REPORT FOR MM_RecSys: 28/06/2022

Data Analysis:

I am currently using movielens 20m. My preprocessing included the following:

1. Remove all the movies that I don’t have posters for.
2. Remove all the users that I don’t have demographics for.

So, the final statistics for the dataset can be found in the following table:

Number of unique users	6.040
Number of unique movies	7.861
Number of ratings	2.020.157
Range of ratings	[1-5]

All the users have rated at least 35 times.
The maximum number of ratings from a single user is 3.503.
All the users have metadata (Age, Gender, Occupation, Postcode)

Potential Problems:

1. 2.426 movies out of 7.861 have been rated less than 10 times
2. Slightly imbalanced dataset (see figure 2)

Potential Solutions:

1. Remove these movies
2. Add ratings artificially for each one so they can reach a minimum of 15 ratings each.

Notes:

I also have approximately 1.200 movie posters and metadata, but I do not have ratings for these movies.

Figure 1 shows the top 5 rows and the structure of the dataset that I am using.

	userId	movieId	rating	timestamp
1729638	4864	17	5.0	825499934
1729636	4864	11	5.0	825499934
1729635	4864	10	3.0	825499934
1729653	4864	79	1.0	825499934
1729637	4864	14	4.0	825499935

Figure 1

Figure 2 shows the label distribution: the number of ratings per label.

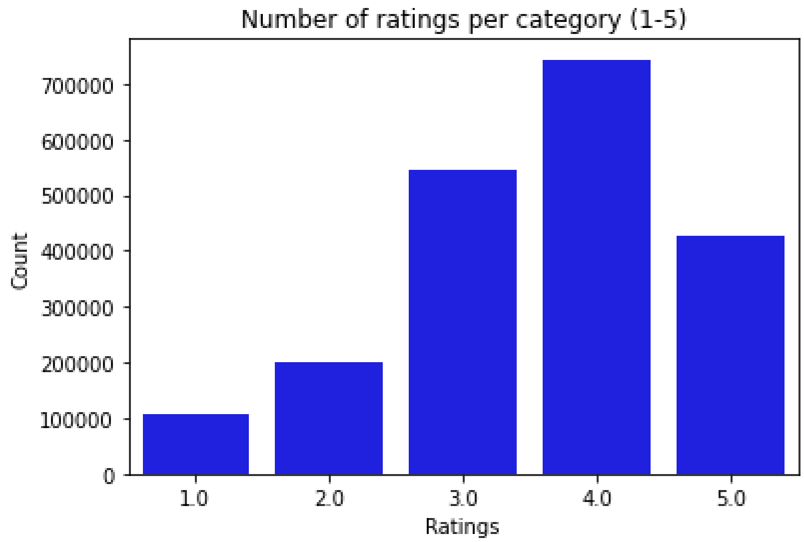


Figure 2

In the appendices, I have added some plots for reference. These include the user distribution over the different metadata, rating distribution over the user metadata and an example of a movie's poster.

Modelling:

I have developed different models based on how I want to approach the problem.

1st model: I treat this problem as a regression so the output is a number between 1 and 5. The model used as a baseline is a Neural Collaborative Filtering (Figure 3) with MSE as a loss function. I used learnable Embedding layers for movies and users.

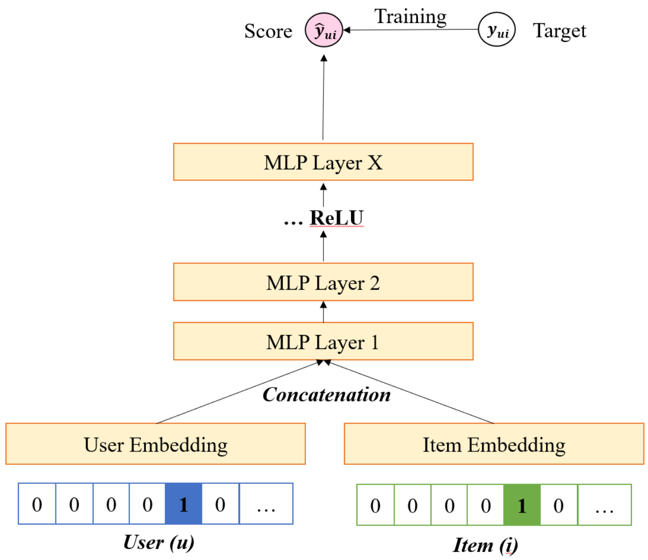


Figure 3: Model 1 architecture

2nd model: I treat this problem as a regression so the output is a number between 1 and 5. The model used as a backbone is a Neural Collaborative Filtering (Figure 4) with MSE as a loss function. I used learnable Embedding layers for the users and then a concatenation from the output of Bert and the output of CNN to represent the movies.

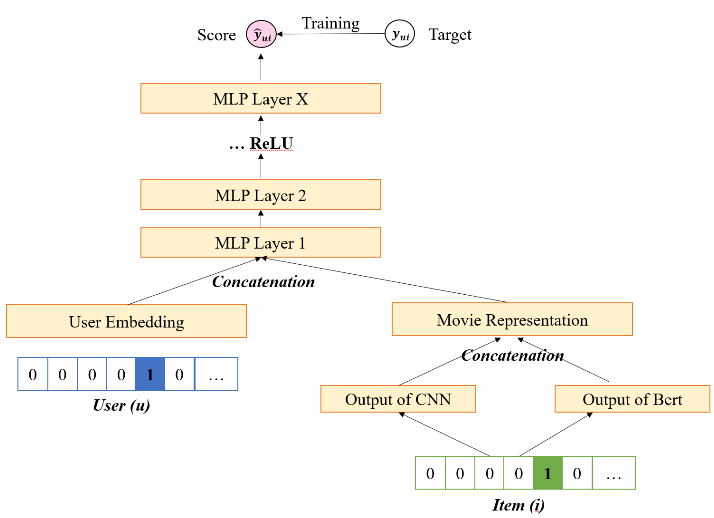


Figure 4: Model 2 architecture

Note: Given the number of ratings I have (~2 million), I could not fine-tune the BERT and CNN. So I stored the tensors (outputs of the models) in a file and I read them while I was training the model.(If this part is unclear I can explain it in detail during our call)

Models	MSE
Model 1: Baseline	0.93
Model 2: Baseline adding multimodal movie features *	1.03 (overfitting)

* I think that this would be boosted if the movies have a more balanced distribution for the movie ratings.

Models 3 and 4

I treated the same models as above as a classification problem. I used cross-entropy loss and I had a classification of 5 labels (1,2,3,4,5). So models 3 and 4 have the same architecture as models 1 and 2 (Figure 3 and 4)

Models	Accuracy
Model 3 (Baseline)	43.2
Model 4: Baseline adding multimodal movie features *	41.5

Models 5 and 6

Finally, I used Nvidia's PyTorch implementation for the NCF (see figure 5). If the differences are unclear I can explain them in detail during our call.
The Nvidia repo can be found here: link
Again I tackled the problem as a classification similarly as explained before:

Models	Accuracy
Model 5: Baseline	44.6
Model 6: Baseline adding multimodal movie features *	To be conducted

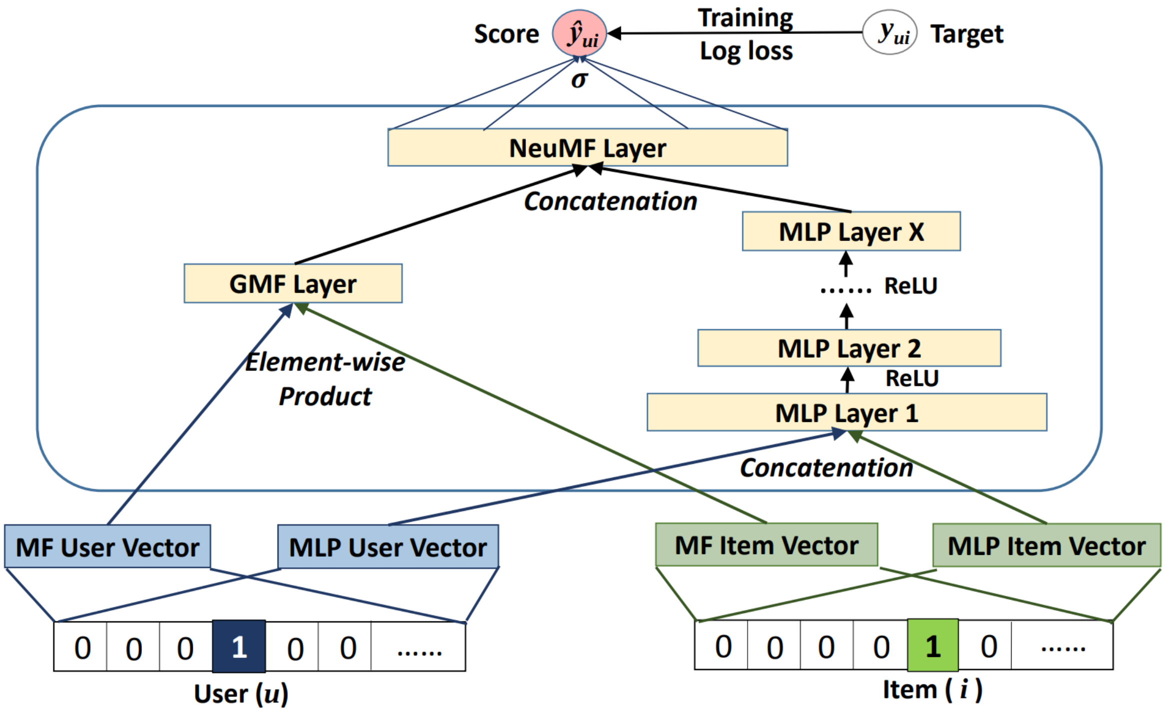


Figure 5: The architecture of a Neural Collaborative Filtering model. Taken from the Neural Collaborative Filtering paper.

Notes:
Nvidia treats this problem as a binary classification (positive or negative review). Figure 4 shows the structure of the dataset that Nvidia used:

```
train:
  - type: csv
    features:
      - user_gender
      - user_age
    files:
      - train_data_0_0.csv
      - train_data_0_1.csv
  - type: csv
    features:
      - user_id
      - item_id
      - label
    files:
      - train_data_1.csv
```

user_gender	user_age	user_id	item_id	label
0	34	1	1	1
0	29	2	5	1
1	54	4	2	0
1	34	3	2	0
0	19	6	4	1
1	23	3	5	0
0	63	6	7	1
1	62	3	8	1
1	75	7	1	1

Figure 4: Dataset structure used by Nvidia

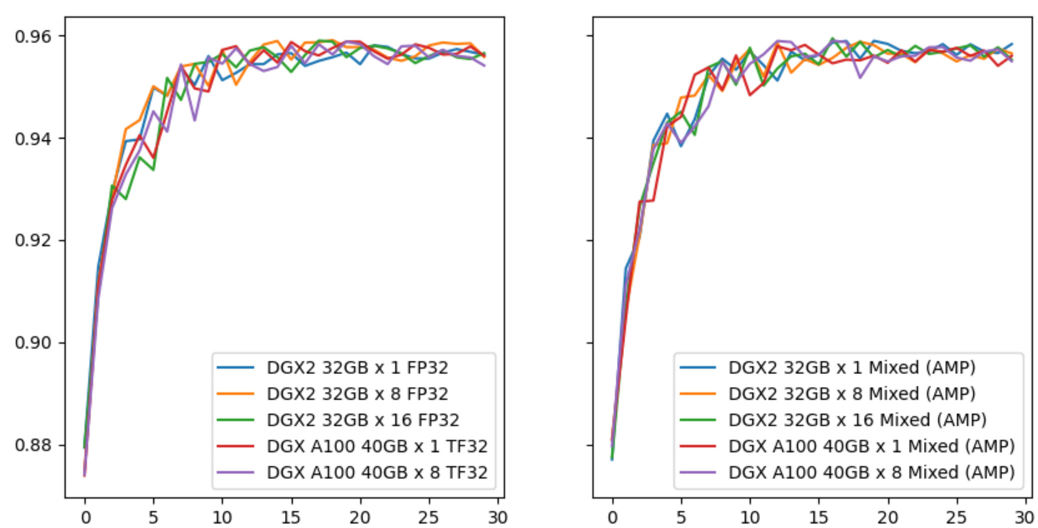


Figure 5: Nvidia's results: The plots below show the validation accuracy over the course of training. One sample curve is shown for each configuration.

Note: I can not compare my results with them since they are working with a really powerful GPU.

Questions:

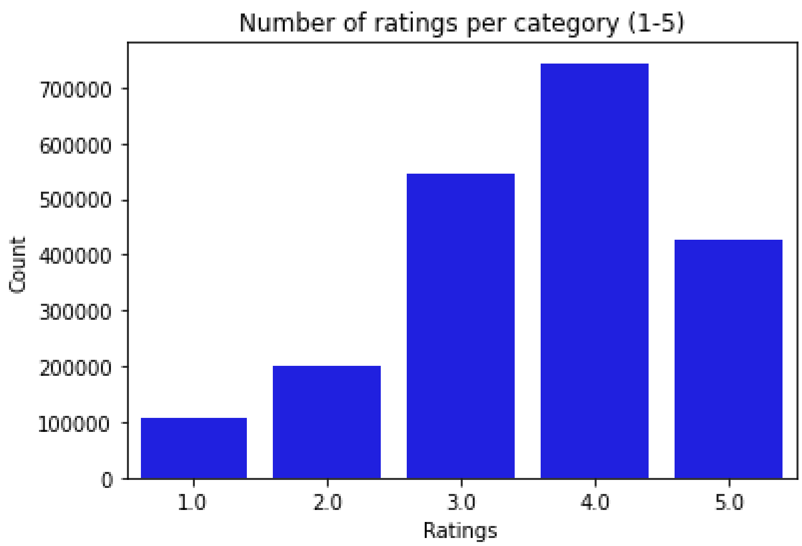
- What kind of problem is the best to try to solve:
  - Regression (predict the rating that a user will give to a movie)
  - Classification (predict the rating that a user will give to a movie from the labels [1-5])
  - Binary Classification (predict if a user will give a positive or negative rating)
- Is it going to be a big problem to not fine-tune Bert and CNN?
- Any other suggestions for recommendation models to try?
  - My ideas are
    - DLRM from Facebook
    - Attention recommenders from Microsoft
- Finally not sure how to calculate K-accuracy, K-precision, and K-recall during training. I can see how to compute it during testing
- Is the performance of the models okay (around 43 %)?

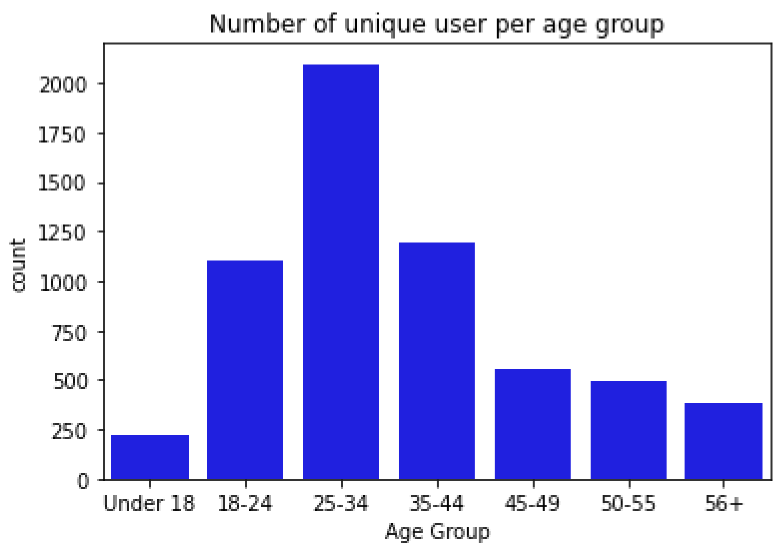
Next steps:

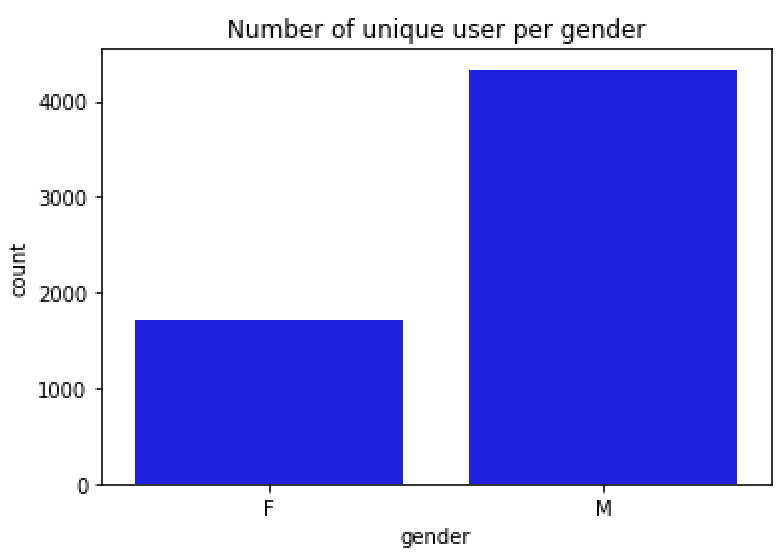
1. Conclude to what type of problem I am aiming to solve
2. Run experiments with Nvidia model and Multi modal for movies
3. Create the user representation using the user's metadata
4. Try to use these models by fine-tuning Bert and CNN

Appendicies:

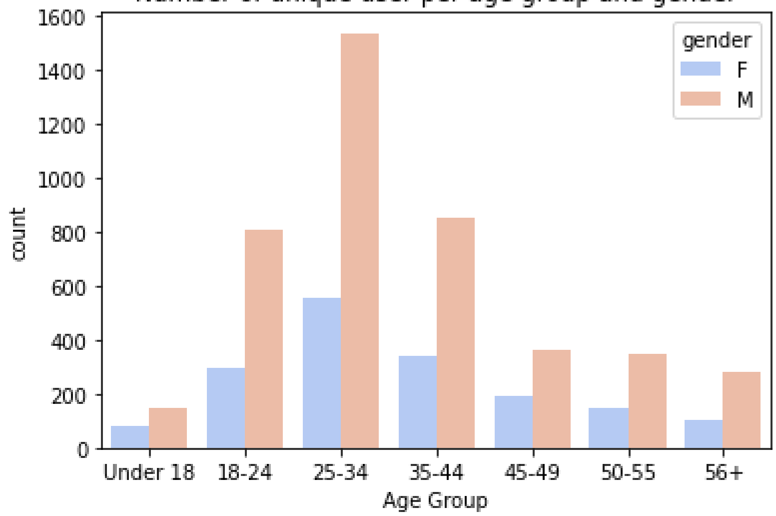
Poster Example:


Number of ratings per category (1-5)
Count
0
100000
200000
300000
400000
500000
600000
700000
1.0
2.0
3.0
4.0
5.0
Ratings

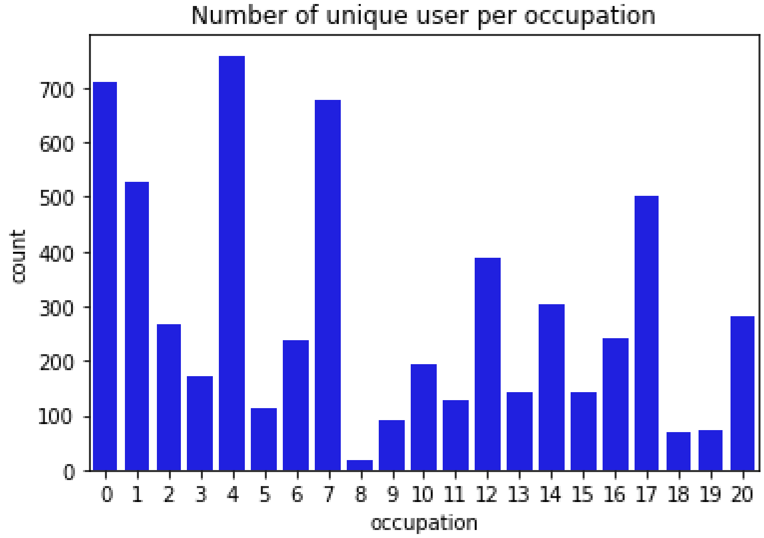

Number of unique user per age group
count
0
250
500
750
1000
1250
1500
1750
2000
Under 18
18-24
25-34
35-44
45-49
50-55
56+
Age Group


Number of unique user per gender
count
0
1000
2000
3000
4000
F
M
gender

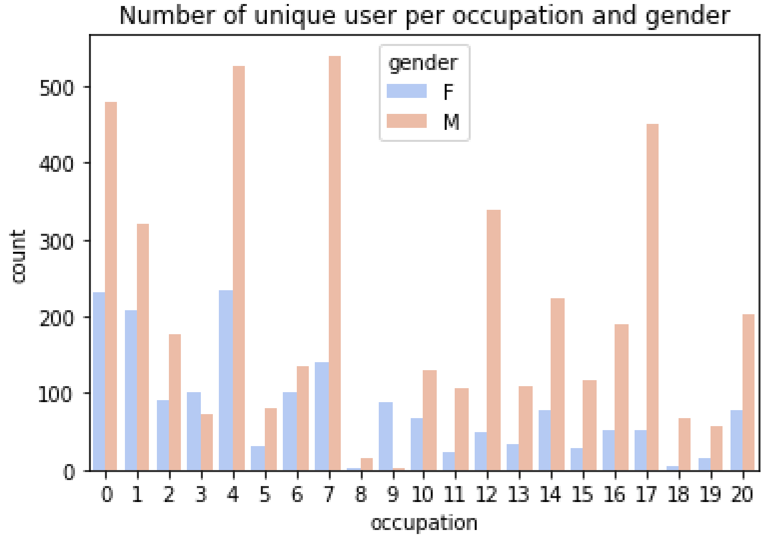
Number of unique user per age group and gender

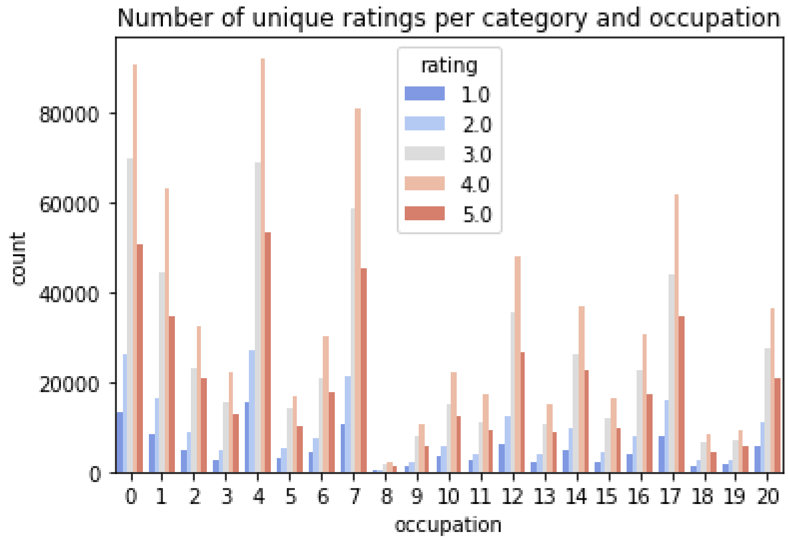


Number of unique user per occupation

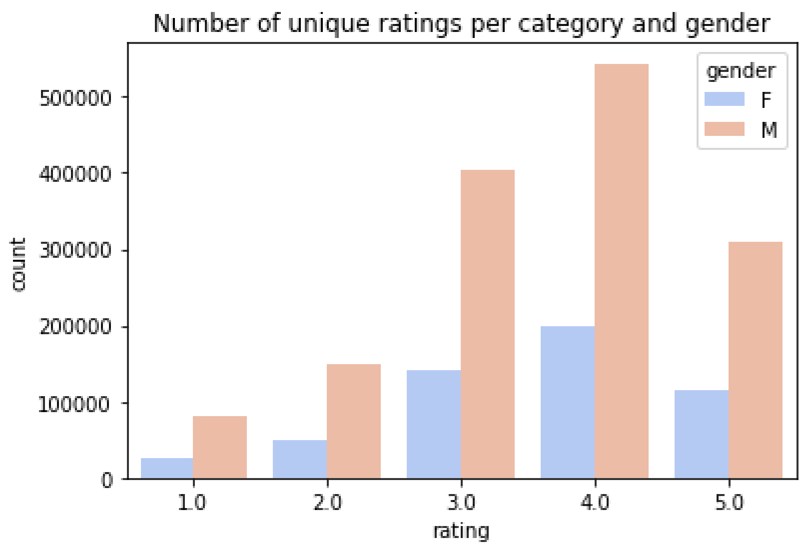


Number of unique user per occupation and gender

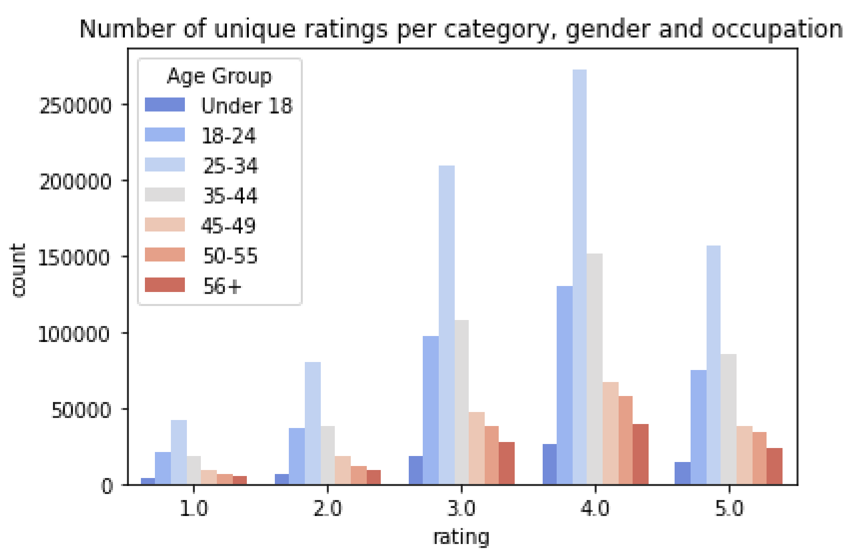




Number of unique ratings per category and occupation
rating
1.0
2.0
3.0
4.0
5.0
count
0
20000
40000
60000
80000
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20
occupation



Number of unique ratings per category and gender
gender
F
M
count
0
100000
200000
300000
400000
500000
1.0
2.0
3.0
4.0
5.0
rating



Number of unique ratings per category, gender and occupation
Age Group
Under 18
18-24
25-34
35-44
45-49
50-55
56+
count
0
50000
100000
150000
200000
250000
1.0
2.0
3.0
4.0
5.0
rating