INFOSYS SPRINGBOARD INTERNSHIP

A project report on

“AI STYLIST”

TABLE OF CONTENTS

1. PROBLEM STATEMENT :

In today's fast-paced world, finding the perfect outfit that suits individual preferences, body type, and occasions can be overwhelming and time-consuming. While personal stylists offer tailored clothing recommendations, these services are often expensive and inaccessible for many. AI Styling addresses this challenge by automating outfit recommendations using a machine learning model that understands fashion preferences, trends and contextual needs.

The AI-powered system personalizes suggestions to help users feel confident and satisfied with their clothing choices while saving time and effort. It democratizes access to styling services, offering a cost-effective alternative to traditional methods.

Business Motivation :

The fashion industry is undergoing rapid digital transformation and businesses are increasingly seeking innovative solutions to enhance customer engagement and satisfaction. Automating personal stylist tasks enables companies to provide a scalable and cost-effective styling service. This approach leverages machine learning models to analyze customer preferences, past choices and emerging fashion trends, offering outfit recommendations that resonate with user's unique styles. By integrating automation into the styling process, fashion retailers can enhance customer retention, improve sales conversions and streamline inventory management.

Expected Benefits :

For Users:

- Personalized outfit recommendations save time and effort in finding suitable clothing.
- Users can explore curated options based on their preferences, reducing time spent searching for the perfect outfit.
- Simplifies decision-making by highlighting seasonally and contextually appropriate clothing.

- Cost-effective access to high-quality styling services.
- Integrated suggestions ensure a smoother shopping journey, potentially pairing items into complete outfits.

For Stakeholders:

- Improved customer engagement and loyalty through tailored recommendations.
- Better matching of products to user preferences decreases return rates, lowering logistical costs.
- Analyzing user interactions provides valuable data on consumer preferences, helping improve inventory and marketing strategies.
- A personalized, enjoyable shopping experience strengthens user trust and builds long-term relationships.
- Insights into trends and preferences help stock the right products, reducing overstock and unsold inventory.

2. DATA COLLECTION:

The dataset sourced from Kaggle provides comprehensive information that forms the backbone of the AI-powered recommendation system. It comprises detailed and structured data that encompasses the following key components:

The dataset includes valuable user-generated data such as product ratings and reviews.The dataset captures the base color of each product and its associated season like Summer, Winter, Fall, Spring.

Products are organized into hierarchical categories such as Apparel, Accessories and Footwear, with further classification into subcategories like Bottomwear, Topwear, Watches and Shoes.

Each product entry is linked to an image, referenced through filenames or URLs.These images play a crucial role in visual analysis and aesthetic evaluations, enabling the system to incorporate visual appeal into its recommendations.

Challenges Faced and Resolutions:

Initially, we worked with a dataset that lacked image information, focusing solely on product attributes and user data. Recognizing the importance of visual data for styling recommendations, we explored various approaches to incorporate images:

1. Adding image URLs directly into the dataset.
2. Embedding image files directly within dataset cells.

But, these approaches proved to be time-intensive and inefficient due to the following reasons:

- Collecting and linking individual images to corresponding product entries required extensive manual effort.
- Embedding image files directly increased the dataset size significantly, causing computational inefficiencies during analysis and processing.
-

Resolution:

To overcome these challenges, we opted for a pre-existing dataset that included image URLs. This dataset was further updated and enriched by adding key attributes such as user_id, prices, ratings and year. By utilizing image URLs instead of embedding images directly, we streamlined the integration process while maintaining the flexibility to fetch images dynamically during analysis or visualization.

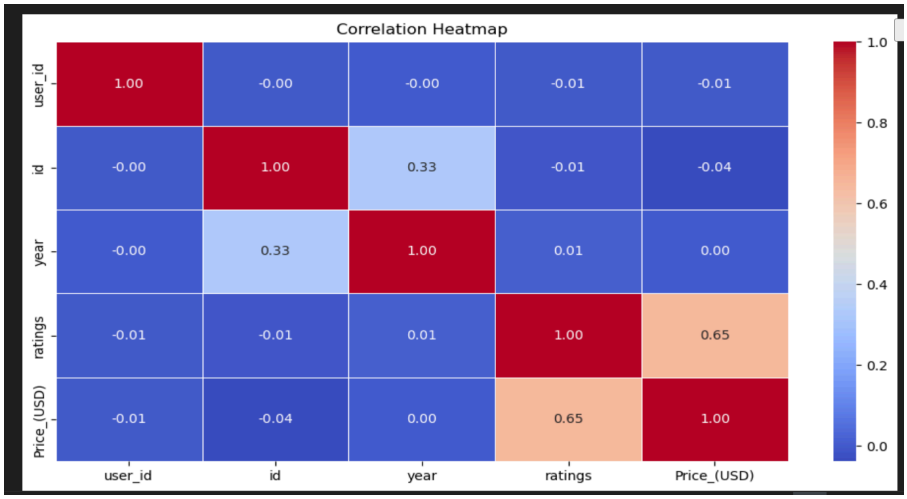
This approach not only saved time but also ensured the dataset remained manageable and scalable for downstream tasks.

- **Challenge**: **Balancing the Dataset**
  The dataset exhibited an uneven distribution across categories, with women's apparel significantly overrepresented compared to other categories such as men's or unisex products.
- **Resolution**:
  To mitigate this, data balancing techniques were applied. Sampling was used to increase the representation of underrepresented categories by simulating additional entries.

3.DATA PREPROCESSING:

EDA:

- We standardized the column names by replacing spaces with underscores using df.columns.str.replace() to avoid any potential issues during analysis. Missing values were identified using df.isnull().sum() to check which columns had missing data.
- Tools like **Seaborn**, **Matplotlib**, and **Pandas** were used to generate heatmaps, histograms, bar charts and line plots, enhancing data interpretation.
- We identified categorical columns like gender, masterCategory, subCategory, and articleType.
- Bar charts were created to visualize the most common categories in each column, using sns.countplot().This helped us identify dominant categories and distribution patterns for categorical variables.
- **Visualization**: A heatmap (sns.heatmap()) was generated to visually inspect missing data across the dataset, allowing us to see patterns of missingness.

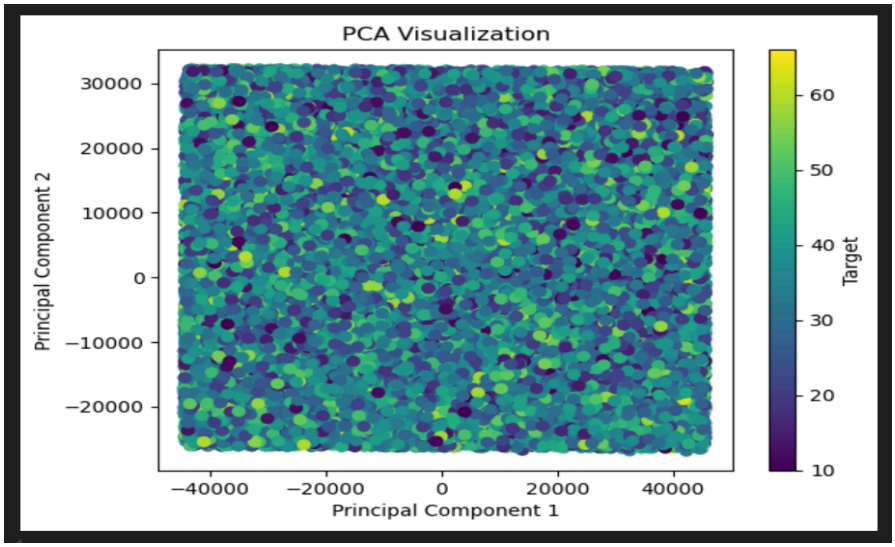


Principal Component Analysis (PCA):

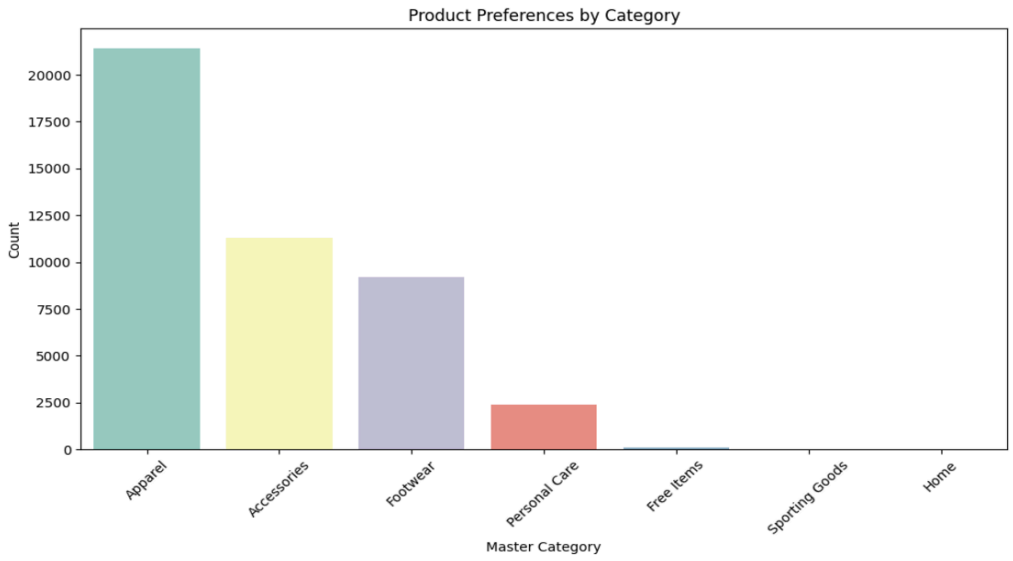
- PCA was performed on the selected numerical data using PCA(n_components=2), where n_components=2 was chosen to reduce the data

to 2 principal components. This allowed us to plot the data in a two-dimensional space.

- The PCA model was then fitted and transformed using pca.fit_transform(), which calculated the first two principal components that explain the maximum variance in the dataset.



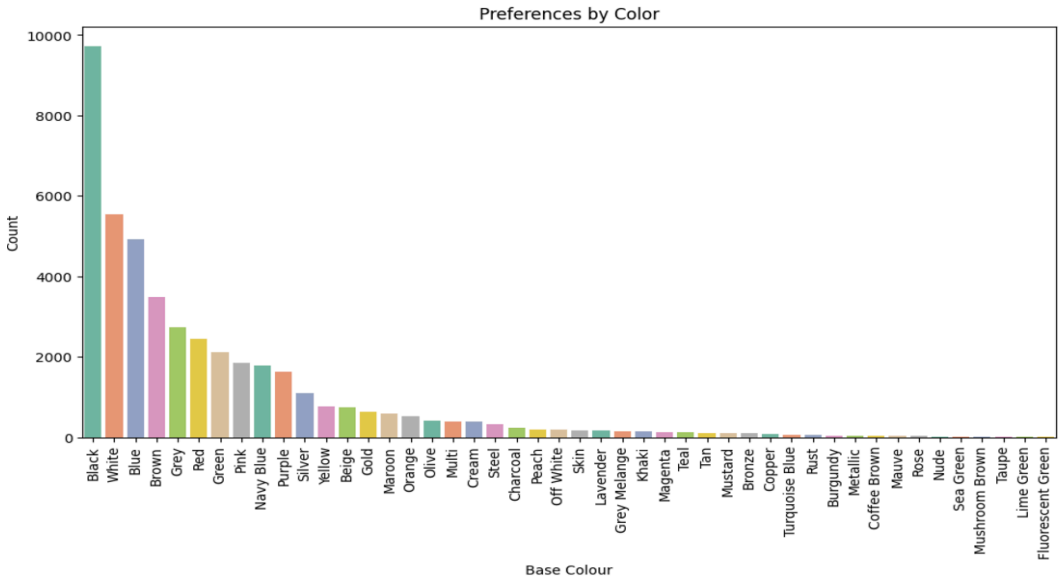
Product Preferences by Category:



This countplot highlights the distribution of product preferences across various masterCategory values in the dataset. The categories represent broad classifications, such as clothing, accessories, or footwear.

- Helps identify the most popular product categories based on count.
- Useful for understanding overall trends and demand at a high level.

Preferences by Color:

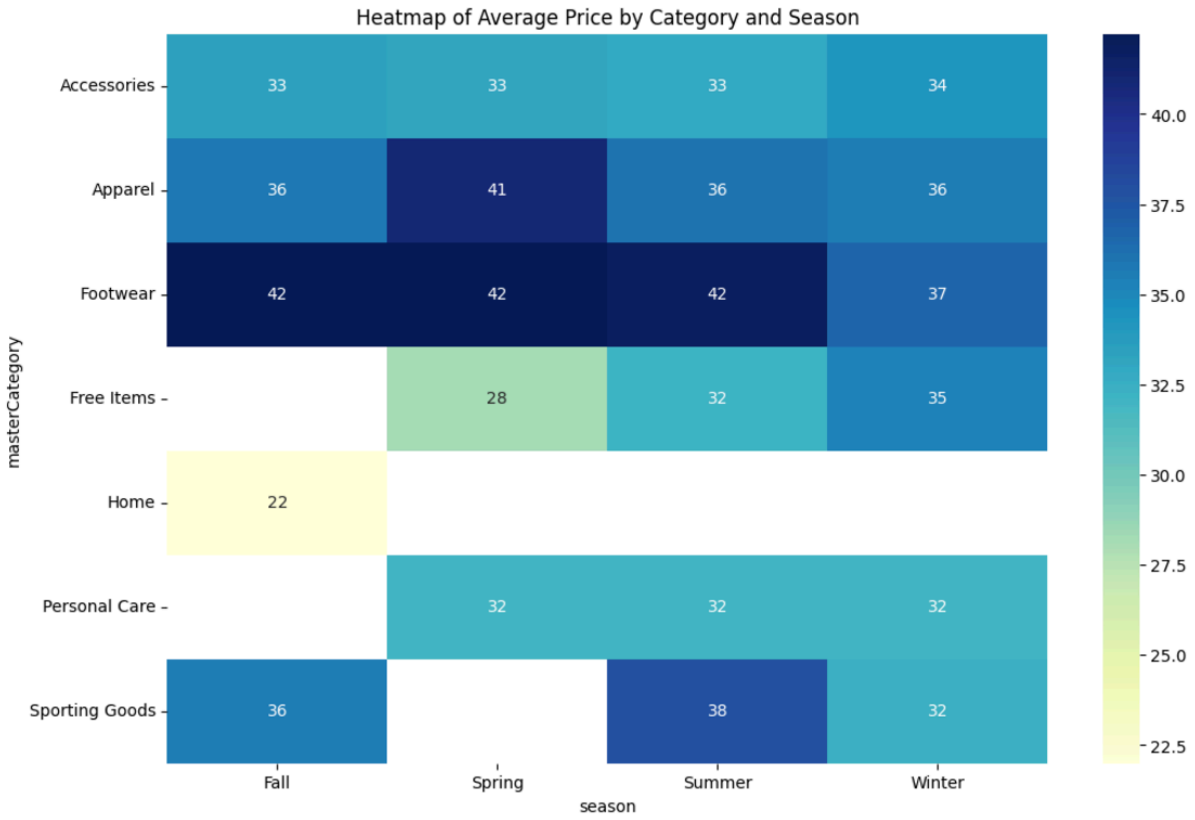


This chart showcases the distribution of product preferences based on baseColour. It emphasizes which colors are most commonly preferred or associated with the products in the dataset.

Clustering Analysis: Customer Segments:

- Features (Price (USD) and ratings) were standardized using StandardScaler to ensure uniform scaling. K-Means clustering was then applied with 4 clusters to group similar products based on price and customer ratings.
- A scatterplot visualizes the clusters, with standardized price and ratings as axes. Each cluster is distinguished by a unique color, providing insights into product segmentation and similarities.

Heatmap of Average Price by Category and Season:



- A pivot table was used to compute the average Price (USD) for each combination of masterCategory and season. This provides a structured view of how pricing varies across categories and seasons.
- The heatmap uses a YlGnBu color palette with annotated values, making it easy to spot trends, such as which seasons are associated with higher or lower prices for specific categories.

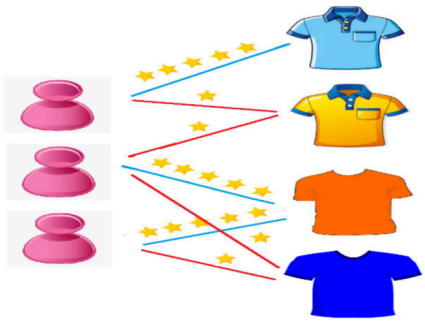
4.Model Building:

Traditional Recommendation Models

Content based filtering:

Content-based filtering is a recommendation technique that suggests items to users based on the characteristics of the items they have interacted with in the past. It focuses on the properties of items and user preferences without relying on interactions with other users.

The system computes the similarity between the user's profile and available items using similarity measures like cosine similarity.Items with the highest similarity scores are recommended to the user.



Collaborative Filtering:

This method recommends items based on user interaction data, such as ratings or purchase history. It is a recommendation technique that suggests items to users based on the preferences and behaviors of other users.

Identifies users who are similar to the target user based on their interactions with items.Recommends items that these similar users liked but the target user has not interacted with.Computes similarity between items based on how users interact with them.Recommends items that are similar to those the user has previously interacted with.

- **PMF :** Matrix Factorization (MF) is a collaborative filtering technique used in recommendation systems to predict user preferences based on past interactions.MF breaks down the user-item interaction matrix into two smaller matrices:

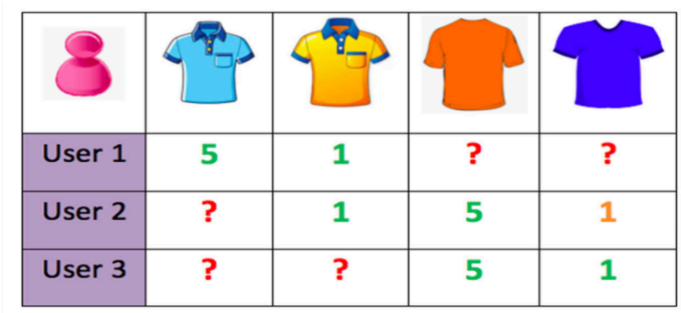
1. **User Latent Factors**: A matrix capturing the preferences of each user as a set of abstract features.
2. **Item Latent Factors**: A matrix capturing the characteristics of each clothing item.

The model predicts a user's preference for an item by taking the dot product of the corresponding user and item latent factors.

- **SVD :** SVD is another matrix factorization technique that simplifies the user-item interaction matrix by identifying the most important patterns in user behavior and item attributes.SVD can predict the likelihood that a user will select a specific clothing item based on the overall trends and patterns observed in the dataset.

  SVD factorizes the interaction matrix into three components:

1. **User Matrix** : Represents users as vectors of latent features.
2. **Singular Value Matrix** : Diagonal matrix indicating the significance of each latent feature. Larger values correspond to more critical features, such as preferences for specific clothing categories.
3. **Item Matrix** : Represents items as vectors of latent features.



User 1	5	1	?	?
User 2	?	1	5	1
User 3	?	?	5	1

DNN Model:

DNN and Hybrid Model:

The deep neural network (DNN) model is designed to leverage user preferences, item features, and contextual data to generate personalized clothing recommendations.

Architecture

1. Input Layer

The input layer processes multiple features to capture diverse information about users, items, and context.

- **User Features**:
  - user_id is embedded into a dense vector of latent features that represent user preferences (e.g., favorite categories or styles).

- **Item Features**:
    - id (item ID) is embedded to capture unique item characteristics (e.g., popularity, suitability for occasions).
    - Item-specific attributes like masterCategory, subCategory, articleType, and baseColour are embedded or one-hot encoded.
- **Contextual Features**:
    - gender, season, usage, and Month are embedded to represent contextual relationships (e.g., seasonal trends).
    - Continuous features like Price (USD) and year are normalized to ensure numerical stability.

2. Embedding Layers

- User, item, and contextual features are transformed into dense, low-dimensional embeddings.
- These embeddings capture latent patterns .

3. Feature Interaction Layer

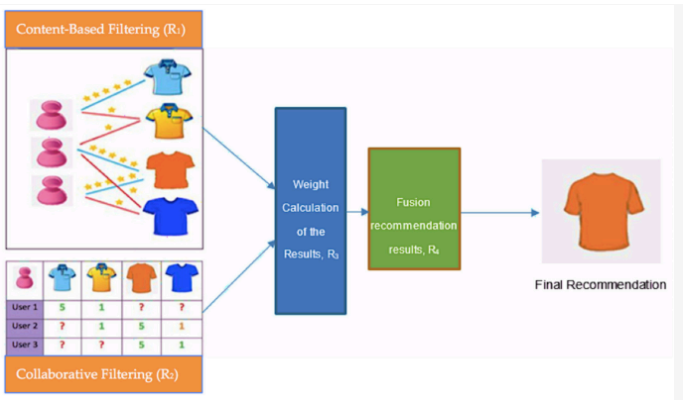
- Concatenates all embeddings into a single feature vector to model relationships between users, items, and context.

4. Hidden Layers

- Fully connected (dense) layers process the concatenated feature vector:
    - Multiple layers with decreasing units and ReLU activations.
    - These layers learn hierarchical representations of user-item-context interactions.
- **Regularization**:
    - Dropout layers reduce overfitting by randomly disabling neurons during training.
    - Batch normalization stabilizes learning and speeds up convergence.

5. Output Layer

- Produces a relevance score.
- Activation functions:
    - **Sigmoid**: For binary predictions.
    - **Linear**: For continuous predictions.



Training Details

1. **Data Preparation**:
   - The dataset contains columns such as user_id, id, articleType, baseColour, and interaction data (e.g., clicks, purchases).
   - Features are preprocessed:
     - **Categorical Features**: Encoded and embedded.
     - **Continuous Features**: Standardized using StandardScaler.
     - **Images**: Preprocessed using CNN-based feature extraction if required.
2. **Training Data**:
   - Positive samples: Actual interactions (e.g., a user purchased or clicked an item).
   - Negative samples: Randomly sampled non-interacted items to balance the dataset.
3. **Loss Function**:
   - **Binary Cross-Entropy Loss**: Measures the difference between predicted scores and true labels (1 for interacted, 0 for non-interacted items).
4. **Optimizer**:
   - **Adam Optimizer** is used for faster convergence with learning rate tuning.
5. **Training Configuration**:
   - **Batch Size**: 64–128 (varied based on data size).
   - **Epochs**: 10–20 (based on early stopping criteria).
   - **Validation Split**: 20% of data used for validation.
6. **Evaluation Metrics**:
   - **Precision@K**: Measures how many of the top-K recommendations are relevant.
   - **Recall@K**: Measures coverage of relevant items within the top-K recommendations.

Methodology

1. **Feature Representation**:
    - Content features (e.g., articleType, baseColour) are embedded and concatenated with user-item interaction embeddings to create a unified feature vector.

    For instance:

    User ID: 45601 → [0.12, 0.34, -0.22]

    Item ID: 34990 → [0.56, -0.42, 0.19]

    Content Features: [0.23, 0.41, 0.87, -0.15]

    Combined: [0.12, 0.34, -0.22, 0.56, -0.42, 0.19, 0.23, 0.41, 0.87, -0.15]

2. **Model Training**:
    - **Collaborative Signals**: User and item embeddings are learned from interaction data.
    - **Content Signals**: Feature embeddings (e.g.,articleType) are combined with collaborative embeddings to refine recommendations.
3. **Hybrid Learning**:
    - The model balances collaborative and content-based signals, learning from both user behavior patterns and item attributes.
    - Regularization techniques, such as dropout, are applied to embeddings and dense layers to prevent overfitting.
4. **Post-Processing Rules**:
    - Gender matching: Ensures recommendations align with the input item's gender.
    - Diversity: Enforces unique articleType recommendations within a recommendation set.
    - Complementary mapping: Suggests items that complement the input item, such as pairing a "Black Saree" with "Red Earrings."

Evaluation Metrics:

Evaluation metrics are critical in assessing the effectiveness of recommendation systems. They help quantify how well the model meets user expectations and ensures high-quality recommendations. Below are the commonly used metrics:

1. Recall@K:

- **Definition**: Recall@K measures the proportion of relevant items successfully recommended within the top *K* recommendations. It emphasizes retrieving all relevant items for a user.

Recall@K= (Number of relevant items in top K)/(Total number of relevant items)

- **Role**: Recall is especially important in scenarios where missing a relevant recommendation is more costly than recommending non-relevant items, such as product discovery.

2. Precision@K:

- **Definition**: Precision@K measures the proportion of relevant items among the top *K* recommendations.

Precision@K= (Number of relevant items in top K)/(Number of items in top K)

- **Role**: Precision helps ensure that the recommendations presented to the user are highly relevant, minimizing irrelevant suggestions.

3. F1@K:

- **Definition**: F1@K is the harmonic mean of Precision@K and Recall@K. It balances the trade-off between precision and recall to provide a single performance metric.
  F1@K= 2 * ((Precision@K*Recall@K)/(Precision@K+Recall@K))
- **Role**: F1@K is useful for evaluating models where both precision and recall are equally important.

Insights into Performance Outcomes

- **Content-Based Model**:
  High precision due to personalized recommendations, but recall may suffer if the model cannot generalize beyond the user's history.

- **Collaborative Filtering Model**:
  Strong recall, especially in dense datasets, as it leverages community-based patterns. Precision may drop if the dataset is sparse or user preferences are highly unique.
- **Hybrid Model**:
  Achieves a balance of precision and recall by combining content and collaborative approaches. The hybrid model demonstrated higher NDCG@K, indicating better ranking quality, and improved F1@K, showcasing its overall effectiveness.
-

Comparative Analysis:

1. Content-Based Filtering:

Content-based filtering relies on the attributes of items (e.g., articleType, baseColour, subCategory) to find recommendations that are similar to a user's preferences or an input item.

- **Strengths**:
  - Personalization: Provides highly relevant results for users by closely matching item features. For example, if the input item is a "Black Dress," the model might suggest other black dresses or dresses of similar styles.
  - Effective for cold-start items: Works well when no user interaction data is available, as it depends solely on item attributes.
  - High precision: Since recommendations are strictly based on item similarity, irrelevant suggestions are rare.
- **Weaknesses**:
  - Limited diversity: Recommendations are often too similar to the input item. For instance, suggesting multiple black dresses or items in the same subCategory without offering diversity.
  - Lack of collaborative insight: Fails to capture collective user behavior, which might introduce biases based on individual preferences.
- **Performance**:
  - **Precision@K**: High because the model targets closely related items.
  - **Recall@K**: Moderate due to the lack of variety in recommendations.
  - **Example Outcome**: Given an input "Grey Handbag," the system may recommend handbags of similar colors, missing complementary items like shoes or earrings.

2. Collaborative Filtering:

Collaborative filtering focuses on user-item interaction data. It uses techniques like matrix factorization or similarity measures to find patterns in user preferences and suggests items based on what similar users interacted with.

- **Strengths**:
    - Discovery of latent patterns: Finds relationships between users and items that are not apparent from attributes alone. For example, users who purchased "Blue Jeans" are also likely to buy "White Sneakers."
    - Effective for sparse datasets: By pooling interactions from multiple users, it generates recommendations even for items with limited individual interaction data.
- **Weaknesses**:
    - Cold-start problem: Struggles with recommending new items or for new users with no prior data.
    - Lower precision: The model sometimes recommends items that, while popular among similar users, may not match the input user's specific needs or preferences.
- **Performance**:
    - **Recall@K**: High, as it suggests a variety of items from collective user behavior.
    - **Precision@K**: Moderate, since some recommendations may not align with the user's immediate preferences.
    - **Example Outcome**: For a user who interacts with "Formal Shirts," the model might recommend "Leather Belts" and "Formal Shoes," capturing trends from similar users.

3. Hybrid Models with Deep Neural Networks (DNNs):

Hybrid models integrate content-based and collaborative filtering approaches using deep learning to create a robust recommendation framework. DNNs leverage embeddings to model complex relationships in user-item data and item features.

- **Strengths**:
    - Rich feature extraction: Embedding layers enable the model to capture nuanced patterns in the data, such as complementary relationships (e.g., recommending "Red Heels" and a "Black Handbag" for a "Black Saree").

    - Diverse and personalized recommendations: Combines user preferences, item attributes, and collaborative insights, providing well-rounded suggestions.
    - Solves cold-start issues: By generalizing embeddings, the model handles new items or users effectively.
    - Adaptable to constraints: Implements additional rules (e.g., gender matching, color contrast) for improved contextual relevance.
- **Weaknesses**:
    - Computational complexity: Requires significant computational resources for training and inference.
    - Dependency on data quality: Performance is tied to the quality and diversity of training data.
- **Performance**:
    - **Precision@K**: High due to the model's ability to learn detailed user-item relationships.
    - **Recall@K**: Very high as it balances diversity and relevance in recommendations.
    - **Example Outcome**: For a "Grey Saree" input, the model might suggest:
        - "Silver Earrings" (jewelry complementing the saree),
        - "Red Heels" (contrasting color footwear),
        - "Black Clutch" (neutral color accessory), creating a complete outfit.

Impact on User Experience

1. **Content-Based Filtering**:
    - Provides highly personalized but narrow recommendations.
    - Example: For a "Black Dress," the recommendations might include only dresses in similar styles and colors, which limits exploration.
2. **Collaborative Filtering**:
    - Enhances variety but lacks precision for specific user needs.
    - Example: Suggesting "Casual Shoes" and "Sports Watches" for a user who browses "Formal Shirts" may feel slightly off-context.
3. **Hybrid DNN Model**:
    - Offers a balanced mix of relevance and diversity.
    - Example: For an input "Red Kurta," it recommends:
        - "White Palazzos" (complementary item),
        - "Golden Earrings" (accessory),
        - "Brown Sandals" (matching footwear), providing a complete outfit.
    - Users enjoy a cohesive shopping experience, reducing effort in finding matching items.

Conclusion:

The recommendation model revolutionizes the shopping experience by combining the power of machine learning with the nuances of personal styling. By providing personalized, contextually appropriate outfit suggestions, it bridges the gap between affordability and high-quality fashion advice, making styling services accessible to all.

This system not only enhances user satisfaction through tailored recommendations but also empowers stakeholders with insights into consumer behavior, helping optimize inventory management, reduce costs, and improve customer loyalty.

Ultimately, the AI-powered styling tool represents a step forward in democratizing fashion, enabling users to confidently explore their unique styles while simplifying their decision-making process. It serves as a cost-effective, efficient, and innovative solution that benefits both users and businesses in today’s dynamic retail landscape.