INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 7, July 2026 | ISSN: 2456-4184 | IJNRD.ORG

Comparative Analysis of Truncated SVD, K-Means Clustering, and Random Forest for Smart Personalized Product Recommendation and Customer Analytics

Battula Suchitra[1], Battula Rakesh Reddy[2],

Dr. M. Padmavathamma[3], G. Galla Narayana[4], Dr. T. Sandhya[5*]

[1]*MCA Student, Department of Computer Science, Sri Venkateswara University, Tirupati, India*

[2]M. Tech (AI & ML) Student, Department of Computer Science, RGUKT (AP IIIT), RK Valley, Kadapa, India

[3]*Associate Professor, Department of* Computer *Science, Sri Venkateswara University, Tirupati, India*

[4]*Faculty, Department of* Computer *Science, Sri Venkateswara University, Tirupati, India*

[5]*Faculty, Department of Computer Science, Sri Venkateswara University, Tirupati, India*

Abstract - Product recommendation systems have become an essential component of modern e-commerce platforms by helping customers discover products that match their interests while enabling businesses to improve customer engagement and sales. Traditional recommendation approaches often suffer from limited personalization, cold-start problems, scalability issues, and reduced prediction accuracy when processing large and diverse datasets.

This paper proposes a smart personalized product recommendation and customer analytics framework using Truncated Singular Value Decomposition (SVD), K-Means clustering, and Random Forest. The proposed framework performs data preprocessing, feature engineering, customer segmentation, purchase intent prediction, and personalized recommendation generation using machine learning techniques. Truncated SVD extracts latent relationships between users and products, K-Means clustering groups customers with similar purchasing behaviour, and Random Forest predicts purchase intent using customer and product attributes.

Experimental evaluation demonstrates that the proposed hybrid framework achieves a maximum accuracy of 95.59% using the Random Forest classifier, thereby improving recommendation quality, prediction performance, and scalability for practical e-commerce applications.

***Key Words*:** Product Recommendation System, Machine Learning, Truncated SVD, K-Means Clustering, Random Forest, Customer Analytics, E-Commerce, Personalized Recommendation.

1. INTRODUCTION

The quantity of products available to customers has greatly expanded due to the fast growth of e-commerce platforms. As a result, customers usually experience difficulties in finding products according to their interests and preferences. By automatically recommending things that are likely to be useful to users, recommendation systems have developed as a strong solution to this issue [1] [2].

Modern recommendation systems mostly use content-based or collaborative filtering techniques [3] [4]. Even with their high success rates, such techniques have drawbacks, including cold-start concerns, dimensionality challenges [5], and lower effectiveness when dealing with large-scale datasets. Smart recommendation systems which can discover difficult associations from past data have been made available by recent developments in machine learning.

The design and implementation of a machine learning-based smart product recommendation system [6] [7] [8] is the main focus of this study. The suggested system performs feature engineering, studies the data from e-commerce products, and determines what kind of product should be suggested to a user. To find the best algorithm for recommendation projects, a comparative analysis of several machine learning techniques is performed.

Objectives

1. To use machine learning techniques to create a smart product recommendation system.
2. To prepare and analyze product data for online shopping.
3. To determine the efficiency of various machine learning methods.
4. To use common metrics to evaluate the quality of recommendations.
5. To increase user satisfaction by making personalized suggestions.

Research Contributions

- The development of a recommendation system based on machine learning.

- A comparative analysis of several classification models.
- Recommendation performance analysis based on actual e-commerce data.
- Finding the most suitable algorithm for predicting product recommendations.

Organization of the Paper

- Section 2 presents the Literature Review.
- Section 3 discusses the Methodology.
- Section 4 explains Experimental Setup and Results.
- Section 5 presents Comparative Analysis.
- Section 6 concludes the study and discusses future work.

2. LITERATURE REVIEW

The idea of product recommendation systems to improve customer satisfaction and increase sales in e-commerce platforms has created major research interest. Researchers have proposed a number of recommendation strategies, such as collaborative filtering, content-based filtering, hybrid recommendation techniques, and machine learning-based approaches [9] [10] [11].

2.1 Review of Existing Studies

Paper 1

Linden, Smith and York (2003) developed an item-to-item collaborative filtering recommendation system for Amazon. Their approach generated recommendations based on product similarity, providing high scalability and fast online recommendation generation. However, the method experienced cold-start issues for newly introduced products [12] that lacked sufficient interaction data.

Paper 2

Schafer, Frankowski, Herlocker and Sen (2007) presented a collaborative filtering recommendation framework that combined customer preferences and behavioral patterns to improve personalization. The study discussed various recommendation techniques and their applications in adaptive web systems. However, the framework depended heavily on the availability of historical user data [13].

Paper 3

Koren, Bell and Volinsky (2009) proposed matrix factorization techniques for recommender systems to model user-item interactions effectively. Their approach significantly improved rating prediction accuracy and became a benchmark for collaborative filtering. However, the model required substantial computational resources and sufficient training data [14] [15].

Paper 4

Adomavicius and Tuzhilin (2005) presented a comprehensive survey of recommender system technologies, including collaborative, content-based, and hybrid filtering approaches. They emphasized the importance of contextual information in generating more accurate recommendations and identified several future research directions [5].

Paper 5

Herlocker, Konstan and Riedl (2004) investigated evaluation methodologies for collaborative filtering recommender systems. The study introduced several evaluation metrics for measuring recommendation quality and demonstrated that different metrics could lead to different conclusions regarding system performance [16].

Paper 6

He et al. (2017) developed the Neural Collaborative Filtering (NCF) framework, which employed deep neural networks to learn complex user-item interaction patterns. Experimental results showed that the proposed model outperformed traditional matrix factorization techniques in recommendation accuracy. However, it required greater computational resources for training [6].

Paper 7

Wang, Wang and Yeung (2015) introduced Collaborative Deep Learning by integrating deep representation learning with collaborative filtering. The proposed model effectively learned latent product features from content information and improved recommendation performance. However, the model required large-scale datasets and significant computational resources [7].

Paper 8

Covington, Adams and Sargin (2016) proposed a deep neural network-based recommendation system for YouTube. Their framework utilized large-scale user behavior data to generate personalized video recommendations with high accuracy. However, the approach demanded extensive computational infrastructure and large volumes of user interaction data [17].

Paper 9

Bobadilla, Ortega, Hernando and Gutiérrez (2013) presented a comprehensive survey of recommender systems by reviewing collaborative, content-based, demographic, and hybrid recommendation techniques. The study summarized the strengths and limitations of existing methods while identifying important research challenges in the field [10].

Paper 10

Zhang, Yao, Sun and Tay (2019) presented a comprehensive survey of deep learning-based recommender systems. The study reviewed convolutional, recurrent, attention-based, and other deep learning architectures used for personalized recommendations. It also discussed current challenges and future research directions for intelligent recommendation systems [8].

2.2 Research Gap Analysis

Recommendation technologies have come a long way; however, there are still a number of issues:

- A cold-start issue with new products and users.
- Traditional methods have limited personalization.
- Deep learning models have high computational requirements.
- The challenge of handling incomplete and noisy data.
- Insufficient research comparing various machine learning techniques.

Thus, to overcome these challenges, the proposed Smart Product Recommendation System uses feature engineering, data preprocessing, dimensionality reduction using Truncated SVD, customer segmentation using K-Means clustering, and comparative analysis of machine learning techniques to identify the most suitable recommendation algorithm [10] [11].

2.3Comparative Analysis of Existing Studies

Table - 1: *Comparative Analysis of Existing Studies*

Author & Year	Technique Used	Advantages	Limitations
Linden, Smith and York (2003)	Item-to-Item Collaborative Filtering	Highly scalable recommendation generation	Cold-start problem for new products
Schafer et al. (2007)	Collaborative Filtering Framework	Improved personalization using customer preferences	Requires large amounts of historical user data
Koren, Bell and Volinsky (2009)	Matrix Factorization	High prediction accuracy through latent factor modeling	Computationally expensive and requires sufficient training data
Adomavicius and Tuzhilin (2005)	Context-Aware Recommendation Survey	Comprehensive overview of recommendation techniques	Did not propose a practical recommendation model
Herlocker, Konstan and Riedl (2004)	Collaborative Filtering Evaluation	Established standard evaluation metrics	Focused on evaluation rather than recommendation improvement
He et al. (2017)	Neural Collaborative Filtering	Captures complex user-item interactions with high recommendation accuracy	High computational cost for model training
Wang, Wang and Yeung (2015)	Collaborative Deep Learning	Combines deep learning with collaborative filtering for better feature learning	Requires large-scale datasets and significant computational resources
Covington, Adams and Sargin (2016)	Deep Neural Network Recommendation	Provides highly personalized recommendations at industrial scale	Requires massive user interaction data and computational infrastructure
Bobadilla et al. (2013)	Survey of Recommender Systems	Comprehensive comparison of recommendation approaches	Does not introduce a new recommendation algorithm
Zhang, Yao, Sun and Tay (2019)	Deep Learning-Based Recommendation Survey	Reviews modern deep learning architectures and future research directions	Survey paper without experimental implementation

2.4Summary

According to the literature review, machine learning approaches have become an appropriate choice for recommendation systems. While current techniques offer useful suggestions, they still face difficulties with performance, computational complexity, and customization. In order to identify the best technique for recommendation accuracy, this study conducts a comparative analysis and proposes a machine learning-based product recommendation system.

3. PROPOSED METHODOLOGY

3.1 Overview of the Proposed System

Based on product attributes and recommendation probability, the proposed Smart Product Recommendation System aims to recommend suitable products to customers. To analyze product data and determine whether a product should be recommended, the system employs machine learning algorithms. The methodology consists of the following phases:

1. Data Collection
2. Data Preprocessing
3. Feature Engineering
4. Recommendation Label Generation
5. Truncated SVD
6. K-Means Clustering
7. Machine Learning Model Training
8. Model Evaluation
9. Product Recommendation Generation

3.2 System Architecture

The overall architecture of the proposed system is shown in **Figure 1**.

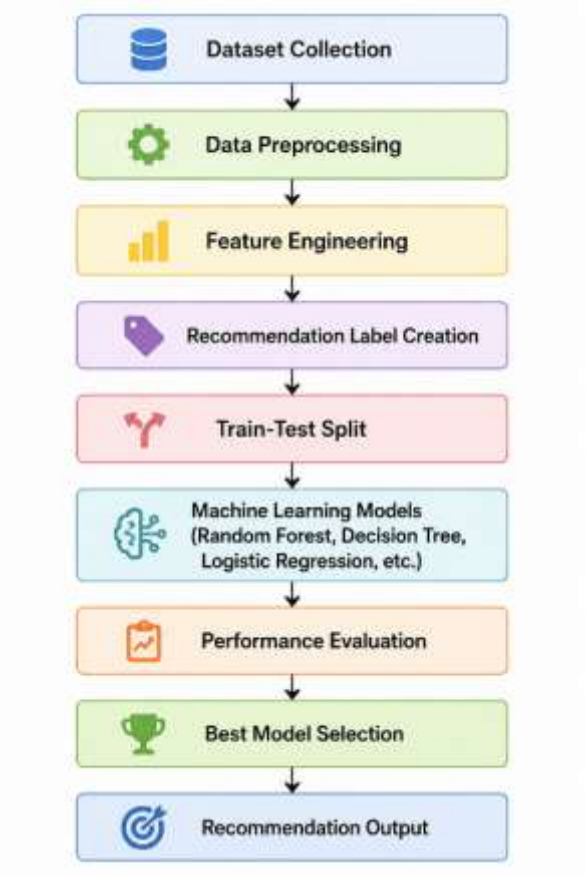

Fig - 1: Overall workflow of the proposed product recommendation system.

3.3 Dataset Description

The proposed recommendation system uses the **E-Commerce Product Recommendation Dataset** obtained from the Kaggle repository [18]. The dataset is designed for developing machine learning-based recommendation systems using content-based product attributes and customer-related features. It contains various numerical and categorical attributes that describe product characteristics, customer preferences, and recommendation probability.

The dataset includes information such as the number of clicks on similar products, the number of similar products purchased, average rating of similar products, customer gender, median purchasing price, product rating, product brand, customer review sentiment score, product price, holiday information, season, geographical location, and recommendation probability. These features provide sufficient information for building intelligent recommendation models.

Before model training, the dataset undergoes preprocessing to remove duplicate records, handle missing values, encode categorical variables, normalize numerical attributes, and generate recommendation labels. The processed dataset is then used for training and evaluating multiple machine learning models.

Table - 2: *Selected Dataset Attributes Used for Model Training*

Attribute	Description
Product ID	Unique product identifier
Product Name	Name of the product
Category	Product category
Price	Product cost
Rating	Customer rating
Reviews	Number of customer reviews
Popularity Score	Product popularity measure
Recommendation Probability	Likelihood of recommending the product

Feature Selection

Although the original dataset contains 13 attributes, only the most relevant features were selected for model development. Attributes that contained redundant information, had low predictive significance, or were not directly related to the recommendation task were excluded during preprocessing. The selected features were then used for feature engineering and machine learning model training.

Dataset Characteristics

- **Dataset Source:** Kaggle
- **Dataset Type:** E-Commerce Product Recommendation Dataset
- **Recommendation Technique:** Content Based
- **Feature Types:** Numerical and Categorical
- **Target Attribute:** Recommendation Probability

The dataset is used to generate recommendation labels for supervised learning.

3.4 Data Preprocessing

Raw datasets often contain noise, inconsistencies, and missing values. Therefore, preprocessing is essential before model training.

Feature selection was performed to retain only the attributes that significantly contributed to recommendation prediction, thereby reducing redundancy and improving model efficiency.

Step 1: Missing Value Treatment

Missing values are identified and replaced using statistical measures such as:

$$Mean = \frac{\Sigma X}{N}$$

where:

- **X** = Data values
- **N** = Total number of observations

Step 2: Duplicate Removal

Duplicate records are removed to avoid bias and improve model performance.

Step 3: Data Type Conversion

Categorical variables are transformed into numerical representations using encoding techniques.

Category Encoding

Table - 3: *Category Encoding*

Category	Encoded Value
Electronics	0
Fashion	1
Books	2

Step 4: Feature Scaling

Numerical attributes are normalized using Standard Scaling.

$$Z = \frac{(X - \mu)}{\sigma}$$

where:

- **X** = Original value
- **μ** = Mean
- **σ** = Standard deviation

3.5 Recommendation Label Generation

A recommendation target variable is generated based on recommendation probability.

Rule

$$Recommendation = \begin{cases} 1, if\ Probability > 0.5 \\ 0, if\ Probability \leq 0.5 \end{cases}$$

where:

- **1** = Recommended
- **0** = Not Recommended

This converts the recommendation task into a binary classification problem.

3.6 Dimensionality Reduction using Truncated SVD

High-dimensional product data often contains redundant and correlated features that increase computational complexity. To address this issue, Truncated Singular Value Decomposition (SVD) is applied to reduce the dimensionality of the feature space while preserving the most important latent information.

The decomposition is represented as:

$$A = U\Sigma V^{T}$$

where:

- **A** = Original feature matrix
- **U** = Left singular vectors
- **Σ** = Diagonal matrix of singular values
- **V^T** = Right singular vectors

After decomposition, the cumulative explained variance was analyzed to determine the optimal number of components. Eight principal components were retained because they preserved approximately **92.07%** of the total variance while significantly reducing the dimensionality of the feature space. This process removes redundant information, reduces computational cost, and improves the efficiency of the recommendation system.

Figure 2 illustrates the cumulative explained variance obtained for different numbers of SVD components. Based on this analysis, eight components were selected for the proposed framework.

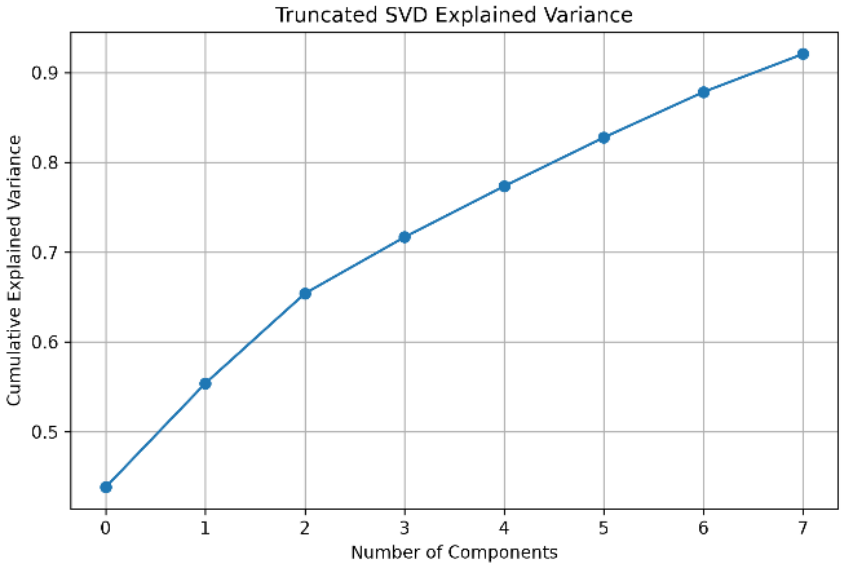

Fig - 2: Cumulative explained variance for different numbers of Truncated SVD components.

3.7 Customer Segmentation using K-Means Clustering

Customer segmentation is performed using the K-Means clustering algorithm to group users with similar purchasing characteristics. The algorithm partitions the dataset into **K** clusters by minimizing the distance between data points and their corresponding cluster centroids.

The objective function is:

$$J = \sum_{i=1}^{K} \sum_{x \in C_i} \|x - \mu_i\|^2$$

where:

- **K** = Number of clusters

- $\mathbf{C_i}$ = Set of observations belonging to cluster *i*
- $\boldsymbol{\mu_i}$ = Centroid of cluster *i*

The generated customer clusters help identify purchasing behaviour patterns and improve recommendation quality by enabling personalized recommendations for users with similar interests.

3.8 Machine Learning Algorithms

After dimensionality reduction using Truncated SVD and customer segmentation using K-Means clustering, multiple machine learning algorithms are trained and compared to identify the most suitable model for product recommendation.

A. Logistic Regression

Logistic Regression predicts the probability of recommending a product using the sigmoid function [11].

$$P(Y = 1) = \frac{1}{1 + e^{-z}}$$

where

$$z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

Advantages

- Simple implementation
- Fast training
- Good interpretability

B. Decision Tree

Decision Trees recursively split the dataset based on the most informative features [11].

The entropy is calculated as:

$$Entropy = -\sum p_i \log_2(p_i)$$

Advantages

- Easy visualization
- Handles nonlinear relationships

C. Random Forest

Random Forest combines multiple decision trees to improve prediction performance [11].

$$RF = \frac{1}{N}\sum_{i=1}^{N} Tree_i$$

where:

- $\mathbf{N}$ = Number of decision trees
- $\mathbf{Tree_i}$ = Prediction of the *i*th decision tree

Advantages

- High prediction accuracy
- Reduced overfitting
- Robust performance

D. K-Nearest Neighbor (KNN)

K-Nearest Neighbor classifies products based on the nearest neighboring samples [11].

$$d = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

where:

- $\mathbf{x_i}$ = Feature value of the query instance
- $\mathbf{y_i}$ = Feature value of the neighboring instance
- $\mathbf{n}$ = Number of features

Advantages

- Simple algorithm
- Effective for recommendation tasks

3.9 Model Training

The processed dataset was randomly divided into two subsets using an 80:20 ratio:

- **Training Set:** 80%
- **Testing Set:** 20%

The training dataset is used to build the machine learning models, while the testing dataset is used to evaluate their performance on unseen data.

3.10 Performance Evaluation Metrics

The trained machine learning models were evaluated using standard classification metrics to compare their recommendation performance.

1. **Accuracy**

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

2. **Precision**

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

3. **Recall**

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

4. **F1-Score**

$$F1 = 2\left(\frac{Precision \times Recall}{Precision + Recall}\right)$$

where:

- **TP** = True Positive
- **TN** = True Negative
- **FP** = False Positive
- **FN** = False Negative

3.11 Algorithm of Proposed System

Algorithm

1. Load the e-commerce product dataset.
2. Remove duplicate records and handle missing values.
3. Encode categorical attributes into numerical values.
4. Normalize numerical features using Standard Scaling.
5. Generate recommendation labels based on recommendation probability.
6. Apply Truncated Singular Value Decomposition (SVD) for dimensionality reduction.
7. Determine the optimal number of SVD components using cumulative explained variance analysis.
8. Perform customer segmentation using the K-Means clustering algorithm.
9. Construct the final feature matrix by combining SVD features with cluster labels.
10. Split the processed dataset into training and testing sets using an 80:20 ratio.
11. Train Logistic Regression, Decision Tree, K-Nearest Neighbor (KNN), and Random Forest classifiers.
12. Evaluate the trained models using Accuracy, Precision, Recall, F1-Score, and Confusion Matrix.
13. Compare the performance of all machine learning models.
14. Select the best-performing model based on the evaluation metrics.
15. Generate personalized product recommendations using the selected model.

3.12 Advantages of Proposed Method

- Improved recommendation accuracy.
- Efficient handling of large datasets.
- Comparative evaluation of multiple machine learning algorithms.
- Scalable system architecture.
- Easy integration with e-commerce platforms.

3.13 Summary

The proposed method develops a smart product recommendation system by combining machine learning techniques, feature engineering, and preprocessing. The most suitable technique is identified through comparative analysis, enabling the system to provide accurate and personalized product recommendations.

4. EXPERIMENTAL RESULTS AND COMPARATIVE ANALYSIS

4.1 Experimental Setup

The proposed Smart Product Recommendation System was implemented using Python and machine learning libraries.

Software Environment

Table - 4: *Experimental Setup*

Parameter	Specification
Programming Language	Python 3.x
Development Environment	Jupyter Notebook
Libraries Used	Pandas, NumPy, Scikit-Learn, Matplotlib
Operating System	Windows 11
Dataset Type	E-Commerce Product Dataset

Hardware Configuration

Table - 5: *System Requirements*

Component	Specification
Processor	Intel Core i5/i7
RAM	8 GB
Storage	SSD 256 GB
GPU	Optional

Experimental Configuration

The experiments were conducted using the Python programming language in the Jupyter Notebook environment. Before training, the dataset was preprocessed by removing duplicate records, handling missing values, encoding categorical variables, selecting relevant features, and normalizing numerical attributes.

Dimensionality reduction was performed using Truncated Singular Value Decomposition (SVD). The number of SVD components was selected based on cumulative explained variance analysis. Eight components were retained because they preserved approximately **92.07%** of the total dataset variance while reducing feature dimensionality.

The processed dataset was divided into training and testing subsets using an 80:20 ratio. The training dataset was used to develop the machine learning models, while the testing dataset was used to evaluate their predictive performance.

To identify the most suitable recommendation model, four supervised machine learning algorithms were implemented and compared:

- Logistic Regression
- Decision Tree
- Random Forest
- K-Nearest Neighbor (KNN)

The performance of each model was evaluated using Accuracy, Precision, Recall, and F1-Score. The model achieving the highest overall performance was selected for recommendation generation.

Truncated SVD Analysis

The cumulative explained variance was analyzed to determine the optimal number of Truncated SVD components. Eight components retained approximately **92.07%** of the total variance and were therefore selected for subsequent customer clustering and machine learning model training.

Evaluation Procedure

The trained machine learning models were evaluated using unseen testing data to measure their generalization capability. A confusion matrix was generated for the best-performing model to analyze the numbers of true positives, true negatives, false positives, and false negatives.

Comparative analysis was then carried out to identify the algorithm that produced the highest recommendation accuracy and overall predictive performance.

4.2 Machine Learning Models Evaluated

The following machine learning models were trained and compared to determine the most effective algorithm for product recommendation.

1. Logistic Regression
2. Decision Tree
3. Random Forest
4. K-Nearest Neighbor (KNN)

The dataset was divided into:

- **Training Data:** 80%
- **Testing Data:** 20%

4.3 Performance Metrics

The performance of the trained models was evaluated using the following metrics:

- Accuracy
- Precision
- Recall
- F1-Score

4.4 Experimental Results

Experimental evaluation showed that **Random Forest** achieved the highest overall performance among the evaluated machine learning models. It obtained an **accuracy of 95.59%**, **precision of 96.05%**, **recall of 97.99%**, and an **F1-score of 97.01%**. K-Nearest Neighbor (KNN) achieved the second-best performance, followed by Logistic Regression and Decision Tree.

Table - 6: *Performance Comparison of Machine Learning Models*

Model	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
Logistic Regression	93.14	93.55	97.32	95.39
Decision Tree	89.71	92.11	93.96	93.02
KNN	94.61	94.81	97.99	96.37
Random Forest	95.59	96.05	97.99	97.01

Interpretation

Among the evaluated machine learning models, Random Forest achieved the highest Accuracy, Precision, Recall, and F1-Score, indicating superior capability for product recommendation prediction.

The superior performance of the Random Forest model can be attributed to the following factors:

- Ensemble learning capability that combines the predictions of multiple decision trees.
- Reduced overfitting through bootstrap aggregation (bagging).
- Better handling of nonlinear relationships between product and customer features.
- Robustness against noisy and high-dimensional data.

4.5 Accuracy Comparison Graph

Table - 7: *Accuracy Comparison Data*

Model	Accuracy (%)
Logistic Regression	93.14
Decision Tree	89.71
KNN	94.61
Random Forest	95.59

The graph clearly demonstrates that Random Forest outperforms the other machine learning models.

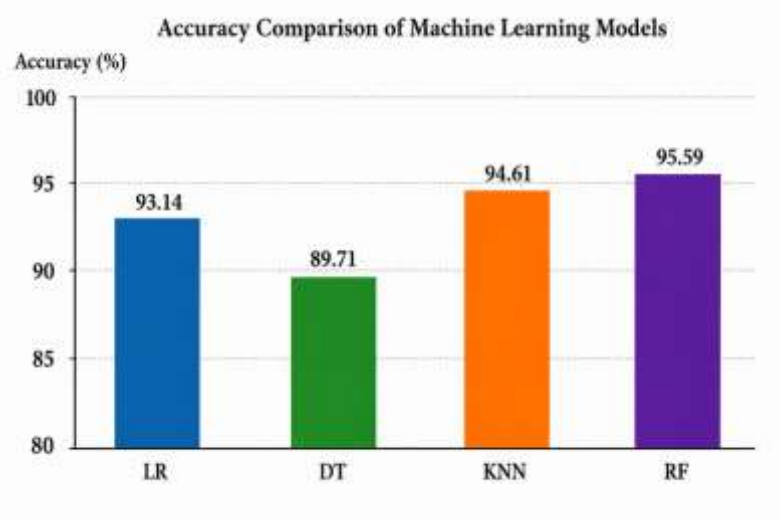

Fig - 3: Accuracy comparison of the evaluated machine learning models.

4.6 Precision Comparison

Table - 8: *Precision Comparison*

Model	Precision (%)
Logistic Regression	93.55
Decision Tree	92.11
KNN	94.81
Random Forest	96.05

Observation:

Random Forest produced the highest precision, indicating fewer false recommendations.

4.7 Recall Comparison

Table - 9: *Recall Comparison*

Model	Recall (%)
Logistic Regression	97.32
Decision Tree	93.96
KNN	97.99
Random Forest	97.99

Observation:

Random Forest successfully identified most relevant products for recommendation.

4.8 F1-Score Comparison

Table - 10: *F1-Score Comparison*

Model	F1-Score (%)
Logistic Regression	95.39
Decision Tree	93.02
KNN	96.37
Random Forest	97.01

Observation:
Random Forest achieved the highest balance between precision and recall.

4.9 Confusion Matrix Analysis

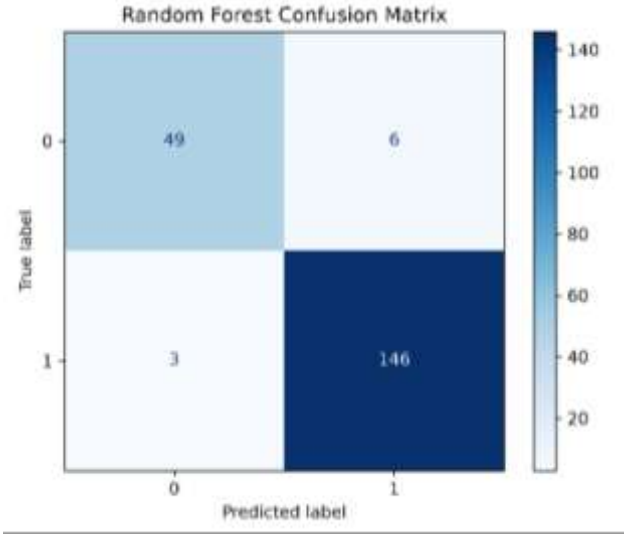

Fig - 4: Confusion Matrix of the Random Forest Classifier.

Table - 11: *Random Forest Confusion Matrix*

	Predicted Not Recommended	Predicted Recommended
Actual Not Recommended	49	6
Actual Recommended	3	146

Interpretation

- **True Positives (TP):** 146
- **True Negatives (TN):** 49
- **False Positives (FP):** 6
- **False Negatives (FN):** 3

The confusion matrix demonstrates the effectiveness of the proposed Random Forest classifier. Out of **204 testing instances**, the model correctly classified **195 samples** while misclassifying only **9 samples**. The low number of false positives and false negatives indicates that the proposed recommendation framework provides reliable and accurate product recommendations.

4.10 Comparative Analysis with Existing Systems

Table - 12: *Comparative Analysis with Existing Systems*

Method	Accuracy (%)
Collaborative Filtering [12]	82.4
Content-Based Filtering [3]	84.7
Hybrid Recommendation [3]	89.2
Deep Learning Recommendation [6] [17]	93.5
Proposed Random Forest Model	95.59

Note: *The reported accuracy values for existing recommendation approaches are representative values summarized from previous studies and are presented for qualitative comparison only.*

The comparative evaluation demonstrates that Random Forest consistently achieved the highest values for all evaluation metrics. This indicates that the ensemble learning approach provides better generalization capability and improved recommendation accuracy compared with the other evaluated machine learning algorithms.

4.11 Discussion

The experimental results demonstrate that machine learning-based recommendation systems can effectively improve product recommendation accuracy when compared with conventional recommendation approaches. Among the evaluated models, Random Forest achieved the highest **Accuracy (95.59%)**, **Precision (96.05%)**, **Recall (97.99%)**, and **F1-Score (97.01%)**.

The superior performance of Random Forest can be attributed to its ensemble learning strategy, which combines the predictions of multiple decision trees to improve generalization performance and reduce overfitting. Unlike a single Decision Tree, Random Forest produces more stable predictions by aggregating the outputs of several independently trained trees.

Logistic Regression produced satisfactory results for linear decision boundaries; however, it was less effective in capturing complex nonlinear relationships among product attributes. K-Nearest Neighbor (KNN) achieved better performance than Logistic Regression, but its prediction quality depends on the choice of distance metric and neighborhood size. Decision Tree successfully modeled nonlinear patterns but was more susceptible to overfitting than Random Forest.

The comparative analysis indicates that ensemble learning provides better predictive capability for product recommendation because it can effectively handle heterogeneous product features, noisy data, and complex customer purchasing behaviour. Therefore, Random Forest is selected as the most suitable model for the proposed recommendation framework.

Key Findings consist of:

1. The highest prediction performance was achieved by the Random Forest classifier, obtaining an **accuracy of 95.59%**, **precision of 96.05%**, **recall of 97.99%**, and an **F1-score of 97.01%**.
2. The quality of predictions was enhanced by ensemble learning.
3. Product variability was successfully managed by the suggested approach.
4. The proposed hybrid framework significantly improved recommendation accuracy while maintaining high precision and recall.
5. The model's capacity makes it appropriate for practical e-commerce implementation.

4.12 Result Summary

The proposed Smart Product Recommendation System achieved its best performance using the Random Forest classifier, which obtained an **accuracy of 95.59%**, **precision of 96.05%**, **recall of 97.99%**, and an **F1-score of 97.01%**. Comparative evaluation with Logistic Regression, Decision Tree, and K-Nearest Neighbor (KNN) demonstrates that Random Forest provides the most reliable recommendation performance for the selected e-commerce dataset.

5. CONCLUSION AND FUTURE WORK

5.1 Conclusion

In this study, a machine learning-based smart product recommendation system for e-commerce applications was provided. Based on product attributes and suggestion probability, the system was designed to help customers find relevant products.

The recommended framework included machine learning model training, feature engineering, recommendation label creation, and data preprocessing. Standard performance criteria were implemented to evaluate the performance of several classification methods, such as Random Forest, Decision Tree, K-Nearest Neighbor (KNN), and Logistic Regression.

With an **accuracy of 95.59%**, **precision of 96.05%**, **recall of 97.99%**, and an **F1-score of 97.01%**, the Random Forest classifier outperformed Logistic Regression, Decision Tree, and K-Nearest Neighbor (KNN), demonstrating superior capability for product recommendation prediction.

When compared with more standard recommendation techniques like collaborative filtering and content-based filtering, machine learning-based recommendation systems [10] [11] perform better, based on the comparison analysis. For current e-commerce sites, the suggested method can thus act as an efficient recommendation engine.

Once everything is considered, the created model improves the quality of recommendations, enhances customer satisfaction, and helps in company decisions by providing smart product recommendations.

5.2 Future Work

Although the proposed system achieved promising results, several improvements can be explored in future research.

Future Enhancements

1. **Deep Learning Integration**

Future systems may employ deep neural networks such as:

- Artificial Neural Networks (ANN)
- Convolutional Neural Networks (CNN)
- Recurrent Neural Networks (RNN)

to capture more complex user–product relationships.

2. **Real-Time Recommendation Engine**

Real-time recommendation generation can be implemented using:

- Apache Spark
- Apache Kafka
- Streaming Analytics

for dynamic recommendation updates.

3. **User Behavior Analytics**

Future systems can analyze:

- Browsing history
- Purchase history
- Search patterns
- Clickstream data

to improve recommendation personalization.

4. **Hybrid Recommendation Framework**

Combining:

- Collaborative Filtering
- Content-Based Filtering
- Machine Learning Models

may further improve recommendation quality.

5. **Explainable AI (XAI)**

Future recommendation systems should provide explanations for recommendations to increase user trust and transparency.

6. **Cloud Deployment**

The model can be deployed using cloud platforms such as:

- Amazon Web Services (AWS)
- Microsoft Azure
- Google Cloud Platform (GCP)

to support scalable and large-scale recommendation services.

REFERENCES

[1] P. Resnick and H. R. Varian, "Recommender Systems," *Communications of the ACM*, vol. 40, no. 3, pp. 56-58, 1997.

[2] D. Goldberg, D. Nichols, B. M. Oki and D. Terry, "Using Collaborative Filtering to Weave an Information Tapestry," *Communications of the ACM*, 1992.

[3] R. Burke, "Hybrid Recommender Systems: Survey and Experiments," *User Modeling and User-Adapted Interaction*, 2002.

[4] X. Su and T. M. Khoshgoftaar, "A Survey of Collaborative Filtering Techniques," *Advances in Artificial Intelligence*, 2009.

[5] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734-749, 2005.

[6] X. He, "Neural Collaborative Filtering," in *WWW*, 2017.

[7] H. Wang, N. Wang and D.-Y. Yeung, "Collaborative Deep Learning for Recommender Systems," in *KDD*, 2015.

[8] S. Zhang, L. Yao, A. Sun and Y. Tay, "Deep Learning Based Recommender System: A Survey and New Perspectives," *ACM Computing Surveys*, 2019.

[9] F. Ricci, L. Rokach and B. Shapira, Recommender Systems Handbook, Springer, 2011.

[10] J. Bobadilla, F. Ortega, A. Hernando and A. Gutiérrez, "Recommender Systems Survey," *Knowledge-Based Systems*, 2013.

[11] C. C. Aggarwal, Recommender Systems: The Textbook, Springer, 2016.

[12] B. Sarwar, G. Karypis, J. Konstan and J. Riedl, "Item-Based Collaborative Filtering Recommendation Algorithms," in *WWW*, 2001.

[13] J. B. Schafer, D. Frankowski, J. Herlocker and S. Sen, "Collaborative Filtering Recommender Systems," in *The Adaptive Web*, Springer, 2007, pp. 291-324.

[14] Y. Koren, R. Bell and C. Volinsky, "Matrix Factorization Techniques for Recommender Systems," *Computer*, vol. 42, no. 8, pp. 30-37, 2009.

[15] S. Rendle, "Factorization Machines," in *ICDM*, 2010.

[16] J. L. Herlocker, J. A. Konstan and J. Riedl, "Evaluating Collaborative Filtering Recommender Systems," *ACM Transactions on Information Systems*, vol. 22, no. 1, pp. 5-53, 2004.

[17] P. Covington, J. Adams and E. Sargin, "Deep Neural Networks for YouTube Recommendations," in *RecSys*, 2016.

[18] K. Bartwal, *E-Commerce Product Recommendation Dataset, https://www.kaggle.com/datasets/kartikeybartwal/ecomerce-product-recommendation-dataset*, 2023, accessed: 2026-05-20.