

电影推荐系统

项目代码

2018年9月10日

**目 录**

[1 目录结构图 3](#_Toc525658743)

[1.1 Node.js代码目录结构 3](#_Toc525658744)

[1.2 推荐算法代码目录结构 3](#_Toc525658745)

[2 项目代码 4](#_Toc525658746)

[2.1 Node.js项目代码 4](#_Toc525658747)

[2.1.1 myapp.js代码 4](#_Toc525658748)

[2.1.2 登录界面代码 7](#_Toc525658749)

[2.1.3 注册界面代码 8](#_Toc525658750)

[2.1.4 登录失败代码 9](#_Toc525658751)

[2.1.5 主页界面代码 10](#_Toc525658752)

[2.1.6 评分界面代码 13](#_Toc525658753)

[2.1.7 推荐界面代码 16](#_Toc525658754)

[2.2 推荐算法代码 19](#_Toc525658755)

[2.2.1 Pom.xml代码 19](#_Toc525658756)

[2.2.2 MoviesRecommond代码 21](#_Toc525658757)

[3 项目截图 26](#_Toc525658758)

[3.1 Node.js截图 26](#_Toc525658759)

[3.1.1 登录界面 26](#_Toc525658760)

[3.1.2 注册界面 26](#_Toc525658761)

[3.1.3 主页截图 27](#_Toc525658762)

[3.1.4 评分截图 27](#_Toc525658763)

[3.1.5 推荐界面 28](#_Toc525658764)

[3.2 推荐算法截图 28](#_Toc525658765)

[3.2.1 获取最佳模型结果图 28](#_Toc525658766)

[3.2.2 推荐结果 29](#_Toc525658767)

# 目录结构图

## Node.js代码目录结构

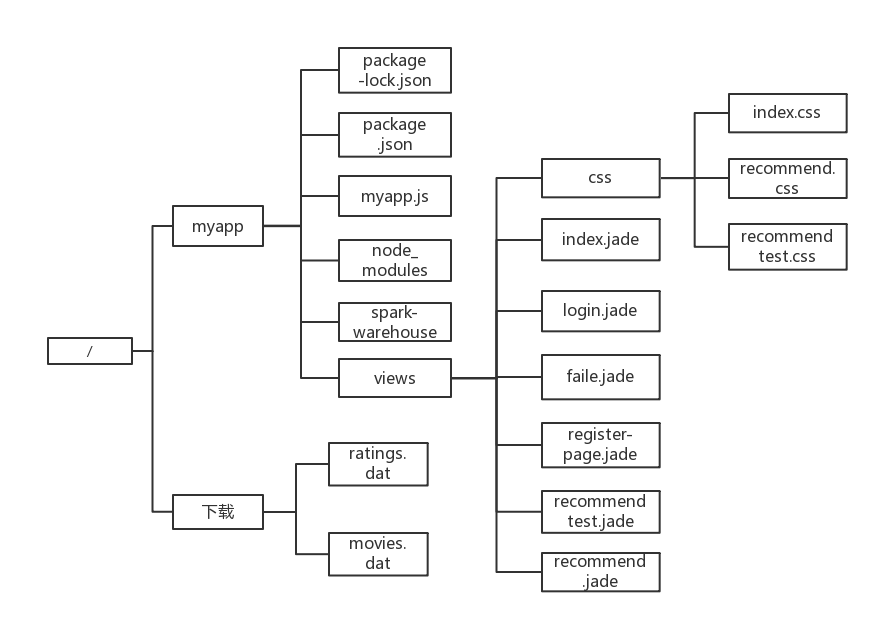


图1 前端结构目录图

## 推荐算法代码目录结构

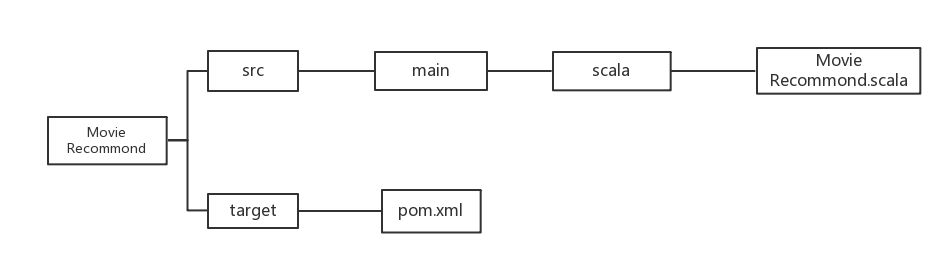


图2 后端结构目录图

# 项目代码

## Node.js项目代码

### myapp.js代码

myapp.js代码如下：

|  |
| --- |
| /\*\*  \* express接收html传递的参数  \*/  var express=require('express');  var bodyParser = require('body-parser')  const exec = require('child\_process').exec  var app=express();  var mysql=require('mysql');  app.set('view engine', 'jade');  app.set('views', './views');  app.use(bodyParser.urlencoded({extended: false}))  app.use(bodyParser.json())  var userid;  var name;  var movieid = new Array(10);  /\*\*  \* 配置MySQL  \*/  var connection = mysql.createConnection({  host : '127.0.0.1',  user : 'root',  password : '123456',  database : 'tuijian',  port:'3306'  });  connection.connect();  /\*\*  \* 跳转到网站首页  \*/  app.get('/',function (req,res) {  res.render('index',{title:'电影推荐系统'});  })  /\*\*  \* 跳转到登录界面  \*/  app.get('/login',function (req,res) {  res.render('login',{title:'登录'});  })  /\*\*  \* 实现登录验证功能，并随机读取数据库的电影  \*/  app.post('/login',function (req,res) {  name=req.body.username.trim();  var pwd=req.body.pwd.trim();  console.log('username:'+name+'password:'+pwd);  var selectSQL = "select \* from userinfo where username = '"+name+"' and password = '"+pwd+"'";  connection.query(selectSQL,function (err,rs) {  if(rs.length==0){  res.render('faile',{title:'登录失败'});  }  else{  userid=rs[0].userid;  console.log(rs);  console.log('ok');  var selectm = "SELECT \* FROM movieinfo where movieid < 7000 ORDER BY rand() LIMIT 10";  connection.query(selectm,function (err,rs) {  for(var i=0;i<10;i++){  movieid[i]=rs[i].movieid;  }  console.log(movieid);  res.render('recommendtest',{title:'推荐测试',rs:rs,message:name});  })  }  })  })  /\*\*  \* 跳转到注册页面  \*/  app.get('/registerpage',function (req,res) {  res.render('registerpage',{title:'注册'});  })  /\*\*  \* 实现注册功能  \*/  app.post('/register',function (req,res) {  name=req.body.username.trim();  var pwd=req.body.pwd.trim();  var user={username:name,password:pwd};  connection.query('insert into userinfo set ?',user,function (err,rs) {  if (err) throw err;  console.log('ok');  res.render('ok',{title:'Welcome User',message:name});  })  })  var server=app.listen(3000,function () {  console.log("userloginjade server start......");  })  /\*\*  \* 跳转到主页面  \*/  app.get('/index',function (req,res) {  res.render('index',{title:'主页'});  })  /\*\*  \* 选择电影  \*/  app.post('/tijiao',function (req,res) {  var rating = new Array(10);  rating[0] = req.body.a;  rating[1] = req.body.b;  rating[2] = req.body.c;  rating[3] = req.body.d;  rating[4] = req.body.e;  rating[5] = req.body.f;  rating[6] = req.body.g;  rating[7] = req.body.h;  rating[8] = req.body.i;  rating[9] = req.body.j;  for(var i=0;i<10;i++)  {  if(rating[i]!=0)  {  var user={userid:userid,movieid:movieid[i],rating:rating[i],ratingtime:1};  connection.query('insert into ratings set ?',user,function (err,rs) {  if (err) throw err;  console.log('ok');  })  }  }  const jarStr = '/usr/local/spark/bin/spark-submit --class "MoviesRecommond" /home/hadoop/MoviesRecommond.jar '+userid  exec(jarStr, function(err, stdout, stderr){  if(stderr){  console.log('ok1111');  var selectm = "select \* from recommend";  var movid = new Array(10);  connection.query(selectm,function (err,rs) {  for(var i=0;i<10;i++){  movid[i]=rs[i].movieid;  }  console.log(movid[0]);  var selectm = "select \* from movieinfo where movieid in("+movid[0]+","+movid[1]+","+movid[2]+","+movid[3]+","+movid[4]+","+movid[5]+","+movid[6]+","+movid[7]+","+movid[8]+","+movid[9]+")";  connection.query(selectm,function (err,rs) {  res.render('recommend',{title:'推荐结果',rs:rs,message:name});  })  })    }  })    }) |

### 登录界面代码

登录界面代码如下：

|  |
| --- |
| html  head  title!= title  style.  body{  background-image:url(https://timgsa.baidu.com/timg?image&quality=80&size=b9999\_10000&sec=1537261291133&di=c04553d39f158272a36be6e3ec0c8071&imgtype=0&src=http%3A%2F%2Fh.hiphotos.baidu.com%2Fzhidao%2Fpic%2Fitem%2Fc2fdfc039245d6885bc3be94a2c27d1ed21b2438.jpg);  }  #log{  padding-top: 2px;  margin-top: 10%;  margin-left: 37%;  background: white;  width: 25%;  height: 40%;  text-align: center;  }  body  div#log  form(action='/login', method='post')  h1 用户登录  br  span 账户:  input(type='text',name='username')  br  span 密码:  input(type='password',name='pwd')  br  br  input(type='submit',value='登录')  br  a(href='/registerpage', title='注册')注册  br  a(href='/index',title='主页')返回主页 |

### 注册界面代码

注册界面代码如下：

|  |
| --- |
| html  head  title!= title  style.  body{  background-image:url(https://timgsa.baidu.com/timg?image&quality=80&size=b9999\_10000&sec=1537261291133&di=c04553d39f158272a36be6e3ec0c8071&imgtype=0&src=http%3A%2F%2Fh.hiphotos.baidu.com%2Fzhidao%2Fpic%2Fitem%2Fc2fdfc039245d6885bc3be94a2c27d1ed21b2438.jpg);  }  #reg{  padding-top: 2px;  margin-top: 10%;  margin-left: 37%;  background: white;  width: 25%;  height: 40%;  text-align: center;  }  body  div#reg  form(action='/register', method='post')  h1 用户注册  br  span 账户:  input(type='text',name='username')  br  span 密码:  input(type='password',name='pwd')  br  br  input(type='submit',value='注册') |

### 登录失败代码

登录失败界面代码如下：

|  |
| --- |
| html  head  title!=title  style.  body{  background-image:url(https://timgsa.baidu.com/timg?image&quality=80&size=b9999\_10000&sec=1537261291133&di=c04553d39f158272a36be6e3ec0c8071&imgtype=0&src=http%3A%2F%2Fh.hiphotos.baidu.com%2Fzhidao%2Fpic%2Fitem%2Fc2fdfc039245d6885bc3be94a2c27d1ed21b2438.jpg);  text-align:center;  }  body  h1 登录失败请重试  a(href='/login', title='登录')返回登录 |

### 主页界面代码

主页界面代码如下：

|  |
| --- |
| html  head  title!=title  meta(charset='utf-8')  meta(name='description')  meta(name='keywords')  meta(name='author')  link(rel='shortcut icon', href='http://eduppp.cn/images/logo4.gif')  link(rel='apple-touch-icon', href='http://eduppp.cn/images/logo.gif')  style  include css/index.css  style(type='text/css').  #frame {/\*----------图片轮播相框容器----------\*/  position: absolute; /\*--绝对定位，方便子元素的定位\*/  width: 1500px;  height: 75%;  overflow: hidden;/\*--相框作用，只显示一个图片---\*/  border-radius:5px;  }  #dis {/\*--绝对定位方便li图片简介的自动分布定位---\*/  position: absolute;  left: -50px;  top: -10px;  opacity: 0.5;  }  #dis li {  display: inline-block;  width: 200px;  height: 20px;  margin: 0 650px;  float: left;  text-align: center;  color: #fff;  border-radius: 10px;  background: #000;  }  #photos img {  float: left;  width:1500px;  height:75%;  }  #photos {/\*---设置总的图片宽度--通过位移来达到轮播效果----\*/  position: absolute;z-index:9px;  width: calc(1500px \* 5);/\*---修改图片数量的话需要修改下面的动画参数\*/  }  .play{  animation: ma 20s ease-out infinite alternate;/\*\*/  }  @keyframes ma {/\*---每图片切换有两个阶段：位移切换和静置。中间的效果可以任意定制----\*/  0%,20% { margin-left: 0px; }  25%,40% { margin-left: -1500px; }  45%,60% { margin-left: -3000px; }  65%,80% { margin-left: -4500px; }  85%,100% { margin-left: -6000px; }  }  .num{  position:absolute;z-index:10;  display:inline-block;  right:10px;top:550px;  border-radius:100%;  background:#778899;  width:50px;height:50px;  line-height:50px;  cursor:pointer;  color:#fff;  background-clor:rgba(0,0,0,0.5);  text-align:center;  opacity:0.8;  }  .num:hover{background:#000;}  .num:hover,#photos:hover{animation-play-state:paused;}  .num:nth-child(2){margin-right:60px}  .num:nth-child(3){margin-right:120px}  .num:nth-child(4){margin-right:180px}  .num:nth-child(5){margin-right:240px}  #a1:hover ~ #photos{animation: ma1 .5s ease-out forwards;}  #a2:hover ~ #photos{animation: ma2 .5s ease-out forwards;}  #a3:hover ~ #photos{animation: ma3 .5s ease-out forwards;}  #a4:hover ~ #photos{animation: ma4 .5s ease-out forwards;}  #a5:hover ~ #photos {animation: ma5 .5s ease-out forwards;}  @keyframes ma1 {0%{margin-left:-1200px;}100%{margin-left:-0px;} }  @keyframes ma2 {0%{margin-left:-1200px;}100%{margin-left:-1500px;} }  @keyframes ma3 {100%{margin-left:-3000px;} }  @keyframes ma4 {100%{margin-left:-4500px;} }  @keyframes ma5 {100%{margin-left:-6000px;} }  body  div#navigation 团队9影片推荐系统  div#logreg  input(type='submit',value='登录',onclick="window.location='/login'")  input(type='submit', value='注册',onclick="window.location='/registerpage'")  div#mid  #frame  a#a5.num 5  a#a4.num 4  a#a3.num 3  a#a2.num 2  a#a1.num 1  #photos.play  img(src='http://img05.tooopen.com/products/20150130/44128217.jpg')  img(src='http://image.17173.com/bbs/v1/2012/11/14/1352873759491.jpg')  img(src='http://t1.27270.com/uploads/tu/201502/103/5.jpg')  img(src='http://img.doooor.com/img/forum/201507/15/171203xowepc3ju9n9br9z.jpg')  img(src='http://4493bz.1985t.com/uploads/allimg/170503/5-1F503140J0.jpg')  ul#dis  li 魔戒：霍比特人  li 魔境仙踪  li 阿凡达  li 大圣归来  li 拆弹专家 |

### 评分界面代码

评分界面代码如下：

|  |
| --- |
| html  head  title!= title  style  include css/recommendtest.css  style.  span{  width:295px;  height:15px;  line-height:15px;  overflow:hidden;  margin: 0 auto;  }  .movie select{  margin: 0 auto;  text-align:center;  position:absolute;  }  body  div#top 请在下面选择你喜欢的电影并评分  div#user  欢迎: #{message}  input(type='submit', value='退出' onclick="window.location='/index'")  form(action='/tijiao', method='post')  div#mid  div#movie  img(src=rs[0].picture+"")  span 电影名:#{rs[0].moviename}  br  select(name='a')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[1].picture+"")  span 电影名:#{rs[1].moviename}  br  select(name='b')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[2].picture+"")  span 电影名:#{rs[2].moviename}  br  select(name='c')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[3].picture+"")  span 电影名:#{rs[3].moviename}  br  select(name='d')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[4].picture+"")  span 电影名:#{rs[4].moviename}  br  select(name='e')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[5].picture+"")  span 电影名:#{rs[5].moviename}  br  select(name='f')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[6].picture+"")  span 电影名:#{rs[6].moviename}  br  select(name='g')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[7].picture+"")  span 电影名:#{rs[7].moviename}  br  select(name='h')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[8].picture+"")  span 电影名:#{rs[8].moviename}  br  select(name='i')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#movie  img(src=rs[9].picture+"")  span 电影名:#{rs[9].moviename}  br  select(name='j')  option(value='0') 未选择  option(value='1') 1  option(value='2') 2  option(value='3') 3  option(value='4') 4  option(value='5') 5  div#buttom  input(type='submit', value='提交') |

### 推荐界面代码

推荐界面代码如下：

|  |
| --- |
| html  head  title!= title  style  include css/recommend.css  style.  img{  border:0  }  body{  behavior:url("csshover.htc");  text-align:center;  }  #movie span{  display:none;  text-decoration:none;  height:330px;  //line-height:2px;  overflow:hidden;  text-align:left;  }  #movie:hover{  cursor:pointer;  }  #movie:hover span {  display:block;  position:absolute;  bottom:0;  left:0;  color:#FFF;  width:295px;  z-index:10;  background:#000;  filter:alpha(opacity=60);  -moz-opacity:0.5;  opacity:0.5;  }  body  div#top 以下是为你推荐的10部电影  div#user  欢迎: #{message}  input(type='submit', value='退出' onclick="window.location='/index'")  form(action='/tijiao', method='post')  div#mid  div#movie  img(src=rs[0].picture+"")  span  |电影名:#{rs[0].moviename}  br  |电影评分:#{rs[0].averating}  br  |电影简介:#{rs[0].description}  div#movie  img(src=rs[1].picture+"")  span  |电影名:#{rs[1].moviename}  br  |电影评分:#{rs[1].averating}  br  |电影简介:#{rs[1].description}  div#movie  img(src=rs[2].picture+"")  span  |电影名:#{rs[2].moviename}  br  |电影评分:#{rs[2].averating}  br  |电影简介:#{rs[2].description}  div#movie  img(src=rs[3].picture+"")  span  |电影名:#{rs[3].moviename}  br  |电影评分:#{rs[3].averating}  br  |电影简介:#{rs[3].description}  div#movie  img(src=rs[4].picture+"")  span  |电影名:#{rs[4].moviename}  br  |电影评分:#{rs[4].averating}  br  |电影简介:#{rs[4].description}  div#movie  img(src=rs[5].picture+"")  span  |电影名:#{rs[5].moviename}  br  |电影评分:#{rs[5].averating}  br  |电影简介:#{rs[5].description}  div#movie  img(src=rs[6].picture+"")  span  |电影名:#{rs[6].moviename}  br  |电影评分:#{rs[6].averating}  br  |电影简介:#{rs[6].description}  div#movie  img(src=rs[7].picture+"")  span  |电影名:#{rs[7].moviename}  br  |电影评分:#{rs[7].averating}  br  |电影简介:#{rs[7].description}  div#movie  img(src=rs[8].picture+"")  span  |电影名:#{rs[8].moviename}  br  |电影评分:#{rs[8].averating}  br  |电影简介:#{rs[8].description}  div#movie  img(src=rs[9].picture+"")  span  |电影名:#{rs[9].moviename}  br  |电影评分:#{rs[9].averating}  br  |电影简介:#{rs[9].description}  div#buttom |

## 推荐算法代码

### Pom.xml代码

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?>  <project xmlns="http://maven.apache.org/POM/4.0.0"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">  <modelVersion>4.0.0</modelVersion>  <groupId>dblab</groupId>  <artifactId>WordCount</artifactId>  <version>1.0-SNAPSHOT</version>  <properties>  <spark.version>2.1.0</spark.version>  <scala.version>2.11</scala.version>  </properties>  <dependencies>  <dependency>  <groupId>org.apache.spark</groupId>  <artifactId>spark-core\_${scala.version}</artifactId>  <version>${spark.version}</version>  </dependency>  <dependency>  <groupId>org.apache.spark</groupId>  <artifactId>spark-streaming\_${scala.version}</artifactId>  <version>${spark.version}</version>  </dependency>  <dependency>  <groupId>org.apache.spark</groupId>  <artifactId>spark-sql\_${scala.version}</artifactId>  <version>${spark.version}</version>  </dependency>  <dependency>  <groupId>org.apache.spark</groupId>  <artifactId>spark-hive\_${scala.version}</artifactId>  <version>${spark.version}</version>  </dependency>  <dependency>  <groupId>org.apache.spark</groupId>  <artifactId>spark-mllib\_${scala.version}</artifactId>  <version>${spark.version}</version>  </dependency>  <dependency>  <groupId>org.apache.spark</groupId>  <artifactId>spark-core\_2.11</artifactId>  <version>2.1.0</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>org.apache.spark</groupId>  <artifactId>spark-sql\_2.11</artifactId>  <version>2.1.0</version>  <scope>provided</scope>  </dependency>  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>5.1.40</version>  </dependency>  </dependencies>  <build>  <plugins>  <plugin>  <groupId>org.scala-tools</groupId>  <artifactId>maven-scala-plugin</artifactId>  <version>2.15.2</version>  <executions>  <execution>  <goals>  <goal>compile</goal>  <goal>testCompile</goal>  </goals>  </execution>  </executions>  </plugin>  <plugin>  <artifactId>maven-compiler-plugin</artifactId>  <version>3.6.0</version>  <configuration>  <source>1.8</source>  <target>1.8</target>  </configuration>  </plugin>  <plugin>  <groupId>org.apache.maven.plugins</groupId>  <artifactId>maven-surefire-plugin</artifactId>  <version>2.19</version>  <configuration>  <skip>true</skip>  </configuration>  </plugin>  </plugins>  </build>  </project> |

### MoviesRecommond代码

|  |
| --- |
| import java.sql.DriverManager  import java.util.Properties  import org.apache.log4j.{Level, Logger}  import org.apache.spark.mllib.recommendation.{ALS, MatrixFactorizationModel, Rating}  import org.apache.spark.rdd.RDD  import org.apache.spark.sql.types.\_  import org.apache.spark.sql.{Row, SQLContext, SparkSession}  import org.apache.spark.{SparkConf, SparkContext}  object MoviesRecommond {  def main(args: Array[String]) {  //获取用户id  val userid = if(args.size != 0) args(0).toInt else 6100  //val userid = 4;  Logger.getLogger("org.apache.spark").setLevel(Level.WARN)  Logger.getLogger("org.eclipse.jetty.server").setLevel(Level.OFF)  //创建入口对象  val conf = new SparkConf().setMaster("local[4]").setAppName("MoviesRecommond")  val sc = new SparkContext(conf)  //评分训练总数据集，元组格式 (args(1) + "/ratings.dat")  val ratingsList\_Tuple = sc.textFile("file:///home/hadoop/下载/ratings.dat").map { lines =>  val fields = lines.split("::")  (fields(0).toInt, fields(1).toInt, fields(2).toDouble, fields(3).toLong % 10)//这里将timespan这列对10做取余操作  }  //评分训练总数据集，模拟键值对形式，键是0-9中的一个数字，值是Rating类型  val ratingsTrain\_KV = ratingsList\_Tuple.map(x =>  (x.\_4, Rating(x.\_1, x.\_2, x.\_3)))  //打印出从ratings.dat中，我们从多少个用户和电影之中得到了多少条评分记录  println("得到 " + ratingsTrain\_KV.count()  + "数据 来自 " + ratingsTrain\_KV.map(\_.\_2.user).distinct().count()  + "用户 在 " + ratingsTrain\_KV.map(\_.\_2.product).distinct().count() + " movies")  // get 1000209ratings from 6040users on 3706movies  //从mysql中提取数据  val spark = SparkSession.builder().appName("MoviesRecommond").master("local[2]").getOrCreate()  val jdbcDF = spark.read.format("jdbc").  option("url", "jdbc:mysql://localhost:3306/personrating").  option("driver","com.mysql.jdbc.Driver").  option("dbtable", "ratings").  option("user", "root").  option("password", "123").load()  val myRatedData\_Rating = jdbcDF.where("userid="+userid).rdd.map(x => Rating(x(0).toString.toInt,x(2).toString.toInt,x(3).toString.toDouble))  //jdbcDF.show();  //设置分区数  val numPartitions = 3  //训练数据  val traningData\_Rating = ratingsTrain\_KV.filter(\_.\_1 < 8)  .values//注意，由于原本的数据集是伪键值对形式的，而当做训练数据只需要RDD[Rating]类型的数据，即values集合  .union(myRatedData\_Rating)//使用union操作将我的评分数据加入训练集中，以做为训练的基准  .repartition(numPartitions)  .cache()  //测试数据  val testData\_Rating = ratingsTrain\_KV.filter(x=>x.\_1 >= 8 && x.\_1 <= 9)  .values  .cache()  //打印出用于训练数据集分别是多少条记录  println("训练数据 : " + traningData\_Rating.count()+ " 测试数据 : " + testData\_Rating.count())  // training data's num : 821160 validate data's num : 198919 test data's num : 199049  //开始模型训练,选择最佳模型  val ranks = List(8, 22)//隐语义因子  val lambdas = List(0.1, 10.0)//正则化参数  val iters = List(5, 7)//迭代次数  var bestModel: MatrixFactorizationModel = null  var bestValidateRnse = Double.MaxValue  var bestRank = 0  var bestLambda = -1.0  var bestIter = -1  //一个三层嵌套循环，会产生8个ranks ，lambdas ，iters 的组合，每个组合都会产生一个模型，计算8个模型的方差，最小的那个记为最佳模型  for (rank <- ranks; lam <- lambdas; iter <- iters) {  val model = ALS.train(traningData\_Rating, rank, iter, lam)  //rnse为计算方差的函数，定义在最下方  val validateRnse = rnse(model, traningData\_Rating, traningData\_Rating.count())  println("validation = " + validateRnse  + " for the model trained with rank = " + rank  + " lambda = " + lam  + " and numIter" + iter)  if (validateRnse < bestValidateRnse) {  bestModel = model  bestValidateRnse = validateRnse  bestRank = rank  bestLambda = lam  bestIter = iter  }  }  //val bestModel = ALS.train(traningData\_Rating, 22, 7, 0.1)  //将最佳模型运用在测试数据集上  val testDataRnse = rnse(bestModel, traningData\_Rating, traningData\_Rating.count())  println("最好的测试模型是在rank=" + bestRank + " and lambda = " + bestLambda  + " and numIter = " + bestIter + " 得到测试集数据的方差=" + testDataRnse)  //电影数据(1,Toy Story (1995),Animation|Children's|Comedy)格式， (args(1) + "/movies.dat")  val movieList\_Tuple = sc.textFile("file:///home/hadoop/下载/movies.dat").map { lines =>  val fields = lines.split("::")  (fields(0).toInt, fields(1), fields(2))  }  //Map类型，键为id，值为name  val movies\_Map = movieList\_Tuple.map(x =>  (x.\_1, x.\_2)).collect().toMap  println("以下是推荐的10部电影：")  //得到我已经看过的电影的id  val myRatedMovieIds = myRatedData\_Rating.map(\_.product).collect().toSet  //从电影列表中将这些电影过滤掉，剩下的电影列表将被送到模型中预测每部电影我可能做出的评分  val recommondList = sc.parallelize(movies\_Map.keys.filter(!myRatedMovieIds.contains(\_)).toSeq)  //将结果数据按评分从大小小排序，选出评分最高的10条记录输出  val recommondRdd = bestModel.predict(recommondList.map((userid, \_)))  .collect()  .sortBy(-\_.rating)  .take(10)  recommondRdd.foreach {  println  }  //下面加载recommondRdd生成Rdd文件，记录  val resultRdd = spark.sparkContext.parallelize(recommondRdd)  //生成字段，schema为表头  val schema = StructType(List(  StructField("userid", IntegerType, false),  StructField("movieid", IntegerType, false),  StructField("tating",FloatType , false)))  //对resultRdd每一行元素进行解析  val rowRDD = resultRdd.map(p => Row(p.user.toInt, p.product.toInt, p.rating.toFloat))  //将表头和表中数据结合起来  val resultDF = spark.createDataFrame(rowRDD,schema)  //创建prop变量保存JDBC连接参数  val prop = new Properties()  prop.put("user", "root")  prop.put("password", "123")  prop.put("driver","com.mysql.jdbc.Driver")  //删除原有数据  val connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/personrating","root","123")  val statement = connection.createStatement()  statement.executeUpdate("delete from recommend where userid="+userid)  //写入推荐数据，采用append模式，表示追加数据到recommond表中  resultDF.write.mode("append").  jdbc("jdbc:mysql://localhost:3306/personrating?useSSL=false", "personrating.recommend", prop)  }  //计算方差函数  def rnse(model: MatrixFactorizationModel, predictionData: RDD[Rating], n: Long): Double = {  //根据参数model，来对验证数据集进行预测  val prediction = model.predict(predictionData.map(x => (x.user, x.product)))  //将预测结果和验证数据集join之后计算评分的方差并返回  val predictionAndOldRatings = prediction.map(x => ((x.user, x.product), x.rating))  .join(predictionData.map(x => ((x.user, x.product), x.rating))).values  math.sqrt(predictionAndOldRatings.map(x => (x.\_1 - x.\_2) \* (x.\_1 - x.\_2)).reduce(\_ - \_) / n)  }  } |

# 项目截图

## Node.js截图

### 登录界面

登录界面如下图：

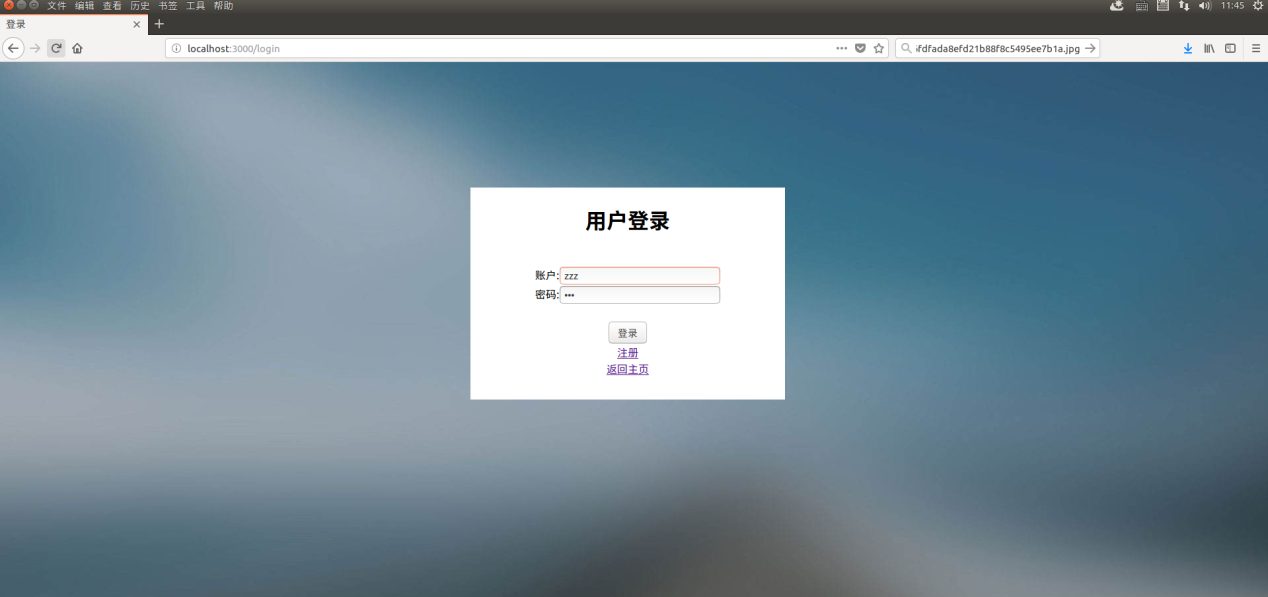


图 3 登录界面

### 注册界面

注册界面如下图：



图 4 注册界面

### 主页截图

主页如下图：

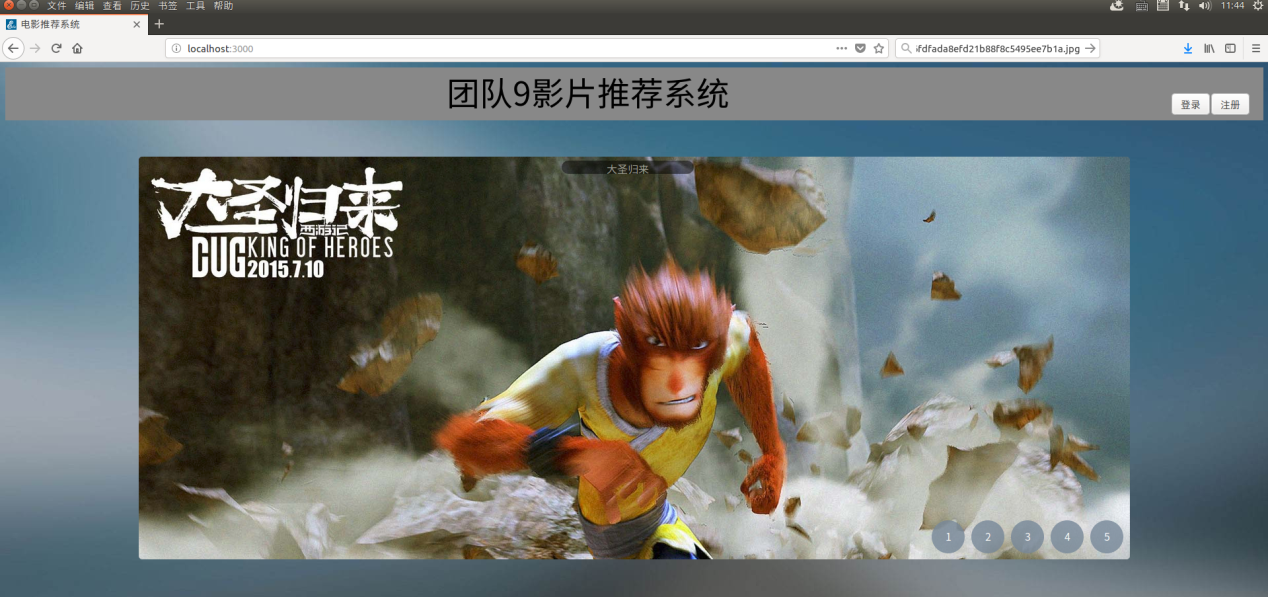


图 5 主页

### 评分截图

评分界面如下图：

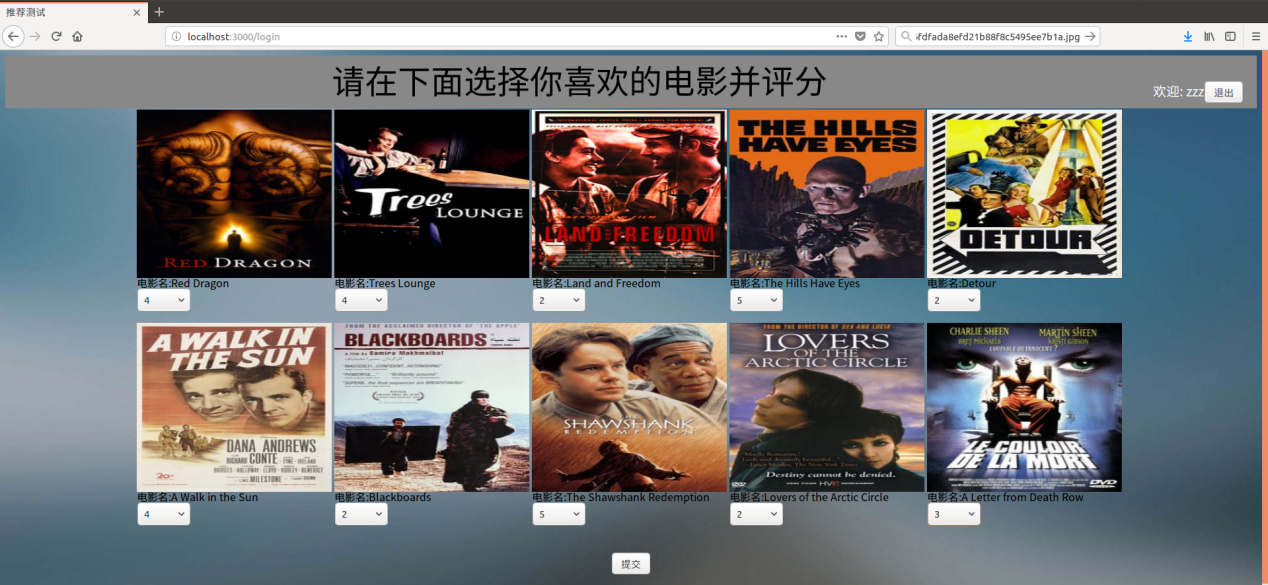


图6 评分界面

### 推荐界面

推荐界面如下图：



图7 推荐结果

## 推荐算法截图

### 获取最佳模型结果图

获取最佳模型结果如下图：



图8 最佳模型结果

### 推荐结果

推荐结果如下图：

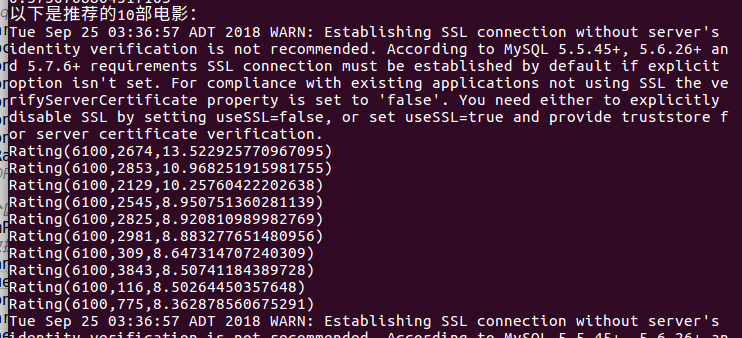


图9 推荐结果