Recommendation Engine based on Customer Behavior Data

Project By Kaustubh Pande

Introduction

The purpose of this project is to develop a recommendation engine for an e-commerce store using behavioral data from a multi-category store. The dataset, available at [https://www.kaggle.com/datasets/mkechinov/ecommerce-behavior-data-from-multi-category-store], contains valuable information about customer behavior, including purchase history and product categories. By analyzing the features like "event_type" in the data, insights into customer behavior can be gained.

The main objective of the recommendation engine is to enhance the personalized shopping experience for customers. With the increasing prevalence of e-commerce, customers now expect tailored recommendations that meet their specific needs. This project aims to meet those expectations by improving customer satisfaction through personalized recommendations. By understanding the events that drive customer behavior, the recommendation engine can provide more accurate and relevant recommendations.

The problem this project addresses is the need for a recommendation system that can provide personalized recommendations based on customer behavior. By developing a recommendation engine, the e-commerce store can better understand customer preferences and deliver tailored recommendations, leading to improved customer satisfaction.

In summary, this project focuses on developing a recommendation engine for an e-commerce store using behavioral data. The aim is to enhance the personalized shopping experience for customers by analyzing the data to gain insights into customer behavior.

Objectives

Technical project objectives:

1. Develop a recommendation engine for an e-commerce store using Hybrid Collaborative Filtering.
2. Evaluate the performance of the recommendation engine

Individual learning objectives:

1. Gain hands-on experience in developing a recommendation engine
2. Develop skills in data analysis and interpretation
3. Improve understanding of the impact of recommendation systems on customer satisfaction.

Data

Ingestion

The data used in this project is the "ecommerce-behavior-data-from-multi-category-store" dataset, obtained from Kaggle. The data set contains over 1 million rows and contains information such as customer ID, event type, product ID, and timestamp. The dataset was stored in a local directory, and cloud storage services were used to backup the data.

Some challenges while obtaining the data included, the size of the dataset, which contained over 1 million rows and 9 columns, requiring significant storage capacity and processing power. Additionally, there were missing values, duplicates, and inconsistencies, which were addressed through data cleaning and preprocessing techniques.

Overall, the data was successfully obtained, stored and overcome with the obstacles through independent effort and utilizing available resources.

Exploration

The dataset for e-commerce behavior collected from a multi-category store is quite expansive, containing more than 1 million rows and 9 columns. The data present in this dataset helps in comprehending user behavior, product categories, events, and timestamps.
The dataset has the following primary characteristics:

- Size: The dataset is comprehensive and extensive, including over 1 million rows and 9 columns.
- Data types: The dataset is an amalgamation of both numerical and categorical data.
- Features: The dataset consists of nine columns that furnish valuable information about user behavior, product categories, events, and timestamps.
- Missing values: The dataset may have missing values, necessitating data cleaning and preprocessing to fill in the gaps.
- Duplicates: The dataset may contain duplicate rows that could skew analysis results, and it is crucial to detect and remove them.
- Time series data: The dataset contains a wealth of timestamps, enabling the analysis of user behavior trends over time.
- Multi-category store: The dataset comprises information about numerous product categories, enabling the analysis of consumer behavior across multiple categories.

Features of the dataset:

- event_time: The time stamp of the event when it occurred.
- event_type: The type of event that occurred, such as 'view', 'cart', 'purchase'

- product_id: The unique identifier for the product that was viewed, added to the cart, or purchased.
- category_id: The unique identifier for the category that the product belongs to.
- category_code: The hierarchical category structure of the product, if available. For example, "electronics.smartphone" indicates that the product is a smartphone in the electronics category.
- brand: The brand of the product.
- price: The price of the product.
- user_id: The unique identifier for the user who performed the event.
- user_session: The unique identifier for the session that the event occurred in.
- date: The date when the event occurred.
- day_of_week: The day of the week when the event occurred.
- hour_of_day: The hour of the day when the event occurred.

Scope:
The scope of this project encompasses the development and evaluation of a recommendation engine for an e-commerce store using Filtering methods. The project involves utilizing historical data from the store to train the recommendation engine and generate personalized recommendations for users. The evaluation of the recommendation engine's performance is conducted using precision and recall metrics. The project aims to provide insights into the feasibility and effectiveness of the recommendation engine in improving customer satisfaction and increasing sales.

Limitations:
1. Historical Data: The recommendation engine is built using historical data, which may not fully capture real-time user preferences and trends. The engine's performance is based on historical patterns and behaviors, and its effectiveness may vary when applied to real-time user interactions.

2. Deployment and Reevaluation: The project does not include the deployment of the recommendation engine in a live environment with real customers. Truly testing the engine's performance would require deployment and subsequent reevaluation based on user interactions and feedback. Live deployment could reveal additional challenges and provide a more accurate assessment of the engine's effectiveness in real-world scenarios.

4. Generalizability: The findings and conclusions drawn from this project may not be directly applicable to all e-commerce stores or industries. The performance of the recommendation engine may vary depending on factors such as the specific domain, product categories, and user preferences.

Plots Generated:

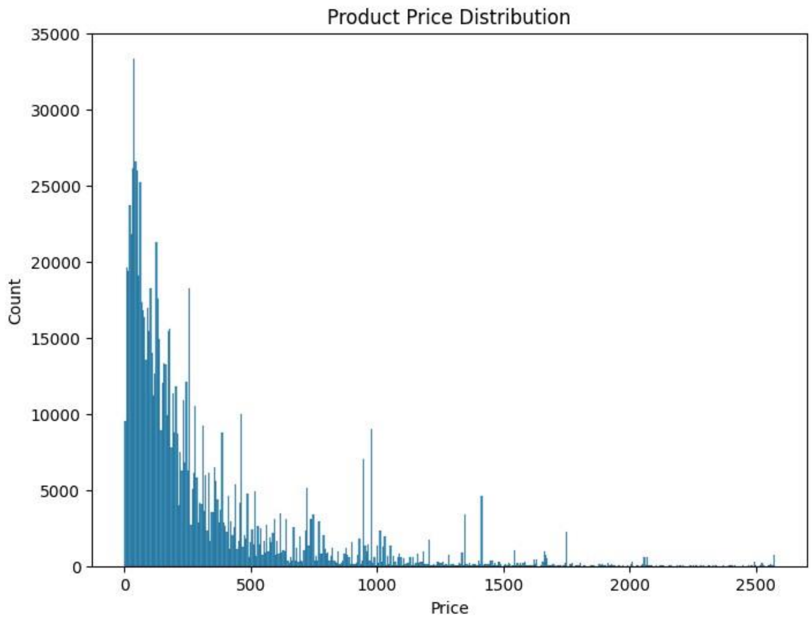


Image 1: Plot showing the distribution of product prices in the form of a histogram

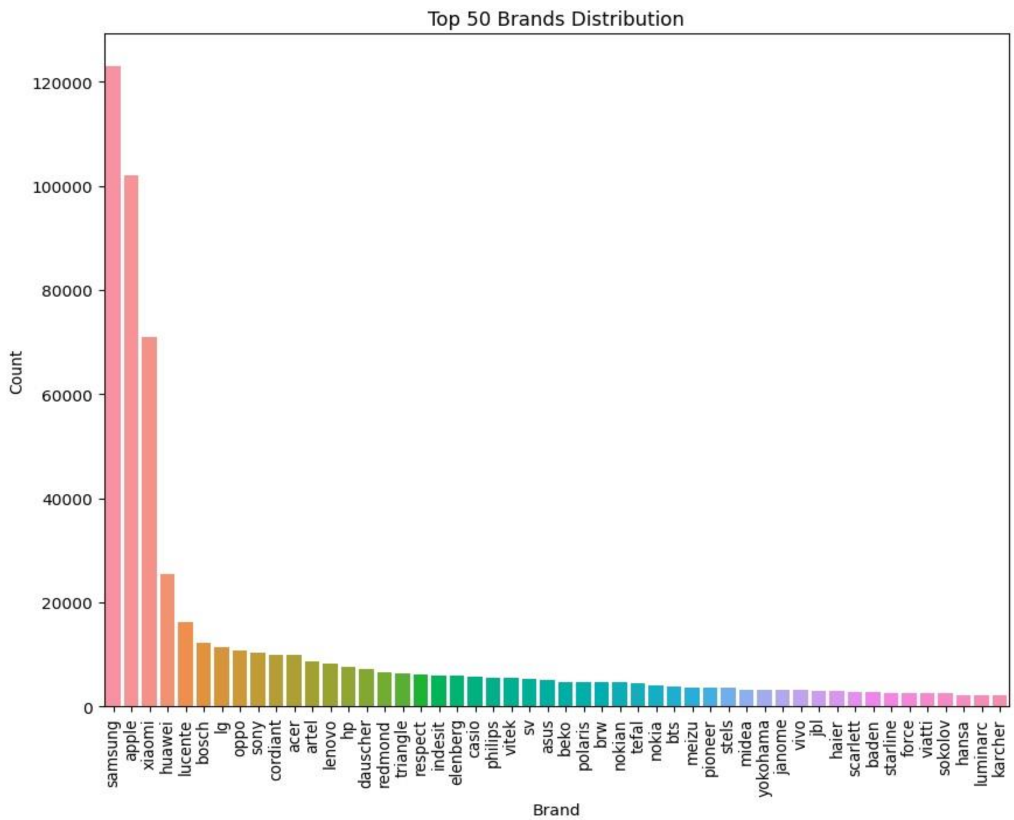


Image 2: Plot presenting a bar graph showing the popularity of the top 50 brands in the dataset

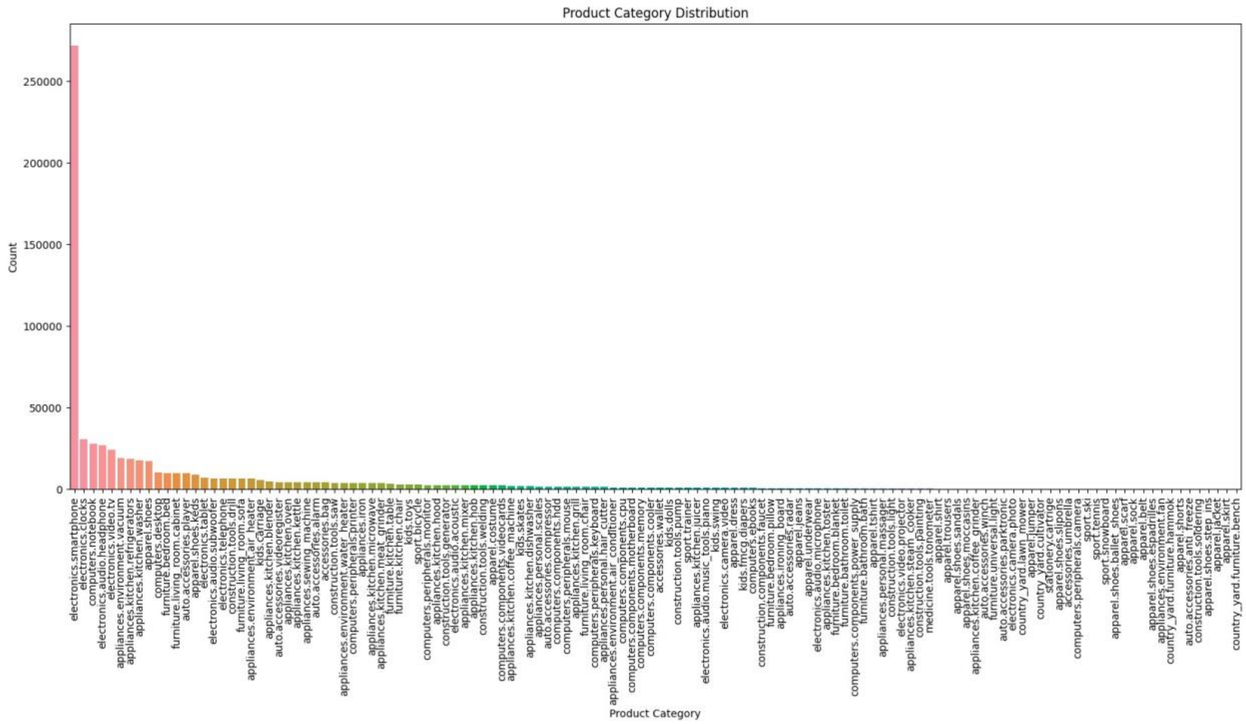


Image 3: Plot displaying a bar graph representing the distribution of product categories in the dataset

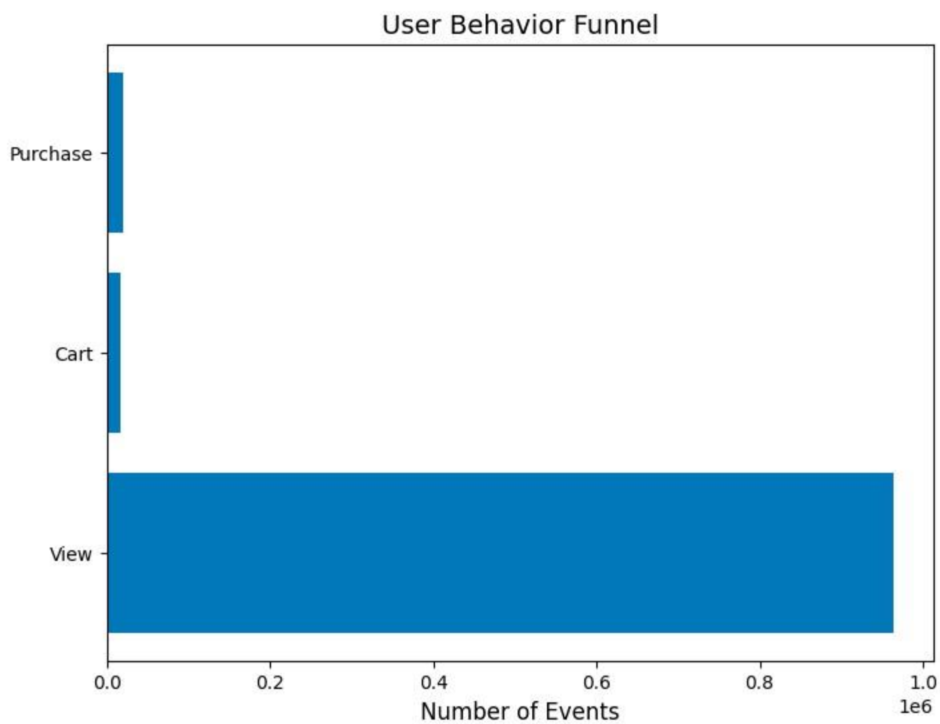


Image 4: Horizontal bar chart showing the number of events for each user behavior

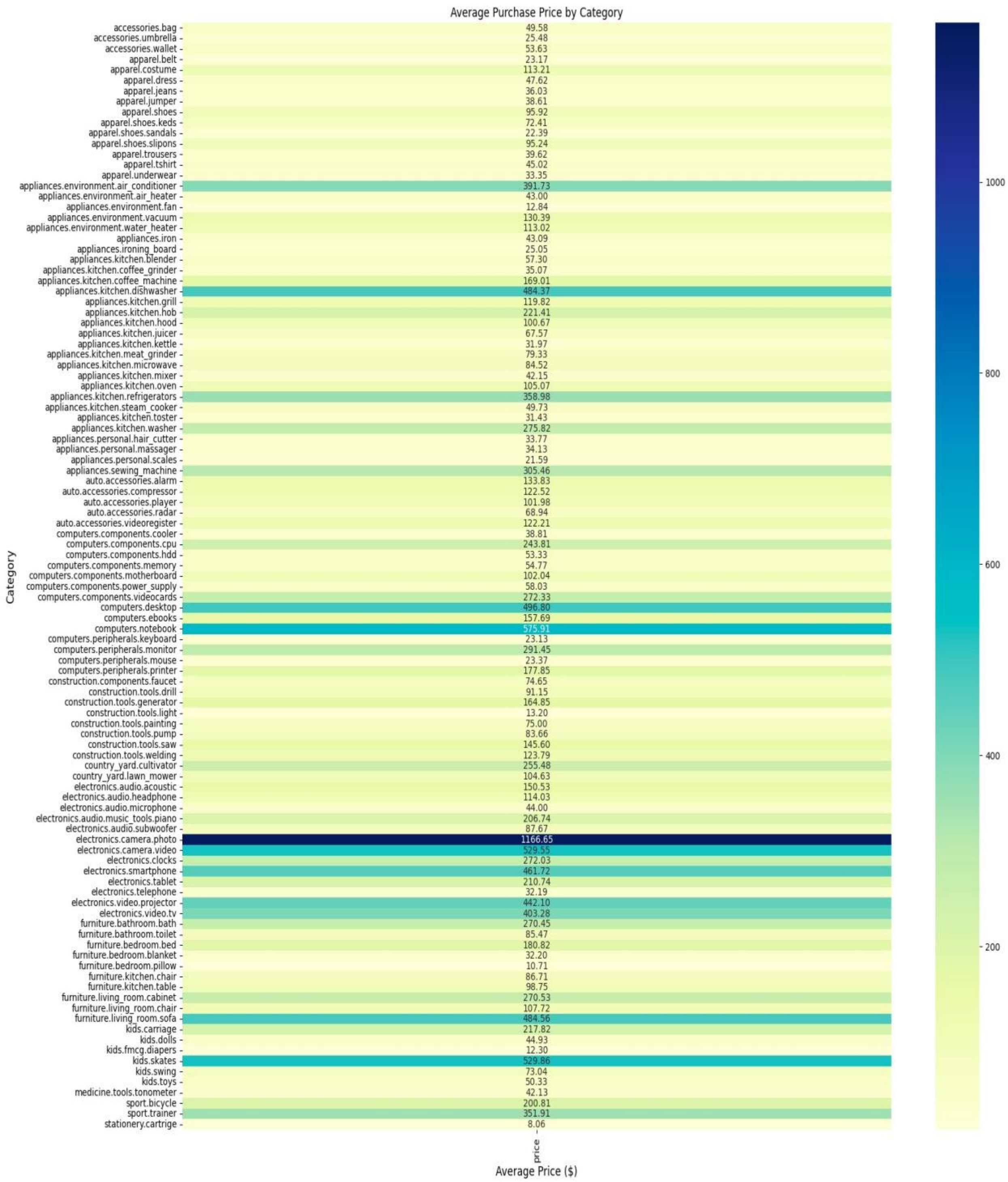
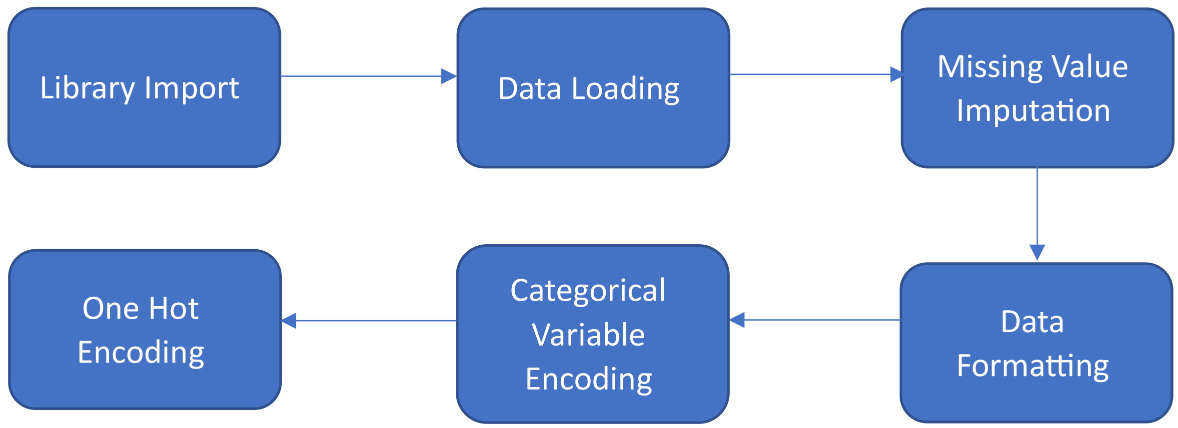


Image 5: Heat Map showing the average purchase price for each product category

Preparation

This data preparation script includes several steps that transform the raw data into a format that can be used for further analysis or modeling. Here is a detailed description of each step:



Methodology

Techniques

Content-Based Filtering, Collaborative Filtering, and Hybrid Filtering are the three popular techniques used in recommendation systems.

1. Content-Based Filtering recommends items based on their features or attributes. It identifies similar items based on the similarity of their attributes. For example, if a user likes a certain genre of music, content-based filtering recommends other songs or albums with similar genre, style, or artist. This method is useful when there is a lot of information available about the items being recommended.

2. Collaborative Filtering recommends items based on the similarity of user preferences. It identifies similar users and recommends items that the user has not seen before but have been liked by similar users in the past. For example, if two users have similar browsing and purchasing behavior, the items liked by one user can be recommended to the other user. Collaborative filtering can be divided into two categories: user-based collaborative filtering and item-based collaborative filtering.

3. Hybrid Filtering is a combination of content-based filtering and collaborative filtering. It uses the strengths of both techniques to improve the quality of recommendations. Hybrid filtering can be done in two ways: by using a weighted approach to combine the results of content-based filtering and collaborative filtering, or by switching between the two techniques based on the user's behavior and the availability of data. Hybrid filtering is often used when either

content-based filtering or collaborative filtering does not provide satisfactory recommendations.

What is a cold-start problem?

In the context of recommender systems, a cold start refers to a situation where a new item or user is added to the system and there is not enough information available about them to provide accurate recommendations.

A cold start problem can occur in two ways:

- New item: When a new item is added to the system and there are no user interactions or historical data available to recommend it to users.
- New user: When a new user joins the system and there is not enough data about their preferences, behaviors, or past interactions with the system to provide personalized recommendations.

The cold start problem is a common challenge for recommender systems and can significantly impact the accuracy of the recommendations provided. Several techniques can be used to address this problem, such as content-based recommendations or hybrid recommendations.

	Content-Based Filtering	Collaborative Filtering	Hybrid Filtering
Definition	Recommends items based on similarity between user's preferences and item's attributes	Recommends items based on similarity between users' preferences and/or behavior	Combines content-based and collaborative filtering to leverage the strengths of both methods
Data Requirements	Information about items' attributes	User-item interaction data	Both item attributes and user-item interaction data
Advantages	Able to recommend niche or less popular items, not affected by the "cold start" problem	Effective at recommending items to new users, does not require explicit data on items' attributes	Can provide more accurate and diverse recommendations by combining two methods
Disadvantages	Limited to recommending items within the same domain or category, does not capture user's evolving preferences	Susceptible to sparsity and scalability issues, unable to recommend niche items	Requires more data and computational resources, more complex implementation

Example Use Cases	Movie or music recommendations based on genre, director, or artist	E-commerce product recommendations based on user's purchase or viewing history	News article recommendations based on both article content and user's reading history

For building a recommendation system, the intended modeling technique here is the LightFM library. It is a hybrid collaborative filtering approach, which combines both user-item interactions and item content information. This is a supervised approach as the model is trained on labeled data. The LightFM algorithm will be used to solve the model.

The dataset will be loaded and converted into LightFM format using the Dataset class from the LightFM library. The interactions between users and products will be built using the build_interactions method, which takes in a generator expression to iterate over the data.

To test the model's performance, the interactions data will be split into training and testing sets using the random_train_test_split function from the LightFM library. The approach taken will be to use a 80/20 split between training and testing data, respectively, and set a random state for reproducibility.

An instance of the LightFM algorithm will be created with a chosen loss function, and the algorithm will be trained on the training set using the fit method. The number of epochs for training will also be set.

To validate the findings, the metrics used will be Precision@k and Recall@k, which measures the proportion of recommended items that are relevant to the user, where k is the number of recommendations to be made. The precision will be calculated using the precision function from the LightFM library and the mean of this metric over the testing set will be taken as the final validation score.

Process Validation

The decision to use certain techniques for building a recommendation system is influenced by several factors, such as the dataset size and type, the problem's nature, available computing resources, and desired performance metrics. For the given dataset, the LightFM algorithm was selected because it is a hybrid recommendation algorithm that combines content-based and collaborative filtering approaches, making it ideal for providing personalized recommendations where explicit feedback is not available.

Precision@k and Recall@k are evaluation metrics used to measure the effectiveness of a recommendation system.

Precision@k measures the proportion of recommended items that are relevant to the user out of the top k recommended items. In other words, it measures how many of the recommended items the user actually liked or found useful. A high precision indicates that the recommended items are highly relevant to the user's interests.

Recall@k measures the proportion of relevant items that are recommended out of all the relevant items. In other words, it measures how many of the items that the user would have liked were actually recommended. A high recall indicates that the system is able to capture a large proportion of the user's preferences.

Both metrics are typically used together to provide a comprehensive evaluation of a recommendation system. High precision indicates that the recommended items are highly relevant to the user, while high recall indicates that the system is able to capture a large proportion of the user's preferences. In general, a good recommendation system should have both high precision and high recall.

For precision@k and recall@k, we need a set of predicted items and a set of relevant items. In the context of an e-commerce recommendation system, the relevant items would be the items that a user has actually purchased or added to their cart, and the predicted items would be the top k items recommended to the user by the recommendation system.

Let's define the following variables:

- TP: number of true positives (items in the recommended set that are also in the relevant set)
- FP: number of false positives (items in the recommended set that are not in the relevant set)
- FN: number of false negatives (items in the relevant set that are not in the recommended set)
- k: number of items in the recommended set

Then, the formulas for precision@k and recall@k are:

$$Precision@k = \frac{TP}{TP + FP}$$

$$Recall@k = \frac{TP}{TP + FN}$$

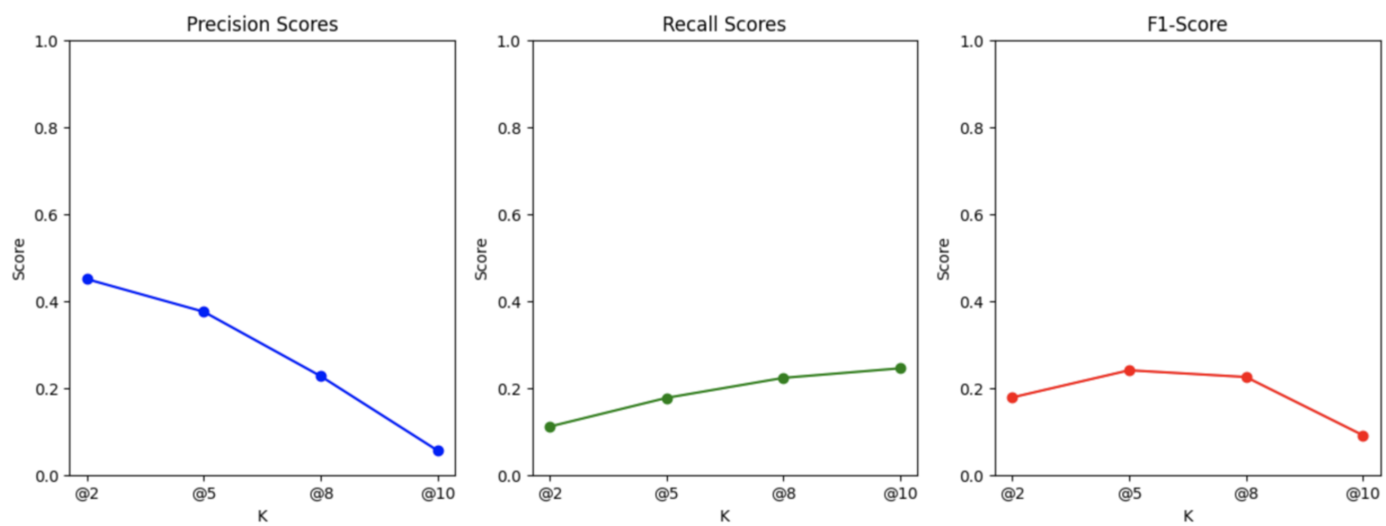
Note that precision@k measures the proportion of recommended items that are relevant, while recall@k measures the proportion of relevant items that are recommended.

Content-Based Filtering:

The results for the content-based filtering system show that the precision and recall scores are lower compared to the hybrid collaborative filtering system. The precision and recall scores decrease as the number of recommendations (k) increases, similar to the hybrid collaborative filtering system.

However, the overall precision and recall scores are lower than the hybrid collaborative filtering system, indicating that the content-based filtering system is not performing as well. The values for precision@2 and recall@2 are relatively higher compared to other values of k, suggesting that the system is more accurate in recommending a smaller number of items.

However, as the number of recommendations increases, the accuracy of the system decreases significantly. The values for precision@8 and precision@10 are particularly low, indicating that the system is not able to accurately recommend a larger number of items to users.

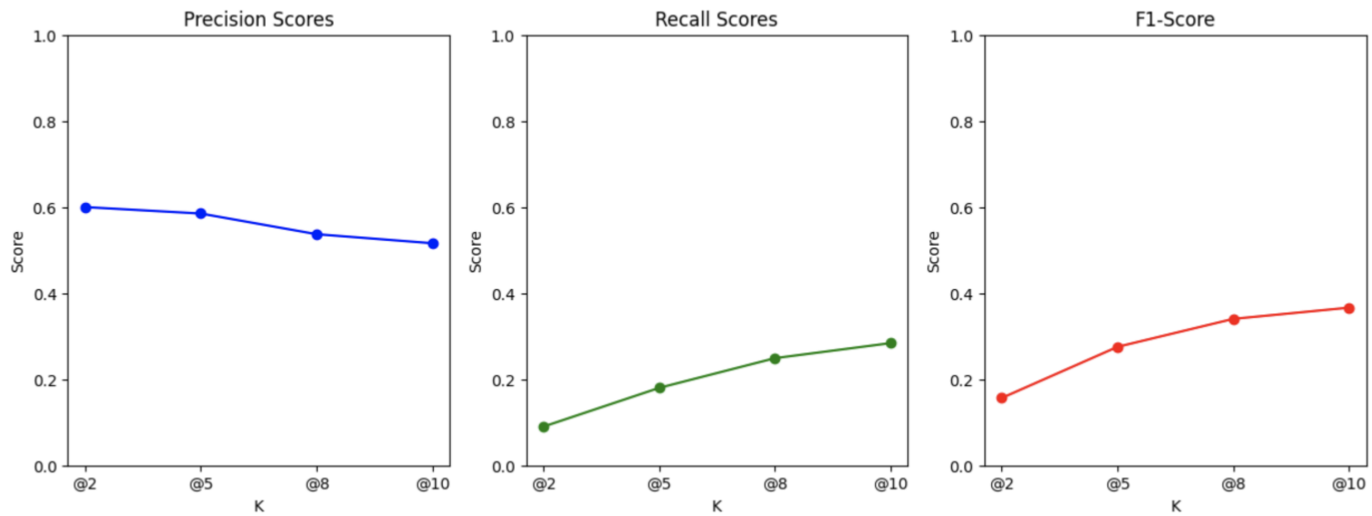


Hybrid Collaborative Filtering:

From the results obtained, it is evident that the precision and recall scores of the system decrease as the number of recommendations increases. For instance, the precision and recall scores for @2 recommendations are higher than those for @5, @8, and @10 recommendations. This means that the system is more accurate in recommending a smaller number of items to users.

However, despite the decrease in precision and recall scores as k increases, the overall precision and recall scores of the system are relatively high. This indicates that the system is performing well in providing recommendations to users.

Furthermore, the precision scores for @5, @8, and @10 recommendations are close, suggesting that the system is consistent in its recommendations across different values of k. This consistency in recommendations is an important aspect of a good recommendation system, as it helps to build trust and confidence in the system among users.



Conclusion:

The recommendation engine's performance metrics, precision and recall, provide valuable insights into its effectiveness in generating accurate and relevant recommendations to users. Precision measures the proportion of recommended items that are relevant to the users, while recall measures the proportion of relevant items that are recommended. These metrics help evaluate the quality and relevancy of the recommendations provided by the engine.

From the performance metrics, we can learn the following:

- Precision: The precision score indicates the accuracy of the recommendations. A higher precision suggests that a larger proportion of the recommended items are relevant to the users' interests and preferences.
- Recall: The recall score indicates the coverage of the recommendations. A higher recall suggests that a larger proportion of relevant items are being captured and recommended to the users.

In conclusion, the evaluation of the content-based filtering system in comparison to the hybrid collaborative filtering system provides valuable insights into their performance. Here are the key points and differentiating factors:

1. Precision and Recall Scores:
   - Content-based Filtering: The precision and recall scores are lower compared to the hybrid collaborative filtering system.
   - Hybrid Collaborative Filtering: Outperforms the content-based filtering system in terms of precision and recall scores.

2. Performance with Varying Recommendation Set Sizes:
   - Content-based Filtering: Shows relatively higher precision and recall scores for smaller recommendation sets (e.g., precision@2 and recall@2), indicating better accuracy in recommending a smaller number of items.
   - Hybrid Collaborative Filtering: Maintains consistent performance across different recommendation set sizes, providing accurate recommendations for a larger number of items.

3. Accuracy Degradation:
   - Content-based Filtering: Experiences a significant decrease in accuracy as the number of recommendations increases, as indicated by low precision@8 and precision@10 scores.
   - Hybrid Collaborative Filtering: Demonstrates better accuracy in recommending a larger number of items, maintaining higher precision and recall scores.

Considering these points, it is evident that the hybrid collaborative filtering system surpasses the content-based filtering system in terms of precision and recall scores, performance across different recommendation set sizes, and accuracy in recommending a larger number of items. The hybrid approach leverages the strengths of collaborative filtering and content-based filtering, resulting in more accurate and diverse recommendations for users.

However, it is important to note that the evaluation is based on the specific dataset and metrics used in this project. Further testing and experimentation with different datasets and evaluation metrics may provide additional insights and perspectives on the performance and differentiating factors of the recommendation systems.

Therefore, while the precision and recall metrics offer valuable information, they should be considered as part of a broader set of metrics to comprehensively evaluate the recommendation engine's effectiveness in meeting the goals of improving customer satisfaction and increasing sales.

Designing a Live Test for Evaluation:

To evaluate the recommendation engine in a live setting with real customers, a randomized controlled trial (RCT) could be conducted. The test would involve dividing users into two groups: a control group that receives recommendations from the existing system, and a treatment group that receives recommendations from the new recommendation engine.

The success of the test could be evaluated based on several factors:
- User Engagement: Monitor metrics such as click-through rates, conversion rates, and time spent on the platform to assess whether users are engaging more with the system and making more purchases.
- User Feedback: Collect qualitative feedback through surveys or interviews to understand users' perceptions of the recommendations and whether they find them useful and relevant.

1. Test Design:

1.1 Randomized Experiment: Randomly divide the customer population into two groups: the recommendation group and the control group.
1.2 Duration: Run the test for an appropriate duration to capture sufficient user interactions and behavior.

2. Metrics and Data Collection:

2.1 User Engagement Metrics:
- Click-through Rate: Measure the percentage of users who click on recommended items.
- Time Spent on Platform: Track the duration of user sessions and compare it between the recommendation and control groups.
- Interaction Frequency: Count the number of interactions with recommended items per user.

2.2 Conversion Metrics:
- Conversion Rate: Calculate the percentage of users who make a purchase after receiving recommendations, compared to the control group.
- Average Order Value: Analyze the average value of orders placed by users in the recommendation group.

2.3 Customer Satisfaction:
- Surveys and Feedback: Collect feedback from users regarding their satisfaction with the recommended items and the overall recommendation experience.

2.4 Business Impact:

- Revenue: Measure the impact on revenue generated from the recommendation group.
- Customer Retention: Assess the retention rate of users in the recommendation group compared to the control group.

3. Evaluation and Analysis:

3.1 Statistical Analysis:

- Conduct statistical tests, such as t-tests or chi-square tests, to compare the performance metrics between the recommendation and control groups.
- Determine the statistical significance of any observed differences.

3.2 A/B Testing:

- Analyze performance metrics and conduct A/B testing to evaluate the effectiveness of the recommendation engine.
- Compare the outcomes of the recommendation group with those of the control group to assess the impact of the engine.

3.3 User Feedback Analysis:

- Analyze qualitative feedback obtained from surveys and feedback forms to gain insights into user satisfaction, preferences, and any potential issues.

4. Evaluation of Success:

4.1 Performance Metrics:

- Set predefined success criteria for each metric, such as a target click-through rate or conversion rate improvement.
- Evaluate whether the recommendation engine achieves better performance compared to the control group based on these metrics.

4.2 Business Impact:

- Assess the overall impact on key business metrics, such as revenue growth, increased average order value, and improved customer retention.
- Determine the contribution of the recommendation engine to these business outcomes.

By following this test design and evaluating the metrics mentioned, we can assess the performance and impact of the recommendation engine in a live environment with real customers. The analysis of user engagement, conversion metrics, customer satisfaction, and business impact will provide valuable insights into the effectiveness of the engine and its ability to drive positive outcomes for the business.

Concerns about Privacy:

1. Protection and Privacy of Data: The security and privacy of data gathered by recommendation systems can pose significant concerns. To alleviate these risks, organizations can employ methods such as anonymizing or pseudonymizing user data, using robust encryption techniques, adopting secure data storage practices, and adhering to applicable data protection laws and privacy regulations.

2. Transparency and Conscious Approval: It's crucial for users to have detailed information regarding data gathering and its application in making recommendations. To ensure this, businesses can be transparent about their data handling methods and the algorithms used, procure informed consent from users, and offer detailed privacy controls to manage data sharing preferences.

3. Limited Data Collection: Organizations should restrict the collection and storage of non-essential user data to minimize privacy risks. This can be accomplished by only capturing vital data for accurate recommendations and by regularly reviewing and eliminating any unnecessary user data.

4. Opting-Out Options and User Authority: It's vital to provide users with control over their data and the option to opt out of data gathering and personalized recommendations. Companies can facilitate this by offering comprehensible privacy settings that let users handle their data preferences and by promptly attending to any opt-out requests to respect user decisions.

By actively addressing issues of fairness and privacy, companies can foster trust among their users, promote ethical practices, and improve the overall user experience in recommendation systems. Utilizing fairness-oriented algorithms, prioritizing data privacy, and supplying clear information alongside user control are critical actions for companies aiming to build responsible and trustworthy recommendation systems.

Deliverables

The results of the evaluation suggest that the hybrid collaborative filtering system outperforms the content-based filtering system in terms of precision and recall scores. This means that the hybrid system is more accurate in providing personalized recommendations to customers, which can result in higher customer satisfaction and increased sales for the e-commerce store. The consistency in recommendations across different values of k is also an important aspect of the system, as it helps to build trust and confidence in the system among users.

However, the results also suggest that the accuracy of the system decreases as the number of recommendations increases. This indicates that it may be more effective to provide a smaller number of more accurate recommendations to users, rather than overwhelming them with a larger number of less accurate recommendations. E-commerce businesses may need to strike a balance between the number of recommendations and their accuracy to optimize customer satisfaction and sales.

Additionally, the results highlight the importance of using advanced recommendation techniques such as hybrid collaborative filtering to provide accurate and relevant recommendations to users. By providing personalized recommendations, businesses can improve the customer experience, increase engagement, and drive sales.

References

1. sklearn.impute.SimpleImputer. (n.d.). Retrieved from https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html
2. Paul, R. (2018). Python-Ecommerce-recommendation-system-using-machine-learning. Retrieved from https://github.com/RudrenduPaul/Python-Ecommerce-recommendation-system-using-machine-learning
3. LightFM Documentation. (n.d.). Retrieved from https://making.lyst.com/lightfm/docs/home.html
4. Gupta, V. (2019). Recommendation System in Python LightFM. Towards Data Science. Retrieved from https://towardsdatascience.com/recommendation-system-in-python-lightfm-61c85010ce17
5. Rai, A. (2020). Introduction to Recommender Systems using LightFM in Azure ML. Plain English. Retrieved from https://python.plainenglish.io/introduction-to-recommender-systems-using-lightfm-in-azure-ml-e86feaff6ac4
6. Singh, V. (2019). A Complete Guide to Categorical Data Encoding. Analytics India Magazine. Retrieved from https://analyticsindiamag.com/a-complete-guide-to-categorical-data-encoding/
7. Rathore, R. (2020). Transforming Skewed Data. Towards Data Science. Retrieved from https://towardsdatascience.com/transforming-skewed-data-73da4c2d0d16
8. Kumar, A. (2019). Day 8: Data Transformation: Skewness, Normalization, and Much More. Medium. Retrieved from https://medium.com/@TheDataGyan/day-8-data-transformation-skewness-normalization-and-much-more-4c144d370e55
9. Kumar, N. (2019). Solving Business Usecases by Recommender System using LightFM. Towards Data Science. Retrieved from https://towardsdatascience.com/solving-business-usecases-by-recommender-system-using-lightfm-4ba7b3ac8e62
10. Hill, E. (2019). Building a Recommendation Engine with Collaborative Filtering. Real Python. Retrieved from https://realpython.com/build-recommendation-engine-collaborative-filtering/
11. Huang, J. (2019). Popular Evaluation Metrics in Recommender Systems Explained. Qloo. Retrieved from https://medium.com/qloo/popular-evaluation-metrics-in-recommender-systems-explained-324ff2fb427d
12. Content-Based and Collaborative Filtering. (n.d.). Oxford Journal. Retrieved from https://www.oxjournal.org/content-based-collaborative-filtering/
13. Content-Based Recommender Systems. (n.d.). Google Developers. Retrieved from https://developers.google.com/machine-learning/recommendation/content-based/basics
14. Towards Data Science. (2021, September 8). Recommendation Systems: Models and Evaluation. https://towardsdatascience.com/recommendation-systems-models-and-evaluation-84944a84fb8e