FINAL PROJECT

RECOMMENDATION ALGORITHM

Zhejiang University

IS THIS A BIG DATA PROBLEM?

- ✓ Yes. Internet companies, such as Alibaba and Netease, will generate a few hundreds G user log data. If we apply the single machine algorithm, even the memory can hold all data, it may last for several days and you cannot handle the next day's data.
- ✓ Popular solutions adopt the distributed frameworks, such as Hadoop and Spark, to build the model offline and then perform the prediction online.
- ✓ For example, it may cost 5-6 hours to do feature extractions for 2 million data, while it only requires 20 minutes for hadoop.

THE PURPOSE OF FINAL PROJECT

- ✓ To think and not to follow
- ✓ We hope you to be creative and not a following-order person
- ✓ If you pursue Phd, you will find this kind of exercises are very helpful
- ✓ Confucius said, "learning without thinking is blind”

THE DESIGN OF FINAL PROJECT

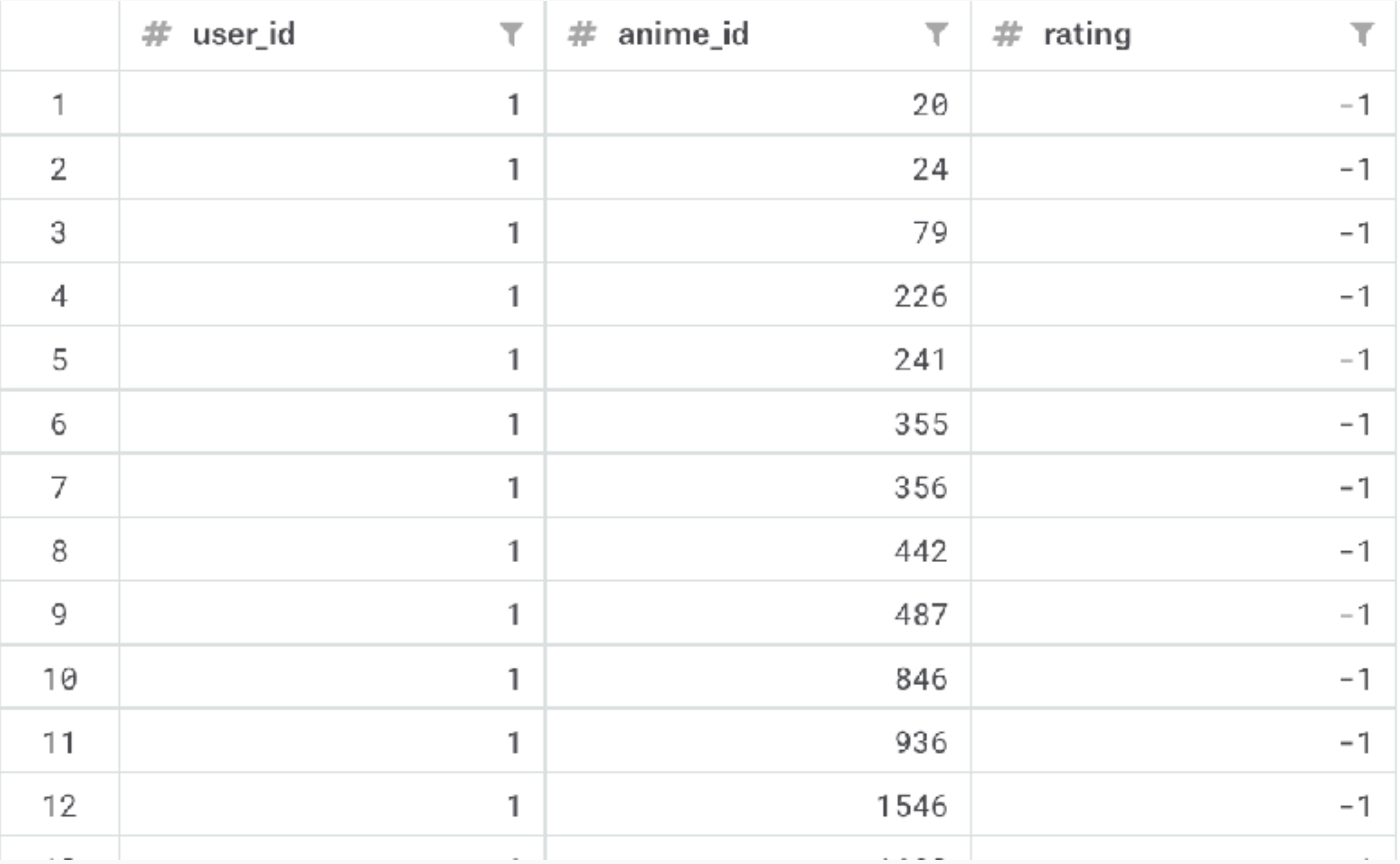
- It cannot be too hard
    - Given a real system requirement (e.g., big data analytic system for e-commerce)
    - Implement one novel big data processing algorithms from recent papers?
- It cannot be too simple
    - It cannot reveal the ability of our students (cannot distinguish us).

ONE PROJECT THAT CAN BE HARD/EASY

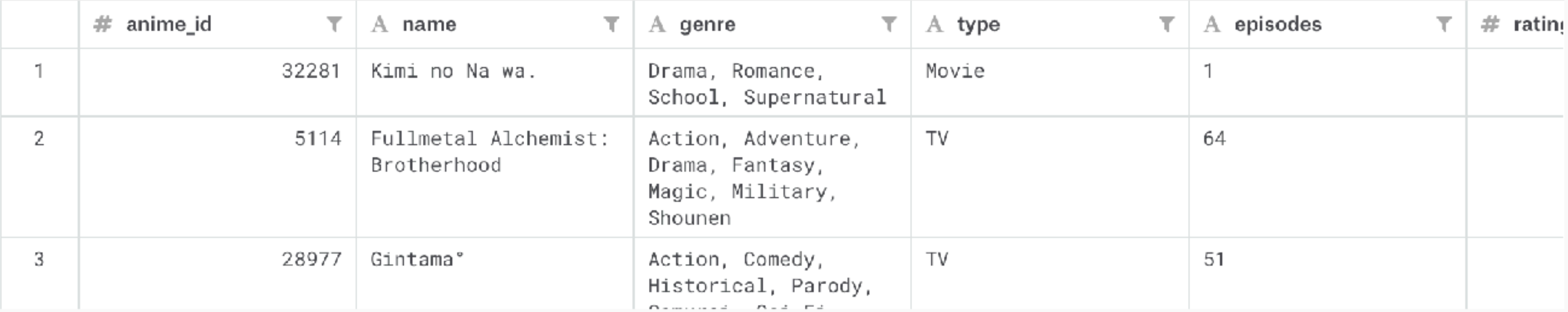
- ✓ Anime recommendations
- ✓ We all love cartoons and anime
- ✓ No optimal solution yet

BASIC DATASET

- Two ranking tables, with 1-10 ranks
- -1 indicates there is no ranking
- Finally, we consider that if you recommend the anime to the user who ranks it above 8, this is a correct recommendation.

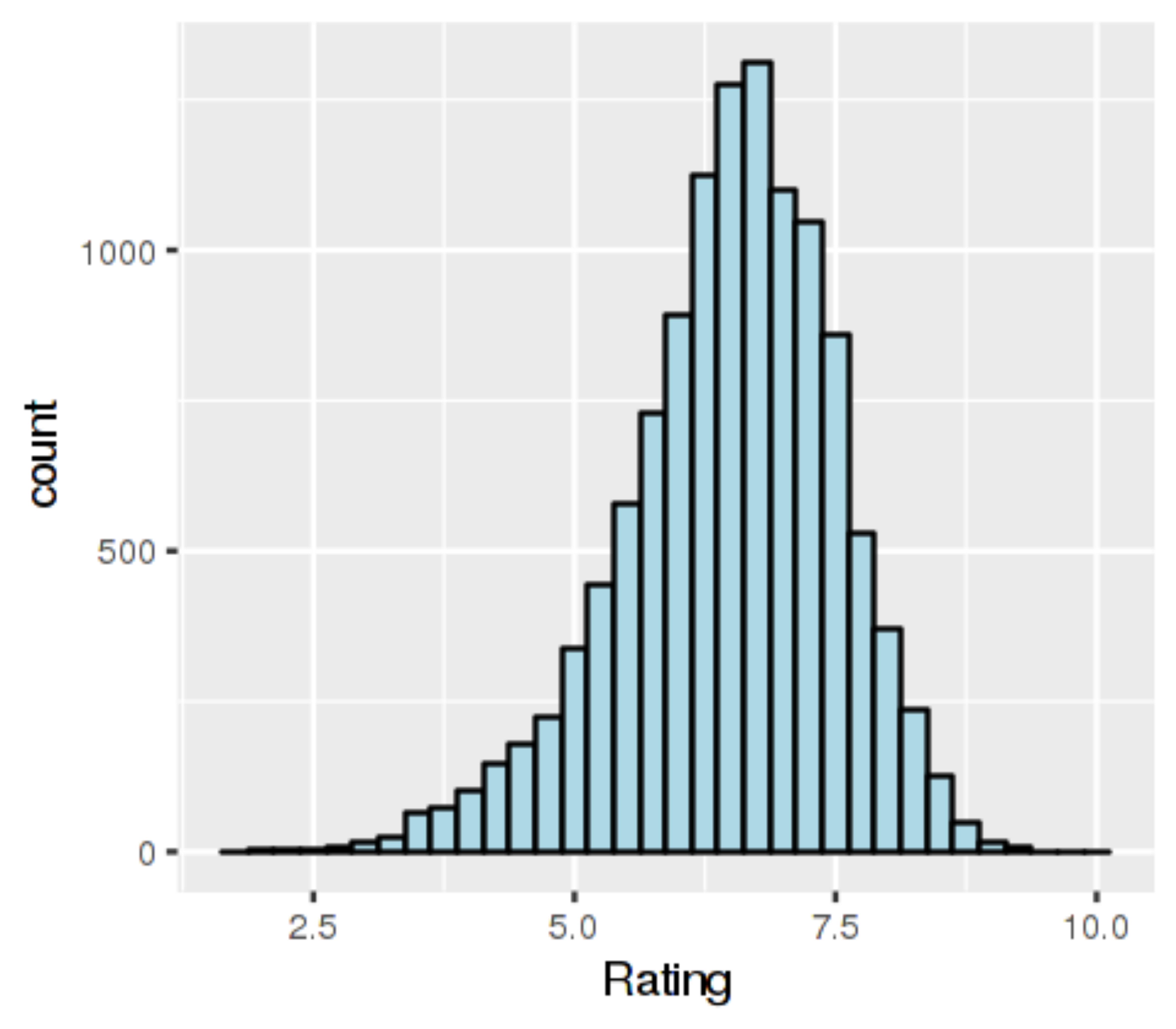


	# user_id	# anime_id	# rating
1	1	20	-1
2	1	24	-1
3	1	79	-1
4	1	226	-1
5	1	241	-1
6	1	355	-1
7	1	356	-1
8	1	442	-1
9	1	487	-1
10	1	846	-1
11	1	936	-1
12	1	1546	-1

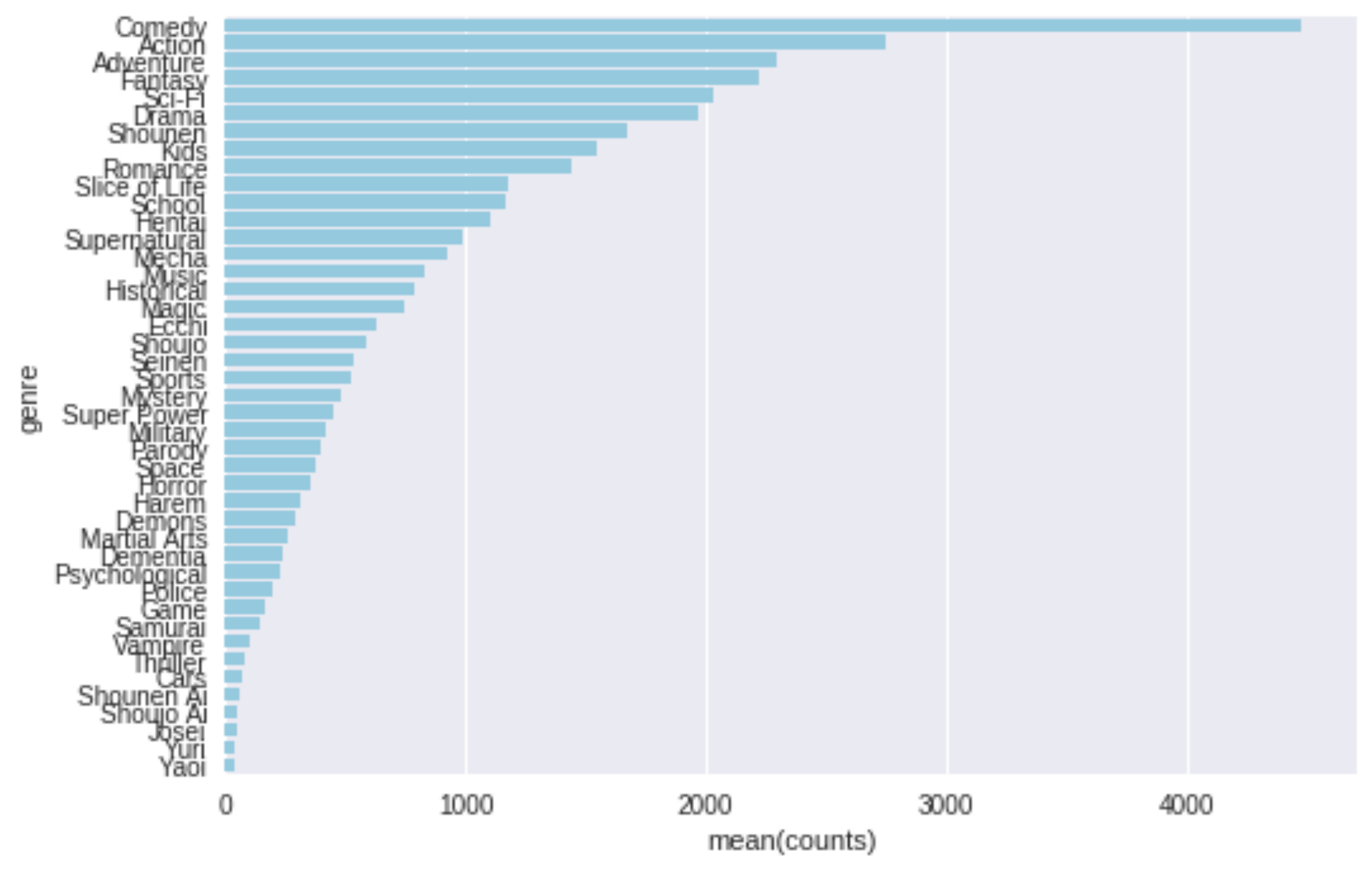


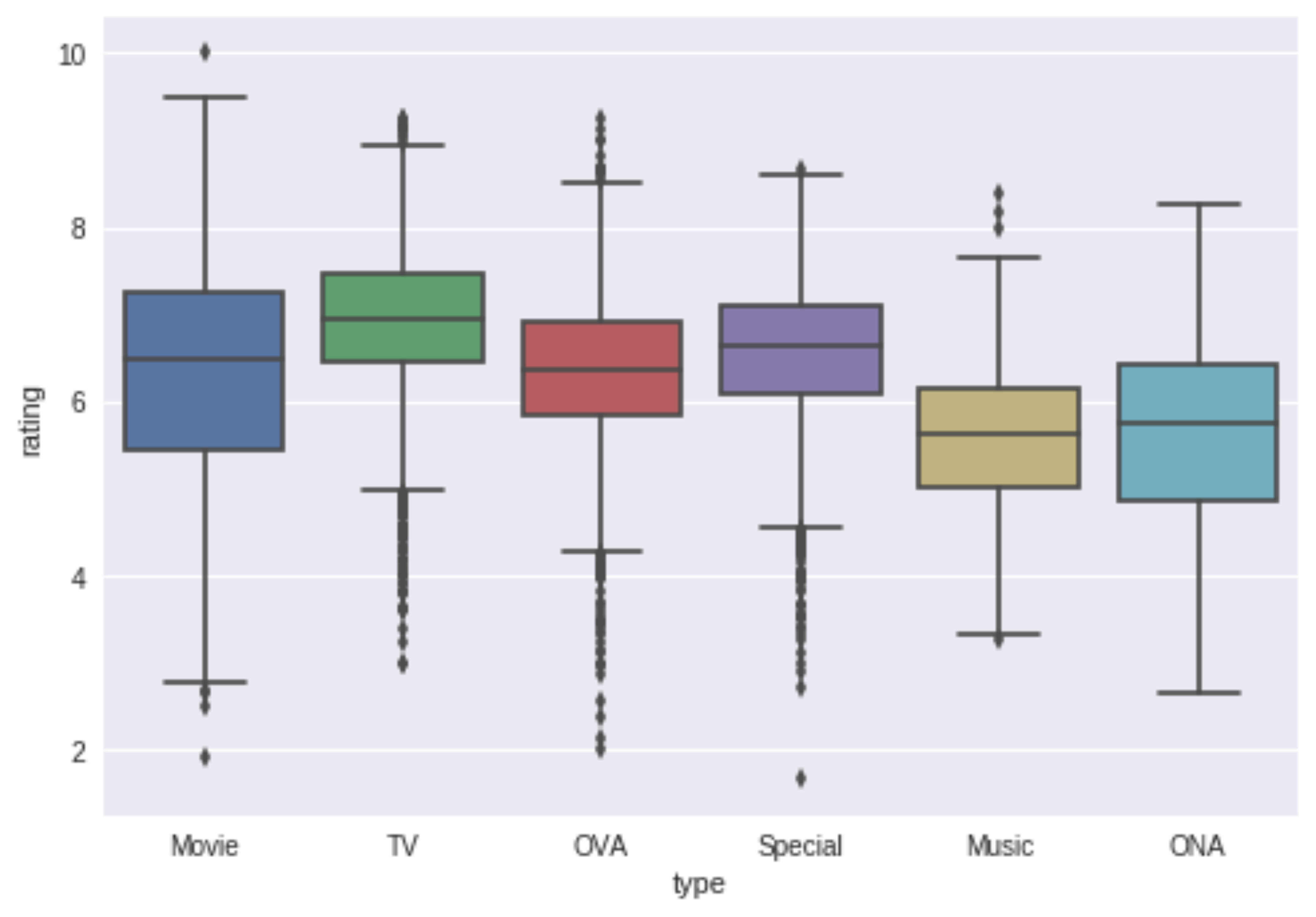
	# anime_id	A name	A genre	A type	A episodes	# ratin
1	32281	Kimi no Na wa.	Drama, Romance, School, Supernatural	Movie	1	
2	5114	Fullmetal Alchemist: Brotherhood	Action, Adventure, Drama, Fantasy, Magic, Military, Shounen	TV	64	
3	28977	Gintama°	Action, Comedy, Historical, Parody,	TV	51	

WHY 8?



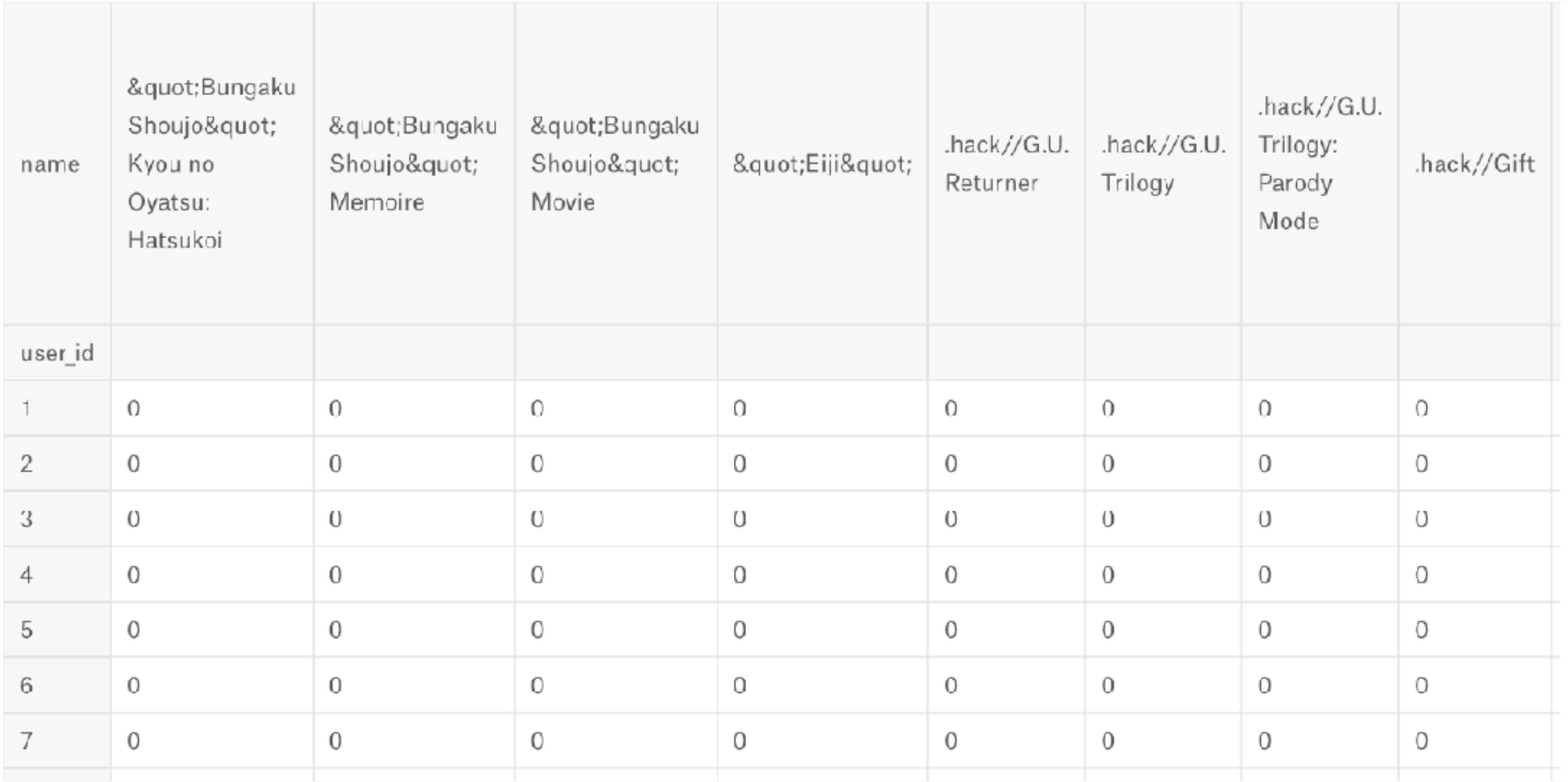
OTHER STATISTICS





POSSIBLE SOLUTION: CLUSTER

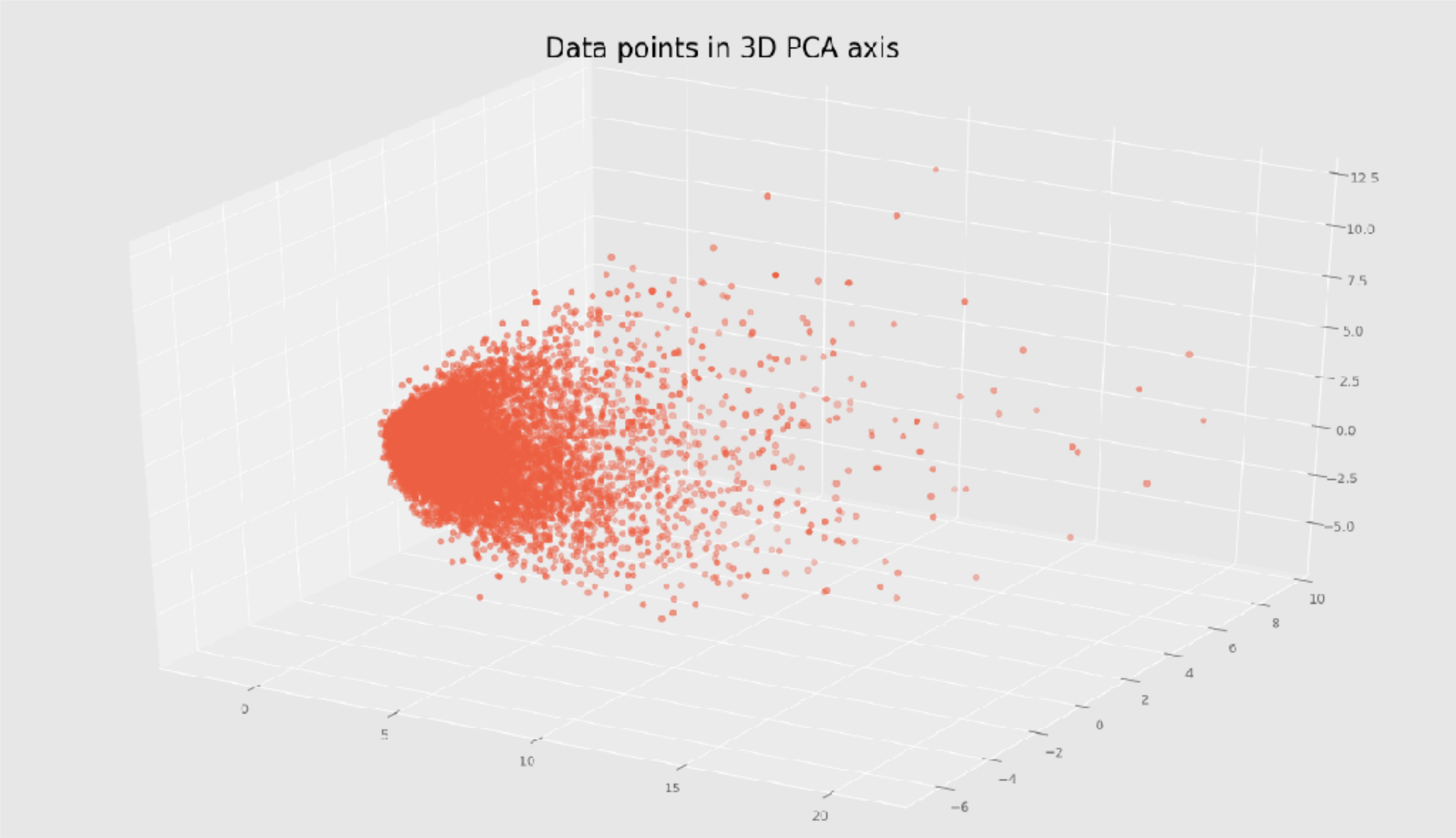
- Merge two tables (need Hadoop/Spark to do the table join)， and get the user-anime （user_id, anime_name） matrix， 0 denotes dislike， 1 = like
- this will result in a very large U*N matrix！！！



name	&quot;Bungaku Shoujo&quot; Kyou no Oyatsu: Hatsukoi	&quot;Bungaku Shoujo&quot; Memoire	&quot;Bungaku Shoujo&quot; Movie	&quot;Eiji&quot;	.hack//G.U. Returner	.hack//G.U. Trilogy	.hack//G.U. Trilogy: Parody Mode	.hack//Gift
user_id								
1	0	0	0	0	0	0	0	0
2	0	0	0	0	0	0	0	0
3	0	0	0	0	0	0	0	0
4	0	0	0	0	0	0	0	0
5	0	0	0	0	0	0	0	0
6	0	0	0	0	0	0	0	0
7	0	0	0	0	0	0	0	0

POSSIBLE SOLUTION: CLUSTER

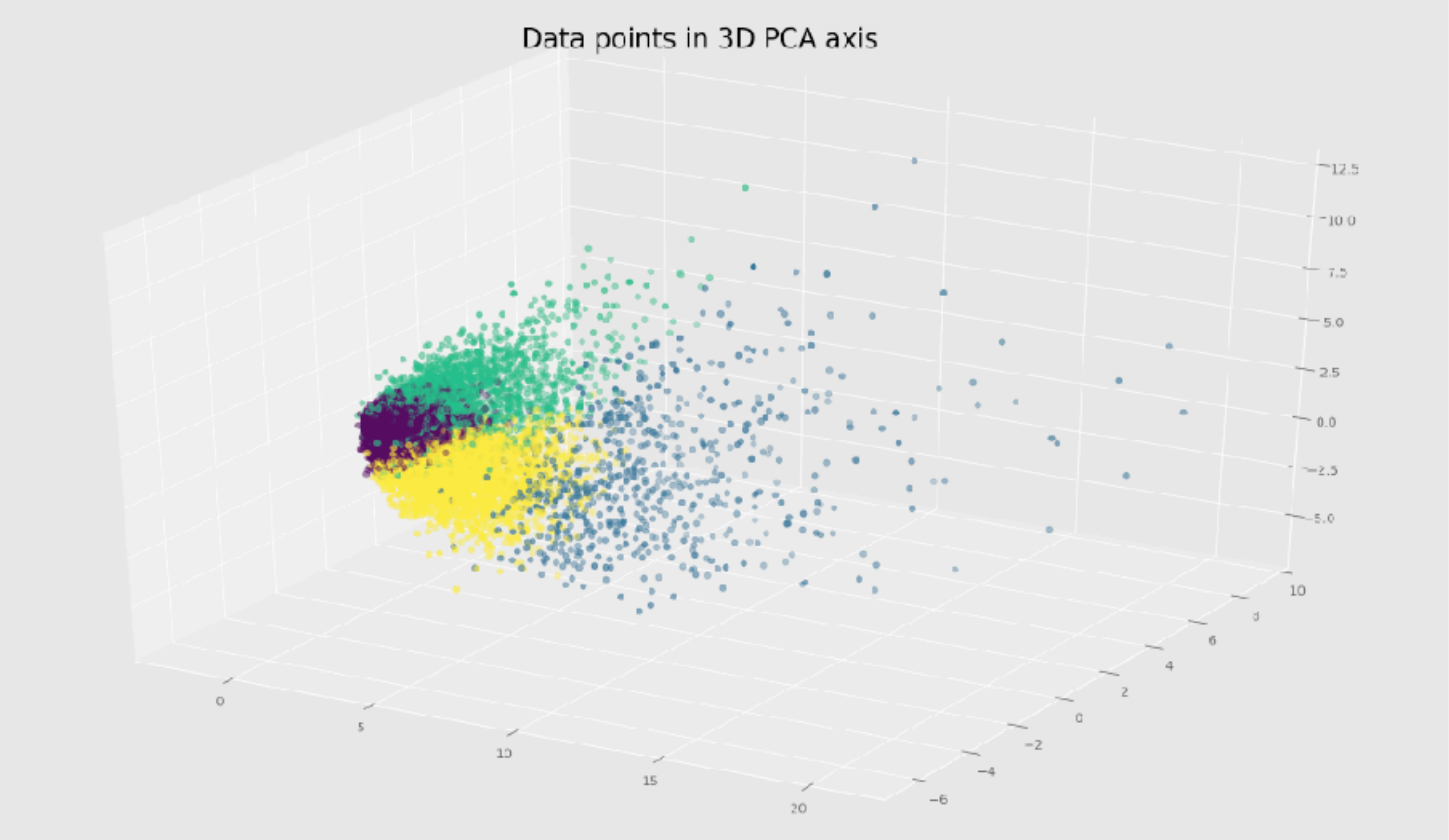
- We need the dimensional reduction
- What we have learned in the class: PCA, SVM
- But how many dimensions should we keep??


Data points in 3D PCA axis

POSSIBLE SOLUTION: CLUSTER

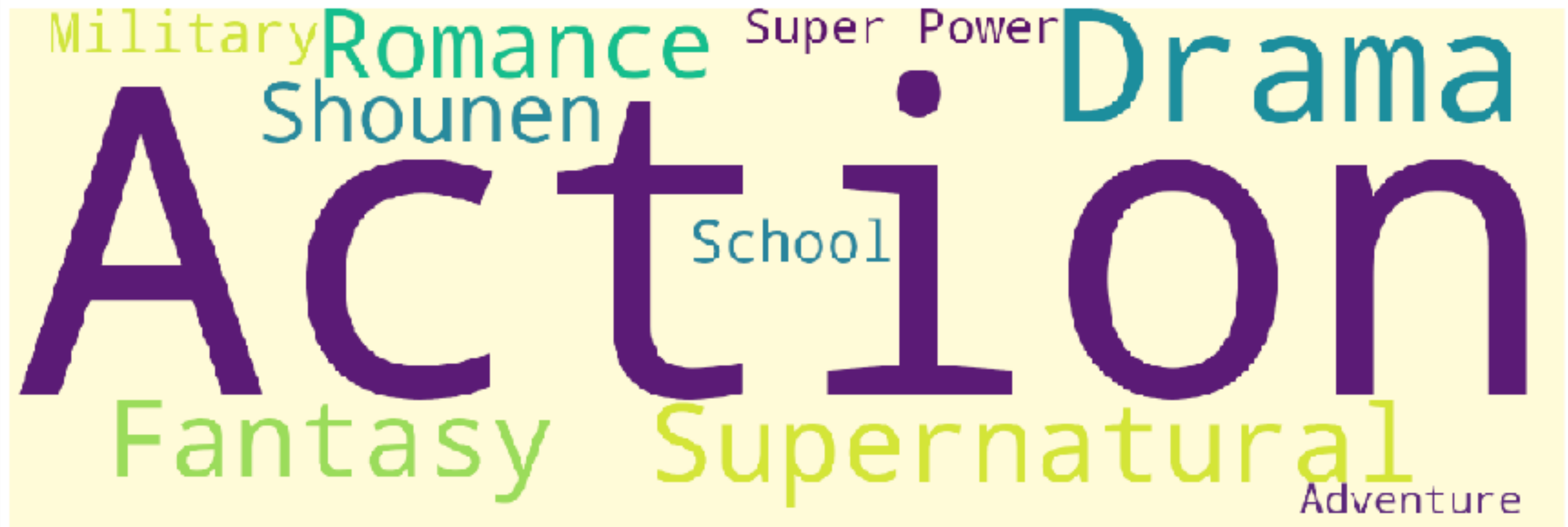
- How to decide the number of K in Means? kernel = 4
- a(i) = the average distance from element i to others in the same cluster
- b(i) = the minimal average distance from element i to other elements in different clusters

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$



POSSIBLE SOLUTION: CLUSTER

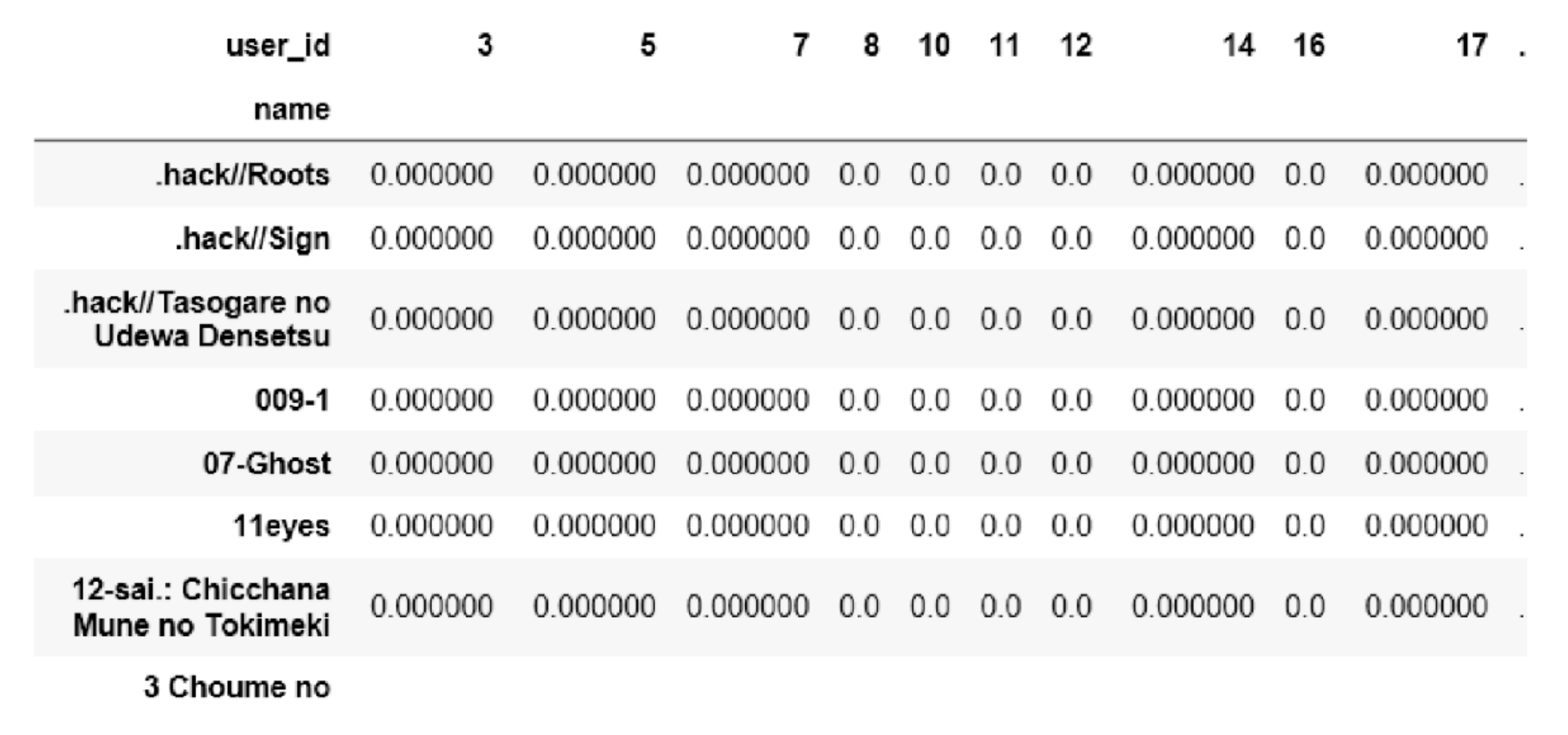
- ✓ Specifically, each anime has its genre. If we use the highest frequency keywords for the genre, we get the topics for a genre like:



- ✓ For other attributes (episodes, ratings, members) , we use the average values as the cluster's attribute. So each anime can be represented as (genre, episodes, ratings, members)
- ✓ Recommend the anime with minimal distance to the user's high ranked anime's cluster

POSSIBLE SOLUTION: CF

✓ Set up the User-Anime matrix



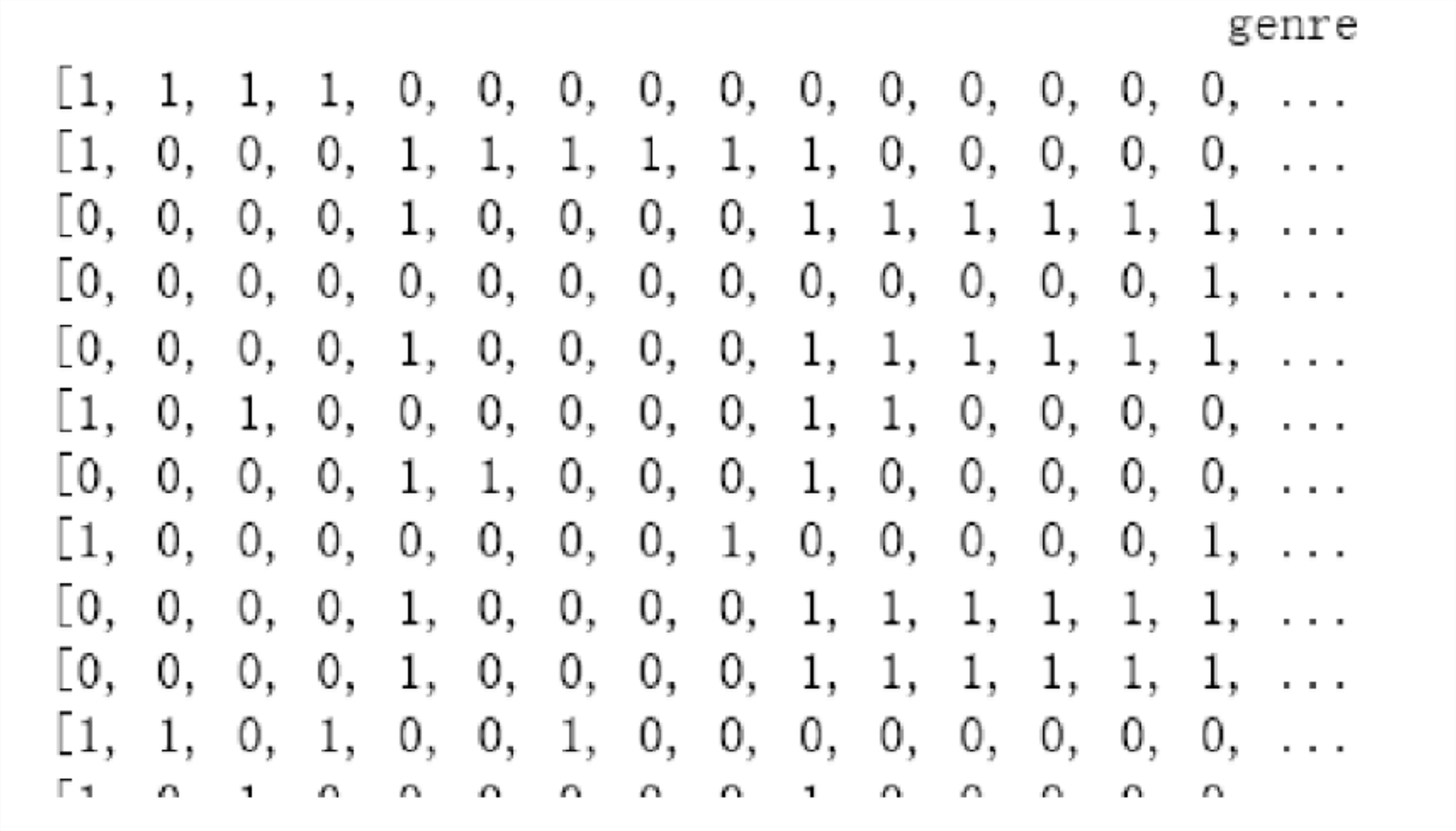
user_id	3	5	7	8	10	11	12	14	16	17	.
name											
.hack//Roots	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	.
.hack//Sign	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	.
.hack//Tasogare no Udewa Densetsu	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	.
009-1	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	.
07-Ghost	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	.
11eyes	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	.
12-sai.: Chicchana Mune no Tokimeki	0.000000	0.000000	0.000000	0.0	0.0	0.0	0.0	0.000000	0.0	0.000000	.
3 Choume no											

POSSIBLE SOLUTION: CF

- ✓ Estimate the cosine distance between anime and do the CF recommendations
- ✓ item/user-based and SVD based
- ✓ item/user-based： compute the user-user distance or anime-anime distance。
- ✓ SVD based： do the SVD and obtain two matrixes represent user to hidden values and hidden values to anime. Use linear regression and the two matrixes to fill in the missing values。

POSSIBLE SOLUTION: CONTENT-BASED RECOMMENDATION

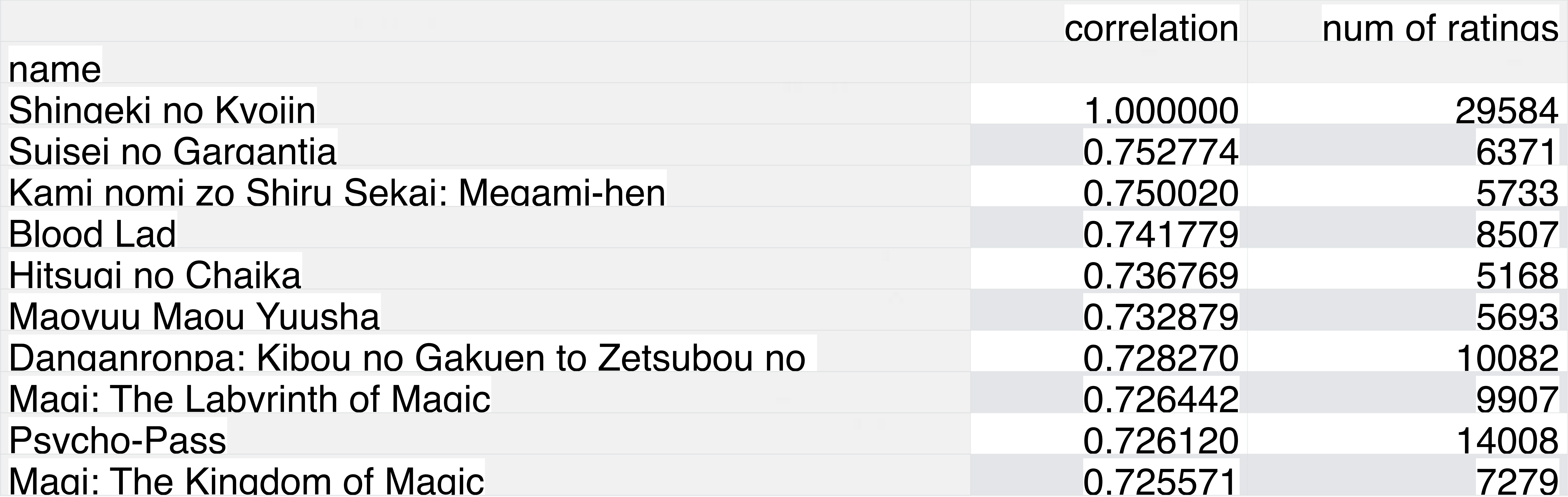
- Based on the genre attribute of the anime, recommend users the anime with similar genre. Other attributes such as ranking score and ranking times are all possible features.
- Set up a anime-genre matrix and estimate anime-anime similarities:



```
                                                        genre
[1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ...
[1, 0, 0, 0, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, ...
[0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, ...
[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, ...
[0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, ...
[1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, ...
[0, 0, 0, 0, 1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, ...
[1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, ...
[0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, ...
[0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, ...
[1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, ...
[1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0
```

POSSIBLE SOLUTION: CONTENT-BASED

- Recommend the top 10 anime based on their similarities



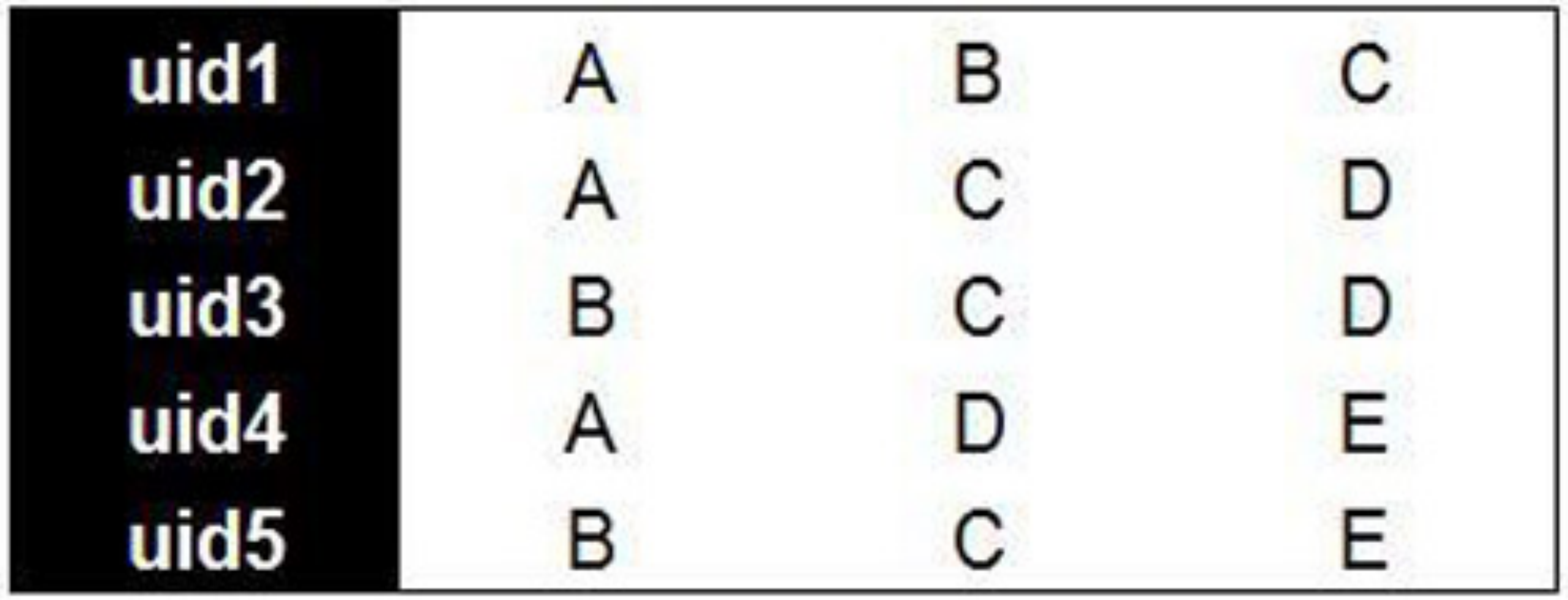
	correlation	num of ratings
name		
Shingeki no Kyojin	1.000000	29584
Suisei no Gargantia	0.752774	6371
Kami nomi zo Shiru Sekai: Megami-hen	0.750020	5733
Blood Lad	0.741779	8507
Hitsugi no Chaika	0.736769	5168
Maoyuu Maou Yuusha	0.732879	5693
Danganronpa: Kibou no Gakuen to Zetsubou no	0.728270	10082
Magi: The Labyrinth of Magic	0.726442	9907
Psycho-Pass	0.726120	14008
Magi: The Kingdom of Magic	0.725571	7279

POSSIBLE SOLUTION: CONTENT-BASED

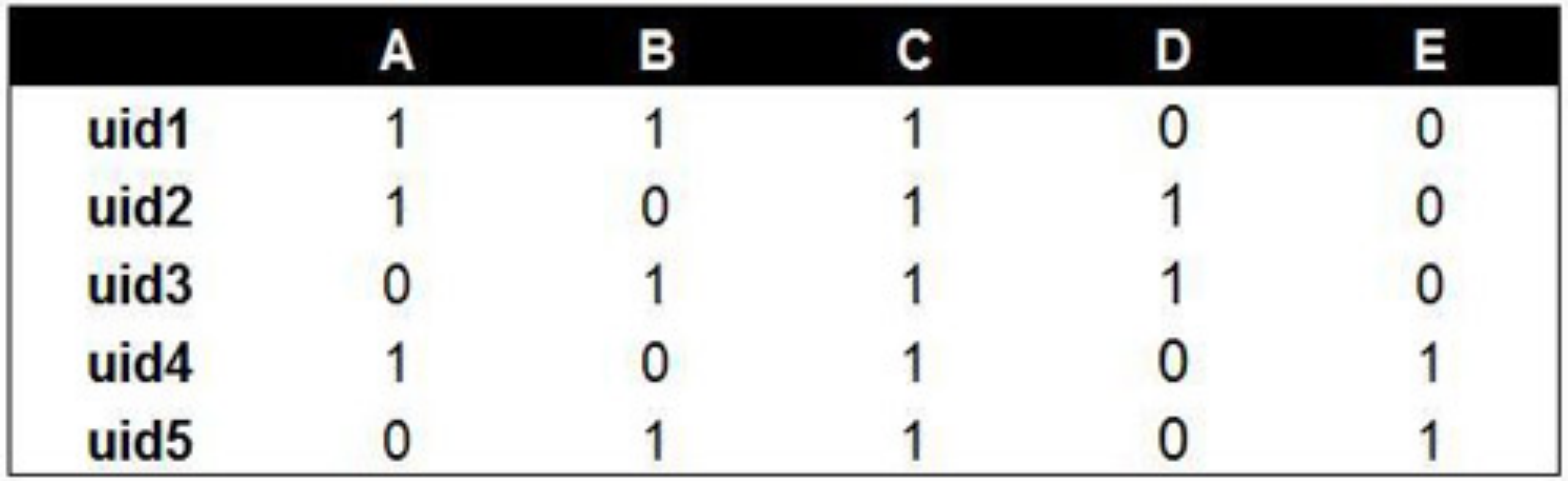
- Decision Tree based Solution
- Generate the history for a user:
  - User History=(Like, Dislike)
  - Like= anime with rank>=8
  - Dislike= anime rank<8
- Set up a decision tree for every user?

Possible Solution: Association Rule

- What is the correlation analysis, beer and diapers
- Based on the users' view history, infer his/her future selections.



uid1	A	B	C
uid2	A	C	D
uid3	B	C	D
uid4	A	D	E
uid5	B	C	E



	A	B	C	D	E
uid1	1	1	1	0	0
uid2	1	0	1	1	0
uid3	0	1	1	1	0
uid4	1	0	1	0	1
uid5	0	1	1	0	1

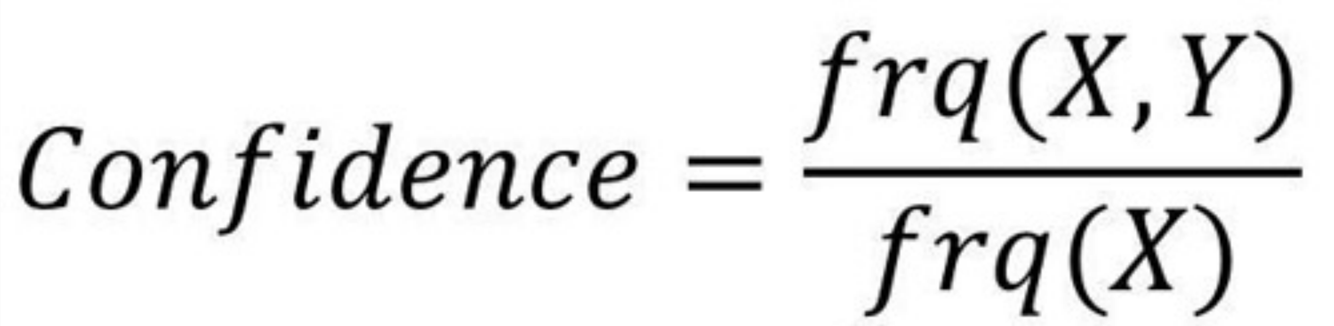
POSSIBLE SOLUTION: ASSOCIATION RULE

- 支持度(Support) 支持度是两件商品在所有购物车中同时出现的概率，可以记录为P(A U B)
- Support = two movies viewed by the same user, denoted as P(AUB)
- So for previous example, P(A U B) = 1/5

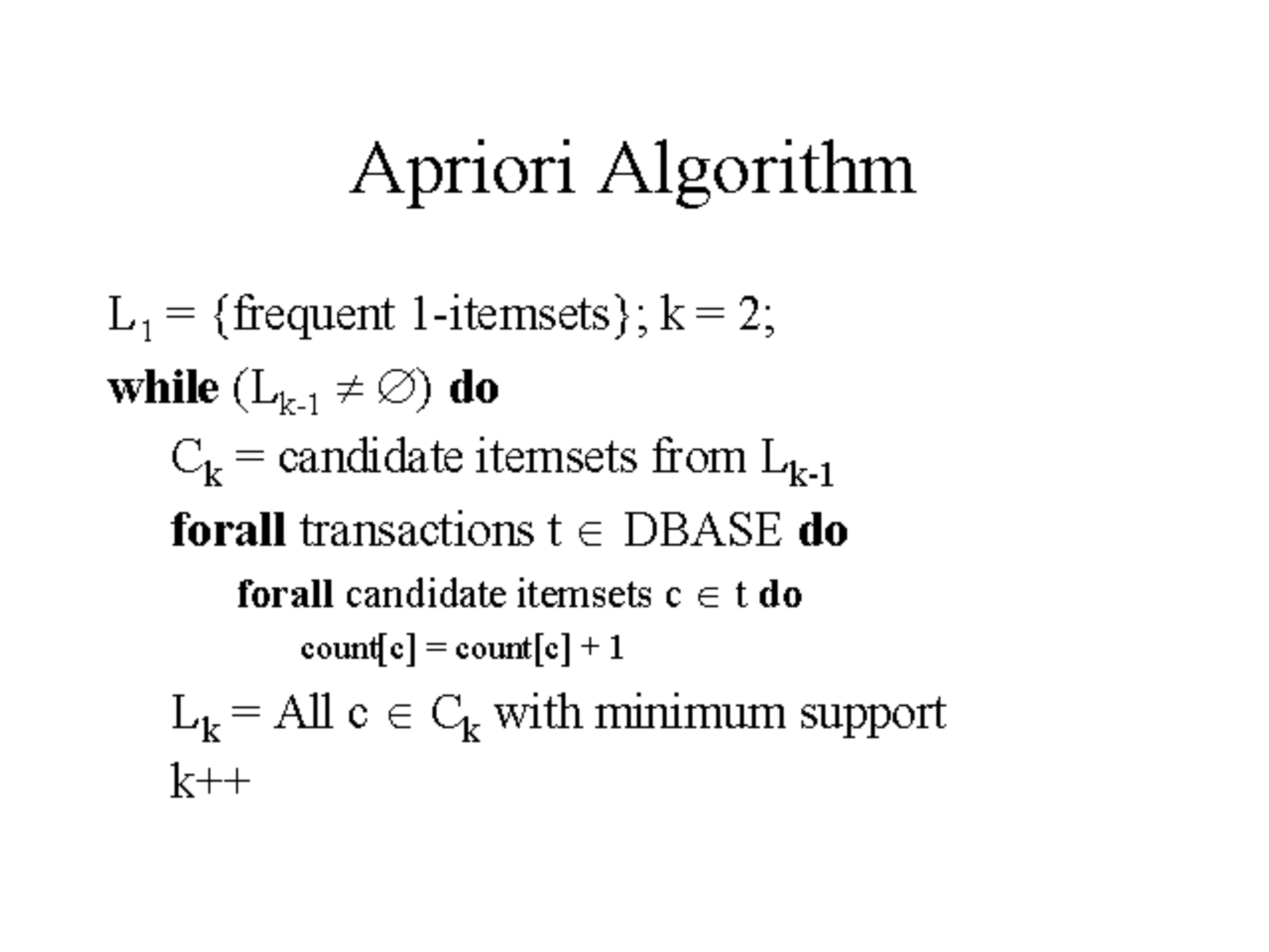
$$Support = \frac{frq(X,Y)}{N}$$

POSSIBLE SOLUTION: ASSOCIATION RULE

- 置信度(confidence)是一个条件概率，两件商品其中一件出现在购物车中时，另一件也会出现的概率。可以记录为P(B|A)
- Confidence = If A is selected, B is also picked. Denoted as P(B|A)
- In previous example, P(B|A)=1/3



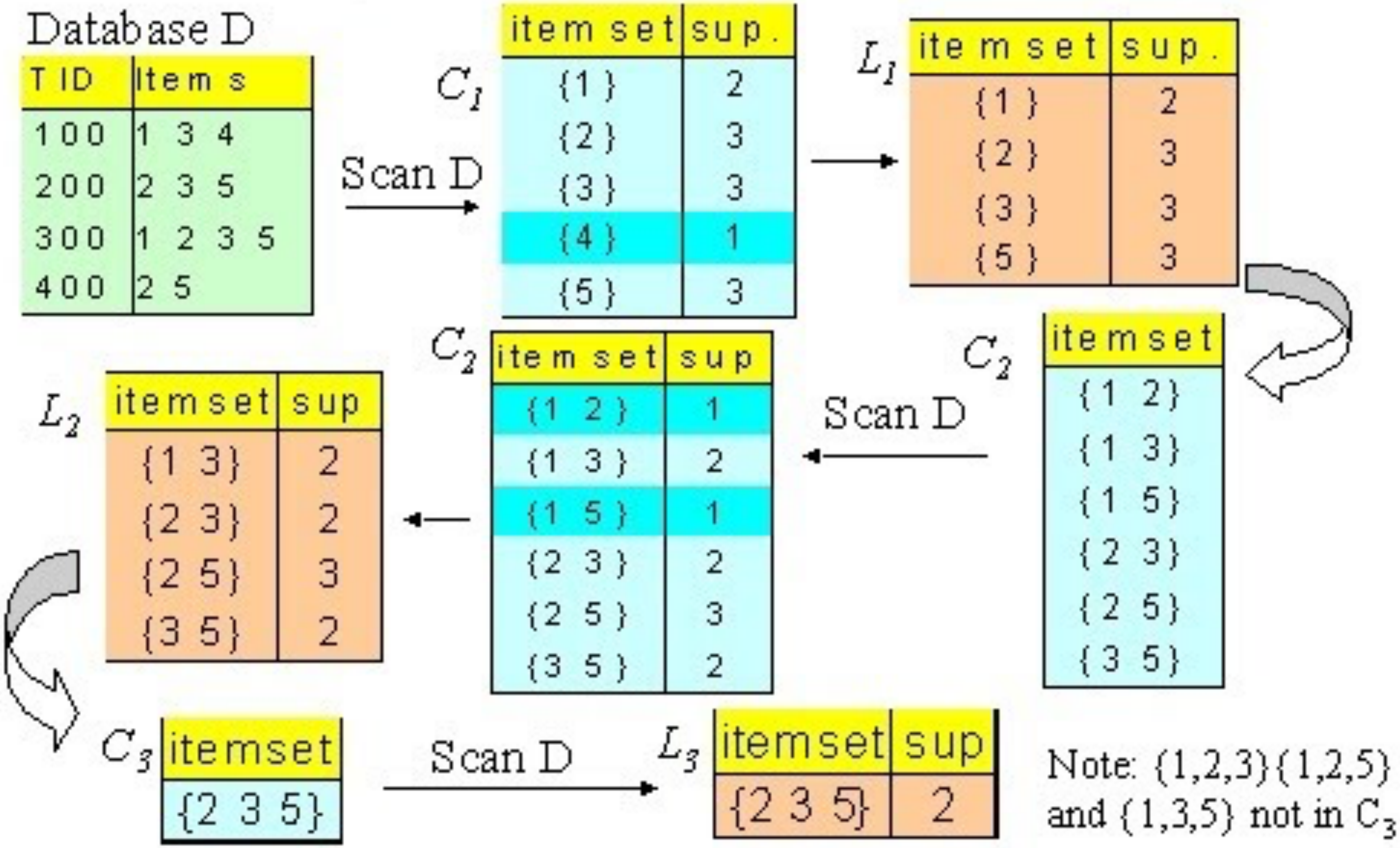
$$Confidence = \frac{frq(X,Y)}{frq(X)}$$



Apriori Algorithm

```
$L_1$ = {frequent 1-itemsets}; k = 2;
while ($L_{k-1} \neq \emptyset$) do
    $C_k$ = candidate itemsets from $L_{k-1}$
    forall transactions t $\in$ DBASE do
        forall candidate itemsets c $\in$ t do
            count[c] = count[c] + 1
    $L_k$ = All c $\in$ $C_k$ with minimum support
    k++
```

The Apriori Algorithm -- Example



Database D

TID	Items
100	1 3 4
200	2 3 5
300	1 2 3 5
400	2 5

Scan D

C_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{4}	1
{5}	3

L_1

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

C_2

itemset
{1 2}
{1 3}
{1 5}
{2 3}
{2 5}
{3 5}

Scan D

C_2

itemset	sup
{1 2}	1
{1 3}	2
{1 5}	1
{2 3}	2
{2 5}	3
{3 5}	2

L_2

itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

C_3

itemset
{2 3 5}

Scan D

L_3

itemset	sup
{2 3 5}	2

Note: {1,2,3}{1,2,5} and {1,3,5} not in C_3

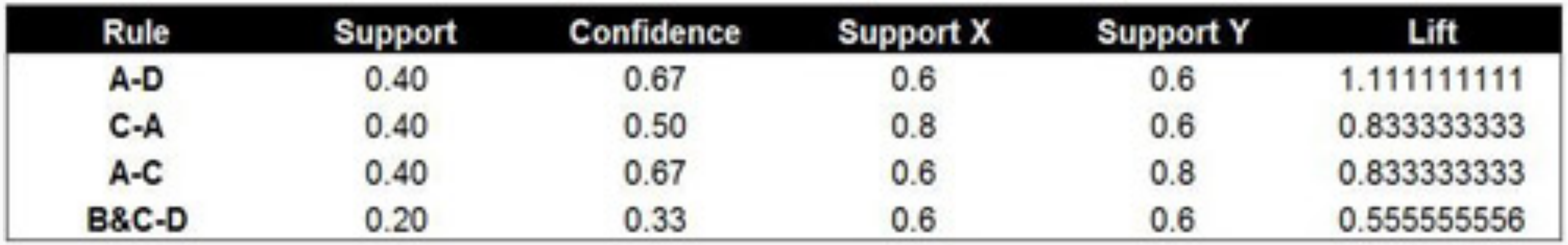
POSSIBLE SOLUTION: ASSOCIATION RULE

- The lift value of an association rule is the ratio of the confidence of the rule and the expected confidence of the rule
- Lift>1 indicates that the rule is good. Otherwise, the rule is not working.In previous example, to infer if A=>B is valid， we have: (1/5)/(3/5*3/5)=(0.2)/(0.6*0.6)=0.2/0.36=0.55。So A=>B does not work for B

$$Lift = \frac{Support}{Supp(X) \times Supp(Y)}$$

POSSIBLE SOLUTION: ASSOCIATION RULE

- How to recommend?
- So we estimate all the support, confidence and lift for the rules。



Rule	Support	Confidence	Support X	Support Y	Lift
A-D	0.40	0.67	0.6	0.6	1.111111111
C-A	0.40	0.50	0.8	0.6	0.833333333
A-C	0.40	0.67	0.6	0.8	0.833333333
B&C-D	0.20	0.33	0.6	0.6	0.555555556

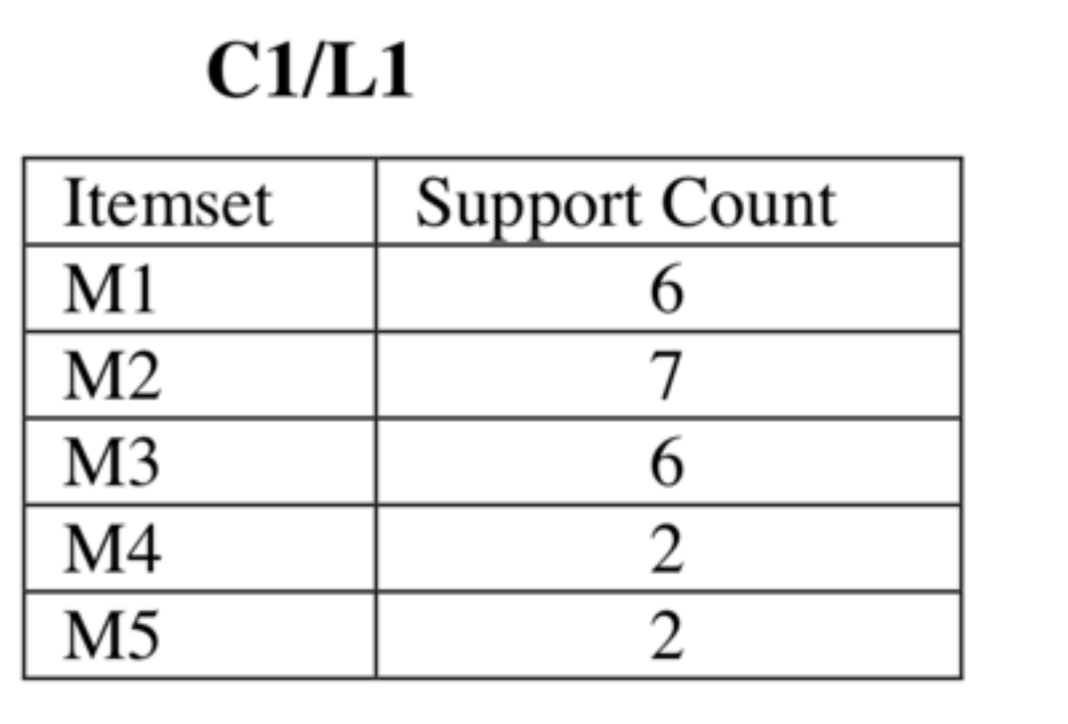
EXERCISE

Transaction id	Items
t1	{1, 2, 4, 5}
t2	{2, 3, 5}
t3	{1, 2, 4, 5}
t4	{1, 2, 3, 5}
t5	{1, 2, 3, 4, 5}
t6	{2, 3, 4}

表格 1

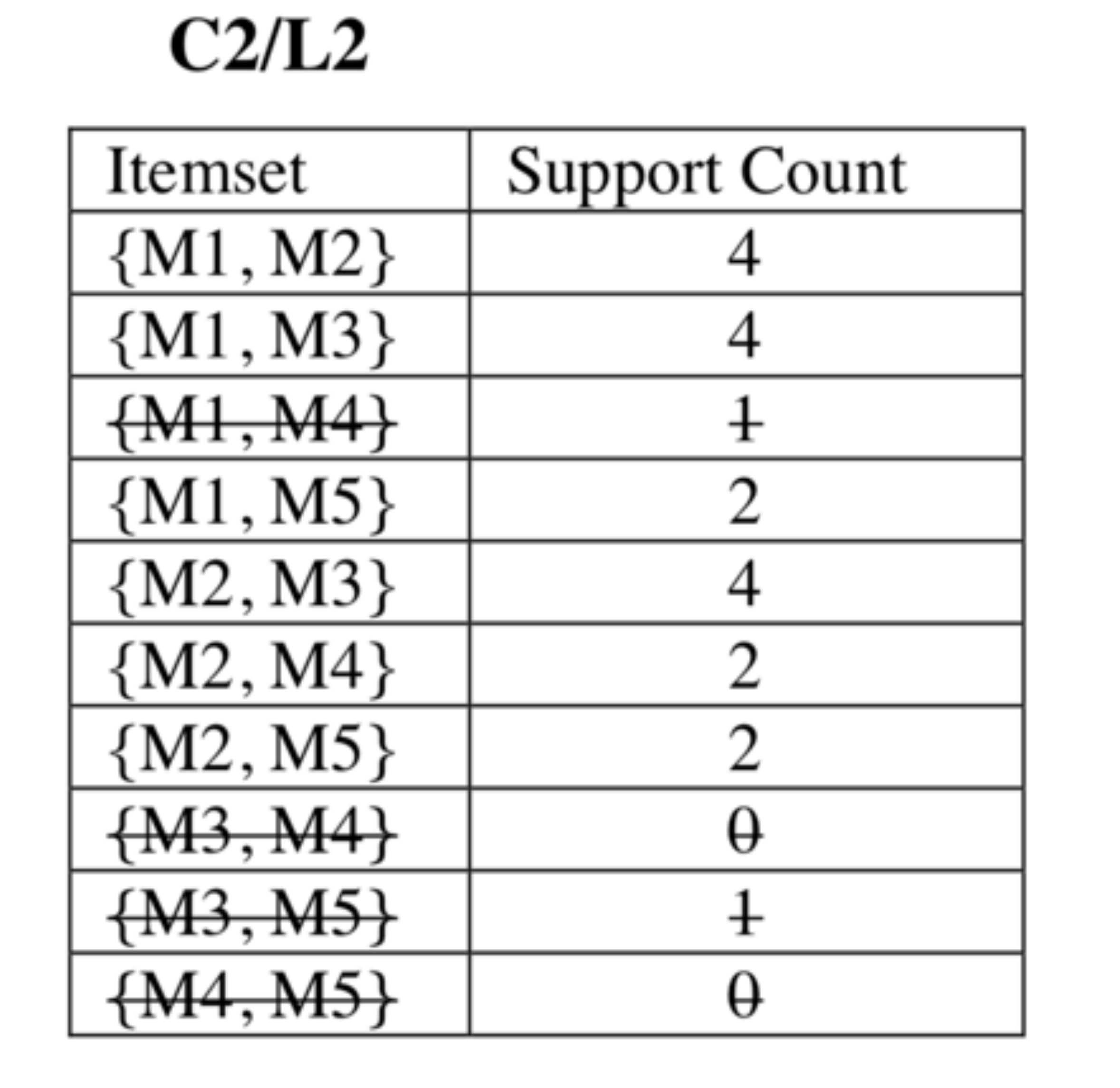
5	==>	2	support	:	0.833(5/6)	confidence	:	1.0 lift	:	1.0
4 5	== >	2	support	:	0.5 (3/6)	confidence	:	1.0 lift	:	1.0
4 5	== >	1 2	support	:	0.5 (3/6)	confidence	:	1.0 lift	:	1.5

EXERCISE



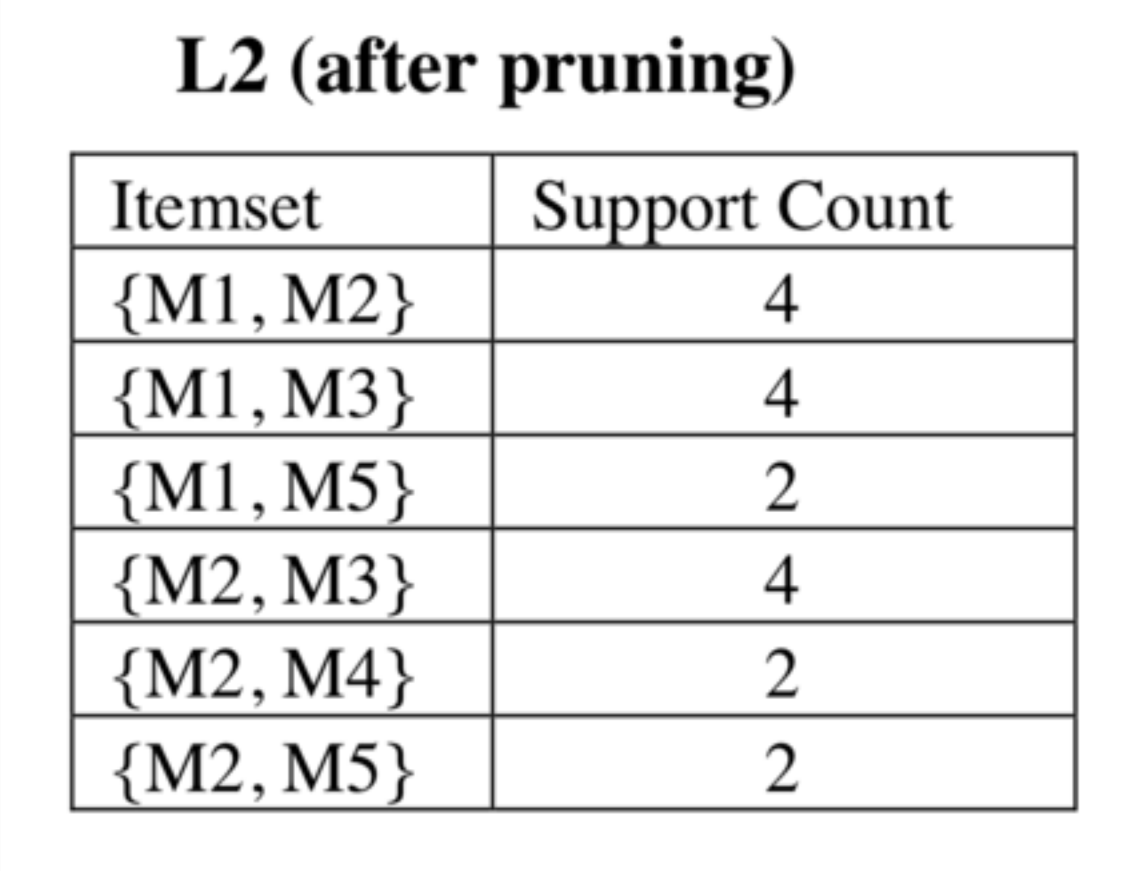
C1/L1

Itemset	Support Count
M1	6
M2	7
M3	6
M4	2
M5	2



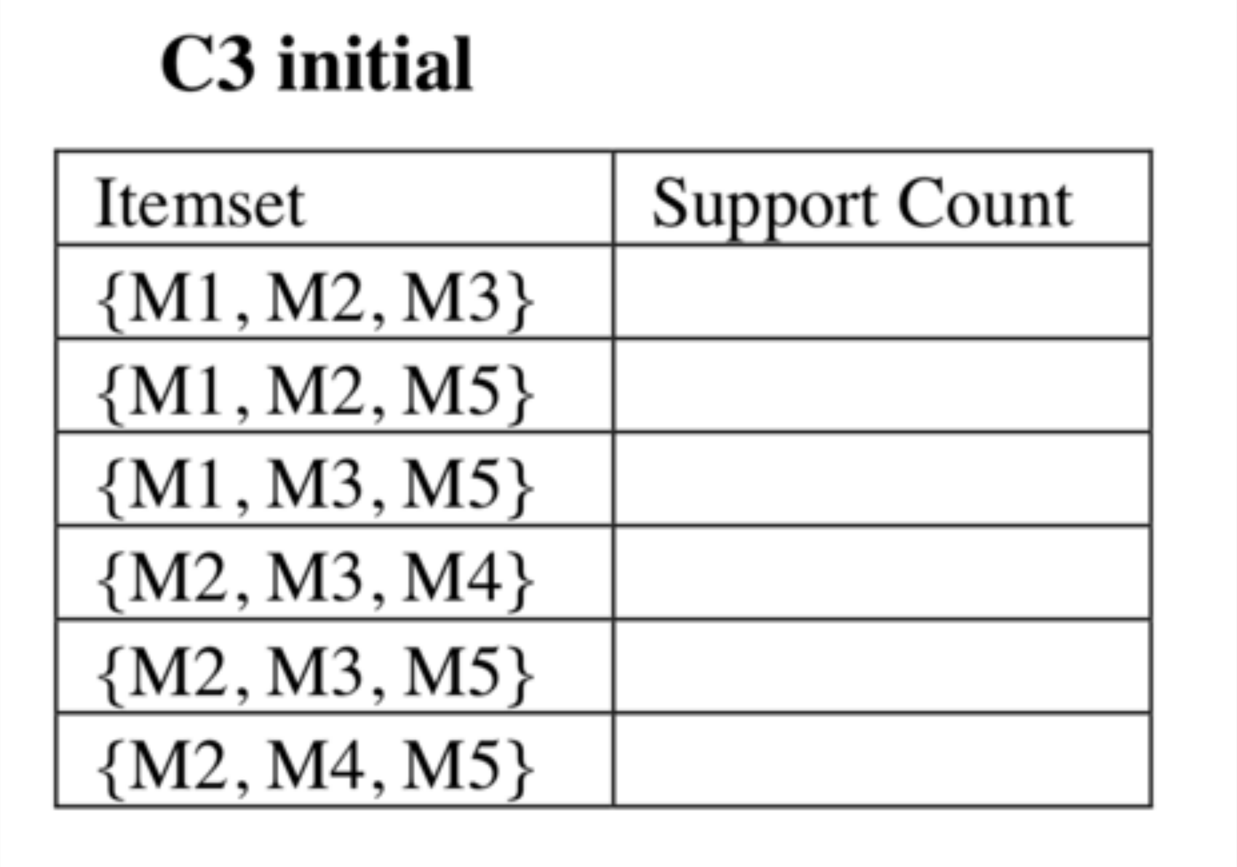
C2/L2

Itemset	Support Count
{M1, M2}	4
{M1, M3}	4
~~{M1, M4}~~	~~1~~
{M1, M5}	2
{M2, M3}	4
{M2, M4}	2
{M2, M5}	2
~~{M3, M4}~~	~~0~~
~~{M3, M5}~~	~~1~~
~~{M4, M5}~~	~~0~~



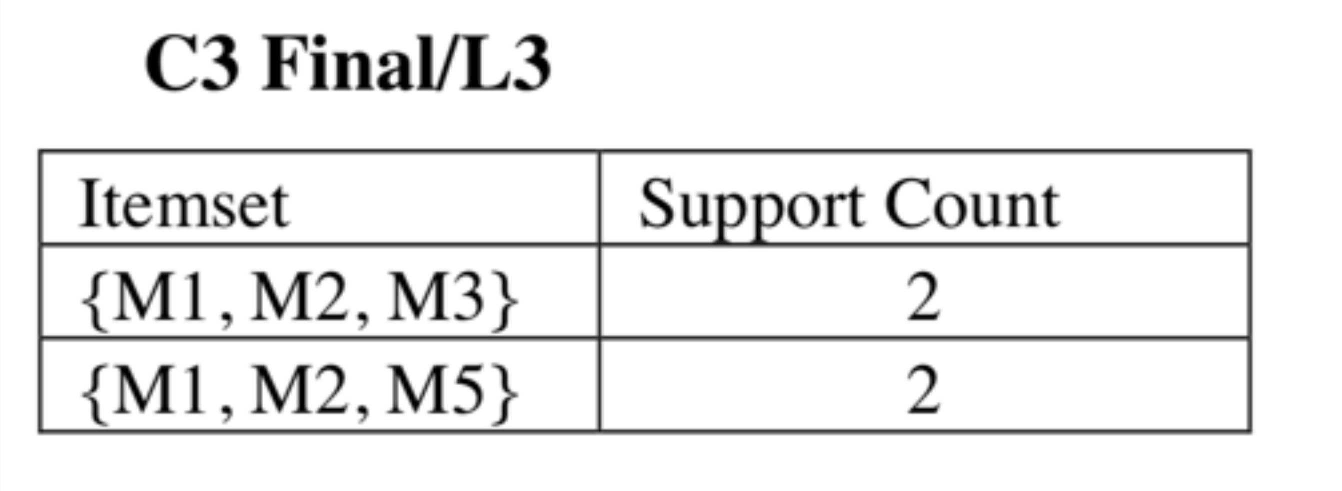
L2 (after pruning)

Itemset	Support Count
{M1, M2}	4
{M1, M3}	4
{M1, M5}	2
{M2, M3}	4
{M2, M4}	2
{M2, M5}	2



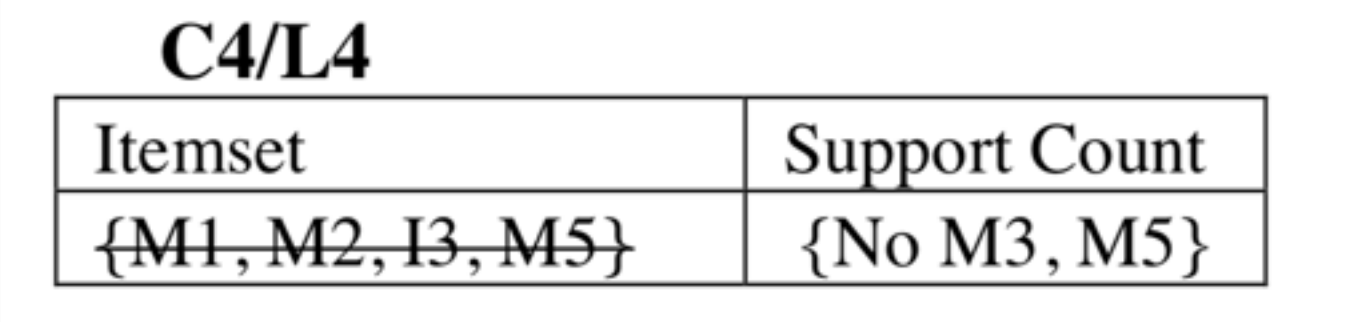
C3 initial

Itemset	Support Count
{M1, M2, M3}	
{M1, M2, M5}	
{M1, M3, M5}	
{M2, M3, M4}	
{M2, M3, M5}	
{M2, M4, M5}	



C3 Final/L3

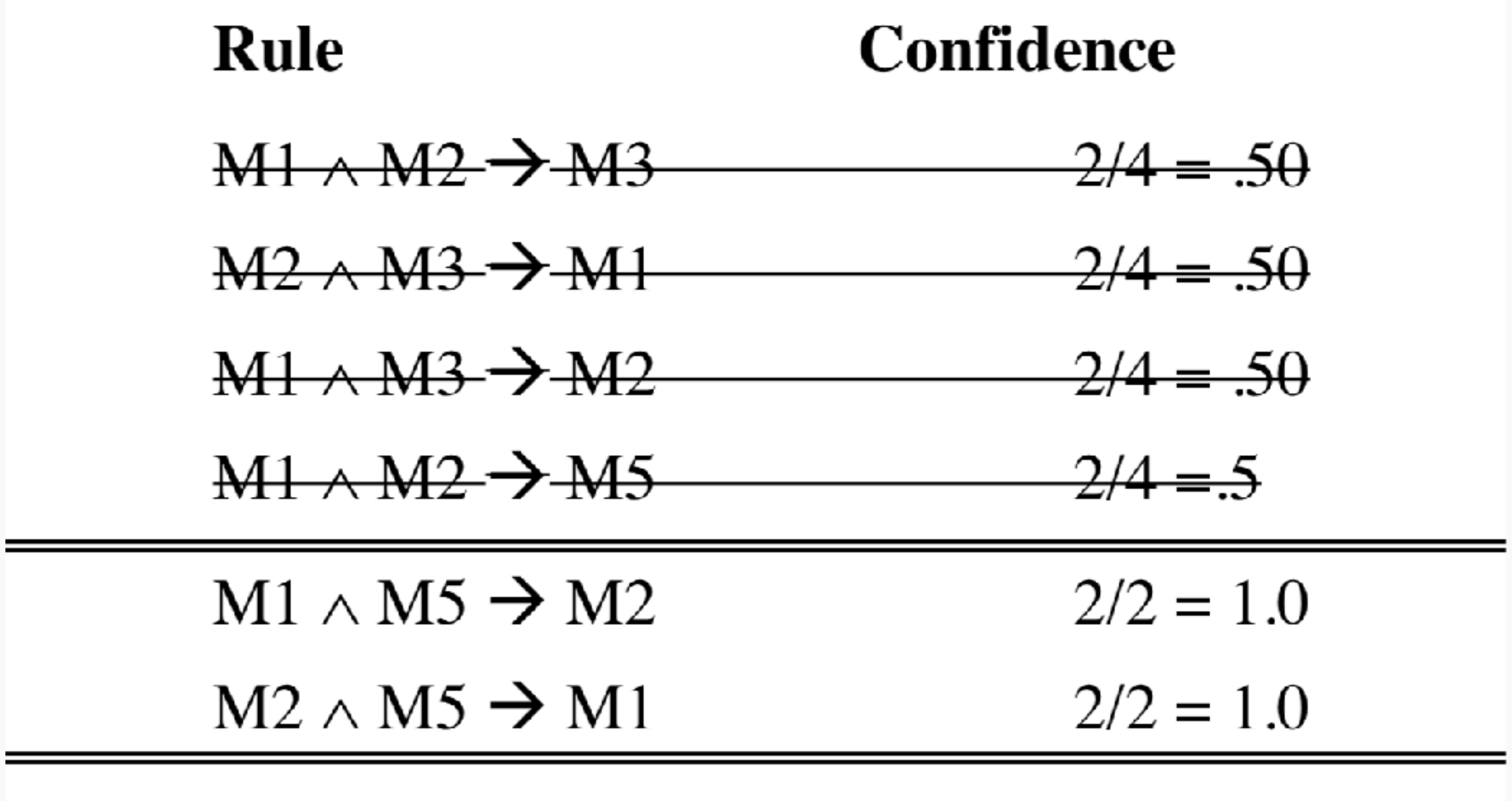
Itemset	Support Count
{M1, M2, M3}	2
{M1, M2, M5}	2



C4/L4

Itemset	Support Count
~~{M1, M2, I3, M5}~~	{No M3, M5}

EXERcISE



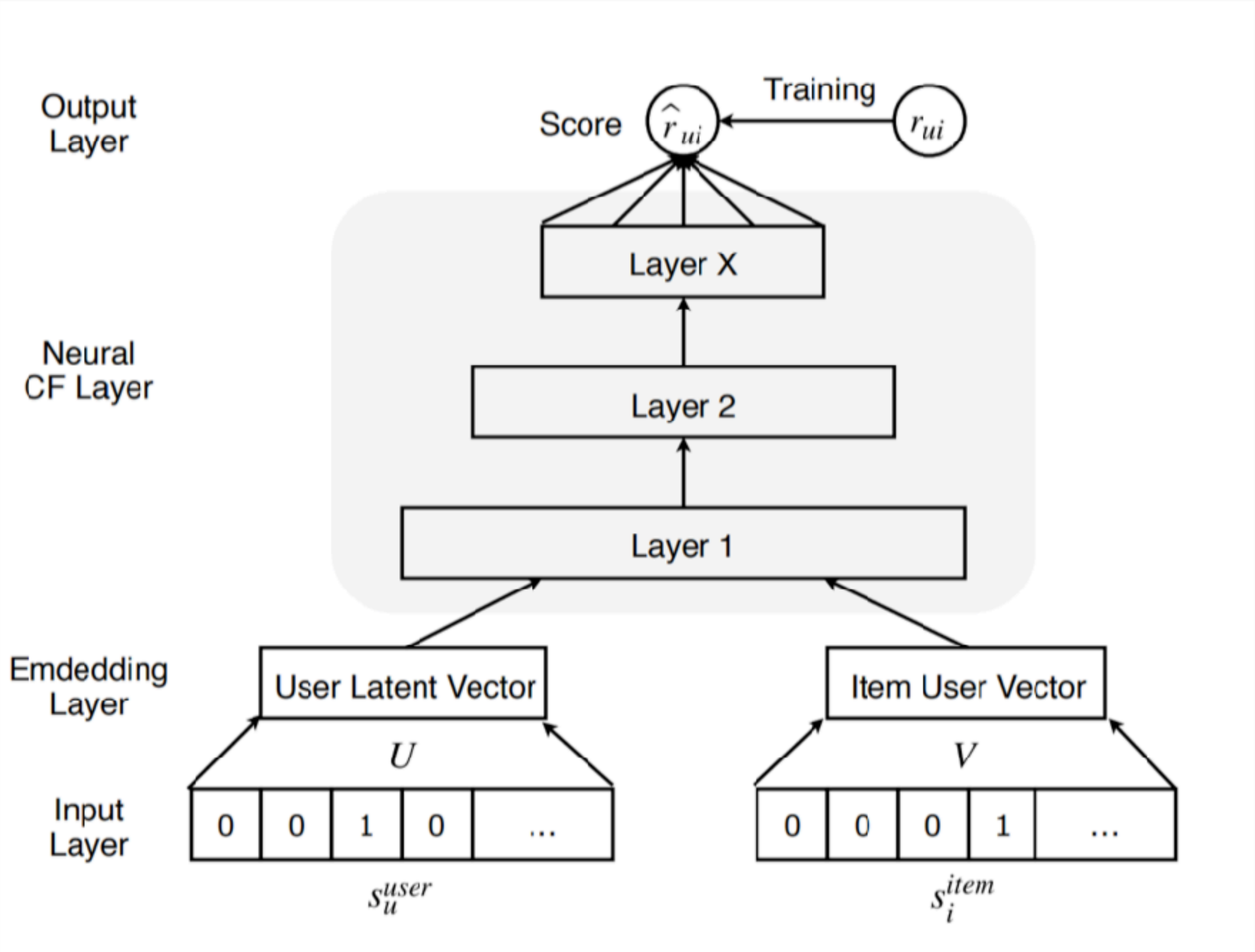
Rule	Confidence
~~M1 ∧ M2 → M3~~	~~2/4 = .50~~
~~M2 ∧ M3 → M1~~	~~2/4 = .50~~
~~M1 ∧ M3 → M2~~	~~2/4 = .50~~
~~M1 ∧ M2 → M5~~	~~2/4 =.5~~
M1 ∧ M5 → M2	2/2 = 1.0
M2 ∧ M5 → M1	2/2 = 1.0

POSSIBLE SOLUTION: MIXED MODELS

- ✓ CF+Context based algorithm together to combine their advantages.
- ✓ How to mix: weighted average and voting
- ✓ However, it is difficult to tuning, especially for weighted averages.

POSSIBLE SOLUTIONS: NEURAL NETWORK

✓ Good results, if you have enough training samples.

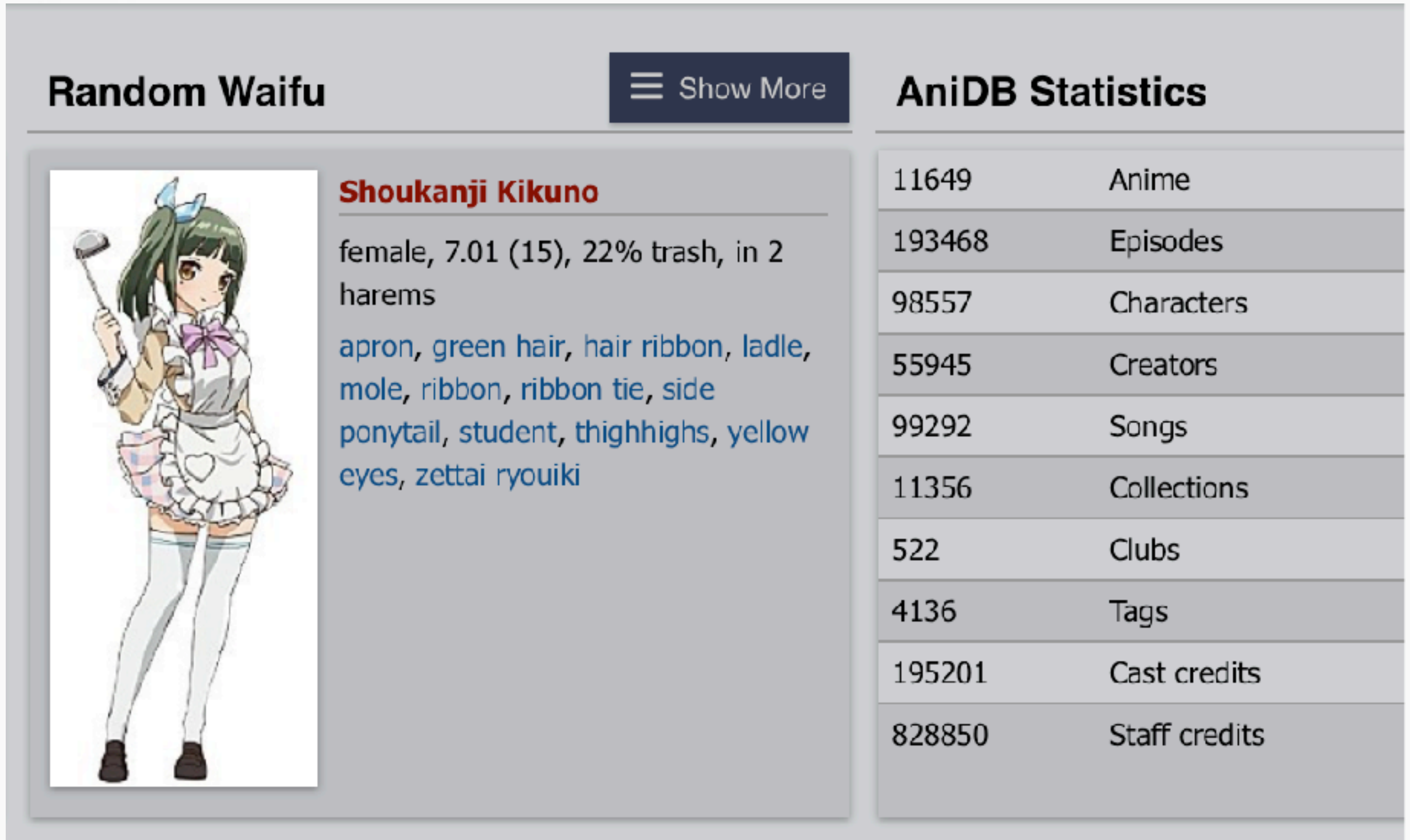


THIS IS IT?

别忘了，我们是开放性问题

SOME HINTS

- Looking for outside datasets
- https://wiki.anidb.net/w/API
- https://github.com/AniList/ApiV2-GraphQL-Docs



Shoukanji Kikuno

female, 7.01 (15), 22% trash, in 2 harems

apron, green hair, hair ribbon, ladle, mole, ribbon, ribbon tie, side ponytail, student, thighhighs, yellow eyes, zettai ryouiki

AniDB Statistics

11649	Anime
193468	Episodes
98557	Characters
55945	Creators
99292	Songs
11356	Collections
522	Clubs
4136	Tags
195201	Cast credits
828850	Staff credits

REFERENCES

- Courses for recommendation systems (https://www.coursera.org/specializations/recommender-systems)
- CCF A conference: RecSys
- Airbnb's Real-time Personalization get best paper at 2018 kdd

推荐算法的坑……

- 推荐系统太难了。难到工程师和产品都还没清楚自己要的是什么。“推荐”这个问题本身都不是well-defined的。按照道理来讲，推荐系统要做的事情其实是“推荐用户希望看到的东西”，但是“用户希望看到的东西”落实到指标上，可就让人头大了
  - 高CTR？那么擦边球的软色情以及热门文章就会被选出来
  - 高Staytime？那么视频+文章feed流就退化为视频feed流
  - 高read/U？那么短文章就会被选出来

今日头条的做法是，优化CTR同时关注其他指标的变动；也有的从CTR开始，优化到瓶颈后进行STAYTIME的优化等等…

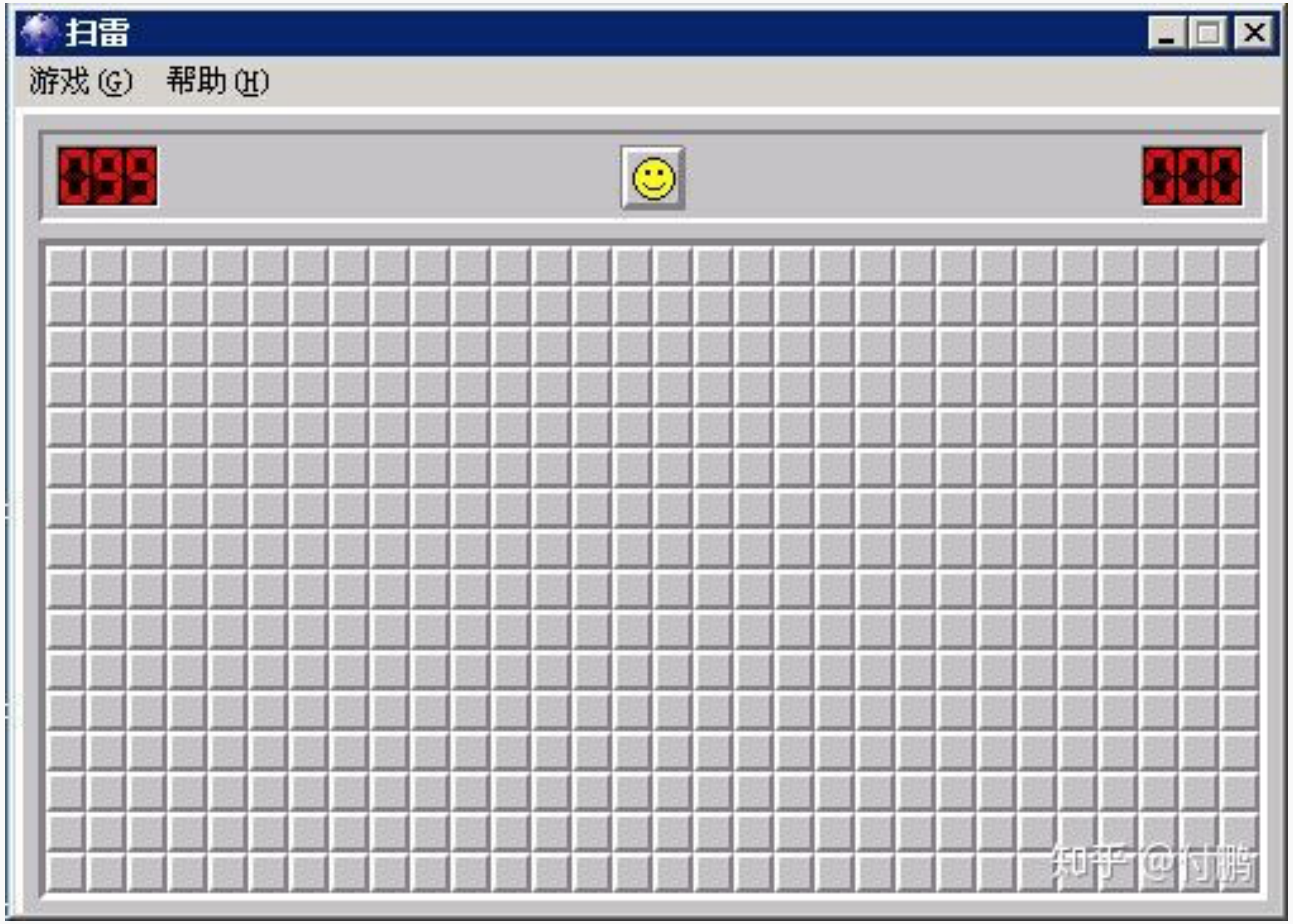
PORNHUB的做法是，优化一个-STAYTIME的指标，用户停留时长越短，则越好。

推荐算法的坑……

- 好的算法与不那么好的效果
- 有的算法确实很好，好到推荐的每个我都想点，但是算法越精准，在用户体验上未必是越好的。
- 举个例子，我喜欢汽车，电竞和科技。
- 好的推荐算法真的就推荐汽车电竞和科技，都是根据我的历史记录推荐的我确实喜欢的。
- 但也就只有汽车电竞和科技而已。换句话说，好的推荐算法毫无疑问地会局限你的视野。

推荐算法的坑……

- ✓推荐中的保证精准推荐的同时，进行兴趣探索=扫雷



推荐算法的坑……

- 用户输入：我要吃排骨。
    - 相似性召回：1. 排骨套餐。 2. 排骨咋样做好吃。 3. 糖醋排骨好吃吗?
    - 这个 case 很容易，意图和语义相似性都在一块。
- 用户输入：请问便宜的和面V3型机的哪有卖?
    - 相似性召回：1. 面包机价格。2. 便宜的面包机 3. 搅拌机价格
    - 实体名词匹配有问题，不过可解，代价是上线 NER 有点重
- 用户输入：长城那边有天下最好吃的苹果卖嘛?
    - 相似性召回：1. 长城苹果出售 2.苹果如果做好吃 3 天下牌苹果

推荐算法的坑……

1. 数据来源，抓取平行语料库，里面还有 badcase，甩不掉的。规模都是 10 亿起，当然训练可以挑一部分，不过还要离线刷全库，所以兼顾速度，所以模型不能做太复杂。
2. 工具 spark，tensorflow，mapreduce，yarn 集群。
3. 首先是 Embedding 问题，BERT 算了， 我没有 google 这个干爹。只能选择混用一下 word2vec 和 glovec 这些。查表和 token 的 index 过程，tensorflow 都能完成。什么依赖上下文的 embedding，不如查表 embedding 快呀。
4. 分词。 中文需要将 word 拆成 char，两个粒度的 embedding concat 起来。英文嘛，好不舒服。
5. 文件有点大，就用 spark 跑 tfrecord，几百 G 数据压起来不算太慢。

推荐算法的坑……

1. 模型探索：翻译模型，解决高频句对的换词和句法的改变，模型跑个几天几夜。。开源模型，跑的慢
2. 扩召回，尝试解决长尾问题（SeqGan, DSSM）
3. 相关性过滤，召回效果肯定存在不咋地的 case，过滤一下
4. 规则模型过滤，解决一些明显人就能清晰判断的 case

最终结果：惨不忍睹