

期中大作业讲评与交流

—MovieLens数据集

王瑞环

作业目标

- 对所学Python和数据分析内容的综合演练
- Python部分
 - 常见的(矩阵)数据类型及运算: numpy、pandas
 - 图像数据的处理: OpenCV、PIL、img2vec
 - 机器学习方法: scikit-learn、(pytorch)
 - 可视化: matplotlib、seaborn
- 数据科学部分
 - 特征工程: 特征设计、特征降维、特征提取、特征融合...
 - 数据分析算法: 预测、聚类、分类、模型训练与测试...
 - 数据分析整体流程: 任务界定、数据观察、特征选择、模型选择、算法应用、结果分析...

Task 1.1 偏好分析

- 基本要求
 - 合理的衡量偏好程度的指标
 - 根据指标衡量并筛选出不同类型用户的前10个电影并进行展示
- 主要考察点
 - pandas基础使用
 - *对偏好的定义： 定义是否符合直觉、结果是否符合直觉
 - 筛选方案设计

Task 1.1 偏好分析

- Ref. 曾为帅
- 对于群体偏好的两种范式
 - 给定一个对象看不同群体的偏好程度 (群体间-相对偏好)
 - 给定一个群体看对于不同对象的偏好程度 (群体内-绝对偏好)
 - m_1 : 评分的均值, 即一个年龄段的人对于一部电影的评分的均值。评分的均值越高, 相应的偏好程度也越大。
 - m_2 : 整体评分的方差, 即一个年龄段的人对于一部电影评分的方差。评分的方差反映了评分的一致性, 一致性越高, 对应的方差也就越小。所以在实际情况中, 我们考虑的是将方差 cov 归一化后考虑 $1-\text{cov}$ 作为 m_2
 - m_3 : 高评分的数量占总评分数量的比例, 即对于一部电影打分为 4 分和 5 分的人数占总人数的比例。该比例越大, 表示该年龄段的人偏好程度越高。

Task 1.1 偏好分析

- Ref. 周宇亮
- 何为“偏好”：某类观众对该电影的整体评价减去其中该电影本身的客观评分所造成的影响

那么如何定义评价函数？首先考察某类观众对某电影的打分遵循什么样的分布。粗略地来看，最主要的成分肯定是以平均评分作为峰值的正态分布，其权重为某一“置信概率”，其余部分期望值我认为是电影本身的平均评分。为了方便计算，将各电影所有评分减去平均评分，那么评价函数的形式就可以简单地写为“置信系数 * 均值”的形式，置信系数正比于正态分布峰值附近小范围内的概率。根据概率论，峰值处概率密度与标准差成反比，而多次取样得到的平均值相对真值的标准差为该组数据的标准差除以根号（样本数量-1）因此我定义评价函数是 “ $(\text{平均评分} - \text{该电影平均评分}) * \text{sqrt}(\text{人数} - 1) / \text{标准差}$ ”。

Task 1.1 偏好分析

- Ref. 朱逸飞
- 偏好的描述
 - 不同类群的平均打分的差异 $preference_{ij} = rating_i - \max_{k \neq i}(rating_k)$
 - 该类群对该电影的平均打分
 - 二者的加权组合

$$score = (\alpha * preference + (1 - \alpha) * \log(ave_rating)) * \log(count)$$

Task 1.1 偏好分析

- Ref. 刘沛雨
- 一种通用的偏好程度度量

$$preference = difference \times representativeness$$

其中 `difference` 指与其它群体或与整体的差异程度，`representativeness` 则表示观看该类电影的属于特定群体的人在该群体中的代表性，即这些人的偏好能否代表其所属的整个群体的偏好。

在职业偏好分析的上下文中，对于职业 `i` 和电影 `j`，我们定义：

$$difference_{i,j} = mean_{i,j} - mean_{all,j}$$

$$representativeness_{i,j} = \frac{WatchingNumber_{i,j}}{WatchingNumber_{all,j}} + 2 \times \frac{WatchingNumber_{i,j}}{OccupationNumber_i}$$

Task 1.1 偏好分析

- Ref. 马天麟
- SVD分解+协同过滤预测评分+指数均值

Under 18			
	movie_id	Scores	title
0	3114	0.288399	Toy Story 2 (1999)
1	1	0.250837	Toy Story (1995)
2	588	0.242431	Aladdin (1992)
3	1517	0.235376	Austin Powers: International Man of Mystery (1...)
4	2572	0.232161	10 Things I Hate About You (1999)
5	2694	0.222750	Big Daddy (1999)
6	2683	0.220737	Austin Powers: The Spy Who Shagged Me (1999)
7	3793	0.213900	X-Men (2000)
8	3948	0.211556	Meet the Parents (2000)
9	104	0.207035	Happy Gilmore (1996)

Task 1.2 可视化

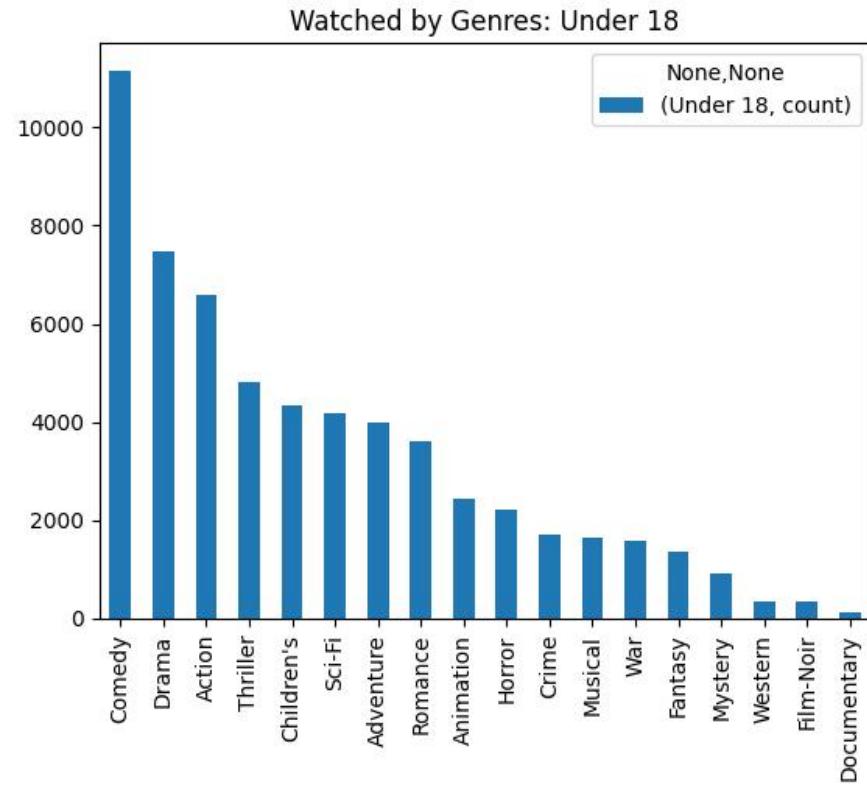
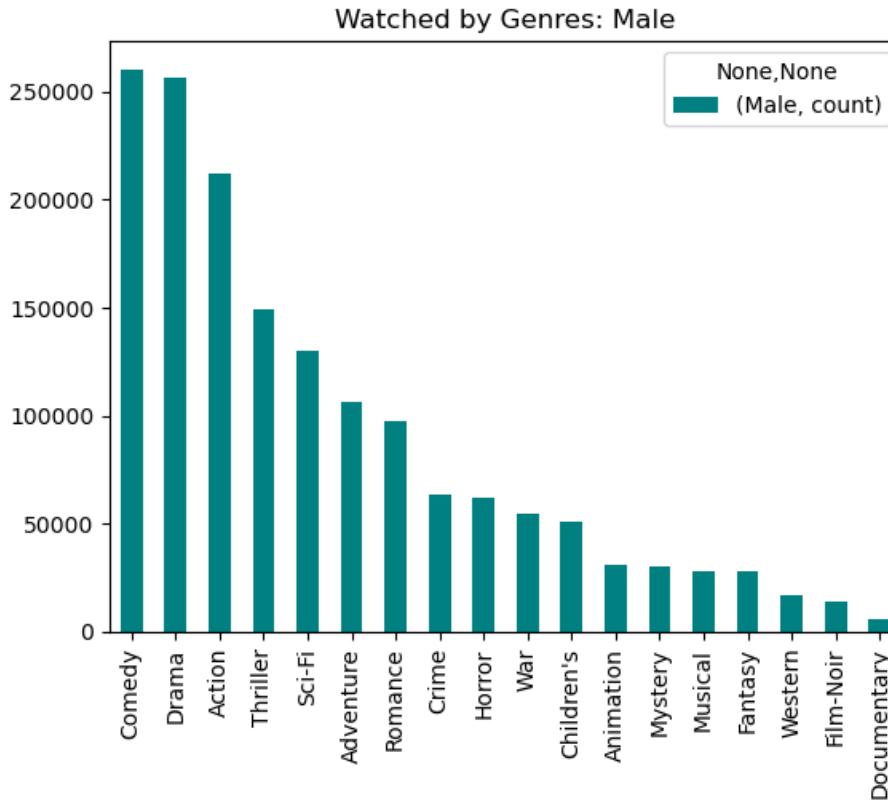
- 基本要求：基于电影风格的可视化
 - 不同类型用户对不同类型电影的评分的可视化
 - 不同类型用户对不同类型电影的观看数量的可视化
 -
- 主要考察点
 - Matplotlib基础使用
 - 电影类型的处理
 - *可视化指标与方法的选取
- 一点遗憾：几乎所有同学的可视化都是对示例代码的重复，没有更具创新性的可视化指标

Task 1.2 可视化

- 可视化为了什么?
 - 让人能够直观地理解复杂数据包含的信息
- 合理的可视化方法（人的角度）：能够让人快速地识别数据的分布、趋势、关联等
 - 如分布情况用饼图/柱形图；趋势用折线图；关联用热力图
- 合理的可视化指标（信息的角度）：被可视化的数据应充分包含与实际应用场景与任务目标一致的信息

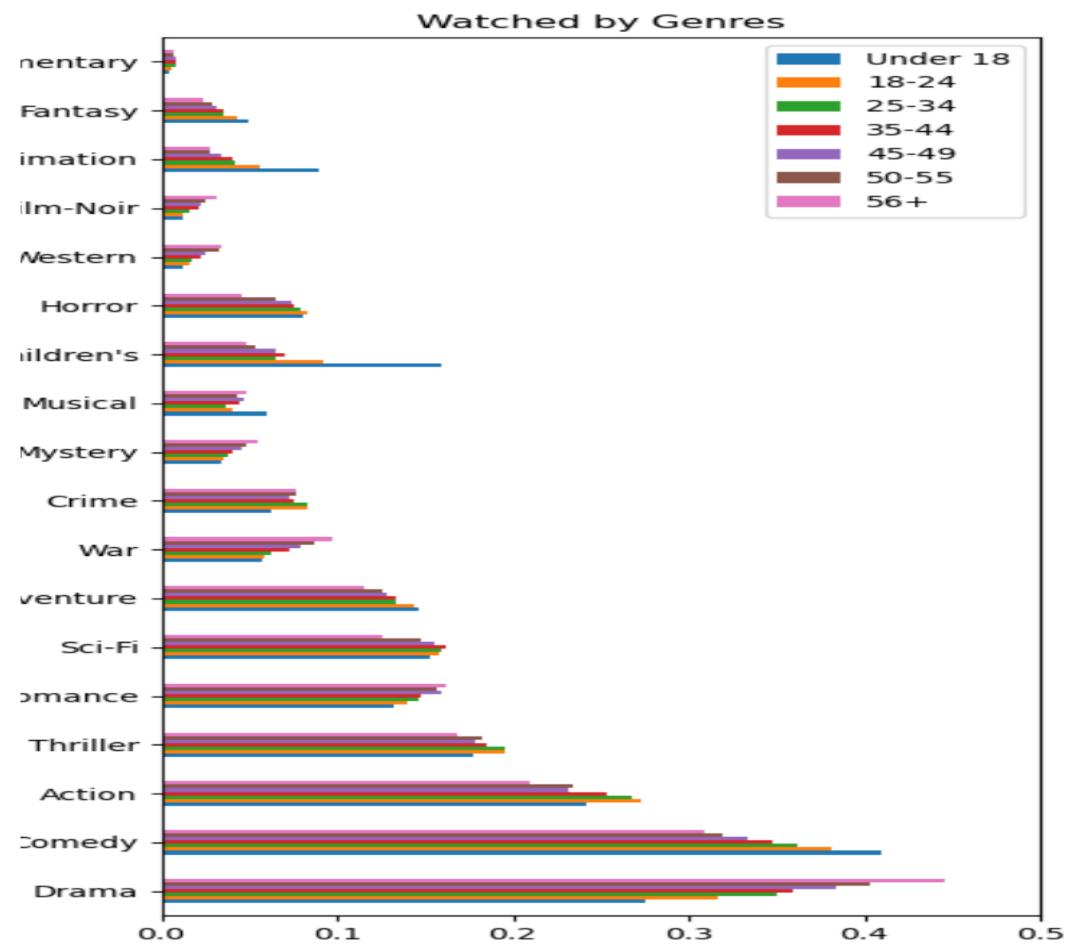
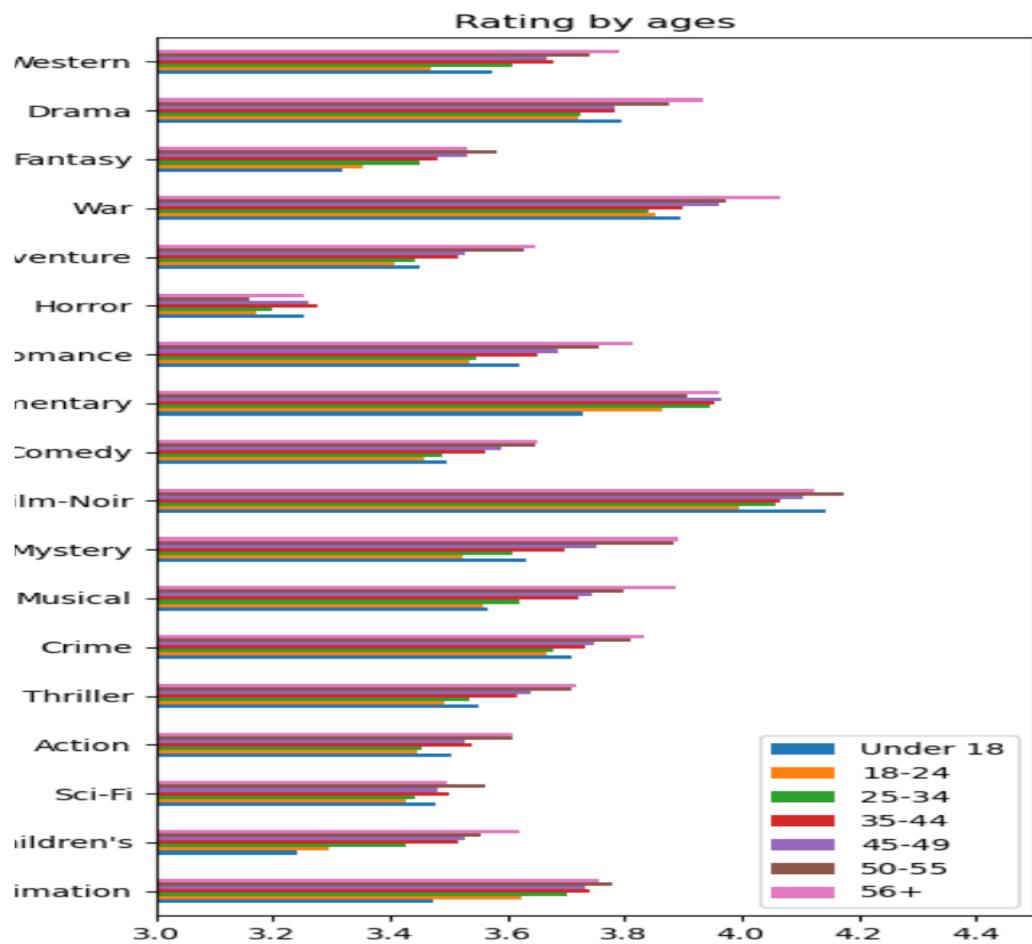
Task 1.2 可视化

- 观看次数 (是否是一个有助于我们理解数据的合理指标?)



Task 1.2 可视化

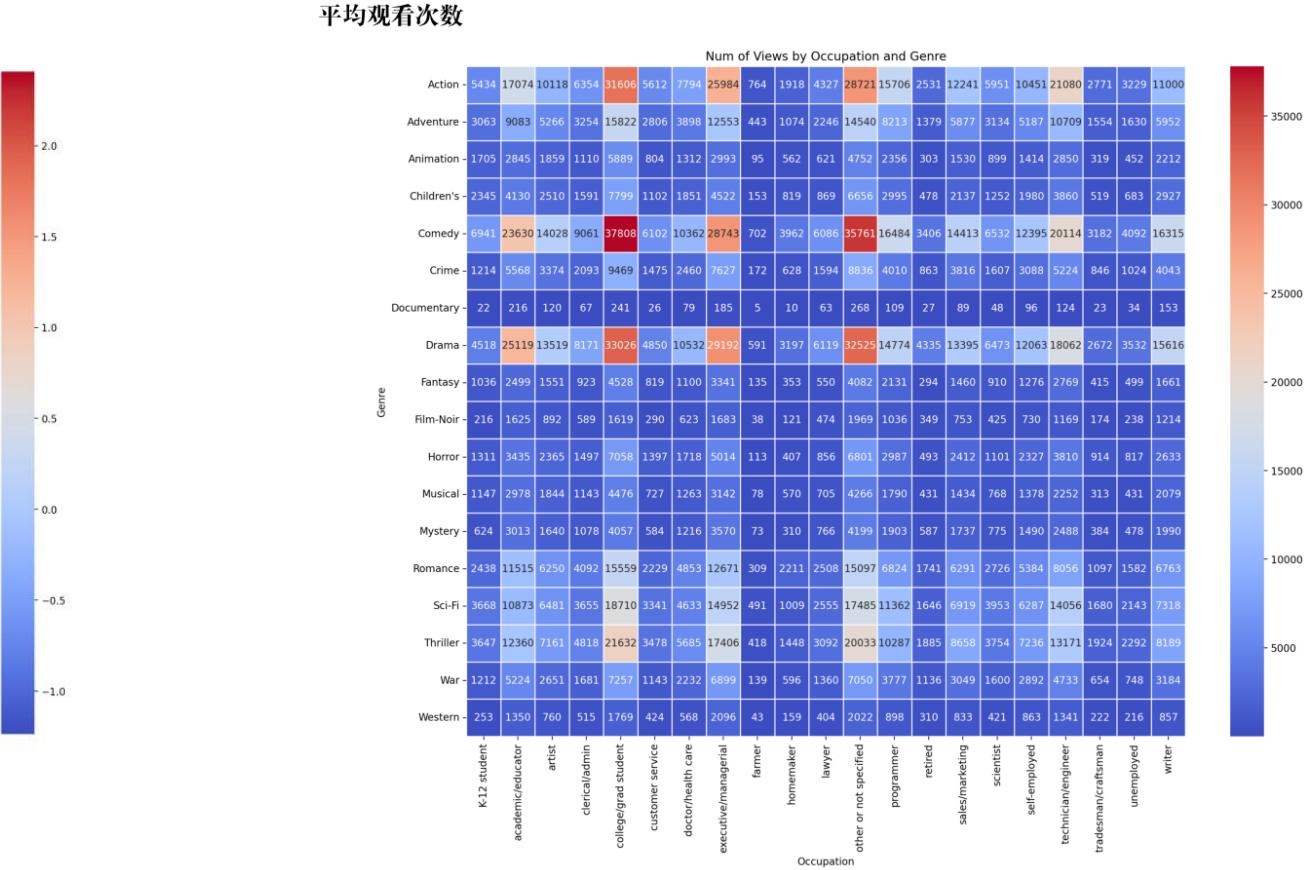
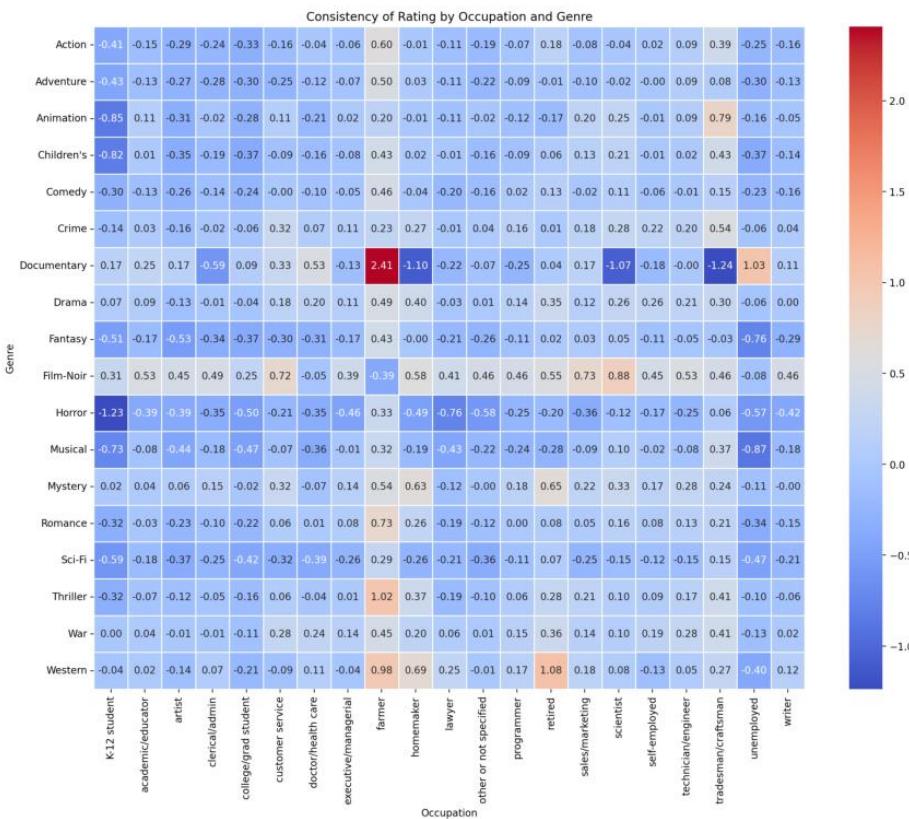
- 平均评分/观看比例



Task 1.2 可视化

- Ref. 王子宸

评分标准差 (Z-scaled)



Task 2 用户评分预测

- 基本要求
 - 合理提取用户和电影的特征，包括用户的性别、年龄、职业，电影的类型、国家、年份、简介、海报信息等
 - 对特征的融合、降维等处理
 - 使用机器学习模型进行训练、评估等
- 主要考察点
 - 特征的选取与使用
 - 问题建模与模型选择
 - —用户和电影特征如何结合？
 - —多标签如何处理？
 - —离散有序标签：分类？回归？

Task 2 用户评分预测

- 几个值得思考的小细节
- 用户年龄特征的提取：One-hot? 一个数字? 其他方式?
- 用户职业/电影类别特征：相似性/相关性是否能够纳入建模?
- 问题的建模：分类问题 or 回归问题?
 - 分类建模要考虑的问题：预测类别概率分布到最终值的转化、可拓展性、预测标签的序关系
 - 回归建模要考虑的问题：小于0和大于5的预测结果的处理

特征提取与应用分析

分享者:朱逸飞

Task1: 特征工程

```
genre_vector
0 [1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
1 [0, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
2 [0, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
3 [0, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
4 [0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
```

genres_Romance Thriller	genres_Romance War	genres_Romance Western	genres_Sci-Fi	genres_Sci-Fi Thriller	genres_Sci-Fi Thriller War
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
...
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0
0	0	0	0	0	0

特征提取与one-hot向量构造：

- 年龄、职业、性别、电影类型的提取：使用pd.merge函数对三张表快速合并即可。
- 对电影类型进行处理时，没有选择像第三Part中有监督聚类的示例办法一样，将复合型的genre拆开处理。而是保留了复合型的genre。例如:Romance|War、Romance|Western。

Task1: 特征工程

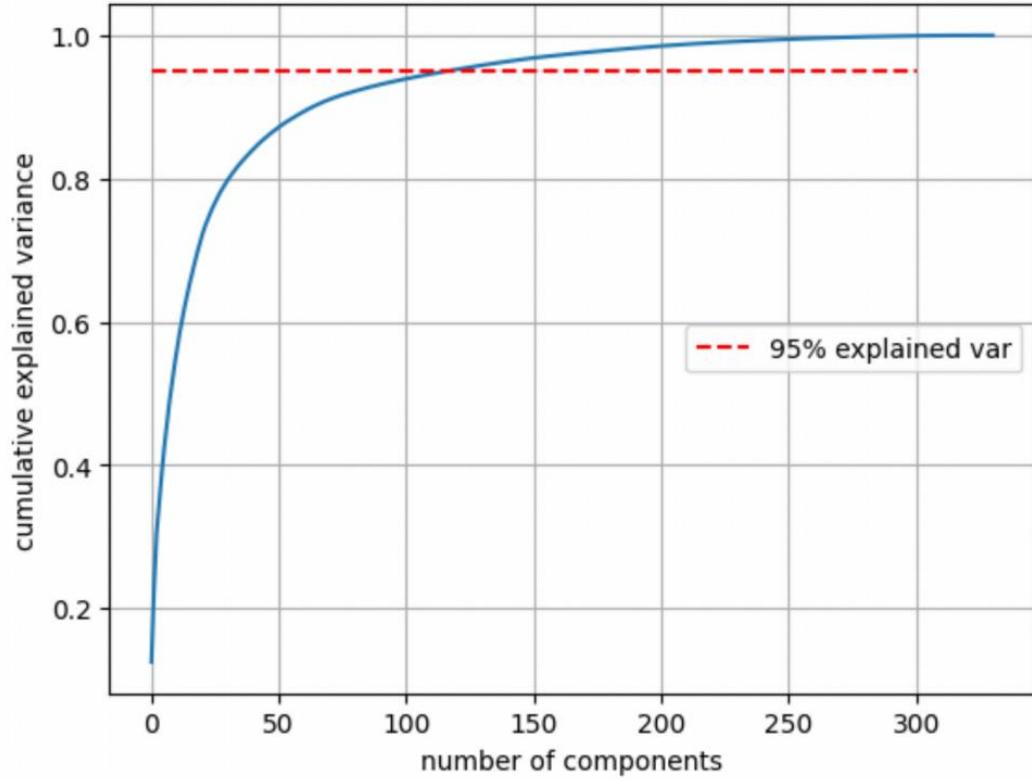


图 10: 累积方差 PCA

降维处理:

- `np.cumsum(pca.explained_variance_ratio_)`, 绘图处理, 选择成分数量使得可解释方差保留在 95% 的水平。

Task4：项目改进、新特征的加入

movies_info

movie_id	name	genre	release_time	intro	directors	stars
0	1 Toy Story (1995)	Animation Adventure Comedy	22 November 1995 (USA)	A cowboy doll is profoundly threatened and jea...	John Lasseter	Tom Hanks Tim Allen Don Rickles
1	2 Jumanji (1995)	Adventure Comedy Family	15 December 1995 (USA)	When two kids find and play a magical board ga...	Joe Johnston	Robin Williams Kirsten Dunst Bonnie Hunt
2	3 Grumpier Old Men (1995)	Comedy Romance	22 December 1995 (USA)	John and Max resolve to save their beloved bai...	Howard Deutch	Walter Matthau Jack Lemmon Ann-Margret
3	4 Waiting to Exhale (1995)	Comedy Drama Romance	22 December 1995 (USA)	Based on Terry McMillan's novel, this film fol...	Forest Whitaker	Whitney Houston Angela Bassett Loretta Devine
4	5 Father of the Bride Part II (1995)	Comedy Family Romance	8 December 1995 (USA)	George Banks must deal not only with the pregn...	Charles Shyer	Steve Martin Diane Keaton Martin Short
...
9737	193581 Kuroshitsuji: Book of the Atlantic (2017)	Animation	21 January 2017 (Japan)	A young lord and his demon butler board a luxu...	Noriyuki Abe Stephen Hoff	Bryn Appril Dawn Michelle Bennett Justin Briner

新特征选取

- 用户、评分两张表中可直接使用的特征已使用完，电影信息表中仍可以使用的特征有：上映时间、简介、导演、明星。
- 其中导演、明星两个特征高度稀疏，不利于使用。

```
In [6]: movies_info.directors.unique().shape
```

```
Out[6]: (4215,)
```

Task4：项目改进、新特征的加入

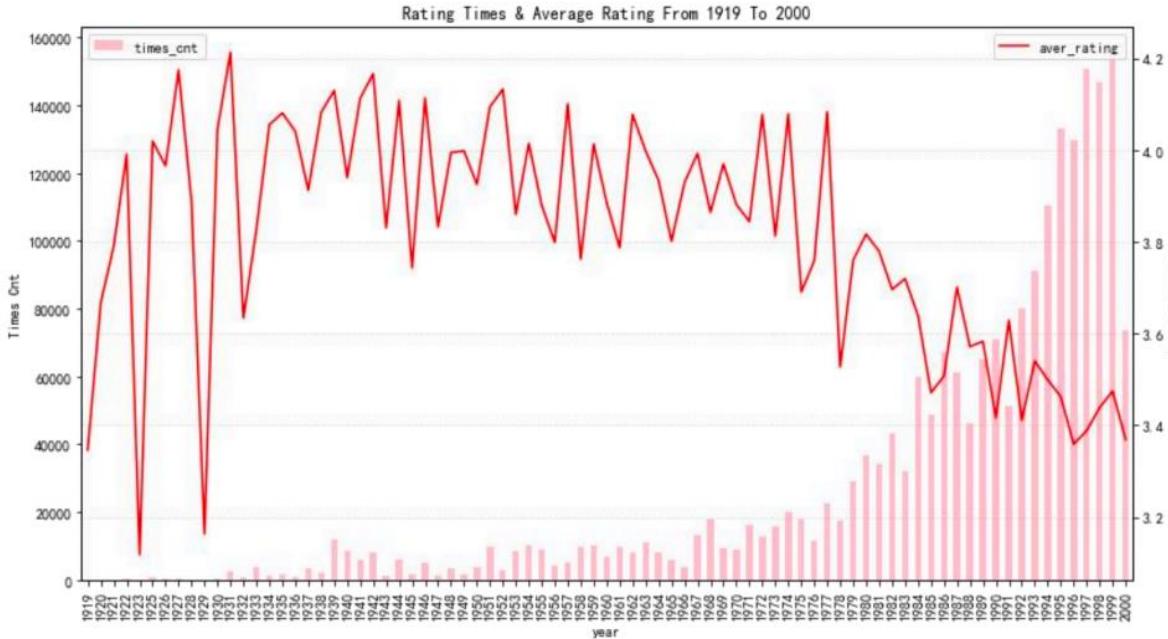
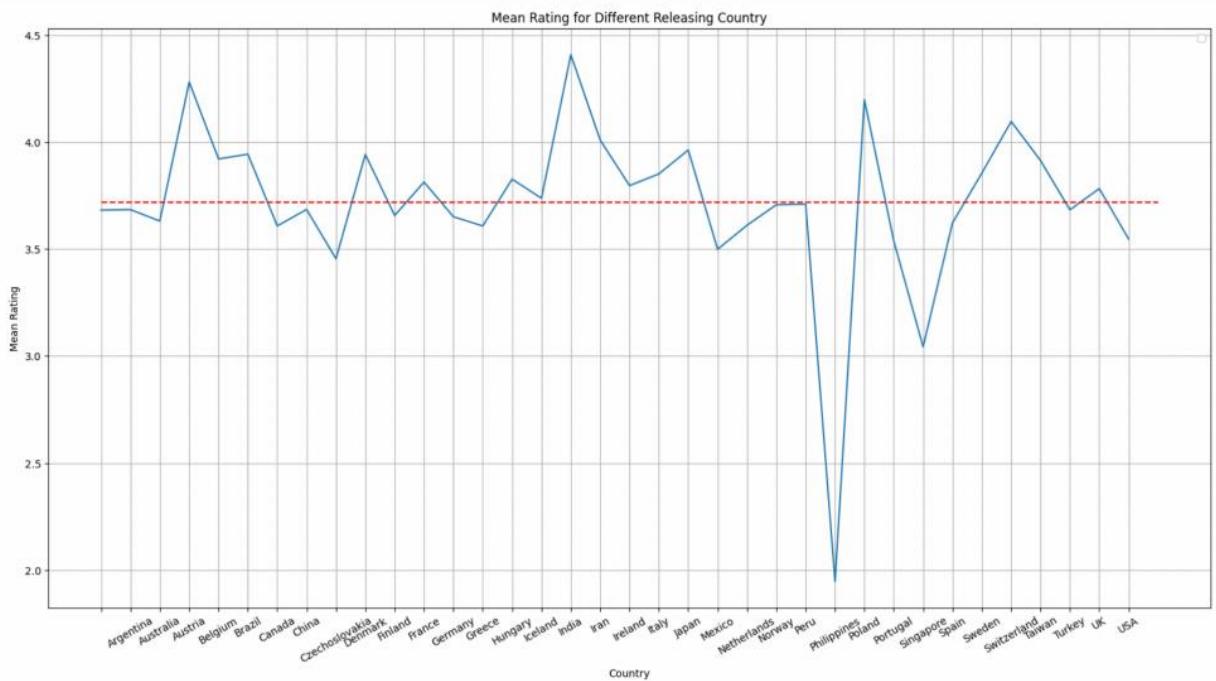


图 14: 时间-用户平均评分

1. 上映时间对评分的影响

- 数据集在时间跨度上是很大的，从1919年至2000年。直观感受上看，年份越古早，观影人数就越少，可能观众非常professional、也可能电影拍摄成本高，本身具有高分特征。而年份越晚，电影数量上升，质量鱼龙混杂，观众的打分水平可能下降。以上情况都可能存在，需要可视化数据集来具体判断。
- 左图展示了“用户平均评分、平均观影次数”两个变量在时间维度上的分布。发现观影次数陡增，我们关系的平均评分明显下降。

Task4：项目改进、新特征的加入



2. 制片国家对评分的影响

- 直观感受上，不少国家属于制片大国，如中国、美国。制片大国产出电影多，“好片”与“烂片”数量多，可能将评分水平拉回到均值附近。而一些制片小国，能产出走进全球观众视野的影片，天然就使得这部影片是高质量影片。同时加入制片国家对于不同观众是具有区分意义的。
- 左图展示了不同制片国家产出的影片获得的平均评分。上文谈到的中国、美国在影片总均分的附近。表现突出的有印度和菲律宾。

Task4：项目改进、新特征的加入

```
def getTFIDF(corpus):
    all_words = set()
    for sentence in corpus:
        all_words.update(set(sentence.split()))
    num_sentence = len(corpus)
    num_words = len(all_words)

    word2idx = {word:i for i, word in enumerate(all_words)}
    tfidf = np.zeros((num_words, num_sentence))

    word2idx_df=pd.DataFrame({'word':word2idx.keys(), 'idx':word2idx.values()})
    tf_j=[]

    for i,sentence in enumerate(corpus):
        #tf_i_j计算:
        words=sentence.split()
        words_df=pd.DataFrame({'word':dict(Counter(words)).keys(), 'num':\
                               dict(Counter(words)).values()})
        #用merge方法，获得每个文本中单词对应的索引，然后在tfidf矩阵中加上相应单词数
        words_idx=pd.merge(left=word2idx_df,right=words_df,how='inner')
        tfidf[words_idx['idx'][i]]+=words_idx['num']

        #加上文本d_j的单词数量
        tf_j.append(len(words))

    #tf_j计算:
    tfidf=np.array(tf_j)
    df_i=(tfidf>0).sum(axis=1)[:,np.newaxis]
    tfidf*=np.log(num_sentence/df_i)

    return tfidf, word2idx
```

3. 电影简介的处理

- tf-idf向量化处理。HW-7曾手搓过TFIDF统计函数，本次数据类型和结构和上次相似，直接使用。
- 统计后的词向量矩阵维度是 $3w+$ 的水平，同样使用累积方差的方法，将维度降到百维的水平。再将降维后的矩阵与原来的特征进行拼接。

Task4：项目改进、新特征的加入

Model: Logistic多分类回归

MSE:1.5177012827306267

	precision	recall	f1-score	support
1	0.23	0.07	0.11	11278
2	0.19	0.05	0.07	21393
3	0.31	0.23	0.26	52116
4	0.37	0.68	0.48	70047
5	0.38	0.20	0.26	45208
accuracy			0.35	200042
macro avg	0.30	0.25	0.24	200042
weighted avg	0.33	0.35	0.31	200042

MSE:1.4097565933228315

	precision	recall	f1-score	support
1	0.26	0.01	0.02	11002
2	0.31	0.00	0.00	21042
3	0.32	0.15	0.20	51104
4	0.36	0.80	0.50	68692
5	0.43	0.21	0.28	44950
accuracy			0.37	196790
macro avg	0.34	0.23	0.20	196790
weighted avg	0.35	0.37	0.29	196790

问题建模分析

分享者:陈锐韬

模型 1 : 线性模型

- One-Hot 提取两种特征，
 - 第一种，用户的性别、年龄和职业合并在一起使用 One-Hot 提取；
 - 第二种，电影的主题组合使用 One-Hot 提取特征。
- 标准化，PCA 降维，保留 90% 的方差。
- 计算 MSE 结果为 1.17178。

模型 2 : 按电影和用户性别分类

- 我们认为用户评分由此电影及用户性别决定。
- 将一用户给一电影的评分预测为训练集中所有同性别用户给同一电影评分的平均值。
- 计算 MSE 结果为 0.96284。

模型 3 : 电影和用户共同作用

- 我们认为用户 i 给电影 j 的评分 $r_{i,j}$ 由以下模型产生:
 - $\bullet \quad r_{i,j} = a_i + b_j + \varepsilon_{i,j}$
- 其中 a_i 是用户作用, b_j 是电影的作用, $\varepsilon_{i,j}$ 是随机噪声。
- 训练模型
 - 首先统计训练集中用户 i 给出的所有评分的平均数, 将这个平均数作为 a_i 的估计。
 - 再计算剩余的评分 $r_{i,j} - a_i$, 统计训练集中电影 j 获得的所有剩余评分 $r_{i,j} - a_i$ 的平均数, 作为 b_j 的估计。

模型 3：电影和用户共同作用

- 模型预测：将用户 i 给电影 j 的评分预测为电影和用户共同作用的结果，即 $a_i + b_j$ 。

- 计算 MSE 结果

为 0.86950。

```
# 训练模型
user_rating_avg = train.groupby('user_id')['rating'].mean() # 计算用户作用
train_data_full = pd.merge(train, user_rating_avg, on='user_id', how='left')
train_data_full['res'] = train_data_full['rating_x'] - train_data_full['rating_y'] # 计算剩余评分
movie_rating_avg = train_data_full.groupby('movie_id')['res'].mean() # 计算电影作用
```

```
# 预测结果
test_data_full = pd.merge(test, user_rating_avg, on='user_id', how='left')
test_data_full = pd.merge(test_data_full, movie_rating_avg, on='movie_id', how='left')
pred_3 = np.array(test_data_full['rating_y'] + test_data_full['res']) # 计算预测结果
pred_3 = np.clip(pred_3, a_min=1.5, a_max=4.5)
mse_score = np.mean(np.power(pred_3 - test['rating'], 2))
print('MSE:', mse_score)
```

MSE: 0.8695020667450292

模型 4 : 矩阵分解

- Low-Rank Matrix Completion
- 模型 3 预测结果的残差为 $y_{i,j} = r_{i,j} - a_i - b_j$ 。
- 希望用 UV^\top 来近似残差矩阵，其中残差矩阵是 $n_u \times n_m$ 矩阵， U 是 $n_u \times d$ 矩阵， V 是 $n_m \times d$ 矩阵， d 小于 n_u, n_m 。可知 UV^\top 是一个秩至多为 d 的低秩矩阵。
- 矩阵 UV^\top 的 (i, j) 位就是 $\langle u_i, v_j \rangle$ 。
- 最小化

$$\sum_{(i,j)} (\langle u_i, v_j \rangle - y_{i,j})^2$$

模型 4 : 与模型 3 关系

- 模型 3 是模型 4 的一个特例。
- 模型 3 用户 i 给电影 j 的评分 $r_{i,j}$ 由以下模型产生:
 - $r_{i,j} = a_i + b_j + \varepsilon_{i,j}$
- 令 $u_i = (a_i, 1), v_j = (1, b_j)$, 则
 - $r_{i,j} = \langle u_i, v_j \rangle + \varepsilon_{i,j}$
- 基于模型 3 预测结果的残差是合理的, 相当于在 $u_i = (a_i, 1), v_j = (1, b_j)$ 的基础上加长 u_i, v_j , 整体就是用低秩矩阵近似评分矩阵 $\{r_{i,j}\}$ 。

模型 4 : 训练

$$\sum_{(i,j)} (\langle u_i, v_j \rangle - y_{i,j})^2$$

- 训练模型
 - Alternating Least Square

- 固定所有 v_j 时，对 u_i 是最小二乘问题，固定所有 u_i 时，对 v_j 是最小二乘问题。
- 先初始化，用最小二乘交替更新 u_i 和 v_j 。

```
# 用交替最小二乘进行训练
from sklearn.linear_model import LinearRegression
for iter in range(15):
    # 更新 movies 的参数
    new_features = []
    for i in movies_features.index:
        X = np.array(list(users_features.loc[movies_features.loc[i, 'index'], 'features']))
        if X.size==0:
            new_features.append([0.]*n_factors)
            continue
        y = np.array(movies_features.loc[i, 'y'])
        new_features.append(LinearRegression(fit_intercept=False).fit(X, y).coef_)
    movies_features['features'] = new_features

    # 更新 user 的参数
    new_features = []
    for i in users_features.index:
        X = np.array(list(movies_features.loc[users_features.loc[i, 'index'], 'features']))
        if X.size==0:
            new_features.append([0.]*n_factors)
            continue
        y = np.array(users_features.loc[i, 'y'])
        new_features.append(LinearRegression(fit_intercept=False).fit(X, y).coef_)
    users_features['features'] = new_features
```

模型 4 : 预测结果

- 模型预测: 将用户 i 给电影 j 的评分预测为
 - $a_i + b_j + \langle u_i, v_j \rangle$
- 参数选择: 维数 $d = 3$, 进行 15 轮 Alternating Least Square 迭代。
- 计算 MSE 结果为 0.78306。
- 我们以 0.1, 0.1, 0.8 的比例混合模型 2、3、4 的预测结果, 得到最终模型的预测结果 MSE 为 0.77740。

Task 3 图像聚类与分类

- 基本要求
 - 合理提取图像的特征，包括直方图、Img2Vec等
 - 对特征的融合、降维等处理
 - 使用机器学习模型进行训练、评估等
- 主要考察点
 - 特征的选取与使用—针对不同的任务选择不同的特征提取方式
 - 问题建模与模型选择—多标签分类、聚类方法等

Task 3.4 有监督分类

- 多标签分类问题的处理
 - MultiOutputClassifier
 - Fitting one classifier per target.
 - Classifier Chain
 - 是否存在其他方式?

多标签分类问题

分享者:肖旭森

Multilabel classification

数据集不是很平衡。

```
genre_count:  
{'Film-Noir': 44,  
'Fantasy': 68,  
'Western': 68,  
'Animation': 105,  
'Mystery': 106,  
'Musical': 114,  
'Documentary': 127,  
'War': 143,  
'Crime': 211,  
"Children's": 251,  
'Sci-Fi': 276,  
'Adventure': 283,  
'Horror': 343,  
'Romance': 471,  
'Thriller': 492,  
'Action': 503,  
'Comedy': 1200,  
'Drama': 1603}
```

MultiOutputClassifier:

为每个目标拟合一个分类器。
将多个标签的分类问题转化为单个
标签的多个分类问题。

```
classifier = MultiOutputClassifier(LogisticRegression())  
classifier.fit(X_train, y_train)  
y_pred = classifier.predict(X_test)  
accuracy = accuracy_score(y_test, y_pred)
```

Accuracy: 0.20207253886010362

features:

img2vec+hist,; 简介, 电影名称,
导演, 演员(word2vec)

Multilabel → Multiclass

从每个电影的多个标签中挑出一个标签，使训练集的所有y都只有一个值。
转化为**18分类问题**。

尽量挑选**最常见**标签：

```
labels = np.where(y[i] == 1)[0]
labels = [idx2genre[label] for label in labels]
max_label = max(labels, key=lambda label: genre_count[label])
y_out[i] = genre_dict[max_label]
```

尽量挑选**最罕见**标签：

```
labels = np.where(y[i] == 1)[0]
labels = [idx2genre[label] for label in labels]
min_label = min(labels, key=lambda label: genre_count[label])
y_out[i] = genre_dict[min_label]
```

随机挑选：

```
y_out[i] = np.random.choice(np.where(y[i] == 1)[0])
```

Multilabel → Multiclass: Result

使用多分类分类器：

```
lgbm = LGBMClassifier(**params)
lgbm.fit(X_train, y_train_s_label)
# 预测
y_pred_s_label = lgbm.predict(X_test)
# 将单标签预测结果转换为多标签格式
y_pred_adv = single2multi_label(y_pred_s_label, 18)
# 计算acc
accuracy = accuracy_score(y_test, y_pred_adv)
```

Accuracy: 0.23661485319516407

Multilabel + Multiclass

比较蠢的方法，但是如果
不加额外的特征，也
可以达到这里的acc。

将Multiclass的预测结果和MultiOutputClassifier的预测结果合并。

```
def merge_predictions(y_pred, y_pred_adv):
    merged = np.logical_or(y_pred, y_pred_adv)
    merged = merged.astype(int)
    return merged

merge_accuracy = accuracy_score(y_test, merge_predictions(y_pred_logistic, y_pred_adv))
```

Lightgbm Merged Accuracy: 0.23488773747841105

Multiclass + Multiclass

多次进行选取标签并训练的流程。得到的所有模型投票获取最终预测结果。

```
def sampling_label(y, times=3):
    y_out_list = []
    assert times >= 3
    for i in range(times-2):
        y_out_list.append(rand_multi2singel_label(y))
    y_out_list.append(inverse_multi2single_label(y))
    y_out_list.append(multi2single_label(y))
    return y_out_list
```

Multiclass + Multiclass

```
models = []
for y_single in y_single_lst:
    y_train_single = y_single
    lgbm = LGBMClassifier(**params)
    lgbm.fit(X_train, y_train_single)
    models.append(lgbm)
```

acc : 0.29

```
y_pred_list = []
for model in models:
    y_pred_single = model.predict(X_test)
    y_pred_list.append(y_pred_single)
```

后续可能的改进：

```
y_pred = np.zeros((y_test.shape[0], 18))
for y_pred_single in y_pred_list:
    y_pred += single2multi_label(y_pred_single, 18)

# 所有>=ratio*times的标签变为1, 其余为0
y_pred = (y_pred >= ratio*times).astype(int)

accuracy = accuracy_score(y_test, y_pred)
```

1. 更好的多分类模型
2. 调参
3. 更有效的特征工程方法
4. 挑选标签的其他思路

Task 3.4 有监督分类

- 能够加入的辅助特征

- 电影信息：电影简介文本TF-IDF、导演、演员信息等
 - 用户信息：用户特征、偏好、评分等

- Ref. 陈锐韬

我们首先从 301 个主题组合中去掉了出现次数较少的组合，对剩余组合，根据用户偏好信息提取特征。

首先我们构建用户是否给电影评分的 0-1 矩阵。对任一类训练集中电影，记作 A 类，如果一个用户总评分次数较多，但给 A 类电影评分极少，我们认为该用户不喜欢 A 类电影，提取出不喜欢 A 类电影的一群人（这群人是由训练数据提取出的），将评分的 0-1 矩阵中属于这群人的向量加起来，作为每个电影的一个特征（训练集和测试集都能同样计算特征），即每个电影的特征为不喜欢 A 类电影的这群人给这一电影评分的总次数。A 类电影在这个特征会比较小，而其他电影在这个特征大概率不会很小，因此这个特征能给出电影是否是 A 类的信息。我们对 49 类电影这样提取特征，特征维数为 49。

- 疑问：新加入的信息带来的效果的提升，究竟是与海报信息联合而产生的，还是本身就具有的？去掉图像信息的消融实验？

Task 3.3 无监督聚类

- 聚类为什么困难?
 - 聚类簇的数目难以估计(KMeans)
 - 难以找到通用的评估数据相似性和性能的度量函数
- 数据特征提取、相似性度量、性能评估方法与任务目标和实际应用场景强相关
- 图像聚类的特征选择
 - 希望基于色彩风格聚类(HW14) — 直方图特征、HSV空间、...
 - 希望基于图像简单内容聚类 — 图像分类预训练模型 (img2vec)、...
 - 希望能够协助预测用户对电影的评分 — ???

Task 3.3 无监督聚类

- 也许全世界没人能回答的问题
- 电影海报蕴含的电影类别信息/电影内容信息有多充分?
- 电影海报对辅助预测不同用户对该电影的评分有多大帮助?
- 电影海报对辅助预测不同用户对该电影的偏好程度有多大帮助?

Task 3.3 无监督聚类

- 色彩分布、图片内容均有相似性，但显然属于不同类别、在不同群体下有不同偏好的图像



Task 3.3 无监督聚类

- Ref. 肖旭森

5.2.2 表现不佳的解释(5')

1. 这可能是由于海报的灰度和颜色信息以及Img2Vec获得的海报的轮廓以及分布等信息等内容，并不能直接和对应电影的类型乃至电影的信息相对应。

例如在实验无监督聚类时，如果在特征中加入了大量的电影相关信息，海报的聚类结果会出现一些混乱，并不是大部分色调或分布接近的海报聚为一类，而是引入了更多的不同海报风格不近似但是电影风格接近的海报。这说明海报的色彩和分布等信息和电影的信息并不能完全对应。如果仅对海报的信息进行聚类，可能获得的类别只是海报色彩和结构类似，而聚类的类别未必与电影的内容和类别有相关性，引入了更复杂的变化。

因此聚类信息的引入可能会反而引入更多误差信息，进导致模型表现不佳。

2. 而对于评分的预测，从直观来说，电影的评分和电影的内容更相关，而和海报的信息相关性较小。因此海报的无监督聚类信息直接作为特征引入预测模型，可能并不能直接起到增加有效信息的作用，反而会引入更多更复杂的信息。

Task 3.3 无监督聚类

• Ref. 王子宸

1. 信息含量有限:

- 虽然通过聚类分析，我们可以捕捉到电影海报中的视觉相似性，但这种视觉特征可能与用户的评分行为关联性不大。用户对电影的评分通常受到剧情、演技、导演、个人喜好等多种因素的影响，而这些因素可能无法通过海报的视觉内容直接反映。

2. 特征表示的局限性:

- 基于距离的 OneHot 编码尽管能够提供关于聚类中心距离的信息，但这种表示可能不够丰富，无法捕捉到更复杂的数据结构和关系。此外，距离编码可能在不同聚类中的解释性和区分度上存在差异，有些聚类可能由于内部差异大而使得这种编码效果不佳。

3. 聚类质量和相关性:

- 聚类的效果很大程度上依赖于所选择的特征和聚类算法的适用性。如果聚类算法没有很好地捕捉到对评分预测有用的信息，或者聚类的结果与目标变量（用户评分）关联性低，那么聚类标签和距离特征对预测模型的贡献也会很有限，例如电影海报聚类更多是基于图像（色彩/灰度）的特征，但是电影评分更多基于电影内容的优劣，二者之间并无因果性，相关性也非常存疑。

Task 3.3 无监督聚类

- Ref. 马佳昊

一种猜想是海报内容嵌入与简介嵌入的内容重合性较强。同时，由于resnet18的预训练是在ImageNet上进行，并未对电影海报这种艺术风格化的图像进行特别训练，进而嵌入效果不佳。尝试以当前的数据重新训练resnet18，但数据集较小导致极易过拟合。

- Ref. 尹奕涵

可以看出，引入了海报信息之后，各种模型评分预测的效果并没有显著提高，这可能是因为在未引入海报信息的数据集中，已经包含了电影的类别信息，而海报信息和电影类别强相关，因此引入海报信息的过程并没有引入很多额外的信息量，从而使得模型无法从数据集中获得更多特征信息；另外，海报类别与电影类别相异的部分会引入更多迷惑性，由于海报的特征信息来源于颜色直方图和灰度直方图，而直方图相似的海报有可能联系着两类差异非常大的电影，比如海报中大面积的红色系色块，可能代表着浪漫电影也可能代表恐怖电影，因此海报聚类还有可能引入这样的信息，不利于用户评分的拟合。

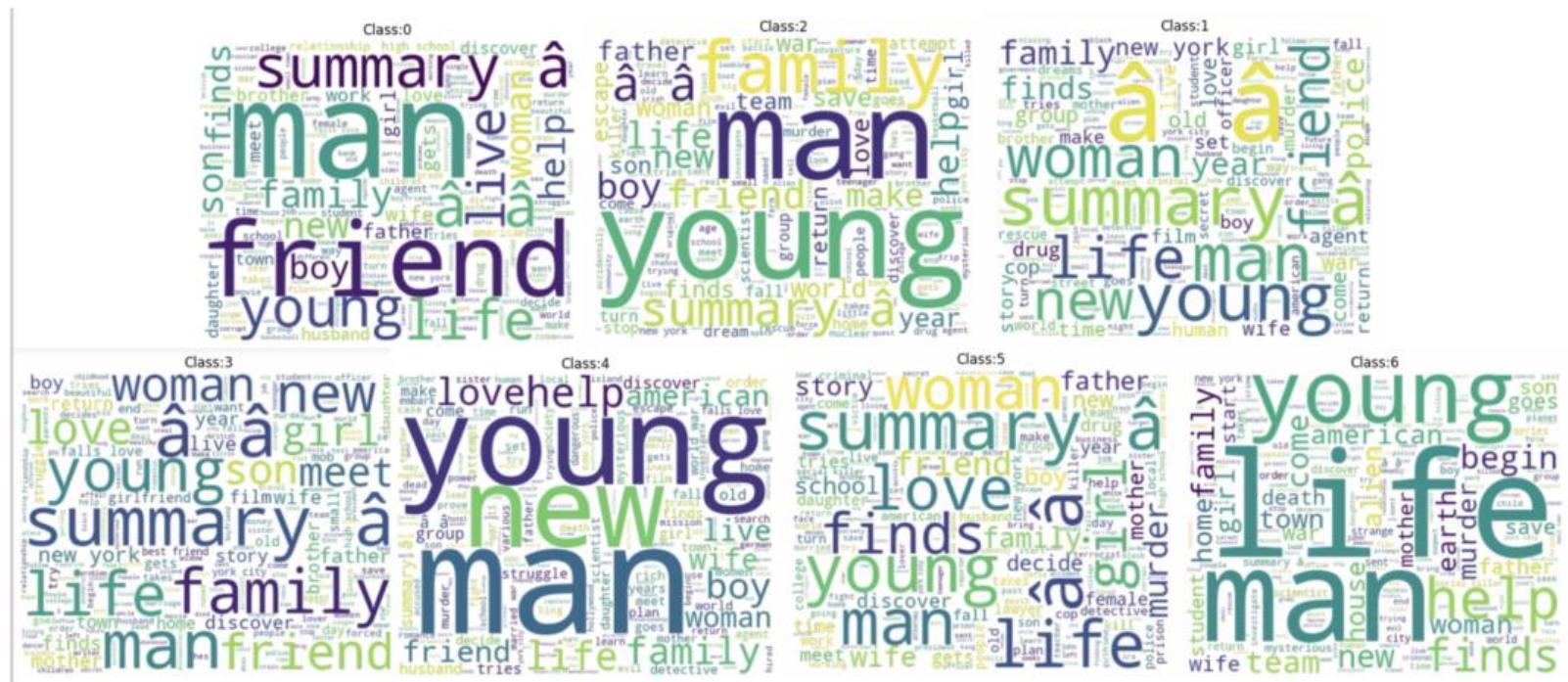
Task 3.3 无监督聚类

这使得我们获得的聚类结果会是一堆视觉艺术手法上接近的电影被聚为一类。比如暗色系、亮色系的会各被聚为一组，海报上有人物大头像的会被聚为一组等。

这样的海报聚类方法使得我们分出来的类和电影本身的内容和类型关系不大，将这个特征对用户评分进行回归无异于让用户单独欣赏一幅海报，然后进行评分，这是不合理的。

为了证明上面的想法，将每个类的电影的 intro 简介部分提取，进行词频统计并绘制词云。如果词云不表现出明显差异，就说明海报聚类与内容关系较小。下面是词云结果，符合预期。

- Ref. 朱逸飞



Task 3.3 无监督聚类

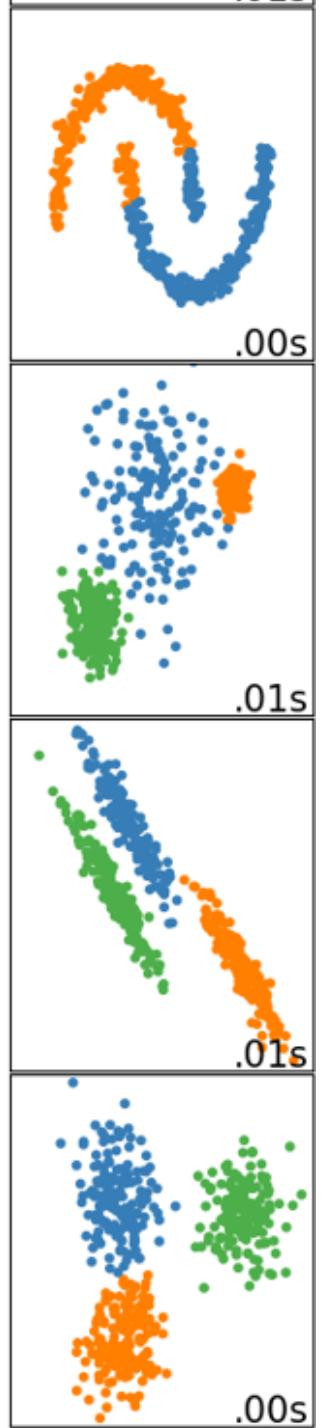
- Ref. 周晨昊
- 只需要简单的将海报特征经过投影层后和其他特征进行拼接，最后通过一个线性层获得预测的评分，就能将MSE从0.970优化到0.851<0.9

```
def forward(self, x_gender, x_age, x_occ, x_genres, x_desc, x_user_id, x_movie_id, x_f):  
    x_user = self.user_encoder(x_gender, x_age, x_occ, x_user_id)  
    x_movie = self.movie_encoder(x_genres, x_desc, x_movie_id)  
    x_f = self.movie_feature_fc(x_f)  
    x = torch.cat([x_user, x_movie, x_f], dim=1)  
    x = self.fc(x)  
    x = self.pred(x)  
    return x
```

- 此前的方法中添加海报信息的尝试结果对用户评分预测提升不大，究竟是海报本身的信息无法支持更好的预测，还是特征提取和模型选择无法完全充分利用海报中的信息？

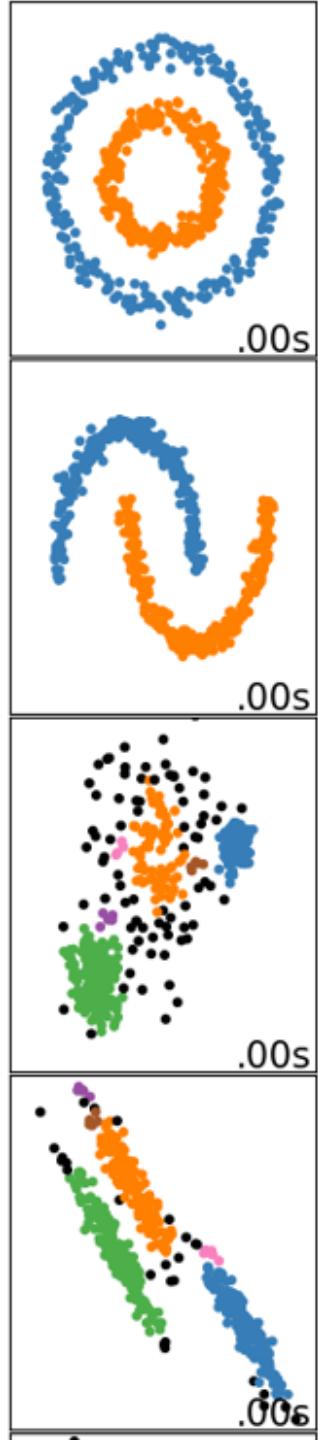
补充：聚类算法：GMM

- `sklearn.mixture.GaussianMixture`
- GMM（高斯混合模型）：基于概率模型，假设数据由多个高斯分布组合，通过使用EM（期望最大化）算法迭代地估计每个高斯分布的参数（均值、协方差、混合系数）直到收敛。最后，根据每个数据点属于不同高斯分布的概率，将其分配到相应的簇中
- 优点
 - 可以提供软聚类结果，即每个数据点属于各个簇的概率
- 缺点
 - 计算复杂度较高，可能需要较长的运行时间
 - 对初始值敏感，可能陷入局部最优解



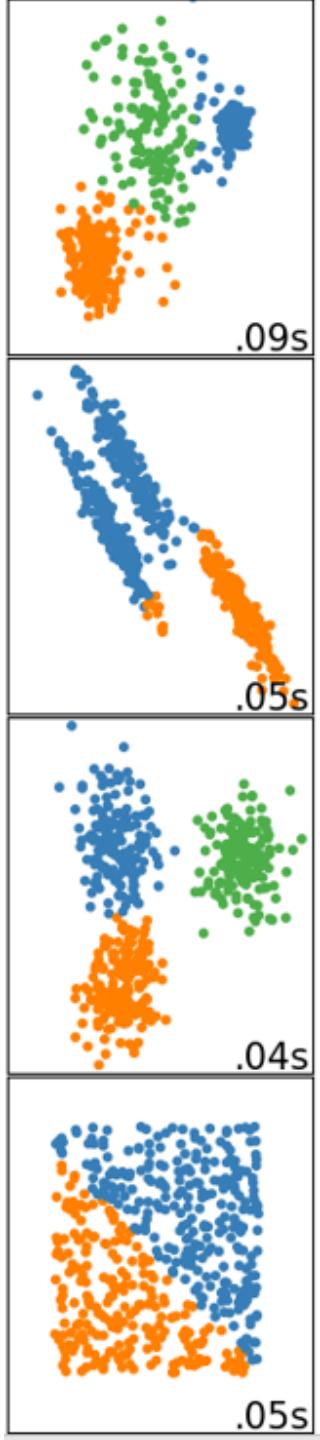
补充：聚类算法：DBSCAN

- `sklearn.cluster.DBSCAN`
- DBSCAN：基于密度
 - 核心点：在给定半径 ϵ 内有至少 MinPts 个数据点的点
 - 从一个核心点开始，将其半径 ϵ 内的所有点归入同一个簇，然后递归地对这些点进行相同的操作
 - 当不能再扩展簇时，选择另一个核心点，继续构建新的簇
 - 不属于任何簇的点属于噪声点
- 优点
 - 不需要预先设定簇的数量
 - 能够识别和处理噪声点
- 缺点
 - 对参数 ϵ 和 MinPts 敏感，需要调整以获得合适的聚类结果
 - 对密度不均匀的数据表现不佳



补充：聚类算法：MeanShift

- `sklearn.cluster.MeanShift`
- **MeanShift**: 让聚类中心向密度更高的地方移动
 - 确定一个初始点C
 - 在以C为圆形, D为半径的圆内计算重心C'
 - 将C移动到C', 反复直至收敛
- 优点
 - 不需要预先设定簇的数量
- 缺点
 - 参数D的选择较困难
 - 高维特征空间表现不佳



补充：聚类算法：层次聚类

- `sklearn.cluster.AgglomerativeClustering`
- 层次聚类：基于树形结构，可以自底向上或自顶向下
- 自底向上的方法
 - 最开始每个数据点都是一个簇
 - 计算簇之间的距离，将最相近的两个簇合并，重复此过程直到所有数据点都在一个簇中，得到具有层次结构的树状图
 - 可以通过截断树状图来选择合适的簇数量
- 优点
 - 不需要预先设定簇的数量
- 缺点
 - 一旦合并或分裂，难以再进行调整
 - 对异常值敏感

