3120190981 陈雨馨 基于矩阵分解的协同过滤的推荐系统

3220190892 温名阳 基于 CNN 的推荐系统

3220190822 李恩　　基于聚类的推荐系统

一、数据集说明

本项目数据集来源于 myanimelist.net 网站，包含 12294 部动画作品的 73516 个用户的偏好信息。每个用户都可以将动画添加到他们的观看记录中，并对其进行评分。数据集包含 anime.csv 和 rating.csv 两个文件。anime.csv 文件包含 12293 部动画作品信息，由动画 id，动画名称，题材，播放类型，集数，评分和评分人数等组成，如图 1-1 所示。rating.csv 文件包含 73516 个用户，共 7813736 个评分意见，由用户 id，动画 id 和每个用户动画评分组成，如图 1-2 所示。

	anime_id	name	genre	type	episodes	anime_rating	members	anime_row
0	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	9.37	200630	0
1	5114	Fullmetal Alchemist: Brotherhood	Action, Adventure, Drama, Fantasy, Magic, Mili...	TV	64	9.26	793665	1
2	28977	Gintama°	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.25	114262	2
3	9253	Steins;Gate	Sci-Fi, Thriller	TV	24	9.17	673572	3
4	9969	Gintama'	Action, Comedy, Historical, Parody, Samurai, S...	TV	51	9.16	151266	4

图 1-1 anime.csv 文件信息

	user_id	anime_id	user_rating
0	1	20	-1
1	1	24	-1
2	1	79	-1
3	1	226	-1
4	1	241	-1

图 1-2 rating.csv 文件信息

二、基于矩阵分解的协同过滤的动画推荐系统

基于矩阵分解的协同过滤推荐算法思想为一个用户评分矩阵可以分解为一个用户喜好矩阵与动画内容矩阵，我们只要能找出正确的用户喜好矩阵参数与动画内容矩阵参数（即表内的值），就能对用户喜好进行预测，再根据预测结果对用户进行推荐。

2.1 总体设计

在本推荐系统中，我们通过构造一个动画评分表，如表 2-1 所示，来预测目标用户对动画的评分，进而根据评分高低，将分高的动画推荐给用户。

y	动画 1	动画 2	动画 3	动画 4	动画 5
用户 1	3	6	1	6	0
用户 2	5	1	9	4	8
用户 3	2	5	7	0	3
用户 4	4	6	5	2	7

表 2-1 动画评分表

在对动画进行评分时，许多因素都会影响到用户对动画的评分，如题材、播放类型、播放时长等因素。因此，我们只要知道所有用户对动画内容等各种因素的喜欢程度与动画的所有相关信息，就能预测出所有用户对动画的评分了。我们可以构造得到用户喜好表和动画内容表，如表 2-2 和表 2-3 所示。对所有用户，我们将两个表相乘就能得到所有用户对所有动画的预测评分，如表 2-1 所示。

x	因素 1	因素 2
用户 1	2	0
用户 2	6	3
用户 3	7	6
用户 4	5	1

表 2-2 用户喜好表

w	动画 1	动画 2	动画 3	动画 4	动画 5
因素 1	0.4	0.6	1.0	1.0	0
因素 2	0.2	0.6	0	0.5	1.0

表 2-3 动画内容表

假设动画评分表 y（为 m 行 n 列的矩阵）,我们考虑 d 种因素，则动画评分表可以分解为用户喜好表 x（为 m 行 d 列的矩阵），与动画内容表 w（为 d 行 n 列的矩阵）。其中 d 为超参数，本项目中定为 10。

将用户喜好矩阵与内容矩阵进行矩阵乘法就能得到用户对动画的预测结果，而我们的目的是预测结果与真实情况越接近越好。所以，我们将预测值与评分表中已评分部分的值构造平方差损失函数，为了防止过拟合，我们添加了正则化项。其中，i 表示第 i 个用户，j 表示第 j 部动画，d 为超参数，x 为用户喜好矩阵，w 为内容矩阵，y 为评分矩阵，r 表示评分记录矩阵。

$$\text{loss} = \frac{1}{2}\sum_{(i,j)\in r(i,j)=1}(\sum_{l=1}^{d} x_{il}\, w_{lj} - y_{ij})^2 + \frac{1}{2}\sum_{l=1}^{d}(\|X\|^2 + \|W\|^2)$$

2.2 实验过程

我们首先对数据进行清洗处理，然后将 anime.csv 和 rating.csv 文件中信息根据 anime_id 这一标称属性整合到一张表中，命名为 urating，包含每个用户的评分记录，以及被评分动画的相关信息。

	user_id	anime_id	user_rating	name	genre	type	episodes	anime_rating	members	anime_row
0	1	20	-1	Naruto	Action, Comedy, Martial Arts, Shounen, Super P...	TV	220	7.81	683297	841
1	3	20	8	Naruto	Action, Comedy, Martial Arts, Shounen, Super P...	TV	220	7.81	683297	841
2	5	20	6	Naruto	Action, Comedy, Martial Arts, Shounen, Super P...	TV	220	7.81	683297	841
3	6	20	-1	Naruto	Action, Comedy, Martial Arts, Shounen, Super P...	TV	220	7.81	683297	841
4	10	20	-1	Naruto	Action, Comedy, Martial Arts, Shounen, Super P...	TV	220	7.81	683297	841

图 2-3 将 anime.csv 和 rating.csv 整合后的 urating 表

但在本推荐系统中，我们只需要根据用户 id、用户评分和动画 id 来对用户进行动画推荐，因此可以剔除多余的属性信息。由于动画 id 是不连续的，因此我们重新对所有动画进行了排序作为动画索引，标注为 anime_row 属性来唯一指代被评分动画，得到精简后的 urating 表。

	user_id	anime_row	user_rating
0	1	841	-1
1	3	841	8
2	5	841	6
3	6	841	-1
4	10	841	-1

图 2-4 精简后的 urating 表

然后构建动画评分矩阵和评分记录矩阵，并对矩阵进行归一化处理。处理完成后，利用动画评分矩阵和评分记录矩阵进行模型构建和训练，并利用 Adam 优化器进行参数优化。

为了最小化损失函数，我们采用梯度下降的方法来得到正确的参数。经过 5000 次梯度下降迭代后，得到误差在 1.99 左右，如图 2-5 所示。

```
step: 4000  train loss:0.00143  test loss:1.99595
step: 4100  train loss:0.00142  test loss:1.99454
step: 4200  train loss:0.00141  test loss:1.99345
step: 4300  train loss:0.00140  test loss:1.99266
step: 4400  train loss:0.00139  test loss:1.99217
step: 4500  train loss:0.00138  test loss:1.99198
step: 4600  train loss:0.00137  test loss:1.99209
step: 4700  train loss:0.00136  test loss:1.99249
step: 4800  train loss:0.00135  test loss:1.99319
step: 4900  train loss:0.00134  test loss:1.99420
```

图 2-5 通过梯度下降得到正确参数

得到的预测模型为动漫-用户矩阵，输入用户的 id,在矩阵中搜索该用户列下前 20 评分最高的动画，如图 2-6 所示。

```
您要向哪位用户进行推荐？请输入用户编号：66
==============================为该用户推荐的评分最高的20部动画是：==============
评分：9.09，动画名：Dirty Pair
评分：6.69，动画名：Ougon Bat
评分：6.68，动画名：Ginga Tetsudou 999: Kimi wa Haha no You ni Aiseru ka!!
评分：6.67，动画名：Lady Georgie
评分：6.67，动画名：Sekai Meisaku Douwa: Hakuchou no Mizuumi
评分：6.66，动画名：Shoukoushi Cedie
评分：6.65，动画名：Usavich
评分：6.65，动画名：Hello! Sandybell
评分：6.64，动画名：Ring ni Kakero 1: Nichibei Kessen-hen
评分：6.64，动画名：Ro-Kyu-Bu!: Tomoka no Ichigo Sundae
评分：6.64，动画名：Tobe! Isami
评分：6.64，动画名：Ai Shoujo Pollyanna Story
评分：6.63，动画名：Uchuu Senkan Yamato: Kanketsu-hen
评分：6.63，动画名：Hoshi no Oujisama Petit Prince
评分：6.63，动画名：Pucca
评分：6.63，动画名：Bananya
评分：6.62，动画名：Shinkon Gattai Godannar!! 2nd Season
评分：6.60，动画名：Tokimeki Tonight
评分：6.59，动画名：Miracle Giants Doumu-kun
评分：6.58，动画名：Bosco Daibouken
```

图 2-6 动画推荐系统

2.3 本章小结

基于矩阵分解的协同过滤推荐算法方法简单，一定程度上可以反应用户对不同动画的喜爱程度。但没有充分利用动画题材、播放类型以及播放时长等信息，推荐结果可能存在误差较大的情况。

三、基于 CNN 的动漫推荐系统设计

3.1 总体设计

上述的基于协同过滤是推荐系统中应用广泛的技术，主要是通过最小化 rating 矩阵和 predict 矩阵的差值，从全局的角度模拟一个 rating 矩阵，该方法能一定程度上反应用户对各动漫的喜爱程度，但缺点同样明显，没有用到 anime 表中关于动漫的分类，播放类型等属性，为了将这些信息用上并做种类相关的推荐，下面采用基于 CNN 来提取相应的特征完成推荐。

设计的思路参考了[1]，并将其修改成符合所选题的类型，整体框架如图 3-1 所示：

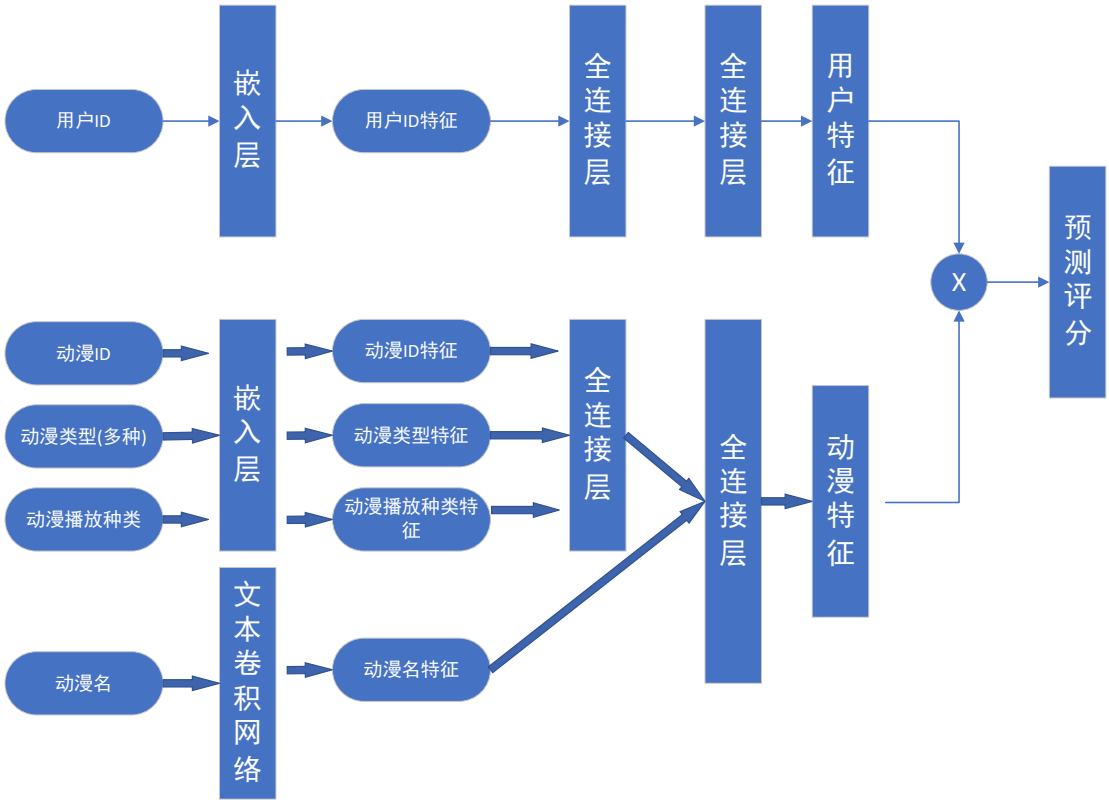


图 3-1：基于 CNN 方案的整体框图

从整体上分成了两个部分，一个为用户部分，一个为动漫属性部分，应该所用数据库中没有用户相关的信息，这里只用了用户 ID 作为用户特征，而动漫属性选了动漫 ID，动漫类型(如 Action, SC-FI)，动漫播放种类(如 TV,OVA)和动漫名，动漫名这个属性比较特征，因为其由若干个字符串组成，因此首先构建一个词向量字典，将其字符串映射为数组以满足网络

的输入条件，之后通过文本卷积网络进行特征提取，经过全连接层映射之后分别得到 1*200 维度的用户特征和动漫特征，对其点乘求和得到评价分数。

3.2 文本卷积网络

和图像处理不同的是，自然语言通常是一段文字，文献[2]用 CNN 来对文本序列提取特征并进行分类。大致的处理方法为，首先构建一个词向量字典，建立文中每一个文本序列对应的标号，如“Death note”对应对应词典向量 1、2，在用 CNN 提特征使 Death note 就用数字[1,2]代替，若动漫名中包含 n 个词，则用 n 维向量表示，这样就构成了 m*n 的词向量矩阵，m 表示动漫数量，因为矩阵列数长度需要统一，设定 n=20，不足的部分用“<PAD>”对应值填充。

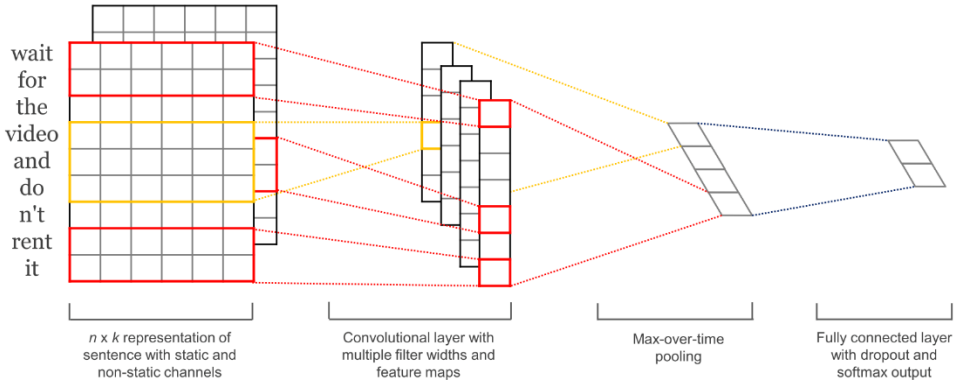


图 3-2：文本网络示意图

卷积核的选择和传统图像的卷积核的选择也有所不同，因此动漫名称比较特殊，名称长度有一定限制，这里滤波器大小使用时，选择了 2,3,4,5 四种长度。然后对文本嵌入层分别进行滑动做卷积，之后进行最大化池化和 dropout 处理，卷积核数量选择为 8，所以一共能得到 8*4 维特征向量，打平后即为 1*32 的动漫名特征。

3.3 实验过程

预处理:

找出剔除两个 CSV 中全部有空值的数据行，因为 rating 表中存在大量 rating 为-1 的情况，也就是该用户未打分，将其删除，为保证两个表的一致性，rating 表中还需要删除 anime 表中有空值 id 对应的打分记录

训练:

本次实验实验环境为 Tensorflow，设定嵌入矩阵的扩充维度为 32，用户 ID 经过嵌入层和两层全连接可以得到 1*200 的用户特征向量。经过嵌入层后的动漫相关属性通过 concat 拼接扩充维度，并接一个全连接层重映射，同样得到一个 1*200 的动漫特征向量，将两者进行用户特征向量和动漫特征向量进行点乘计算最后预测得分。损失函数采用 MSE 均方误差，公式如下 4-1，并利用 Adam 优化器进行参数优化。

$$\frac{1}{m}\sum_{i=1}^{m}(predict^{(i)}-label^{(i)}) \tag{4-1}$$

由于数据量大，为节约时间，在训练过程中采用大 batch 进行训练，这样做的好处是模型能快速收敛，但在收敛到一定精度时再想降下 loss 则非常困难。图 4-3 为模型训练时训练集和测试集收敛情况，横轴表示迭代次数，纵轴表示 MSE，因为是大 batch，可以看出模

型在一开始收敛的非常快，之后几乎不怎么下降，测试集有一些下降的趋势，本次实验跑了 3 个 epoch，整体误差在 1.4 左右。相信多跑几个 epoch，适当降低 batch 大小，迭代效果会更好

训练完成后，通过前向传播可输入用户 id 和动漫 id 可以得到评分，如图 3-4，用户 1 对于 id 为 8074 进行评价的得分为 8.89(ground_truth 为 10)，获取模型中最后一层张量：用户特征和动漫特征，并利用它们做动漫推荐。

A　　B

图 3-3：A 为训练集上收敛情况，B 为测试集上的收敛情况

```
rating_animation(1,8074)
```

```
INFO:tensorflow:Restoring parameters from ./save
[array([[ 8.88977337]], dtype=float32)]
```

图 3-4：正向传播预测评分

下面通过特征矩阵可以实现如下三种推荐，这三种推荐方式的思想类似。

推荐同类型的动漫：

处理方式和协同过滤的思想类型类似，输入动漫 id，计算该 id 动漫的特征向量和整个动漫特征向量矩阵的余弦相似度，去相似度最大的 k 个(20)，并从中随机挑选 5 个作为推荐，推荐结果如图 3-5 所示。

```
recommend_same_type_animation(1, 20)
```

```
INFO:tensorflow:Restoring parameters from ./save
您看的动漫是：[22 1 'Cowboy Bebop' 'Action, Adventure, Comedy, Drama, Sci-Fi, Space' 'TV'
 '26' 8.82 486824]
以下是给您的推荐：
69
[69 12431 'Uchuu Kyoudai' 'Comedy, Sci-Fi, Seinen, Slice of Life, Space'
 'TV' '99' 8.59 72958]
37
[37 31757 'Kizumonogatari II: Nekketsu-hen'
 'Action, Mystery, Supernatural, Vampire' 'Movie' '1' 8.73 34347]
19
[19 1575 'Code Geass: Hangyaku no Lelouch'
 'Action, Mecha, Military, School, Sci-Fi, Super Power' 'TV' '25' 8.83
 715151]
24
[24 164 'Mononoke Hime' 'Action, Adventure, Fantasy' 'Movie' '1' 8.81
 339556]
25
[25 7311 'Suzumiya Haruhi no Shoushitsu'
 'Comedy, Mystery, Romance, School, Sci-Fi, Supernatural' 'Movie' '1' 8.81
 240297]
```

图 3-5：推荐同类型动漫

推荐用户喜欢的动漫:

输入用户 id，并从用户矩阵中提取用户特征，将该特征和动漫矩阵进行相乘得到预测的该用户所有评分，筛选出评分最高的 k(30)个动漫，并从中随机挑选 5 个作为推荐，推荐结果如图 3-6。

```
recommend_your_favorite_animation(1, 30)
```

```
INFO:tensorflow:Restoring parameters from ./save
以下是给您的推荐：
864
[864 2450
 'Crayon Shin-chan Movie 09: Arashi wo Yobu Mouretsu! Otona Teikoku no Gyakushuu'
 'Comedy, Ecchi, Kids, School, Shounen, Slice of Life' 'Movie' '1' 7.79
 2688]
1
[1 5114 'Fullmetal Alchemist: Brotherhood'
 'Action, Adventure, Drama, Fantasy, Magic, Military, Shounen' 'TV' '64'
 9.26 793665]
2
[2 28977 'Gintama°'
 'Action, Comedy, Historical, Parody, Samurai, Sci-Fi, Shounen' 'TV' '51'
 9.25 114262]
17
[17 24701 'Mushishi Zoku Shou 2nd Season'
 'Adventure, Fantasy, Historical, Mystery, Seinen, Slice of Life, Supernatural'
 'TV' '10' 8.88 75894]
10010
[10052 15905 'Qin Shiming Yue Zhi: Wanli Changcheng'
 'Action, Fantasy, Historical, Martial Arts' 'TV' '37' 8.43 273]
```

图 3-6：推荐用户喜欢的动漫

推荐某动漫的观看群体:

输入动漫 id，提取该 id 的动漫矩阵，并与所有用户特征进行相乘，得到所有用户对该动漫的平安，并从中筛选评分最高的 k(10)个用户，进行推荐，推荐结果如图 4-7。

```
recommend_other_users(32281, 10)
```

```
INFO:tensorflow:Restoring parameters from ./save
您看的动漫是：Kimi no Na wa.
同样还喜欢看这个动漫的人是(编号)：[48396  8930 19968 70468 55492 42820 57425 29453 21474 39466]
```

图 3-7：推荐某动漫的观看群体

3.4 实验总结

1.相比于协同过滤矩阵分解方法，CNN 的特征提取方法用上了动漫各类属性信息，能够更有效对动漫相关特征进行提取。
2.利用用户矩阵和动漫矩阵进行推荐有可靠的解释性，并且能完整的进行推荐。
3.用户特征不够，仅有 id 特征，这样构建的用户特征向量信息存在缺陷，加上大 batch 训练，训练的收敛情况不是很理想，有待改善。

四、基于聚类的动画推荐系统

基于聚类的动画推荐系统的主要思路是筛选出每个用户真正喜爱的动漫并将用户进行聚类，然后从抽取的聚类特征中进行推荐，该方法的优点是挑选出用户评分较高的优质动漫进行推荐，缺点就是精读不高，作为群体喜欢的动漫可能个人不一定喜欢，没办法达到“个性化”的目的。下面采用基于 K 近邻聚类方法抽取不同种群用户喜爱的动漫来进行推荐。

4.1 整体框架

1.对每个用户的评分取平均，并定义平均分以上的得分为用户真正喜爱的动漫

2.将用户评价数据集及动漫属性数据集合并

3.利用主成成分分析对数据进行降维处理

4.通过 K 近邻聚类的方式将用户分类

5.获得每个类别的属性（评分最高的动漫，该类别关键词，平均集数等）

4.2 数据预处理

首先定义用户真正喜爱的动漫的标准，由于数据集中用户数量较多并且每位用户对动漫的评价标准不完全相同，因此本实验选择每位用户对所评价动漫的平均评分作为标准，定义平均分以上的动漫为该用户真正喜爱的动漫，得到的数据如图 4-1，分别是用户 1 和用户 2 真正喜爱的动漫。

	user_id	anime_id	rating	mean_rating
47	1	8074	10	-0.712418
81	1	11617	10	-0.712418
83	1	11757	10	-0.712418
101	1	15451	10	-0.712418
153	2	11771	10	2.666667

图 4-1. 获得用户真正喜爱的动漫

其次利用主成成分分析的方法对数据特征维数进行降维处理，将特征维数降成 3 维得到如图 2 所示的数据点。

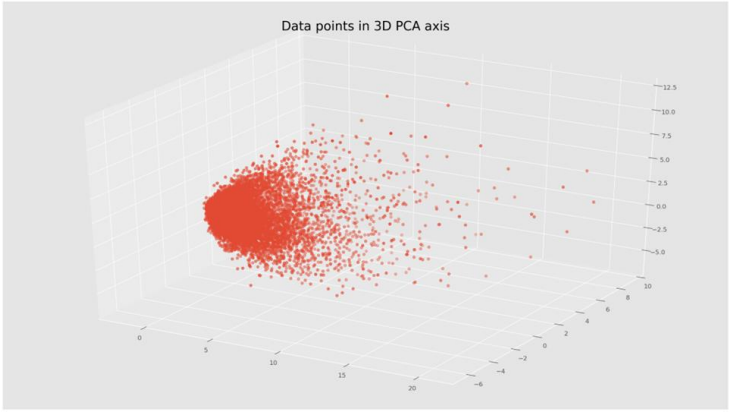


图 4-2. 降维处理后的数据分布

接下来利用轮廓系数（silhouette_score）分数对聚类的个数进行效果评估如图 3，得到需要聚类的个数为 4.

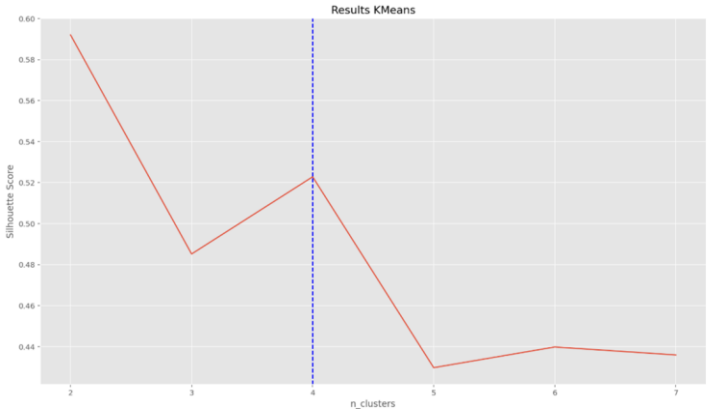


图 4-3. 对聚类效果评估

通过 K 近邻聚类算法对数据进行聚类，指定聚类的个数为 4.首先生成聚类的中心点，之后对其他数据点进行聚类。聚类后的效果如图 4 分别以 2 维和 3 维效果呈现。

聚类效果

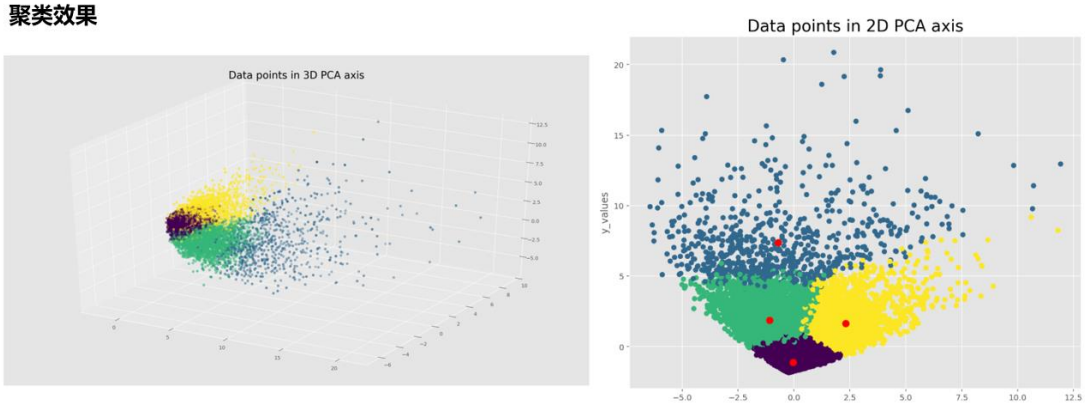


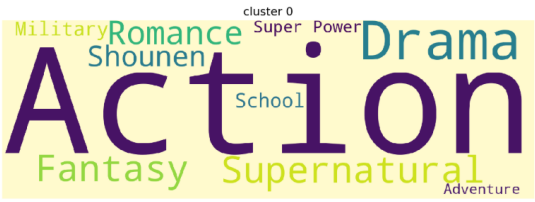
图 4-4. 聚类后的效果

4.3 聚类分析

对每个类别的动漫进行分析，得到该类别最受喜爱的动漫，该类别动漫题材关键词词云以及该类别动漫的属性如平均集数，平均得分，评分人数等。

```
name
Death Note                          0.320642
Shingeki no Kyojin                  0.230444
Sword Art Online                    0.198824
Fullmetal Alchemist: Brotherhood    0.181616
Code Geass: Hangyaku no Lelouch     0.178820
Sen to Chihiro no Kamikakushi       0.159963
Angel Beats!                        0.157095
Fullmetal Alchemist                 0.155661
Code Geass: Hangyaku no Lelouch R2  0.153510
Naruto                              0.146125
Elfen Lied                          0.144045
Ouran Koukou Host Club              0.133792
Mirai Nikki (TV)                    0.121388
Toradora!                           0.117803
Howl no Ugoku Shiro                 0.114433
dtype: float64
```

图 4-5. 每个类别最受喜爱的前 15 电影




```
cluster 0
AVG episode : 38.46666666666667
AVG movie rating : 7.866666666666666
AVG member : 668177.0666666667
```

图 4-6. 每个类别的特征

4.4 总结

相比于协同过滤矩阵分解方法和 CNN 特征提取的方法，基于聚类的算法在实现上会更简单，并且一定程度上反应用户对不同动画的喜爱程度，可以挑选出用户评分较高的优质动漫进行推荐。但缺点就是精度不高，作为群体喜欢的动漫可能个人不一定喜欢，没办法达到“个性化”的目的，并且没有充分利用到动漫的属性信息，推荐的结果可能存在较大误差。这也是一种为用户提供差异化推荐体验的替代方法。