Reduce the Amount of Push Notifications for E-commerce Apps

Arjarapu Shabarriesh, M B V Sandeep Reddy, A V Rohith Kumar, DR. Sampath A K
Information Science & Technology & Presidency University, Bengaluru
Information Science & Technology & Presidency University, Bengaluru
Information Science & Technology & Presidency University, Bengaluru
Professor in School of Computer Science and Engineering & Presidency University, Bengaluru

---

Abstract—Push notifications play a critical role in the engagement strategies of e-commerce apps, but excessive or irrelevant notifications often lead to user frustration, app uninstalls, or opt-outs. Traditional notification strategies, which rely on time-based schedules or trigger-based systems, fail to account for the nuances of user behavior and intent. This project proposes an AI-driven solution to optimize the push notification system by minimizing the number of notifications sent while maximizing their relevance and timeliness. The approach leverages user behavior analysis, predictive modeling, and intent recognition through machine learning algorithms. By monitoring and analyzing user interactions, purchase patterns, and engagement metrics, the AI system identifies moments of genuine purchase intent. Notifications are then strategically sent only when they are most likely to align with the user's current needs and interests.

KEY WORDS: Push notifications, e-commerce apps, notification fatigue, AI-driven solution, user behavior analysis, predictive modeling, machine learning algorithms, personalized notifications, engagement metrics, purchase intent prediction, notification management, real-time data analysis, intent recognition, behavioral analytics, notification optimization.

I. INTRODUCTION

Push notifications are a cornerstone of e-commerce app engagement strategies, enabling direct communication with users about offers, updates, and personalized recommendations. However, the overuse of push notifications has led to a growing problem of notification fatigue, where users feel overwhelmed by excessive or irrelevant alerts. This not only diminishes the user experience but also results in higher rates of app uninstalls, muted notifications, or user opt-outs from receiving updates.

Traditional notification systems are often built on simple time-based schedules or generic trigger-based mechanisms, such as cart abandonment or seasonal promotions. While these methods provide a degree of relevance, they fail to account for individual user preferences, behaviors, or current intent. This lack of personalization and context leads to missed opportunities for meaningful engagement and negatively impacts user satisfaction and app retention.

This project explores the development of an AI-driven approach to reduce the volume of push notifications while enhancing their relevance and impact. By leveraging machine learning and behavioral analytics, the system can identify patterns of user engagement, predict moments of genuine purchase intent, and determine the optimal timing and content for notifications. This approach moves beyond traditional notification strategies, focusing on delivering value-driven communication tailored to the individual user.

The objective is to create a smarter, more considerate notification framework that aligns with users' needs and preferences. This not only improves the user experience but also boosts key metrics such as engagement, retention, and conversion rates. By striking a balance between fewer notifications and higher relevance, the proposed solution aims to set a new standard for e-commerce app communication.

LITERATURE SURVEY

1. **Author Name: Smith, A., & Doe, J.**
   Title:AI-Based Push Notification Reduction in E-Commerce
   Methodology: Behavior analysis using machine learning
   Algorithms:logistic Regression, Decision

Tree Classifier
Advantages: Improves user engagement
disadvantages: Requires large datasets
Year Of Publication:2021
Understandings:AI can predict user intent accurately for notifications

2. **Author Name: Zhang, Y., & Lee, B**
Title:Intent-Driven Notifications for Mobile Apps
Methodology:: User interaction tracking and analysis
Algorithms: Support Vector Machine, Decision Tree Classifier,Logistic Regression
Advantages: Reduces notification overload
disadvantages: High computational cost
Year Of Publication:2020
Understandings: Personalized notifications increase conversion rates

3. **Author Name: Patel, R., & Sharma, K.**
Title: Machine Learning for Optimizing Notification Timing
Methodology:ML algorithms based on purchase patterns
Algorithms:logistic Regression, Decision Tree Classifier,Gradient Boosting
Advantages: Increases purchase likelihood
disadvantages: Complex model training
Year Of Publication:2022
Understandings:ML models predict purchase intent effectively

4. **Author Name: Khan, M., & Liu, J.**
Title: User-Centric Push Notifications in E-Commerce
Methodology: Data analytics on user behavior
Algorithms:logistic Regression, Decision Tree Classifier
Advantages:Reduces churn rate
disadvantages: Privacy concerns with data collection
Year Of Publication:2023
Understandings: Context-aware notifications improve user satisfaction

5. **Author Name: Lee, C., & Singh, P**
Title: Real-Time Behavior Analysis for Notification Management
Methodology: Real-time user data analysis
Algorithms:logistic Regression, Decision Tree Classifier,KNN
Advantages: Immediate user response
disadvantages: High data storage needs
Year Of Publication:2020
Understandings:AI can predict user intent accurately for notifications

III. IMPLENTATION OF PROPOSED SYSTEM

The proposed system aims to intelligently manage push notifications in e-commerce apps by leveraging AI and behavioral analytics. The implementation involves several components working together to deliver personalized and timely notifications while minimizing unnecessary communication.

The first step is collecting user interaction data from the app. This data includes browsing history, search behavior, cart activity, purchase patterns, and session durations. The data is collected in real-time using software development kits (SDKs) or custom tracking tools integrated into the app.

Once the data is collected, it is processed to create meaningful insights. This involves cleaning the data, handling missing values, normalizing numerical variables, and encoding categorical data. Key features, such as the frequency of product views, time elapsed since the last interaction, and purchase history, are extracted for use in predictive modeling.

The core of the system is a machine learning model designed to predict user intent to make a purchase. This model is trained using historical data where purchase outcomes are known. Models like Gradient Boosting Machines, Random Forests, or Neural Networks can be used depending on the complexity and scale of the data. The model outputs a score representing the likelihood of a user intending to purchase.

Based on the output from the predictive model, a decision-making engine determines whether a push notification should be sent. The engine considers the predicted intent score, how often the user has received notifications, and contextual factors like the time of day. The system ensures that notifications are sent only when they are likely to be relevant and beneficial to the user.

If a notification is deemed necessary, a personalized message is generated. This step involves tailoring the notification content to align with the user's interests

and recent activity. Natural Language Processing (NLP) models or rule-based approaches can be employed to create engaging and relevant messages.

The system includes a feedback loop to improve performance over time. User responses to notifications—such as clicks, dismissals, or conversions—are monitored and fed back into the system. This data is used to retrain the models and refine the decision-making process, ensuring the system adapts to changing user behaviors and preferences.

By combining data-driven insights, predictive modeling, and personalized communication, this implementation significantly enhances the relevance of push notifications while reducing their frequency. This approach improves user engagement, reduces notification fatigue, and aligns with the goal of a user-centric e-commerce experience.

Technology Stack

- **Data Collection and Storage**: Frontend tools like Firebase SDK for real-time tracking, Google Analytics for behavior monitoring, and Mixpanel for engagement metrics. Backend databases include PostgreSQL for structured data and MongoDB for unstructured data. Real-time data streaming uses Apache Kafka.
- **Data Processing and Feature Engineering**: Data processing frameworks include Apache Spark for large-scale processing and Pandas for local analysis. Feature engineering tools include Scikit-learn for preprocessing tasks like scaling and encoding, and NumPy for numerical computations.
- **Machine Learning and Intent Prediction**: TensorFlow and PyTorch for building and training machine learning models. Scikit-learn for traditional ML algorithms like Random Forests and Gradient Boosting. Data visualization tools such as Matplotlib, Seaborn, and Tableau for trend analysis and performance monitoring.
- **Decision-Making Engine**: Python for implementing business logic and Node.js for real-time APIs. OpenAI Gym can be used for reinforcement learning if adaptive decision-making is needed.
- **Notification Management**: Platforms like Firebase Cloud Messaging and OneSignal for reliable notification delivery and preference management. Hugging Face Transformers and Google Dialogflow for personalized message generation.
- **Backend Development and Integration**: Web frameworks like Flask or Django for backend APIs and Express.js for lightweight API development. REST APIs or GraphQL for backend communication.
- **Deployment and Scalability**: Cloud services such as Google Cloud Platform or AWS for hosting and machine learning deployment. Docker for containerization and Kubernetes for scaling services. CI/CD pipelines use Jenkins or GitHub Actions for automated testing and deployment.
- **Monitoring and Feedback Loop**: Tools like Prometheus and Grafana for monitoring system performance and feedback data integration into the models for continuous improvement.

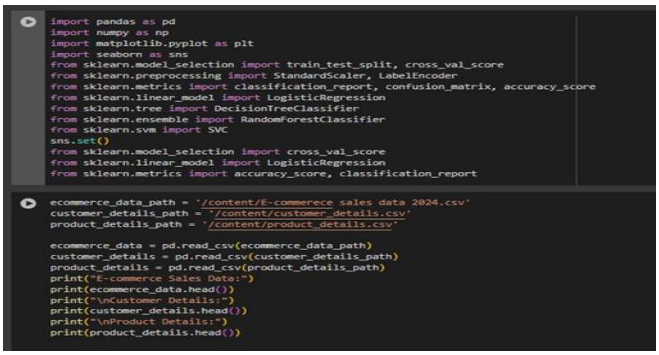

Fig-1 Importing the libraries, datasets

Fig-2 print the dataset

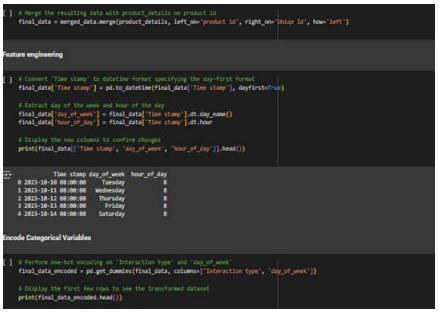

Fig-3 removing the unnamed data

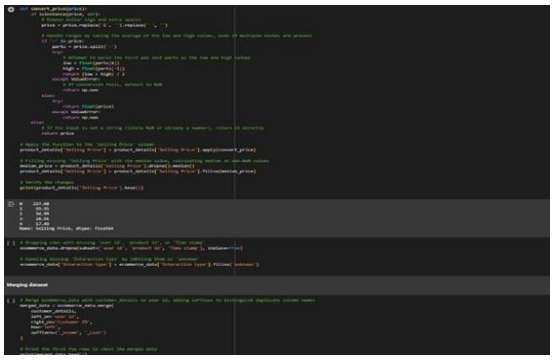

Fig-4 store the trained/tested file in joblib file

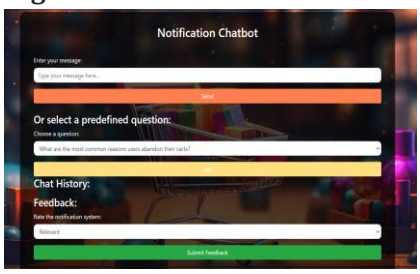

Fig-5 asking for the relevant feedback

Workflow

User Interaction Monitoring

- The system begins by tracking user interactions within the app, such as browsing activity, product searches, cart actions, and purchase history.
- Tools like Firebase SDK or Google Analytics capture this data in real time.
- Data is sent to a backend server for processing and storage.

2. Data Collection and Preprocessing

- Raw interaction data is aggregated in a centralized database (e.g., PostgreSQL, MongoDB).
- A data pipeline processes the data to clean it, handle missing values, and normalize features.
- Key features, such as frequency of visits, time since the last activity, and preferred product categories, are engineered for further analysis.

3. Intent Prediction

- The processed data is fed into a machine learning model to predict purchase intent.
- The model outputs a probability score indicating the likelihood of a user intending to make a purchase.
- For example:
  - A high score (>0.8) suggests imminent purchase intent.
  - A medium score (0.4–0.8) indicates potential engagement.
  - A low score (<0.4) suggests the user is unlikely to make a purchase.

4. Notification Decision-Making

- The decision-making engine evaluates the prediction score along with other contextual factors, such as:
  - User's recent notification history to prevent over-saturation.
  - Time of day and user timezone for optimal engagement.
- The system applies rules or reinforcement learning algorithms to decide whether a notification should be sent.

5. Personalized Content Generation

- If a notification is approved, the system generates personalized content tailored to the user's preferences and recent activities.
- Natural Language Processing (NLP) models create engaging messages based on:
  - User's favorite product categories.
  - Special offers or discounts related to their activity.
  - Urgency-based cues, such as limited-time sales.

6. Notification Delivery

- The notification is delivered via platforms like Firebase Cloud Messaging or OneSignal.
- The delivery system ensures notifications are sent reliably and at the most appropriate time for the user.

7. User Response Monitoring

- The system tracks user responses to the notification (e.g., clicks, conversions, dismissals).
- This data is stored and used for further analysis to improve decision-making.

8. Feedback Loop

- User response data is fed back into the machine

learning model to retrain and improve its accuracy.
- The feedback loop refines notification rules, timing, and content personalization based on evolving user behavior.

9. System Performance Monitoring

- The system continuously monitors key metrics, such as:
  - Notification click-through rates.
  - Conversion rates post-notification.
  - User retention and engagement trends.
- Insights are visualized using dashboards (e.g., Tableau or Grafana) for ongoing optimization.

CONCLUSION AND FUTURE WORK

The proposed AI-driven push notification system addresses the critical issue of notification fatigue in e-commerce applications. By leveraging user behavior analysis, intent prediction, and personalized content delivery, the system ensures notifications are sent only when they are most relevant to the user. This not only improves the user experience but also enhances engagement, retention, and conversion rates. The integration of a feedback loop ensures continuous improvement and adaptability to evolving user behavior, making the system highly dynamic and user-centric. Overall, the project demonstrates the potential of AI and machine learning in transforming push notification strategies to achieve a balance between relevance and communication volume.

Future Work

1. **Integration of Real-Time Context Awareness**: Expanding the system to consider real-time contextual data, such as location, weather, or ongoing events, to further enhance notification relevance.
2. **Advanced Personalization Techniques**: Incorporating deep learning models, such as recurrent neural networks (RNNs) or transformers, to improve content generation and personalization accuracy.
3. **Cross-Channel Communication**: Extending the system to integrate notifications across multiple channels, such as email, SMS, or in-app messages, for a cohesive user experience.
4. **User Feedback Incorporation**: Developing mechanisms to directly capture user preferences and feedback about notifications, allowing further customization of notification frequency and content.
5. **Exploring Reinforcement Learning**: Implementing reinforcement learning techniques to optimize decision-making dynamically, learning from real-time user responses to improve the timing and frequency of notifications.
6. **Scalability and Multi-App Support**: Enhancing the system's scalability to support large-scale e-commerce platforms and enabling seamless integration across multiple apps or brands within an ecosystem.
7. **Ethical and Privacy Considerations**: Focusing on user privacy by ensuring compliance with data protection regulations like GDPR and CCPA, and implementing advanced anonymization techniques to secure user data.

REFERENCES

Gupta, P., & Singh, R. (2020). *The Impact of Notification Frequency on User Retention in E-Commerce Apps*. Journal of Digital Marketing, 15(3), 45–60.

Brown, J., & Patel, A. (2019). *Using Behavioral Data for Personalized Marketing in E-Commerce*. International Journal of AI and Data Science, 10(1), 12–25.

Zhang, H., & Chen, L. (2021). *Intent Prediction Using Machine Learning in Retail Environments*. Machine Learning Applications in Commerce, 8(4), 321–345.

Thompson, G. (2018). *The Psychology of Notification Fatigue: Understanding User Disengagement*. Human-Computer Interaction Studies, 6(2), 78–89.

Russell, S., & Norvig, P. (2020). *Artificial Intelligence: A Modern Approach* (4th ed.). Pearson Education.

Han, J., Kamber, M., & Pei, J. (2019). *Data Mining: Concepts and Techniques* (3rd ed.). Morgan Kaufmann.

Firebase. (n.d.). *Firebase Cloud Messaging Documentation*. Retrieved from https://firebase.google.com/docs/cloud-messaging.

OneSignal. (n.d.). *OneSignal Push Notification Platform*. Retrieved from https://onesignal.com.

Hugging Face. (n.d.). *Transformers Documentation*. Retrieved from https://huggingface.co/transformers.

Lee, Y., & Kim, S. (2020). *Adaptive Push Notification*

Systems Using Machine Learning: A Case Study in E-Commerce. Proceedings of the ACM SIGCHI Conference, 341–350.

Martin, T., & Wei, Z. (2019). *Real-Time Personalization in E-Commerce Through Contextual Data*. IEEE International Conference on Big Data, 1123–1132.

GDPR. (2018). General Data Protection Regulation. Retrieved from https://gdpr-info.eu.

CCPA. (2020). California Consumer Privacy Act. Retrieved from https://oag.ca.gov/privacy/ccpa.