Project Mile Stone 4

Tianye Song(tianyes), Udaikaran Singh (udaikars), Luís Gomes (lfgomes), Liangwei Zhang (liangwez), Tushar Vatsa(tvatsa)

April 2022

1 Process for Conceptual Analysis of Potential Problems

For developing drift and feedback loops in our machine learning systems, we first developed a requirement document on the aspects that we wanted our system to achieve. This includes aspects like the uptime of the system, accuracy metrics that we looked for our recommender system to achieve, fairness metrics, and the potential threats we looked for our model to be safeguarded against. We wanted our requirement document to be generally high-level with limited details on the implementation. From this requirement document, we created a mindmap to analyze these different aspects of our system and the intersections of these different ideas. Lastly, we made fault tree in order to specifically look at the potential threats to our system and how they can be mitigated.

1.1 Requirements Document

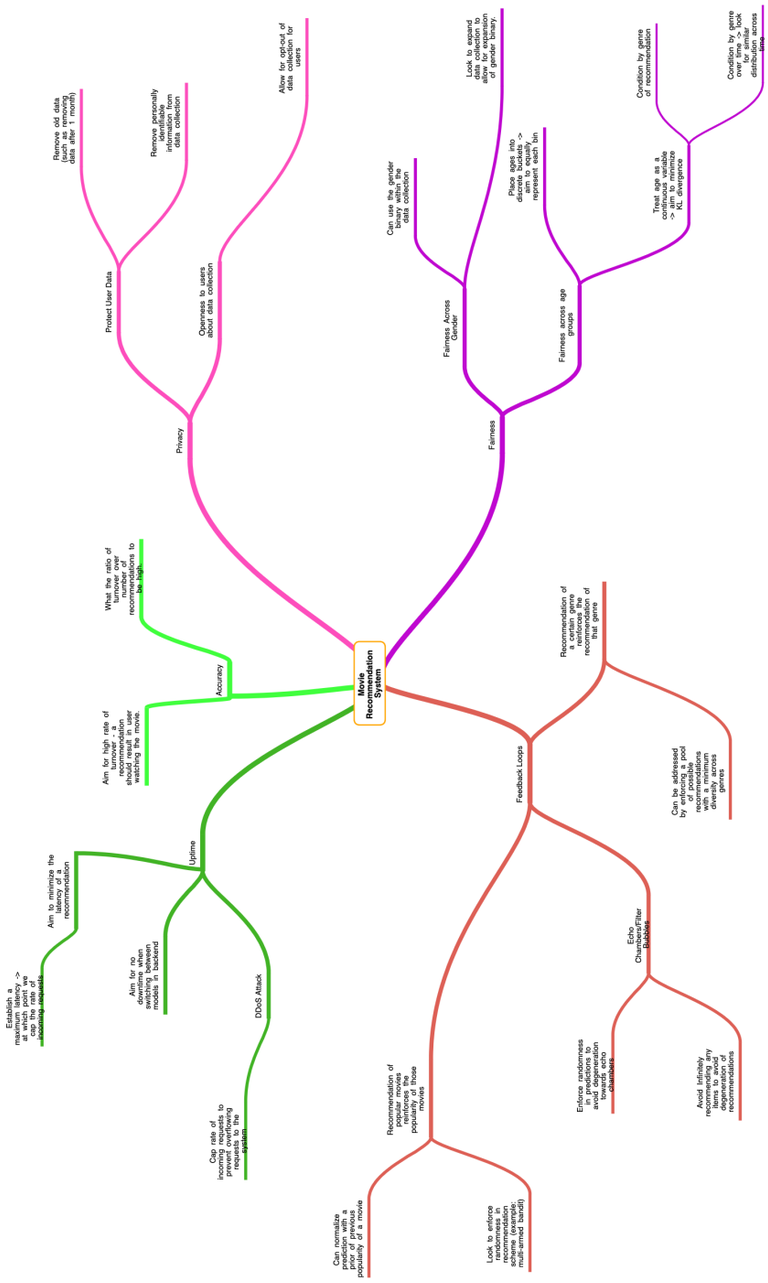
First, we look to a set of requirements that can lead to problems in our system. A list of requirements is presented in table 1.

Table 1: Critical Requirements.

Subgroup	Requirement	Measurement/Method example
Uptime	We want our system to be available for use for users at all time. This includes when experimenting on different models, we want our system to work for users regardless of differences in the models used.	Percentage of uptime vs. total time.
Privacy	We want to ensure that the data collected about users and their behavior is not revealed or can be inferred in the recommendations.	ability to address adversarial attacks.
Latency	We want our system to provide quick responses to recommendation requests.	Time per recommendation.
Accuracy	We want our system to have uptime even in the cases of adversarial attacks, such as data poisoning	Percentage of uptime vs. total time.
Fairness	We want our system to give statistically equivalent recommendations to males and females using our service.	group fairness across gender.
Fairness	We want our system to give statistically equivalent recommendations across age groups	group fairness across age groups.
Accuracy	We want our system to avoid a feedback loop of recommending previously recommended movies.	Accuracy result
Security	We want our system to avoid network attacks like denial of service	Availability of Service

1.2 Mind-map

From the list of requirements, we get more insights by building a mind-map expanding the previous table. By doing this, we have a better understanding of the relationship between each of the possible points of failure.



1.3 Fault Tree for Risk Analysis

Fault trees are a good method to identify possible issues and failure points in a system. We use this method to further investigate specific problems that can remain undetected on the higher levels of analysis, i.e. on the mind-map and requirements table. On this case, we analyse the possible causes that can lead users to get less accurate recommendations than what we would expect or stop getting recommendations at all. Figure 1 show the constructed fault tree.

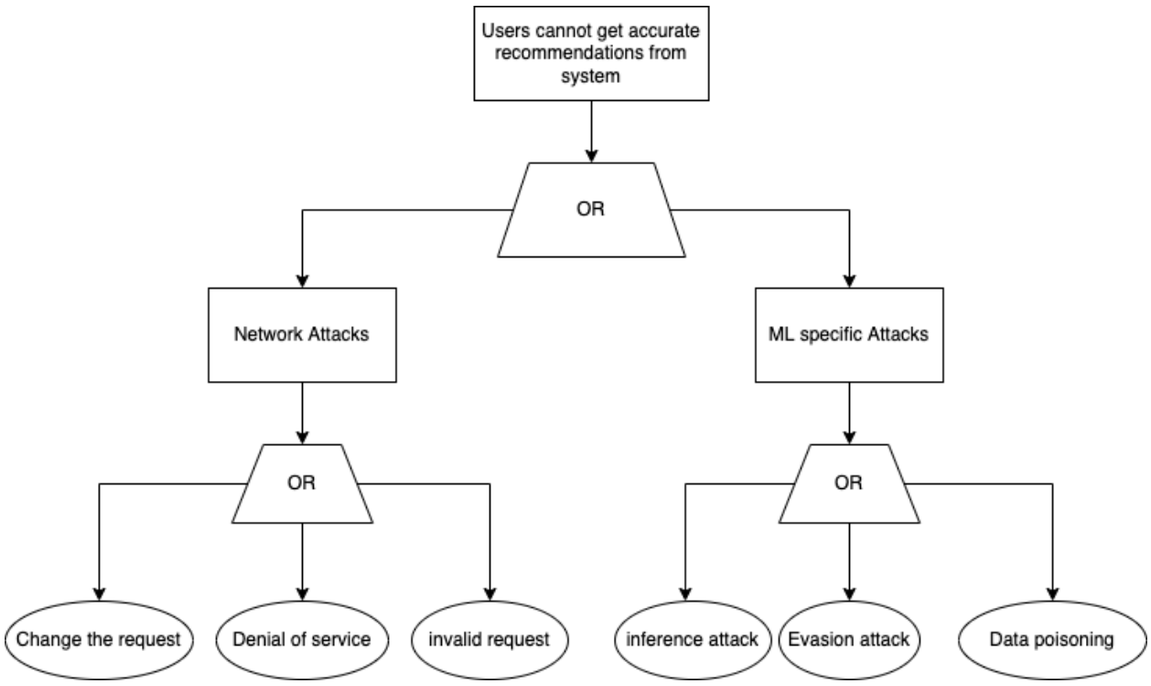


Figure 1: Fault tree analysis.

2 Results for Conceptual Analysis of Potential Problems

According to the process analysis in the first part, For the movie recommendation system, the requirement here is to recommend popular movies that the user could be interested based on the users history preference. Assumptions here are the user's history records could reflect their preference and the training rating data could reflect the possible popular movie among similar users. Therefore, there could be risk that the user's history records could not correctly represent their preference and the machine learning model trained on the data-set couldn't successfully return the possible movies that user would like because of the poisoned data-set or the unsuitable model. Based on these, there might be following feedback loop, fairness and attacks issues.

2.1 Feedback loops

One of the feedback loops includes the future movie recommendation will be reinforced by user's history selection or rating result. The whole process could be described as the service give recommendation to the users and if they select, watch and give ratings for those movies, those rating data will be imported into the rating history data to train the future model and also the user's history records as their preference. If we successfully recommend movies that user like then they will give high ratings for them. The negative consequences, which is also called "echo chamber", is that next time this user will get only similar kind movie recommendation because those movies will have high popularity score among similar users. This is not a good sign because it could narrow a user's content exposure and ultimately shift their worldview. This effect could be detected by measuring the categories of the movies recommended to the user. For example, whether the recommendation result is clustered at only one or two categories.

Recommendation systems have popularity bias, a few popular movies are recommended frequently while the majority of the other items are ignored. Collaborative Filtering generate recommendations suffer from bias against certain groups. Since most of the people prefer to watch popular movies, the users interaction can intensify the bias in the algorithm and it can be worse for people who do not prefer to watch the popular ones. Users select some of the recommended items and the system will add those items to their profiles as part of their interaction history. That's how a feedback loop is formed between user profiles and feedback. The easy way to detect it is to go through the recommendation history of people with different views history and compare the recommendations for each one of them. If there is a match beyond a certain threshold, we can conclude that the same movies are being recommended to different users. It can also be detected by calculating the average number of times each movie has been shown and then comparing the popular ones with the non popular ones. Since most of the users prefer to watch the popular ones, it will work well for them but the taste changes over time. For example : After bashing Chris Rock in the Academy Awards, most people will not prefer to watch Will Smith movies and therefore if the recommendation system keeps showing the popular ones, it will not qualify as a good one.

Researchers have tried to calculate the Kullback Leibler Divergence(KLD) between the initial genre distribution and the genre distribution after t interactions. They found out that the divergence increases with time and concluded that the recommendation model is not able to capture the users preferences when generating the recommendations.

2.2 Fairness issues

One possible fairness issue could happen when it comes to gender. Now, movie recommendation system gives different result for different gender when all the movie ratings history are the same. However, to achieve fairness, different gender should be treated equally. This fairness issue could be examined by using

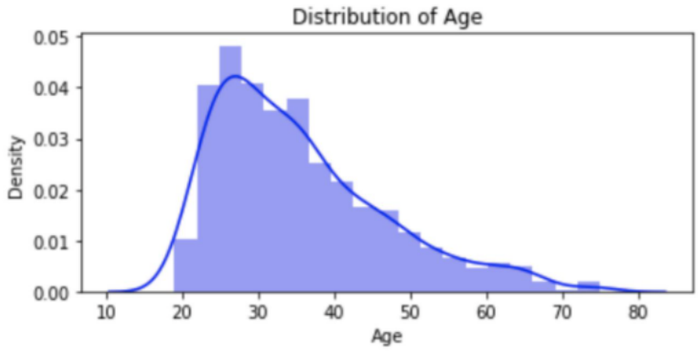


Figure 2: Age distribution in our data

metrics of group fairness, which is the conditional probability that for different gender, if their rating history is the same then check whether the probability of recommending the same movie is the same. If there is significant difference between two groups, then there exists a problem. This might be caused by the imbalanced training data collected from the current movie platform. This unfairness could be reduced by setting different threshold for different groups, adjusting the number of different groups of people in the training data set.

The other fairness issue could happen when it comes to the age feature. In our dataset we have age varying from 8 to 70 and we want the recommendation system to achieve same accuracy on different age groups. For example : A recommendation system showing people of age group [8 - 15] cartoon movies regardless of the user history is certainly biased towards the age feature. Same goes with a recommendation system showing people of age group [60 - 70] "Death in Paradise" recommendation. This is the case when the model is taking a large weight for the age feature. It could also happen when the model is trained on biased data where there is high correlation between the age feature and the genre label. The age can be divided in a binary variable from people in age between [0-25] as 0th group and the rest in group 1 since our data distribution is highly skewed towards left as can be seen from Fig 2. Group Separation can be used to calculate the number of false positives, false negatives, true positives and true negatives for each of the group and we can compare the FPR parity and FNR parity for each group. If they vary significantly, that means the model is biased. It can be reduced in different ways :

1. **Pre processing** : Use log transformation to have a more uniform distribution and then calculate the separation. In our case, it reduces the separation for both gender and age.

2. **Post processing** : Change the thresholds for different age groups and if it reduces the separation, and it doesn't compromise much with accuracy we can stick to it.

2.3 Attacks

The exposed parts of the system include the rating history of users which will be fed back as the model input and also the request to the server API for recommendation result through network. Therefore, except for machine learning specific attack like data poisoning talked about above, there exists other possible attack for the system aiming at the availability of the API request. To be more specific, Denial of service attack could happen in this process. The attacker would send invalid request, send huge amount of requests or change the request to the service. For sending too many requests to the server at the same time, the attacker could create a large number of bogus accounts to overwhelm the service and stop it from working functionally to provide service for other users. For invalid request, if it is error that is not specified by the system, the system could also be crushed. For changing the request, user couldn't get correct recommendation for them. To mitigate this issue, we could limit the request sent by one IP address of the user in a certain time. After detection we can blacklist the IP addresses that have attack history and use more secured network protection.

The other attack can be the data poisoning attack which is basically changing the labels of data points by gaining access of the database by spoofing the encryption pattern. This can be detected if we keep a model in cloud which produces an evaluation on a certain partition of a dataset everyday. If there are significant change in the accuracy values in a very short period of time, it can be concluded that the database is in some kind of attack. We can also use dvc to version our database, infrastructure and model and produce and evaluation everyday to make sure that we are able to produce the same results everyday. It can be mitigated by changing the encryption model from time to time and not keeping it constant. Once detected, we can remove the one on the database and get the one which was versioned previously.

3 Analysis of Problems in Log Data

In this section, we utilized the recommendation history for four weeks, ranging from March 29 to April 26. We performed group fairness tests for users with different populations, including different ages, occupations or genders. We will present the evaluation result by showing the performance on online evaluation metrics for different populations. We will first introduce some online evaluation metrics.

Record accuracy is obtained by dividing the number of recommended movies users have watched by the total number of movies users have watched.

Recommendation accuracy denotes the ratio of users watching the recommended movies on the total number of users requesting. recommendations.

Average Rating is the average rating of those recommended movies which have been watched and rated by users.

Average Top rating score is the average score for top recommendation movies which have been watched and rated by users. It is vital for users' impression on our recommendation system.

The implementation of those online evaluation metrics is the following: https://github.com/cmu-seai/group-project-s22-dsu/blob/master/ml/M4$_feedback.py L75 - L99$

3.1 Group Fairness on Gender

The first issue is the group fairness regarding the gender of users. We pick up the recommendation history as telemetry data from March 29 to April 26. The implementation of the online evaluation metrics is the following: https://github.com/cmu-seai/group-project-s22-dsu/blob/master/ml/M4$_feedback.py L48 - L54$

The result is shown below:

Gender	No. Users	Correct Recommendations	No. Users with correct Recommendations
Male	440036	314174	249455
Female	90121	64456	51209

Table 2: Evaluation Result for Gender populations

Gender	Record accuracy	Recommendation Accuracy	Average Rating	Average Top Rating
Male	43.02%	56.69%	3.78	4.07
Female	43.01%	56.82%	3.80	4.23
Differences	0.02%	0.02%	0.5%	3.9%

Table 3: Evaluation Result for Gender populations

We used the online evaluation metrics to evaluate the group fairness on gender. From above, we can see the recommendation accuracy is slightly higher on female users than males, and female users tend to give higher rating scores for our recommendations.

In addition, it is shown that the record accuracy, the recommendation accuracy, and the average rating score are all very consistent within the difference of 1%. The difference on average rating score on top recommendation movies is a bit more significant than others, considering the significance confidence is 3%.

3.2 Group Fairness on Ages

The second issue is the group fairness regarding the ages of users. We pick up the recommendation history as telemetry data from March 29 to April 26. Since age is a continuous numerical feature, we have categorized it into four different groups. We classify those under 20 as teenagers, adults for those between 20 and 30, seniors for those between 30 and 50, and elderly for those above 50. The pre-processing of age attribute is implemented as follows: https://github.com/cmu-seai/group-project-s22-dsu/blob/master/ml/online$_evaluation.py L267 - L284$

The implementation of the online evaluation metrics is the following: https://github.com/cmu-seai/group-project-s22-dsu/blob/master/ml/M4$_feedback.py L64 - L71$

The result is shown below:

Age	No. Users	Correct Recommendations	No. Users with correct Recommendations
Teenager	24064	17135	13535
Adult	246333	175910	139596
Senior	243896	174238	138501
Elderly	15864	11347	9032

Table 4: Evaluation Result for different age populations

Age	Record accuracy	Recommendation Accuracy	Average Rating	Average Top Rating
Teenager	42.69%	56.25%	3.85	3.93
Adult	42.93%	56.67%	3.79	4.09
Senior	43.13%	56.79%	3.78	4.11
Elderly	43.37%	56.93%	3.81	4.24
Difference	1.60%	1.20%	1.85%	7.89%

Table 5: Evaluation Result for different age populations

We used the online evaluation metrics to evaluate the group fairness on Age. For each metric we calculate the largest relative difference between the minimum value and maximum value obtained on the metric. From above, we can see the record accuracy and recommendation accuracy increases with the increase of the average ages of the user populations. It indicates that in some way, older users may have more fixed tastes on movies whose patterns can be better captured by the recommendation system, compared to younger users who are more likely to watch various movies. However, in reverse, younger users tend to give a better scores on recommendations than older users.

In addition, it is shown that the record accuracy, the recommendation accuracy, and the average rating score are all very consistent within the difference of 2%. However, the difference on average rating score on top recommendation movies is more significant than others, considering the significance confidence is 3%.