**Recommendation Engine Report**

**Introduction**

This project involves the development of a recommendation engine tailored for e-commerce websites. The system leverages user and item interaction datasets to learn patterns and provide personalized recommendations. Initially, four distinct models were created to capture different aspects of the data and recommendation strategies. Building on these, three hybrid models were developed by combining the strengths of the individual approaches to enhance accuracy and user satisfaction. This report outlines the dataset, exploratory data analysis (EDA), feature engineering, model creation, hybridization, and results obtained during the project.

**Dataset Description**

The dataset for the recommendation engine project is composed of four interconnected components, each providing critical information for building a comprehensive recommendation system tailored for e-commerce websites. The details of the dataset are as follows:

***User Data (user\_data.csv)***

This dataset contains detailed demographic and personal information about users, such as their unique identifiers (UserID), names, email addresses, gender, age groups, and location details. This data is essential for personalizing recommendations based on user-specific attributes like age, gender, or geographic preferences.

***Product Data (product\_data.csv)***

This dataset describes the products available in the e-commerce store. It includes unique product identifiers (ProductID), product names, categories, base prices, discounts, stock availability, and descriptions. This information allows the recommendation engine to categorize and rank products based on user preferences and product attributes.

***Transaction Data (transaction\_data.csv)***

This dataset records historical transactions, mapping users (UserID) to the products they purchased (ProductID). It also includes transaction amounts, payment methods, purchase dates, delivery times, and applied discounts. This transactional history helps identify user preferences, spending behavior, and seasonal purchasing trends.

***Interaction Data (interaction\_data.csv)***

This dataset captures user-product interactions beyond purchases. It includes details such as pages viewed, search queries, time spent on pages, wishlist additions, cart interactions (added and abandoned), and product reviews (ratings and text). This interaction data is crucial for understanding user engagement and predicting potential purchases based on user activity patterns.

Together, these datasets provide a rich foundation for training and evaluating the recommendation engine, supporting collaborative, content-based, and hybrid approaches.

| **Dataset** | **File Name** | **Description** |
| --- | --- | --- |
| **User Data** | user\_data.csv | Contains demographic and personal details of users such as unique IDs, names, emails, gender, age group, city, and country. |
| **Product Data** | product\_data.csv | Includes product details such as unique IDs, names, categories, base prices, discounts, stock quantity, and descriptions. |
| **Transaction Data** | transaction\_data.csv | Records user-product transactions, including transaction amounts, payment methods, purchase dates, delivery times, and discounts. |
| **Interaction Data** | interaction\_data.csv | Captures user-product interactions like pages viewed, search queries, wishlist additions, cart actions, and reviews. |

**Exploratory Data Analysis (EDA)**

During the EDA process, the following steps were performed to clean, preprocess, and prepare the datasets for modeling:

1. **Handling Missing Values**
   * In the **User Data**, missing email values were replaced with a default placeholder (unknown@example.com).
   * In the **Product Data**, missing values in BasePrice and DiscountAmount were replaced with their respective medians.
   * In the **Transaction Data**, missing values in TransactionAmount and DiscountAmount were filled with their medians.
   * In the **Interaction Data**, missing values in TimeSpentOnPages were filled with the median, and ReviewRating was replaced with the mean. Missing review text was replaced with a default placeholder (No review provided).
2. **Removing Outliers**
   * The Z-score method was used to remove outliers in BasePrice and DiscountAmount from the **Product Data**.
   * The Interquartile Range (IQR) method was applied to detect and remove outliers in TransactionAmount and DiscountAmount from the **Transaction Data** and TimeSpentOnPages from the **Interaction Data**.
3. **Feature Scaling**
   * Numerical columns such as BasePrice, DiscountAmount, and StockQuantity in the **Product Data** were scaled using Min-Max normalization to bring values into a uniform range.
   * Similarly, ViewedPages and TimeSpentOnPages in the **Interaction Data**, and TransactionAmount in the **Transaction Data**, were normalized using Min-Max scaling.
4. **Categorical Encoding**
   * Categorical columns in the datasets were encoded using **Label Encoding** for efficient processing in machine learning models.
     + **User Data**: Gender, AgeGroup, City, and Country.
     + **Product Data**: Category.
     + **Transaction Data**: PaymentMethod.

These steps ensured the datasets were clean, devoid of inconsistencies, and prepared for further analysis and model development. This preprocessing pipeline improved data quality and facilitated meaningful insights during feature engineering and model training.

**Feature Engineering**

The feature engineering process involved deriving new features and aggregating data to enhance the dataset's predictive capabilities. Here are the steps undertaken:

1. **Calculating Age**
   * Derived the Age feature from the DateOfBirth column in the **User Data** by subtracting the birth year from the current year.
   * Dropped the DateOfBirth column after extracting the Age feature to avoid redundancy.
2. **Aggregating Transaction Data**
   * Calculated the total amount spent (TotalSpent) and the total number of transactions (TotalTransactions) for each user in the **Transaction Data**.
   * Merged these aggregated features with the **User Data**, filling missing values with 0 for users with no transactions.
3. **Effective Price Calculation**
   * Computed the EffectivePrice for products in the **Product Data** as the difference between BasePrice and DiscountAmount.
4. **Average Product Rating**
   * Derived the AverageRating for each product by calculating the mean of the ReviewRating column in the **Interaction Data**.
   * Merged this feature with the **Product Data**, with missing ratings filled as 0 for products with no reviews.
5. **User Interaction Aggregation**
   * Aggregated user interactions from the **Interaction Data** to calculate:
     + TotalViews: Total number of pages viewed by a user.
     + TotalTimeSpent: Total time spent on pages by a user.
     + WishlistCount: Total number of wishlist items added by a user.
   * Merged these aggregated features with the **User Data**, filling missing values with 0 for users with no recorded interactions.
6. **Composite Score Calculation**
   * Defined a weight matrix to assign different importance levels to features in the **Interaction Data**:
     + High weights were given to features like WishlistItems, CartAddedItems, and ReviewRating to emphasize their significance in determining user behavior.
     + Lower weights were assigned to features like TimeSpentOnPages and CartAbandonedItems.
   * Computed a CompositeScore for each user-product interaction by applying the weights to the relevant features.
7. **Interaction Matrix**
   * Created a user-product interaction matrix from the **Interaction Data**, with CompositeScore values representing user engagement with each product. Missing values were replaced with 0 to handle users or products with no interactions.

**Outcome of Feature Engineering**

* Enhanced datasets with meaningful derived features like Age, EffectivePrice, TotalSpent, and CompositeScore.
* Improved understanding of user behavior and product performance through aggregated and weighted interaction data.
* Provided a structured interaction matrix suitable for recommendation systems and machine learning tasks.

This feature engineering pipeline is designed to add depth and predictive power to the dataset, enabling better insights and more accurate modeling outcomes.

**User-Based Collaborative Filtering Model**

The **User-Based Collaborative Filtering (CF)** model identifies recommendations for a user by leveraging the behavior and preferences of similar users. This approach assumes that users with similar interaction patterns will have overlapping interests in products.

**Key Steps in the Model:**

1. **User Similarity Matrix**
   * The cosine similarity between users is calculated using their interaction data (e.g., Composite Scores for product interactions).
   * This creates a similarity matrix where each entry represents how closely one user is related to another based on their behavior.
2. **Finding Similar Users**
   * For a given user, the most similar users are identified from the similarity matrix by sorting the similarity scores in descending order.
3. **Aggregating Product Scores**
   * The products interacted with by the similar users are aggregated.
   * Products are scored based on the sum of interaction scores from the similar users.
   * Products with higher scores are prioritized for recommendations.
4. **Generating Recommendations**
   * The top products are recommended to the target user based on their composite scores.
   * Recommendations are filtered to exclude products the target user has already interacted with.
5. **Enhanced Recommendations with Explanations**
   * Each recommendation is explained by identifying the specific similar users who interacted with the product.
   * If product details (e.g., name, category, description) are available, they are displayed alongside the recommendation for better user understanding.

**Model Example**

For a user with ID 0152ed8e-51a1-4ffa-a06e-4f10e3af0e0b, the model provides recommendations like:

* **Product Name**: Vacuum Cleaner
  + **Explanation**: Recommended because users similar to you (9a0cb7c8-8359-4a58-968f-b984894b0f31,225aca0c-951d-4783-910d-00678b9ce09f, 9aa4d705-f8fd-47a7-bb8e-7e96b5173483) interacted with this product.

This model serves as a foundational recommendation system, combining simplicity with the ability to personalize effectively. It is well-suited for scenarios where user interaction data is abundant and product diversity is high.

**Item-Based Collaborative Filtering Model**

The Item-Based Collaborative Filtering (CF) model provides recommendations by identifying products similar to those a user has interacted with. Unlike User-Based CF, which focuses on similarities between users, Item-Based CF leverages similarities between items to suggest products that are frequently associated with one another based on user behavior.

**Key Steps in the Model**

1. **Item Similarity Matrix**
   * **Computation**: The cosine similarity metric is applied to the interaction matrix, transposed to calculate similarities between items instead of users.
   * **Result**: A similarity matrix where each entry indicates how closely two items are related based on user interactions (e.g., purchases, reviews, or views).
2. **Identifying Similar Items**
   * For a given product, the similarity matrix is queried to rank other products based on similarity scores in descending order.
3. **Recommendation Generation**
   * For each product a user has interacted with, the top similar items are identified.
   * A consolidated list of recommendations is created by aggregating these similar items, ensuring that items already interacted with by the user are excluded.
4. **Weighting and Filtering**
   * **Weighting**: Similar items are scored higher if they have strong similarities with multiple products the user has engaged with.
   * **Filtering**: Results are filtered by category, price range, or other user-specific preferences to ensure relevance.
5. **Explanation**
   * Each recommendation is accompanied by an explanation, e.g., "Recommended because it is similar to products you viewed or purchased."

**Example**

For a product with the following details:

ProductID: 99ffb796-d71c-44f7-b147-b4a80cd4a3ae

ProductName: Dress

Category: 2

**Recommended Products**:

1. **Moisturizer**
   * Similarity Score: 0.89
   * Reason: Frequently purchased together with similar fashion products.
2. **Jacket**
   * Similarity Score: 0.85
   * Reason: Customers who bought dresses often explore outerwear options.
3. **Running Shoes**
   * Similarity Score: 0.78
   * Reason: Complements activewear collections.
4. **Lipstick**
   * Similarity Score: 0.76
   * Reason: Often viewed alongside fashion accessories.

**Neural Collaborative Filtering (NCF) Model**

The **Neural Collaborative Filtering (NCF)** model is a deep learning-based approach to collaborative filtering, which is widely used in recommendation systems. It leverages neural networks to model the interaction between users and items, providing better generalization and expressiveness than traditional matrix factorization techniques.

**Key Components of the NCF Model:**

1. **Embedding Layers**:
   * The NCF model uses embedding layers to map users and items into dense vector representations in a latent space.
   * These embeddings capture the characteristics of users and items, enabling the model to learn their implicit relationships.
2. **Concatenation of Embeddings**:
   * The embeddings for a pair of products (or user-item pair in other applications) are concatenated to create a combined representation.
   * This combined representation is the input for the fully connected layers that follow.
3. **Fully Connected Layers**:
   * These layers process the concatenated embeddings, capturing non-linear and complex relationships between the products.
   * The layers have decreasing sizes (e.g., 256 → 128 → 64), which helps in learning hierarchical patterns.
4. **Output Layer**:
   * The final output layer consists of a single neuron with a sigmoid activation function.
   * It outputs a probability score indicating the likelihood of similarity (or interaction) between two products.

**Implementation Details:**

1. **Data Preparation**:
   * **Product Indexing**: Each product is assigned a unique integer index using LabelEncoder.
   * **Feature Engineering**: Numeric features like BasePrice, StockQuantity, and DiscountAmount are standardized using StandardScaler. These features could further enhance product embeddings when combined with other features.
   * **Training Pairs**: Positive pairs consist of a product with itself, while negative pairs are randomly sampled product pairs that are expected to be dissimilar.
2. **Model Architecture**:
   * The embedding dimension for products is set to 30.
   * Embedding layers encode each product index into a vector representation.
   * The embeddings are concatenated and passed through fully connected layers to predict the similarity between products.
3. **Training**:
   * The model is trained using binary cross-entropy loss, where the targets are 1 for positive pairs and 0 for negative pairs.
   * The optimizer used is **Adam**, with a learning rate of 0.001 for efficient weight updates.
4. **Recommendations**:
   * After training, the model predicts similarity scores for a given product pair.
   * For recommendations, the model evaluates the similarity of a product with all other products in the dataset, sorts them by score, and selects the top-N products with the highest similarity.
5. **Example: NCF Recommendations**
6. To illustrate the recommendation process, consider the product T-Shirt as the input product.
7. **Input Product Details:**

| **ProductID** | **ProductName** | **Category** | **BasePrice** | **Description** |
| --- | --- | --- | --- | --- |
| 88720aa8-541f-4e8a-80c8-b1977a39492d | T-Shirt | 2 | 0.25 | Lightweight, casual, and comfortable design. |

1. **Top-10 Recommendations:**

| **ProductID** | **ProductName** | **Reason** | **Similarity Score** |
| --- | --- | --- | --- |
| 88720aa8-541f-4e8a-80c8-b1977a39492d | **Jeans** | Recommended because it's similar to 'T-Shirt'. | 0.01 |
| 80d92a13-5beb-43df-bcac-0abc7ee7f44d | **Jacket** | Recommended because it's similar to 'T-Shirt'. | 0.00 |
| ad142eef-d985-4144-ad72-2d3cdec45bf2 | **Fiction** | Recommended because it's similar to 'T-Shirt'. | 0.01 |
| 2cc6a077-f277-43c4-b740-0dc6c7070944 | **Sweater** | Recommended because it's similar to 'T-Shirt'. | 0.00 |
| 92ab99a2-20f8-490a-b9a6-c2582f4375c9 | **Cookware Set** | Recommended because it's similar to 'T-Shirt'. | 0.00 |
| 98912765-ee7c-41a1-a7fd-3f250e490e1a | **Yoga Mat** | Recommended because it's similar to 'T-Shirt'. | 0.01 |
| 9f6dd1ad-a127-4012-9abc-70f539795f94 | **Running Shoes** | Recommended because it's similar to 'T-Shirt'. | 0.01 |

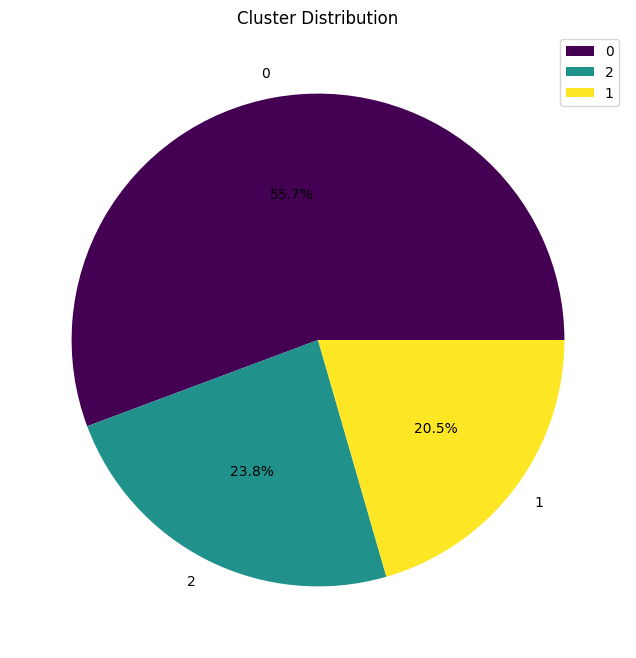
The NCF model implemented above creates a robust recommendation framework that predicts product similarities and ranks items for content-based recommendations. By leveraging neural networks, it outperforms traditional methods in capturing intricate patterns in product interactions.

**User Clustering Model**

The User Clustering Model is designed to group users based on behavioral and demographic attributes, enabling targeted recommendations and personalized user experiences. The implementation leverages advanced preprocessing, clustering techniques, and evaluation methods to identify user patterns effectively.

**Key Components of the User Clustering Model:**

1. **Feature Selection and Scaling:**
   * The model selects key attributes like Gender, AgeGroup, TotalSpent, TotalTransactions, TotalViews, TotalTimeSpent, and WishlistCount.
   * Missing values are replaced with 0, and features are standardized using StandardScaler to ensure uniform scaling.
2. **Dimensionality Reduction for Visualization:**
   * PCA (Principal Component Analysis) reduces the feature dimensions to 2, retaining maximum variance.
   * This allows visualization of clusters in a 2D space for better interpretability.
3. **KMeans Clustering:**
   * KMeans algorithm clusters users into 3 distinct groups.
   * Each user is assigned a cluster label based on their behavior and attributes.
   * The identified clusters are visualized using the 2D PCA representation.
4. **Cluster-Based Recommendations:**
   * Recommendations are derived from transaction data filtered by cluster.
   * For each cluster, the most frequently purchased products are identified.
   * The top-N products are recommended to users belonging to the same cluster.

**Implementation Details:**

1. **Data Preparation:**
   * The clustering features are extracted from the user\_data DataFrame, preprocessed, and standardized.
2. **PCA Transformation:**
   * Dimensionality reduction is performed for visualizing clusters effectively.
3. **Clustering:**
   * The KMeans model groups users into clusters based on their feature vectors.
4. **Recommendations:**
   * A user’s cluster is determined, and recommendations are made based on the purchasing behavior of other users in the same cluster.

**Example:**

* **Input User:**  
  User ID: 0000d51a-e6b0-4333-8007-a3824b1d53a2.
* **Cluster Assignment:**  
  The user belongs to Cluster 1.
* **Top-N Recommendations for Cluster 1:**

| **ProductID** | **ProductName** | **Category** | **BasePrice** | **Description** |
| --- | --- | --- | --- | --- |
| f758cc3e | Sweater | 2 | 0.593994 | Start specific always place tell candidate. |
| f2a59177 | Perfume | 0 | 0.418594 | Our help night answer everybody business again... |
| a9e202a4 | Football | 5 | 0.637020 | Myself glass behind early though decade effort... |
| c886525f | Laptop | 3 | 0.733324 | Religious lot economy manage sell when alone f... |
| 4b38f41f | Lipstick | 0 | 0.456262 | Test news represent debate talk break their ca... |

This user clustering model forms a robust framework for personalized recommendations, improving user engagement and satisfaction.

**Hybrid Recommendation Model**

The **Hybrid Recommendation Model** combines multiple recommendation techniques to provide personalized and context-aware product suggestions. By integrating collaborative filtering (user-based and item-based), content-based filtering, and clustering-based recommendations, this approach ensures a comprehensive and diverse set of recommendations tailored to individual user preferences and product similarities.

**Key Features of the Hybrid Model:**

1. **Multi-Source Recommendations:**
   * The model utilizes multiple recommendation techniques, each contributing its strengths to the final recommendations.
2. **Weighted Aggregation:**
   * Each recommendation source is assigned a weight (e.g., user-based: 0.6, item-based: 0.5, etc.), allowing fine-tuning of the influence of each method in the final scoring.
3. **Context-Aware Recommendations:**
   * Product context (e.g., product\_id) can be incorporated, making the recommendations more relevant based on specific product interactions.

**Recommendation Techniques:**

1. **User-Based Collaborative Filtering (CF):**
   * Suggests products based on the purchasing behavior of similar users.
   * Example: If User A and User B have overlapping preferences, products purchased by User B but not by User A are recommended to User A.
2. **Item-Based Collaborative Filtering (CF):**
   * Identifies products frequently purchased together.
   * Example: If T-Shirt and Jacket are often bought together, users viewing or purchasing a T-Shirt may receive Jacket recommendations.
3. **Content-Based Filtering:**
   * Recommends products similar to the user’s past interactions based on attributes like category, description, and other metadata.
   * Example: If a user purchases a T-Shirt, the model might suggest other products in the "Clothing" category.
4. **Clustering-Based Recommendations:**
   * Leverages precomputed user clusters to recommend products popular among similar users in the same cluster.
   * Example: Users in a "Fashion Enthusiasts" cluster might receive trending clothing recommendations.

**Example Workflow:**

* **Input:**
  + User ID: 0001582d-b85d-4803-8167-f44bc0d70f67.
  + Product ID: 0098b0d4-7879-4e51-9618-a127e12fd060 (Context: User interacted with this product).
* **Recommendation Sources:**
  + User-Based CF: ['Sweater', 'Football', 'Running Shoes'].
  + Item-Based CF: ['Shampoo', 'Biography', 'Jacket'].
  + Content-Based Filtering: ['Puzzle', 'Tennis Racket', 'Vacuum Cleaner'].
  + Clustering-Based: ['Biography', 'Shampoo', 'Jacket'].
* **Weighted Aggregation:**  
  Each product is scored based on the weights of the respective recommendation sources.
* **Final Recommendations:**  
  After sorting by scores, the top-N recommendations are:
  + Sweater
  + Football
  + Running Shoes
  + Shampoo
  + Biography

**Benefits of the Hybrid Model:**

1. **Diversity in Recommendations:**
   * Combines collaborative and content-based insights for a broader range of suggestions.
2. **Improved Accuracy:**
   * Balances multiple algorithms, reducing biases of any single technique.
3. **Contextual Relevance:**
   * Incorporates real-time user-product interactions (e.g., product\_id) to enhance relevance.
4. **User Engagement:**
   * By leveraging cluster preferences and individual behavior, the model creates a personalized shopping experience.

This hybrid approach ensures that recommendations are not only personalized but also diverse, adaptive, and contextually aligned with user preferences and interactions.

1-2 page small summary , dataset, models , evalution , result