CXC Federato

Joey Shum
Nam Ho
Quang Bui
Tung Nguyen

February 24th, 2025

Contents

1. Executive Summary
2. Initial Data Preprocessing
3. Initial EDA
4. Data Processing
5. EDA On Retention
6. EDA On Session Time
7. Modeling
8. Recommendations

Summary

In this report, we analyze and develop a recommendation system based on a given dataset. Our approach begins with an in-depth exploration of the original dataset to understand its structure, key features, and potential insights. Based on our findings, we decided to process and aggregate the data into 2 distinct datasets, each designed to build a specialized model.

The first model aims to capture user behavior patterns and predict the likelihood of a user engaging with a given event. The second model focuses on user retention, identifying factors that contribute to sustained engagement over time. The third model is designed to estimate the time users spend on the platform, providing valuable insights into content consumption and engagement duration.

By integrating these three models, we construct an ensemble recommendation system that optimally suggests events. This system is designed to maximize user retention, increase time spent on the platform, and enhance the probability of users selecting recommended events, thereby improving overall user engagement and satisfaction.

Initial Data Processing

Handling event_properties and user_properties

The original format of these two columns (dictionary/JSON) is difficult to process and extract meaningful information from, so we expand them into approximately 100 separate columns. However, since most of these columns are either empty or contain NaN values, we filter out those with more than 50% NaN values. Additionally, we remove columns that are overly dominated by a single category. As a result, we retain only the following columns: slug from event_properties and roles, isInternalUser, referrer from user_properties

Handling event_type

Since the event_type column contains approximately 600 unique categories, analyzing and visualizing it in its raw form would be overwhelming and difficult to interpret. To create more meaningful insights and improve visualization clarity, we have decided to group these categories into broader, more relevant groups based on repeated keywords. This approach helps reduce noise, enhances pattern recognition, and allows for easier comparison across different event types while preserving the overall structure and significance of the data.

Category	event_type keywords
‘Sessions & Navigation’	session_start, session_end, application-window, nav-header, dashboard
‘Account/Policy Mgmt’	account, policy, rating
‘Dashboard & UI’	dashboard, widget, layout, insights, table

‘Action Center’	action-center, task, workflow, take-action
‘Submissions & Forms’	submit-click, form, create, definition, save-click, submissions
‘Filter & Searching’	filter, sort, search, advanced-filters
‘Document & Report’	document, report, download, csv
‘Other’/ System Events	EMPTY, help-page, user-signed-out, everything else

Handling slug

Original Slug Pattern	Mapped Slug
Numeric values	numeric
Starts with "account"	account
Starts with "email-inbox"	email-inbox
Starts with "excess" or "Excess"	excess
Contains "access-control"	access-control
Contains "agency"	agency
Contains "auto"	auto
Contains "aviation" or "fleet"	aviation
Contains "bond"	bond
Contains "classification"	classification
Contains "compliance"	compliance
Contains "dashboard"	dashboard
Contains "financial"	financial
Contains "flood"	flood
Contains "general" or "gen-upsert"	general
Contains "insurance"	insurance
Contains "policy" or "policies"	policy
Contains "property"	property
Contains "submission"	submission
Contains "user"	user
Contains "vehicle"	vehicle
Contains "workers-comp"	workers-comp
Contains "casualty"	specialty-casualty
Contains "rating" or "commodity"	rating-commodity
Contains "fmcsa" or "mvr"	auto
Contains "guidelines" or "document"	documentation

Contains "aviation", "cyber", "terrorism", "marine"	specialty

Due to the column's high cardinality, we have decided to use leave-one-out target encoding. This method encodes each category as the mean target value of all other rows with the same category, excluding the current row. This approach reduces dimensionality compared to one-hot encoding and prevents data leakage, helping the model learn meaningful patterns without overfitting.

Why do we want to overfit the model initially?
Overfitting the training data initially can help us understand the capacity of our model, revealing its upper limits and potential performance on the training set without regularization or other constraints. More specifically, it can help us with the following things:

1. Baseline: Establish a baseline performance for the model.
2. Model Complexity Check: Allowing the model to overfit helps confirm that it has sufficient capacity to capture patterns in the data. If it's unable to overfit the training data, the model may be too simple, or the limitation could stem from preprocessing or data quality issues.

We will attempt to overfit the model by adding more features through feature engineering, guided by our intuition and business insights.

Engineering
Time to Server (time_to_server): Calculated as the difference between client_upload_time and client_event_time, this feature measures the delay between when an event occurs on the client-side and when it is uploaded to the server.

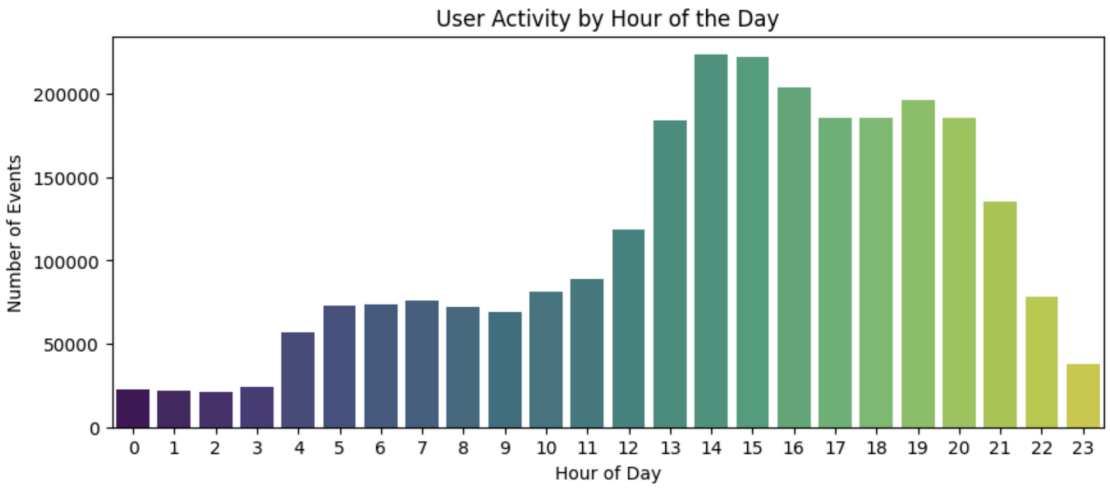
Server to Process (server_to_process): Derived by subtracting server_received_time from server_upload_time, this captures the time taken for the server to acknowledge and begin processing the uploaded data.

Processing Time (processing_time): Computed as the difference between processed_time and server_upload_time, this feature represents the time required for the server to fully process an event.

Initial EDA (EDA on full dataset)

The dataset captures event-based user interactions within the Federato RiskOps platform. The goal of the EDA is to uncover user behavior patterns that influence retention and engagement over a 7-day period.

Event Type Analysis



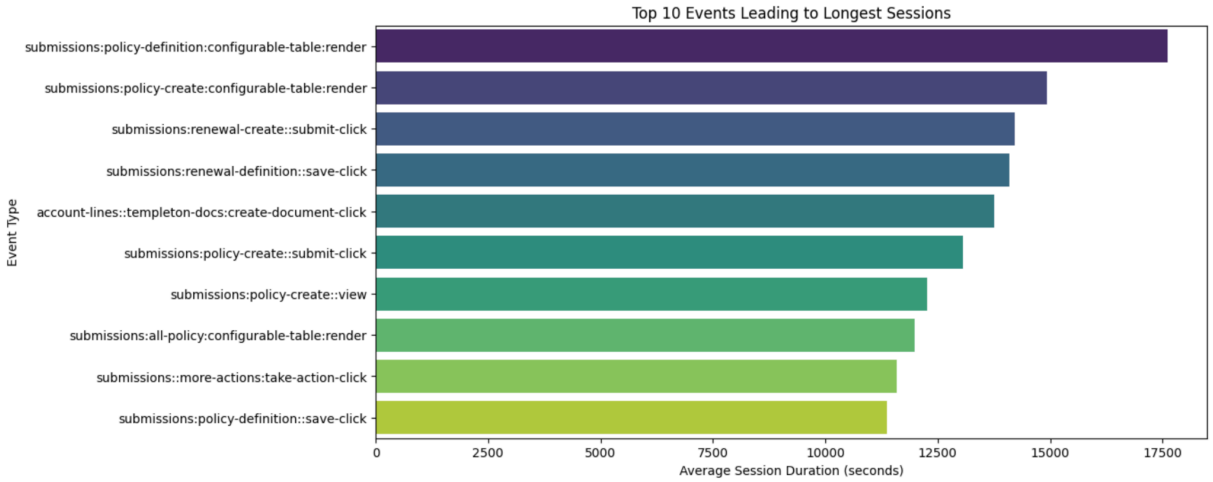
User Activity by Hour of the Day

This visualization presents user activity distribution across different hours of the day. The bar chart indicates that user engagement is relatively low during early morning hours (midnight to 6 AM) but steadily increases as the day progresses. The peak activity period is observed between 1 PM and 4 PM, suggesting that users are most active on the platform during the afternoon hours. After 4 PM, the events gradually decline, with a noticeable drop after 9 PM.

This pattern suggests that the application experiences the highest engagement during working hours, which intuitively makes sense given the nature of the platform. With that said, a secondary peak in the late evening could indicate users revisiting the platform post-work hours.

Implications:

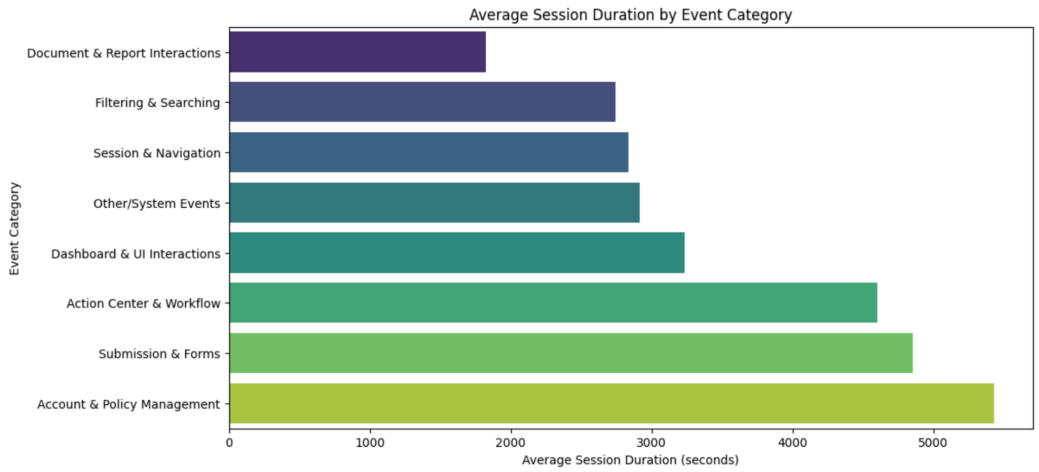
- System performance optimizations should focus on peak usage hours (2 PM – 7 PM).
- Engagement strategies could be more effective if triggered before peak hours.
- A/B testing for new features within the platform should ideally be conducted during peak times to ensure higher participation.



Events Leading to Longest Sessions

Based on the chart above, we can see that **the longest user sessions are strongly associated with form submission events**. This trend suggests that users engage deeply when completing underwriting tasks, likely due to the complexity of decision-making processes and the need to enter detailed information. Unlike high-frequency events that revolve around viewing or navigating, form submission represents a more **intentional** and **high-commitment** action, where users are actively progressing toward a business outcome.

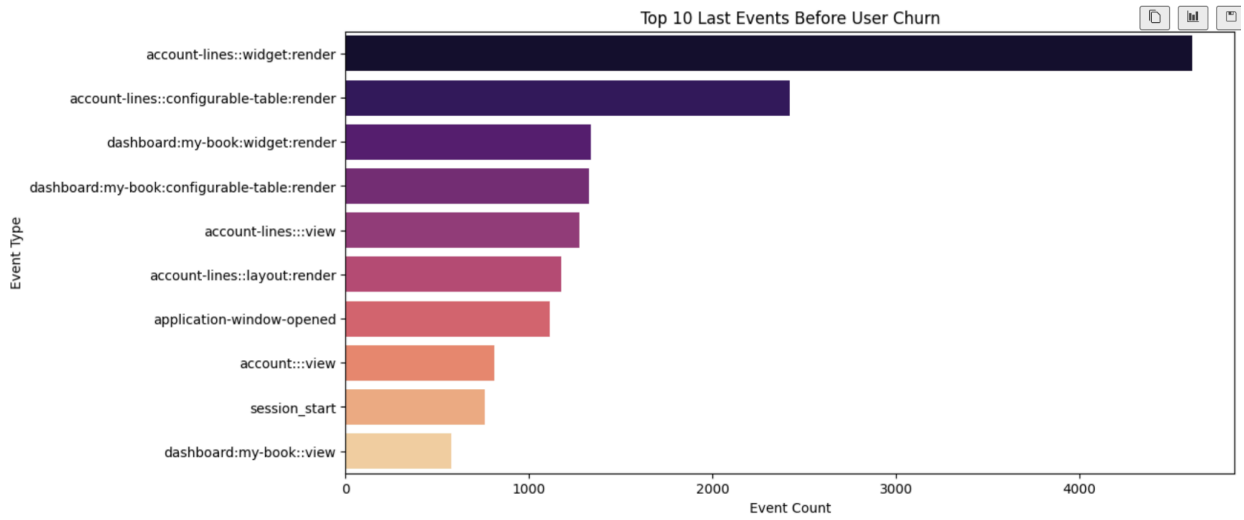
There are several possible explanations for this extended engagement. First, **the cognitive load required for underwriting tasks** may necessitate careful evaluation of multiple data points before finalizing submissions. Users might spend additional time cross-referencing policies, adjusting risk factors, or ensuring that entered information is accurate. Second, lengthy form submissions may indicate **potential usability challenges**, where users take longer than necessary to complete tasks due to **unclear field requirements, complex dependencies, or excessive manual inputs**.



Average Session Duration by Event Category

The chart illustrates the average session duration categorized by event type. Categories such as "Account & Policy Management" and "Submission & Forms" exhibit the highest average session durations, implying that users spend significant time on administrative or form-filling tasks. Conversely, interactions related to "Document & Report Interactions" and "Filtering & Searching" have shorter session durations, suggesting that users can simply quickly access the required information and move on.

These insights are crucial for identifying pain points or areas that may need UX optimization. Long session durations in form submissions, for example, might indicate cumbersome processes that could benefit from streamlining.



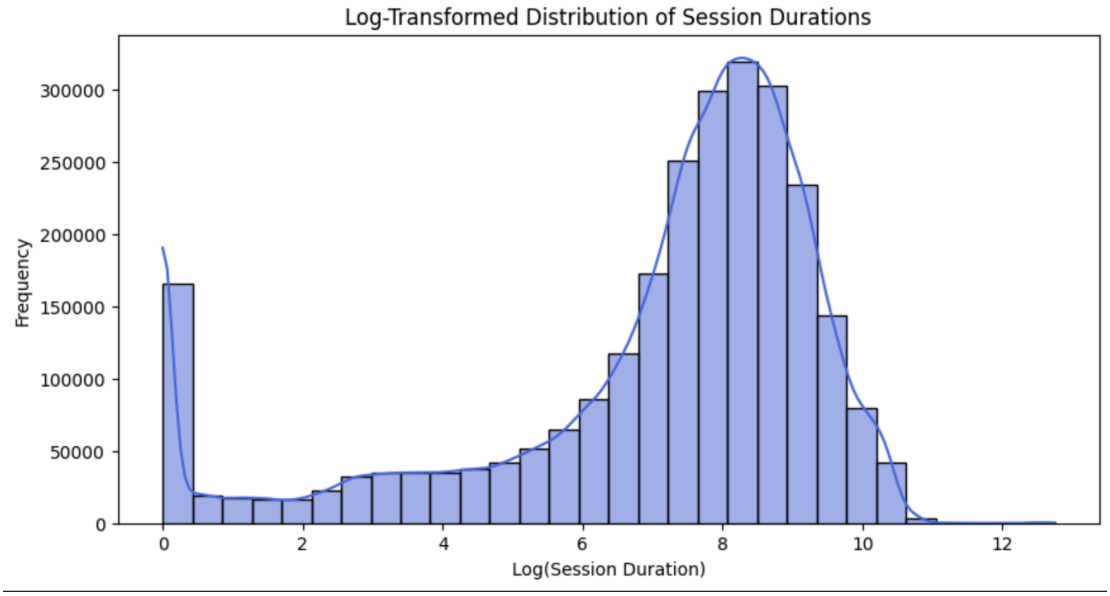
Top 10 Last Events Before User Churn

The following chart highlights the most common last events before user churn. The most frequent event preceding churn is "account-lines::widget:render," followed by similar interactions related to account-line features and dashboard navigation. These events suggest that users might be disengaging after specific interactions, possibly due to frustration, unclear workflows, or unmet expectations.

Analyzing these final interactions can inform targeted interventions, such as improving UI elements, enhancing feature explanations, or triggering retention-focused messaging at critical junctures. Understanding user drop-off points is fundamental for reducing churn and increasing long-term engagement.

Implications:

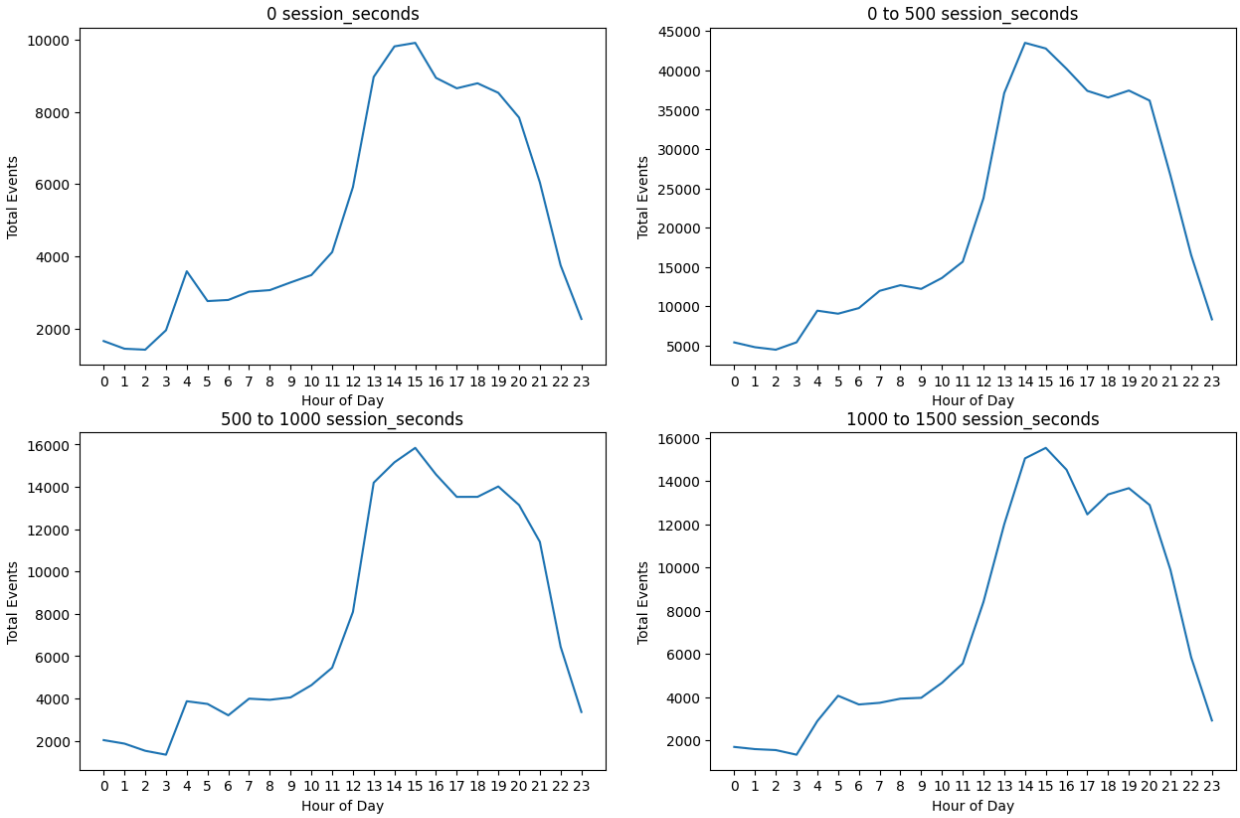
- Users who frequently engage with dashboards but do not proceed to underwriting actions may need **guidance on the next steps**.
- Introducing **personalized recommendations or tooltips** could encourage users to take meaningful actions instead of passively navigating.



Log-Transformed Distribution of Session Durations

The graph illustrates the distribution of how long an user spent on a session using a log transformation. The histogram demonstrates a right-skewed distribution, indicating that a significant number of users have longer session durations, while a smaller proportion engages for short periods, potentially just to check or verify their actions.

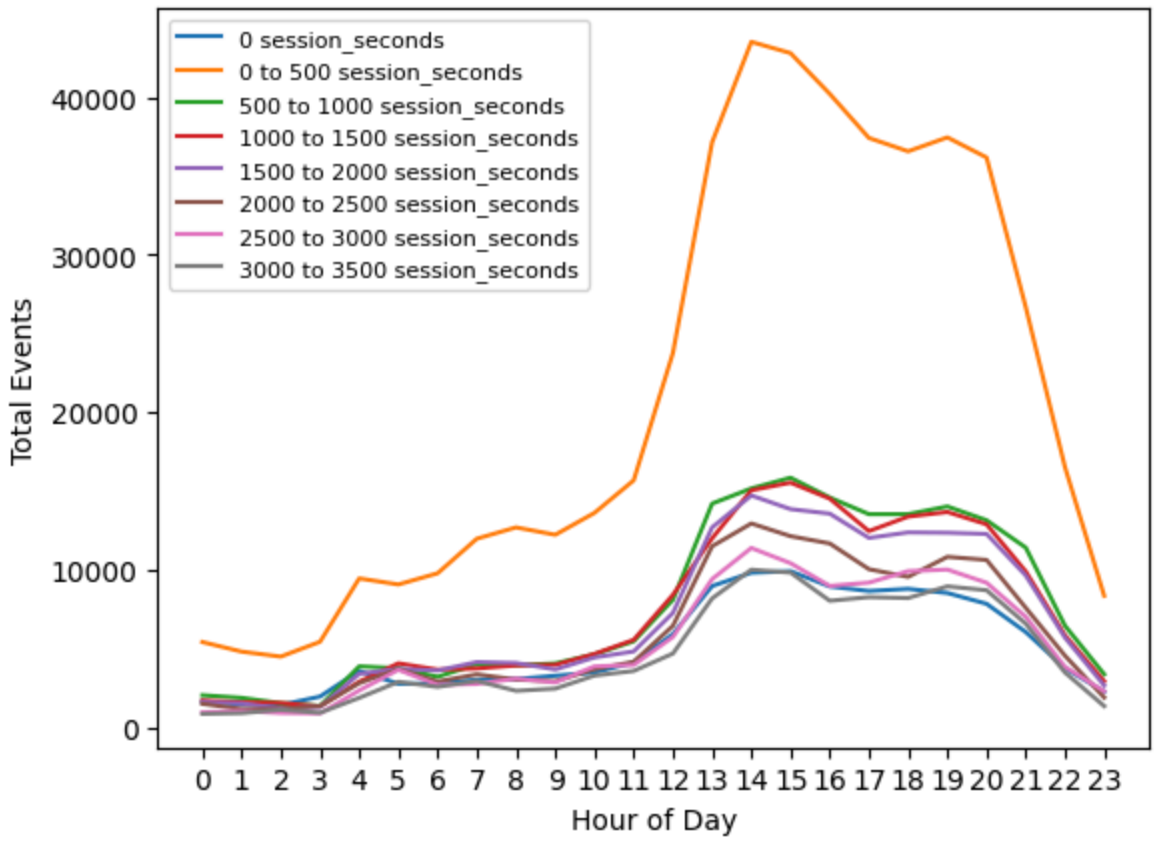
A sharp peak is observed around a log-transformed value of ~8, suggesting a modal session duration within that range. The presence of long-tail values implies that some users have extended engagement, which may correlate with higher retention or deeper interactions within the platform. The log transformation effectively normalizes the data, making it easier to interpret patterns and outliers in session behavior.



Total Events by Hour of Day, Segmented by Session Duration

By segmenting the number of events into different groups of session duration, we aim to identify the differences between sessions. Overall, we see that the shape for all graphs are fairly similar. Small differences include a spike at 4am when session_seconds is 0. Having session_seconds = 0 implies that there is only one action in that section. There is more analysis later on the event types that have a session duration of 0.

Another difference is that the number of total events have a bigger difference from 17h to 19h in the bottom right graph, while other graphs have a flatter curve at those times. In the following graph, we dive deeper into the differences by overlaying the curves on top of each other.

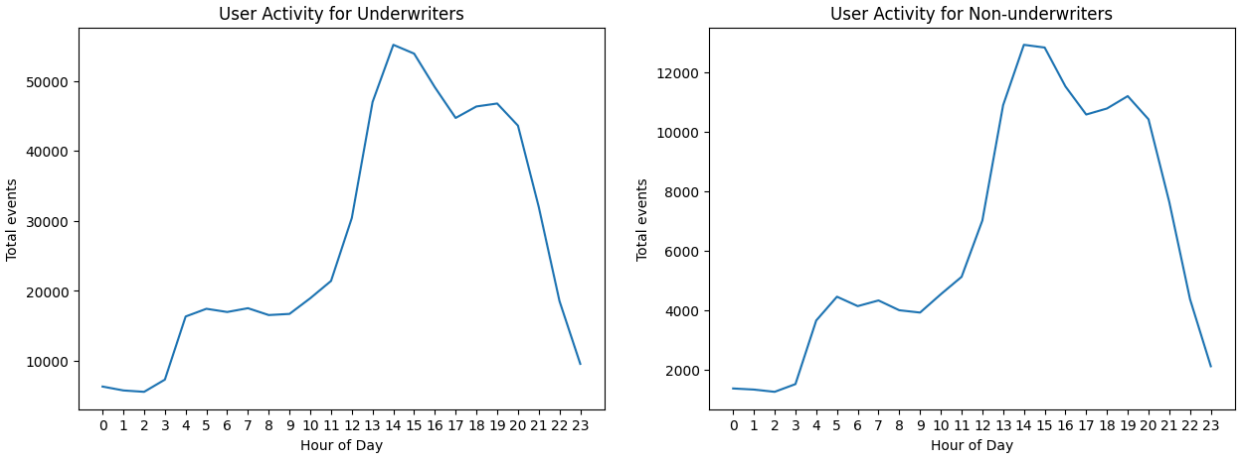


We create buckets of size 500 and plot the curves all on the same graph. This visualization indicates that the total events in small session durations (0 to 500 seconds) have high engagement. Although all curves have a similar shape, this curve sets apart from the others.

event_type	
session_start	26572
session_end	26246
application-window-opened	17115
dashboard:my-book::view	11650
account-lines:::view	5397
account:::view	5240
dashboard:my-book:widget:render	4280
account-lines::widget:render	3515
dashboard:my-book:configurable-table:render	3436
:all-accounts::view	1874
account-lines::configurable-table:render	1572
all-accounts:::view	1449
dashboard:my-book:layout:render	1327
action-center:action-details::view	1191
action-center:::view	1172
action-center:::close-click	1093
all-accounts::accounts-table:account-click	1032
account-lines:::change-rating-click	1003
::nav-header:action-center-click	779
account-lines::layout:render	746
account-property-rating:perils::view	740
submissions:all-policy::view	683
::configurable-table:render	668
account-page-view	617
:all-accounts:configurable-table:render	517
unified-dashboard-page-view	497

Event_type Where Session Duration is 0

As mentioned above, a session duration of 0 implies that there is only one action taken for a particular session id. This insight is important because it shows the causes of low engagement. The table above shows the different categories in event_type and their corresponding value counts, arranged in descending order. After the top 3 values, the rest of the top values are all view or render. They are mostly related to dashboard and account, followed by action-center.



User Activity for Underwriters and Non-underwriters

In these graphs, “underwriters” are defined as any user that has “underwriter” as one of their roles, if they have multiple. The shape of both curves are nearly identical and align with the visualization for “User Activity by Hour of the Day” as presented in an earlier section. However, note that the total number of events for underwriters are significantly higher than that of non-underwriters, as indicated by the scale in the y-axis. This implies that underwriters perform more actions and engage with the Federato RiskOps Platform more.

Data Processing - Creating training datasets

Metrics

From the initial round of EDA, we define our goal is to recommend user’s action that will optimize retention and time spent on the platform. Specifically, we define retention as **“If the user returns to the platform 7 days after the day they visit”** and time spent on the platform as **“Total time spent for each day”.**

Goal

Create two datasets to train three models for a recommendation system that optimizes user engagement and retention.

- The first dataset focuses on retention and session duration, where the target variables include whether a user returns within seven days after a session and the time spent on the platform per session. This dataset enables the training of two models: one predicting user retention and another estimating session duration.
- The second dataset is designed to predict the user's most probable action, capturing behavioral patterns that drive engagement.

By integrating these models, the recommendation system can not only anticipate user actions but also suggest content or features that enhance session duration and increase retention, ultimately improving overall user experience.

First Dataset
In the first dataset, we first perform one hot encoding for the categorical columns, we then drop all rows that has more than 50% of missing values, which leaves us with the following columns”

columns = ['user_id_first', 'device_family_linux_max',
'device_family_mac os x_max', 'device_family_windows_max',
'region_grouped_international_max', 'region_grouped_midwest_max',
'region_grouped_northeast_max', 'region_grouped_south_max',
'region_grouped_west_max',
'event_category_account & policy management_max',
'event_category_action center & workflow_max',
'event_category_dashboard & ui interactions_max',
'event_category_other/system events_max',
'event_category_session & navigation_max',
'event_category_submission & forms_max', 'uw_max', 'admin_max',
'manager_max', 'broker_max', 'google_max', 'microsoft_max',
'client_event_hour_mean', 'client_upload_hour_mean', 'event_hour_mean',
'server_received_hour_mean', 'server_upload_hour_mean',
'time_to_server_mean', 'server_to_process_mean', 'processing_time_mean',
'returned_within_7_days', 'session_seconds',
'uw_max/manager_max', 'uw_max/admin_max', 'uw_max/broker_max',
'uw_max/manager_maxadmin_max']

For retention the target column is ‘returned_within_7_days’, and for time spent on the platform per day the target column is 'session_seconds'.

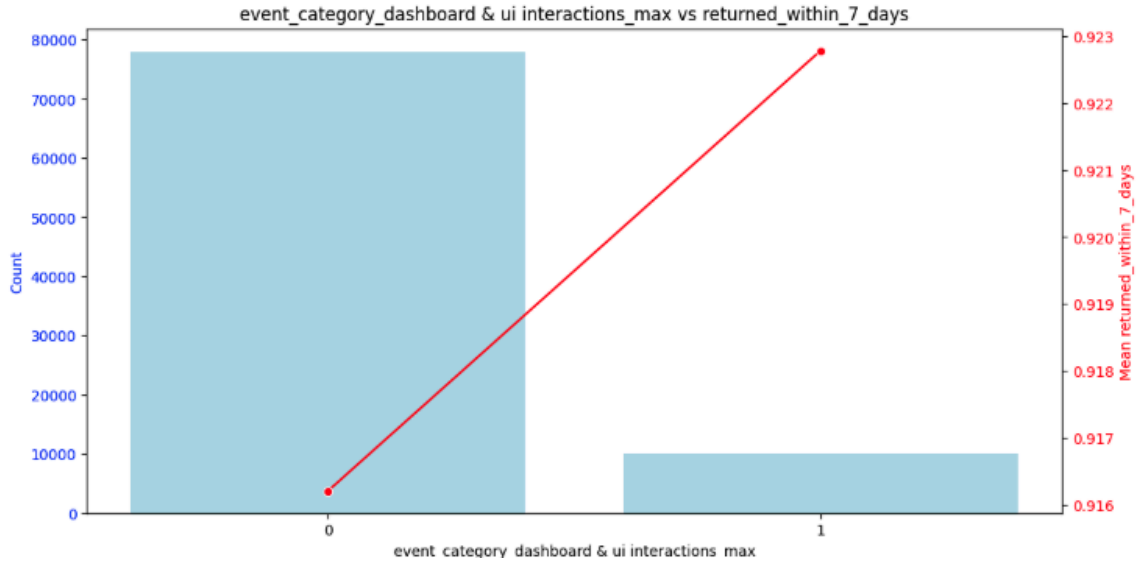
Second Dataset
In the second dataset, we group by user_id and session_id and aggregate sequence of actions taken during that session. Each action is assigned a numerical token ranging from 0 to 740, where 740 represents an empty token. To train the model effectively, we transform each sequence into multiple training samples by progressively hiding the next action in the sequence. This means that for each session, we generate multiple rows where the model is given a partial sequence as input and is trained to predict the next action. This approach helps the model learn action patterns and anticipate what a user is most likely to do next based on their past interactions.

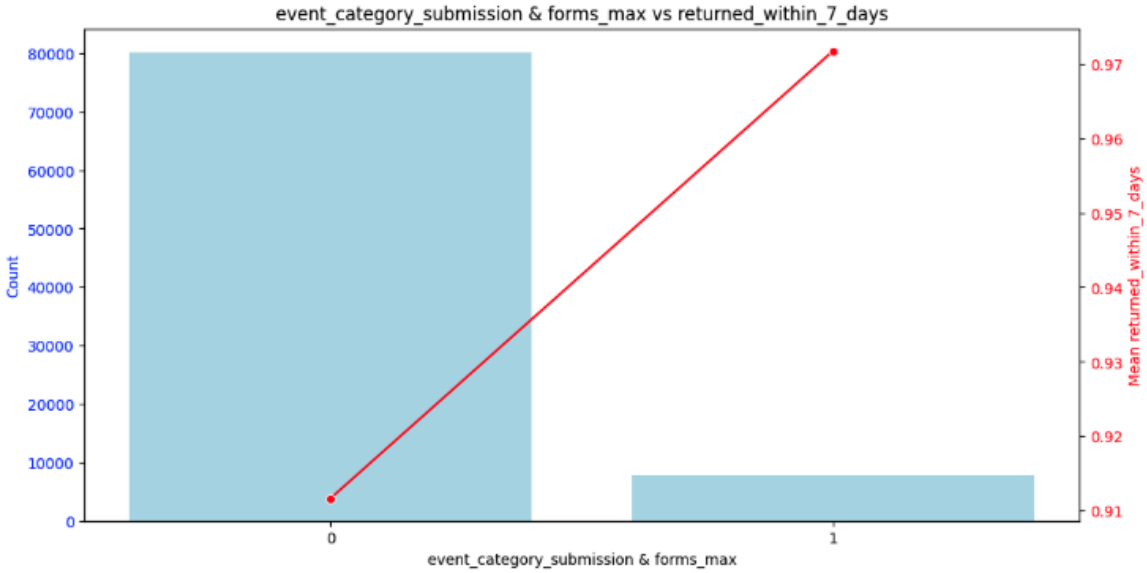
EDA On Retention

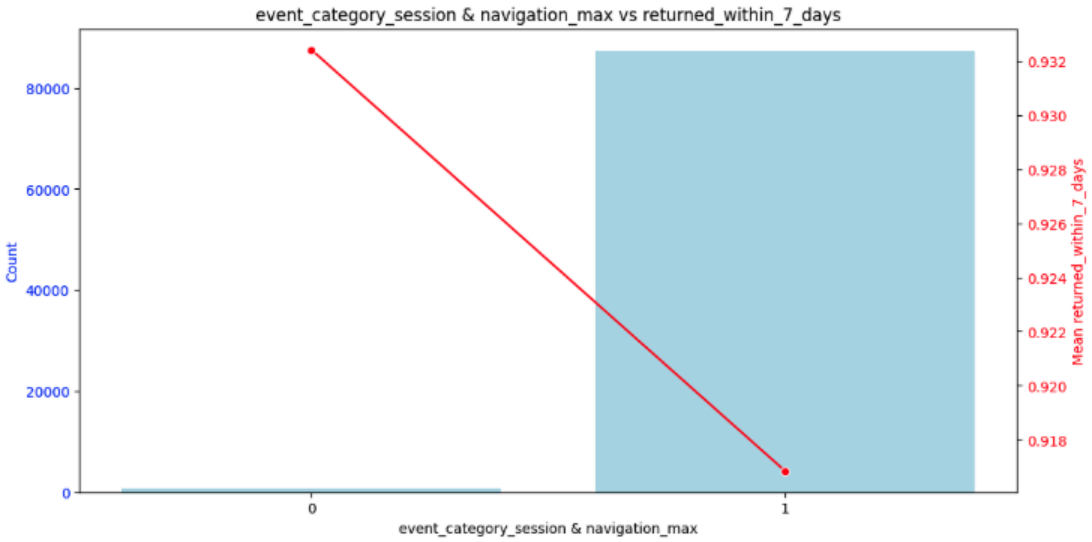
Event_category


event_category_account & policy management_max vs returned_within_7_days
Count
Mean returned_within_7_days
event_category_account & policy management_max
0
1
0
10000
20000
30000
40000
50000
60000
70000
0.80
0.82
0.84
0.86
0.88
0.90
0.92
0.94


event_category_action center & workflow_max vs returned_within_7_days
Count
Mean returned_within_7_days
event_category_action center & workflow_max
0
1
0
10000
20000
30000
40000
50000
0.88
0.90
0.92
0.94
0.96
0.98

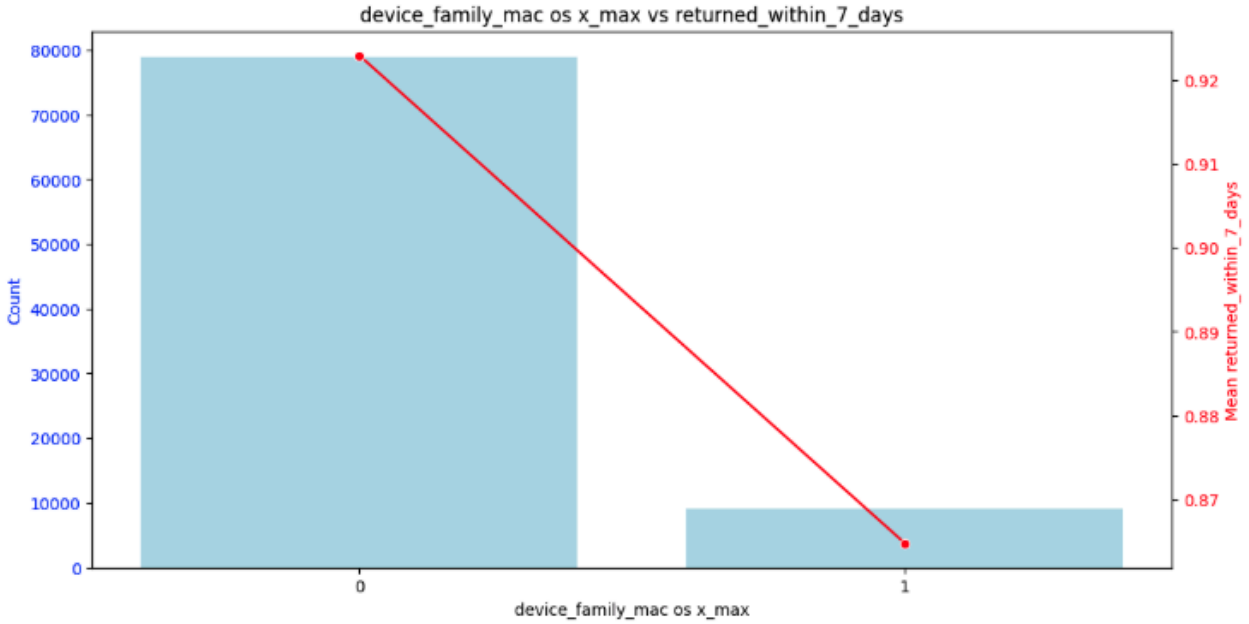

event_category_dashboard & ui interactions_max vs returned_within_7_days
Count
Mean returned_within_7_days
event_category_dashboard & ui interactions_max
0
1
0
10000
20000
30000
40000
50000
60000
70000
80000
0.916
0.917
0.918
0.919
0.920
0.921
0.922
0.923

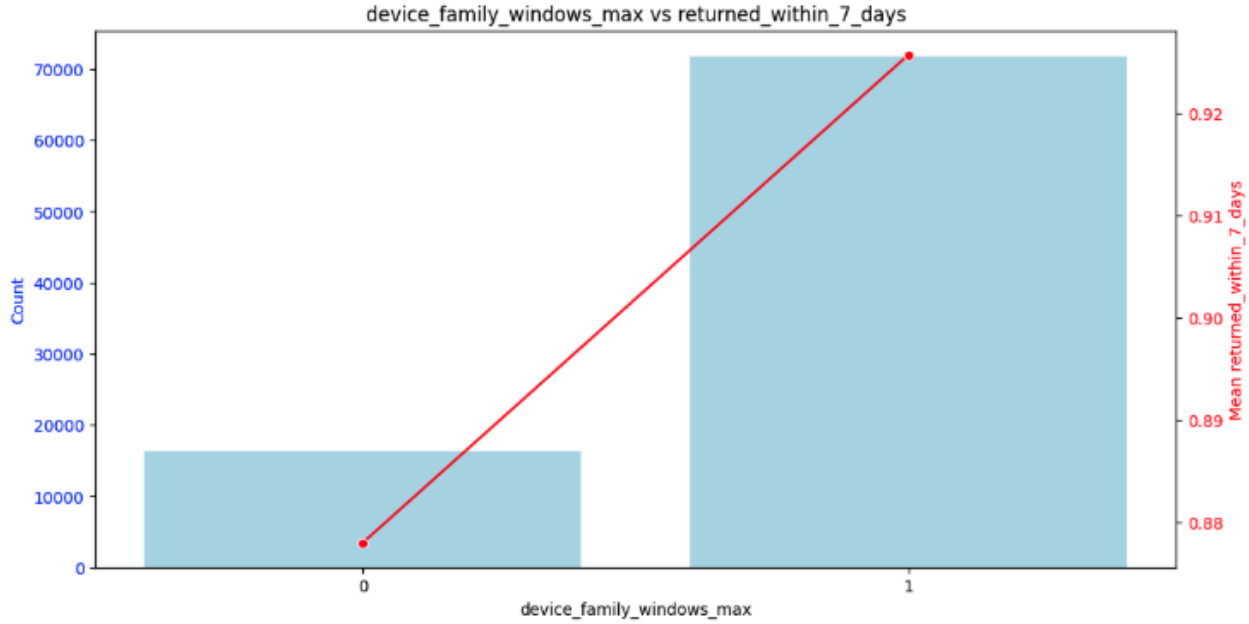


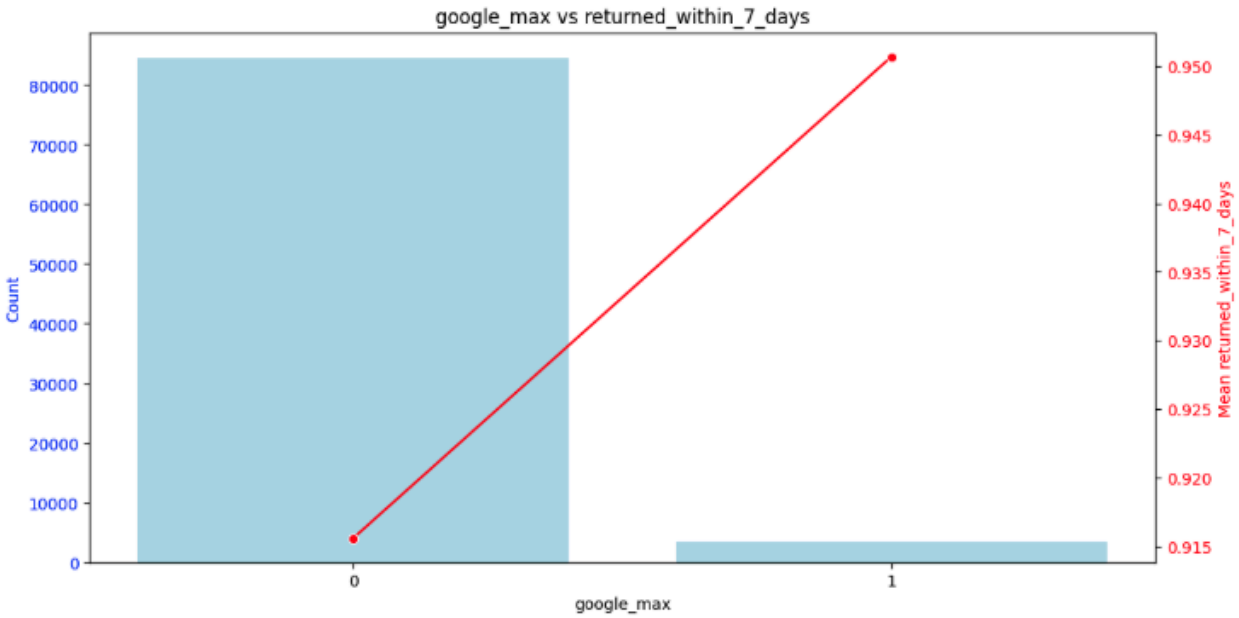


Account & Policy Management and Submission & Forms show positive correlations with return rates, suggesting that users who engage more with these features are more likely to return. Conversely, Session & Navigation displays a negative trend, indicating potential usability issues. Action Center & Workflow sees lower event counts but high return rates, highlighting its importance for retention despite limited usage. Dashboard & UI Interactions show moderate retention influence with increased activity, albeit among fewer users. These patterns suggest that enhancing navigation and leveraging high-return features could improve overall user retention.

Device_family

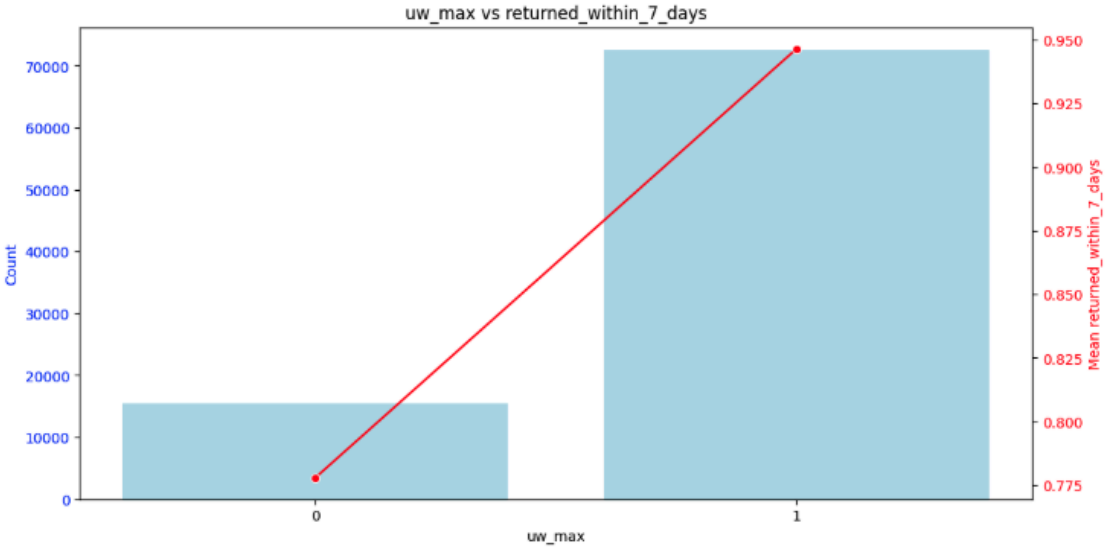

device_family_mac os x_max vs returned_within_7_days
Count
Mean returned_within_7_days
device_family_mac os x_max
0
1
80000
70000
60000
50000
40000
30000
20000
10000
0
0.92
0.91
0.90
0.89
0.88
0.87

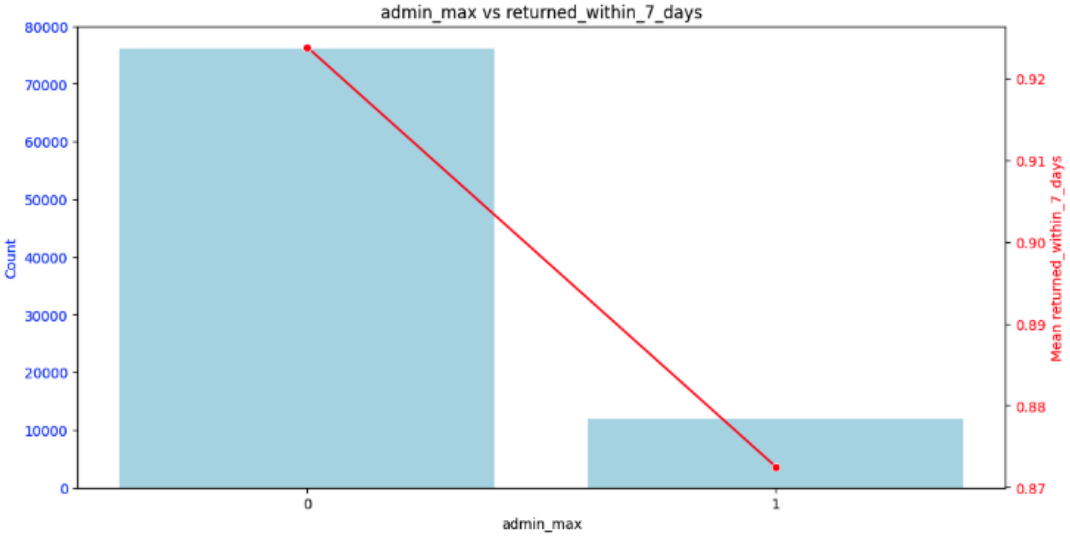

device_family_windows_max vs returned_within_7_days
Count
Mean returned_within_7_days
device_family_windows_max
0
1
70000
60000
50000
40000
30000
20000
10000
0
0.92
0.91
0.90
0.89
0.88

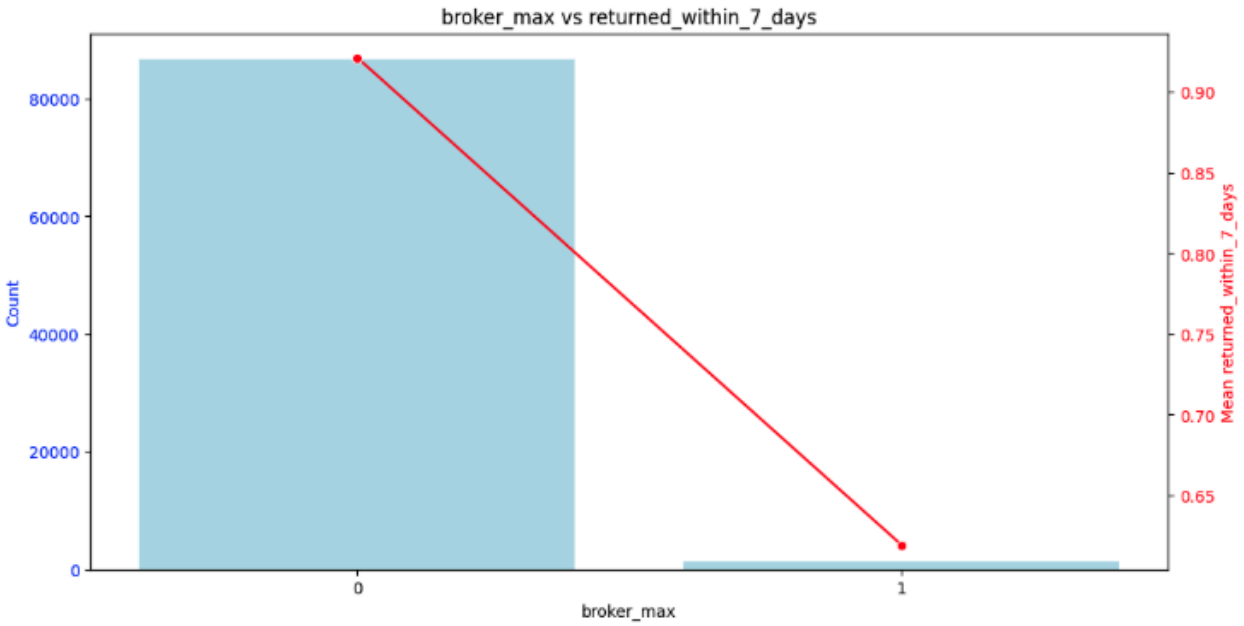


The data reveals distinct platform-based retention patterns: Mac OS X usage correlates with decreased retention (negative trend), while both Windows and Google usage show strong positive correlations with higher retention rates. The substantially larger Windows user base with its positive retention trend suggests focusing optimization efforts there, while the exceptionally strong retention among the smaller Google user segment indicates growth potential. These patterns suggest prioritizing platform-specific enhancements to leverage the positive Windows and Google trends while addressing factors behind the negative Mac retention trend.

Roles:

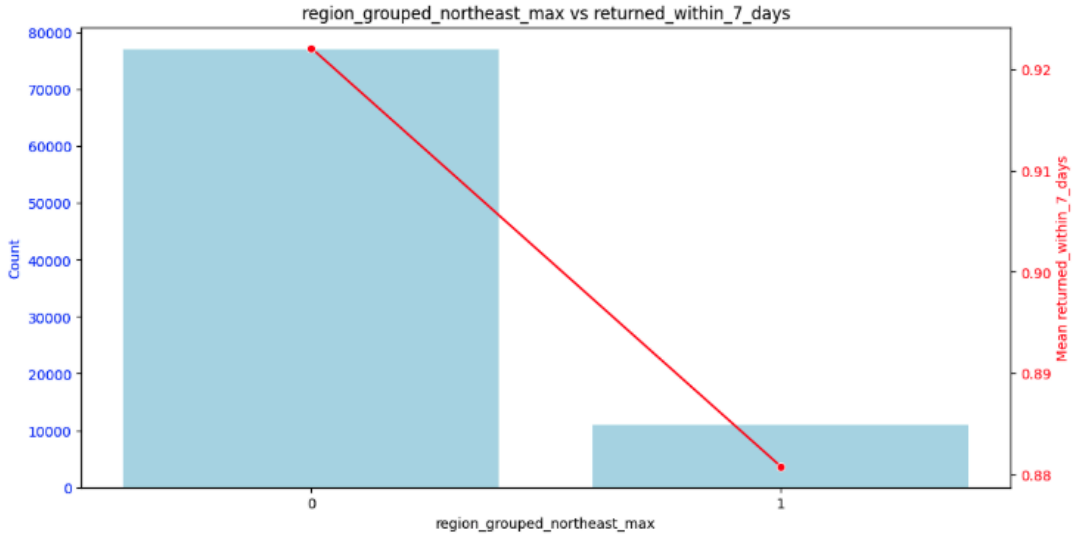

uw_max vs returned_within_7_days
Count
Mean returned_within_7_days
uw_max
0
1
0
10000
20000
30000
40000
50000
60000
70000
0.775
0.800
0.825
0.850
0.875
0.900
0.925
0.950

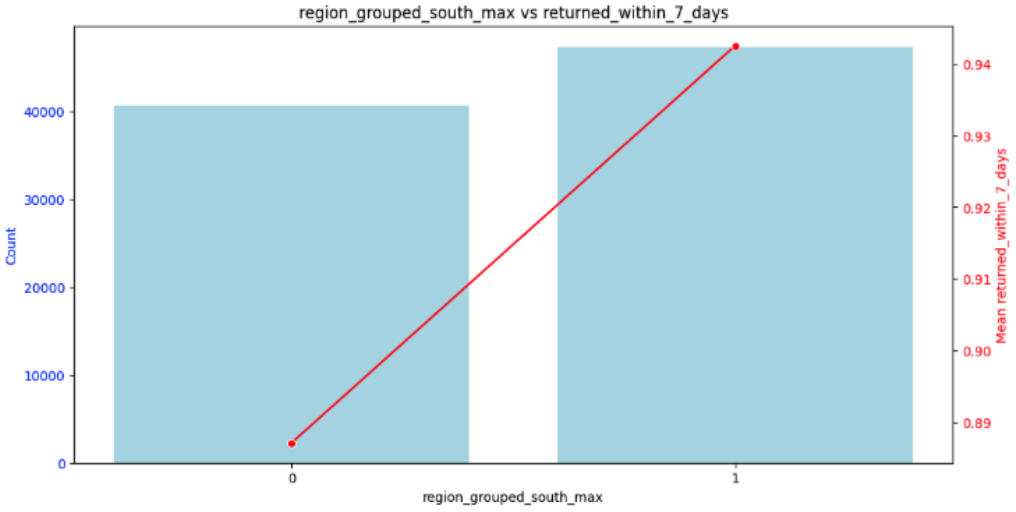

admin_max vs returned_within_7_days
Count
Mean returned_within_7_days
admin_max
0
1
0
10000
20000
30000
40000
50000
60000
70000
80000
0.87
0.88
0.89
0.90
0.91
0.92

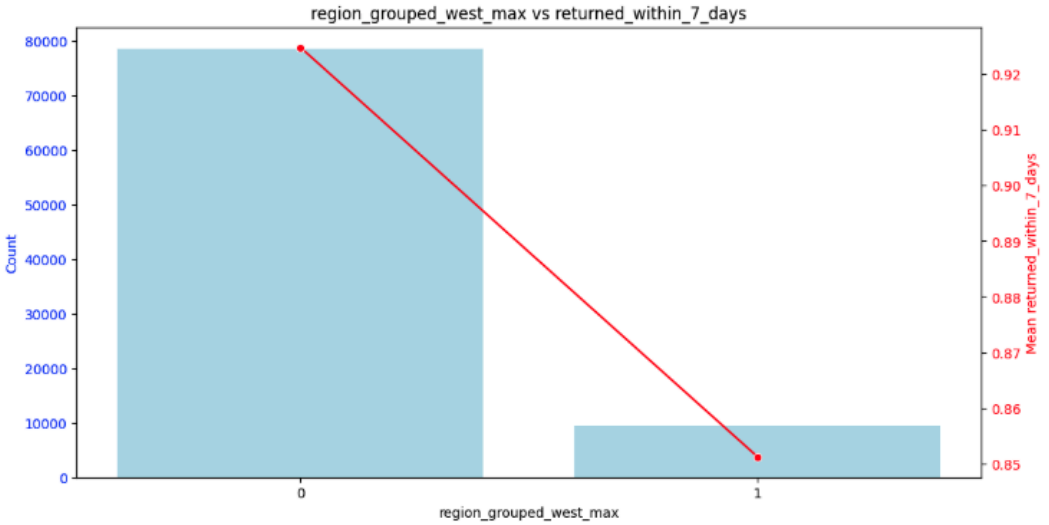


The data reveals clear role-based patterns in user retention behavior: Underwriter users show a strong positive retention trend, with significantly higher 7-day return rates compared to non-UW users. In contrast, both Admin and Broker roles display negative retention trends, with Broker users exhibiting the most dramatic decline - dropping approximately 25% compared to non-Broker users. The steepness of these trend lines indicates that user role serves as a powerful predictor of retention behavior, with the Underwriter experience appearing to satisfy user needs while the Admin and especially the Broker experiences contain elements that potentially discourage continued engagement. These opposing directional trends suggest focusing enhancement efforts on replicating positive aspects of the Underwriter experience within the significantly underperforming Broker interface.

Region:



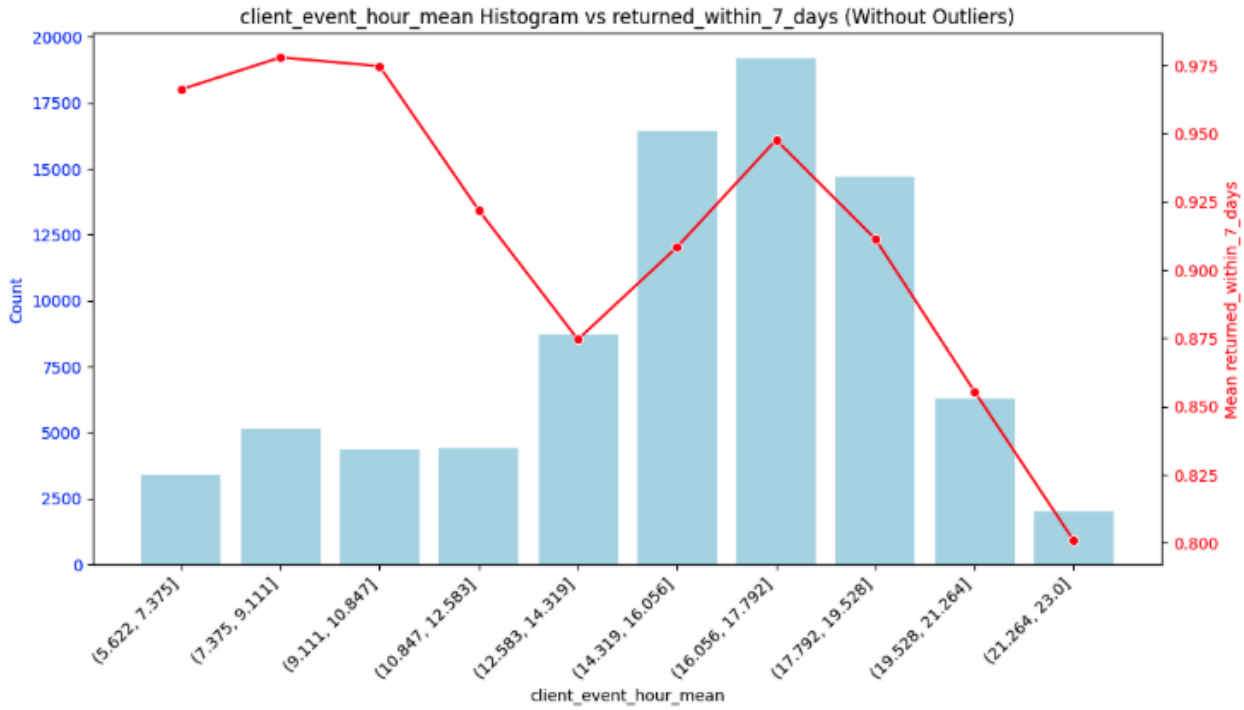




The data reveals pronounced geographical patterns in user retention trends: Northeast and West regions both exhibit strong negative correlation with retention rates, with users from these regions demonstrating significantly lower 7-day return probabilities compared to non-regional

users. In stark contrast, the South region displays a robust positive retention trend, with Southern users showing markedly higher retention compared to users outside this region. The steepness of these trend lines, particularly the sharp decline for West region users, indicates geographic location functions as a powerful predictor of retention behavior, suggesting region-specific factors are substantially influencing user engagement patterns and highlighting opportunities for targeted improvements in the underperforming Northeast and West regions.

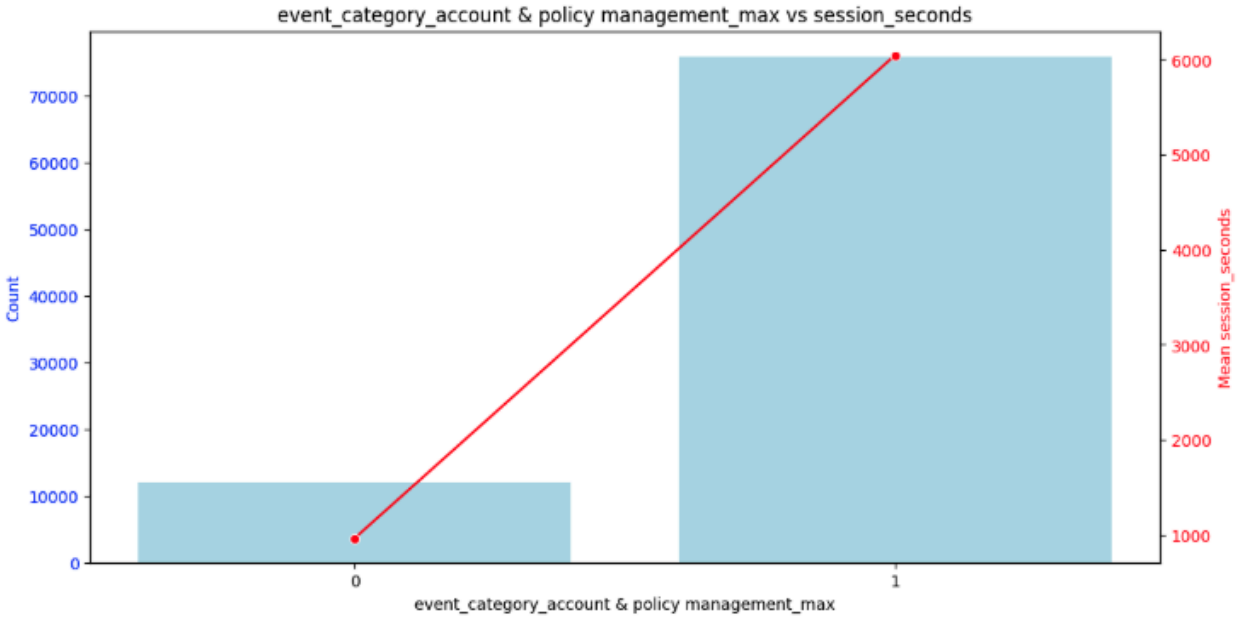
User's Usage time during the day

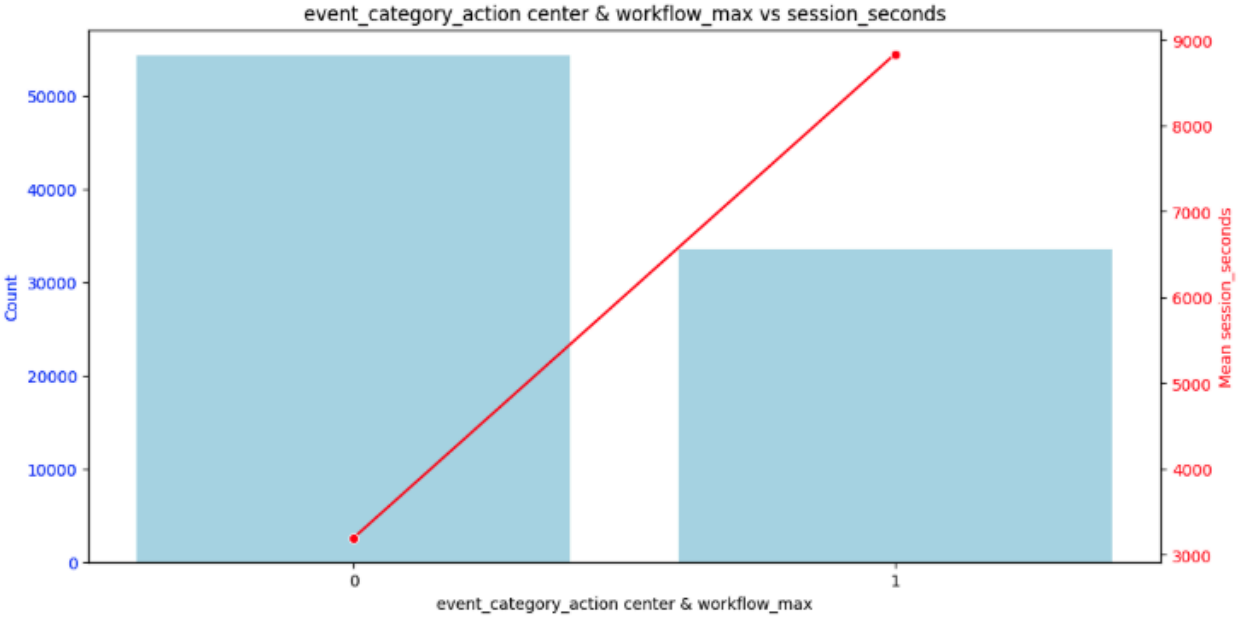


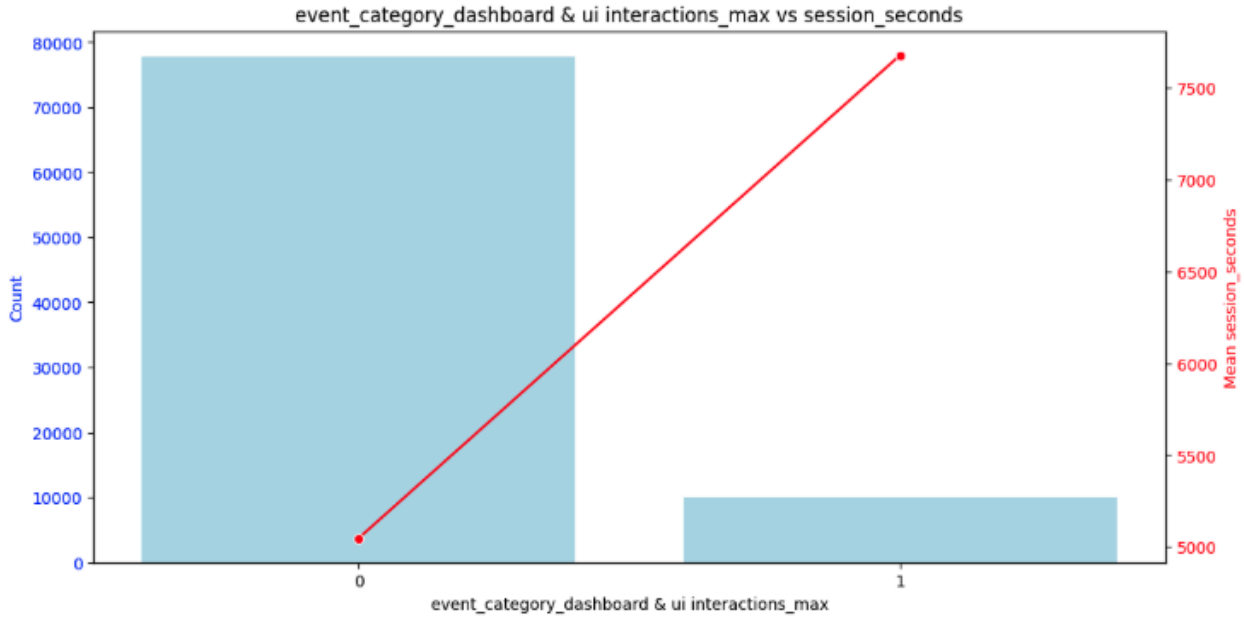
Retention rates show an inverse U-shape throughout the day, peaking at 97.5% in the morning (07:00-11:00), dipping to 87.5% around midday (12:00-14:00), and briefly rising in the afternoon (15:00-16:00) before dropping to 80% late at night (23:00-24:00). User activity is low in the morning despite high retention but peaks midday and afternoon when retention is moderate. This contrast suggests a need to enhance the midday user experience when engagement is highest, and address issues impacting evening retention despite lower traffic.

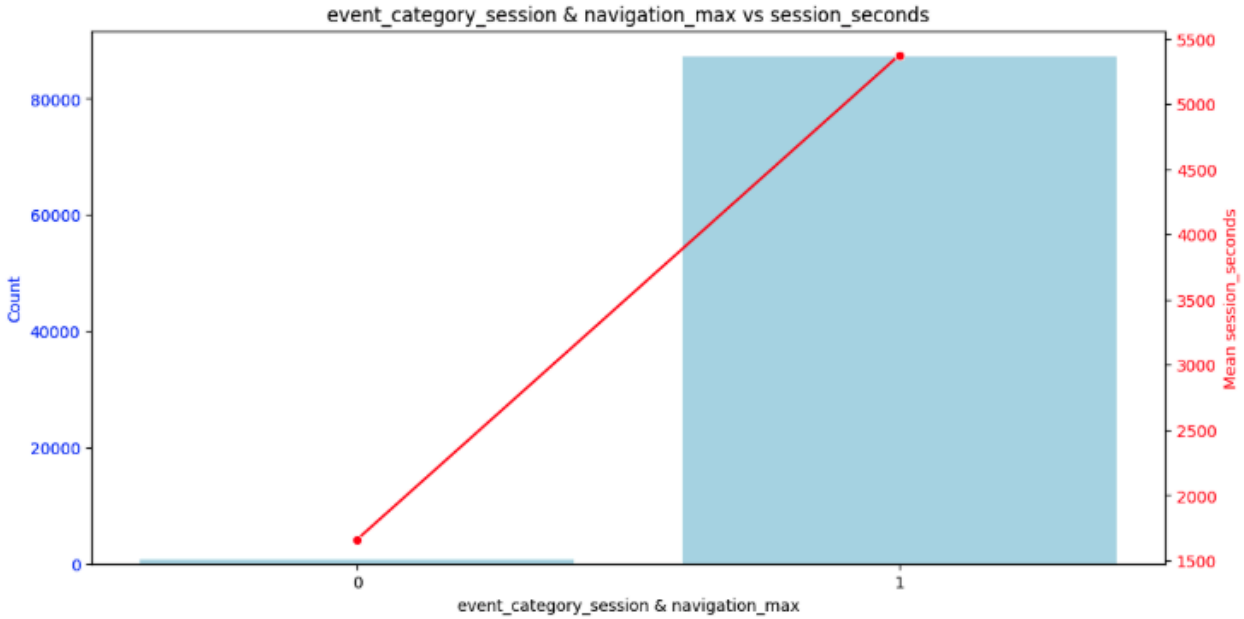
EDA On Session Time

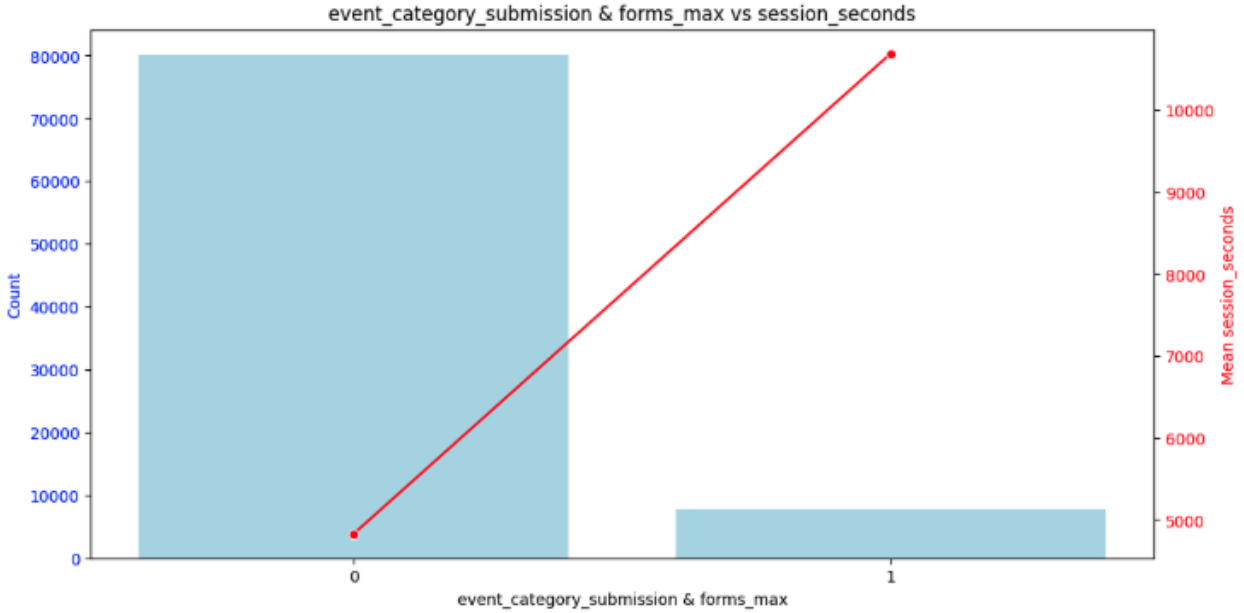
Event_category


event_category_account & policy management_max vs session_seconds
Count
Mean session_seconds
0
1
event_category_account & policy management_max


event_category_action center & workflow_max vs session_seconds
Count
Mean session_seconds
0
1
event_category_action center & workflow_max

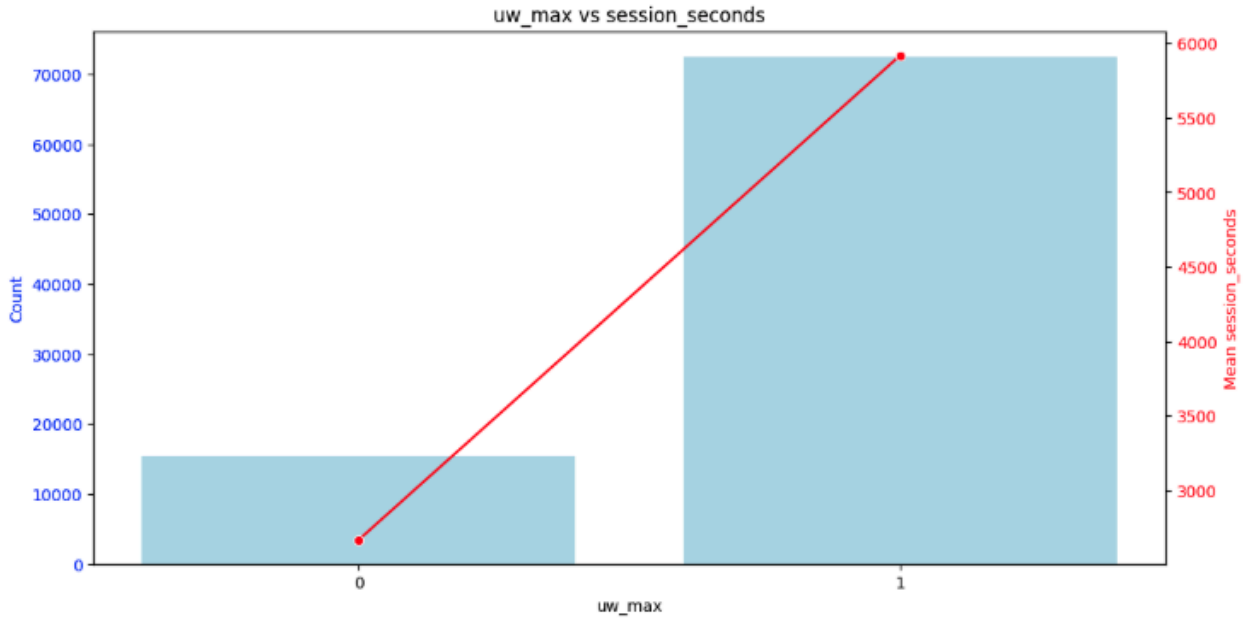

event_category_dashboard & ui interactions_max vs session_seconds
Count
Mean session_seconds
event_category_dashboard & ui interactions_max
0
1
80000
70000
60000
50000
40000
30000
20000
10000
0
7500
7000
6500
6000
5500
5000

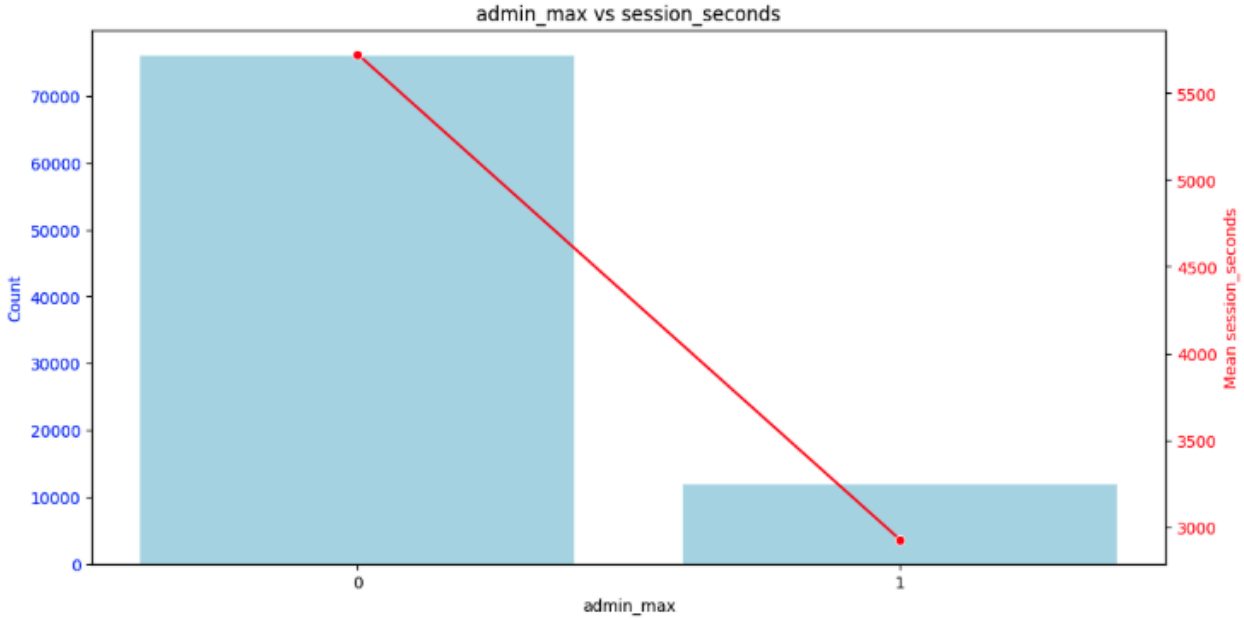


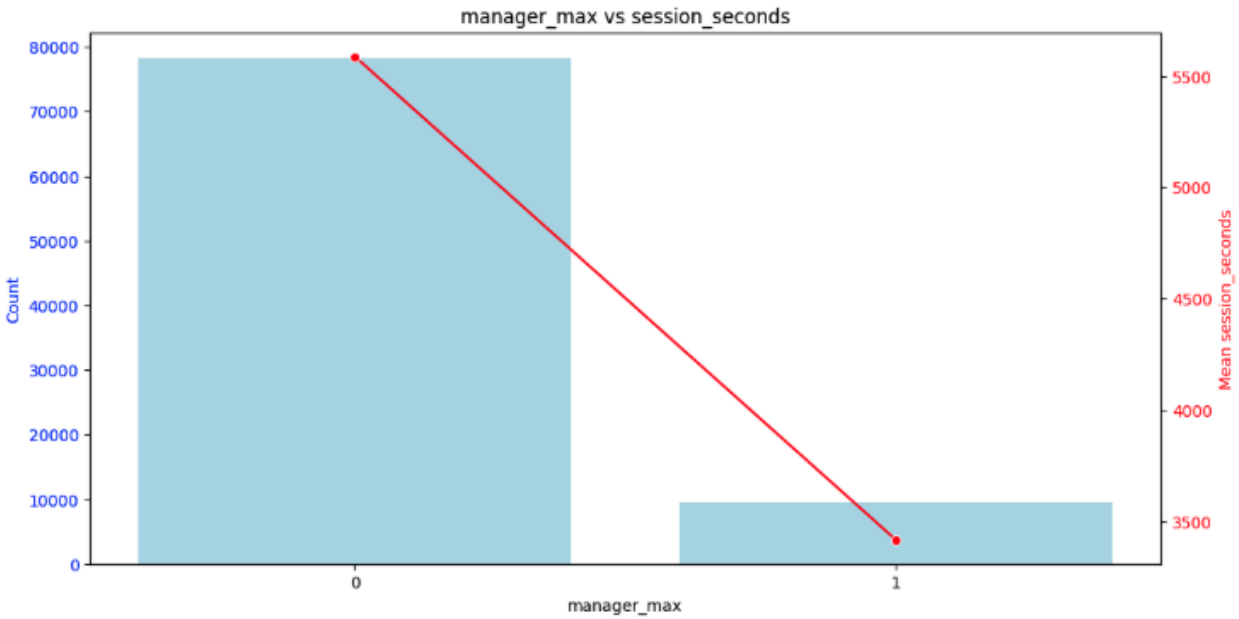


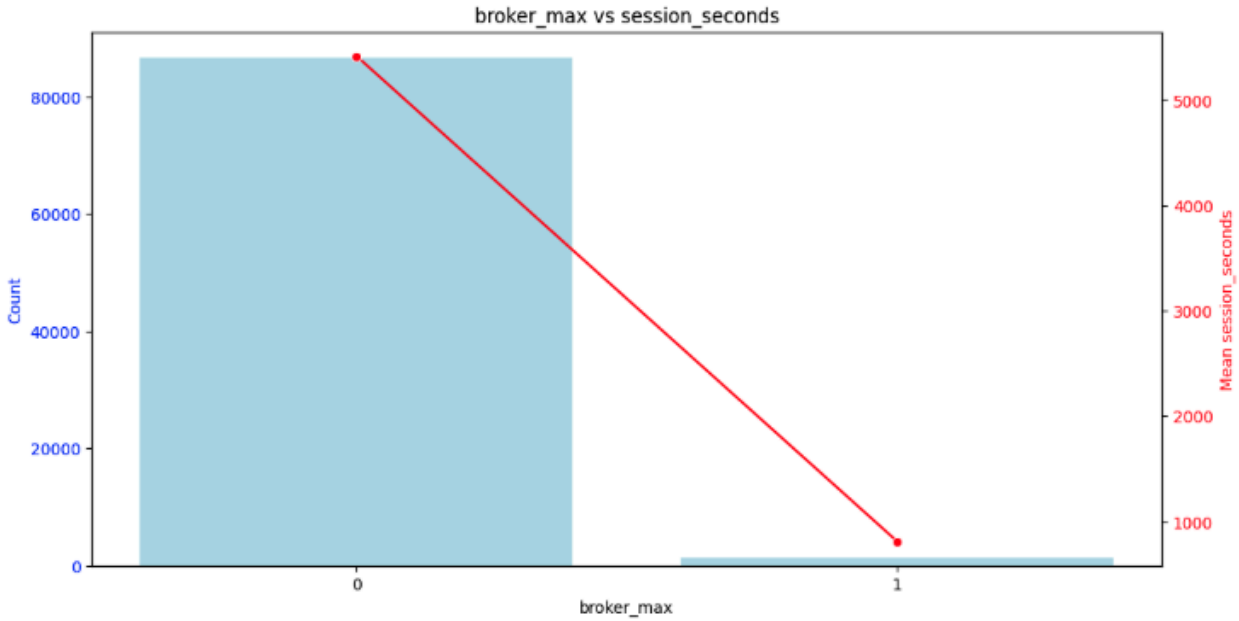
All event categories show positive correlation with session duration. Focusing on the difference in session duration in the presence and absence of an event, Submission & Forms has the biggest difference of more than 5000 seconds. This makes sense intuitively as the user would have to spend time gathering information and filling out the form on the platform. Dashboard & UI Interactions has the lowest difference of 2500 seconds, which may be due to the nature of the event. A dashboard is typically set up initially, then refreshed on a regular basis for viewing and tracking purposes. The user could quickly capture the information on the platform and incorporate it into their report.

Roles


uw_max vs session_seconds
Count
Mean session_seconds
uw_max
0
1
0
10000
20000
30000
40000
50000
60000
70000
3000
3500
4000
4500
5000
5500
6000

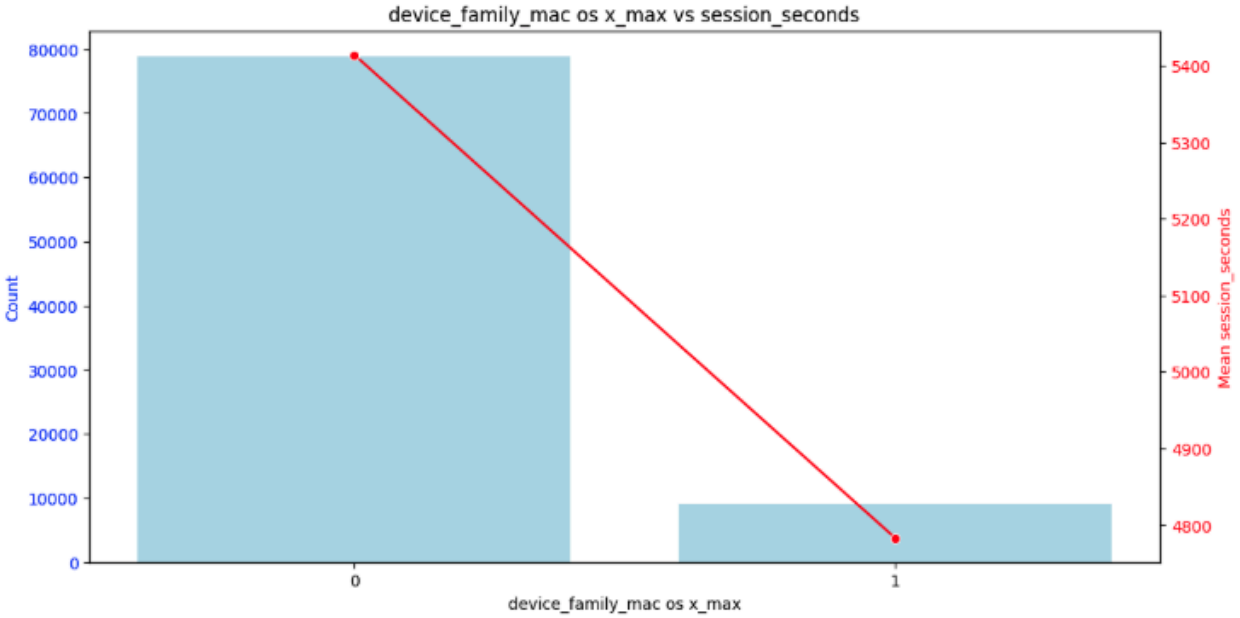

admin_max vs session_seconds
Count
Mean session_seconds
admin_max
0
1
0
10000
20000
30000
40000
50000
60000
70000
3000
3500
4000
4500
5000
5500

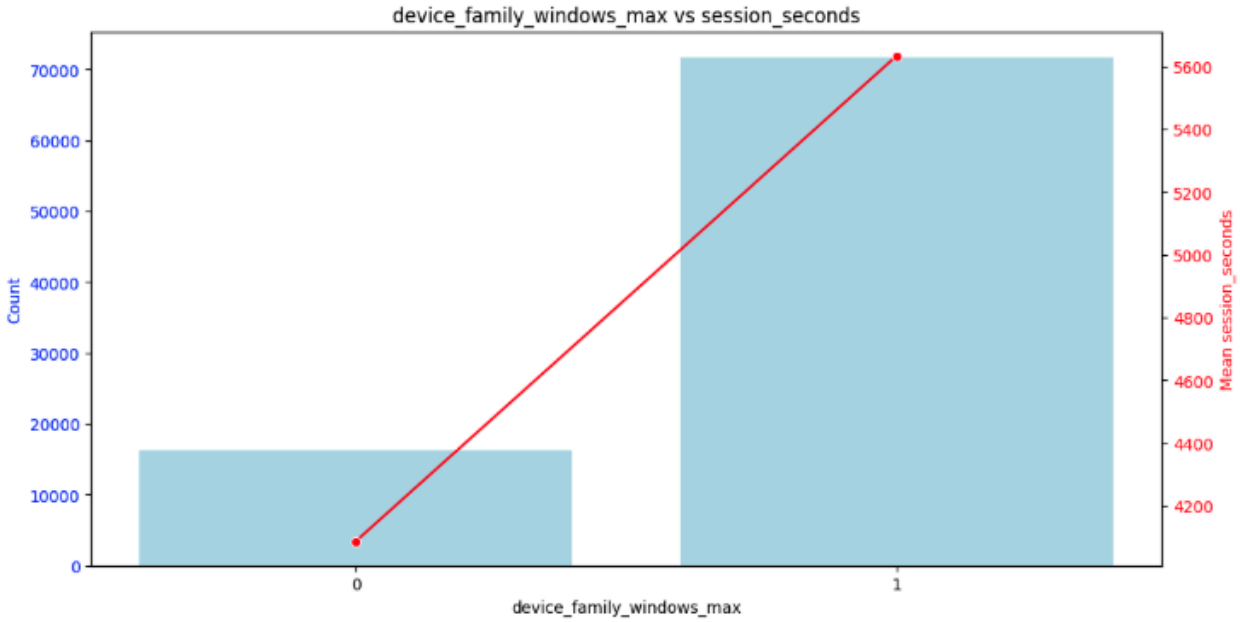


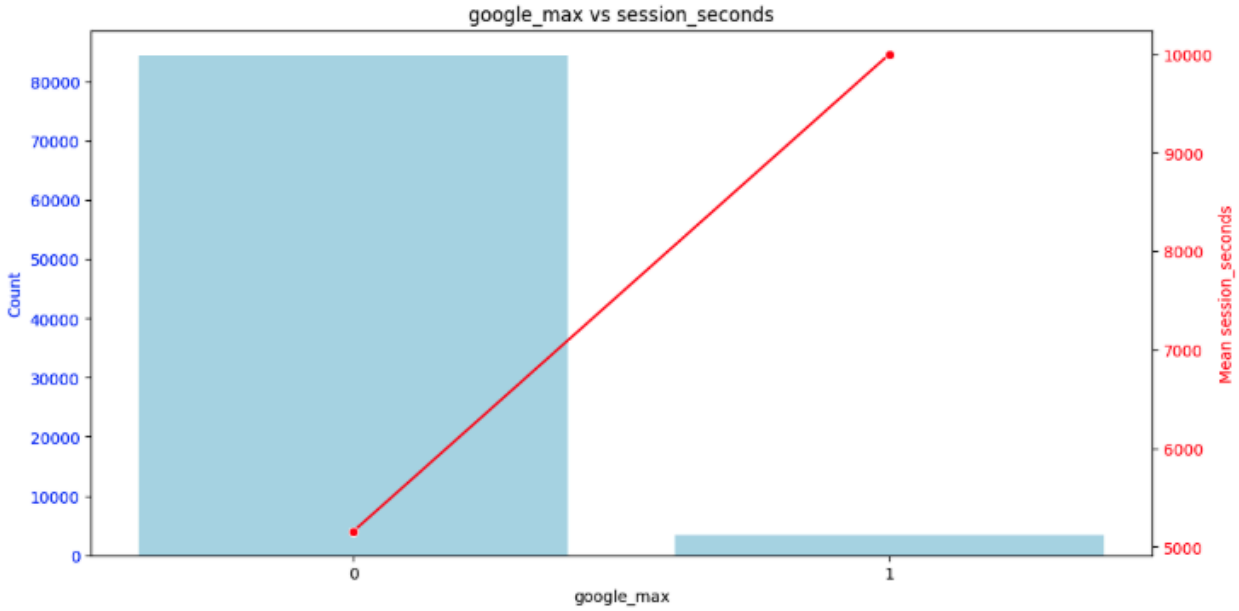


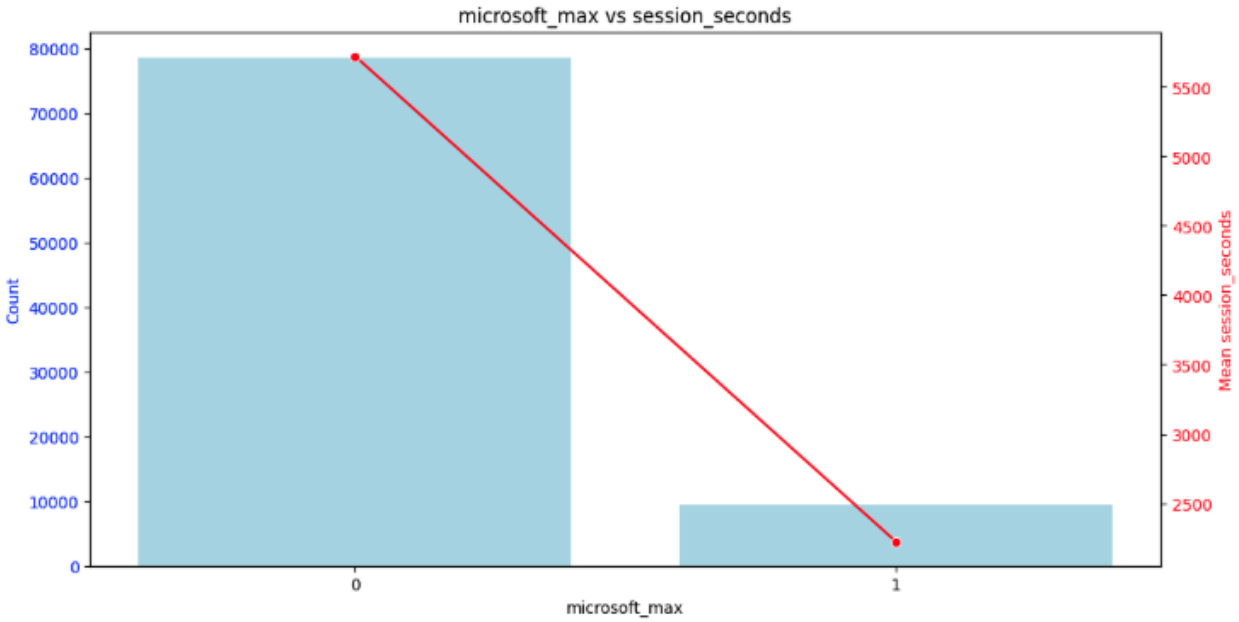
The graphs show a positive correlation between underwriters and session duration, while other roles have a negative correlation. Note that underwriters have the highest count in roles, indicating that underwriters are the main users on the platform and they are actively using the platform. These patterns suggest we could focus on underwriters' needs when allocating resources to improve the platform.

Device family / os


device_family_mac os x_max vs session_seconds
Count
Mean session_seconds
device_family_mac os x_max
0
1
80000
70000
60000
50000
40000
30000
20000
10000
0
5400
5300
5200
5100
5000
4900
4800

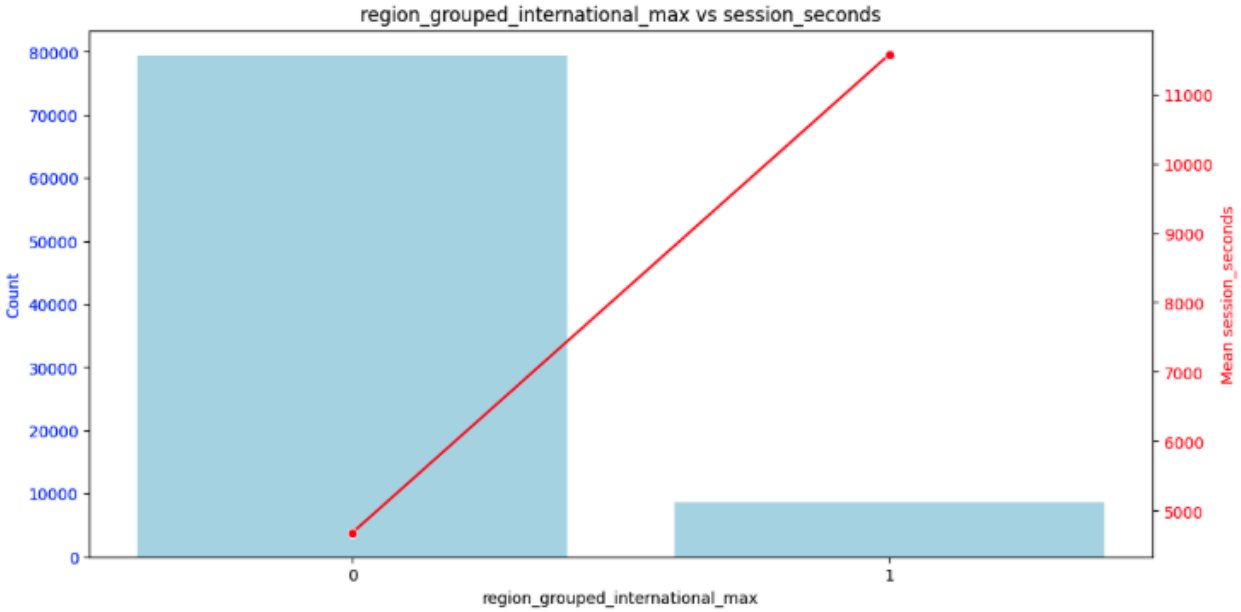

device_family_windows_max vs session_seconds
Count
Mean session_seconds
device_family_windows_max
0
1
70000
60000
50000
40000
30000
20000
10000
0
5600
5400
5200
5000
4800
4600
4400
4200

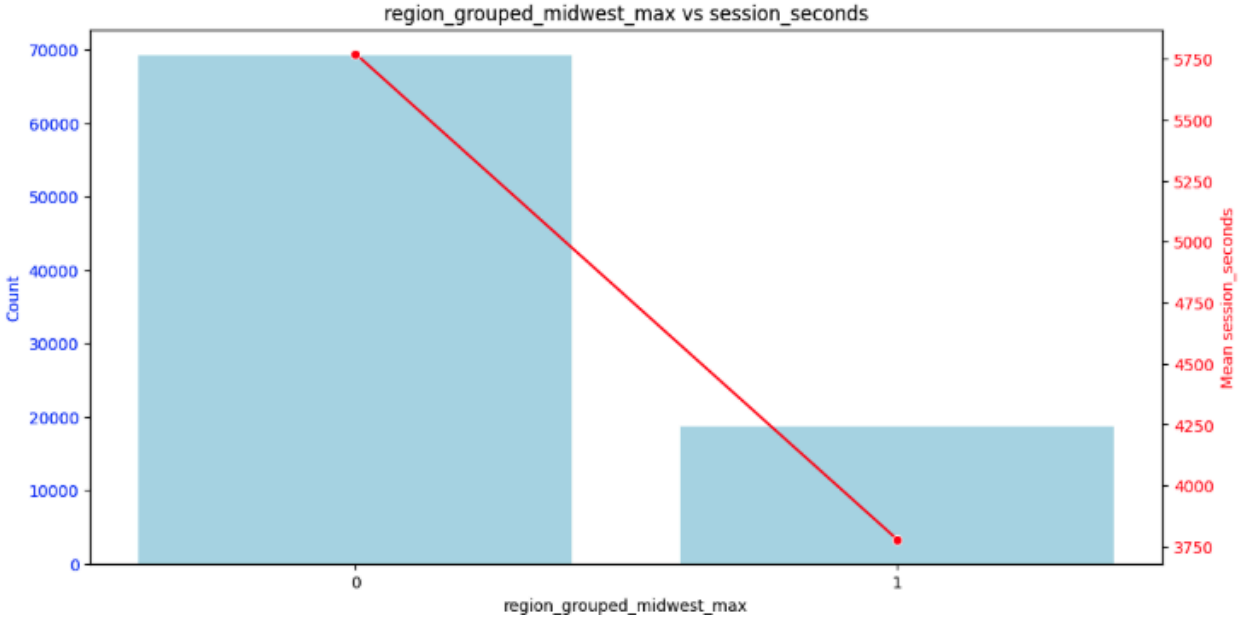


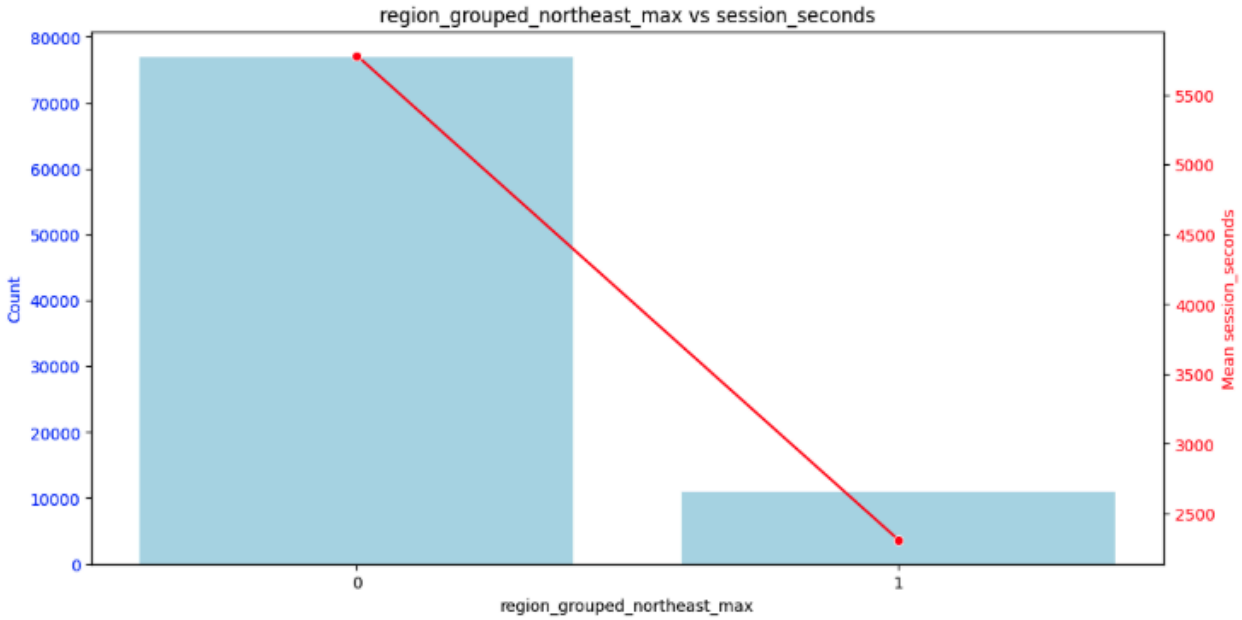


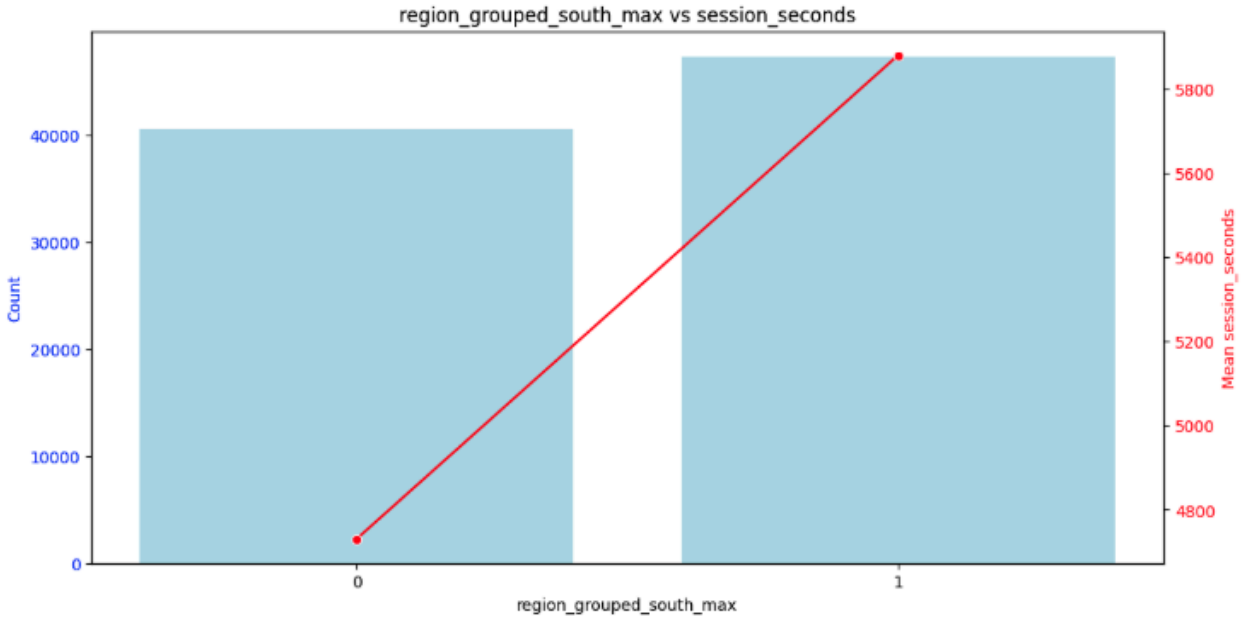
The graphs indicate Windows and Google device users have a higher session duration on average. In terms of category count, Windows has the highest frequency, followed by Mac and Microsoft. However, Mac and Microsoft have a negative correlation with session duration. To increase user engagement, we may wish to investigate the customer experience of Mac and Microsoft users and see if the platform functions can be improved for their machines.

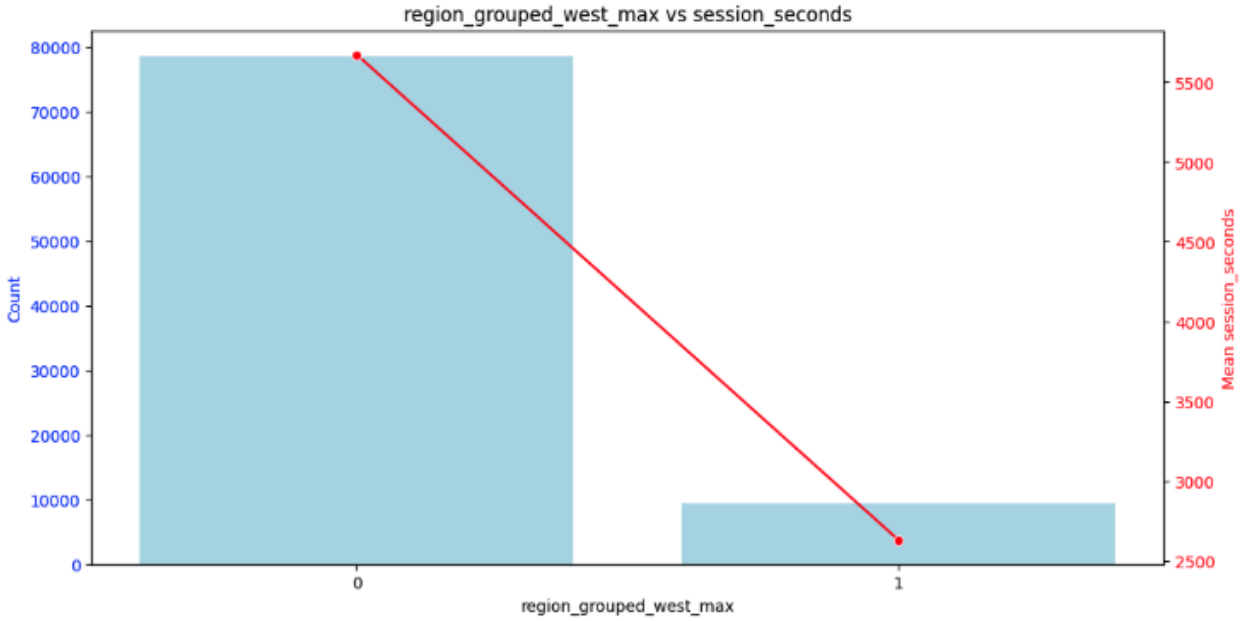
Region


region_grouped_international_max vs session_seconds
Count
Mean session_seconds
region_grouped_international_max
0
1
80000
70000
60000
50000
40000
30000
20000
10000
0
11000
10000
9000
8000
7000
6000
5000


region_grouped_midwest_max vs session_seconds
Count
Mean session_seconds
region_grouped_midwest_max
0
1
70000
60000
50000
40000
30000
20000
10000
0
5750
5500
5250
5000
4750
4500
4250
4000
3750

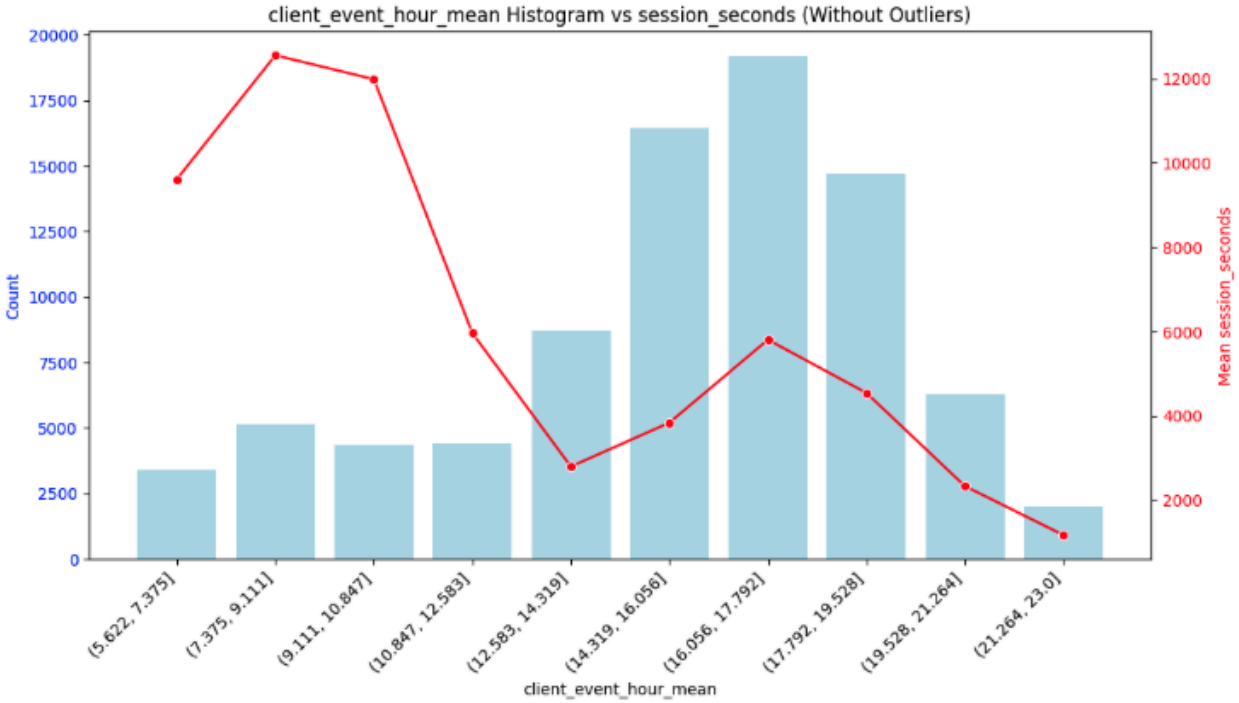

region_grouped_northeast_max vs session_seconds
Count
Mean session_seconds
region_grouped_northeast_max
0
1
80000
70000
60000
50000
40000
30000
20000
10000
0
5500
5000
4500
4000
3500
3000
2500





The graph for South indicates that it is the major category in this variable. The difference in session duration between South and non-South is the lowest, indicating that South is close to the overall variable mean and the effects of the other categories cancel out with each other. The category with the biggest difference is International, with a difference of more than 6000 seconds. The graphs indicate a positive correlation between session duration with International and South, and a negative correlation for Midwest, Northeast and West. With midwest being the second largest category, we may want to look into how the platform can be improved to better suit their needs.

User's usage time during the day



There are a few insights we can get from this visualization. Notice the two peaks at 7-9am and 4-5pm for both counts (bar graph) and session duration (line graph). These seem to coincide with the regular working hours of 9am to 5pm. This may involve users getting on the platform at the start of the day, turning off their laptops at the end of the day and closing the app, so this is a potential reason why there is high user engagement at those times. Additionally, notice the trough in session duration around 12pm-2pm, which is lunch hours. This may not be the platform's shortcomings, but rather, the users' behavioral patterns. We see that the second peak has a high count and low session duration. In comparison, the first peak has low count but high session duration, which may possibly indicate inefficient working times. To improve the platform, we can consider displaying relevant actions for the user to execute, based on their platform usage and action patterns.

Modeling

1. **User's likely action**

Dataset

Split the dataset into training (80%), validation (10%), and test sets (10%) based on user identifiers. Use a rule to divide the data, ensuring each set represents different user groups or sessions for training, validation, and testing. Extract the sequences and targets from each subset as lists, preparing them for use in creating data batches for the model.

Finally, create a custom dataset class to manage the sequences and targets. This class handles padding shorter sequences to a fixed length and truncating longer ones, ensuring uniformity for

model processing. Use a padding value outside the valid range of item identifiers, such as 740, to indicate non-relevant positions in the sequences.

BERT4Rec Structure
The BERT4Rec model builds on a transformer architecture, specifically adapting the BERT model for sequential recommendations. It processes input sequences of item identifiers, padded or truncated to a consistent length, and predicts the next item using a transformer-based encoder.

The model starts with an input layer that accepts sequences of numbers, padded with a special value to maintain uniform length. An attention mask is applied to ignore padded positions, allowing the model to focus on actual interaction data. The core of the model consists of transformer layers, configured with a set number of layers and attention heads, which capture contextual relationships within the sequences. These layers transform the input into hidden representations, capturing patterns and dependencies.

After processing through the transformer layers, the model extracts the hidden state of the last token in each sequence, representing the context of the entire sequence. This hidden state passes through an additional layer to produce output logits, which are numerical scores for each possible item in the vocabulary. The logits correspond to the likelihood of each item being the next action, enabling the model to predict the most probable item.
The model configuration includes parameters such as the vocabulary size, the size of hidden representations, the number of transformer layers, the number of attention heads, and the maximum sequence length. These settings define the model's capacity and behavior, tailored to the dataset's characteristics.

Evaluation
Evaluate the model using accuracy metrics tailored to recommendation systems. Focus on determining whether the correct item is among the top predicted options, such as the top five or ten recommendations. Use the loss value from training and validation to gauge overall model performance, ensuring it aligns with expected outcomes for recommendation tasks.

2. **Retention Model**

Dataset
Split the dataset into training (80%), validation (10%), and test sets (10%) based on user identifiers. This approach ensures that each set represents unique user groups or sessions, maintaining independence between training, validation, and testing data. This division helps the model generalize better by avoiding data leakage across sets.

After splitting the data, extract the input features and target variable (indicating whether a user returns after a session) from each subset. Organize them into lists for efficient batching during model training and evaluation.

XGBoost Structure

The XGBoost model is used for predicting customer retention, leveraging gradient boosting techniques for robust and accurate classification. XGBoost is chosen for its capability to handle complex patterns and interactions within the features, making it suitable for modeling user behavior and retention.

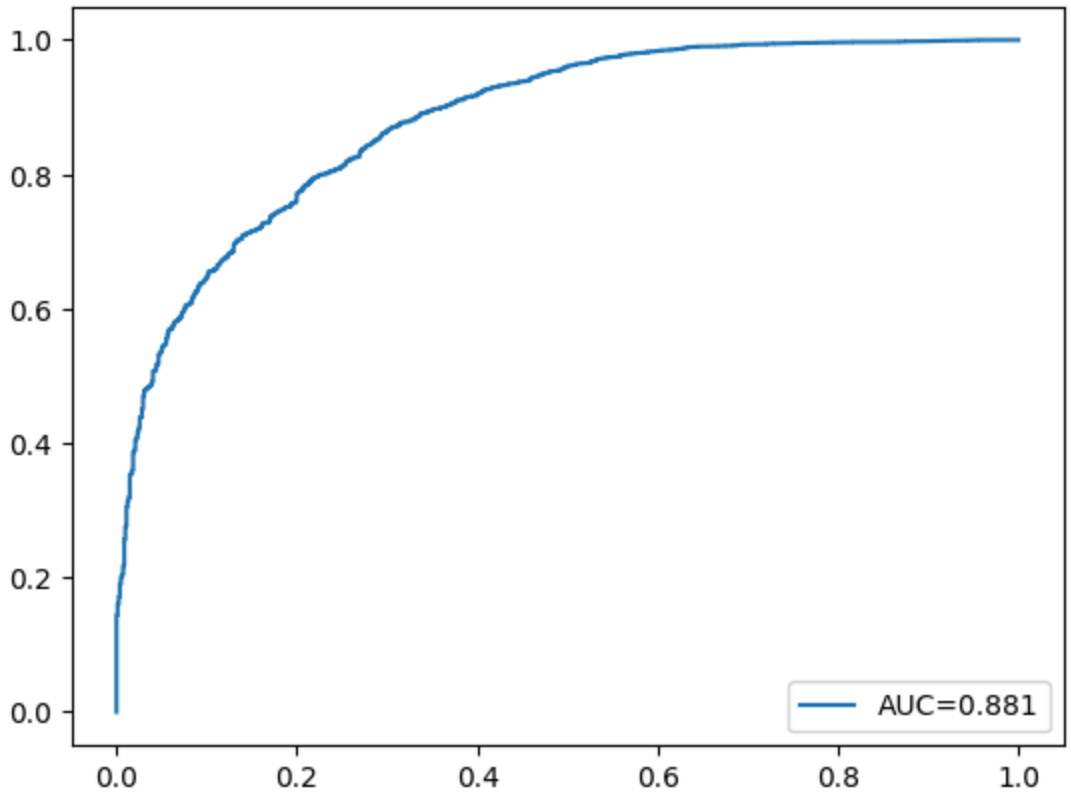
The model begins by taking the input features, standardized to ensure consistent scaling and improved model convergence. It utilizes decision trees as base learners, gradually refining predictions by learning from the residuals of previous iterations. This boosting approach allows XGBoost to effectively capture non-linear relationships and interactions within the dataset.

Key model parameters include the number of trees (estimators), maximum tree depth, learning rate, regularization terms, and subsampling ratios. These settings control the model's complexity and generalization, ensuring it balances bias and variance effectively. Hyperparameter tuning is conducted to optimize these values for the best performance on the validation set.

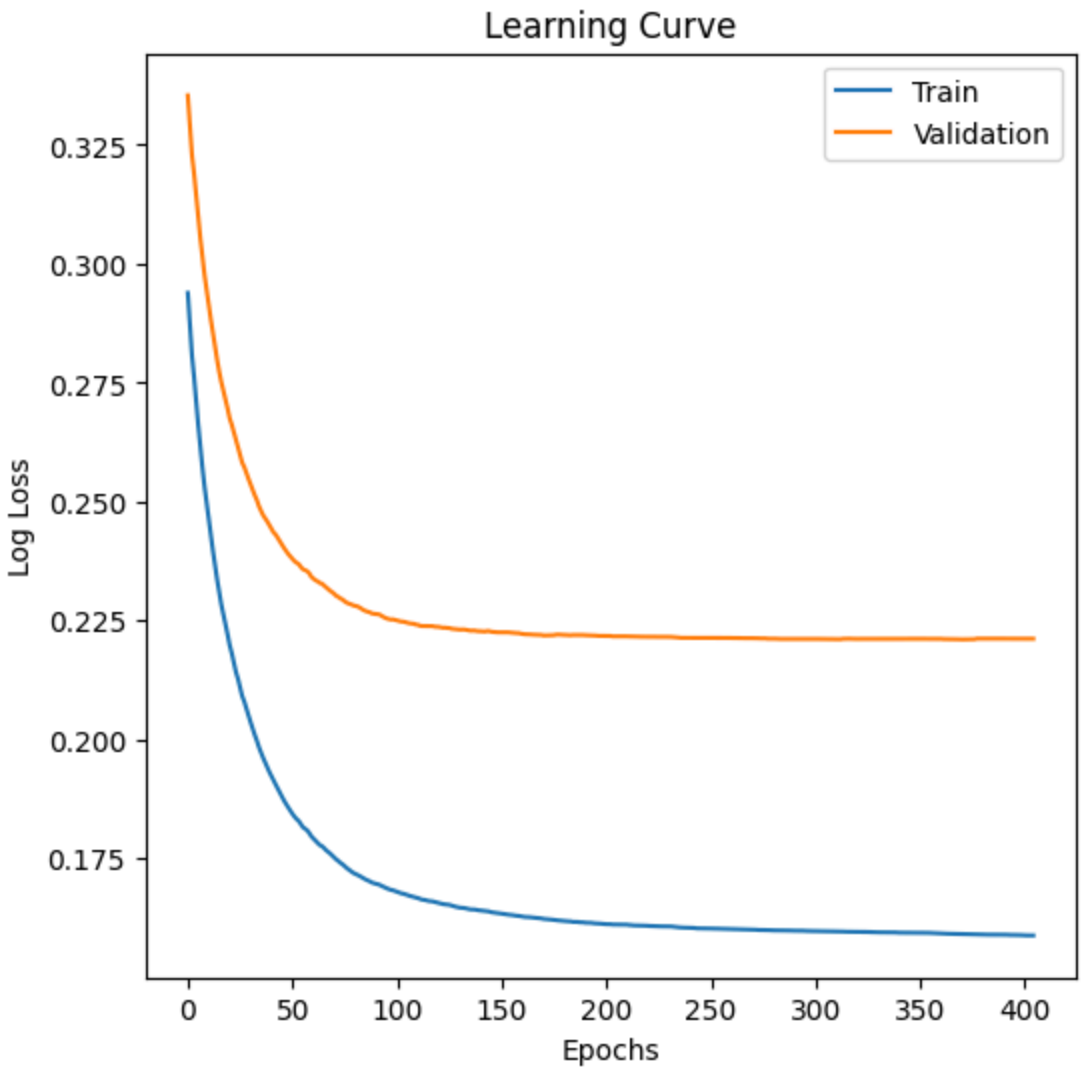
Evaluation

Evaluate the model using the ROC AUC metric, as it measures the model’s ability to distinguish between the two classes (retained vs. not retained). ROC AUC is particularly suitable for this task because it provides a threshold-independent measure of performance, focusing on the model's ranking capabilities rather than accuracy at a specific threshold.

During training, monitor the loss value on both training and validation sets to ensure proper model learning and to detect overfitting. Additionally, track the ROC AUC on the validation set to gauge how well the model generalizes to unseen data. The final evaluation on the test set provides an unbiased estimate of the model's performance in predicting customer retention. The following are the visualization graph of ROC AUC, learning curve and feature importance, respectively.

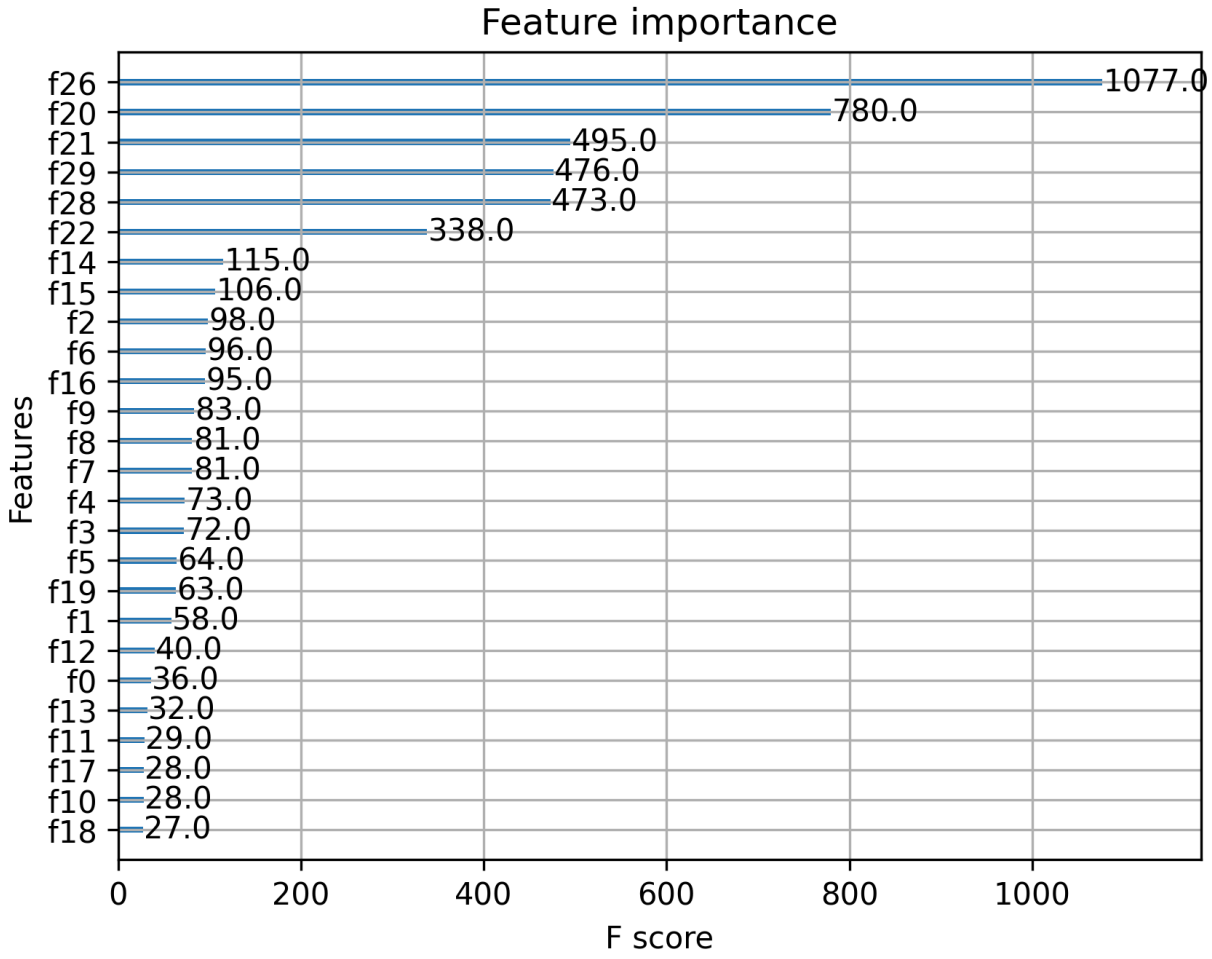


The ROC curve presented in the image provides a visual representation of the model's ability to differentiate between classes. With an AUC score of 0.881, the model demonstrates strong discriminatory power, effectively distinguishing between positive and negative instances. A random classifier would yield an AUC of 0.5, while a perfect model would achieve a score of 1.0. The current AUC, approaching 0.9, indicates that the model performs very well and is highly reliable. While the performance is strong, further refinements such as hyperparameter tuning, feature engineering, or addressing potential class imbalances could push the AUC even higher. This result suggests that the model is robust and well-suited for real-world applications, minimizing the risk of misclassification.



The learning curve illustrates the model's training progress, with both training and validation log loss decreasing over time. Initially, both curves exhibit a sharp decline, indicating that the model is quickly learning patterns in the data during the early epochs. As training progresses, the training loss continues to decrease steadily, while the validation loss plateaus at a higher value. The noticeable gap between the training and validation curves suggests a slight degree of overfitting, where the model performs better on the training data than on unseen validation data. To enhance generalization, techniques such as early stopping, dropout, or adjusting regularization parameters (e.g., L1/L2 penalties) could be considered. Overall, while the model learns effectively, further refinements may be necessary to optimize validation performance and

prevent overfitting.considered to fine-tune generalization.



The feature importance plot from the XGBoost model reveals the most influential factors driving predictions. The top-ranked feature, f26, has the highest F-score of 1077, indicating that it plays a crucial role in the model's decision-making process. Other highly significant features include f20 (F-score: 780), f21 (F-score: 495), and f29 (F-score: 476), suggesting that these features have strong predictive power. Notably, features such as f28 and f22 also contribute significantly, reinforcing their relevance in the classification task. The steep drop in F-score beyond the top six features suggests that the model relies heavily on a few key predictors while others contribute marginally. This insight can be leveraged to optimize the model by focusing on the most important features and potentially reducing dimensionality. Further analysis of the top-ranked features could help refine feature engineering strategies to enhance predictive performance.

Validation Set Metrics:

Accuracy:	F1:	AUC:	Confusion Matrix:
0.92	0.96	0.87	[[217 572] [45 6893]]

Test Set Metrics:

Accuracy:	F1:	AUC:	Confusion Matrix:
0.91	0.95	0.88	[[284 532] [58 5991]]

3. Time Usage Model

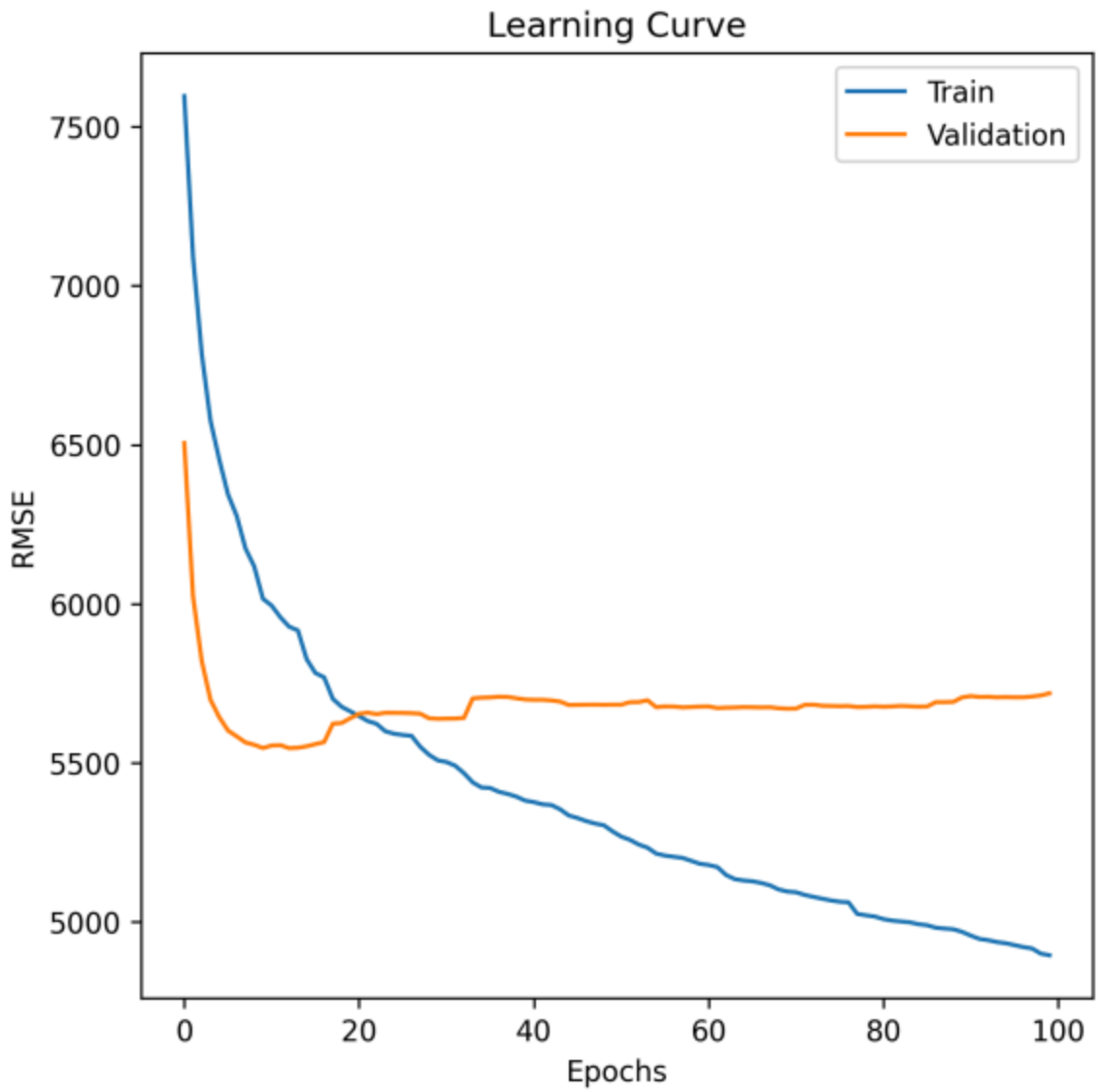
Dataset
We use the same dataset as the retention model but instead of the target variable 'returned_within_7_days' we use 'session_seconds' (which is actually total time per day after aggregating).

XGBoost Structure
The XGBoost regression model is used to predict time usage per day, leveraging gradient boosting to provide accurate and robust continuous value predictions. XGBoost is chosen for its ability to handle complex feature interactions and non-linear relationships, making it suitable for modeling time-based patterns in user behavior.

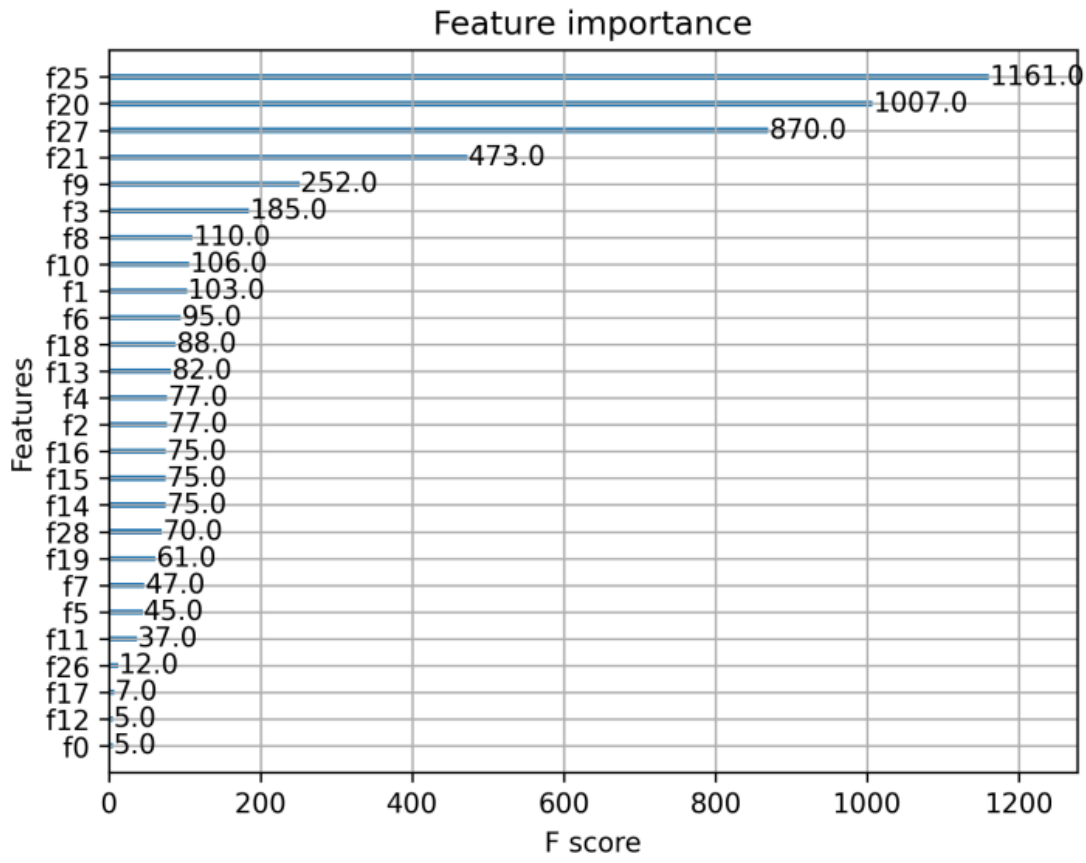
The model begins by taking the input features. It uses decision trees as base learners, refining predictions by minimizing the residuals from previous iterations. This boosting technique enables the model to capture intricate patterns within the data effectively.

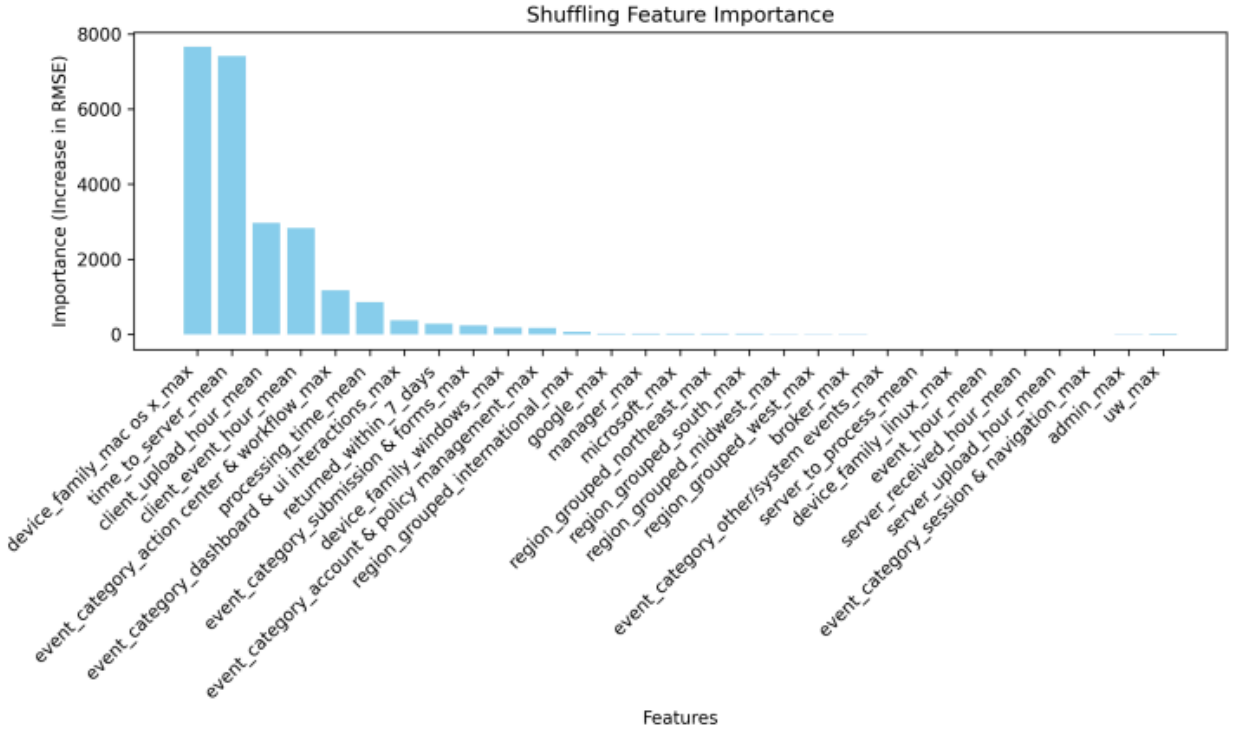
Key model parameters include the number of trees (estimators), maximum tree depth, learning rate, regularization terms, and subsampling ratios. These parameters control the model's complexity and its ability to generalize. Hyperparameter tuning is conducted to optimize these values, ensuring the best performance on the validation set.

Evaluation
Evaluate the model using Root Mean Squared Error (RMSE), which measures the average magnitude of the prediction errors. RMSE is chosen because it penalizes large errors more significantly than smaller ones, making it suitable for assessing the accuracy of time usage predictions.



The learning curve illustrates the Root Mean Squared Error (RMSE) for both the training and validation sets over 100 epochs. Initially, both curves experience a sharp decline, indicating that the model is effectively learning patterns in the data. However, as training progresses, the validation RMSE stabilizes around epoch 20-30, while the training RMSE continues to decrease. This divergence suggests that the model is starting to overfit, performing well on training data but not improving on unseen validation data. While the overfitting is not extreme, techniques such as early stopping, dropout, or L2 regularization could enhance generalization and prevent unnecessary complexity in the model.


Feature importance
Features
F score
f25 1161.0
f20 1007.0
f27 870.0
f21 473.0
f9 252.0
f3 185.0
f8 110.0
f10 106.0
f1 103.0
f6 95.0
f18 88.0
f13 82.0
f4 77.0
f2 77.0
f16 75.0
f15 75.0
f14 75.0
f28 70.0
f19 61.0
f7 47.0
f5 45.0
f11 37.0
f26 12.0
f17 7.0
f12 5.0
f0 5.0
0
200
400
600
800
1000
1200


Shuffling Feature Importance
Importance (Increase in RMSE)
Features
8000
6000
4000
2000
0
device_family_mac os x_max
time_to_server_mean
client_upload_hour_mean
client_event_hour_mean
event_category_action center & workflow_max
processing_time_mean
event_category_dashboard & ui interactions_max
returned_within_7_days
event_category_submission & forms_max
device_family_windows_max
event_category_account & policy management_max
region_grouped_international_max
google_max
manager_max
microsoft_max
region_grouped_northeast_max
region_grouped_south_max
region_grouped_midwest_max
region_grouped_west_max
broker_max
event_category_other/system events_max
server_to_process_mean
device_family_linux_max
event_hour_mean
server_received_hour_mean
server_upload_hour_mean
event_category_session & navigation_max
admin_max
uw_max

The feature importance plots for the XGBoost model highlight the most significant variables influencing its predictions. In the top plot, the most important feature is f25, followed closely by f27, f21, and f3, as indicated by their high F-scores. These features contribute significantly to the model's decision-making process, suggesting they capture crucial patterns in the data. The lower-ranked features, such as f12 and f0, have minimal impact, implying they add little predictive value.

The second plot, which evaluates feature importance based on RMSE increase after shuffling, reinforces these findings. The top contributing features, such as device_family_max, event_category_x_max, and session_time_mean, lead to a substantial increase in RMSE when shuffled, confirming their importance in model performance. Conversely, features toward the right of the plot have negligible effects when shuffled, indicating limited predictive power. These insights suggest that refining the model by focusing on key features and possibly removing less influential ones could enhance performance and reduce complexity.

Given the learning curve and feature importance, the model appears to perform reasonably well. The training and validation errors follow a decreasing trend, which means the model is learning from data effectively. However, since the validation RMSE does not decrease significantly after a certain point, it may indicate that further fine-tuning of hyperparameters is needed. Additionally, an analysis of residual errors could help determine whether certain features contribute to systematic biases in the predictions. By leveraging these insights, the model can be further optimized for better generalization and performance in real-world applications.

Test Set Metrics:

R2:	MSE:	MAE:
0.42	20462751.32	1837.64

Recommendation

A/B Testing Approach:

- **Control Group:** Standard feature presentation regardless of time
- **Treatment Group:** Time-optimized interface that prioritizes different features based on usage patterns by hour
- **Key Metrics:**
    - Primary: Average session_seconds_mean (as measured in our time usage model)

    - Secondary: Feature engagement rate, conversion rates on key actions
- **Duration:** 3 weeks with 50/50 split
- **Success Criteria:** 12% increase in average session duration during traditionally low-engagement hours

3. Predictive Next-Action Recommendations

Rationale: Our BERT4Rec model identifies likely next actions based on user behavior sequences. The model's top predictions can be leveraged to create a recommendation system that guides users toward high-retention actions.

A/B Testing Approach:

- **Control Group:** Standard interface without recommendations
- **Treatment Group:** Interface with contextual suggestions for next actions based on BERT4Rec predictions, prioritizing actions correlated with higher retention
- **Key Metrics:**
    - Primary: Action adoption rate (percentage of recommended actions taken)
    - Secondary: 7-day retention rate, session_seconds_mean
- **Duration:** 4 weeks with 60/40 split (treatment/control)
- **Success Criteria:** 20% adoption rate of recommended actions with 10% improvement in 7-day retention

Implementation Priority

Based on potential impact and implementation complexity, we recommend prioritizing these initiatives as follows:

1. **Role-Specific Engagement Optimization** - Highest potential impact on retention metrics for underrepresented user segments
2. **Predictive Next-Action Recommendations** - Leverages our existing BERT4Rec model for immediate implementation
3. **Time-of-Day Optimized Features** - Requires more complex implementation but addresses clear usage patterns

Measurement Framework

For each A/B test, we recommend implementing the following measurement approach:

1. **Pre-test Baseline:** Establish clear baseline metrics for all KPIs one month prior to testing
2. **Cohort Analysis:** Segment results by user roles, regions, and device types
3. **Significance Testing:** Implement rigorous statistical testing with a 95% confidence threshold

4. **Interaction Effects:** Monitor for unexpected interactions between tests if running concurrently

By systematically implementing and measuring these recommendations, Federato can leverage the insights from our models to improve retention rates and session duration while optimizing the overall user experience on the platform.