We've processed your Assignment extension request FORM-AEX-124404

N

no-reply@qut.edu.au

To: Rodo Nguyen

Tue 07/06/2022 17:06

Hi Dac Duy Anh,

Thank you for your assignment extension request **(FORM-AEX-124404)**.

We have approved your request and the due date for your assignment **Assignment 1C**, for unit CAB420 has been extended by 48 hours from the original due date. If your unit outline does not specify that your assignment is eligible for an extension, this confirmation email is not valid and unless you submit by the original due date, the late assessment policy will apply.

You are responsible for ensuring that this assignment is eligible for extension before submitting it after the original due date. Check your unit outline for eligibility.

Be aware that a copy of this email is kept on file. You should not alter this email in any way. Email notifications that have been altered or differ in any way from the original may result in an allegation of student misconduct as set out in the Student Code of Conduct.

Need extra support? You can access free, confidential counselling with qualified professionals. We also offer planning and support if you have a disability, injury or health condition that affects your ability to study. If you are struggling to meet your university commitments, academic support is also available.

Have a question? You can contact us by email or phone. We're also available to help you on campus or via online chat. Visit the HiQ website for our contact details and opening hours.

Email us | +61 7 3138 2000

HiQ is your place to go for **student services, support** and **general enquiries**: qut.to/abouthiq

hi, how can we help you?

ASSIGNMENT 1C

CAB420 Machine Learning, Semester 1 2022

Student: Dac Duy Anh Nguyen (Rodo Nguyen)

Student ID: 10603280

Table of Contents

Problem 1: Clustering and recommendations

1 Data pre-processing

For 'NaN' values where ratings do not exist in average ratings per genre for each user, we replace them with the mean value of existed ratings of that user row. This assumes that users will give approximate ratings to genres that they have not seen. However, it is noted that this may induce the risk of giving similar movies recommendation to many users. For example, user 999 has average ratings of 5 and 1 for Action and Comedy respectively, user 001 has average ratings of 2 and 4 for Action and Documentary respectively so, they end up having the same mean value of 3 for all other genres).

Replacement with 0 was also considered but this will suggest users do not like that genre by default, which is inaccurate for most people. One other way is to customize the distance function so that it ignores 'NaN' or 0 values. However, for the benefit of simplicity, this was not implemented.

Given the fixed score range from 1 to 5, standardisation is also unnecessary.

2 Choosing a Clustering Method

Gaussian Mixture Model (GMM) is the most suitable method for this problem out of 4 clustering methods. A detailed discussion regarding their pros and cons is below.

In practice, the dataset of users, movies, and ratings can increase so for Hierarchical Agglomerative Clustering (HAC) and DBScan, whenever new data is added, the clustering result is subject to change or requires re-calculation. Especially, HAC requires a lot more computation with a large dataset while our provided one is already relatively large (with ~100 000 user ratings and ~9700 movies). This will become computational costly over time or not ideal for real-life applications in which the calculation time should be minimal and stable.

Furthermore, no clear number of clusters can be pre-determined in advance. Using the number of genres would not be reasonable since the goal is to cluster users. Other information such as distances between datapoints (for DBScan) or stopping criteria (for HAC) are unknown or required more steps to attain. Hence, all the mentioned reasons make HAC and DBScan less suited for our problem.

The disadvantage of K-Means is its hard assignment characteristic. It means that a particular object can only be assigned to one cluster and K-Means does not provide any extra metrics to analyse how this object is related to the nearby cluster. This feature available in GMM is useful in some circumstances when users are in an adventurous mood for discovering new movies, but the new ones are still required to have a minimum degree of connection to previously watched movies of the users.

Having shown the traits of other methods, GMM clearly has more strengths for this particular problem. Regarding the number of clusters K, it can be easily calculated by using the Bayesian Information Criterion (BIC) while considering the model's complexity behind each K.

3 GMM fitting

It is noticed that the optimal number of clusters K changes with the random state. So, a for loop is implemented to pick the most popular K in random states ranging from 0 to 50, inclusive. Obviously, an associated random state with this K is also recorded and used in defining the GMM model. In this solution, a random state of 42 is associated with the most common K=4 (see also Appendices 1. K frequencies).

Average ratings per genre for each user, which were collated using the provided auxiliary functions, are the only data used to fit the GMM model. An analysis of the T-SNE of the GMM model shows that different clusters are not clearly separated from each other (Figure 1). However, this should be taken as a caution as our data contains 20 genres while T-SNE is trying to represent clusters on a 2D plane.

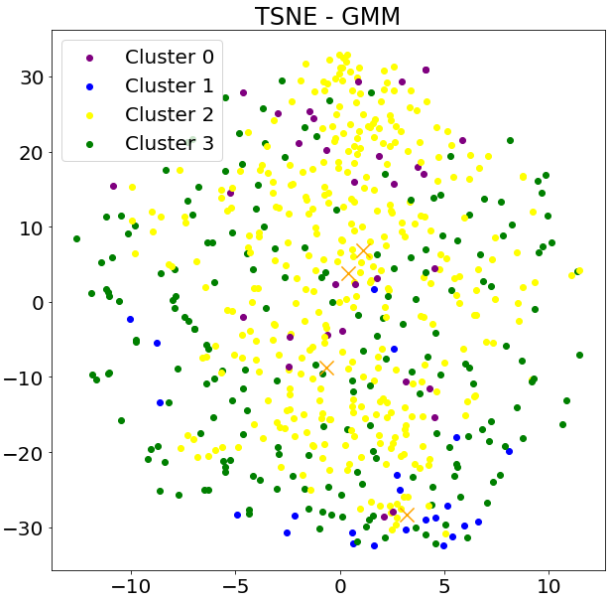


FIGURE 1 T-SNE OF THE GMM MODEL

The figures from average genre ratings per cluster (Table 2) also support the T-SNE plot. Clusters 2 and 3 have little difference, the former has the ratings centre around 3.7-3.8 while the latter's is around 3.5-3.6 with some genres having ratings within the former's range and vice versa. Ratings from 3.5 to 3.8 (Cluster 2 and 3) are common in the average ratings per genre for each user, which explains the broad coverage of Cluster 2 and 3. In contrast, cluster 0 and 1 respectively captures extremely high and low ratings which are very rare in the dataset and thus, cover only a small part at the top and bottom of the T-SNE plot. To conclude, our GMM model clusters users largely based on user average ratings and not just on partial ratings across a few genres. Samples proving this can be found in Appendices 2.

Cluster population

Cluster 0	31
Cluster 1	22
Cluster 2	356
Cluster 3	301

TABLE 1 CLUSTER POPULATION

Cluster 0

Action	3.772901
Thriller	3.729827
Drama	3.917985
IMAX	3.820238
Film-Noir	4.097355
Mystery	3.988388
Adventure	3.850692
Romance	3.952662
Comedy	3.909405
Children	3.940564
Horror	3.712230
Documentary	3.962322
Sci-Fi	3.741105
(no genres listed)	3.995974
Animation	4.008374
Fantasy	3.915711
Western	3.976175
Crime	3.901705
Musical	3.818365
War	3.789045

Cluster 1

Action	2.493381
Thriller	2.518421
Drama	2.963798
IMAX	2.523202
Film-Noir	2.956315
Mystery	2.892565
Adventure	2.556202
Romance	2.777055
Comedy	2.939068
Children	2.610568
Horror	2.529228
Documentary	2.670285
Sci-Fi	2.394677
(no genres listed)	2.615854
Animation	2.516522
Fantasy	2.672005
Western	3.008027
Crime	2.620940
Musical	2.677069
War	3.214158

Cluster 2

Cluster 3

Action	3.672905	Action	3.425623
Thriller	3.742267	Thriller	3.523692
Drama	3.845569	Drama	3.661561
IMAX	3.866304	IMAX	3.651739
Film-Noir	3.847494	Film-Noir	3.579080
Mystery	3.862573	Mystery	3.593537
Adventure	3.711888	Adventure	3.529562
Romance	3.729036	Romance	3.522824
Comedy	3.644601	Comedy	3.448720
Children	3.679807	Children	3.262530
Horror	3.654453	Horror	3.293507
Documentary	3.814653	Documentary	3.617616
Sci-Fi	3.673297	Sci-Fi	3.375843
(no genres listed)	3.764764	(no genres listed)	3.515299
Animation	3.788606	Animation	3.466486
Fantasy	3.681292	Fantasy	3.428075
Western	3.749409	Western	3.526953
Crime	3.839402	Crime	3.704675
Musical	3.754523	Musical	3.486343
War	3.972436	War	3.692309

TABLE 2 AVERAGE GENRE RATINGS PER CLUSTER

3 Movie Recommendations

To recommend movies to a user, we perform the following steps:

- Identify the cluster which the user falls into
- Collect the movies that the cluster has watched, excluding the user's watched movies
- Finally, recommend movies that are popular and received 'above 4' ratings from other users in the same cluster.
- The order of recommendation is prioritising movies with the highest number of 'above 4' rating counts to the lowest. This makes sure our recommendation is highly rated by many other users.

We have the top5 movie recommendations for 3 targeted users as the following table:

UserID 4 – Cluster 2 (predicted probability: 0.974)		
movieId	*Name*	*Genres*
318	Shawshank Redemption, The (1994)	Crime|Drama
356	Forrest Gump (1994)	Comedy|Drama|Romance|War
527	Schindler's List (1993)	Drama|War
858	Godfather, The (1972)	Crime|Drama
50	Usual Suspects, The (1995)	Crime|Mystery|Thriller
UserID 42 – Cluster 2 (predicted probability: 0.87)		
movieId	*Name*	*Genres*
4993	Lord of the Rings: The Fellowship of the Ring, The (2001)	Adventure|Fantasy
1198	Raiders of the Lost Ark (Indiana Jones and the Raiders of the Lost Ark) (1981)	Action|Adventure
7153	Lord of the Rings: The Return of the King, The (2003)	Action|Adventure|Drama|Fantasy
2858	American Beauty (1999)	Drama|Romance
4226	Memento (2000)	Mystery|Thriller
UserID 314 – Cluster 2 (predicted probability: 0.999)		

movieId	*Name*	*Genres*
2571	Matrix, The (1999)	Action|Sci-Fi|Thriller
2959	Fight Club (1999)	Action|Crime|Drama|Thriller
858	Godfather, The (1972)	Crime|Drama
1196	Star Wars: Episode V - The Empire Strikes Back (1980)	Action|Adventure|Sci-Fi
4993	Lord of the Rings: The Fellowship of the Ring, The (2001)	Adventure|Fantasy

TABLE 3 MOVIE RECOMMENDATIONS FOR TARGETED USERS AND THEIR CLUSTER ID

Action	1828.0
Thriller	1894.0
Drama	4361.0
IMAX	158.0
Film-Noir	87.0
Mystery	573.0
Adventure	1263.0
Romance	1596.0
Comedy	3756.0
Children	664.0
Horror	978.0
Documentary	440.0
Sci-Fi	980.0
(no genres listed)	34.0
Animation	611.0
Fantasy	779.0
Western	167.0
Crime	1199.0
Musical	334.0
War	382.0

TABLE 4 GENRE FREQUENCIES IN THE PROVIDED MOVIE LIST

Having had the full recommendation for these users, it comes to our attention that the users' genre ratings (see Table 5 Targeted user profiles) greater than or equal to 4 are uncommon in the movies list (see Table 4 Genre frequencies). Some identified instances are Film-noir, Horror, Documentary, Animation, Musical in userID 4's average ratings and IMAX, Documentary, Animation, Western in userID 42's. Therefore, the movies in these genres can hardly climb to the top of the recommendation list. Another speculation is because cluster 2 is too large, or in other words, the problem is under-clustered, objects having little in common are forced to be in the same group as a result.

With this considered, it is easy to understand why the movies recommended are very popular and enjoyed in the cluster, but their genres do not intersect the users' favourite genres (from 4 to 5).

Therefore, if we perceive the new high rating range is from 3.5 to 5 and disregard uncommon genres, the recommended movies (as well as their genres) become more reasonable to userID 4 and 42 profiles. For picky users such as userID 314, their average ratings are much lower. So, in this particular case, the recommendation also matches most of their highest genre ratings which are higher than 2.9 except for Action, Thriller and Adventure.

In general, the recommendation system is functioning properly as it does suggest related movies with one or more genres in the user's top list and does not suggest anything that is obviously unrelated. For example, Sci-fi movie having the lowest rating from userID 4 is not recommended for this user; the same thing with Children genre for userID 42 and Western genre for UserID 314. However, the current GMM model is suggesting movies based on the whole cluster's behaviour without personalizing each user profile. This however can be improved further by experimenting and choosing the optimal number of clusters to group smaller users with more similar interests and adding code logic to prioritize users' favourite genres.

Average ratings of user: 4	
Action	3.320000
Thriller	3.552632
Drama	3.483333
IMAX	3.000000
Film-Noir	4.000000
Mystery	3.478261
Adventure	3.655172
Romance	3.379310
Comedy	3.509615
Children	3.800000
Horror	4.250000
Documentary	4.000000
Sci-Fi	2.833333
(no genres listed)	3.638532
Animation	4.000000
Fantasy	3.684211
Western	3.800000
Crime	3.814815
Musical	4.000000
War	3.571429

Average ratings of user: 42	
Action	3.400000
Thriller	3.577586
Drama	3.818713
IMAX	4.500000
Film-Noir	3.500000
Mystery	3.962963
Adventure	3.513889
Romance	3.640449
Comedy	3.412322
Children	2.928571
Horror	3.000000
Documentary	4.333333
Sci-Fi	3.250000
(no genres listed)	3.671503
Animation	4.000000
Fantasy	3.590909
Western	3.916667
Crime	3.746479
Musical	3.857143
War	3.809524

Average ratings of user: 314	
Action	2.820000
Thriller	2.710526
Drama	3.400000
IMAX	3.666667
Film-Noir	3.095759
Mystery	3.000000
Adventure	2.842105
Romance	3.357143
Comedy	3.062500
Children	3.153846
Horror	2.833333
Documentary	3.095759
Sci-Fi	2.961538
(no genres listed)	3.095759
Animation	3.125000
Fantasy	3.214286
Western	2.600000
Crime	3.000000
Musical	3.166667
War	3.714286

TABLE 5 TARGETED USER PROFILES: MEAN RATINGS

Problem 2: Multi-Task Learning

1 Data Pre-processing

The given dataset is colour and of the size 100x60. As two of the output require us to classify colour so images are not converted to grayscale to keep the rich RGB information for the learning process. Semantic segmentation is not utilised in this solution.

The given train, test data is 520 and 196 samples respectively. This is a small number considering the complexity of the problem (i.e., the number of outputs and classes to classify) and may not allow the model to generalize well (further discussion in section 2. Model Design & Training).

To help mitigate this issue, data augmentation is applied with great caution as the objects can lose their original meaning. For example, reducing brightness or increasing contrast may make grey clothes darker and become black or increasing width may make women's body looks like men's. So, in this problem, augmentation affecting the object's width-height ratio and colour is not applied, and the final settings are as below:

- Rotation range: 12 degrees
- Width shift range: 7% (note: width and height shift are about object position in the image, not about object width-height ratio)
- Height shift range: 7%
- Shear range: 10%
- Zoom range: 10%
- Horizontal flip: True
- Fill mode: Constant, cval=0, to avoid creating more background noise

The augmented images are examined and ensured to keep their original meaning compared to their labels. As suggested from the current augmented samples, greater range augmentation will be prone to information loss.

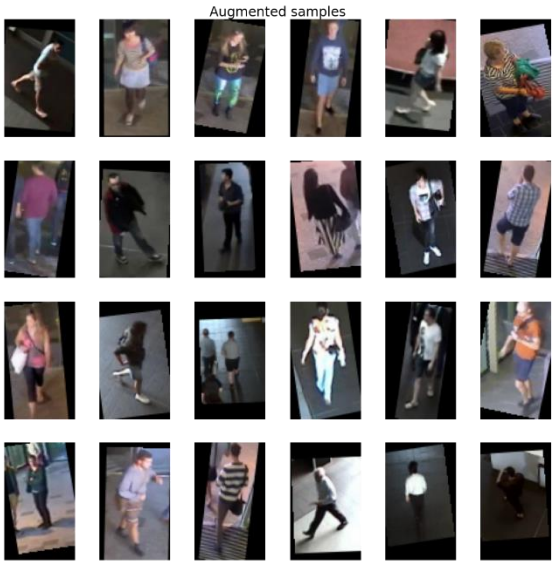


FIGURE 2 AUGMENTED SAMPLES

Despite we want to create more augmented data, other augmentation methods are not applied as they may affect the meaning of the image. For example, contrast can decrease pixel value and make clothes blacker, in addition, the low light condition of the images will make the situation even worse. Width/height zoom can affect the width-height ratio and may inhibit gender identification learning.

A closer examination of the unknown labelled samples shows that despite the majority can actually be assigned to an available class, a small portion of them can be classified into neither available class. Therefore, this eliminates the avenue in which we use '-1' as a separate class since they are not totally true. In the end, they remain an 'unknown' class.

Data imbalance is at an extreme level in this case (see Appendices 3. Data distribution regarding the numbers of samples per class in train/validation/test data). Some classes in either train, validation or test data don't

have any sample at all, which is very likely to negatively affect the prediction result. The impact is expected to be more severe when the available number of datapoints in this problem is too small.

To create validation data, 120 samples are randomly selected from train data, which still leaves the majority of 400 samples (approximately 56% of the total images) for training.

2 Model Design

Since the given samples are limited, it is decided to preserve the original size of the image and use it for our model input. This is an attempt to maximize the input information.

We applied masked sparse categorical cross-entropy loss to each output to exclude the 'unknown' (or -1) label from train data from loss calculation while other information where the label exists can still be used to contribute to the model.

The model is designed following VGG style and consists of 4 Convolutional blocks. Deeper networks than the proposed one may inhibit the information being transferred to the last layers. Additionally, it is even harder to train a deep neural network well on limited samples. Details regarding layer by layer can be viewed in Appendices 4.

Regarding the 6 outputs, each of the outputs is connected to a separate Dense layer of 256 neurons. A different approach is to merge the 'types' and the 'colors' output of torso/leg to the same Dense layer. But we believed they require adjusting their separate layer of neuron weights for their specific outcome. And indeed, the experiment's average accuracy from 6 outputs from the former design is higher than the latter so the former is used. Note that the difference between the 2 models' average accuracy is relatively small (~3%).

There are 28 classes from 6 outputs in total. We conjectured a rather unideal scenario where 1 class appears at least once in 1 image. So, a batch size greater than or equal to 28 is thought to be a reasonable size. However, with the imbalance dataset, the situation may be worse if some classes may not show up at all, so numbers higher than 28 but also not too large (to provide the model with more updating steps) are preferred. Out of the batch sizes experimented with: 32, 64, and 100, 64 is the winner with the highest average accuracy although the difference is also fairly small (~1 to 3%). Further discussion and evaluation along with other metrics will be discussed in detailed in the next section.

2 callbacks are utilised in the fit function: ModelCheckpoint to save only the model with the lowest validation loss and EarlyStopping to stop the training when no loss decrease is detected in 25 epochs. The best-saved model's weights will be loaded to the model instead of using the overfitted one at the end of the training.

3 Evaluation

First of all, our model performance is moderate from the accuracy in each output (see Table 6 Accuracy and F1-Score of each output). At least, our model did learn some things and predict better than random guesses. However, F1-scores share a different aspect of the model.

F1-score is the metric that takes each class's performance into consideration. As we can see, F1-scores are all smaller than the accuracy which indicates that our model is heavily biased towards one or more classes. We can see this sign more clearly in Figure 4 Confusion Matrixes of the predictions as well. With more classes

being falsely predicted, F1-score will be further decreased, which are the cases of 'Torso colour' and 'Leg clothing colour'.

Output	*Number of classes*	*Accuracy*	*F1-score (macro-average)*
Gender	2	.54	.35
Torso type	2	.63	.29
Torso colour	11	.33	.05
Leg clothing type	2	.56	.38
Leg clothing colour	11	.45	.06
Has Luggage	2	.65	.41

TABLE 6 ACCURACY AND F1-SCORE OF EACH OUTPUT

2 identified reasons causing both the average performance and the bias are: limited samples and an extremely imbalanced dataset. First of all, the number of samples should be thousands or more in order to provide the model with enough iterations as well as a diverse dataset to generalize and learn features in the images. Secondly, data imbalance combined with the lack of samples makes the model tend to take the lazy shortcut by giving the prediction as the major classes (in train data) more as this guarantees the high accuracy. For example, the number of short torsos in train data is 250 out of 400 samples so the model consequently predicted every test case as short torsos. The rest of the prediction results show a similar pattern and can be viewed in Figure 4 Confusion matrixes of prediction results.

Moreover, some classes are not even present in the validation or test data. Without samples in the validation data, the model will not see the feedback from loss value and thus never optimize its accuracy for these empty classes. And without samples in the test data, the test result will not reflect the correct overall performance for all classes. From the arguments above, we can also predict that the model performance may be worse if the test data distribution is altered.

It's also worth noting that some provided images are of low quality (false label, bad light condition) as shown in Figure 3. As a result, false labels may even harm the model instead of improving it by giving wrong loss calculation.

a) Low light b) Incorrect torso colour label

c) Incorrect torso colour label d) Incorrect leg clothing colour

FIGURE 3 LOW-QUALITY IMAGES SAMPLED FROM THE DATASET

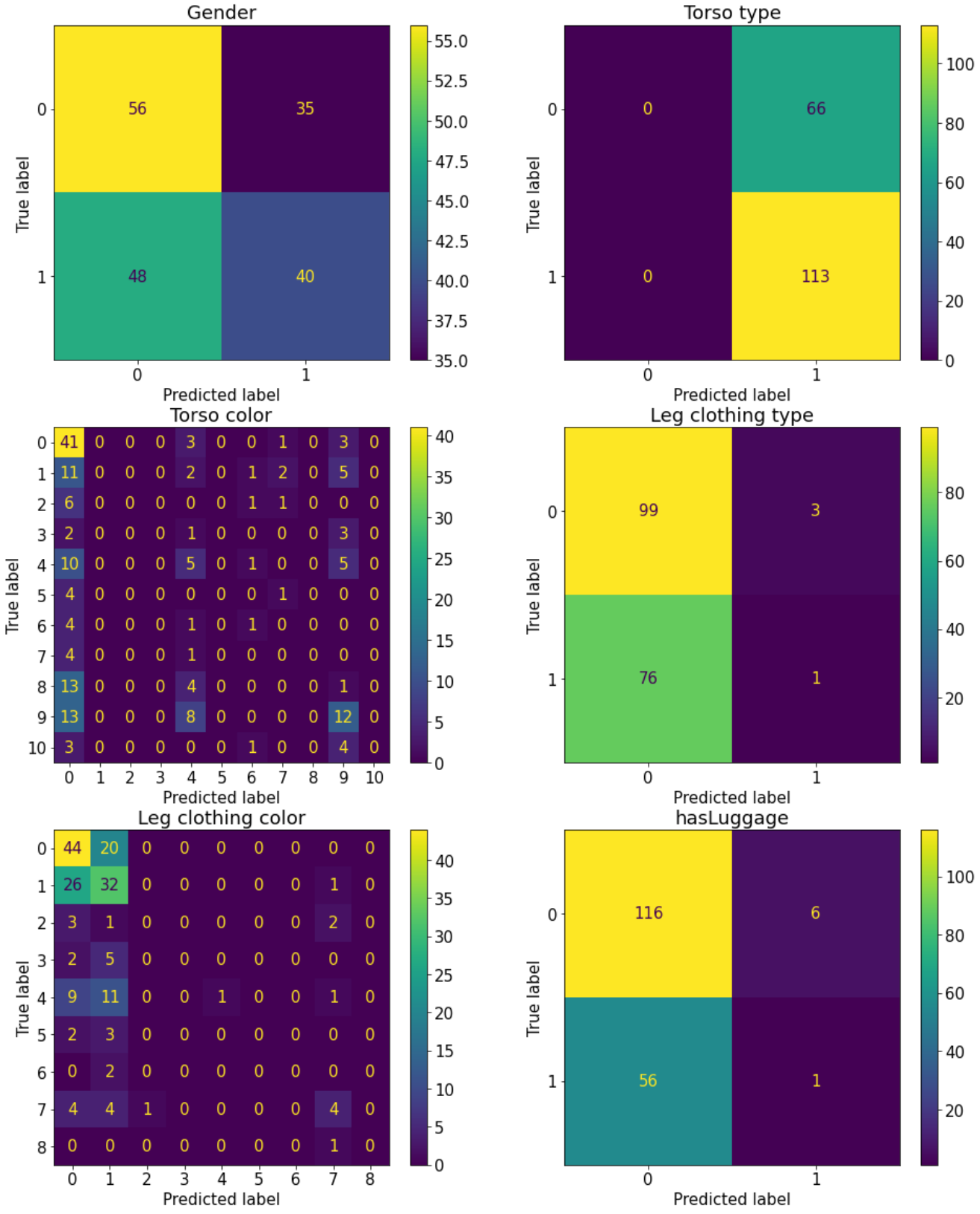


FIGURE 4 CONFUSION MATRIXES OF PREDICTION RESULTS

Since there are not many images to learn from, the model overfitted quickly after approximately 20 epochs and was stopped at epoch 43. Overall, we only see a minor decrease in most outputs' loss from the beginning to the lowest point. Interestingly, for the model, luggage may have been a bit harder to detect as they are small items in the images. So, it took the model about 3 epochs to bring down the significantly high loss of 'luggage' to the local minimum while others keeps a flat line in their validation loss from the beginning (see Figure 5 Training history)

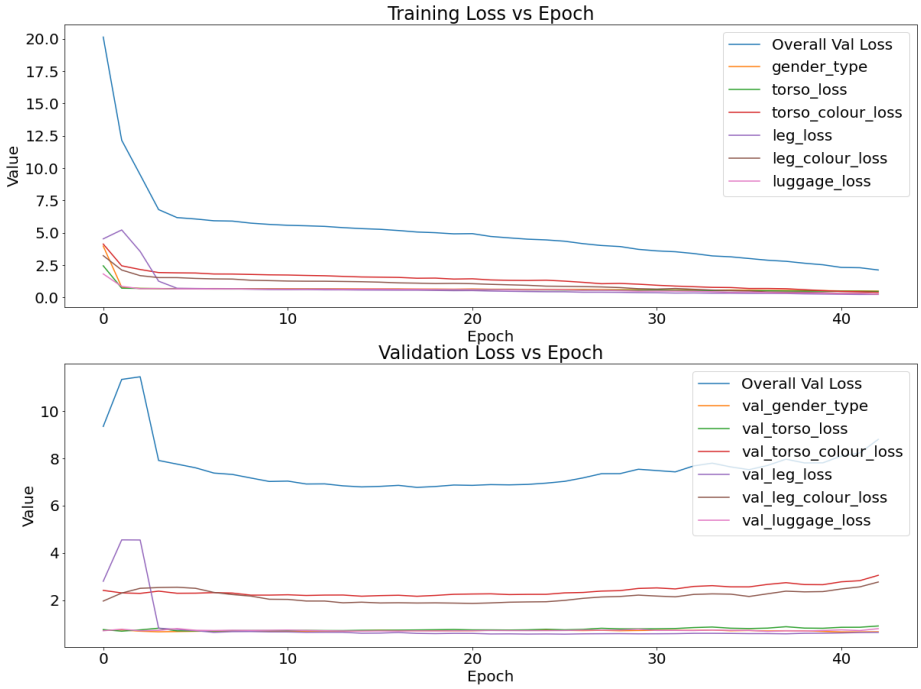


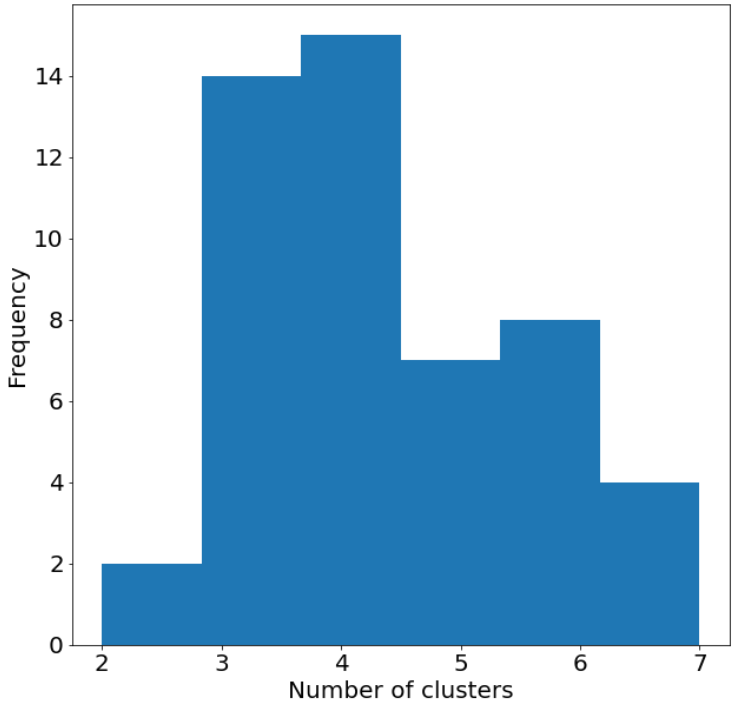
FIGURE 5 TRAINING HISTORY

For future improvement, we recommend labelling the missing data where possible, collecting more samples as well as balancing out the dataset in order to avoid overfitting so that the model can actually learn the target features instead of prioritizing the major class. With more samples becoming available, it opens opportunities for us to design deeper models and may as well improve the prediction accuracy.

Appendices

Problem 1:

1. K frequencies

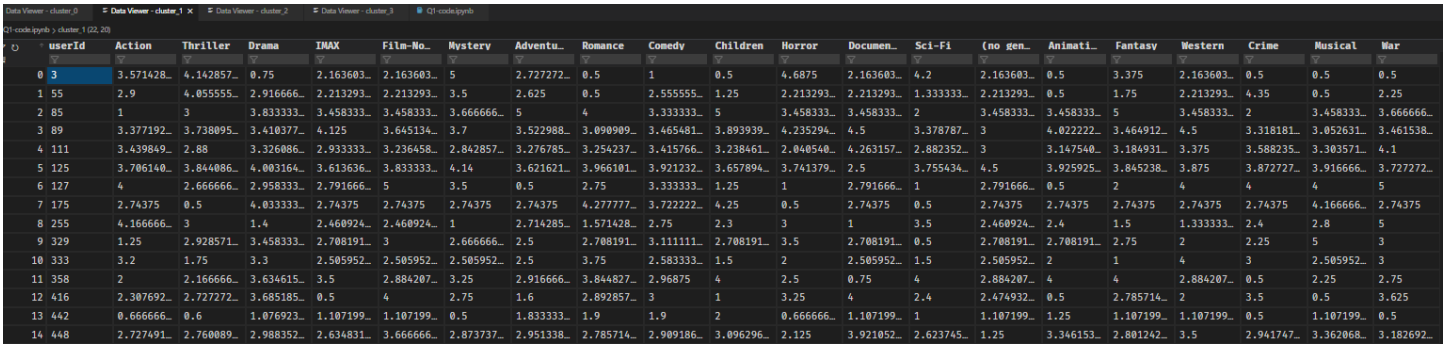


2. Samples of average ratings per genre for each cluster's users

Cluster 0: High ratings

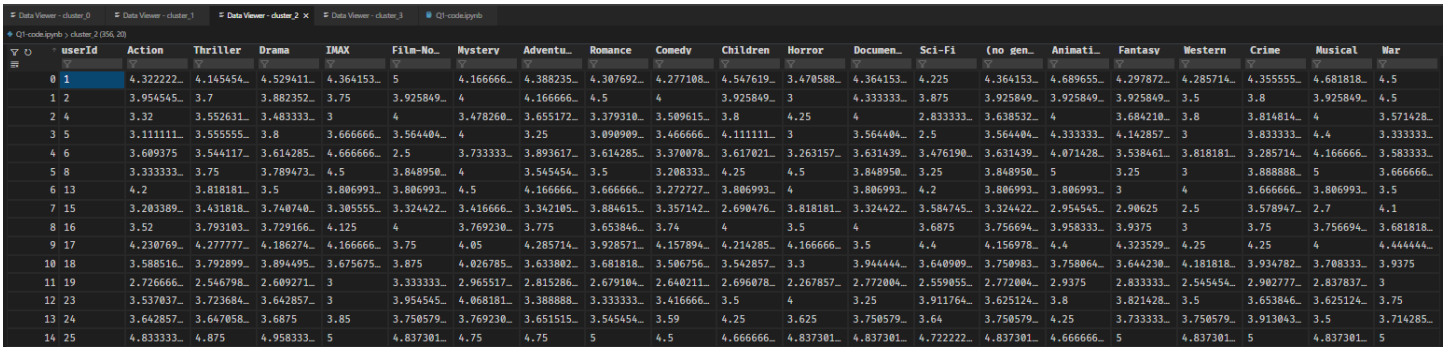
	userId	Action	Thriller	Drama	IMAX	Film-No…	Mystery	Adventu…	Romance	Comedy	Children	Horror	Documen…	Sci-Fi	(no gen…	Animati…	Fantasy	Western	Crime	Musical	War
0	21	3.455752…	3.545081…	2.952127…	3.596491…	3.5	3.361111…	3.663841…	3.238461…	3.201492…	3.204918…	1.766666…	3.308727…	3.448113…	4	3.241379…	3.087301…	4	3.178082…	2.8	3.625
1	31	4.333333…	4.454545…	3.666666…	4.132634…	4.132634…	5	4	3.75	3.863636…	4.142857…	3.5	5	3.777777…	4.132634…	4	3.7	4.132634…	3.333333…	4.6	5
2	50	2.450617…	2.824561…	3.039682…	2.068181…	3.5	3.225	2.475	2.818181…	2.590909…	2.261904…	2.625	3.142857…	2.55	2.75	2.622641…	2.829545…	3.166666…	2.933333…	2.566666…	3.454545…
3	62	3.988023…	4.152173…	4.28125	4.058823…	4.5	4.220930…	3.933333…	3.782608…	3.873626…	3.645833…	4.023809…	4.040229…	3.973913…	4.5	3.392857…	3.894366…	4.5	4.468253…	3.416666…	4.157894…
4	70	3.75	3	4.411764…	4	4.193946…	4	4.375	4.464285…	4.138888…	4.625	4.5	4.5	4.25	4.193946…	4	4.625	4	4.357142…	4.193946…	4.3
5	105	3.894578…	4.071428…	4.160220…	3.924242…	4.166666…	4.25	3.932142…	4.012096…	4.032653…	4.112903…	4.066666…	4.454545…	3.962765…	4.833333…	4.268518…	3.976744…	4	4.229323…	3.892857…	4.113636…
6	154	4.181818…	2.5	4	4.428571…	4.351975…	5	4.571428…	5	4.4375	4.9375	4.351975…	4.351975…	4.733333…	4.351975…	4.625	4.8125	4.351975…	3.2	4.351975…	4.5
7	184	3.540322…	3.758620…	3.976190…	3.916666…	3.791331…	4.115384…	3.513157…	4.166666…	3.574074…	3.214285…	3.1	4.4375	3.817073…	4.5	3.431034…	3.35	3.791331…	4.041666…	3.791331…	4
8	206	3.285714…	3.444444…	4.583333…	4.129535…	4.129535…	3.666666…	3.625	4.375	4.25	4.25	4.129535…	4.129535…	3.333333…	4.129535…	4.666666…	4.333333…	5	5	4	4.129535…
9	210	4.139705…	3.857142…	3.890625	4.259259…	4.043323…	4.333333…	4.230769…	4.4375	3.896551…	4.017857…	3.333333…	4.043323…	4.201219…	4	4.108695…	3.987179…	4.043323…	3.25	4.043323…	4.75
10	212	3.504629…	3.436363…	3.612903…	3.446428…	4	3.526315…	3.574712…	3.625	3.614942…	3.683333…	3.428571…	3.586634…	3.441176…	3.5	3.736111…	3.727272…	3.5	3.638297…	3.75	3.4
11	236	5	3.857142…	4.4	4.262441…	4.262441…	4.333333…	5	4.333333…	4.058823…	4.262441…	3.5	4.262441…	3.666666…	4.262441…	4.262441…	4.333333…	4.262441…	4.666666…	4	4.262441…
12	248	3.772727…	3.833333…	3.375	3.25	3.725590…	4.25	3.728571…	4	3.958333…	3.714285…	3.725590…	3.5	3.730769…	4	3.6875	3.75	3.725590…	3.333333…	3.725590…	3.725590…
13	252	3.863636…	4	4.321428…	4	4.052648…	4	4.194444…	4.3	4.3	4.433333…	4.052648…	4.052648…	3.857142…	3	4.394736…	4.25	4.052648…	3.5	4.375	4.052648…
14	258	3.7	4.142857…	4.538461…	2.5	4.133199…	4.5	3.65	5	5	4.833333…	4.5	4.133199…	3.5	4.133199…	4.8	4.083333…	4.133199…	5	4.133199…	2.25
15	296	3.409090…	4.785714…	4.666666…	3	4.320235…	5	2.944444…	5	4.833333…	5	4.320235…	4.5	2.571428…	4.5	4.666666…	3.9	5	4.916666…	4.320235…	4.75
16	318	3.547727…	3.622395…	3.830049…	3.583333…	3.866666…	3.6875	3.566666…	3.695238…	3.811355…	3.676470…	3.370967…	4.017045…	3.544871…	4.25	3.741758…	3.581081…	4.05	3.761029…	3.589285…	3.852272…
17	380	3.705785…	3.575163…	3.741042…	3.704225…	4.666666…	3.553921…	3.773657…	3.517045…	3.643243…	3.613496…	3.441919…	3.75	3.788401…	4	3.940397…	3.641552…	4.113636…	3.720873…	3.836734…	4.030303…
18	414	3.281437…	3.2904	3.561497…	3.719298…	3.857142…	3.573333…	3.423413…	3.376227…	3.271084…	3.415254…	3.083333…	3.677419…	3.402476…	4.25	3.786290…	3.411504…	3.576923…	3.475961…	3.454545…	3.830645…

Cluster 1: Low ratings



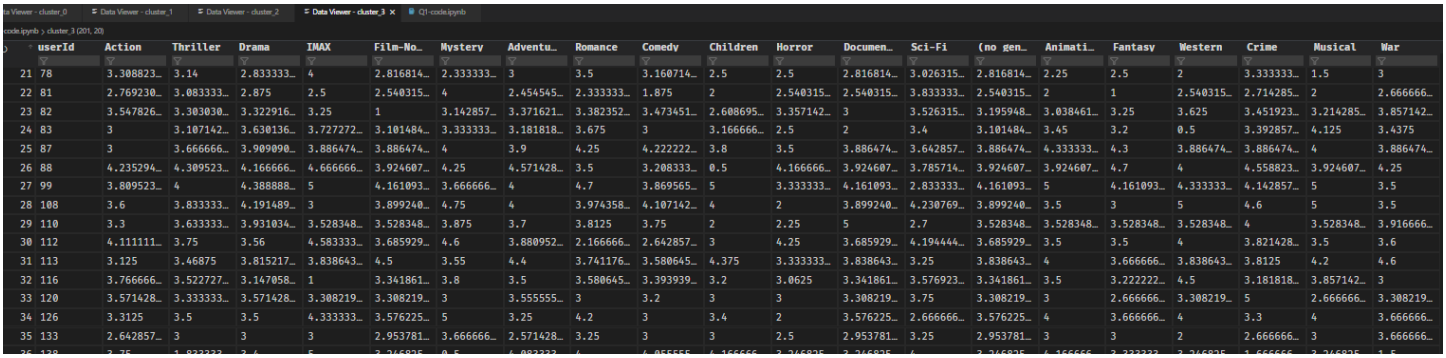
	userId	Action	Thriller	Drama	IMAX	Film-No…	Mystery	Adventu…	Romance	Comedy	Children	Horror	Documen…	Sci-Fi	(no gen…	Animati…	Fantasy	Western	Crime	Musical	War
0	3	3.571428…	4.142857…	0.75	2.163603…	2.163603…	5	2.727272…	0.5	1	0.5	4.6875	2.163603…	4.2	2.163603…	0.5	3.375	2.163603…	0.5	0.5	0.5
1	55	2.9	4.055555…	2.916666…	2.213293…	2.213293…	3.5	2.625	0.5	2.555555…	1.25	2.213293…	2.213293…	1.333333…	2.213293…	0.5	1.75	2.213293…	4.35	0.5	2.25
2	85	1	3	3.833333…	3.458333…	3.458333…	3.666666…	5	4	3.333333…	5	3.458333…	3.458333…	2	3.458333…	3.458333…	5	3.458333…	2	3.458333…	3.666666…
3	89	3.377192…	3.738095…	3.410377…	4.125	3.645134…	3.7	3.522988…	3.090909…	3.465481…	3.893939…	4.235294…	4.5	3.378787…	3	4.022222…	3.464912…	4.5	3.318181…	3.052631…	3.461538…
4	111	3.439849…	2.88	3.326086…	2.933333…	3.236458…	2.842857…	3.276785…	3.254237…	3.415766…	3.238461…	2.040540…	4.263157…	2.882352…	3	3.147540…	3.184931…	3.375	3.588235…	3.303571…	4.1
5	125	3.706140…	3.844086…	4.003164…	3.613636…	3.833333…	4.14	3.621621…	3.966101…	3.921232…	3.657894…	3.741379…	2.5	3.755434…	4.5	3.925925…	3.845238…	3.875	3.872727…	3.916666…	3.727272…
6	127	4	2.666666…	2.958333…	2.791666…	5	3.5	0.5	2.75	3.333333…	1.25	1	2.791666…	1	2.791666…	0.5	2	4	4	4	5
7	175	2.74375	0.5	4.033333…	2.74375	2.74375	2.74375	2.74375	4.277777…	3.722222…	4.25	0.5	2.74375	0.5	2.74375	2.74375	2.74375	2.74375	2.74375	4.166666…	2.74375
8	255	4.166666…	3	1.4	2.460924…	2.460924…	1	2.714285…	1.571428…	2.75	2.3	3	1	3.5	2.460924…	2.4	1.5	1.333333…	2.4	2.8	5
9	329	1.25	2.928571…	3.458333…	2.708191…	3	2.666666…	2.5	2.708191…	3.111111…	2.708191…	3.5	2.708191…	0.5	2.708191…	2.708191…	2.75	2	2.25	5	3
10	333	3.2	1.75	3.3	2.505952…	2.505952…	2.505952…	2.5	3.75	2.583333…	1.5	2	2.505952…	1.5	2.505952…	2	1	4	3	2.505952…	3
11	358	2	2.166666…	3.634615…	3.5	2.884207…	3.25	2.916666…	3.844827…	2.96875	4	2.5	0.75	4	2.884207…	4	4	2.884207…	0.5	2.25	2.75
12	416	2.307692…	2.727272…	3.685185…	0.5	4	2.75	1.6	2.892857…	3	1	3.25	4	2.4	2.474932…	0.5	2.785714…	2	3.5	0.5	3.625
13	442	0.666666…	0.6	1.076923…	1.107199…	1.107199…	0.5	1.833333…	1.9	1.9	2	0.666666…	1.107199…	1	1.107199…	1.25	1.107199…	1.107199…	0.5	1.107199…	0.5
14	448	2.727491…	2.760089…	2.988352…	2.634831…	3.666666…	2.873737…	2.951338…	2.785714…	2.909186…	3.096296…	2.125	3.921052…	2.623745…	1.25	3.346153…	2.801242…	3.5	2.941747…	3.362068…	3.182692…

Cluster 2: Medium ratings 3.7-3.8



	userId	Action	Thriller	Drama	IMAX	Film-No…	Mystery	Adventu…	Romance	Comedy	Children	Horror	Documen…	Sci-Fi	(no gen…	Animati…	Fantasy	Western	Crime	Musical	War
0	1	4.322222…	4.145454…	4.529411…	4.364153…	5	4.166666…	4.388235…	4.307692…	4.277108…	4.547619…	3.470588…	4.364153…	4.225	4.364153…	4.689655…	4.297872…	4.285714…	4.355555…	4.681818…	4.5
1	2	3.954545…	3.7	3.882352…	3.75	3.925849…	4	4.166666…	4.5	4	3.925849…	3	4.333333…	3.875	3.925849…	3.925849…	3.925849…	3.5	3.8	3.925849…	4.5
2	4	3.32	3.552631…	3.483333…	3	4	3.478260…	3.655172…	3.379310…	3.509615…	3.8	4.25	4	2.833333…	3.638532…	4	3.684210…	3.8	3.814814…	4	3.571428…
3	5	3.111111…	3.555555…	3.8	3.666666…	3.564404…	4	3.25	3.090909…	3.466666…	4.111111…	3	3.564404…	2.5	3.564404…	4.333333…	4.142857…	3	3.833333…	4.4	3.333333…
4	6	3.609375	3.544117…	3.614285…	4.666666…	2.5	3.733333…	3.893617…	3.614285…	3.370078…	3.617021…	3.263157…	3.631439…	3.476190…	3.631439…	4.071428…	3.538461…	3.818181…	3.285714…	4.166666…	3.583333…
5	8	3.333333…	3.75	3.789473…	4.5	3.848950…	4	3.545454…	3.5	3.208333…	4.25	4.5	3.848950…	3.25	3.848950…	5	3.25	3	3.888888…	5	3.666666…
6	13	4.2	3.818181…	3.5	3.806993…	3.806993…	4.5	4.166666…	3.666666…	3.272727…	3.806993…	4	3.806993…	4.2	3.806993…	3.806993…	3	4	3.666666…	3.806993…	3.5
7	15	3.203389…	3.431818…	3.740740…	3.305555…	3.324422…	3.416666…	3.342105…	3.884615…	3.357142…	2.690476…	3.818181…	3.324422…	3.584745…	3.324422…	2.954545…	2.90625	2.5	3.578947…	2.7	4.1
8	16	3.52	3.793103…	3.729166…	4.125	4	3.769230…	3.775	3.653846…	3.74	4	3.5	4	3.6875	3.756694…	3.958333…	3.9375	3	3.75	3.756694…	3.681818…
9	17	4.230769…	4.277777…	4.186274…	4.166666…	3.75	4.05	4.285714…	3.928571…	4.157894…	4.214285…	4.166666…	3.5	4.4	4.156978…	4.4	4.323529…	4.25	4.25	4	4.444444…
10	18	3.588516…	3.792899…	3.894495…	3.675675…	3.875	4.026785…	3.633802…	3.681818…	3.506756…	3.542857…	3.3	3.944444…	3.640909…	3.750983…	3.758064…	3.644230…	4.181818…	3.934782…	3.708333…	3.9375
11	19	2.726666…	2.546798…	2.609271…	3	3.333333…	2.965517…	2.815286…	2.679104…	2.640211…	2.696078…	2.267857…	2.772004…	2.559055…	2.772004…	2.9375	2.833333…	2.545454…	2.902777…	2.837837…	3
12	23	3.537037…	3.723684…	3.642857…	3	3.954545…	4.068181…	3.388888…	3.333333…	3.416666…	3.5	4	3.25	3.911764…	3.625124…	3.8	3.821428…	3.5	3.653846…	3.625124…	3.75
13	24	3.642857…	3.647058…	3.6875	3.85	3.750579…	3.769230…	3.651515…	3.545454…	3.59	4.25	3.625	3.750579…	3.64	3.750579…	4.25	3.733333…	3.750579…	3.913043…	3.5	3.714285…
14	25	4.833333…	4.875	4.958333…	5	4.837301…	4.75	4.75	5	4.5	4.666666…	4.837301…	4.837301…	4.722222…	4.837301…	4.666666…	5	4.837301…	5	4.837301…	5

Cluster 3: Medium ratings 3.5-3.6



	userId	Action	Thriller	Drama	IMAX	Film-No…	Mystery	Adventu…	Romance	Comedy	Children	Horror	Documen…	Sci-Fi	(no gen…	Animati…	Fantasy	Western	Crime	Musical	War
21	78	3.308823…	3.14	2.833333…	4	2.816814…	2.333333…	3	3.5	3.160714…	2.5	2.5	2.816814…	3.026315…	2.816814…	2.25	2.5	2	3.333333…	1.5	3
22	81	2.769230…	3.083333…	2.875	2.5	2.540315…	4	2.454545…	2.333333…	1.875	2	2.540315…	2.540315…	3.833333…	2.540315…	2	1	2.540315…	2.714285…	2	2.666666…
23	82	3.547826…	3.303030…	3.322916…	3.25	1	3.142857…	3.371621…	3.382352…	3.473451…	2.608695…	3.357142…	3	3.526315…	3.195948…	3.038461…	3.25	3.625	3.451923…	3.214285…	3.857142…
24	83	3	3.107142…	3.630136…	3.727272…	3.101484…	3.333333…	3.181818…	3.675	3	3.166666…	2.5	2	3.4	3.101484…	3.45	3.2	0.5	3.392857…	4.125	3.4375
25	87	3	3.666666…	3.909090…	3.886474…	3.886474…	4	3.9	4.25	4.222222…	3.8	3.5	3.886474…	3.642857…	3.886474…	4.333333…	4.3	3.886474…	3.886474…	4	3.886474…
26	88	4.235294…	4.309523…	4.166666…	4.666666…	3.924607…	4.25	4.571428…	3.5	3.208333…	0.5	4.166666…	3.924607…	3.785714…	3.924607…	3.924607…	4.7	4	4.558823…	3.924607…	4.25
27	99	3.809523…	4	4.388888…	5	4.161093…	3.666666…	4	4.7	3.869565…	5	3.333333…	4.161093…	2.833333…	4.161093…	5	4.161093…	4.333333…	4.142857…	5	3.5
28	108	3.6	3.833333…	4.191489…	3	3.899240…	4.75	4	3.974358…	4.107142…	4	2	3.899240…	4.230769…	3.899240…	3.5	3	5	4.6	5	3.5
29	110	3.3	3.633333…	3.931034…	3.528348…	3.528348…	3.875	3.7	3.8125	3.75	2	2.25	5	2.7	3.528348…	3.528348…	3.528348…	3.528348…	4	3.528348…	3.916666…
30	112	4.111111…	3.75	3.56	4.583333…	3.685929…	4.6	3.880952…	2.166666…	2.642857…	3	4.25	3.685929…	4.194444…	3.685929…	3.5	3.5	4	3.821428…	3.5	3.6
31	113	3.125	3.46875	3.815217…	3.838643…	4.5	3.55	4.4	3.741176…	3.580645…	4.375	3.333333…	3.838643…	3.25	3.838643…	4	3.666666…	3.838643…	3.8125	4.2	4.6
32	116	3.766666…	3.522727…	3.147058…	1	3.341861…	3.8	3.5	3.580645…	3.393939…	3.2	3.0625	3.341861…	3.576923…	3.341861…	3.5	3.222222…	4.5	3.181818…	3.857142…	3
33	120	3.571428…	3.333333…	3.571428…	3.308219…	3.308219…	3	3.555555…	3	3.2	3	3	3.308219…	3.75	3.308219…	3	2.666666…	3.308219…	5	2.666666…	3.308219…
34	126	3.3125	3.5	3.5	4.333333…	3.576225…	5	3.25	4.2	3	3.4	2	3.576225…	2.666666…	3.576225…	4	3.666666…	4	3.3	4	3.666666…
35	133	2.642857…	3	3	3	2.953781…	3.666666…	2.571428…	3.25	3	3	2.5	2.953781…	3.25	2.953781…	3	3	2	2.666666…	3	3.666666…

Problem 2:

3. Data distribution regarding the numbers of samples per class in train/validation/test data

```
Train data distribution
gender
[-1.  0.  1.]
[  2 215 183]
torso_type
[0. 1.]
[150 250]
torso_colour
[-1.  0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
[  4 116  24  21  19  61   6  16  15  14  96   8]
leg_type
[0. 1.]
[270 130]
leg_colour
[-1.  0.  1.  2.  3.  4.  5.  6.  8.  9.]
[  6 177 116  28   1  46   1   4   3  18]
luggage
[0. 1.]
[245 155]


Val data distribution
gender
[-1.  0.  1.]
[ 1 80 39]
torso_type
[0. 1.]
[69 51]
torso_colour
[ 0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
[24 15  6  2 22  1  6  6  8 26  4]
leg_type
[0. 1.]
[86 34]
leg_colour
[ 0.  1.  2.  3.  4.  5.  6.  9. 10.]
[33 38 15  4 15  2  1  7  5]
luggage
[0. 1.]
[63 57]


Test data distribution
gender
[-1.  0.  1.]
[ 8 98 90]
torso_type
[0. 1.]
[ 75 121]
torso_colour
[ 0.  1.  2.  3.  4.  5.  6.  7.  8.  9. 10.]
[51 23 12  9 21  5  7  5 19 35  9]
leg_type
[0. 1.]
[114  82]
leg_colour
[ 0.  1.  2.  3.  4.  6.  7.  9. 10.]
[69 63  8  7 27  6  2 13  1]
luggage
[-1.  0.  1.]
[  9 126  61]
```

4. Model design

Layer (type)	Output Shape	Param #	Connected to
Input_image (InputLayer)	[(None, 100, 60, 3)]	0	
conv2d_8 (Conv2D)	(None, 100, 60, 8)	224	Input_image[0][0]
conv2d_9 (Conv2D)	(None, 100, 60, 8)	584	conv2d_8[0][0]
spatial_dropout2d_4 (SpatialDro	(None, 100, 60, 8)	0	conv2d_9[0][0]
batch_normalization_4 (BatchNor	(None, 100, 60, 8)	32	spatial_dropout2d_4[0][0]
activation_4 (Activation)	(None, 100, 60, 8)	0	batch_normalization_4[0][0]
max_pooling2d_3 (MaxPooling2D)	(None, 50, 30, 8)	0	activation_4[0][0]
conv2d_10 (Conv2D)	(None, 50, 30, 16)	1168	max_pooling2d_3[0][0]
conv2d_11 (Conv2D)	(None, 50, 30, 16)	2320	conv2d_10[0][0]
spatial_dropout2d_5 (SpatialDro	(None, 50, 30, 16)	0	conv2d_11[0][0]
batch_normalization_5 (BatchNor	(None, 50, 30, 16)	64	spatial_dropout2d_5[0][0]
activation_5 (Activation)	(None, 50, 30, 16)	0	batch_normalization_5[0][0]
max_pooling2d_4 (MaxPooling2D)	(None, 25, 15, 16)	0	activation_5[0][0]
conv2d_12 (Conv2D)	(None, 25, 15, 32)	4640	max_pooling2d_4[0][0]
conv2d_13 (Conv2D)	(None, 25, 15, 32)	9248	conv2d_12[0][0]
spatial_dropout2d_6 (SpatialDro	(None, 25, 15, 32)	0	conv2d_13[0][0]
batch_normalization_6 (BatchNor	(None, 25, 15, 32)	128	spatial_dropout2d_6[0][0]
activation_6 (Activation)	(None, 25, 15, 32)	0	batch_normalization_6[0][0]
max_pooling2d_5 (MaxPooling2D)	(None, 12, 7, 32)	0	activation_6[0][0]
conv2d_14 (Conv2D)	(None, 12, 7, 64)	18496	max_pooling2d_5[0][0]
conv2d_15 (Conv2D)	(None, 12, 7, 64)	36928	conv2d_14[0][0]
spatial_dropout2d_7 (SpatialDro	(None, 12, 7, 64)	0	conv2d_15[0][0]
batch_normalization_7 (BatchNor	(None, 12, 7, 64)	256	spatial_dropout2d_7[0][0]
activation_7 (Activation)	(None, 12, 7, 64)	0	batch_normalization_7[0][0]
flatten_1 (Flatten)	(None, 5376)	0	activation_7[0][0]
dense_6 (Dense)	(None, 256)	1376512	flatten_1[0][0]
dense_7 (Dense)	(None, 256)	1376512	flatten_1[0][0]
dense_8 (Dense)	(None, 256)	1376512	flatten_1[0][0]
dense_9 (Dense)	(None, 256)	1376512	flatten_1[0][0]
dense_10 (Dense)	(None, 256)	1376512	flatten_1[0][0]
dense_11 (Dense)	(None, 256)	1376512	flatten_1[0][0]
gender_out (Dense)	(None, 2)	514	dense_6[0][0]
torso_out (Dense)	(None, 2)	514	dense_7[0][0]
torso_colour_out (Dense)	(None, 11)	2827	dense_8[0][0]
leg_out (Dense)	(None, 2)	514	dense_9[0][0]
leg_colour_out (Dense)	(None, 11)	2827	dense_10[0][0]
luggage_out (Dense)	(None, 2)	514	dense_11[0][0]

Total params: 8,340,870
Trainable params: 8,340,630
Non-trainable params: 240

The code

```
inputs = layers.Input((100, 60, 3), name="Input_image")
x = layers.Conv2D(filters=8, kernel_size=(3,3), padding='same', activation='relu')(inputs)
x = layers.Conv2D(filters=8, kernel_size=(3,3), padding='same', activation=None)(x)
x = layers.SpatialDropout2D(spatial_dropout)(x)
x = layers.BatchNormalization()(x)
x = layers.Activation('relu')(x)
x = layers.MaxPool2D(pool_size=(2, 2))(x)

x = layers.Conv2D(filters=16, kernel_size=(3,3), padding='same', activation='relu')(x)
x = layers.Conv2D(filters=16, kernel_size=(3,3), padding='same', activation=None)(x)
x = layers.SpatialDropout2D(spatial_dropout)(x)
x = layers.BatchNormalization()(x)
x = layers.Activation('relu')(x)
x = layers.MaxPool2D(pool_size=(2, 2))(x)

x = layers.Conv2D(filters=32, kernel_size=(3,3), padding='same', activation='relu')(x)
x = layers.Conv2D(filters=32, kernel_size=(3,3), padding='same', activation=None)(x)
x = layers.SpatialDropout2D(spatial_dropout)(x)
x = layers.BatchNormalization()(x)
x = layers.Activation('relu')(x)
x = layers.MaxPool2D(pool_size=(2, 2))(x)

x = layers.Conv2D(filters=64, kernel_size=(3,3), padding='same', activation='relu')(x)
x = layers.Conv2D(filters=64, kernel_size=(3,3), padding='same', activation=None)(x)
x = layers.SpatialDropout2D(spatial_dropout)(x)
x = layers.BatchNormalization()(x)
x = layers.Activation('relu')(x)
encoded = layers.Flatten()(x)

def gender_out(encoded):
    x = layers.Dense(256, activation="relu")(encoded)
    gender = layers.Dense(2, activation="softmax", name="gender_out")(x)
    # gender output: 0 = male, 1 = female
    return gender

# v2
def torso_out(encoded):
    x1 = layers.Dense(256, activation="relu")(encoded)
    torso_out = layers.Dense(2, activation="softmax", name="torso_out")(x1)
    # torso clothing: 0=long, 1=short

    x2 = layers.Dense(256, activation="relu")(encoded)
    torso_colour_out = layers.Dense(11, activation="softmax", name="torso_colour_out")(x2)
    # torso colour: 0 (black), 1 (blue), 2 (brown), 3 (green), 4 (grey), 5 (orange), 6 (pink),
    return torso_out, torso_colour_out

def leg_out(encoded):
    x1 = layers.Dense(256, activation="relu")(encoded)
    leg_out = layers.Dense(2, activation="softmax", name="leg_out")(x1)

    x2 = layers.Dense(256, activation="relu")(encoded)
    lego_colour_out = layers.Dense(11, activation="softmax", name="leg_colour_out")(x2)
    return leg_out, lego_colour_out

def luggage_out(encoded):
    x = layers.Dense(256, activation="relu")(encoded)
    luggage_out = layers.Dense(2, activation="softmax", name="luggage_out")(x)
    return luggage_out

gender = gender_out(encoded)
torso, torso_colour = torso_out(encoded)
leg, leg_colour = leg_out(encoded)
luggage = luggage_out(encoded)
```

References

Keras API reference. Keras. Retrieved 4 June 2022, from https://keras.io.

Scikit-learn.org. (2022). Retrieved 4 June 2022, from https://scikit-learn.org.