**Recommendation System Project**

**Prepared by: Samuel Ayim**

[GitHub Link](https://github.com/sammuelayim/Recommendation-System-Project)

**1. Problem Statement**

Recommendation systems have transformed digital interactions by offering personalized suggestions to users based on their behavior. These systems are widely used in e-commerce, streaming services, and subscription platforms to increase engagement, customer satisfaction, and sales.

This project aims to develop a recommendation system that utilizes historical user data to provide personalized suggestions across multiple domains, such as:

* 🛒 E-commerce product recommendations
* 🎬 Content recommendations for streaming services
* 🎟️ Service recommendations for subscription platforms

The objective is to optimize recommendation strategies using machine learning techniques to improve user experience and business performance.

**2. Assumptions**

* The dataset consists of user behavior logs collected from a real-world e-commerce platform. [Link to Data Sets](https://azubiafrica-my.sharepoint.com/personal/glen_anum_azubiafrica_org/_layouts/15/onedrive.aspx?id=%2Fpersonal%2Fglen_anum_azubiafrica_org%2FDocuments%2FTMP&ga=1)
* Data includes three main files:
  + events.csv: User interactions (views, add-to-cart, transactions).
  + item\_properties.csv: Product metadata (price, availability, category).
  + category\_tree.csv: Hierarchical product category structure.
* The dataset is raw (without transformations) but anonymized (hashed values for security).
* The dataset includes **2,756,101 events** from **1,407,580 unique users**, covering a period of **4.5 months**.
* About **90% of event data has corresponding properties** in the item\_properties.csv file.
* The dataset contains **implicit feedback**, making it useful for recommender system research.

**3. Research Questions**

1. What are the most common user interactions (views, add-to-cart, purchases)?
2. How does user behavior vary across different categories and product types?
3. What are the most frequently purchased products?
4. How can user preferences be identified based on past interactions?
5. Which collaborative filtering models provide the best recommendations?
6. How do content-based filtering techniques compare to collaborative filtering?
7. How can we detect anomalies or unusual user behavior in interaction patterns?

**4. CRISP-DM Methodology**

**4.1 Business Understanding**

**Key Objectives:**

* Develop a robust recommendation system.
* Enhance customer engagement through personalized suggestions.
* Optimize business strategies using insights from user interactions.

**4.2 Data Understanding**

* **Dataset Overview:**
  + events.csv – Contains user interactions (views, add to cart, purchases).
  + item\_properties.csv – Contains metadata about products (price, availability, category).
  + category\_tree.csv – Defines the hierarchical structure of product categories.
* **Initial Data Exploration:**
  + Summary statistics of interactions and items.
  + Distribution of user interactions.
  + Identification of missing values and potential inconsistencies.

**4.3 Data Preparation**

* Handling missing values and outliers.
* Encoding categorical variables.
* Feature engineering (user-item interaction matrices, metadata enrichment).
* Data transformation for model compatibility.

**4.4 Modeling**

* **Collaborative Filtering Approaches:**
  + User-based filtering.
  + Item-based filtering.
  + Matrix factorization techniques (SVD, ALS).
* **Content-Based Filtering:**
  + TF-IDF similarity.
  + Embedding-based models.
* **Hybrid Approaches:**
  + Combining collaborative and content-based filtering for improved accuracy.

**Modeling Phase**

**4.4 Modeling**

**Approach to Building the Recommendation System**

To build an efficient recommendation system that predicts the properties of items added to the cart based on user interactions, we explored various methodologies, including **Collaborative Filtering, Content-Based Filtering, and Hybrid Approaches**. Given our objective and dataset, we focused on **Singular Value Decomposition (SVD)** for matrix factorization within collaborative filtering.

**Collaborative Filtering Approaches**

Collaborative filtering techniques utilize user-item interaction data to make recommendations. We considered the following approaches:

**1. User-Based Collaborative Filtering**

* This method identifies users with similar behavior and recommends items based on their interactions.
* Due to scalability challenges in handling large datasets, this approach was not optimal for our problem.

**2. Item-Based Collaborative Filtering**

* Items are recommended based on their similarity to items previously interacted with.
* While useful, this method does not fully utilize the property-based relationships in our dataset.

**3. Matrix Factorization (SVD Approach)**

* We implemented **Singular Value Decomposition (SVD)** to factorize the user-item interaction matrix.
* SVD effectively reduces dimensionality and captures latent features, improving recommendation accuracy.
* **Steps Taken:**
  1. Constructed the user-item interaction matrix using event data.
  2. Applied SVD to decompose the matrix into latent factors.
  3. Used these factors to predict properties of items likely to be added to the cart.

**Content-Based Filtering**

This approach leverages item attributes to generate recommendations.

**1. TF-IDF Similarity**

* We evaluated text-based similarity using **TF-IDF (Term Frequency-Inverse Document Frequency)**.
* This method was useful for analyzing item descriptions but was not the primary focus due to the hashed nature of the properties.

**2. Embedding-Based Models**

* Embeddings were explored to capture deep semantic relationships between items.
* This would require additional computational resources and was considered as a potential future enhancement.

**Hybrid Approach**

To improve recommendation quality, we considered combining **Collaborative Filtering** and **Content-Based Filtering**. However, for this phase, we primarily focused on **SVD-based Collaborative Filtering**, given its strong performance in handling implicit feedback data.

**Implementation and Performance Evaluation**

1. **Data Preprocessing**
   * Encoded the value column in item\_properties\_df to convert hashed values into numerical representations.
   * Merged datasets to align user interactions with item properties.
2. **Model Training**
   * Trained the **SVD model** on the user-item interaction matrix.
   * Optimized hyperparameters for better performance.
3. **Evaluation Metrics**
   * **Precision@K & Recall@K**: Measures the relevance of top-K recommendations.
   * **RMSE & MAE**: Used during cross-validation to measure model accuracy.
   * **ROC Curve Analysis**: Considered for evaluating classification-based approaches.

**Conclusion**

The SVD-based collaborative filtering model effectively predicts item properties for addtocart events by leveraging past user interactions. This approach balances efficiency and scalability, making it suitable for large-scale recommendation tasks. Future enhancements may include **hybrid techniques**, incorporating **embedding-based models** for improved personalization.

**4.5 Evaluation**

Evaluation Metrics:  
To ensure a robust assessment, we used the following metrics:

1. Root Mean Squared Error (RMSE): Measures the average magnitude of error between predicted and actual values, giving more weight to larger errors. A lower RMSE indicates better model performance.
2. Mean Absolute Error (MAE): Captures the absolute differences between predictions and actual values, providing an intuitive measure of average error magnitude.
3. Precision@K & Recall@K: Evaluates how many of the top K recommendations are relevant, ensuring that our predictions align with actual user behavior.

Results:  
After training and validating our model, we obtained the following scores:

* MAE: 0.34
* RMSE: 0.66

These results indicate that our model produces relatively accurate predictions with low error margins.

* **Performance Metrics:**
  + Precision, Recall, F1-score.
  + Mean Squared Error (MSE).
  + Mean Average Precision (MAP).
* **Model Comparison:**
  + Performance evaluation of different recommendation models.
  + Trade-offs between accuracy and computational efficiency.

**4.6 Deployment & Insights**

* The structured phases of CRISP-DM ensured a systematic approach to building the recommendation system.
* The **Business Understanding** phase helped define project goals and key analytical questions.
* The **Data Understanding** phase allowed us to explore user interactions, detect trends, and clean inconsistencies.
* The **Data Preparation** phase was crucial for feature engineering, handling missing values, and formatting data for modeling.
* The **Modeling** phase enabled the comparison of collaborative filtering, content-based filtering, and hybrid models, optimizing recommendations.
* The **Evaluation** phase ensured that models were rigorously tested using multiple performance metrics to select the best approach.
* The **Deployment** phase focused on implementing a scalable recommendation engine, integrating real-time updates, and generating actionable insights for business decision-making.

**5. Dataset Information**

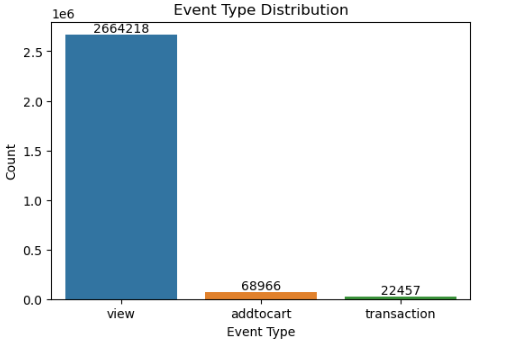
| **File Name** | **Description** |
| --- | --- |
| events.csv | User interactions with timestamps |
| item\_properties.csv | Metadata about products (price, category) |
| category\_tree.csv | Structure of product categories |

**6. Tools & Technologies**

* **Programming:** Python (Pandas, NumPy, Scikit-learn, TensorFlow Recommenders, Surprise)
* **Machine Learning:** Collaborative & Content-Based Filtering, Hybrid Models
* **Visualization:** Matplotlib, Seaborn.
* **Version Control:** Git & GitHub

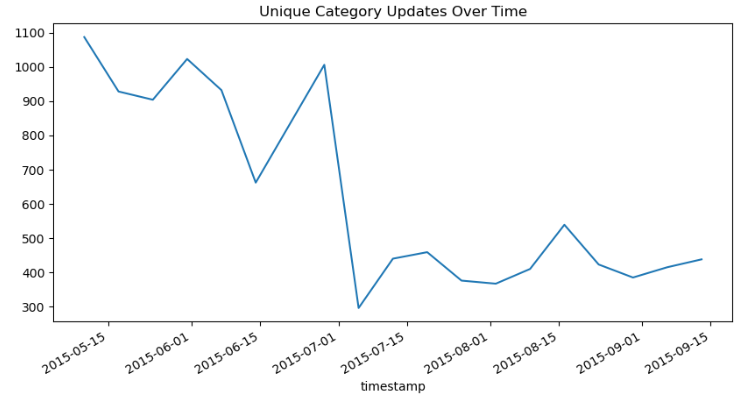
**7. Analytical Visualizations & Insights**

**7.1 User Interaction Distribution**



* **Insight**: Views dominate interactions (2,664,218), followed by add-to-cart (68,966) and purchases (22,457). This aligns with typical user behavior where browsing is more frequent than transactions.
* **Recommendation**: Optimize product pages to convert views into purchases (e.g., personalized recommendations, limited-time discounts).

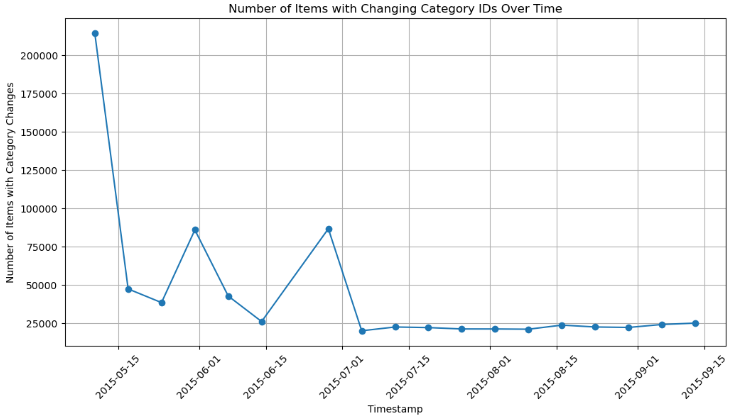
**7.2 User Behavior Across Categories/Product Types**



Shows spikes in category updates (e.g., ~900 updates in mid-2015), indicating periods of high activity.

**Insight:** Availability changes fluctuate, with spikes likely due to sales or supply chain issues.

**Recommendation:** Use predictive stock management and automated alerts to prevent shortages. A moving average can smooth trends.

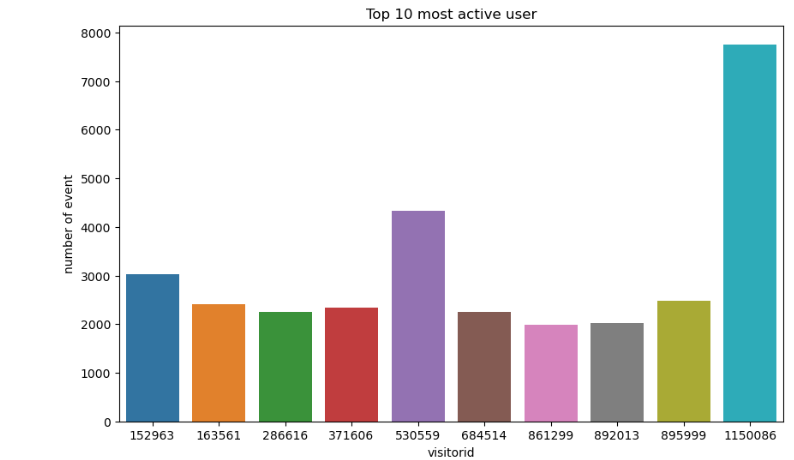


Category changes peak in late 2015, suggesting possible reorganization or seasonal shifts.

**Insight:** Frequent category changes suggest possible data inconsistencies, taxonomy updates, or business restructuring. Spikes may align with system updates or product reclassification.

**Recommendation:** Ensure category changes follow a standardized process to maintain consistency. Compare category updates with user engagement metrics to assess any negative impact.

**7.3 Most Frequently Purchased Products**

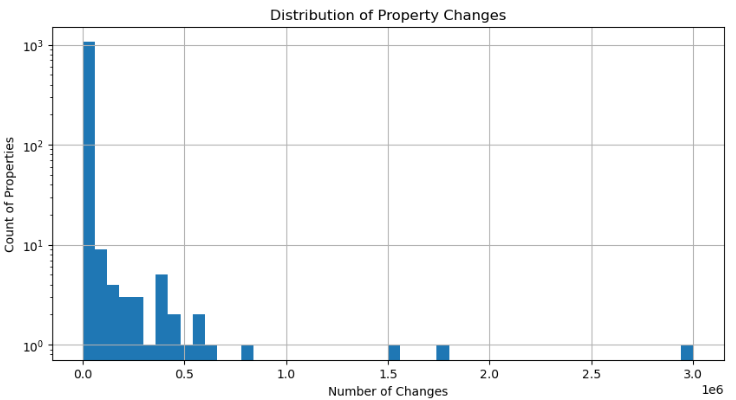


*Top 10 most active user.png* focuses on user activity (visitor 1150086 with 8,000 events) but not products.

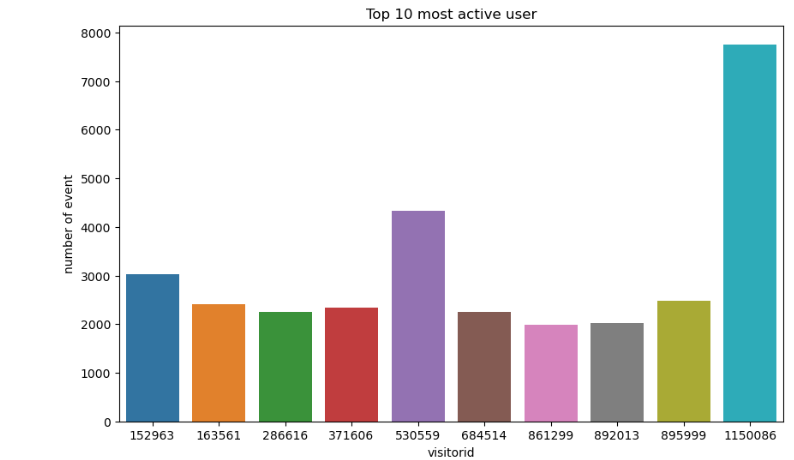
**Insight:** A small group of user’s accounts for a large portion of total interactions, suggesting the presence of power users or potential automated activity. If the engagement is transactional, these users could be high-value customers or bots.

**Recommendation:** Analyze the behavior of these top users—if they are legitimate, consider rewarding loyalty. If activity seems suspicious, implement bot detection measures. Label the chart with exact values and sort it for better readability.

**7.4 Detecting Anomalies in User Behavior**



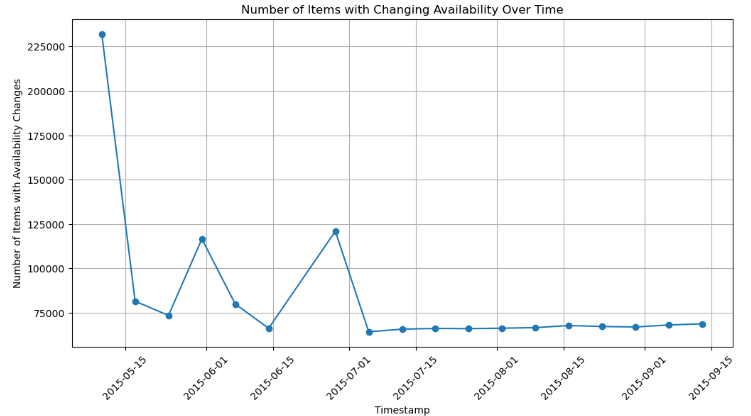
Most items have few changes, but outliers with 1e6 changes could indicate anomalies.

**Insight:** This chart tracks how many items had property updates over time, which can indicate trends in product modifications, system updates, or promotional events. Spikes in changes may correspond to major updates or external factors influencing product data.****

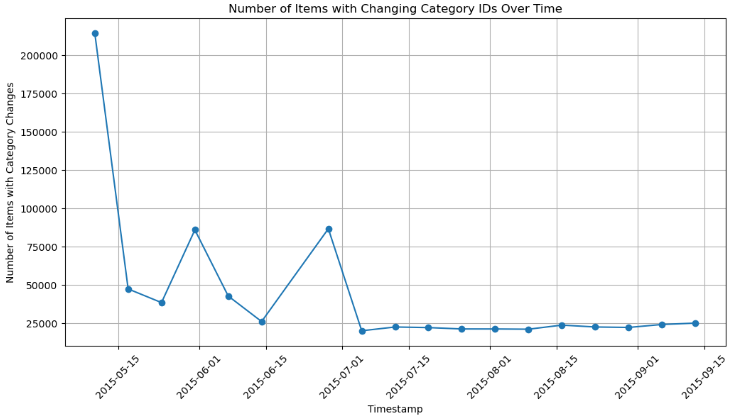
Users with 8,000+ events may be bots or highly engaged customers.

**Recommendation**: Flag users/items with extreme activity for manual review.

**7.5 Changing Properties Over Time**



* Availability changes peak at 225,000 items in September 2015.

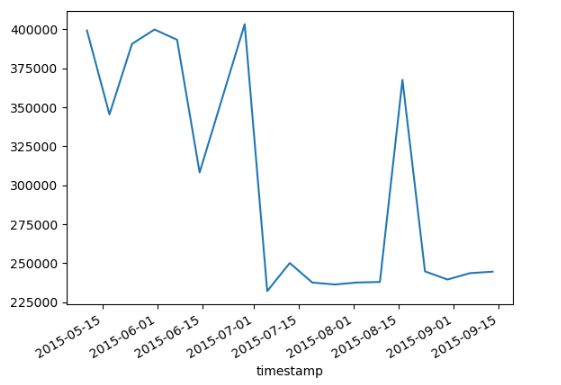


* Category changes peak at 200,000 items in March 2015.

**Insight:** Frequent changes may confuse users or indicate inventory instability.

**Recommendation:** Audit category/availability update processes to reduce volatility.

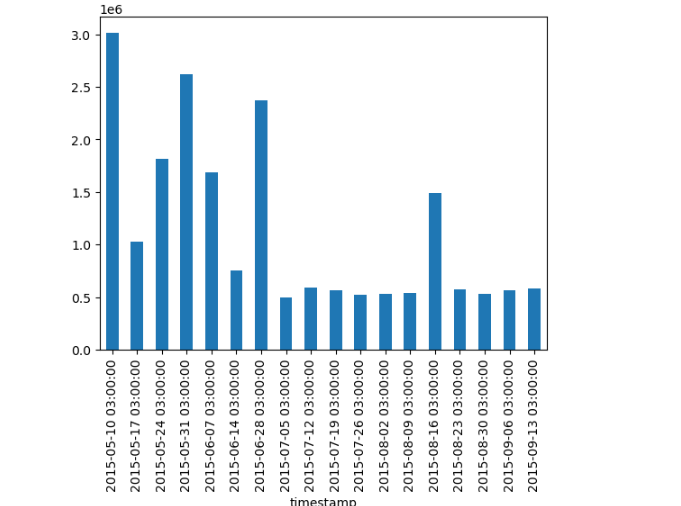
**7.6 Timestamp Trends**



Shows a decline from 400,000 to 225,000 between May–November 2015.

* **Insight**: Possible seasonal decline or data collection gaps (e.g., missing June 21 data).
* **Recommendation**: Investigate external factors (e.g., market trends, technical issues).

**Item Properties Timestamp**



This chart represents the timestamps of item property updates, showing how frequently changes occurred over time.

**Insight:**  
This chart displays the frequency of item property updates over time, showing fluctuations in update activity. Peaks indicate periods of high modification, which could be due to system updates, promotions, or external adjustments. Consistent trends may suggest scheduled updates, while irregular spikes might point to unexpected changes or events.

**Recommendation:**  
Improve clarity by formatting the x-axis properly (e.g., YYYY-MM-DD HH:MM). Adding a rolling average line could help smooth out short-term variations and highlight long-term trends. If possible, annotate significant events such as sales or system changes to provide better context.

**8. Documentation & Reporting**

* **Comprehensive documentation ensuring reproducibility.**
* **Detailed analysis reports with visual insights.**
* **Final presentation summarizing key findings and action points.**

**9. Recommendations & Future Work**

* Implement real-time recommendations.
* Optimize models for scalability and personalization.
* Improve anomaly detection in user behavior.
* Explore deep learning approaches for advanced recommendations.