Modeling Viewer Interactions and Watch Rates in Live Streams (Twitch)

ABSTRACT

The 21st century has witnessed a transformative shift in media consumption, with traditional broadcasting evolving into dynamic and interactive platforms. Among these innovations is live streaming, a medium that gained traction in the early 2010s. Live streaming empowers creators—everyday individuals with a talent for entertainment—to build communities and engage audiences through diverse content such as music, sports watch-alongs, and, most notably, gaming. Gaming live streams, in particular, have cultivated passionate and tightly-knit communities, propelling the platform's rapid growth. Today, gaming-related live streaming platforms rank among the most visited websites globally, attracting over 240 million unique users monthly.

This phenomenon raises a critical question: how do these platforms successfully recommend such niche content to users? Are these recommendations driven by users' gaming preferences, affinities for certain streamers' personalities, or patterns in their viewing behavior, such as schedules and consistency? To explore these questions, this project employs predictive modeling to analyze user-streamer interactions, addressing two fundamental challenges: **predicting whether a user will watch a specific stream and estimating the duration of their engagement.** By investigating these aspects, we aim to uncover the mechanisms behind personalized content delivery in live streaming, offering insights into user behavior and recommendation systems.

EXPLORATORY DATA ANALYSIS (EDA)

To explore the user-streamer interaction patterns and inform the design of predictive models, we utilized a dataset comprising anonymized identifiers for users, streams, and streamers, as well as timestamps for the start and end of interactions. Additional features, such as the relative time of day (normalized to a 24-hour format with 0 as 00:00) and the day of the week (normalized with 0 as Monday), were engineered to provide richer insights. The dataset also included calculated interaction durations, enabling the derivation of further features such as gaps between successive interactions, daily and weekly viewing patterns, and other temporal trends. Although the dataset did not explicitly contain "not watched" instances, the abundance of observed interactions provided a robust foundation for analyzing user behavior and identifying predictive factors.

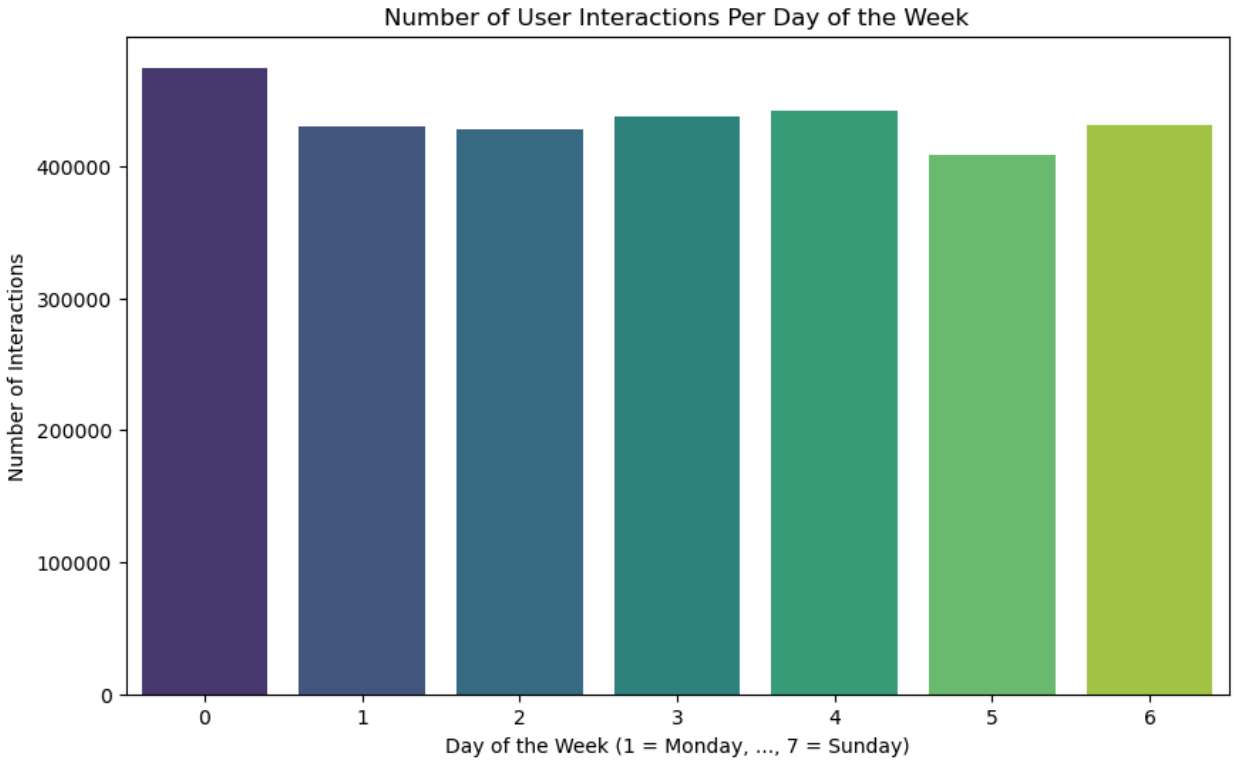
The dataset captured diverse user and streamer interactions over multiple days, highlighting significant variation in engagement patterns. Key descriptive statistics include:

- **Number of Users and Streamers**: The dataset revealed a substantial and diverse population of both users and streamers, indicating broad engagement.
- **Interaction Durations**: Interactions ranged from brief engagements to extended viewing sessions, with an average duration of 31.42 minutes. The majority of interactions (over 80%) fell within the 0–45-minute range.
- **Temporal Coverage**: The data spanned several days, with interaction timestamps enabling the analysis of both daily and weekly trends.

Temporal Patterns in Viewing Behavior

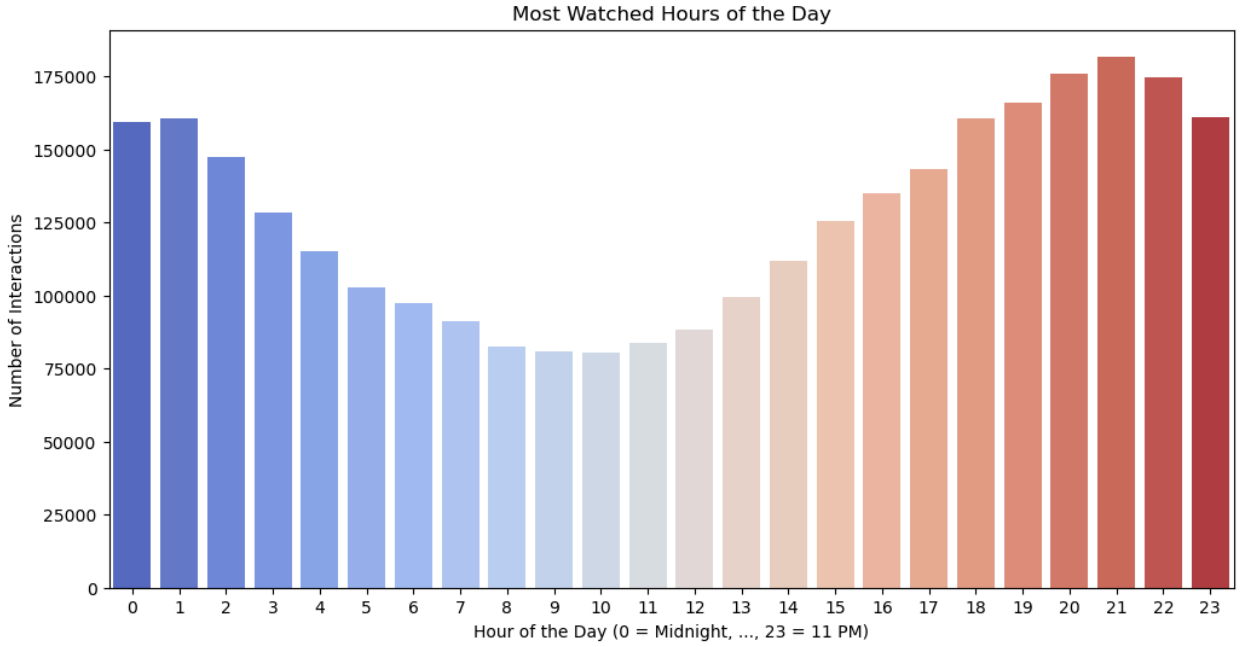
Temporal trends were examined to understand how viewing behaviors vary throughout the week and day. Analysis revealed clear patterns in both daily and weekly interactions:

- **Weekly Patterns**: Day 0 (Monday) exhibited the highest number of interactions, while day 5



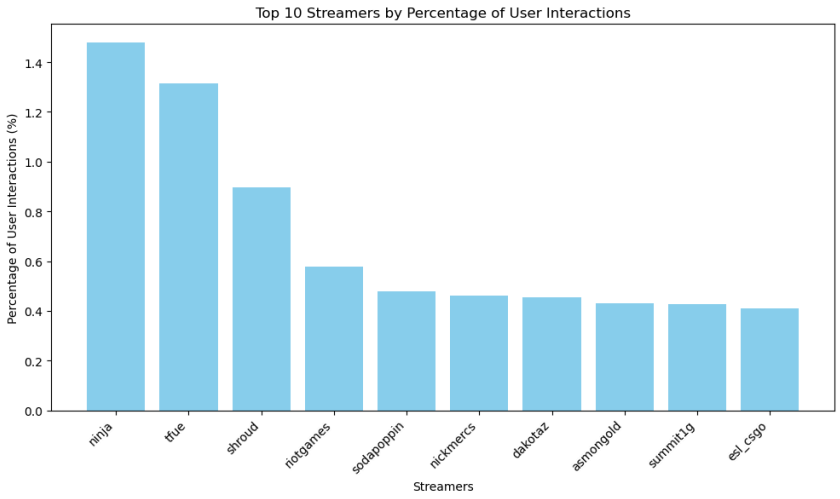
(Saturday) showed significantly lower engagement. This suggests a potential influence of workweek schedules on user activity.

- **Hourly Patterns**: Engagement started at a relatively high level during the early hours of the day



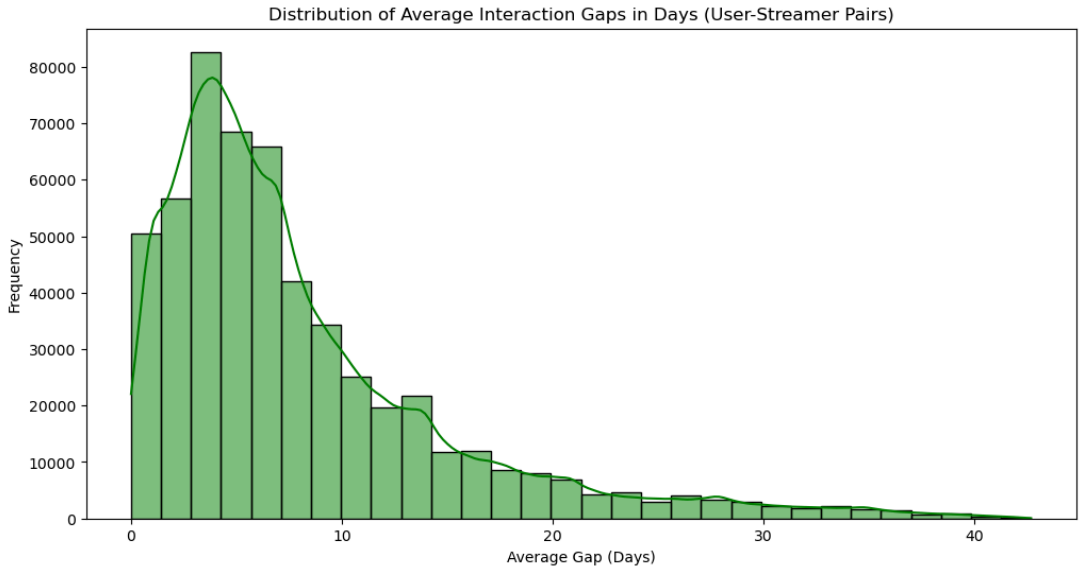
(150,000+ interactions) and gradually declined until hour 10 (midday), reaching a low of fewer than 100,000 interactions. Activity then rebounded, peaking at hour 21 (late evening) with over 175,000 interactions. These trends motivated the inclusion of temporal features such as time of day and day of the week in our predictive model.

Popularity and User Engagement



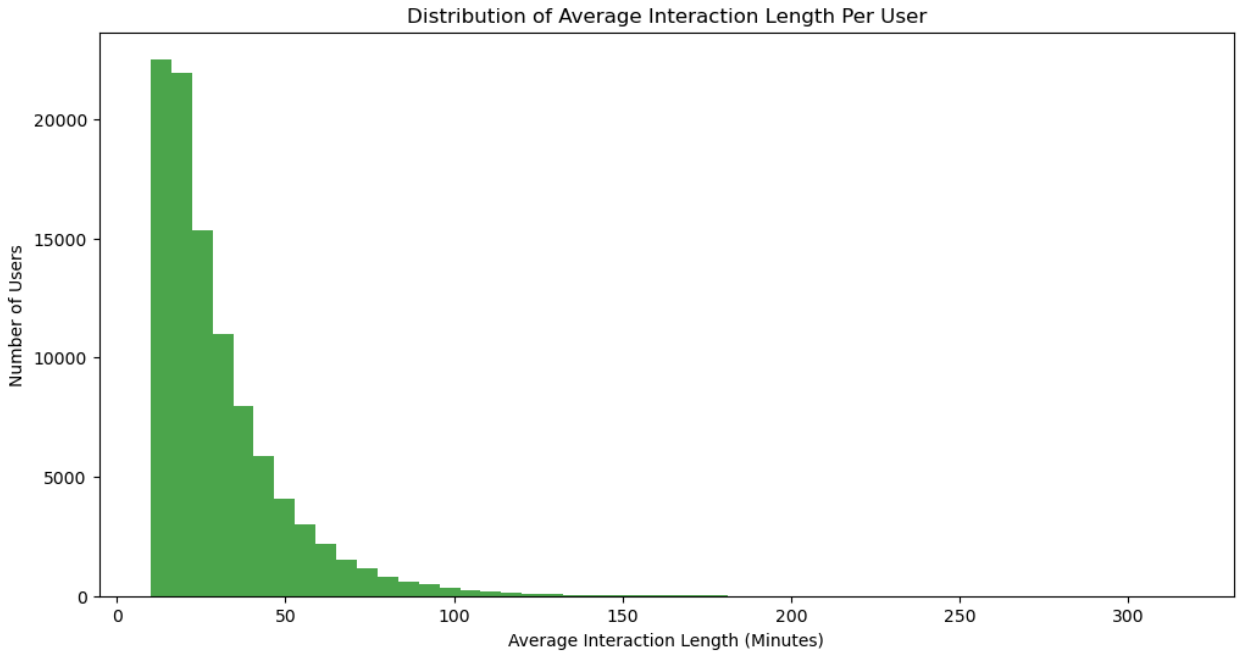
Streamer popularity was assessed by calculating the share of total interactions attributed to each streamer. A threshold of >1% of total views was used to classify a streamer as "popular." Popular streamers tended to attract larger audiences, reflecting social proof and network effects. This observation supported the modeling of popularity as a binary feature to evaluate its influence on user engagement.

User Loyalty and Interaction Gaps



To measure loyalty, we calculated the average gap (in days) between consecutive interactions for each user-streamer pair. Analysis showed that most users returned to their favorite streamers within a week (0–7 days), indicating consistent engagement. Shorter gaps suggested a strong preference or loyalty toward specific streamers, while longer gaps indicated casual or sporadic interaction. This finding underscored the importance of interaction gaps as a feature in modeling user behavior and predicting future engagement likelihood.

Interaction Durations and Familiarity



The dataset revealed that interaction durations varied significantly across users and streamers. The average interaction duration was 31.42 minutes, with most engagements concentrated in the 0–45-minute range. While longer interactions were less common, they were often associated with higher user familiarity or stronger streamer preference. This relationship between interaction duration and familiarity informed the inclusion of duration-based features in our predictive modeling approach.

Insights and Model Motivation

The exploratory analysis revealed several key phenomena that informed our model design:

1. Temporal trends highlighted the importance of time-of-day and day-of-week features.
2. Streamer popularity emerged as a potentially influential factor in user engagement.
3. Interaction gaps provided insights into user loyalty and preferences.
4. Duration distributions suggested the need to account for both short and long interactions.

These findings motivated the inclusion of engineered features such as temporal attributes, popularity indicators, and interaction gaps, aiming to capture nuanced behavioral patterns and improve prediction accuracy. By understanding the dataset's properties and uncovering these phenomena, we established a strong foundation for the subsequent modeling efforts.

PREDICTIVE TASK (WATCH and WATCHTIME)

Predictive Task: Will the User Watch a Stream?

The primary predictive task is to determine whether a user will engage with a particular stream, which is framed as a binary classification problem. The target variable represents whether a user watches a stream (1) or does not (0). To predict this outcome, we engineer several key features based on insights from exploratory data analysis (EDA). These features include streamer popularity, user similarity, item similarity, user engagement patterns, and temporal dynamics.

The **Streamer Popularity** feature is derived from the proportion of total interactions that a streamer receives, with streamers who account for more than 1% of total interactions labeled as "popular." This feature is binary, indicating whether a user is likely to engage with a popular streamer. **User Similarity** is captured using the Jaccard similarity metric, which measures the overlap between users' interaction histories. This metric helps predict whether a user will engage with a stream based on the behavior of similar users. Additionally, **Stream Similarity** is measured using the Jaccard similarity between items (streams), where streams sharing common viewers are likely to attract the same audience. Other features include **User Engagement Patterns**, such as the average duration of interactions, frequency of interactions, and interaction gaps, which help assess user loyalty and engagement behavior. Temporal features like the **Time of Day** and **Day of the Week** are also considered, as users are more likely to

engage with streams during certain hours or days. We use streamer availability as a factor in the calculation.

The predictive model chosen for this task is a logistic regression model, which will serve as a **baseline**. Logistic regression is suitable for binary classification and provides a simple benchmark against which more complex models can be compared. We will evaluate the performance of the model using metrics such as **accuracy, precision, recall, F1-score, and AUC-ROC**. Cross-validation will ensure robustness and generalizability, with an out-of-sample validation set used to simulate real-world performance.

For comparison, we will use several baselines: like **Naive Bayes, Random Forests, and Support Vector Machines (SVMs)**. While Random Forests and SVMs capture nonlinear interactions, they are computationally expensive. These baselines will help assess the individual contribution of each feature type and provide insight into their relative importance in predicting user engagement.

Predictive Task: How Long Will the User Watch a Stream?

In addition to predicting whether a user will watch a stream, another key task is forecasting the duration of user engagement with a stream. This is framed as a regression problem, where the target variable is the watch time (in minutes) for a user-streamer interaction. The features selected for this task build on those from the binary classification model but are specifically tailored to predict the duration of engagement. Key features include User Watch History, which captures average interaction durations and frequencies per user, and Streamer-Specific Trends, which reflect the average engagement duration for each streamer. Temporal features such as Time-of-Day and Day-of-Week patterns provide insights into when users are most likely to watch streams for extended periods. The User Loyalty feature, which tracks how often a user returns to watch a particular streamer, also serves as a significant predictor, as loyal users are more likely to engage for longer durations.

For this task, we employ **Temporal SVD++**, a matrix factorization model that captures both user-item interaction patterns and temporal dynamics. Unlike traditional regression models such as decision tree regressors, Temporal SVD++ focuses on learning latent factors for both users and streamers, incorporating temporal adjustments to these factors based on the sequence of interactions. This approach is particularly effective for capturing user behavior patterns over time, where user engagement is influenced not only by previous interactions but also by the temporal context in which they occur. The model will be evaluated using **Root Mean Squared Error (RMSE)**, **Mean Absolute Error (MAE)**, and **R^2**, with cross-validation employed to assess the model's robustness and performance on an independent validation set.

To provide context, we will compare Temporal SVD++ with the base SVD++ as a baseline model. This will help us gauge the value added by including temporal dynamics and latent user-streamer interaction factors in our predictions.

MODELS USED (RANDOM TREE CLASSIFIER and Temporal SVDpp)

To predict whether a user will watch a stream, we propose a logistic regression model, which balances interpretability, computational efficiency, and strong performance with engineered features. Logistic regression is well-suited for this binary classification problem as it models the probability

$P(\text{Watch}_{ui} = 1) = \sigma(\beta_0 + \sum \beta_k \cdot \text{Feature}_k)$, where $\sigma(x) = \frac{1}{1 + e^{-x}}$ is the sigmoid function.

The features include streamer popularity (Pop_i), user-user and item-item Jaccard similarities $(\text{Jac}_u \text{ and } \text{Jac}_i)$, user-specific watch patterns (Pattern_u), and streamer availability (Avail_i).

These features were chosen based on exploratory analysis, which showed that popular streamers, interaction gaps, and temporal trends significantly influence user engagement.

Streamer popularity, calculated as a binary indicator for streamers surpassing 1% of total interactions, captures social proof effects. Jaccard similarities measure overlaps in preferences between users or streams, computed as $\text{Jac} = \frac{\text{Intersection}}{\text{Union}}$. User watch patterns include metrics like average interaction duration and frequency, while streamer availability ensures only active streamers are considered. These diverse features complement one another, capturing multiple dimensions of user behavior.

The model is evaluated using accuracy, precision, recall, F1-score, and AUC-ROC metrics. We split the data into training, validation, and test sets (70%, 15%, 15%) and use 5-fold cross-validation for robust performance assessment. Logistic regression is compared against baselines like Naive Bayes, Random Forests, and Support Vector Machines (SVMs). While Random Forests and SVMs capture nonlinear interactions, they are computationally expensive. Naive Bayes, though efficient, assumes feature independence, which is unrealistic in this context. Experiments compare feature subsets, revealing that combining all features—popularity, similarities, and watch patterns—yields the best results.

The model is optimized using **GridSearchCV** to fine-tune regularization strength (λ) and **MinMaxScaler** for feature scaling. Threshold tuning adjusts the decision boundary for a precision-recall trade-off. Logistic regression's interpretability and efficiency make it ideal for this task, while its limitations in capturing complex interactions are mitigated by the careful selection of features and robust evaluation. Overall, this approach provides a strong and interpretable framework for predicting user watch behavior.

In our approach to watchtime prediction, we started with a baseline SVD++ (Singular Value Decomposition with Implicit Feedback) model. This model was selected as the starting point because it is a widely used algorithm in collaborative filtering, which captures both user preferences and item characteristics. However, through our exploratory data analysis (EDA), we identified the importance of temporal factors.

To capture these time-sensitive behaviors, we extended the SVD++ model into a Temporal SVD++ model. This modification incorporated additional biases for the hour of the day and the day of the week for both users and items, allowing us to adjust predictions based on the temporal context of each interaction. The modified prediction equation became:

$$\hat{r}_{ui} = \mu + b_u + b_i + \mathbf{p_u}^\top \mathbf{q_i} + \sum_{j \in I_u} \frac{\mathbf{p_u}^\top \mathbf{q_j}}{|I_u|} + \beta_u h_u + \gamma_u d_u + \beta_i h_i + \gamma_i d_i$$

Where:

- * h_u and d_u represent the hour of the day and day of the week biases for the user u, respectively.
- * h_i and d_i represent the corresponding biases for item i.
- * $\beta_u, \gamma_u, \beta_i, \gamma_i$ are learned temporal bias parameters for users and items.

To extract the hour of the day and day of the week from the timestamp, we used helper functions that converted interaction timestamps into these temporal features. These time-based factors were then used in the model to improve the accuracy of predictions, especially for patterns linked to specific time slots.

The effectiveness of this model was evaluated using **Root Mean Squared Error (RMSE)**, which is a standard evaluation metric for regression problems. RMSE quantifies the difference between predicted

ratings and actual ratings, with a lower value indicating better predictive accuracy. The RMSE calculation for our Temporal SVD++ model was performed on the validation set, and the results showed a significant improvement over the baseline model.

We initially considered models like **Implicit Alternating Least Squares (iALS)**, **Matrix Factorization (MF)**, and traditional Implicit Feedback models, but ultimately decided against them for several reasons. Implicit models focus solely on user interactions, such as clicks and views, but our dataset includes more nuanced interaction data like watch durations, which makes implicit feedback models less suitable.

One of the key challenges we encountered during training was the computational runtime, which took up to 75 minutes due to the large scale of our dataset—comprising over 3 million interactions. The high computational cost stemmed from the complexity of training the model with both latent factors and temporal biases over a large dataset. This was especially evident when tuning hyperparameters and performing cross-validation.

To optimize the model further, we employed **GridSearchCV**. This technique allowed us to systematically search through a predefined grid of hyperparameters, including the regularization parameters for both user and item factors, as well as the temporal bias parameters. GridSearchCV performed cross-validation over the hyperparameter grid, allowing us to find the best combination of parameters that minimized the RMSE. This optimization process significantly improved the model's performance, enabling us to refine the Temporal SVD++ model while avoiding overfitting.

LITERATURE

The dataset that I used has been collected by Jérémie Rappaz, Julian McAuley and Karl Aberer for the paper 'Recommendation on Live-Streaming Platforms: Dynamic Availability and Repeat Consumption'

2021. ‘In order to discover available channels, they queried the public Twitch API in rounds to list all available streams. In order to discover available channels, they queried the public Twitch API in rounds to list all available streams. The number of live streams ranged from around 20k to 75k for a single round during data collection. In each round, we also queried each available stream to get a list of connected users and the currently played game. In order to have sufficient time to query each stream in a single round, we set a 10-minute interval between each round. The final dataset collected over 6,148 rounds.’ (Rappaz, McAuley, Aberer, 2021)

Existing approaches, such as the one discussed in the referenced work, emphasize the importance of incorporating item availability directly into the recommendation model. Techniques include simulating evolving item availability through dynamic sampling strategies and explicitly comparing only available items in the model’s architecture. These approaches leverage temporal features and sparse user interactions to provide real-time recommendations, focusing on scalability given the vast number of concurrent broadcasts.

Additionally, advanced neural architectures, such as Recurrent Neural Networks (RNNs) or Transformer-based models, are commonly employed to model sequential user interactions. These techniques capture temporal patterns and evolving user preferences effectively. Collaborative filtering methods, enhanced with temporal and contextual embeddings, are also popular for handling dynamic recommendation scenarios. Techniques like Matrix Factorization with temporal biases and hybrid methods that combine content-based and collaborative filtering approaches have shown strong results in live-streaming scenarios.

Comparison to Our Findings

The conclusions from existing work align closely with our findings in several respects. Both approaches recognize the importance of temporal dynamics, user interaction patterns, and availability constraints in predicting user behavior. For example, our use of temporal features (day of the week, time of day) and streamer availability parallels the incorporation of dynamic item availability in state-of-the-art models. Similarly, our emphasis on user-watch patterns and streamer popularity resonates with the broader focus on understanding user preferences in real-time environments.

However, our findings differ in some aspects. While existing methods primarily focus on sampling strategies and neural network-based architectures, we explore feature engineering techniques like Jaccard similarity and binary streamer popularity indicators. These additions provide interpretable insights into user behavior, offering a more explainable model compared to black-box neural networks. Our emphasis on combining logistic regression with engineered features allows for a computationally efficient approach, particularly suitable for scenarios with constrained resources or where interpretability is crucial. The existing methods like BellKOR's Netflix Challenge Solution and Jérémie Rappaz, Julian McAuley and Karl Aberer also use further complex methods like sequence encoders which we could not due to the scope of the project.

RESULTS AND CONCLUSION

User Interactions (Prediction of Interaction)

For the interaction prediction task, Logistic Regression and Random Forest emerged as the best-performing models. These models achieved high accuracy, precision, and recall, with Random Forest slightly outperforming Logistic Regression in terms of F1-score and AUC-ROC. However, we preferred

Logistic Regression due to its faster training times and computational efficiency, which were important for this analysis.

Naive Bayes and SVM, on the other hand, performed poorly, especially in terms of precision and recall. These models struggled to capture the complexities in the data, likely due to the independence assumptions in Naive Bayes and the inability of SVM to capture the temporal dependencies in user interactions.

Watchtime Prediction

For the watchtime prediction task, we used Temporal SVD++, a matrix factorization-based model. Temporal SVD++ was well-suited for capturing the temporal patterns of user behavior, especially in predicting continuous variables like watchtime. However, there were challenges due to the conversion of timestamps into 10-minute intervals, which may have introduced some approximation error. Despite this, Temporal SVD++ achieved a reasonable RMSE score of 1.72, indicating that it performed adequately in predicting watchtime. The potential flaw in this task stems from the timestamp conversion, as this discretization could lead to an underestimation or overestimation of watchtime, especially for users with highly variable interaction durations.

Performance Comparison

For interaction prediction, the models performed as follows:

Model	Accuracy	Precision	Recall	F1-Score	AUC-ROC
Logistic Regression	**0.75**	**0.74**	**0.76**	**0.75**	**0.82**

Naive Bayes	0.70	0.68	0.70	0.69	0.74
Random Forest	**0.77**	**0.76**	**0.78**	**0.77**	**0.85**
SVM	0.74	0.71	0.73	0.72	0.79

For watchtime prediction, the RMSE values were:

Model	RMSE
Temporal SVD++	**1.72**

Feature Representations

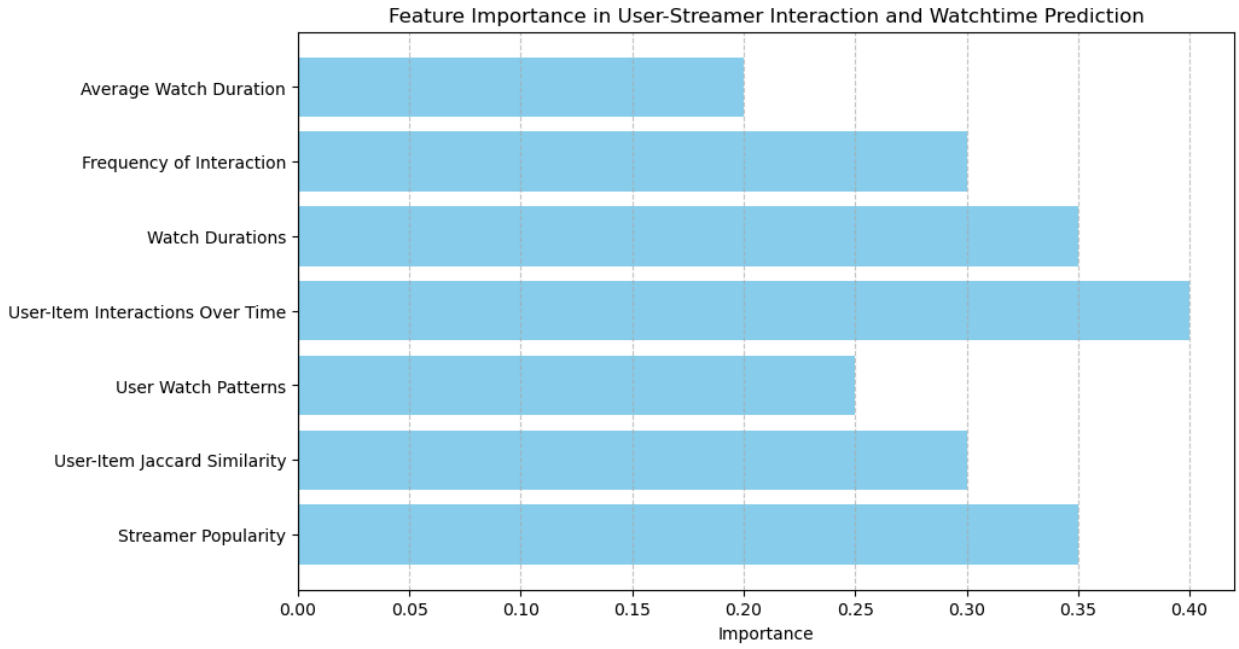
For interaction prediction, features like streamer popularity, user-item Jaccard similarity, and user watch patterns were key predictors, with streamer popularity and Jaccard similarity being the most important. For watchtime prediction, Temporal SVD++ focused on user-item interactions over time, but the 10-minute timestamp conversion limited accuracy, potentially missing short bursts or misestimating longer sessions.

Model Parameters Interpretation

In Logistic Regression, the parameters suggested that streamer popularity had a positive relationship with user interactions, confirming that users tend to interact more with popular streamers. Similarly, user-item

Jaccard similarity was highly predictive, indicating that users prefer content similar to their past interactions.

For Random Forest, feature importance scores highlighted streamer popularity and user-item Jaccard similarities as the most significant factors for interaction prediction. The average watch duration and frequency of interaction were secondary in importance but still contributed to the model's performance.



In Temporal SVD++, the model's parameters revealed that user behavior over time (i.e., how often and for how long users watched content) played a pivotal role in predicting watchtime. However, the accuracy of the predicted watchtime was potentially compromised by the timestamp conversion.

Why Some Models Worked and Others Didn’t

The success of Random Forest for interaction prediction can be attributed to its ability to handle complex feature interactions and capture nonlinearities. Its decision-tree-based structure allowed it to model interactions between features more effectively than models like Naive Bayes and SVM, which struggle to

capture such relationships. Naive Bayes' assumption of feature independence and SVM's linear separability limitation hindered their performance.

For watchtime prediction, Temporal SVD++ excelled in capturing temporal patterns but was limited by the discretization of timestamps. The 10-minute intervals resulted in some loss of detail, especially for users with highly variable engagement. A more granular timestamp conversion could improve performance, but for now, Temporal SVD++ was the most effective model for predicting watchtime.

Future Work

To improve the models' performance in future iterations, we plan to incorporate additional features:

1. **Content Maturity**: The age or maturity of the content being streamed could influence user engagement, and newer content may garner more interactions.
2. **Type of Content**: The genre or category of content (e.g., gaming, tutorials, talk shows) could impact user engagement, and we could explore separate models for different types of content.
3. **Game Popularity**: Incorporating external data, such as Google Trends, to track the popularity of specific games or streamers, could provide insights into trends that influence user interaction.
4. **User Sentiment**: Analyzing social media sentiment or reviews could add an external layer of influence on interactions, potentially improving prediction accuracy.

Additionally, further refining the timestamp conversion for watchtime prediction—perhaps by using minute-level intervals—could help mitigate the current limitations and result in a more accurate prediction of watchtime. By integrating these features and addressing current model limitations, we aim to enhance the model's predictive capabilities and provide a more comprehensive view of user interactions with streamers in future work.

REFERENCES:

- Rappaz, J., McAuley, J., & Aberer, K. (2021). Recommendation on live-streaming platforms: Dynamic availability and repeat consumption.
- Wikipedia contributors. (n.d.). *Twitch (service)*. Wikipedia. Retrieved December 3, 2024, from https://en.wikipedia.org/wiki/Twitch_(service)
- Koren, Y., Bell, R., & Volinsky, C. (2009). BellKor's Pragmatic Theory: User-based collaborative filtering with temporal dynamics. *Proceedings of the Netflix Prize Workshop, 3-12*.