

华中科技大学

课程实验报告

课程名称：Python大数据与人工智能实践

专业班级 CS2202

姓 名 王雨凝 陈梦琴 冯瑞琦

指导教师 周正勇 邹逸雄

报告日期 2023年 12 月 25日

计算机科学与技术学院

目 录


1 实验目的和总述	1
1.1 实验目的	1
1.2 基本内容	1
1.3 实验环境	1
2 数据集分类	2
2.1 数据来源	2
2.2 数据实例	2
3 系统设计与分析	10
3.1 程序总体设计	10
3.2 详细内容及代码	11
4 测试结果	17
4.1 数据集测试结果	17
4.2 电影和评分信息	17
4.3 缺失值	19
4.4 平均准确率	19
5 实验小结	20
5.1 总结与心得	20

1 实验目的和总述

1.1 实验目的

Movie Lens Dataset: <https://www.kaggle.com/datasets/aigamer/movie-lens-dataset>

- 9742部电影
- 100836个星级评分
- 610位用户
- 3683个标签
- 任务：根据电影属性，预测电影星级评分

 links.csv	XLS 工作表	86 KB	否	194 KB
 movies.csv	XLS 工作表	166 KB	否	483 KB
 ratings.csv	XLS 工作表	680 KB	否	2,426 KB
 tags.csv	XLS 工作表	36 KB	否	116 KB

1.2 基本内容

将9742部电影随机划分训练集(data_train)与验证集(data_val)，使用训练集训练模型，预测验证集电影的标签。

重复三次“训练集/验证集划分、训练、测试”，汇报三次的平均准确率

- 数据集划分：

```
from sklearn.model_selection import train_test_split  
data_train, data_val = train_test_split(data, test_size=0.1)
```

1.3 实验环境

操作系统：Windows 10

Python版本：3.12.0

开发工具：Pycharm

2、数据集分类

2.1 数据来源

<https://www.kaggle.com/datasets/aigamer/movie-lens-dataset>

2.2 数据实例

2.2.1 links数据集

该数据集包含电影的ID和其在互联网电影排行榜以及及库管理系统对应的ID(共9742条数据)。

movieId	imdbId	tmdbId
1	114709	862
2	113497	8844
3	113228	15602
4	114885	31357
5	113041	11862
6	113277	949
7	114319	11860
8	112302	45325
9	114576	9091
10	113189	710
11	112346	9087
12	112896	12110
13	112453	21032
14	113987	10858
15	112760	1408
16	112641	524
17	114388	4584
18	113101	5
19	112281	9273
20	113845	11517
21	113161	8012
22	112722	1710
23	112401	9691

图表 2.1.1 links数据1-23

24	114168	12665
25	113627	451
26	114057	16420
27	114011	9263
28	114117	17015
29	112682	902
30	115012	37557
31	112792	9909
32	114746	63
34	112431	9598
36	112818	687
38	113442	33689
39	112697	9603
40	112749	34615
41	114279	31174
42	112819	11443
43	114272	35196
44	113855	9312
45	114681	577
46	113347	11861
47	114369	807
48	114148	10530
49	114916	8391
50	114814	629
51	112810	11448

图表 2.1.2 links数据集24-50

华 中 科 技 大 学 课 程 实 验 报 告

183199	6149818	432987	189043	5686062	447682
183227	7808620	494368	189111	5359048	525662
183295	5726086	406563	189333	4912910	353081
183301	92046	31615	189381	7690670	500475
183317	95835	121870	189547	1665744	111196
183611	2704998	445571	189713	7349662	487558
183635	4500922	336843	190183	4073790	445651
183897	5104604	399174	190207	1680019	79159
183911	5461956	422615	190209	7620650	487541
183959	7379330	497520	190213	3977428	364002
184015	5783956	433310	190215	7293380	479871
184053	1016024	17571	190219	179011	48610
184245	78090	110775	190221	3333182	460631
184253	2548396	384521	191005	5805470	432985
184257	5189670	502892	193565	1636780	71172
184349	453047	22916	193567	2323836	255413
184471	1365519	338970	193571	3110014	297825
184641	5607028	426285	193573	3837248	333623
184721	6053438	458737	193579	5342766	360617
184791	7924798	502616	193581	5476944	432131
184931	1137450	395990	193583	5914996	445030
184987	1620680	407451	193585	6397426	479308
184997	5164432	449176	193587	8391976	483455
185029	6644200	447332	193609	101726	37891
185031	1211000	200260			

图表 2.1.3 links数据集183199-185029

图表 2.1.4 links数据集189043-193609

2.2.2 movies数据集

该数据集包含电影的ID，每个ID对应的电影名和体裁分类（如喜剧、恐怖、浪漫、动画等），共有9742条数据。

movieId	title	genres				
1	Toy Story	Adventure Animation Children Comedy Fantasy				
2	Jumanji (1995)	Adventure Children Fantasy				
3	Grumpier (Comedy)	Romance				
4	Waiting to	Comedy Drama Romance				
5	Father of	Comedy				
6	Heat (1995)	Action Crime Thriller				
7	Sabrina (1995)	Comedy Romance				
8	Tom and Huck	Adventure Children				
9	Sudden Death	Action				
10	GoldenEye	Action Adventure Thriller				
11	American History	Comedy Drama Romance				
12	Dracula: The Undead	Comedy Horror				
13	Balto (1995)	Adventure Animation Children				
14	Nixon (1994)	Drama				
15	Cutthroat	Action Adventure Romance				
16	Casino (1995)	Crime Drama				
17	Sense and Sensibility	Drama Romance				
18	Four Rooms	Comedy				
19	Ace Ventura: When Nature Calls	Comedy				
20	Money Train	Action Comedy Crime Drama Thriller				
21	Get Shorty	Comedy Crime Thriller				
22	Copycat (1995)	Crime Drama Horror Mystery Thriller				
23	Assassins	Action Crime Thriller				

图表 2.2.1 movies 1-23

25	Leaving L	Drama	Romance			
26	Othello (Drama				
27	Now and Th	Children	Drama			
28	Persuasior	Drama	Romance			
29	City of L	Adventure	Drama	Fantasy	Mystery	Sci-Fi
30	Shanghai 1	Crime	Drama			
31	Dangerous	Drama				
32	Twelve Mor	Mystery	Sci-Fi	Thriller		
34	Babe (199	Children	Drama			
36	Dead Man V	Crime	Drama			
38	It Takes 1	Children	Comedy			
39	Clueless	Comedy	Romance			
40	Cry, the I	Drama				
41	Richard II	Drama	War			
42	Dead Pres	Action	Crime	Drama		
43	Restoratio	Drama				
44	Mortal Kor	Action	Adventure	Fantasy		
45	To Die For	Comedy	Drama	Thriller		
46	How to Ma	Drama	Romance			
47	Seven (a.	Mystery	Thriller			
48	Pocahontas	Animation	Children	Drama	Musical	Romance
49	When Nigh	Drama	Romance			
50	Usual Sus	Crime	Mystery	Thriller		

图表 2.2.2 movies25-50

191005	Gintama (2	Action	Adventure	Comedy	Sci-Fi	
193565	Gintama: 1	Action	Animation	Comedy	Sci-Fi	
193567	anohana: 1	Animation	Drama			
193571	Silver Sp	Comedy	Drama			
193573	Love Live	Animation				
193579	Jon Stewa	Documentary				
193581	Black But	Action	Animation	Comedy	Fantasy	
193583	No Game N	Animation	Comedy	Fantasy		
193585	Flint (20	Drama				
193587	Bungo Str	Action	Animation			
193609	Andrew Dic	Comedy				

图表 2.2.3 movies 191005-193609

2.2.3 ratings数据集

该数据集包含用户ID、电影ID，用户给相应电影的评分和相应的时间戳，共有数据1000836条。

userId	movieId	rating	timestamp
1	1	4	964982703
1	3	4	964981247
1	6	4	964982224
1	47	5	964983815
1	50	5	964982931
1	70	3	964982400
1	101	5	964980868
1	110	4	964982176
1	151	5	964984041
1	157	5	964984100
1	163	5	964983650
1	216	5	964981208
1	223	3	964980985
1	231	5	964981179
1	235	4	964980908
1	260	5	964981680
1	296	3	964982967
1	316	3	964982310
1	333	5	964981179
1	349	4	964982563
1	356	4	964980962
1	362	5	964982588
1	367	4	964981710

图表 2.3.1 ratings userID1

2	318	3	1.446E+09
2	333	4	1.446E+09
2	1704	4.5	1.446E+09
2	3578	4	1.446E+09
2	6874	4	1.446E+09
2	8798	3.5	1.446E+09
2	46970	4	1.446E+09
2	48516	4	1.446E+09
2	58559	4.5	1.446E+09
2	60756	5	1.446E+09
2	68157	4.5	1.446E+09
2	71535	3	1.446E+09
2	74458	4	1.446E+09
2	77455	3	1.446E+09
2	79132	4	1.446E+09
2	80489	4.5	1.446E+09
2	80906	5	1.446E+09
2	86345	4	1.446E+09
2	89774	5	1.446E+09
2	91529	3.5	1.446E+09
2	91658	2.5	1.446E+09
2	99114	3.5	1.446E+09

图表 2.3.2 ratings userID2

华中科技大学课程实验报告

608	1917	3.5	1.118E+09
608	1918	3	1.118E+09
608	1921	4.5	1.117E+09
608	1923	3.5	1.117E+09
608	1953	4.5	1.118E+09
608	1954	4.5	1.118E+09
608	1961	4.5	1.117E+09
608	1968	4	1.117E+09
608	1991	3	1.118E+09
608	1994	3	1.118E+09
608	1997	4.5	1.118E+09
608	2000	3.5	1.118E+09
608	2001	3.5	1.117E+09
608	2002	3	1.118E+09
608	2003	2.5	1.117E+09
608	2004	2	1.118E+09
608	2006	2.5	1.117E+09
608	2009	3	1.118E+09
608	2011	2.5	1.118E+09
608	2012	2.5	1.118E+09
608	2021	3.5	1.118E+09
608	2023	4.5	1.189E+09
608	2028	4.5	1.117E+09
608	2042	1.5	1.118E+09

图表 2.3.3 ratings userID608

610	156726	4.5	1.494E+09
610	157296	4	1.494E+09
610	158238	5	1.48E+09
610	158721	3.5	1.48E+09
610	158872	3.5	1.494E+09
610	158956	3	1.494E+09
610	159093	3	1.494E+09
610	160080	3	1.494E+09
610	160341	2.5	1.48E+09
610	160527	4.5	1.48E+09
610	160571	3	1.494E+09
610	160836	3	1.494E+09
610	161582	4	1.494E+09
610	161634	4	1.494E+09
610	162350	3.5	1.494E+09
610	163937	3.5	1.494E+09
610	163981	3.5	1.494E+09
610	164179	5	1.494E+09
610	166528	4	1.494E+09
610	166534	4	1.494E+09
610	168248	5	1.494E+09
610	168250	5	1.494E+09
610	168252	5	1.494E+09
610	170875	3	1.494E+09

图表 2.3.4 ratings userID610

380	6936	3	1.494E+09
380	6942	4	1.494E+09
380	6946	4	1.494E+09
380	6947	5	1.494E+09
380	6952	3	1.494E+09
380	6957	4	1.494E+09
380	6958	2	1.494E+09
380	6979	3	1.495E+09
380	7022	5	1.495E+09
380	7044	5	1.495E+09
380	7076	3	1.508E+09
380	7099	5	1.495E+09

图表 2.3.5 ratings userID380

2.2.4 tags数据集

该数据集包括用户ID、电影ID、用户给相应电影打的标签名及对应时间戳，共有数据3683条。

userId	movieId	tag	timestamp				
2	60756	funny	1.446E+09	474	7382	adolescence	1.138E+09
2	60756	Highly quot	1.446E+09	474	7382	crime	1.138E+09
2	60756	will ferre	1.446E+09	474	7438	revenge	1.137E+09
2	89774	Boxing sto	1.446E+09	474	7440	Holocaust	1.142E+09
2	89774	MMA	1.446E+09	474	7451	High Schoo	1.137E+09
2	89774	Tom Hardy	1.446E+09	474	7479	World War	1.138E+09
2	106782	drugs	1.446E+09	474	7493	multiple p	1.138E+09
2	106782	Leonardo I	1.446E+09	474	7572	cancer	1.137E+09
2	106782	Martin Sco	1.446E+09	474	7584	Hepburn ar	1.138E+09
7	48516	way too lo	1.17E+09	474	7584	marriage	1.138E+09
18	431	Al Pacino	1.462E+09	474	7614	Rogers and	1.138E+09
18	431	gangster	1.462E+09	474	7618	biopic	1.137E+09
18	431	mafia	1.462E+09	474	7618	In Netfli	1.137E+09
18	1221	Al Pacino	1.462E+09	474	7619	blindness	1.138E+09
18	1221	Mafia	1.462E+09	474	7619	deaf	1.138E+09
18	5995	holocaust	1.456E+09	474	7646	Stephen K	1.137E+09
18	5995	true story	1.456E+09	474	7649	space stat	1.138E+09
18	44665	twist end	1.457E+09	474	7700	In Netfli	1.137E+09
18	52604	Anthony Ho	1.458E+09	474	7705	Hepburn ar	1.138E+09
18	52604	courtroom	1.458E+09	474	7705	sports	1.138E+09
18	52604	twist end	1.458E+09	474	7713	sexuality	1.138E+09
18	88094	britpop	1.457E+09	474	7714	King Arthu	1.138E+09
18	88094	indie rec	1.457E+09	474	7728	adultery	1.138E+09

图表 2.4.1 tags userId2、7、18

图表 2.4.2 tags userId474

599	296	gore	1.498E+09
599	296	great acti	1.498E+09
599	296	great dial	1.498E+09
599	296	great sour	1.498E+09
599	296	gritty	1.498E+09
599	296	guns	1.498E+09
599	296	Harvey Ke	1.498E+09
599	296	heroin	1.498E+09
599	296	Highly qu	1.498E+09
599	296	hit men	1.498E+09

图表 2.4.3 tags userId 599

600	273	gothic	1.238E+09
606	1357	music	1.177E+09
606	1948	British	1.178E+09
606	3578	Romans	1.173E+09
606	5694	70mm	1.176E+09
606	6107	World War	1.178E+09
606	7382	for katie	1.171E+09
606	7936	austere	1.173E+09
610	3265	gun fu	1.494E+09
610	3265	heroic blo	1.494E+09
610	168248	Heroic Blo	1.494E+09

图表 2.4.4 tags userId600、606、610

3 系统分析与设计

3.1 程序总体设计

首先导入必要的库，然后使用Pandas读取四个CSV文件（ratings.csv, movies.csv, tags.csv, links.csv），并将它们存储为Pandas的DataFrame。随后，通过一些可视化和数据分析操作对电影数据集进行处理，步骤如下。

设置Pandas的显示选项，以支持中文字符的显示。将数据集文件加载到DataFrame中。显示每个DataFrame的前几行，以了解数据的结构。使用matplotlib和seaborn库进行可视化操作：绘制电影平均评分分布的直方图；绘制评分次数分布的直方图（使用对数尺度）；分析电影类型，并创建电影类型分布的条形图；检查每个数据集中是否存在缺失值。处理电影评分数据集中的多个用户评分问题，选择保留平均评分。检查电影数据集中是否存在重复行，导入额外的库（sklearn, numpy, MultiLabelBinarizer, train_test_split, mean_squared_error, r2_score）用于构建回归模型。配置随机森林回归器的参数。使用MultiLabelBinarizer将电影类型转换为二进制列。创建包含电影特征和评分的新DataFrame。准备数据集用于模型训练，将特征（X）和目标变量（y）分开。训练随机森林模型，预测验证集上的评分，并计算均方误差（MSE）和R2评分。打印平均MSE和R2评分。

设计流程大致如下图。

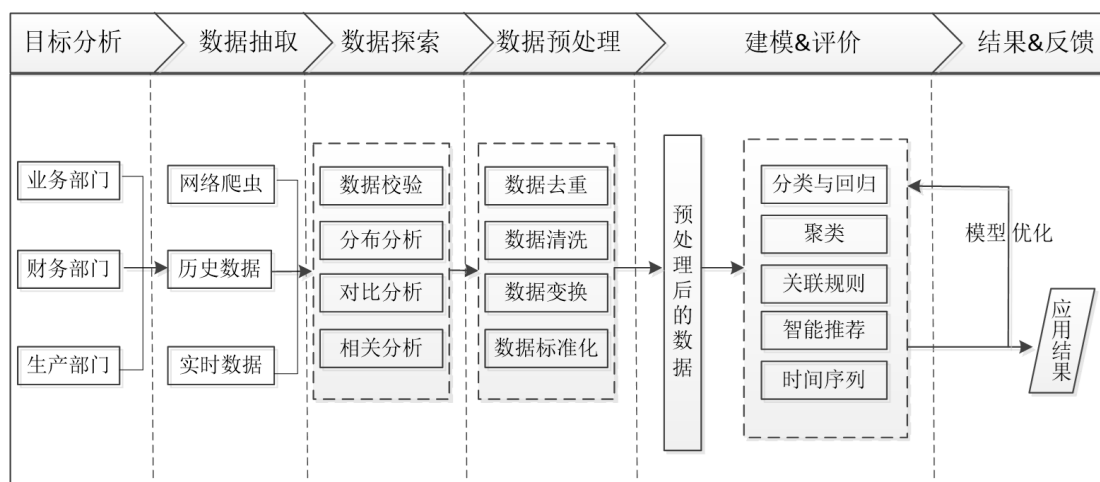


图 3.1 设计流程

3.2 详细内容及代码

```
import pandas as pd
```

```
# 设置全局字体为微软雅黑, 以支持中文显示
```

```
pd.set_option('display.max_columns', None)
```

```
pd.set_option('display.expand_frame_repr', False)
```

```
pd.set_option('display.unicode.east_asian_width', True)
```

```
# 读取数据集
```

```
ratings_path = 'ratings.csv'
```

```
movies_path = 'movies.csv'
```

```
tags_path = 'tags.csv'
```

```
links_path = 'links.csv'
```

```
# 加载数据集到DataFrame
```

```
ratings_df = pd.read_csv(ratings_path)
```

```
movies_df = pd.read_csv(movies_path)
```

```
tags_df = pd.read_csv(tags_path)
```

```
links_df = pd.read_csv(links_path)
```

```
# 显示数据集的前几行以了解其结构
```

```
display_data = {
```

```
    'Ratings': ratings_df.head(),
```

```
    'Movies': movies_df.head(),
```

```
    'Tags': tags_df.head(),
```

```
    'Links': links_df.head()
```

```
}
```

```
display_data
import matplotlib.pyplot as plt
import seaborn as sns

# 设置绘图风格和字体, 以便于显示中文
sns.set(style="whitegrid", font='Microsoft YaHei')
plt.rcParams['font.sans-serif'] = ['Microsoft YaHei']
plt.rcParams['axes.unicode_minus'] = False

# 创建一个包含电影评分平均值和评分次数的DataFrame
movie_ratings = ratings_df.groupby('movieId').agg({'rating':
['mean', 'count']})
movie_ratings.columns = movie_ratings.columns.droplevel(0)
movie_ratings.columns = ['average_rating',
'number_of_ratings']

# 合并电影信息和评分信息
movie_data_merged = pd.merge(movies_df, movie_ratings,
on='movieId')

# 查看电影平均评分分布
plt.figure(figsize=(10, 6))
sns.histplot(movie_data_merged['average_rating'], kde=True,
bins=30)
plt.title('电影平均评分分布')
plt.xlabel('平均评分')
plt.ylabel('电影数量')
plt.show()
```

查看评分次数分布

```
plt.figure(figsize=(10, 6))
sns.histplot(movie_data_merged['number_of_ratings'],
kde=False, bins=30)
plt.title('电影评分次数分布')
plt.xlabel('评分次数')
plt.ylabel('电影数量')
plt.xscale('log') # 由于评分次数差异很大，使用对数尺度
plt.show()
```

电影类型分析：将genres列分割成单独的类型，并统计每种类型的电影数量

```
genres_df = movies_df['genres'].str.get_dummies(sep='|')
movie_genres = genres_df.sum().sort_values(ascending=False)
```

可视化电影类型分布

```
plt.figure(figsize=(12, 6))
movie_genres.plot(kind='bar')
plt.title('电影类型分布')
plt.xlabel('类型')
plt.ylabel('电影数量')
plt.xticks(rotation=45)
plt.show()
```

检查缺失值

```
missing_values = {
    'Ratings': ratings_df.isnull().sum(),
    'Movies': movies_df.isnull().sum(),
    'Tags': tags_df.isnull().sum(),
```

```
'Links': links_df.isnull().sum()
}

# 处理电影评分数据集中的多用户评分问题：这里我们可以选择保留平均评分
# 对于每部电影，我们已经计算了平均评分和评分次数，所以可以忽略单个用户的评分细节

# 检查电影数据集中的重复项
duplicates_movies = movies_df.duplicated().sum()

# 汇总缺失值和重复项的信息
missing_values, duplicates_movies
from sklearn.ensemble import RandomForestRegressor
import numpy as np # 用于数值计算
from sklearn.preprocessing import MultiLabelBinarizer
from sklearn.model_selection import train_test_split # 用于划分数据集为训练集和验证集
from sklearn.metrics import mean_squared_error, r2_score # 用于计算评分

# 配置随机森林参数
rf_params = {
    'n_estimators': 100, # 决策树的数量
    'max_depth': None, # 树的最大深度
    'min_samples_split': 2, # 分裂内部节点所需的最小样本数
    'random_state': 42 # 随机数生成器的种子
}
```


将电影类型转换为多个二进制列

```
mlb = MultiLabelBinarizer()
genres_mlb =
mlb.fit_transform(movies_df['genres'].str.split('|'))
```

创建新的电影特征DataFrame

```
movie_features_df = pd.DataFrame(genres_mlb,
columns=mlb.classes_, index=movies_df.index)
```

合并电影特征和评分数据

```
full_data = pd.merge(ratings_df, movie_features_df,
left_on='movieId', right_index=True)
```

准备用于模型训练的数据

```
X = full_data.drop(columns=['userId', 'movieId', 'rating',
'timestamp'])
y = full_data['rating']
```

重新训练模型并计算评分

```
mse_scores_rf = []
r2_scores_rf = []
for _ in range(3):
    X_train, X_val, y_train, y_val = train_test_split(X, y,
test_size=0.1)
```

训练随机森林模型

```
rf_model = RandomForestRegressor(**rf_params)
rf_model.fit(X_train, y_train)
```

预测验证集

```
predictions_rf = rf_model.predict(X_val)
```

计算评分

```
mse_scores_rf.append(mean_squared_error(y_val,
predictions_rf))
r2_scores_rf.append(r2_score(y_val, predictions_rf))
```

计算平均MSE和R2

```
average_mse_rf = np.mean(mse_scores_rf)
average_r2_rf = np.mean(r2_scores_rf)

print(average_mse_rf, average_r2_rf)
```

4 测试结果

4.1 数据集测试结果

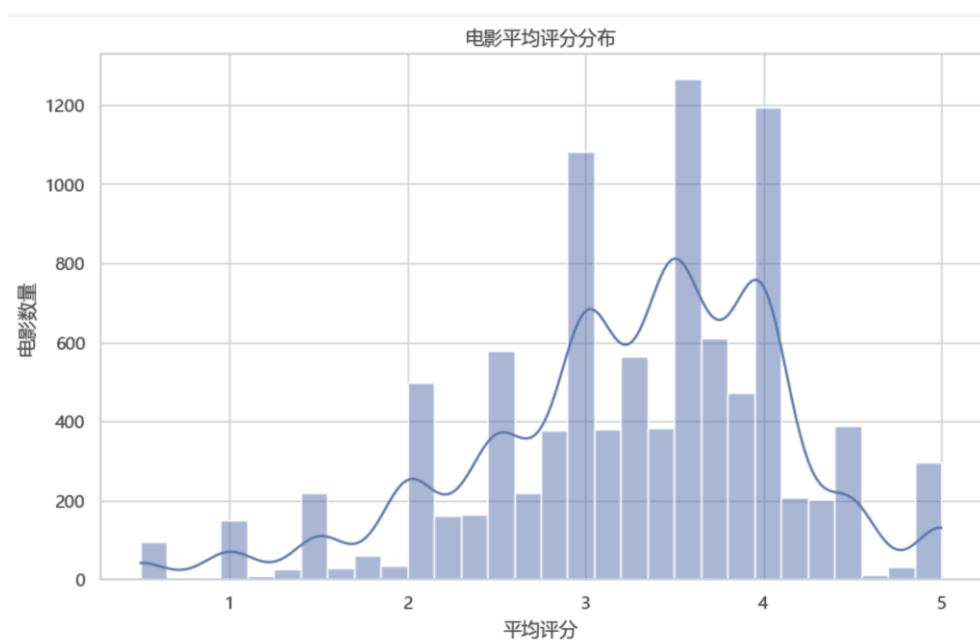
读取和加载数据集测试结果，得到结果如下图。

{ 'Ratings':					userId	movieId	rating	timestamp	
0	1	1	4.0	964982703					
1	1	3	4.0	964981247					
2	1	6	4.0	964982224					
3	1	47	5.0	964983815					
4	1	50	5.0	964982931,					
'Movies':					movieId	title			genres
0	1		Toy Story (1995)		Adventure Animation Children Comedy Fantasy				
1	2		Jumanji (1995)		Adventure Children Fantasy				
2	3		Grumpier Old Men (1995)		Comedy Romance				
3	4		Waiting to Exhale (1995)		Comedy Drama Romance				
4	5	Father of the Bride Part II (1995)		Comedy,					
'Tags':					userId	movieId	tag	timestamp	
0	2	60756	funny		1445714994				
1	2	60756	Highly quotable		1445714996				
2	2	60756	will ferrell		1445714992				
3	2	89774	Boxing story		1445715207				
4	2	89774	MMA		1445715200,				
'Links':					movieId	imdbId	tmdbId		
0	1	114709	862.0						
1	2	113497	8844.0						
2	3	113228	15602.0						
3	4	114885	31357.0						
4	5	113041	11862.0}						

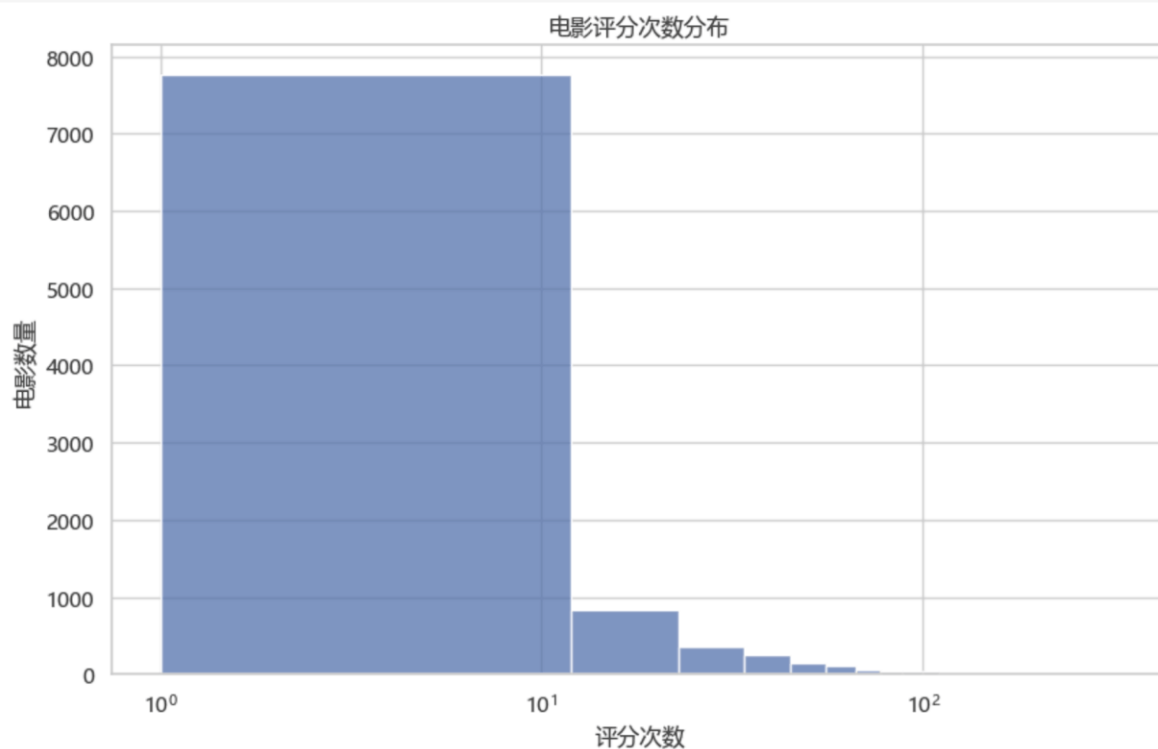
4.2 电影和评分信息

合并电影信息和评分信息，分别得到电影平均评分分布、电影评分次数分布、电影类型分布，如下图所示。

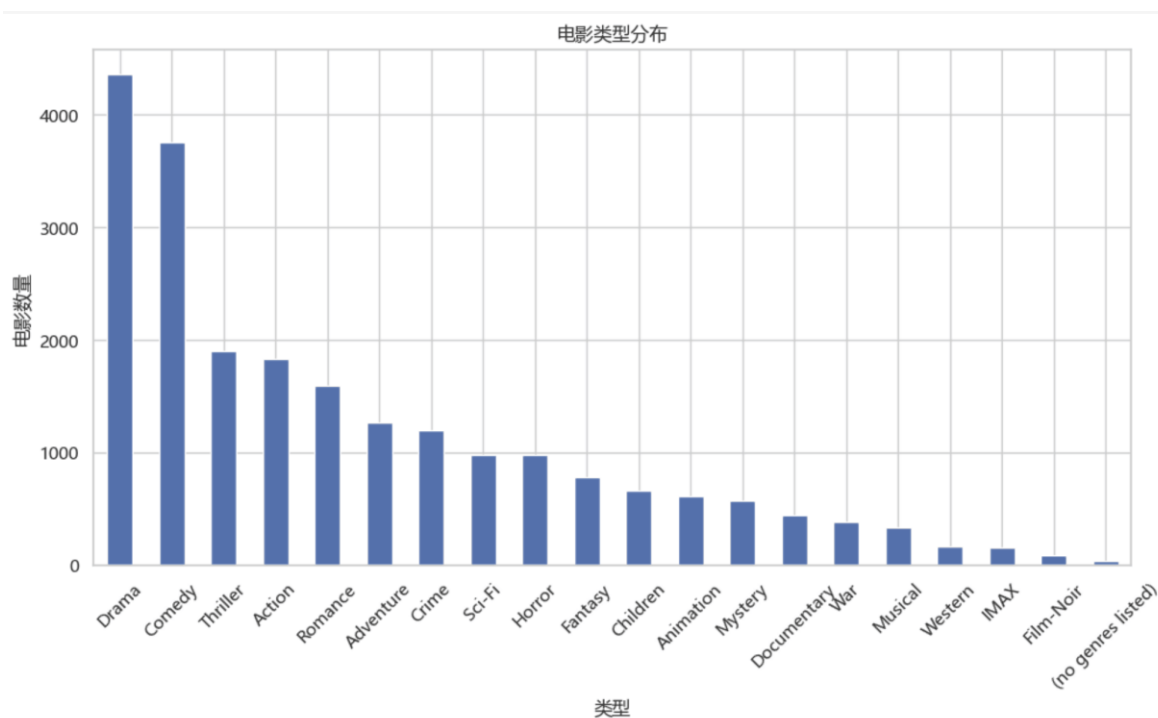
4.2.1 电影平均评分分布



4.2.2 电影评分次数分布



4.2.3 电影类型分布



4.3 缺失值

检查缺失值，得到结果如下图。

```
{'Ratings': userId      0
  movieId      0
  rating       0
  timestamp    0
  dtype: int64,
'Movies': movieId      0
  title        0
  genres       0
  dtype: int64,
'Tags': userId      0
  movieId      0
  tag          0
  timestamp    0
  dtype: int64,
'Links': movieId      0
  imdbId       0
  tmdbId       8
  dtype: int64},
0)
```

4.4 平均准确率

计算三次平均MSE和R2，得到结果如下图。

```
1.0556046399072636 0.03510834943552099
```

5 实验总结

5.1 总结与心得

王雨凝：

这次实验让我学会了问题定义和需求分析，在开始编写代码之前，确保清晰地定义了并分析了需求。明确你的目标是什么，需要实现什么功能，对用户来说哪些是重要的。

同时，选择适当的数据结构对于电影评分系统至关重要。你可能需要使用字典、列表或其他数据结构来存储电影信息、用户评分等。了解每种数据结构的优势和限制，并选择最适合你需求的一种。并且在用户输入、文件读写等可能出现问题的地方添加适当的异常处理机制。这可以增加代码的健壮性，确保程序在面对不同情况时不会崩溃。

最后，为你的代码添加清晰的注释和文档。这有助于其他人理解你的代码，也有助于你自己在一段时间后重新审视代码时快速理解每个部分的功能。

陈梦琴：

本次实验是我第一次以python作为编程语言，使用python库实现具有一定具体功能的程序的实验，也让我第一次接触人工智能和机器学习算法。

在本次实验中，我对人工智能领域有了初次的探索，在实验期间，逐一克服各种问题让我对大数据处理、代码是一行行编译的，报告是一行行撰写的，人工智能领域的学习从来没有捷径，只有不断试错，不断优化，才有最终测试结果的喜悦。

此外，将代码划分为小的、独立的模块，每个模块负责特定的任务。这样的设计能够提高代码的可读性和可维护性。确保每个函数或类都只关注一项任务，遵循单一职责原则。

希望我在未来的学习中也能像执行深度学习框架步骤一样，在损失中成长，在优化中磨砺！

冯瑞琦：

编写第一次用于电影评分的Python代码是一个充满挑战和收获的经历。

华中科技大学课程实验报告

以下是我从中得到的几点深刻的总结：首先，了解需求是任何软件开发的基础。在开始编码之前，我花了大量时间研究和明确电影评分系统的各项功能和特性。这不仅帮助我建立起明确的开发目标，还有助于我在后续的设计和实现过程中避免走弯路。

其次，选择合适的数据结构和算法是成功实现代码的关键。考虑到电影评分系统涉及到用户、电影、评分等多方面的数据，我决定使用字典来存储电影信息，并通过列表或集合来管理用户评分。在处理这些数据时，我还学会了如何利用 Python 的内置函数和库来实现高效的数据操作和处理。在用户交互方面，我认识到设计一个简洁、直观的界面是至关重要的。为了提供良好的用户体验，我研究了一些用户界面设计的基本原则，并尝试将其应用到我的代码中。