REMA - Final Project
Video System Recommendation on KuaiRec Dataset

Mateo Lelong
mateo.lelong@epita.fr - EPITA

1. Introduction

As part of our final project in Recommender System, we explore multiple modeling approaches on a dataset of video recommendations [2]. The goal of this project is to explore multiple recommendation and pre-processing techniques to benchmarks approaches.

1.1. Outline

We develop several models to address the following tasks:
- **Collaborative models** aimed at recommending similar content when user have similar interests
- **Content-based models** focused on evaluating the similarity between two items and recommending items similar to liked items.

In addition to model's implementation, we provide:
1. A comprehensive **exploratory dataset analysis**
2. A detailed overview of **pre-processing** pipelines
3. Descriptions of **hyperparameter tuning** and **training**
4. **Benchmarks** for each models

1.2. Process

For each approach, we proposed the following process:
1. **Model Selection**: Choose suitable models for each task, considering how each contributes to our experimental goals.
2. **Metric Definition**: Select appropriate evaluation metrics adapted to each model and task.
3. **Training and Validation**: Train models using a validation set and monitor performance based on the chosen metrics.
4. **Hyperparameter Tuning**: Analyze the influence of key hyperparameters to identify the best-performing model in each category.
5. **Evaluation**: Assess the selected models on the test set using the test metrics.

2. KuaiRec Dataset Analysis

This dataset consists of user-video interactions from a well-known chinese social media Kuaishou with a dense test and a sparse training interactions matrices

2.1. Generic information

Attribute	Description
Source	KuaiRec Website
Paper	https://arxiv.org/abs/2202.10842
Authors	Gao, Chongming and Li, Shijun and Lei et al.
Format	CSVs
Number of User/Videos	7176 - 10728
Temporal Coverage	Unknown
Provenance	Kuaishou social media
Update Frequency	Last update in 2024 to add captions

Table 1. General information about the KuaiRec dataset

2.2. Dataset Composition

The KuaiRec dataset is composed of multiple subsets that capture both video and user characteristics:

- **small_matrix.csv & big_matrix.csv**: user-item interaction matrices for evaluation and training. The small matrix is dense (99%), while the training matrix is sparse (16%).
- **social_network.csv**: maps user friendships in a one-to-many format (user → [user_id]).
- **item_categories.csv**: links videos to their respective categories (video → [category_id]).
- **item_daily_features.csv**: provides daily video metrics (likes, comments, shares) and static features (upload type, resolution, duration, visibility).
- **user_features.csv**: includes user-level data such as engagement metrics, role (streamer, video maker), social stats (friends, fans), and registration date.
- **kuairec_caption_category.csv**: contains textual content like tags, captions, manual covers, and category labels.

For further details on the dataset structure and features, visit the KuaiRec website.

2.3. Exploratory data analysis (EDA)

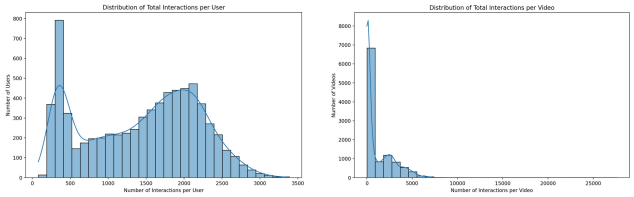
This section presents an exploratory data analysis (EDA) focused on the following areas:

- **User–Video Interactions**: Examines how users interact with videos, including patterns of engagement and data sparsity.
- **User Features**: Looks at user information, such as roles, activity, and social connections, to see how it may influence recommendations.
- **Video Features**: Reviews video attributes, including content tags, metadata, and daily engagement stats, to identify useful signals for recommending content.

The goal is to understand which features are most useful for building a recommendation system.

2.3.1. User-Video Interactions

Distribution of interactions



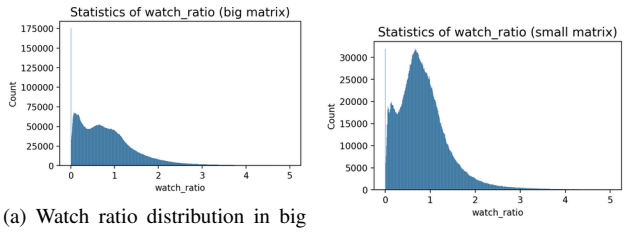
(a) Interaction per user (b) Interaction per video

Figure 1. User and video interaction distributions

The number of interactions per user approximately follows a Gaussian distribution centered around 2000, with a clear outlier peak near 250 interactions. Approximately **99.35% of users** have between 200 and 3,000 total interactions.

In contrast, the distribution of interactions per video is **highly imbalanced**: a **large number of videos receive very few interactions**, while a smaller subset of popular videos attracts significantly more attention. This indicates a strong imbalance that should be taken into consideration as it may affect model performance.

Watch Ratio Distribution



(a) Watch ratio distribution in big matrix (b) Interaction per video

Both the large and small matrices show similar patterns for the watch ratio. However, the large matrix tends to give slightly lower values, which suggests the model might be underestimating watch ratios on the test set.

Video Duration Analysis

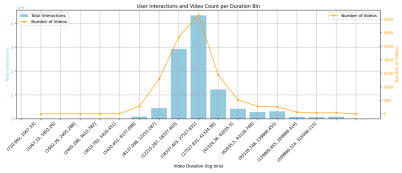


Figure 3. Number of Interactions vs. Video Duration

We plotted the number of interactions per video across different duration bins. The majority of videos (9,821 in total) have durations **between 5 and 40 seconds**, showing a **strong preference for short-form content**.

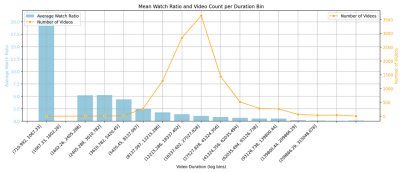


Figure 4. Average Watch Ratio vs. Video Duration

We also analyzed the average watch ratio by video duration. **Shorter videos tend to be watched more completely or rewatched**. An extreme case is seen for videos shorter than 1 second, with an average watch ratio near 20 but this is based on only four videos and should be treated as an outlier.

2.3.2. User Features

Active degree

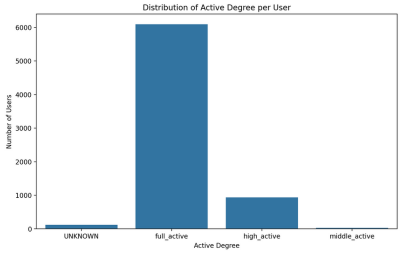
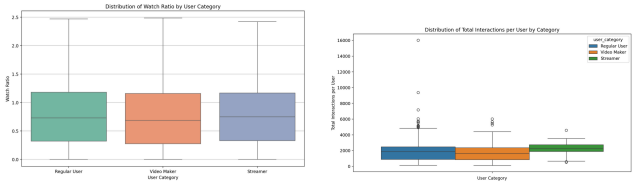


Figure 5. User activity degree repartition

Most users belong to the *full_active* category. Therefore, it would be useful **to encode this information as a binary variable** indicating whether a user is *full_active* or not.

User category analysis



(a) Watch ratio per user category (b) Interaction per user category

Figure 6. User and video interaction distributions

There is **no clear correlation between user category and watch ratio**. However, streamers tend to have a slightly higher number of interactions compared to other user types.

User socials analysis

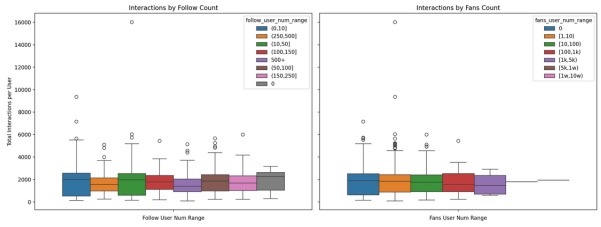


Figure 7. User interaction per fans and follows

We analyzed the number of interactions based on different ranges of followers and fans for each user bin. However, **the plot does not reveal any meaningful patterns**. Users with a large following do not appear to behave differently in terms of their video consumption.

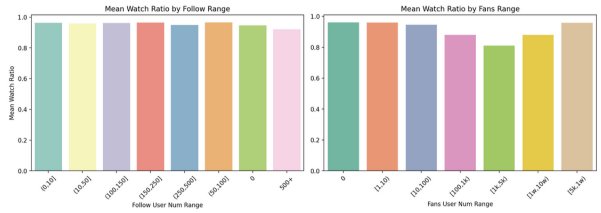
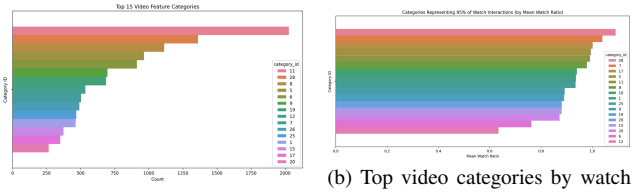


Figure 8. Watch ratio per fans and follows

The number of fans and followers **does not appear to significantly influence** the watch ratio. Across all user groups, the average watch ratio remains relatively stable, ranging between 0.90 and 0.95. Users with 1k to 5k followers show a **slightly lower average watch ratio** of around 0.8, but this difference is marginal and likely not meaningful.

2.3.3. Video Features

Video Category Analysis:



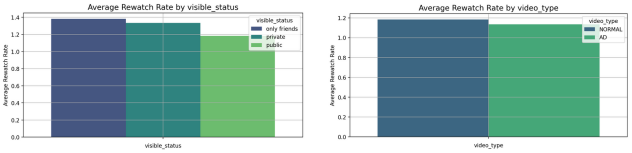
(a) Top video categories (b) Top video categories by watch ratio

Some categories tend to be more viewed than others:

- 11: Life
- 28: Peoples livelihood information
- 5: fashion
- 8: Appearance
- 6: Star Entertainment

It's clear that a small subset of categories accounts for the majority of user engagement, with **14 categories (out of 31) representing 90% of total interactions**.

Visibility Status, Video Type & Upload Type:



(a) Watch Ratio per Upload Type (b) Watch ratio per Video Type

The watch ratio by upload and video type shows that:

- Videos set to **private or friends-only** generally receive **more rewatch** than **public** ones.
- **Advertisement (AD) videos reduce** rewatch rate compared to regular videos.

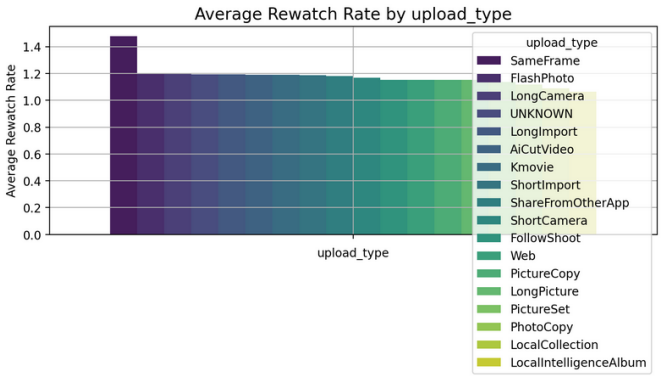


Figure 11. Watch ratio per Video Upload Type

The upload type generally does not significantly impact the average watch ratio, excepted the SameFrame upload type which deviates from this trend.

Video Age:

There is a clear trend showing that **as videos age, they tend to receive fewer views and interactions over time**.

This decline is expected, as **older content typically becomes less visible** and **less engaging to users**. However, some notable outliers exist, particularly in view and like counts, suggesting that **certain older videos regain attention**.

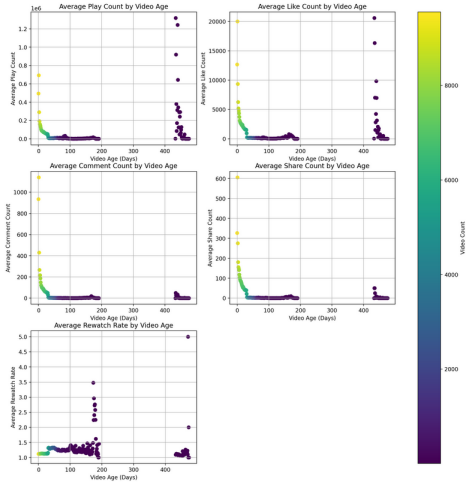
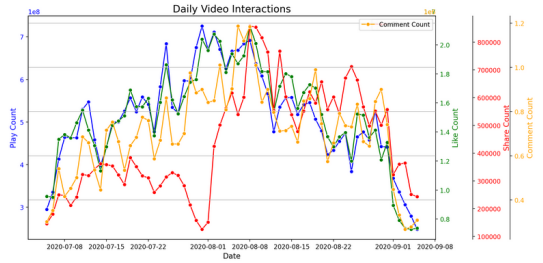


Figure 12. Video Age by metrics

Temporal Trends:



(a) Daily metrics trends

The temporal trends of user interaction metrics such as shares, likes, and comments reveal a **strong correlation between plays and likes**. In contrast, **shares and comments appear to be less correlated** with playback activity. It could be interesting to take daily ratio of non correlated metrics like plays with shares.

3. Collaborative Filtering

3.1. Goal

Generating recommendations based only on the user-item matrix, by identifying and suggesting items aligned with similar user interests.

3.2. Models

The proposed model aims to predict the user's watch ratio, a metric used to indicate their level of interest in a video.

1. **Baseline - User based Nearest Neighbor**: Serves as a starting point to compare more advanced models. It recommends video using similar user interests with a cosine similarity measurement.
2. **Ridge ALS - Matrix Factorization**:
This technique proposed in Chen et al. [1] can be used to approximate a user-item interaction matrix $\hat{Y}$ using matrix factorization into two low-rank embeddings: $\hat{Y} \approx U^T V$
Predictions: $Y = U^T V$
Cost Function:

$$E_{RALS}(U,V) = \|R \cdot (\hat{Y} - U^T V)\|^2 + \lambda \|U\|^2 + \lambda \|V\|^2$$

R is a binary matrix used to mask out NaNs.
Learning: According to the referenced paper, the derivative of the cost function with respect to U and V yields convex subproblems. The global minimum can be found by solving the following update rules iteratively:
   - $V = (UU^T + \lambda_1 I)^{-1} U(R \cdot \hat{Y})$
   - $U = (VV^T + \lambda_2 I)^{-1} V(R \cdot \hat{Y})^T$
3. **Implicit ALS - Matrix Factorization**: This technique, based on the formulation in Tikk and Zibriczky [4], is very similar to the Ridge ALS model but is supposed to solve sparsity issue inherent of our training dataset.
Predictions: see Ridge ALS
Cost Function:

$$E_{IALS}(U,V) = \|C \cdot (\hat{Y} - U^T V)\|^2 + \lambda \|U\|^2 + \lambda \|V\|^2$$

C is a weighted confidence matrix that assigns higher importance to certain observed ratings.
Learning: Similar to Ridge ALS, the derivative of the cost function with respect to U and V yields convex subproblems. The global minimum can be found by solving the following update rules iteratively:
   - $V = (UCU^T + \lambda I)^{-1} U(C \cdot \hat{Y})$
   - $U = (VCV^T + \lambda I)^{-1} V(C \cdot \hat{Y})^T$
4. **Neural Collaborative Filtering (NCF)**: The final approach presented in [3] uses a single-layer neural network to learn latent representations of user-item interactions. This method is capable of capturing non-linear patterns and may outperform ALS and other baseline techniques.
Predictions: $\hat{Y} = U^\mathsf{T} V + b$
Cost Function:

$$E_{\text{NCF}}(U,V,b) = \tfrac{1}{2} R \odot \left\| U^\mathsf{T} V + b - Y \right\|^2 + \lambda \|U\|^2 + \lambda \|V\|^2$$

where R is a binary mask that selects the non-zero (observed) entries of Y, and λ controls L_2 regularization on the embedding matrices.
Learning: Parameters are optimized via (mini-batch) gradient descent until convergence

3.3. Training

3.3.1. Preparation

During data preparation, we construct a **user-item pivot matrix** where the watch ratio serves as a **metric for user interest**. A **validation matrix extracted** excluded to the big-matrix is used **to evaluate hyperparameters** impacts and avoid hyperaparameters overifitting. The **small matrix is not used during training**.

3.3.2. Ground Truth

The ground truth is represented as a **binary matrix** indicating whether a user could be interested in a given item. Several ground truth construction methods were evaluated and the most appropriate for our task is **to mark a user as interested if there is at least one full interaction** (i.e. the watch ratio is higher than 1).

3.3.3. Hyperparameters Tuning

For each model, several hyperparameters were explored. Performance was evaluated using **MSE, and MAE when appropriate**. Note that metrics **should not be compared across different model types**, as they are only meaningful within the same model category.

- **Ridge ALS**: *num_features*, *lambda* (reg factor), *initialization scale*, *regularization*, *normalization*, early stopping is enabled by default, max number of iteration is 100

Ridge ALS

Feat.	λ	Scale	Norm.	MSE	Train Time(s)
50	1.0	0.1		2.49	78
50	1.0	0.1	✓	3.20	80
50	1.0	0.2	✓	3.21	76
200	1.0	0.3	✓	2.54	179
30	0.1	0.2	✓	3.34	73
30		0.2	✓	3.35	78
50		0.5	✓	3.20	76
500		0.5	✓	**1.88**	249

We observe that removing **normalization** appears to improve performance on the training set but in practice, it leads the model to **learn a degenerate solution that tends toward predicting a zero matrix**. Increasing the **regularization strength** and the **number of latent features** both tend to improve generalization and overall performance.
Selected Model: 50 features, $\lambda = 1$, $scale = 0.1$, regularization and normalization enabled.

- **Implicit ALS**: *num_features*, *lambda* (reg factor), *alpha* (weights for existing interaction)

Implicit ALS

Feat.	λ	α	MSE	Train Time(s)
50	1.0	15	0.062	29.95
100	1.0	15	0.055	30.28
100	1.0	50	0.030	27.19
100	1.0	75	**0.023**	31.39
100	10	50	0.034	31.78
100	0.1	15	0.054	33.18
30	0.0	15	0.063	41.5
100	0.0	30	0.038	31.37

We observe that increasing both the **number of latent features** and the **confidence scaling** α generally leads to lower MSE as expected. Moderate regularization ($\lambda > 0$) outperforms no regularization. Extremely high α or feature counts offer diminishing returns in some cases. Training time grows slightly with model size but remains stable across configurations.
Selected Model: 100 features, $\lambda = 1$, $\alpha = 75$

- **NCF**: *num_features*, *lambda*, *learning_rate*, with max number of iterations being 50, normalization enabled

Neural Collaborative Filtering (NCF)

Feat.	λ	Norm.	MAE	Train Time(s)
50	1.0	✓	0.525	49.00
100	1.0	✓	0.533	52.61
100	10	✓	0.544	54.96
30	0.1	✓	0.549	51.03
100	0.1	✓	0.523	56.61
30		✓	0.531	50.78
50		✓	0.530	51.71
100		✓	**0.515**	55.44

Models **without regularization outperform those with regularization**, the best MAE being 0.515 and achieved using 100 features. This indicates that regularization may negatively affect performance in this setup.
Selected Model: 100 features, no regualization

3.4. Evaluation

3.4.1. Evaluation Metrics

The following evaluation metrics have been selected:
- **NDCG@100**
- **MAP@100** (Mean Average Precision)
- **MAR@100** (Mean Average Recall)

3.4.2. Benchmark

The benchmark is calculated as the average performance of the top 100 videos for 40 randomly selected users, averaged over 10 separate evaluation runs.

Model	NDCG	MAP	MAR	Pred. Time(s)
Baseline	0.384	0.450	0.009	
Ridge ALS	0.420	0.450	0.013	55
Impl. ALS	0.322	0.334	0.009	39
NCF	0.121	0.103	0.005	94

Table 2. Benchmark evaluation results for best collaborative models

We can see from the results that surprisingly **Implicit ALS and NCF do not perform well**. Their scores for NDCG, MAP, and MAR are lower than those of the Baseline and Ridge ALS models. In particular, NCF performs the worst overall, which suggests that it may not be well suited for this dataset. On the other hand, **Ridge ALS gives slightly better results than the Baseline** and keeps a similar prediction time, making it the best choice among the collaborative models tested.

3.5. Discussion

3.5.1. Error Analysis & Critics

- Additional metrics, such as novelty, could be explored to assess the model's ability to recommend less popular or unseen content.
- The MAR metric appears unsuitable in this context, as it is heavily influenced by the number of relevant items. Its effectiveness depends on how "relevance" is defined, which can introduce bias and result in misleadingly low scores.
- The ground truth is somewhat arbitrary and may not accurately reflect true user preferences. Additional metrics, such as those used in online testing, should be considered to better evaluate the model's recommendation quality.
- The choice of top-100 for ranking evaluation is arbitrary; alternative cutoffs could have been considered to better align with real-world usage.
- NCF's poor performance during training is quite surprising. This might be due to implementation issues, such as incorrect use of TensorFlow or improper feature scaling.

3.5.2. Model Strengths and Weaknesses

Model	Strengths	Weaknesses
Baseline Model	Simple, interpretable, fast to train	Sensitive to popularity bias
Ridge ALS	Addresses popularity bias	Performs poorly on unseen interactions
Implicit ALS	Better generalization, faster prediction and training	Does not outperform baseline in practice
NCF	Captures non-linear relationships	Poor performance, longer training time

Table 3. Strengths and weaknesses of collaborative filtering models

4. Content based filtering

4.1. Goal

Generating recommendations based on user features, video features, and some custom features from both. The goal is to learn representations that match well together, using the existing watch ratio as a relevance score, while ignoring user-item pairs with no interactions in the training and testing steps.

4.2. Models

1. **Random Forest**: Random Forest Regressor serves as a **baseline model** for content-based methods. It performs regression using the MSE criterion on our custom relevance score derived from watch ratio.
2. **XGBRanker**: XGBRanker is a gradient boosting model tailored for learning-to-rank tasks. It offers specific objective functions like *rank:ndcg*, **enabling direct optimization of evaluation metrics**. This model allows fine-grained control over training and is expected to provide more accurate ranking performance.
3. **Feed-Forward Neural Network (FFNN)**: FFNN aims to capture **non-linear relationships** between users and videos. By learning latent patterns beyond what tree-based models might capture, it can reveal whether deeper feature interactions are essentials.

4.3. Training

4.3.1. Preparation

The most crucial step in building a conetnt-based model lies in the preparation of user, video, and interaction-based features to serve as a foundation for a meaningful latent representation of user preferences and content attributes.

User Features:

We build a user representation by combining static user data with behavioral patterns. Based on our EDA, the following features are especially useful:

- **Engagement & Activity Metrics**: *follower_fan_ratio* (ratio of followed users to number of fans), *popularity_score* (sum of followed users and fans), *video_id_count* (number of videos the user interacted with).
- **Binary Categorical Features**: *is_user_active* (indicates whether a user is flagged as "full_active"), *active_degree*, *lowactive_period*, *live_streamer*, *video_author* (derived from user-level categorical attributes).
- **Preference-Based Features**: *preferred_category*, *preferred_upload_type*, *preferred_video_type* (each based on the category/type with the highest total watch ratio for the user), *preferred_duration* (average duration of videos watched by the user).

Video Features:

The video representation combines content characteristics and user interaction statistics to describe each video in terms

of popularity, type, and format. These features help model how users engage with different kinds of content.

- **Engagement Ratios**: *like_play_ratio*, *comment_play_ratio*, *share_play_ratio*, *follow_play_ratio*, *like_cancel_ratio*, *like_to_comment_ratio*
- **Binary Categorical Feature**: *is_add*:
- **Textual Features for Embedding**: *tags_caption_cover*: Concatenation of the 10 first parsed tags from *topic_tag*, caption text and manual_cover_text.
- **Other Merged Features**: Video metadata from *video_features* and category information from *video_categories*.

Pipeline Overview

1. **Video Feature Aggregation**
2. **User Feature Derivation**
3. **Interaction Cleaning and Enrichment:**
   - Drops duplicates using *(user_id, video_id)*.
   - Computes engagement (derived from watch ratio)
   - *is_weekend* feature from timestamps.
4. **Merging Representations**
5. **Feature Engineering:** *category_match*, *upload_type_match*, *duration_diff*
6. **Missing Value Handling:** impute with median for numerical features and impute with mode fr categorical.
7. **Scaling / Encoding:** OHE for categorical columns, standard scaling for numerical columns.
8. **Cleanup:** Casts features to appropriate types for memory efficiency (*int16*, *float32*)

4.3.2. Ground Truth

The exact same process has been used to evaluate best ground truth. The most appropriate method is to **mark a user as interested if the engagement score** (which is derived from the watch ratio) **belongs to the first top 10% of non NaNs engagement scores**.

4.3.3. Hyperparameters Tuning

- **Random Forest Regressor**: *criterion*, *max_depth*, *n_estimators*

  Training was performed using KFold technique with MSE as objective function.

3-Fold CV Results (NDCG@100)

Depth	#Trees	Criterion	NDCG	Train Time (s)
6	25	MSE	0.4571	192.54
10	50	MSE	0.4706	449.55
12	80	MSE	**0.4720**	837.89

Increasing the number of trees and depth slightly improves performance, with the best NDCG score of 0.4720. These results are better than those from collaborative filtering models, suggesting that including user and video features helps improve ranking quality.

Selected Model: 10 depth, 50 features, with mse criterion

- **XGBRanker**: *learning_rate*, *objective*, *max_depth*, *n_estimators*, *tree_method*

  Training was performed using KFold technique to avoid overfitting on hyperparameters with default learning rate of 0.05, rank:ndcg as the objective function.

3-Fold CV Results (NDCG@100)

Depth	#Trees	Method	NDCG	Train Time (s)
4	20	hist	0.6789	164.12
6	50	hist	0.6822	250.03
10	35	hist	**0.6828**	208.92
12	25	hist	0.6827	172.56
4	25	approx	0.6849	304.39
8	75	approx	**0.6895**	795.09

The *approx* method gives slightly higher NDCG@100 scores, but the improvement is marginal compared to the longer training time. We choose the *hist* method for its better trade-off between performance and efficiency. The number of estimators help the model converges faster but comes at a greater training time cost.
Selected Model: 10 of max depth, 35 estimators and hist method

- **FFNN**: *n_layers*, *dropout*, *LR*, *L2 reg.*, *batch size*, *activation*

H. Layers	Drop.	LR / L2 / BS / Act	MSE	Time (s)
64, 32	0.3	0.01 / 0.0005 / 512 / relu	5.2708	920.74
128, 64, 32	0.3	0.01 / 0.0005 / 512 / relu	5.2788	1343.54
64, 32, 16	0.3	0.1 / 0.0005 / 512 / relu	5.3925	599.12
64, 32, 16	0.3	0.0001 / 0.0005 / 512 / relu	**5.2561**	281.80
64, 32, 16	0.3	0.001 / 0.0005 / 512 / elu	5.2566	277.80
64, 32, 16	0.3	0.001 / 0.0005 / 128 / relu	5.2627	706.83
64, 32, 16	0.3	0.001 / 0.0005 / 512 / relu	5.2568	277.73

Table 4. Summary of Neural Network Training Configurations with loss and training time

The configuration with hidden layers [64, 32, 16], dropout 0.3, and learning rate 0.0001 achieved the lowest loss (5.2561), showing that a smaller learning rate improves performance but it also significantly reduced training time compared to larger networks. Increasing network depth or size did not yield meaningful gains and led to much longer training times. Activation functions and batch size had marginal effects.
Selected Model: [64, 32, 16], LR = 0.0001

4.4. Evaluation

4.4.1. Evaluation Metrics

The following metrics have been selected:

- **NDCG@100**
- **MAP@100**
- **MAR@100**

4.4.2. Benchmark

We provide a benchmark on a small subset of the small matrix using the selected metrics for 10 random users.

Model	NDCG	MAP	MAR	Pred. Time(s)
RF	0.6269	0.5730	0.1746	0.0513
XGB	0.6008	0.5520	0.1674	0.1928
FFNN	0.5325	0.4920	0.1492	1.4178

Table 5. Benchmark evaluation results for best content-based models.

The RF model **gives the best results overall**, with the highest NDCG and MAP scores and the shortest prediction time. The FFNN **performs worse** than RF and XGBoost. It is possible the FFNN model may be overfitting the training data even though dropout and regularization were used. It's possible that the features used don't work well with neural networks, and more work is needed to improve how the FFNN uses the data.

4.5. Discussion

4.5.1. Model Post-Hoc Interpretation

- **Random Forest**
  The RF model offers built-in interpretability by ranking feature importances, allowing us to understand which inputs most significantly impact the final recommendations.

F.	Name and Interpretation	Score
f6	like_cancel_ratio — Indicates quality of engagement.	0.303
f21	upload_type_match — Strong match between user and video upload type improves personalization.	0.225
f18	is_user_active — Active users are more likely to engage.	0.056
f16	popularity_score — Virality helps estimate expected watch behavior.	0.039
f9	like_to_comment_ratio — Suggests user engagement style.	0.034

Table 6. Top 5 most important features in the RF.

Features such as like cancel ratio and like-to-comment ratio have a strong influence on the model's predictions. The upload type match also plays an important role, indicating that aligning video format with user preferences improves recommendation. User activity and video popularity also contribute and may reflect that active users may behave differently than non active users.

- **XGBRanker**
  The XGBRanker model offers the same feature, we can also exploit it for our post-hoc interpretation.

F.	Name and Interpretation	Score
f6	like_cancel_ratio — Users canceling likes may reflect low-quality engagement.	251.16
f4	comment_play_ratio — High comment-to-play ratio shows content sparks reactions.	139.92
f37	upload_type_ShortCamera — Short camera uploads are likely preferred by users.	106.56
f29	upload_type_LongCamera — Long-form camera videos may attract niche interest.	101.63
f21	upload_type_match — Matching user and video upload types shows personalization.	27.29
f28	upload_type_LocalIntelligenceAlbum — Possibly trending or contextually relevant.	26.35
f20	category_match — Indicates shared interests between user and video.	21.91

Table 7. Top 7 most important features in XGBRanker

It is clear that **engagement-related features**, such as like cancellation and comment ratios, play a dominant role in the model's predictions. Interestingly, the **type of video upload** also contributes significantly, suggesting that **content format** influences recommendations. Additionally, **category matching** proves useful in predicting watch ratios, which aligns with our expectations.

4.5.2. Model Strengths & Weaknesses

Model	Strengths	Weaknesses
RF	short prediction time, explainable model	exponentially increasing training time as the model growth
XGBRanker	highly scalable, performs well on the training set, explainable model	Longer training time, higher memory footprint, overfitting issues
FFNN	catch non linear relationship	not really explainable

Table 8. Comparison of content-based models

4.5.3. Error Analysis & Critics

- Due to high memory usage, text-based features such as manual cover text, captions, and category descriptions were not utilized. Incorporating these features could **significantly improve recommendation quality**, but would also greatly increase training and prediction times, as well as require more complex models to effectively process the textual data.
- The chosen ground truth may not accurately reflect true user interest. Some users engage with many videos, while others are more selective, so using the top 10% engagement scores might not be a reliable measure of relevance.

To better evaluate the model's effectiveness, online testing with alternative metrics would be necessary.
- **Alternative preprocessing techniques could be explored** to understand their impact on the model's performance.
- **Dimensionality reduction appears to be an interesting approach** to decrease model complexity, especially since **only 5 to 7 features account for more than 80% of the total feature importance** in RF and XGB models.

5. Conclusion

In this project, we explored recommendation strategies on the KuaiRec dataset using both **collaborative filtering** and **content-based** approaches to model user-video interactions. The objective was to compare different models and feature sets to enhance **recommendation quality**.

We began with an **exploratory data analysis (EDA)** to better understand user behavior, video patterns, and the dataset structure. From this, we selected and engineered **meaningful features** to represent users and items.

For each task, we:

- Built **custom pipelines** for training and evaluating various models, guided by insights from the EDA
- Applied **ranking and regression metrics** (NDCG@100, MAE, MSE) alongside **cross-validation** for less complex models and **hyperparameter tuning** for reliable evaluation
- Interpreted model performance to understand the influence of **feature sets** and **learning methods**
- Discussed the strengths and limitations of each technique, especially in relation to the dataset's characteristics

Overall, content-based methods **showed stronger performance** than collaborative models. Surprisingly, **FFNN tends to underperform** despite regularizations and carefully selected hyperparameters. This may indicate issues with its implementation, **unsuitable features** for this architecture, or **insufficient complexity** in the input data for deep architectures to provide value.

Improvements could come from:

- More advanced **feature engineering**, including **temporal**, **contextual**, or **sequence-based information**;
- Incorporating alternative offline metrics such as **serendipity**, **coverage**, or **diversity** to offer a broader evaluation of model quality;
- Implementing **hybrid** or **sequential-aware models**.

Ultimately, **online testing** (A/B testing or user feedback loops) will be essential for validating **real-world recommendation performance**.

References

[1] Bo-Wei Chen, Wen Ji, Seungmin Rho, and Yu Gu. Supervised collaborative filtering based on ridge alternating least squares and iterative projection pursuit. *IEEE Access*, 5:6600–6607, 2017. 4

[2] Chongming Gao, Shijun Li, Wenqiang Lei, Jiawei Chen, Biao Li, Peng Jiang, Xiangnan He, Jiaxin Mao, and Tat-Seng Chua. Kuairec: A fully-observed dataset and insights for evaluating recommender systems. In *Proceedings of the 31st ACM International Conference on Information amp; Knowledge Management*, page 540–550. ACM, 2022. 1

[3] Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th International Conference on World Wide Web*, page 173–182, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steering Committee. 4

[4] Domonkos Tikk and Dávid Zibriczky. Fast als-based matrix factorization for explicit and implicit feedback datasets. In *Proceedings of the 4th ACM Conference on Recommender Systems (RecSys)*, pages 71–78. ACM, 2010. 4