Video Recommendation System Porject: Summary Report

Jack Bosco, Bartolomeo Attolico, Godric Li, Seongho Lee

December 13, 2024

1 Background

Recommender systems are an integral part of modern digital platforms, providing personalized suggestions to users across domains such as e-commerce, video streaming, social media, and news platforms. These systems enhance user experience and engagement by predicting preferences and delivering relevant content. A significant challenge in designing such systems is balancing exploration (recommending novel items to discover new user preferences) with exploitation (maximizing user satisfaction based on known preferences). This trade-off is critical in sequential decision-making scenarios where recommendations must be updated dynamically based on user feedback.

1.1 Multi-Armed Bandits in Recommender Systems

Multi-Armed Bandits (MAB) are a well-suited framework for this dynamic decision-making problem. MAB models treat each recommendation opportunity as a "bandit arm" pull, aiming to maximize cumulative rewards (e.g., user engagement) over time while minimizing regret. The LinUCB (Linear Upper Confidence Bound) algorithm, a contextual bandit approach, is particularly effective as it leverages contextual features about users and items to inform decisions. LinUCB uses linear models to estimate the reward of each arm (recommendation) and incorporates uncertainty through an upper confidence bound, enabling efficient exploration and exploitation.

1.2 The KuaiRec Dataset

The KuaiRec dataset provides an exceptional foundation for experimenting with MAB-based recommendation systems due to its unique characteristics:

1. High Density: The dataset includes a nearly fully observed user-item interaction matrix with a 99.6% density in its "small matrix" subset. This completeness ensures a robust basis for training and evaluating recommendation models without the common challenges of sparse data.

2. Rich Features: KuaiRec incorporates diverse user and item metadata, such as video categories, play durations, and user demographics, which are crucial for contextual recommendation models like LinUCB.

3. Dynamic Contexts: The inclusion of temporal information and evolving user interactions allows for simulating real-world recommendation scenarios.

4. Scalability: The dataset's structure enables efficient preprocessing, transformation, and merging of features, making it suitable for scaling experiments across multiple recommendation strategies.

1.3 This Study

The study focuses on implementing the LinUCB algorithm on the KuaiRec dataset. The workflow involves:

1. Data Preparation: Preprocessing the small matrix to handle missing values, aggregate watch ratios by video category, and merge contextual features.

2. Contextual Bandit Model: Training a LinUCB model with user-item contexts to recommend video categories dynamically. The model predicts rewards for each arm (category) and updates parameters iteratively based on observed rewards.

3. Evaluation: Simulating recommendations over multiple trials and measuring performance through cumulative regret, highlighting the trade-offs between exploration and exploitation.

The unique combination of LinUCB's capability and KuaiRec's data richness provides valuable insights into the effectiveness of contextual bandit models for personalized recommendations.

2 Data and Experimental Setup

In this study, we utilize the KuaiRec dataset, which offers a highly dense user-item interaction matrix coupled with rich contextual information about both users and items. As described, the "small matrix" subset within KuaiRec is 99.6% dense, making it particularly suitable for reliable training and evaluation. We focus on the subset due to its completeness, ensuring minimal uncertainty caused by missing data and reducing the need for extensive imputation or filtering.

2.1 Data Description and Preprocessing

The KuaiRec dataset comprises interactions between users and videos, along with item-specific metadata (e.g., categories, temporal information) and user-specific features (e.g., demographics, social connections, usage patterns). Each user-video interaction record includes a watch ratio, indicating the proportion of the video consumed by the user, which serves as a proxy for user engagement or reward.

To prepare the data for our contextual bandit model:

1. **Data Cleaning and Integration:** We first cleaned the dataset by addressing missing values. NaN values in numerical columns were replaced by median values to preserve as many data points as possible. Non-numeric or malformed entries were either removed or assigned a placeholder value (e.g., -124 for unknown categories).

2. **Feature Engineering:** Contextual information was leveraged to form user and item feature vectors. User features (such as demographic and behavioral traits) were merged with aggregated item-level categories. We pivoted the data to produce a user-by-category matrix that represents how each user engaged with each first-level video category, summing watch ratios across relevant videos. These transformations allowed us to directly incorporate contextual vectors into the LinUCB framework.

3. **Normalization and Encoding:** Where appropriate, categorical features were encoded, and time-based features were converted to numeric values representing time since the earliest observation. This step ensured that temporal sequences could be easily interpreted by the bandit model.

4. **Train-Test Split:** We split the data into training and testing sets. The training set was used to update the LinUCB model parameters iteratively, while the testing set served to evaluate the model's performance out-of-sample, ensuring a fair assessment of its ability to generalize.

3 Model Formulation and Estimation

3.1 Contextual Multi-Armed Bandit Framework

We frame the recommendation problem as a Contextual Multi-Armed Bandit (CMAB). In this setting:

- **Context:** At each decision point (i.e., each recommendation opportunity), we observe a feature vector describing the user and current environment. This vector includes user-level features (such as demographics and historical engagement patterns) as well as item-level characteristics (aggregated by category).

- **Arms:** The set of items (or categories, in our chosen approach) available for recommendation at each time step. Each arm corresponds to a category we can recommend.
- **Rewards:** After recommending a category, we observe the watch ratio (or a related engagement metric) as the reward. Higher watch ratios indicate better alignment with user preferences.

The goal is to select arms (categories) over a sequence of trials to maximize cumulative rewards while learning from evolving user contexts.

3.2 LinUCB Algorithm

To efficiently incorporate contextual information, we employ the LinUCB algorithm. LinUCB assumes that the expected reward of each arm is a linear function of the contextual features:

$$r_a(x) = \theta_a^\top x + \epsilon,$$

where $r_a(x)$ is the reward for arm a given context x, θ_a is a parameter vector associated with arm a, and ϵ is noise. The algorithm maintains an estimate of θ_a for each arm and uses an upper confidence bound strategy to balance exploration and exploitation.

For each arm a, LinUCB maintains a matrix A_a and vector b_a:

$$A_a = I + \sum_{t:a_t=a} x_t x_t^\top, \quad b_a = \sum_{t:a_t=a} r_t x_t,$$

where x_t is the context vector at time t, and r_t is the observed reward. The estimate of θ_a is:

$$\hat{\theta}_a = A_a^{-1} b_a.$$

When selecting an arm at time t, LinUCB computes an upper confidence bound:

$$\mathrm{UCB}_a(x_t) = \hat{\theta}_a^\top x_t + \alpha \sqrt{x_t^\top A_a^{-1} x_t},$$

where α controls the degree of exploration. The algorithm selects the arm with the highest UCB value. Over time, this approach refines its parameter estimates and improves its ability to recommend categories that yield higher rewards.

4 Demonstration of Improved Decision-Making

We demonstrate LinUCB's effectiveness by simulating a sequence of recommendations on the prepared KuaiRec dataset:

1. **Iterative Training:** We iteratively present users from the training set to the bandit model. Each user's context and available arms (categories) are fed into LinUCB. The model selects a category to recommend, observes the resulting reward (watch ratio), and updates its parameters accordingly.

2. **Cumulative Regret Analysis:** We measure the cumulative regret of the recommendations, defined as the difference between the reward of the chosen arm and the best possible arm in hindsight. As the model learns, the cumulative regret curve flattens, indicating that LinUCB is converging toward optimal decisions. Reduced regret over time demonstrates improved decision-making and more effective recommendations.

3. **Testing and Generalization:** After training, we apply the learned model to the test set without further parameter updates. By comparing cumulative regret and average rewards between the training and testing phases, we ensure that the model's performance gains are not limited to the training sample. We observe that LinUCB generalizes well, continuing to provide high-quality recommendations and minimal regret on unseen user data.

The result is a recommendation strategy that adaptively learns user preferences and adjusts its recommendations to maximize engagement, all while efficiently exploring new categories that might interest users.

5 Contributions and Limitations

This study contributes to the literature on sequential recommendation strategies by demonstrating how a contextual bandit model, specifically LinUCB, can be applied to a dense real-world dataset like KuaiRec. By leveraging rich contextual data, the LinUCB approach:

- **Enhances Adaptability:** The model dynamically updates its estimates of user preferences, continually improving the relevance of its recommendations.

- **Balances Exploration and Exploitation:** The upper confidence bound framework encourages the algorithm to explore new categories while leveraging existing knowledge, ultimately reducing regret.

- **Provides a Practical Framework:** LinUCB is computationally efficient and readily interpretable, making it suitable for large-scale deployment on digital platforms.

However, there are several limitations and opportunities for future work:

- **Contextual Complexity:** Although we included user and item features, real-world scenarios may demand even more complex contextual signals (e.g., temporal trends, social network influences, and cross-platform behaviors). Incorporating such complexity might require more sophisticated models, such as neural-based contextual bandits or hybrid systems.

- **Reward Definition:** We used watch ratio as a proxy for user engagement. Future studies could explore other reward metrics, such as retention, downstream conversions, or long-term user satisfaction, to better align the model's objectives with platform goals.
- **Scalability and Real-Time Constraints:** While LinUCB is relatively efficient, extremely large user bases or item catalogs could still pose computational challenges. Developing incremental or approximate update strategies would be beneficial for real-time industrial applications.
- **Robustness to Non-Stationary Environments:** User preferences may shift over time, especially in rapidly changing content landscapes. Incorporating adaptive learning mechanisms or drift-detection methods could help maintain performance over extended periods.

6 Conclusion

In summary, this work illustrates the application of the LinUCB algorithm to the KuaiRec dataset for personalized, contextualized recommendations. The dense and feature-rich KuaiRec dataset enabled a thorough exploration of the algorithm's behavior, including its ability to learn contextual cues, balance exploration and exploitation, and achieve low regret. While the approach has demonstrated potential in improving sequential decision-making in recommendation scenarios, further studies may enhance its adaptability, robustness, and long-term utility in dynamic, real-world environments.