**AMAZON RECOMMENDATION SYSTEM**

Project submitted to the

SRM University – AP, Andhra Pradesh

for the partial fulfilment of the requirements to award the degree of

**Bachelor of Technology**

In

**Computer Science and Engineering**

**School of Engineering and Sciences**

**Submitted by**

Prafull Raj | AP21110011016

Atharva Narkhede | AP21110011028

Rahul Bajaj | AP21110011022

Vrijeshwar Singh | AP21110010922

Aditya Dubey | AP21110010729

**A picture containing text

Description automatically generated**

Under the Guidance of

**Murali Krishna Enduri**

**SRM University–AP**

**Neerukonda, Mangalagiri, Guntur**

**Andhra Pradesh – 522 240**

**APPLIED DATA SCIENCE LAB PROJECT**

TOPIC **-AMAZON RECOMMENDATION SYSTEM**

**1.PROBLEM STATEMENT**

1. **Introduction**

In the contemporary retail and e-commerce landscape, businesses are constantly striving to optimize customer experiences and drive sales through effective product recommendations. Traditional recommendation systems often lack personalization and fail to capture the nuanced relationships between products. This report delves into the application of Market Basket Analysis (MBA) as a data science solution to enhance product recommendations and address the evolving needs of businesses in the retail and e-commerce sectors.

1. **Problem Statement**

The problem revolves around the inefficiency of existing product recommendation systems in providing personalized suggestions to customers. Businesses face the challenge of understanding and leveraging the intricate patterns within transactional data to offer relevant and compelling product recommendations. This problem necessitates a data-driven approach that can harness the power of association rule mining and customer segmentation techniques to deliver tailored product suggestions that resonate with individual preferences and purchasing behaviors.

1. **Data Science Application**

Market Basket Analysis emerges as a pertinent data science application to tackle the problem. By leveraging transactional data encompassing customer purchase histories, MBA enables businesses to:

Uncover intricate associations and patterns between products frequently purchased together.

Utilize association rule mining algorithms such as Apriori or FP-Growth to generate actionable insights.

Implement personalized recommendation systems that cater to individual customer preferences and enhance the overall shopping experience.

In the subsequent sections of this report, we will delve deeper into each aspect of Market Basket Analysis and its implications for enhancing product recommendations and driving business success in the retail and e-commerce sectors

**2. DATA PREPROCESSING**

The collected data is pre-processed to ensure its suitability for further analysis. This may involve cleaning the data, handling missing values and transforming variables.

Here are the steps for data preprocessing:

**Data Collection:** Obtain transactional data from the dataset, such as information about customer purchases, such as user IDs, product IDs, and quantities purchased.

**Data Cleaning:**

Remove duplicate transactions: Eliminate duplicate records to ensure that each transaction is unique.

Handle missing values: Check for missing values in user IDs, product IDs, or other essential fields and handle them appropriately, such as imputation or removal.

**Data Transformation:**

Transaction-level data: Convert the dataset into transaction-level format, where each row represents a single transaction and contains a list of items purchased.

Basket-level data: Aggregate transaction-level data to create basket-level data, where each row represents a unique combination of items purchased together.

**Data Encoding:**

One-Hot Encoding: Convert categorical variables like product IDs into binary vectors using one-hot encoding. Each column represents a unique item, and a value of 1 indicates that the item was present in the transaction, while 0 indicates absence.

Transaction ID encoding: Optionally encode user IDs to facilitate tracking and analysis.

**Transaction Filtering:**

Remove low-support items: Exclude items with low support as they may not contribute significantly to the analysis and may introduce noise.

Filter out irrelevant items: Exclude items that are not relevant to the analysis or are considered outliers.

**Association Rule Mining:**

Use algorithms like Apriori or FP-Growth to mine association rules from the preprocessed dataset.

Define parameters such as minimum support, minimum confidence, and maximum length of the itemset to control the quality and quantity of generated rules.

**Rule Filtering:**

Filter rules based on support, confidence, and lift: Remove rules that do not meet predefined thresholds for support, confidence, or lift to focus on the most significant associations.

**Prune redundant rules:** Eliminate redundant rules that convey similar information to reduce the rule set's size and improve interpretability.

Interpretation and Visualization:

Interpret and analyze the generated association rules to gain insights into customer purchasing patterns, cross-selling opportunities, and market trends.

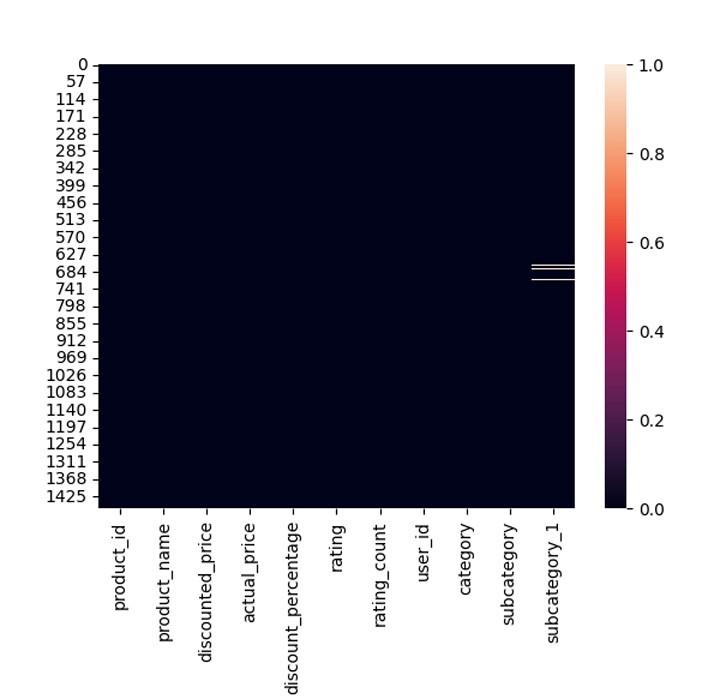
Visualize association rules using techniques such as scatter plots, heatmaps, or network graphs to facilitate interpretation and communication of findings.

**Evaluation:**

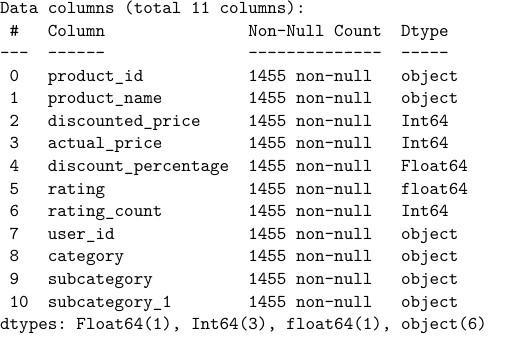
Evaluate the performance of the association rules based on metrics such as lift, confidence, and support.

Validate the discovered associations using domain knowledge and expert judgment.

**Visualizing the data:**



**Preprocessed data:**



**3. EXPLORATORY DATA ANALYSIS**

EDA involves analyzing and visualizing the data to gain insights and understand its underlying patterns. Here are the steps involved in EDA:

**Summary Statistics:**

Summary statistics provide an overview of numerical columns in the dataset. These statistics include measures such as mean, median, standard deviation, minimum, maximum, and quartiles. They help in understanding the central tendency, dispersion, and shape of the distribution of numerical variables.

**Distribution of Numerical Variables:**

Visualizing the distribution of numerical variables allows us to understand their spread and shape. Histograms, density plots, and box plots are common visualization techniques used for this purpose. Understanding the distribution can help identify outliers, skewness, and potential data issues.

**Correlation Analysis:**

Correlation analysis examines the relationship between numerical variables. Correlation coefficients such as Pearson's correlation coefficient can indicate the strength and direction of the relationship. Heatmaps and scatter plots are commonly used to visualize correlations. This analysis helps in identifying variables that are highly correlated or potentially redundant.

**Categorical Variables Analysis:**

For categorical variables, EDA involves exploring their frequency counts, proportions, and distributions. Bar plots, pie charts, and frequency tables are commonly used for visualizing categorical data. This analysis helps in understanding the distribution of categories and identifying any imbalances or patterns.

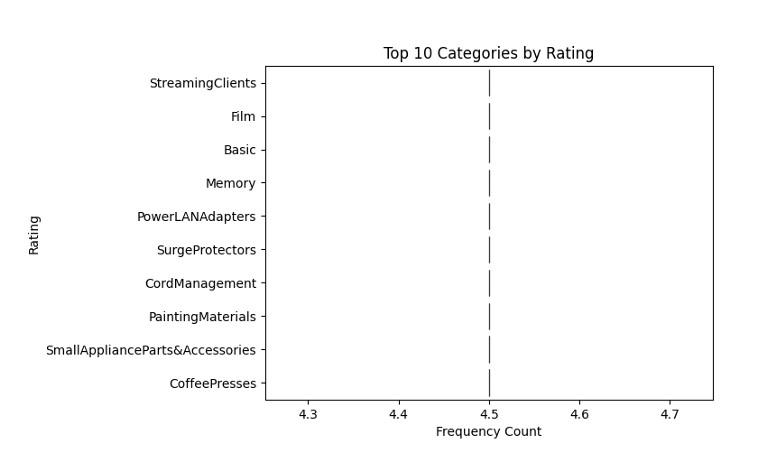
**Text Analysis:**

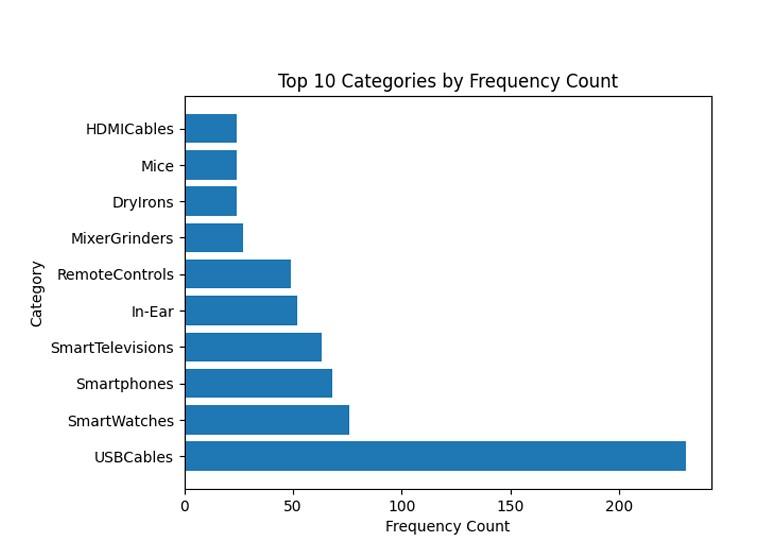
Text analysis involves processing and analyzing text data, which can include product descriptions, reviews, or any other textual information. Techniques such as word frequency analysis, sentiment analysis, and topic modeling can be applied to extract insights from text data. This analysis provides valuable insights into customer sentiments, product features, and trends.

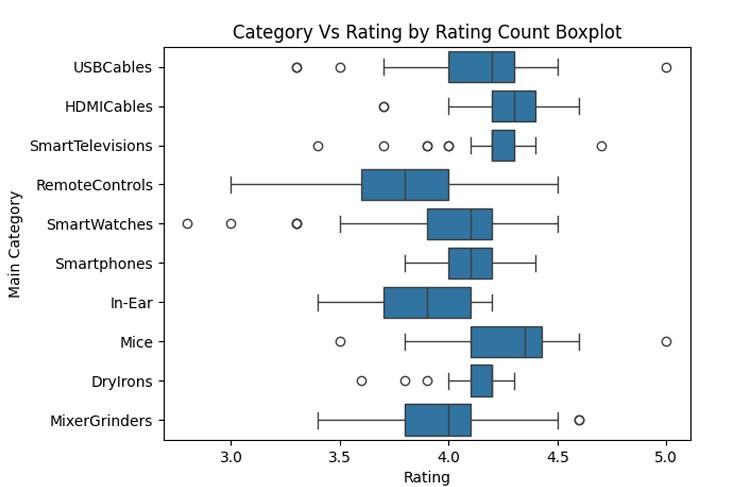
| Information about the dataset:  <class 'pandas.core.frame.DataFrame'> RangeIndex: 1465 entries, 0 to 1464 Data columns (total 16 columns): | | | | |  |
| --- | --- | --- | --- | --- | --- |
| # | Column | | | Non-Null Count | Dtype |
| --- | ------ | | | -------------- | ----- |
| 0 | product\_id | | | 1465 non-null | object |
| 1 | product\_name | | | 1465 non-null | object |
| 2 | category | | | 1465 non-null | object |
| 3 | discounted\_price | | | 1465 non-null | object |
| 4 | actual\_price | | | 1465 non-null | object |
| 5 | discount\_percentage | | | 1465 non-null | object |
| 6 | rating | | | 1465 non-null | object |
| 7 | rating\_count | | | 1463 non-null | object |
| 8 | about\_product | | | 1465 non-null | object |
| 9 | user\_id | | | 1465 non-null | object |
| 10 | user\_name | | | 1465 non-null | object |
| 11 | review\_id | | | 1465 non-null | object |
| 12 | review\_title | | | 1465 non-null | object |
| 13 | review\_content | | | 1465 non-null | object |
| 14 | img\_link | | | 1465 non-null | object |
| 15 | product\_link | | | 1465 non-null | object |
| dtypes: object(16) memory usage: 183.3+ KB  None  Missing values:  product\_id 0 product\_name 0 category 0 discounted\_price 0 actual\_price 0 discount\_percentage 0 rating 0 rating\_count 2 about\_product 0 user\_id 0 user\_name 0 review\_id 0 review\_title 0 | | | |
| review\_content | | 0 |
| img\_link | | 0 |
| product\_link dtype: int64 | | 0 |

Duplicate rows:

0







**4. FEATURE SELECTION AND FEATURE GENERATION**

Feature selection involves choosing relevant attributes or columns from the dataset for analysis or modeling. In this code:

1. Selection of Numeric Columns: The code identifies specific columns

('discount\_percentage', 'rating', 'rating\_count') as numeric features essential for calculating the weighted sum.

1. Selection of Categorical Column: The 'subcategory\_1' column is created by splitting the 'category' column. This selection is crucial for grouping products by subcategories later in the code.
2. Column Selection for Display: When displaying top products, only certain columns

('product\_id', 'subcategory\_1', 'discount\_percentage', 'rating', 'rating\_count', 'WeightedSum') are chosen, indicating their relevance in presenting product information.

**Feature Generation:** Feature generation involves creating new attributes or columns from existing data to extract more information or improve analysis. In this code:

1. Creation of WeightedSum: A new feature column, 'WeightedSum', is generated by combining the numeric features ('rating', 'rating\_count', 'discount\_percentage') with predefined weights. This provides a comprehensive metric for evaluating products based on multiple criteria.
2. Creation of Subcategory Column: The 'subcategory\_1' column is created by splitting the 'category' column and selecting the last part. This generates a new categorical feature, allowing for analysis and visualization by subcategory.
3. Summarization by Subcategory: The code computes the sum of 'WeightedSum' for each subcategory, generating summary information useful for understanding the distribution of product quality and discounts within different subcategories.
4. Selection of Top Products by Subcategory: By grouping products by subcategory and selecting the top-performing product based on weighted average, new insights are generated regarding the best products within each category.

Overall, feature selection and generation in this code contribute to a more comprehensive analysis of the dataset, enabling informed decision-making regarding product recommendations and understanding the distribution of product characteristics across different subcategories.

1. **LINEAR REGRESSION USING RIDGE REGRESSION MODEL AND**

**POLYNOMIAL REGRESSION**

Ridge regression is a linear regression technique used for modeling the relationship between a dependent variable and one or more independent variables. It's a regularized version of linear regression, designed to handle multicollinearity (high correlation among predictor variables) and reduce overfitting by adding a penalty term to the standard linear regression objective function.

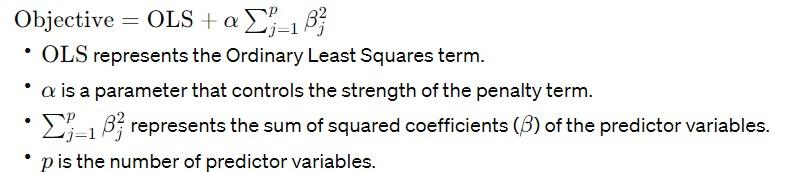
Polynomial Features: Before fitting the Ridge regression model, polynomial features are created from the original features. This involves generating new features by raising the original features to different powers.

The formula for polynomial regression is expressed as:



**Objective Function:**

In linear regression, the objective is to minimize the sum of squared differences between the observed and predicted values. This is known as the Ordinary Least Squares (OLS) objective function. In Ridge regression, a penalty term is added to the OLS objective function. The new objective function becomes:



**Penalizing Large Coefficients:**

The penalty term penalizes large coefficients, effectively shrinking them towards zero.

This helps to reduce the impact of multicollinearity (where predictor variables are highly correlated) by discouraging the model from assigning excessively large weights to correlated features.

**Controlling Model Complexity:**

The parameter α (alpha) controls the strength of the regularization.

Larger values of α result in greater regularization, leading to simpler models with smaller coefficients.

Smaller values of α allow the model to fit the training data more closely, potentially leading to overfitting.

**Bias-Variance Tradeoff:**

Ridge regression introduces a bias into the model in exchange for reducing variance.

This tradeoff can lead to better generalization performance on unseen data, especially when dealing with noisy or high-dimensional datasets.

Solving Ridge Regression:

Ridge regression can be solved using various optimization techniques such as gradient descent or closed-form solutions.

In practice, libraries like scikit-learn provide efficient implementations of Ridge regression that handle the optimization process internally.

1. **Data Preparation:**

The dataset is loaded from a CSV file named "new\_amazon.csv".

Rows with missing values are dropped.

1. **Feature Selection:**

The features for training the model are selected from the dataset: 'actual\_price', 'discount\_percentage', 'rating', and 'rating\_count'.

- The target variable to predict is 'discounted\_price'.

**3. Model Initialization and Training:**

Polynomial features with a degree of 2 are created using PolynomialFeatures.

A Ridge regression model is initialized with an alpha (regularization strength) value of 0.1.

The model is trained using the selected features and target variable.

1. **Model Evaluation:**

The model's performance is evaluated using cross-validation with 5 folds.

The average mean squared error (MSE) is calculated for evaluation.

1. **Prediction:**

The model predicts discounted prices for the first five rows of the dataset.

Predicted and actual discounted prices are printed for comparison.

1. **Evaluation on Larger Test Set:**

A larger subset of the dataset (100 values) is selected for testing.

Features and target variable are extracted from the larger test data.

Discounted prices are predicted for the larger test data.

Mean squared error (MSE) and R-squared values are calculated to evaluate the model's performance on the larger test set.

1. **Model Accuracy:**

The percentage MSE is calculated relative to the range of the target variable.

Model accuracy is computed as 100 minus the percentage MSE.

1. **Visualization:**

Scatter plots, residual plots, and histograms are created to visually analyze the model's performance in predicting discounted prices compared to actual values.

The combination of Ridge regression with polynomial features allows the model to capture non-linear relationships between features and the target variable while controlling for overfitting through regularization.

Here's a breakdown of how the model is used:

1. **Initialization and Training:**

The model is initialized using the make\_pipeline function, which creates a pipeline of transformations.

Within the pipeline, PolynomialFeatures is used to generate polynomial features of degree 2 from the original features.

* 1. Ridge regression model is then applied to the polynomial features. Ridge regression is selected to handle potential multicollinearity and overfitting issues that may arise in the presence of polynomial features.

The model is trained using the fit method on the selected features (X) and the target variable (y).

1. **Evaluation:**

The model's performance is evaluated using cross-validation with 5 folds. This helps to estimate how the model will perform on unseen data.

Cross-validation scores are obtained using the cross\_val\_score function, with the scoring metric set to negative mean squared error (MSE).

The average mean squared error (MSE) across all folds is then calculated as -np.mean().

1. **Prediction:**

After training, the model is used to predict discounted prices for the first five rows of the dataset (X.head()).

The predicted discounted prices are stored in the variable predicted\_discounted\_prices\_test.

1. **Evaluation on Larger Test Set:**
   1. larger subset of the dataset (100 values) is selected for testing purposes.

Features (X\_test\_large) and target variable (y\_test\_large) are extracted from this larger test set.

The model is used to predict discounted prices for the larger test set, resulting in predicted\_discounted\_prices\_large.

The performance of the model is evaluated on this larger test set by calculating the mean squared error (MSE) and R-squared values using mean\_squared\_error and r2\_score functions.

1. **Model Accuracy:**

The percentage mean squared error (MSE) is calculated relative to the range of the target variable to gauge the model's accuracy.

The model accuracy is computed as 100 minus the percentage MSE, indicating how well the model predicts the discounted prices compared to the actual values.

1. **Visualization:**

- Various visualizations, including scatter plots, residual plots, and histograms, are created to provide a visual assessment of how well the model predictions align with the actual discounted prices.

Overall, the Ridge regression model with polynomial features is leveraged to make predictions and evaluate its performance in predicting discounted prices of Amazon products, while the visualizations help in understanding the model's behavior and potential areas for improvement.

The output of the code provides insights into the performance of the ridge regression model with polynomial features for predicting discounted prices:

1. **Average Mean Squared Error:**

The average mean squared error (MSE) calculated using the model is approximately 1017.32.

1. Examples of Actual vs. Predicted Discounted Prices:

For the first five examples, the actual discounted prices and the corresponding predicted discounted prices are printed. It shows how close the predictions are to the actual prices.

1. **Ill-Conditioned Matrix Warnings:**

The warnings indicate that some matrices involved in solving the ridge regression problem are ill-conditioned, which may affect the accuracy of the results. This could be due to multicollinearity or other issues in the dataset.

1. **Mean Squared Error (Large Test Set):**

The MSE calculated on a larger test set consisting of 100 values is approximately 1253.24.

1. **R-squared (Large Test Set):**

The R-squared value, indicating the proportion of the variance in the dependent variable that is predictable from the independent variables, is very high at approximately 0.99998. This suggests that the model fits the data extremely well.

1. **Mean Squared Error (Percentage):**

The MSE expressed as a percentage of the range of the target variable is approximately 3.80%.

1. **Model Accuracy:**

The model accuracy is approximately 96.20%.

