 0.39 | 0.25 | 0.9263 |
| 0.11 | 0.15 | 0.31 | 0.43 | 0.9301 |
| 0.23 | 0.42 | 0.11 | 0.24 | 0.9189 |
| 0.01 | 0.09 | 0.32 | 0.58 | 0.9232 |

图6-12预测值与实际值相关度

## 6.4 性能测试

票房预测系统除了在功能上要满足实际需求外，其系统的整体性能也是一个重要的考量因素。对票房预测系统进行性能测试主要体现在数据抓取与数据预处理的速度、特征抽取的速度以及算法执行的速度。其中数据抓取与数据预处理是并行执行的，因此需要从整体上考虑它们的性能。而特征抽取与算法执行是串行执行的，故分别独立测试这两部分的性能。由于电影上映时间分散，每一次系统进行处理的电影数目不会超过10部，当有新电影准备上映时系统会自动将该电影电影添加到目标电影列表进行票房预测。

数据抓取与数据预处理采用了分布式并行的方式进行相应处理。表6-13记录了该部分功能在不同的条件下的运行时间。从实验结果可以看出，在正常情况下每次处理10部电影并且只处理这些电影过去90天的数据，此时的运行时间为为31分钟。当电影数据增加到30天，需要处理的数据范围增加到过去120天时，此时的运行时间为106分钟。该部分需要在两小时内完成，因此该部分性能满足需求。

表6-13数据抓取与数据预处理运行耗时

|  |  |  |
| --- | --- | --- |
| 电影数目 | 数据范围 | 运行时间 |
| 10 | 90天 | 31min |
| 120天 | 39min |
| 30 | 90天 | 83min |
| 120天 | 106min |

特征抽取功能的性能测试结果如表6-14所示。分析数据可得，即使电影数据增加到30部，需要处理的数据范围增加到120天，此时的运行时间仅需698s，而正常情况下的运行时间为162s。

表6-14特征抽取运行耗时

|  |  |  |
| --- | --- | --- |
| 电影数目 | 数据范围 | 运行时间 |
| 10 | 90天 | 162s |
| 120天 | 229s |
| 30 | 90天 | 523s |
| 120天 | 698s |

算法预测功能的性能测试结果如表6-15所示。对于不同的数据集，每个样本的特征数是不变的，因此处理的数据范围的大小对算法运行时长的影响不大，而电影数目的增长会增加模型训练和预测的耗时。从实验结果可得，算法预测可以在数十秒内完成，性能较好。

表6-15特征抽取运行耗时

|  |  |  |
| --- | --- | --- |
| 电影数目 | 数据范围 | 运行时间 |
| 10 | 90天 | 46s |
| 120天 | 48s |
| 30 | 90天 | 87s |
| 120天 | 91s |

综上所述，电影票房预测系统在实际测试中拥有较好的性能表现，能够满足实际业务的性能需求。同时，系统性能进一步的提升可以通过增加集群的计算资源实现。

## 6.5本章小结

本章首先对系统的硬件与软件环境进行了简要的介绍。然后描述了Hadoop分布式平台的部署过程，接着通过不同的测试用例对用户登录功能、电影数据查询功能、数据抓取功能、数据预处理功能、特征抽取功能以及算法预测功能进行了测试，测试结果验证了上述各个功能的正确性。系统最终预测结果的R2值为0.9301，表明票房预测值具有较高准确性。在本章的最后对系统进行了性能测试，测试结果表明系统具有较好的性能。



# 第7章 总结与展望



## 7.1 论文工作总结

本文详细地介绍了基于Scikit-learn的电影票房预测系统的设计与实现。在明确了选题的背景意义以及对国内相关领域的研究现状做了大量调研工作后，本文对系统应具有的功能进行了详细阐述。在数据展示方面，系统支持用户对电影数据的查询，在后台处理上，系统提供了网络数据抓取、数据预处理、数据特征抽取以及算法预测等功能。在完成需求分析后，对系统的概要设计进行了阐述，主要论述了系统的整体架构和模块划分，并介绍了各个模块的主要功能和相关数据表设计。系统的底层存储设备包括分布式文件系统HDFS和Mysql数据库，系统的计算采用Hadoop MapReduce并行计算框架，最终的票房预测算法由Scikit-learn机器学习算法库提供，同时本人实现了两个预测算法。系统共分为用户登录、电影数据查询、数据抓取、数据预处理、特征抽取以及算法预测六个模块。其中用户登录模块提供了用户访问系统的接口；电影数据查询模块使得用户可以直接访问电影相关信息；数据抓取模块主要对外网与电影相关的数据进行抓取；预处理模块则对抓取的数据进行解析、过滤、存储等操作；特征抽取模块完成了电影特征数据的量化计算；算法预测模块则在电影特征数据的基础上对电影票房进行最终预测，在进行票房预测时选用了普通最小二乘法、局部加权线性回归、随机森林和随机梯度回归树四个预测算法，并对每个算法预测的最优模型的预测结果进行算法融合得到最终预测结果。完成系统概要设计后，本文讨论了系统的详细设计和实现，对各个模块内部的具体接口进行了描述，同时给出了重要函数的核心实现代码。本文的最后对系统的整体功能和性能进行了测试，结果表明系统各项功能正确且系统性能较好。票房预测值的R2值达到0.9301，具有较高准确性。

本文所设计的票房预测系统目前在腾讯科技有限公司内部使用，同时对外部的某些电影相关企业提供了使用接口。系统的票房数据对影片制作方和宣发方提供数据支持，能帮助他们设计更好的营销策略。

## 7.2 不足与展望

电影票房预测系统涉及电影数据的抓取、预处理、数据存储和计算等诸多方面的内容，在系统的设计与开发的过程中存在一些不足之处需要改进。第一，数据种类不够丰富，数据类型越多则越能从不同的角度描述一部电影。除了表4-5所展示的当前系统使用的电影数据类型外，还应挖掘新的电影数据，例如电影主要演员当前的社交舆论倾向；第二，特征挖掘不够深入，特征质量的高低对最终的预测结果有较大的影响。除了当前系统使用的特征外，还应更深层次地挖掘其它特征，例如进行特征的交叉学习或者使用深度学习技术学习更多的新特征；第三，算法融合方法有待改善。本系统在进行算法预测时使用的融合方法是线性加权融合，这种融合算法比较直观，但是对结果的提升并不是特别明显，因此还应采用其它融合技术，如利用神经网络学习不同的权重值进行最终模型融合。第四，系统功能需要完善，目前系统提供的预测只是针对电影的总票房而言。除了总票房预测外还应增加电影同步票房预测等其它功能。为了进一步完善电影票房预测系统的功能和性能，未来的主要工作应围绕上述四点展开。

# 参考文献

1. 吴宣文．影响电影票房的六大因素[J]．中国电影市场，2016, 16(4): 14-15.
2. 陶雪娇，胡晓峰，刘洋．大数据研究综述[J]．系统仿真学报，2013, 25(6): 142-146.
3. 程学旗，靳小龙，王元卓，等．大数据系统和分析技术综述[J]．软件学报，2014, 25(9): 1889-1908.
4. 李航．统计学习方法[M]．北京：清华大学出版社，2012: 1.
5. 陈洪涛，肖如良，倪友聪等．融合推荐潜力的个性化趋势预测的混合推荐模型[J]．计算机应用，2014, 34(1): 218-221.
6. 项亮．推荐系统实践[M]．北京：人民邮电出版社，2012: 193.
7. 隋建凯，刘惠义，严烁.一种基于随机森林的特征匹配方法[J]．计算机与现代化，2014, 2(4): 81-85.
8. Geoffrey H, Li D, Dong Y, et al. Deep Neural Networks for Acoustic Modeling in Speech Recognition[J]. [IEEE Signal Processing Magazine, 2012,](http://ieeexplore.ieee.org/xpl/RecentIssue.jsp?punumber=79)29(6): 82-87.
9. Sanjay G, Howard G, Shun L. The Google File System[J]. Acm Sigops Operating Systems Review, 2003, 37(5): 29-43.
10. Xu L. Implementing WebGIS on Hadoop: A case study of improving small file I/O performance on HDFS[A]. In:[2009 IEEE International Conference on Cluster Computing and Workshops](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5281774)[C]. New Orleans, LA: IEEE, 2009: 1-8.
11. Dean J, Ghemawat S. MapReduce: Simplified Data Processing on Large Clusters.[J]. Communications of the ACM, 2008, 51(1): 107-113.
12. Matei Z, Mosharaf C, Das T, et al. **Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing[A]. In:** NSDI'12 Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation[C]. USA, CA: USENIX Association Berkeley, 2012: 1765-1773.
13. Shkapenyuk V, Suel T. Design and implementation of a high-performance distributed Web crawler[A]. In: [18th International Conference on](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=7807) Data Engineering[C]. San Jose, CA: IEEE, 2002: 357-368.
14. 钟晓，马少平．数据挖掘综述[J]．模式识别与人工智能，2001, 12(1): 119-124.
15. Condie T, Mineiro P, Polyzotis N, et al. Machine learning for big data[A]. In: Kenneth Ross. Proceedings of the 2013 International Conference on Management of Data[C]. New York: ACM, 2013: 939-942.
16. Litman B.Predicting success of theatrical movies: An empirical study[J]. The Journal of Popular Culture, 1983, 16(4): 159-175.
17. Sharda R, Delen D. Predicting box-office success of motion pictures with neural networks[J]. Expert System with Application, 2006, 30(2): 243-254.
18. Asur S, Huberman B. Predicting the Future with Social Media[J]. [Web Intelligence and Intelligent Agent Technology](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=5613987), 2010, 1(6): 492-499.
19. Berman D, Chowdhury N, Singha R. To predict possible profit/loss of a movie to be launched using MLP with back-propagation learning[A]. In: [Communications, Devices and Intelligent Systems (CODIS), 2012 International Conference on](http://ieeexplore.ieee.org/xpl/mostRecentIssue.jsp?punumber=6414853) [C]. Koljata: IEEE, 2012: 322-325.
20. Panaligan R, Andrea C. Quantifying Movie Magic with Google Search[OL]. https://www.thinkwithgoogle.com/research-studies/quantifying-movie-magic.html
21. Zheng J, Zhou S. Modeling on box-office revenue prediction of movie based on neural network[J]. Journal of Computer Application, 2014, 34(3): 742-748.
22. 罗捷．基于电影评价的进口影片票房预测研究[D]．重庆：重庆大学，2015.
23. 周江，王伟平，孟丹，等．面向大数据分析的分布式文件系统关键技术[J]．计算机研究与发展，2014, 51(2): 382-394.
24. Lakshman A, Malik P. Cassandra: a decentralized structured storage system[J]. Acm Sigops Operating Systems Review, 2010, 44(2): 35-40.
25. 李妹芳．ODPS权威指南：阿里大数据平台应用开发实践[M]．北京：人民邮电出版社，2014: 2.
26. Melnik S, Gubarev A, Long J, et al. Dremel: Interactive Analysis of Web-Scale Datasets.[J]. Proceedings of the VLDB Endowment, 2010, 3(2): 330-339.
27. Watkins C, Dayan P. Q-learning[J]. Machine Learning, 1992, 8(3): 279-292.
28. Chang C, Lin C. LIBSVM: A library for support vector machines[J]. ACM Transactions on Intelligent Systems and Technology (TIST), 2011, 2(3): 113-129.
29. Fan R, Chang K, Lin C, et al. LIBLINEAR: A Library for Large Linear Classification[J]. Journal of Machine Learning Research, 2008, 9(8): 1871-1874.
30. **Yih W, Goodman J, Carvalho V. Finding advertising keywords on web pages[A].** In: Proceedings of the 15th international conference on World Wide Web. New York, NY: ACM, 2006: 213-222.
31. Pudil P, Kitter J. Floating search methods in feature selection[J]. [Pattern Recognition Letters](http://www.sciencedirect.com/science/journal/01678655), 2003, 15(11): 1119-1125.
32. Guyon I, Elisseeff A. An Introduction to Variable and Feature Selection[J]. Journal of Machine Learning Research, 2003, 38(11): 1157-1182.
33. 计智伟, 胡珉, 尹建新. 特征选择算法综述[J]. 电子设计工程, 2011, 19(9): 46-51.
34. Kohavi R. Wrappers for feature subset selection[J]. [Artificial Intelligence](http://www.sciencedirect.com/science/journal/00043702), 1997, 97(2): 273-327.
35. 孔康，汪群山，梁万路．L1正则化机器学习问题求解分析[J]．人工智能及识别技术，2011, 37(17): 175-178.
36. 孙志军，薛磊，许阳明，等．深度学习研究综述[J]．计算机应用研究，2012, 29(8): 2806-2810.
37. Cleveland W, Devlin S. Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting[J]. [Journal of the American Statistical Association, 2012, 83(40): 596-610.](http://amstat.tandfonline.com/toc/uasa20/current)
38. Svetnik V, Liaw A, Tong C, et al. Random Forest:  A Classification and Regression Tool for Compound Classification and QSAR Modeling[J]. 2003, 43(6): 1947-1958.
39. Friedman J. Greedy Function Approximation: A Gradient Boosting Machine[J]. The Annals of Statistics, 2001, 29(5): 1189-1232.

# 致谢

时光飞逝，两年多的硕士学习生活即将步入尾声。这两年中，我的每一步成长都离不开良师益友的帮助。在这里，我要对他们表达诚挚的感谢。

首先，我要感谢毕业论文校内导师殷保群教授和企业导师林民龙博士，感谢你们在整个毕业论文的选题、开题、实验到最终完成论文书写的过程中给予我的建议和鼓励。殷老师知识渊博，思维严谨，在阅读和修改我的开题报告和论文时指出了多处关键问题，并给出了宝贵的修改建议。此外，殷老师还教授我正确的研究方法和工程理念，这些知识令我受益匪浅。林博士在指导我的毕业论文和实验的过程中教给我许多理论原理和工程技巧，这些知识使我的论文所设计的电影票房预测系统取得了一个较好的成果。再次对殷老师和林博士表示最诚挚的感谢。

其次，我要感谢母校中国科学技术大学对我的培养。能成为中科大的一员，我感到无比的骄傲和自豪。衷心祝福母校，愿母校越来越好。

最后，我要感谢我的家人对我的支持，他们的支持帮助我顺利完成学业，感谢他们一路伴我进步和成长。